Luiz Fernando Bessa Seibel

Bio-AXS: Uma Arquitetura para Integração de Fontes de Dados e Aplicações de Biologia Molecular

Tese apresentada ao Departamento de Informática da PUC/RJ como parte dos requisitos para obtenção do título de Doutor em Informática: Ciência da Computação.
Orientador: Sérgio Lifschitz

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Rio de Janeiro, 20 de Maio de 2000

Aos meus pais, por ensinarem a importância e o gosto do estudo.

Aos meus tios Geraldo e Cristina, pelo incentivo.

Aos meus filhos, como exemplo.

À minha querida esposa pelo estímulo, dedicação e compreensão.

Agradecimentos

A DEUS pelas graças recebidas e por ter colocado em meu caminho pessoas maravilhosas, meus familiares e amigos.

Ao amigo e professor Sérgio Lifschitz, pela amizade, compreensão e apoio irrestrito. Agradeço também a sua orientação serena na execução deste trabalho, que, por ser multidisciplinar, exigiu frequentemente a busca de outros conhecimentos e experiências para troca de informações.

À minha querida esposa Aline, pelo amor, carinho, atenção e estímulo, sem o que não teria conseguido realizar este trabalho. Agradeço toda a força que recebi, a companhia e a solidariedade em todos os momentos.

Aos meus filhos Pedro, Héline, Bruno e Ricardo, agradeço a torcida e o incentivo, além da compreensão pelos diversos momentos em que deixamos de estar juntos.

Aos meus pais Ada e Romeo e ao meu irmão Carlos Felipe, pelo amor, apoio e pelo incentivo na continuidade de meus estudos.

Aos meus tios Geraldo e Cristina Jofily pelo carinho e incentivo. Deles recebi orientação para toda a vida.

Aos amigos e professores Celso da Cruz Carneiro Ribeiro, Elvira Maria Antunes Uchôa, Antonio Miranda e Carlos José Pereira de Lucena, pela confiança absoluta, amizade, apoio e constante otimismo. Sem as suas inúmeras orientações não teria conseguido chegar ao final desta jornada. Gostaria que soubessem que me considero um privilegiado por ter contado com sua amizade e atenção em todos os momentos importantes, seja na decisão de fazer o doutorado, na escolha da área, na definição da tese, na orientação de estudos e na revisão de artigos e apresentações. Agradeço por todos os ensinamentos, técnicos e não técnicos, que recebi e que iluminaram o meu caminho.

Aos amigos e professores Rubens Nascimento Melo e Asterio Tanaka pela orientação quando da defesa de proposta de tese. O caminho que eu acreditava tortuoso ficou plano depois de seus direcionamentos.

Aos amigos e companheiros José Antonio Fernandes de Macedo, Flávio Freitas, Melissa Lemos, Michelle Santos Sá, Roberto Bastos e Valeria Menezes Bastos, que tive a felicidade de conhecer e compartilhar todos os bons e maus momentos. Não tenho como agradecer e retribuir o carinho, o suporte, a paciência, a ajuda nos comentários e críticas e o apoio nas revisões.

Agradeço especialmente à Melissa Lemos, de quem me tornei grande admirador pelo seu alto grau de responsabilidade, capacidade de trabalho e lealdade. Seu apoio foi fundamental para que fosse atingida a qualidade desejada no trabalho. Tudo o que foi planejado foi feito graças à sua tenacidade e vontade de ajudar. Espero poder retribuir a sua dedicação e amizade.

Outro agradecimento especial deve ser feito ao amigo, sócio e irmão José Antonio Fernandes de Macedo. A sua tranquilidade nos momentos difíceis, sua palavra amiga, seu otimismo, incentivo e apoio em todos os momentos foram fundamentais para que eu pudesse realizar esta tarefa junto a tantas outras que empreendemos. Agradeço também à Cristina Gerbassi pelo mesmo incentivo e força em todos os momentos desta jornada, especialmente na melhoria da qualidade da apresentação da defesa.

Ao pessoal da coordenação e da secretaria do curso de bacharelado em informática e tecnólogo em processamento de dados do DI, em especial à querida amiga Sandra Abreu, pelo apoio, incentivo, e carinho. Aos funcionários do DI, da secretaria, da biblioteca e do Lab-DI, pela atenção e presteza com que sempre atenderam às minhas solicitações.

À Coordenação de Pós-Graduação do DI, em especial ao Professor Bruno Feijó, pela confiança depositada em minha capacidade de trabalho.

À CAPES, à Vice-Reitoria Acadêmica e ao Departamento de Informática, pelo apoio financeiro dado através de bolsas durante o doutorado.

Resumo

Existem inúmeros bancos de dados de biologia molecular, também conhecidos como bancos de dados de genoma, e existe a necessidade de integração destas fontes de dados e das aplicações a elas relacionadas. Esta tese propõe a utilização de um framework orientado a objetos para acesso e manipulação desses dados. Trata-se de uma abordagem de integração que responde integralmente aos requisitos de flexibilidade, reuso e extensibilidade deste domínio de aplicação. Neste trabalho também é proposto um esquema conceitual de informações do dogma central da biologia molecular, definindo uma ontologia que pode ser efetivamente usada nas fontes de dados e confrontada com as propostas existentes. O framework, denominado Bio-AXS, é apresentado em detalhe utilizando os diagramas de classes e de sequências que exploram os aspectos estruturais e dinâmicos da arquitetura. Serão discutidos aspectos de implementação do Bio-AXS e também as funcionalidades do protótipo implementado. Foram feitos diversos estudos de caso de forma a demonstrar a adequação e a aplicabilidade da arquitetura proposta.

Abstract

There are many molecular biology databases, also known as genome databases, that need to be integrated with all related applications and other data sources. This work proposes the use of an object-oriented framework for genome data access and manipulations. It is an innovative approach for an integration tool, which deals well with these application domain requirements, such as flexibility, reusability and extensibility. This works also presents a conceptual schema for the central dogma of molecular biology, defining an ontology which can be used within the data sources and compared to previous approaches. The framework, called Bio-AXS, is presented in details using class and sequence diagrams that explore both the structural and dynamic parts of the architecture. The Bio-AXS implementation issues are discussed, as well as some functionalities of the implemented prototype. Indeed, many case studies were explored in order to show the adequacy and applicability of the proposed architecture.

Índice

Figuras

1 Introdução

Este trabalho tem como objetivo o desenvolvimento de ferramentas computacionais que auxiliem os pesquisadores na área de biologia (biólogos, geneticistas, bioquímicos, farmacêuticos, etc) a compreender melhor as informações disponíveis sobre a pesquisa na área, especificamente em biologia molecular e suas inter-relações.

Em termos mais específicos, trata de conceber, projetar e construir uma ferramenta que possibilita a integração dos dados provenientes da pesquisa em biologia molecular e o uso das ferramentas existentes, a serem aplicadas sobre este conjunto de dados.

Atualmente, essa ferramenta é fundamental para os usuários, que se vêem às voltas com massas de dados dispersas, com informações não relacionadas diretamente, e que contam com um enorme conjunto de aplicações que exigem dados em formatos específicos e não padronizados.

Com o propósito de demonstrar as funcionalidades implementadas, foram realizados diversos estudos de casos práticos / reais. Um exemplo destes estudos trata de verificar a integridade (coerência) das anotações biológicas de uma dada proteína (classificação e função), que estão armazenadas em diferentes fontes de dados. Outros estudos com objetivo de descobrir novos conhecimentos biológicos vem sendo implementados e estão descritos no final do trabalho.

1.1 Relevância do Estudo

Recentes técnicas de laboratório bastante sofisticadas permitem uma coleta de dados de sequências de DNA que é mais rápida do que sua interpretação. Há grandes volumes de dados de sequências ao alcance dos pesquisadores. A principal tarefa hoje é a interpretação de tais dados, ou seja, entender como as informações contidas no DNA se manifestam, gerenciando e controlando os processos vitais.

Seguem-se algumas questões básicas em termos de investigação biológica:

- Como achar fragmentos de DNA que tem função biológica ? Que função é essa ?
- Se o fragmento tem relação com uma nova proteína, qual a sua função ? E a sua estrutura ?
- Como regular a expressão e a atividade destas proteínas ?

As suas respostas irão propiciar novos conhecimentos, ajudar na busca de soluções de disfunções bioquímicas, na compreensão de relações moleculares e mesmo na classificação das espécies.

A resposta a essas questões tem início hoje em uma recente área do conhecimento denominada Biologia Computacional, que tem como fim a pesquisa de modelos, técnicas e ferramentas que auxiliem o entendimento dos fenômenos biológicos. O termo Bioinformática é também usado porém, algumas vezes, com um sentido menor, ou seja, refere-se apenas à aplicação de técnicas computacionais acessíveis e o uso das ferramentas disponíveis.

A base da Biologia Computacional reside no reconhecimento de padrões, em métodos estatísticos e no desenvolvimento de modelos que representem as relações biológicas. O reconhecimento de padrões é uma atividade básica e é feita sobre cadeias de caracteres porque a representação da estrutura primária de uma molécula de DNA é feita desta forma. Cada caracter indica um nucleotídeo da estrutura (‘A’ para adenina, ‘T’ para timina no DNA ou ‘U’ para uracila no RNA, ‘C’ para citosina e ‘G’ para guanina). Se for conhecida a estrutura primária de um fragmento, a sequência de DNA / RNA e a sua função, este conhecimento pode ser, a princípio, estendido para outros fragmentos semelhantes do organismo em estudo e também de outros organismos.

A mesma representação, na forma de cadeia de caracteres, é usada para proteínas, onde os nucleotídeos são agrupados de três em três, para especificar os aminoácidos.

Essa hipótese acerca do comportamento de uma certa sequência de DNA / RNA / proteína, que foi obtido de uma sequência semelhante, é, de fato, um conhecimento potencialmente significativo, uma pista, que deve ser confirmada pelos biólogos experimentalmente, em bancada.

A comparação de sequências é, possivelmente, a ferramenta computacional mais útil para os biólogos moleculares. As famílias de algoritmos de comparação de sequências são conhecidas por Fast [Pea91, Pea94, Pea90, PL88] (fornece comparações mais precisas) e Blast [AGM+90, AMS+97, GS93, NCBI02b, GDB02a] (executa mais rapidamente). O estudo da informação biológica de regiões de DNA é tratado por outros algoritmos aliados a métodos estatísticos [RE01, TDZ01, ORP+99, Ben99, KTN94]. Por exemplo, a identificação de regiões que codificam para proteínas (um grande número de pesquisadores utiliza o termo "gene" para estas regiões) exige a execução de algoritmos que reconhecem padrões de início e fim destas regiões, analisam a ocorrência dos nucleotídeos citosina e guanina nos padrões identificados, e identificam outros sinais aos quais são dados tratamentos estatísticos, no sentido de apontar novos conhecimentos, que são, mais uma vez, potencialmente significativos e que também devem ser confirmados em bancada. Estes algoritmos são denominados "localizadores de genes" (gene finder [SS97]).

Além da aplicação destes algoritmos, técnicas de modelagem bastante conhecidas na área de computação são aplicadas à biologia no sentido de representar interações e relacionamentos ali verificados. Um exemplo pode ser aplicado às transformações moleculares via diagramas de estado, na modelagem conceitual das informações biológicas via diagramas de entidades e relacionamentos ou via linguagem de modelagem tipo UML *(Unified Model Language*) [BRJ99], e mesmo em outras abstrações onde modelos matemáticos têm sido utilizados para explicar o comportamento de sistemas biológicos.

O gerenciamento das informações biológicas tem sido feito via arquivamento em meio magnético, em formato textual, ou através de Sistemas Gerenciadores de Bancos de Dados (SGBD's). São ainda utilizados aplicativos específicos para recuperação e

análise das informações. No entanto, algumas tarefas que são características dos SGBD's não são apropriadas, sendo necessário o uso de aplicativos específicos, como por exemplo, para se evitar a duplicidade de informações nas fontes de dados, é necessário utilizar o algoritmo Blast ou Fast.. Mesmo assim, é crescente a adoção de SGBD's para o gerenciamento de informações biológicas devido às facilidades de consulta e atualização das informações, de facilidades de gerenciamento e manutenção [Doo90].

Cada pesquisa na área contém dados que representam um pequeno domínio do conhecimento. O conhecimento completo está na união das fontes de dados existentes, sejam elas armazenadas em arquivos texto, planilhas ou mesmo SGBD's. Um desafio a ser vencido consiste na construção de ferramentas de integração das informações que residem nestas fontes de dados, que pressupõe um modelo integrador das informações biológicas.

Outro desafio consiste no uso dos aplicativos disponíveis, a serem executados sobre os dados da base completa pois atualmente cada repositório de informações biológicas admite o uso de um reduzido número de aplicações.

A importância do presente trabalho bem como a sua aplicabilidade na área da Biologia Molecular estão detalhadas e evidenciadas ao longo do texto, no entanto, para os biólogos que tiveram contato com a arquitetura proposta, ela se constitui em uma poderosa ferramenta para a descoberta de novos conhecimentos biológicos e vem sendo utilizada por eles com este objetivo.

1.2 Justificativas para o Estudo

A área de Bioinformática tem estado em evidência nos últimos anos em grande parte devido à importância do Projeto Genoma Humano [DOE02a, DOE02b, DOE02c, HGD02], que tem como objetivo mapear e sequenciar o DNA do homem, ou seja, conhecer com detalhe o DNA humano e sua organização e expressão.

As ferramentas utilizadas em Bioinformática são bastante amplas e incluem: sistemas de gerenciamento de informações de laboratórios, fontes de dados de genoma (sejam elas formadas por dados gerados por um único laboratório ou por redes de pesquisadores e centros de pesquisa), formas de disseminação de dados (via *e-mail*, FTP, *web*), sistemas de conversão e integração de dados entre diferentes fontes, além de sistemas de análise de dados e suporte ao trabalho dos biólogos, geneticistas e bioquímicos, que manipulam e constróem modelos e hipóteses acerca das relações existentes no DNA e combinações entre o DNA e outras moléculas.

Existem muitos desafios nesta área que não foram ainda resolvidos de forma satisfatória e um dos mais importantes é, sem dúvida, o de fornecer aos cientistas o acesso a todos os dados relevantes para a sua pesquisa e o de possibilitar que sejam executadas as aplicações / sistemas necessários sobre os dados. Somente com o conhecimento do mundo real da pesquisa, é possível se compreender a magnitude deste desafio.

Atualmente, as principais ferramentas utilizadas pelos pesquisadores em biologia molecular são associadas a dezenas de arquivos e bancos de dados públicos contendo informações relativas a sub-domínios específicos do conhecimento, tais como: sequências de nucleotídeos (GenBank [NCBI02a, GenBank02], AceDB [AceDB02], Embl [EMBL02, SBB02, BBC+00], DDBJ[OTG+98], GDB [GDB02b, LCP98], GSDB [NCGR02, HSB+98, HCF+00]), sequências de proteínas (Swiss-Prot [ExPASy02b, EBI02, BA00], PIR [PIR02, WHA+02, BGH+00], TREMBL[ExPASy02b, EBI02, BA00]), estruturas de proteínas (PDB [PDB02a, PDB02b, WFJ+02, BWF+00], DSSP [DSSP02a, DSSP02b, KS83] ou outros dados biológicos relativos a um ou a vários organismos (humano, mamíferos, bactérias, etc.) [TIGR02, Sanger02]. Tal fato tem provocado a proliferação das fontes de dados aplicadas à biologia molecular [AG97] e [SLL00].

Existem ainda centenas de fontes de dados menores que são altamente especializadas, isto é, armazenam dados de um tipo de organismo ou célula [AceDB02]. Outras se concentram em uma específica função biológica [ExPASy02a]. Algumas tentam

registrar as mutações e diferenças encontradas em um dado gene ou grupo de genes [HMD02]. Em muitos casos estas informações são produzidas por um único laboratório e suas informações são únicas, isto é, não estão replicadas nos repositórios públicos.

As fontes de dados relevantes para os pesquisadores contêm informações biológicas (homologias, estruturas, mapas, sítios, similaridades entre espécies, etc.) e outras informações a elas relacionadas tais como, anotações relevantes, pesquisadores, centros de pesquisa, artigos científicos, entre outras [NCBI02a].

Em sua maioria estas fontes de dados diferem na forma de armazenamento de dados e nas informações relevantes à pesquisa. Tais fontes de dados estão associadas a aplicativos que também diferem nos serviços oferecidos à comunidade científica, como por exemplo, os serviços de visualização dos dados (de cromossomos, de sítios de informações biológicas relevantes, de sequências, etc.), de busca, de alinhamentos, de comparação de sequências, entre outros.

Cada fonte de dados é implementada por um grupo de pesquisa independente, sendo que cada grupo procura representar as informações em um modelo de dados que considere o mais adequado. Ao longo do tempo, surgiram implementações de sistemas que armazenam informações biológicas em arquivos texto (ex. GenBank), em bancos de dados relacionais (ex. Swiss-Prot) e em bancos de dados orientados a objetos (ex. AceDB). No entanto, como a pesquisa na área está em constante evolução, houve a necessidade de alteração dos esquemas já implantados, sugerindo a adoção de modelos de dados mais flexíveis. Surgiram então implementações que adotam o modelo objeto-relacional (ex. AatDB) e mesmo o modelo semi-estruturado. Estas mudanças têm ocorrido porque não está claro que modelo de dados é o mais adequado para representar as informações biológicas, sendo que um dos aspectos mais importantes nesta representação refere-se à flexibilidade do modelo adotado em termos de facilidade de alteração do esquema. Isto ocorre porque novas informações biológicas surgem a cada momento e é fundamental que elas possam ser representadas nas fontes de dados existentes.

As informações armazenadas em diferentes fontes de dados podem ainda ter esquemas diferentes e atributos correspondentes representados com formatos diferentes, apesar de usarem o mesmo modelo, por exemplo relacional. A semântica dos termos utilizados para representação das informações armazenadas também não é a mesma. Assim, um dado termo que é utilizado em fontes de dados distintas pode não ter o mesmo significado. Isto ocorre porque, a cada dia, novas descobertas acontecem no ramo da biologia molecular, que podem alterar conceitos anteriormente estabelecidos. Por exemplo, o termo gene é utilizado por alguns pesquisadores como a região que codifica para proteína. Outros entendem que a definição completa inclui regiões promotoras.

Em resumo, a meta da pesquisa na área de biologia computacional é a de permitir aos usuários a interação, com uma série de fontes de dados, como se estivessem interagindo com apenas uma. As fontes de dados envolvidas na interação são aquelas que contém informações relevantes para uma dada pesquisa. Estas fontes de dados estão distribuídas, são heterogêneas e foram implementadas usando modelos de dados distintos. A interação acima descrita significa acesso via Web, formulação de consultas a objetos biológicos específicos, formulação de consultas complexas, execução de algoritmos específicos e mesmo atualizações envolvendo um ou vários objetos e relações biológicas. O presente trabalho descreve uma arquitetura flexível e extensível (*framework*) [SL01, LSU01] que permite aos usuários capturarem os esquemas das fontes de dados, que estão disponíveis na Web, para instanciar uma base de dados adequada a uma dada pesquisa. O esquema desta base de dados poderá evoluir através de manipulações efetuadas pelos próprios usuários, utilizando-se uma interface adequada. Para tanto, a arquitetura necessita integrar dados presentes nas fontes pré-existentes, sendo que o processo de integração exige uma ontologia para fornecer descrições semânticas. A arquitetura fornece ainda para os usuários uma interface entre a base de dados instanciada e os diversos aplicativos da biologia, cujos códigos-fonte / executáveis estão disponíveis e são bastante utilizados pelos pesquisadores da área.

1.3 Principais Questões

A área de Biologia Molecular está se tornando bastante dependente da área de *software*. Os usuários necessitam de programas que executam análises sobre os dados, de ferramentas de modelagem, de sistemas de gerência de bancos de dados e de ferramentas gráficas de visualização de informações na sua rotina de trabalho.

Da mesma forma como ocorre em outras áreas de aplicações científicas, os usuários se ressentem de programas que forneçam o suporte adequado à pesquisa.

Grupos de pesquisa e laboratórios têm trabalhado de forma isolada para produzir ferramentas específicas para as suas pesquisas, que resolvem problemas pontuais ou são concebidas de forma bastante limitada.

É bastante comum que o desenvolvimento destas ferramentas seja feito pelos próprios cientistas da área, visto que é muitas vezes difícil explicar os problemas a serem resolvidos, e assim, procuram a via mais rápida. Além disso, a contratação de programadores profissionais tem custo elevado pois estes são bem remunerados pelo mercado de trabalho. Consequentemente, os *softwares* desenvolvidos tangenciam conceitos importantes utilizados em engenharia de software, tornando-se pontuais, pouco flexíveis e pouco eficazes.

Existem ainda complicadores adicionais como por exemplo, a evolução extremamente rápida da área em termos científicos, tecnológicos e de negócios, além da enorme especialização que é necessária para o desenvolvimento de ferramentas adequadas. Todos estes fatores têm contribuído para uma produção de *software* aquém da desejada em termos de engenharia de produto e de funcionalidade.

Esse desenvolvimento desorganizado e descentralizado é, no entanto, bastante natural e mesmo necessário em um campo científico vibrante e dinâmico. Por outro lado, a pesquisa descentralizada e autônoma tem resultado em ferramentas e fontes de dados incompatíveis.

O preâmbulo mostra que as principais questões que devem ser resolvidas de forma prioritária referem-se a:

- Integração dos esquemas das fontes de dados

  Os esquemas das fontes de dados devem ser capturados, compreendidos e documentados de forma a permitir relacioná-los com objetivo de construir um esquema global. A documentação deve ser incorporada ao esquema global e deve conter uma completa descrição dos objetos, relacionamentos, atributos e métodos, de forma a se ter a total compreensão dos objetos biológicos envolvidos.

  Esta tarefa deve ser realizada manualmente por pesquisadores experientes pois os objetos contidos nas diferentes fontes de dados podem ter todos os problemas de integração semântica descritos em [SK92, SKS97, EN89]. Por exemplo, o próprio atributo “gene” pode ter semântica diversificada. A definição precisa e completa dos objetos biológicos, se constitui, para muitos pesquisadores, em uma ontologia de fato, que é muito útil para a área.

  Importantes pesquisadores em ciência da computação têm desenvolvido ferramentas de integração bastante interessantes mas que realizam mapeamentos para um esquema global incompleto ou então exigem que os usuários conheçam os esquemas das fontes de dados para a formulação das consultas (Entrez [Entrez02], SRS [SRS02a], LinkDB [LinkDB02, FGM+97], CPL/Kleisli [CPL02,DCB02], K2 e GUS [DCB01], Karp [Kar95], OPM [OPM02a, MR95], IGD [IGD02a, Rit94], TAMBIS [TAMBIS02a, SBB00], GIMS [Gims02a, CPW+01]).

- Representação da evolução dos esquemas nas fontes de dados

  Os esquemas locais e o esquema global devem evoluir. Esta evolução deve estar representada em novas versões de cada esquema.

- Construção de um esquema específico para uma dada pesquisa

Uma vez que existe um esquema global, esquemas específicos podem ser construídos a partir deste, de forma a atender necessidades específicas de uma comunidade de pesquisadores.

- Instanciação de um esquema específico

  Com base na ontologia construída a partir da captura dos esquemas locais, um *data warehouse* pode ser instanciado de forma a atender pesquisas específicas, através de um mediador de acesso às fontes de dados locais [Wie92, KBT+01].

- Integração das aplicações

  Os aplicativos disponíveis para a pesquisa em Biologia Molecular devem executar sobre qualquer fonte de dados, independente do formato dos dados. Novos aplicativos podem ser construídos a partir de formatos padrão de instâncias de objetos (por exemplo, utilizando modelos de dados semi-estruturados), caso contrário, deverão existir *drivers* de conversão para os formatos exigidos pelos aplicativos. O número de *drivers* pode ser reduzido se o ponto de partida, a fonte dos dados, for previamente padronizada (também utilizando dados semi-estruturados).

- Integridade das fontes de dados e qualidade das informações armazenadas

  Deverão ser construídas aplicações que garantam a integridade e a qualidade dos dados. Inúmeros exemplos podem ser apontados com relação a estas questões, como os que são apresentados a seguir.

  Um exemplo importante na falta de integridade reside na questão das anotações biológicas [WB98, Kar98, MLD95]. Anotações são registros de funções biológicas de trechos de DNA, RNA e de proteínas. Muitas anotações feitas nas fontes de dados estão incorretas por diversos motivos. Por exemplo, por erro de interpretação, falha no sequenciamento, registro incorreto na fonte de dados, entre outros. Por algum tempo acreditou-se que seria uma boa prática a exportação do conhecimento anotado de uma sequência para outras semelhantes a ela (semelhança esta determinada pelo Blast). Como podem ter ocorrido erros de

anotação na sequência de origem, estes erros se disseminaram. Atualmente existem bases curadas, onde somente anotações comprovadas via experimentos são registradas [ExPASy02a, EBI02, BA00].

Com relação à qualidade das informações, um exemplo importante trata da qualidade dos registros de sequências de DNA, RNA ou proteína. Os responsáveis pelas informações cadastradas nas fontes de dados públicas de sequências são os próprios pesquisadores, centros de pesquisa ou laboratórios que submeteram os registros para o seu cadastramento. Muitas vezes não houve a devida preocupação com a qualidade da sequência submetida. Assim, as sequências podem conter erros devido à ocorrência de falhas no sequenciamento, à exclusão de sequências de vetores, etc.

- Descobertas da fase pós-genoma

  A meta maior da biologia computacional na fase pós-sequenciamento deve se concentrar na aplicação de técnicas de *data-mining* para a descoberta de novos conhecimentos biológicos e na realização de experimentos *in silico* para verificação / comprovação de suposições formuladas pelos pesquisadores da área.

1.4 Principais Contribuições

A tese apresenta uma arquitetura que engloba soluções para as questões anteriormente mencionadas. A arquitetura, cuja especificação, projeto e implementação constam deste trabalho, propõe uma forma de integração de qualquer fonte de dados previamente existente e dos aplicativos disponíveis. A arquitetura permite ainda que sejam instanciadas novas fontes de dados e que sejam criados novos aplicativos, que são a ela incorporados.

A integração é feita via captura de esquemas para a construção e/ou alteração incremental (à medida que um novo esquema é capturado) de um esquema global. A captura dos esquemas é manual e deve ser feita por um administrador da arquitetura que, obrigatoriamente, deve conhecer os conceitos biológicos pois são necessários

conhecimentos semânticos acerca dos objetos que estão sendo incorporados ao esquema global, além de definições precisas da ontologia efetivamente utilizada pela fonte de dados local.

Essa ontologia (que consta do esquema global) é utilizada pelo mediador de acesso aos dados, com objetivo de instanciar uma base de dados específica para uma dada pesquisa. Essa abordagem tem vantagens em termos de desempenho pois utiliza um *data warehouse* integrador, ao invés de realizar consultas ou executar algoritmos sobre as fontes de dados distribuídas.

A evolução de um esquema local é identificada automaticamente via agentes de *software* que acompanham permanentemente a sua evolução. Ao ser identificada a alteração de um esquema, esta é informada ao administrador, que irá proceder a uma nova captura do esquema local. Agentes de *software* são também utilizados para atualização das bases de dados presentes na arquitetura, a partir da identificação de alterações nas bases locais.

De posse da ferramenta, estudos estão sendo efetuados por pesquisadores da área de biologia molecular com objetivo de resolver os problemas de integridade e de qualidade dos dados e também com intuito de descobrir novos conhecimentos. Um exemplo destes estudos trata da verificação de inconsistências das anotações biológicas que possam existir entre as fontes de dados de proteínas Pir e Swiss-Prot.

De forma resumida, as principais contribuições da tese referem-se a:

- proposta de arquitetura de integração baseada no uso de mediadores, projetada e implementada utilizando conceitos de engenharia de software, no caso, *frameworks* orientados a objetos, até então não utilizados nos projetos de integração descritos na literatura;

- proposta de esquema global da biologia contendo a descrição dos objetos biológicos e seus relacionamentos, além de outras informações necessárias à integração semântica;

- definição de uma ontologia efetivamente utilizada nas fontes de dados, que pode ser confrontada com as propostas para a área;

- construção de uma ferramenta útil para a pesquisa na área de biologia molecular, que é eficaz na solução dos problemas de integração das fontes de dados e dos aplicativos, que vem sendo utilizada para verificar hipóteses e contribuir na busca de novos conhecimentos para a área;

- comparação entre as arquiteturas de integração existentes;

- uso da tecnologia de agentes de *software* para relatar a alteração do esquema de uma certa fonte de dados e também para atualização das bases instanciadas.

1.5 Organização do Trabalho

O presente trabalho está dividido em capítulos, seções, ítens e sub-itens, sendo que os primeiros estão relacionados a seguir, de forma sucinta.

- Capítulo 1: Introdução

  Apresenta os objetivos e a relevância do estudo, os fatores que justificam a sua proposição e execução, as questões envolvidas e as contribuições geradas, além da organização do trabalho.

- Capítulo 2: Motivação

  Apresenta a motivação para a realização do estudo. Inicialmente é tratado o contexto das fontes de dados e das aplicações biológicas, onde são apresentadas as mais importantes fontes de dados de biologia molecular, suas características e as

informações que cada uma armazena e disponibiliza, além da tecnologia utilizada no armazenamento e na disponibilização das informações. São mencionadas também as mais importantes ferramentas utilizadas pelos biólogos. Em seguida são relacionadas as funcionalidades a serem atendidas para a construção de uma ferramenta de integração que atenda às necessidades da pesquisa. Posteriormente é apresentada a abordagem de integração utilizada neste trabalho, bem como as abordagens de integração existentes no contexto da biologia. Finalmente são discutidas as abordagens em termos de vantagens / desvantagens e limitações.

- Capítulo 3: A Proposta de Arquitetura

  Descreve a especificação da arquitetura proposta para a integração de dados e aplicações de biologia molecular. A arquitetura é baseada em um *framework* de aplicação, que é descrito através da linguagem UML. São apresentadas as funcionalidades do framework, o diagrama de classes (modelo estático) e os diagramas de sequência (modelos dinâmicos). Sempre que possível, são identificados na especificação padrões de projeto (*design patterns)*.

- Capítulo 4: Comparação entre as Arquiteturas de Integração

  Este capítulo apresenta os principais trabalhos relacionados com a tese. São comparadas as arquiteturas de integração existentes e descritas as vantagens e desvantagens de cada uma com relação à arquitetura proposta. Ao final do capítulo é apresentado um quadro comparativo entre as arquiteturas, construído segundo critérios pré-definidos.

- Capítulo 5: Modelos de Informações Biológicas

  No capítulo são discutidos aspectos referentes ao modelo conceitual da biologia molecular, as suas classes e as relações entre elas. Inicialmente é proposto um modelo conceitual de informações biológicas. Em seguida, é discutido que modelo lógico é adequado para melhor representar os dados de biologia molecular e é feita uma proposta, que é utilizada na arquitetura. Finalizando, são apresentadas e discutidas as diversas ontologias existentes para a área e novamente é apresentada a proposta da arquitetura.

- Capítulo 6: A Implementação do *Framework*

  O capítulo 6 descreve a implementação do *framework*. Inicialmente é apresentada a sistemática de uso do *framework*, ou seja, como implementar e instanciar os pontos de flexibilização da arquitetura. Em seguida, são apresentadas as premissas para o seu desenvolvimento e a forma de instanciação de cada módulo. Para cada um, são mencionados os componentes de *software* que foram re-utilizados e os desenvolvidos. Finalmente são apresentados casos de uso que demonstram a sua aplicabilidade.

- Capítulo 7: Conclusão

  Apresenta as conclusões do trabalho. Critica a implementação em termos de melhorias e funcionalidades não tratadas. Relaciona os trabalhos futuros e os problemas em aberto, que propiciam a continuidade do estudo e da sua implementação.

- Anexo: Conceitos Básicos de Biologia

  Apresenta os conceitos biológicos relevantes para o entendimento do texto.

1.6 Comentários Finais

Este capítulo apresentou uma introdução ao estudo. Inicialmente foram definidos os objetivos do trabalho e foi apresentada uma visão geral do problema a ser tratado, de forma a demonstrar a relevância do tema. Em seguida foram apresentados os aspectos que serviram como justificativa e motivação do estudo, foram relacionados os problemas com que se deparam os biólogos na sua rotina de trabalho, foram apresentados os principais problemas da área e foram relacionadas as principais contribuições do estudo, além de sua organização.

No capítulo seguinte é apresentada a motivação para a realização do estudo que, de certa forma, foram a sua inspiração.

2 Motivação

Este capítulo apresenta a motivação para realização do estudo. Inicialmente são apresentadas as características das fontes de dados de biologia molecular, as informações que cada uma armazena e disponibiliza, além da tecnologia utilizada no armazenamento e na disponibilização das informações.

Em seguida são apresentadas as ferramentas utilizadas pelos biólogos, classificadas em termos de funcionalidade e ambiente de execução, identificando-se o papel de cada uma em um ambiente de produção de um laboratório.

Posteriormente são relacionadas as funcionalidades a serem atendidas para a construção de uma ferramenta de integração que atenda às necessidades da pesquisa na área de biologia molecular.

A seguir é apresentada a abordagem de integração utilizada neste trabalho, que é baseada na tecnologia de mediadores.

São apresentadas também as abordagens de integração existentes no contexto da biologia molecular e, finalmente, são discutidas as abordagens em termos de vantagens / desvantagens e limitações.

2.1 As Fontes de Dados de Biologia Molecular

Esta seção apresenta as fontes de dados e os aplicativos utilizados na pesquisa em biologia molecular, que são ferramentas indispensáveis para os pesquisadores na área.

2.1.1 Descrição Geral

As informações básicas que estão armazenadas nas fontes de dados da pesquisa em biologia molecular referem-se a informações biológicas, publicações científicas da área e pesquisadores / centros de pesquisa em biologia molecular.

A representação das informações referentes a pesquisadores, centros de pesquisa, publicações e seus relacionamentos, já é bastante conhecida e estudada. O fato novo reside nos dados biológicos, que requerem um estudo mais detalhado em termos de estruturas de dados e formas de representação. Segue-se uma relação de exemplos dos atributos biológicos encontrados nas mais importantes fontes de dados da área.

Atributos Biológicos:

- descrição da sequência;
- identificador principal da sequência;
- identificadores secundários da sequência. Muda sempre que uma dada sequência é alterada. Assim, pelo identificador principal pode-se acessar o histórico de evolução das informações sobre a sequência;
- palavras-chave associadas aos genes localizados na sequência ou a outras informações;
- nome do organismo onde a sequência foi encontrada;
- nome científico formal do organismo (gênero e espécie);
- informações relativas à taxonomia do organismo;
- referências cruzadas para outras sequências e comparações com outras coleções;
- características encontradas em determinadas regiões da sequência;
- contador do número de ocorrências de cada nucleotídeo na sequência;
- localização da primeira base de nucleotídeos da sequência, dentro do genoma. Isto inclui sua localização dentro do mapa genético;
- a sequência de nucleotídeos;
- a sequência de aminoácidos;
- estrutura da proteína, entre outras.

As fontes de dados de biologia molecular armazenam sequências de nucleotídeos ou de proteínas (biossequências), além de anotações sobre estas sequências e são

fundamentais para a pesquisa em biologia molecular. Estas fontes de dados podem ser:

- arquivos texto com alguma formatação padrão, como por exemplo no formato ASN.1 [ASN02, Dub00] (definido inicialmente com objetivo de padronizar a comunicação de dados das camadas mais altas do modelo OSI e hoje utilizado como forma de armazenamento de dados de genoma pelo *National Center for Biotechnology Information* [NCBI02a]) e no formato .ACE [AceDB02] (inicialmente utilizado no Sanger Center [Sanger02] mas hoje bastante difundido e utilizado em diversos laboratórios de sequenciamento de genoma);

- bancos de dados implementados via Sistemas Gerenciadores de Bancos de Dados (SGBD´s), onde cada laboratório pode utilizar um banco de dados distinto e estes podem adotar modelos diferentes, como por exemplo, os modelos de dados relacional, orientado a objeto ou relacional-objeto, que são bastante utilizados. Cada banco conta com interfaces de consulta próprias, bem definidas. Os bancos de dados utilizados estão disponíveis no mercado, ou são utilizados em meios acadêmicos ou mesmo construídos para uso específico, por algum centro de pesquisa, para melhor suportar as aplicações [AceDB02];

- arquivos que tem formato apropriado para a execução de determinadas aplicações, por exemplo os dados armazenados no formato FASTA, específico para a execução do algoritmo de busca de sequências por similaridade, denominado BLAST [AGM+90, AMS+97, GS93, NCBI02b, GDB02a].

Diversos estudos publicados na área utilizam o termo “banco de dados” como genérico, para todas as fontes de dados explicitadas acima. O aspecto mais importante destas fontes de dados é o de permitir gerar novos conhecimentos sobre o genoma através da busca de sequências no banco, por similaridade. Assim, se forem bem conhecidas a estrutura e as proteínas sintetizadas por algum gene já documentado, de algum organismo / espécie, esse conhecimento pode ser estendido para outros organismos / espécies que tenham gene semelhante. Dada a importância da

comparação de sequências na biologia molecular, é natural que se procure organizar as sequências de forma a facilitar tal comparação.

A maior fonte de dados genérica de genoma (Genbank [NCBI02a, GenBank02, BKL02, BML+00]) armazenava, em dezembro de 2001, quase 15 bilhões de bases de nucleotídeos (exatas 14.396.883.064 bases), em quatorze milhões de sequências (exatas 13.602.262 sequências), como mostra a Figura 1. Estes números têm crescido de forma significativa. Além do crescimento do volume dos dados armazenados, existem inúmeros projetos que visam levantar o genoma de diversos organismos, o que tem provocado a proliferação destas fontes de dados.

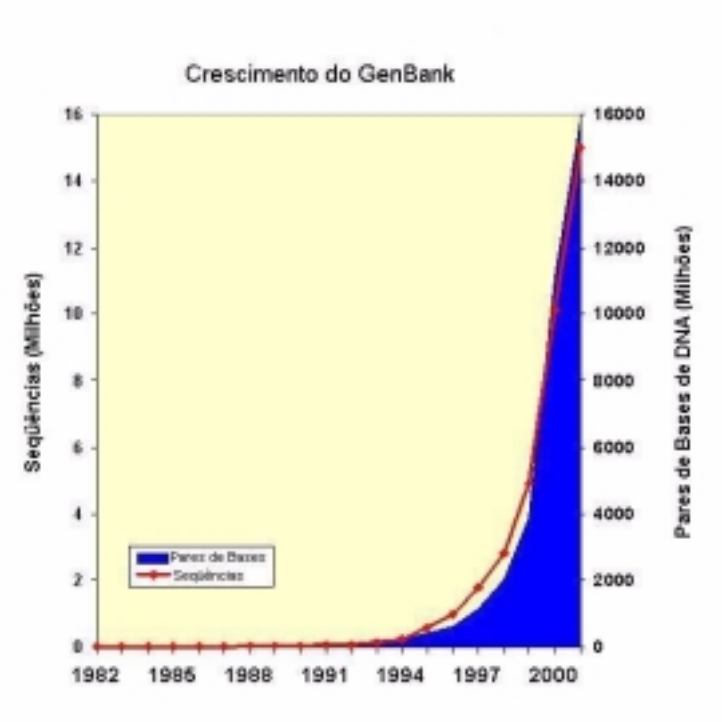

Figura 1 - Crescimento do Genbank

Cabe observar que existem diferenças entre uma fonte de dados genérica para armazenar sequências de nucleotídeos ou proteínas e uma específica para projetos genoma. A diferença básica é que, na primeira, são cadastrados dados referentes às sequências de diversos organismos, além de informações sobre os autores, *links* para publicações, etc. No caso dos projetos genoma, as sequências são de um único organismo (*C. elegans* [AceDB02], genoma humano [TIGR02, Sanger02]). Além

disso, outras informações são cadastradas, tais como: imagens dos filtros, numeração dos tubos contendo os clones, estantes, entre outras. Neste trabalho não faremos tal distinção pois nos interessa o estudo de aspectos das fontes de dados que são relevantes para a biologia molecular, não importando se pertencem a mais de uma espécie ou não. O fato realmente importante é que são armazenadas informações sobre o genoma.

O primeiro arquivo público de sequências de DNA foi o GenBank que é o principal repositório de dados de biossequências. O GenBank é parte do *International Nucleotide Sequence Data Bank* [INSD02], um consórcio internacional de centros de pesquisas que tem com objetivo o cadastro das biossequências de diversos organismos, e que é patrocinado por entidades governamentais de diversos países. O GenBank é a instância do consórcio nos EUA, na Europa é o European Molecular Biology Library (EMBL) , e no Japão o DNA Data Bank of Japan (DDBJ). Existe ainda uma outra instância nos EUA denominada Genome Sequence Database (GSDB), que embora não faça parte do consórcio, possui os mesmos dados. Essas quatro bases de dados trocam informações entre si de modo que a submissão de uma sequência (envio da sequência para armazenamento, que só ocorre se a sequência não existir na base) a uma delas implica, de fato, na submissão da sequência a todas. Tais bases de dados tem facilidades de troca de informações através da importação / exportação de dados, em um formato determinado, porém contam com modelos de dados distintos (arquivos texto estruturados no GenBank, relacional no EMBL e orientado a objetos no DDBJ).

O mais importante banco de dados de sequências de proteínas é o Swiss-Prot. As informações armazenadas neste banco são ainda complementadas por outros que armazenam informações específicas sobre proteínas, como por exemplo o PDB, que armazena estruturas terciárias (representação em três dimensões).

Existem ainda inúmeras outras fontes de dados que armazenam informações sobre genoma. Algumas são fontes especializadas, que armazenam dados de um tipo de organismo [AceDB02] ou célula ou partes da célula [MITOMAP02, KLB+02], outras

se concentram em uma específica função biológica [ExPASy02a], outras tentam registrar as mutações e diferenças encontradas em um dado gene ou grupo de genes [HMD02] e outras diferem na forma de armazenamento de dados de sequências e nas informações relevantes à pesquisa, bem como nos serviços oferecidos de busca e comparação de sequências [LCP98, GJ02, BO98].

Diversas fontes de dados de genoma utilizam *flat files* para armazenamento das informações. O GenBank, por exemplo, armazena ainda hoje informações em arquivos do tipo texto e a interface para a submissão ou busca de uma sequência tem sido mantida desta forma. Os mais recentes projetos de pesquisa de genoma armazenam as informações em bancos de dados relacionais ou orientados a objeto. Esta diversidade ocorre porque não é claro, para os pesquisadores em bioinformática, que modelo de dados é o mais adequado para representar relações entre o genoma. O aspecto mais importante nesta representação refere-se à flexibilidade do modelo adotado (facilidade de alteração do esquema) [NK99] pois novas informações biológicas surgem a cada momento, determinando a inclusão de novos atributos e de alterações dos esquemas implementados. É fundamental que as bases de dados acompanhem a evolução das pesquisas em termos de relacionamentos entre fenômenos biológicos e de representação de novas informações. Este aspecto vem sendo objeto de pesquisa na área [NK99].

Como existem inúmeras fontes de dados que contém informações sobre o genoma dos seres vivos e como estas armazenam informações complementares, cada uma com seu próprio esquema, existe também um grande esforço que já foi e que vem sendo feito no sentido de integrar estas bases de dados (Entrez [Entrez02], SRS [SRS02a], LinkDB [LinkDB02, FGM+97], CPL/Kleisli [CPL02,DCB02], K2 e GUS [DCB01], Karp [Kar95], OPM [OPM02a, MR95], IGD [IGD02a, Rit94], TAMBIS [TAMBIS02a, SBB00], GIMS [Gims02a, CPW+01]).

2.1.2 Classificação das Informações Disponíveis nas Fontes de Dados

As fontes de dados aplicadas à biologia molecular podem se classificadas de acordo com as informações biológicas que armazenam [AG97], que são, principalmente, de:

- sequências (de nucleotídeos ou de proteínas) e anotações sobre as mesmas;
- proteínas e informações sobre as respectivas funcões;
- estruturas de moléculas de proteínas (secundárias, representadas em um plano, ou terciárias, representadas em três dimensões);
- taxonomia (classificações dos organismos vivos);
- bibliografia na área de biologia molecular (artigos, jornais, periódicos, etc).

A seguir serão detalhadas cada uma destas informações.

2.1.2.1 Sequências de nucleotídeos

As fontes de sequências de nucleotídeos armazenam, além da própria sequência, anotações contendo dados de características biológicas relevantes sobre elas, que são: organismo a que pertence, regiões das sequências que codificam potencialmente proteínas, função, fenótipo (características aparentes), e *links* para outras fontes de dados contendo informações biológicas sobre a sequência.

Embora exista um controle sobre erros comuns detectados na submissão de sequências, a responsabilidade pela qualidade da informação é do pesquisador que submeteu a sequência. Os laboratórios que submetem sequências têm diferentes critérios sobre a qualidade da sequência que está sendo submetida. Além disso, apenas alguns têm a preocupação de retirar da sequência os dados de clones vindos do sequenciamento. Outros não agem desta forma, poluindo a sequência com informações desnecessárias. Assim, redundâncias e inconsistências são inevitáveis.

Os bancos de dados de nucleotídeos apresentam, portanto, diversos erros. As sequências existentes nestas fontes de dados estão incompletas, contaminadas e com erros oriundos do próprio sequenciamento [LKS+92, Vec02]. Os administradores de algumas dessas fontes de dados resolveram atacar o problema da redundância onde

sequências similares foram agrupadas, desde que fosse possível inferir que uma delas era a origem das outras. Os bancos que trataram o problema de redundância [Red02] são, por exemplo, UniGene [UniGene02], EGAD [EGAD02] e STACK [STACK02]. O GenBank também possui um conjunto não-redundante de dados.

As fontes de dados genéricas que armazenam sequências de nucleotídeos são aqueles que compõem o International Nucleic Acid Sequence Data Library, formado, como já foi citado anteriormente, pelas bases de dados denominadas de Genbank, DDBJ e EMBL. Estas fontes armazenam também informações sobre partes das sequências que codificam moléculas de proteínas ou de RNA, além de outras informações biológicas relevantes. Tais informações são anotadas no campo *features*. A descrição completa do conteúdo de tal campo pode ser encontrada em [Gbrel02].

Além destas, outras fontes de dados específicas de um organismo também armazenam informações sobre sequências, como por exemplo o AceDB e toda a família de fontes de dados que é baseada na sua arquitetura [AceDB02].

Outras fontes de dados especializadas (de deteminadas células ou componentes, de mutações, de funções gênicas, etc.) também armazenam informações de sequências, como por exemplo o Mitomap [MITOMAP02, KLB+02].

2.1.2.2 Sequências de proteínas

As fontes de dados de sequências de proteínas armazenam, além da própria sequência, informações sobre a função daquela proteína naquele organismo. Tais fontes de dados têm também como característica a redundância e inconsistência das informações sobre as sequências armazenadas.

Como mencionado anteriormente, a principal fonte de sequências de proteínas é o Swiss-Prot. Existe maior cuidado com a qualidade da informação que é anotada nesta fonte, seu conteúdo é não redundante e inclui extensas anotações sobre as sequências. No entanto, este cuidado exige um intervalo de tempo entre a inclusão de uma

sequência no EMBL e sua correspondente tradução para o Swiss-Prot. A fonte de dados que armazena esta tradução automática é o TrEMBL. Algumas fontes de nucleotídeos também armazenam sequências de proteínas, como por exemplo o Genbank.

2.1.2.3 Características especiais das Proteínas

As fontes de dados de proteínas são especializadas. Assim, as fontes de dados ENZYME [ExPASy02a] e LIGAND [LIGAND02] armazenam informações sobre enzimas. A fonte Pfam [Pfam02] armazena documentações acerca de famílias de proteínas, enquanto a fonte PROSITE [PROSITE02, FPB+02] armazena informações de padrões de proteínas. Já a fonte ProDom [ProDom02, CGK98, CSG00] armazena informações sobre domínios. Existem outras fontes de dados de grupamentos de proteínas segundo diferentes algoritmos, como por exemplo BLOCKS [BLOCKS02].

As fontes de sequências de proteínas contém links para as fontes descritas anteriormente, que têm anotações mais completas sobre a proteína.

2.1.2.4 Estruturas de proteínas

Estas fontes de dados armazenam as representações da proteína em um plano (estrutura secundária) ou em três dimensões (estrutura terciária). A principal fonte de estruturas terciárias é o PDB e de estruturas secundárias é a DSSP.

2.1.2.5 Taxonomia

As fontes de dados de taxonomia são bastante questionáveis, uma vez que não existe consenso entre os pesquisadores sobre as classificações ali contidas. Os exemplos destas fontes de dados são: Species 2000 [Species02], International Organization for

Plant Information [IOPI02], Integrated Taxonomic Information System [ITIS02], The Tree of Life Project [TLP02], entre outros.

Cabe ressaltar que o Genbank mantém também informações de taxonomia, que são definidas e mantidas por um grupo de especialistas próprio e independente.

2.1.2.6 Publicações

As fontes de dados de publicações armazenam e disseminam informações sobre a literatura científica de diversas áreas. Na área da biologia molecular, o mais importante repositório de tais informações é o MEDLINE, agora denominado PUBMED [NCBI02a], que pode ser acessado via NCBI, através da interface ENTREZ. O correspondente ao MEDLINE para a área agrícola é o AGRICOLA [AGRICOLA02].

2.2 Classificação das Aplicações da Biologia Molecular

Existe um grande conjunto de aplicações [MS97, KRT96] disponíveis para uso dos pesquisadores em biologia molecular. Muitas destas ferramentas estão disponíveis na Web, seja para uso imediato ou para *download* do código fonte. Uma lista bastante extensa das ferramentas mais utilizadas pode ser encontrada em [FioCruz02].

As ferramentas podem ser classificadas em: sistemas para depuração e submissão de sequências, aplicativos para análise de sequências, fontes de dados e recursos disponíveis para uso via *Web*.

Os sistemas de depuração e submissão de sequências (ex. LabBase [Goo94, RSG95]) são utilizados pelos laboratórios que executam tarefas de sequenciamento de forma continuada. Estes sistemas criticam a saída dos sequenciadores, segundo critérios pré-estabelecidos, cortam da sequência os vetores utilizados nas reações e submetem a sequência às fontes de dados genéricas (por exemplo, o GenBank) de forma

automática, para verificar se ela já existe na base. Se não existir, a sequência é armazenada na fonte de dados, sendo o laboratório (ou o pesquisador) que submeteu a sequência o responsável pela qualidade da informação fornecida. Caso contrário, a sequência é rejeitada. A maioria destes sistemas é de propriedade de seu desenvolvedor.

Os aplicativos para análise de sequências são ferramentas bastante difundidas e executam funções importantes como comparação de sequências (BLAST [AGM+90, AMS+97, GS93, NCBI02b, GDB02a], FAST [Pea91, Pea94, Pea90, PL88]), alinhamentos [NW70, MSAP02, BCM02, SAC02, ClustalW02, THG94], pesquisa de genes (Gene Finder [SS97]), entre outras. Muitos destes aplicativos tem o código fonte disponível publicamente. A disponibilização de um conjunto dos aplicativos mais usados (de comparação de sequências, alinhamentos, montagem, etc.) forma uma poderosa ferramenta de suporte à pesquisa, como por exemplo o pacote GCG - Genetics Computer Group [GCG02]. Esforços no sentido de aumentar o desempenho do algoritmo BLAST tem sido desenvolvidos na PUC-Rio [Lem00, Cos02].

As fontes de dados de biologia molecular são também importantes para armazenamento das informações resultantes da pesquisa, como por exemplo: condições de experimentos, sequências de nucleotídeos, de proteínas, genes, mutações, outras anotações biológicas, estruturas das proteínas, taxonomia, publicações, laboratórios e pesquisadores.

Algumas destas fontes de dados são implementadas em sistemas de gerência de bancos de dados comerciais disponíveis no mercado (ex: Genome Sequence Data Base, GSDB, que utiliza Sybase), outras são repositórios de dados em formato próprio (ex: GenBank, que utiliza *flat files*) e algumas utilizam sistemas de persistência de dados desenvolvidos especificamente para suporte a uma dada pesquisa (ex: AceDB, implementado através de um gerenciador de objetos persistentes próprio). A maioria destas fontes de dados estão inseridas em uma ferramenta mais ampla que também disponibiliza para os usuários algumas das funcionalidades descritas anteriormente, especialmente as de análise de sequências.

O AceDB conta ainda, para visualização dos dados, de uma interface gráfica que fornece uma forma especial de apresentação de sítios relevantes de um cromossomo (mapa do cromossomo), aliado a uma funcionalidade do tipo *drill down* (implementa o detalhamento das informações), onde, a partir de um simples clique do *mouse* em uma região do cromossomo, é exibido o mapa com as informações biológicas que estão inseridas naquela região, e assim sucessivamente, até a forma mais básica de informação que é a própria sequência.

Muitas das ferramentas de suporte à pesquisa em biologia molecular estão disponíveis para utilização via *Web*. Neste caso destacam-se as ferramentas de busca a uma dada fonte de dados ou mesmo a grupos de fontes de dados pré-definidas, onde um formulário é exibido para a submissão de consultas, indicando, por exemplo, atributos sobre os quais se pode definir uma consulta à fonte de dados [Entrez02]. As consultas são enviadas para as fontes de dados e as respostas que atendem à consulta são agrupadas para apresentação do resultado.

Algumas destas ferramentas tem um formato de entrada de dados pré definido e é questionada a localização e o nome do arquivo. Assim, se os dados forem convertidos para o formato apropriado, as ferramentas podem ser utilizadas sem necessidade de alteração do seu código [ClustalW02, NCBI02b].

2.2.1 Requisitos das Fontes de Dados de Genoma e das Ferramentas de Integração

O estudo de fontes de dados para genoma é uma necessidade já que estas estão ficando cada vez mais complexas em termos da quantidade de dados a serem armazenados e tratados e na dinâmica de sua atualização. Assim, é cada vez maior a necessidade de que estas fontes de dados sejam tratadas via SGBD's de forma a dotar tais fontes de funcionalidades relevantes de bancos de dados, como por exemplo:

- facilidades de consulta;
- ferramentas de cópia e recuperação de dados;

- segurança (existem atributos de confidencialidade sobre as sequências indicando se a mesma é pública e pode ser divulgada, ou não), entre outras.

Tais SGBD's devem tratar o crescente volume dos dados armazenados e apresentar tempos de resposta adequados às diferentes consultas que podem ser formuladas.

A consulta pode ser tradicional, via SQL, procurando acessar uma sequência pelo seu identificador (número de acesso), pelo nome do pesquisador que submeteu a sequência, pela data de submissão, ou por outros atributos. A resposta, neste caso, consiste de uma tupla ou de um conjunto de tuplas, que atendem aos requisitos da consulta. Existe, no entanto, a necessidade de se formular outro tipo de consulta, que deve ser processada através de algoritmos específicos, quais sejam: de busca de sequências similares no banco de dados, de alinhamentos de sequências, de concatenação de sequências, e outros. A resposta, nestes casos, varia em função do algoritmo utilizado.

Os requisitos das fontes de dados de genoma são:

- aumento do volume de dados armazenados;
- melhora do tempo de resposta a consultas mais elaboradas, que exigem algoritmos específicos;
- tratamento da complexidade dos tipos de dados e dos algoritmos envolvidos;
- atendimento ao requisito de flexibilidade do esquema;
- adoção de um modelo de dados que esteja em conformidade com a pesquisa na área da biologia molecular;
- uso de filtros ou índices para acesso às sequências (atualmente os algoritmos de busca efetuam uma varredura completa na base);
- disponibilização, para os usuários, de uma interface adequada; e o
- uso de técnicas de *data mining*, para análises posteriores à fase de sequenciamento (como por exemplo a descoberta de regiões que codificam proteínas, de relações entre estas regiões, entre outras).

Além disso, a proliferação de tais fontes de dados com informações diferentes e complementares tem acarretado a necessidade de comunicação e integração entre elas, de forma a se compor um repositório completo de informações acerca do código genético que vem sendo descoberto em centros de pesquisa de todo o mundo.

Os requisitos das ferramentas de integração são:

- acesso a dados atualizados, esta atualização ocorre constantemente nas fontes de dados componentes da integração;
- formulação de consultas a diversas fontes de dados, ou seja, as fontes não devem ser limitadas pela ferramenta;
- gerenciamento da heterogeneidade das fontes, tratamento das diferenças semânticas;
- transparência de localização, ou seja, os usuários não devem conhecer os detalhes de localização das fontes componentes da integração, e
- transparência de esquema, ou seja, os usuários não devem conhecer os detalhes dos esquemas das fontes componentes da integração.

É, portanto, de fundamental importância o estudo das fontes de dados, no sentido de investigar melhorias no projeto e na implementação dos sistemas de integração, para suportar de forma adequada as necessidades da pesquisa em biologia molecular.

2.3 Abordagens para o Problema de Integração da Biologia Molecular

Como observado nas seções anteriores, uma ferramenta de suporte adequado à pesquisa em biologia molecular passa pelo acesso a diferentes fontes de dados da área, que são distribuídas e heterogêneas. Assim, essa questão, bastante discutida na literatura, deve ser contemplada.

Além disso, é necessário que a ferramenta de suporte permita a formulação de consultas aos dados e também a execução de aplicações que exigem dados

representados em formatos específicos. Esse aspecto não tem sido considerado nas propostas de integração existentes.

Outro ponto fundamental para a pesquisa na área reside na frequente alteração dos esquemas das fontes de dados componentes da integração, que também não é uma funcionalidade tratada de forma usual e sistemática nas propostas e nos projetos de integração publicados.

De forma geral, existem diferentes abordagens na literatura para tratar o problema de integração de informações de fontes de dados distribuídas e heterogêneas. A primeira trata da integração de bancos de dados através de um Sistema Gerenciador de Bancos de Dados Distribuídos e Heterogêneos [UML96, ULM97, Uch99, SL90, OV99, NK99, SKS97]. Essa abordagem, no entanto, não é adequada para tratar o problema de integração de dados da biologia molecular pois as fontes de dados, em sua maioria, se constituem de arquivos texto, com alguma estruturação. Também, não existe um esquema global, pactuado entre os diferentes laboratórios. Na verdade, informações que indiquem fenômenos biológicos (por exemplo, em atributos do tipo características de dados de sequências), bem como algumas instâncias de dados, são muitas vezes confidenciais.

Uma segunda abordagem presente na literatura trata de possibilitar a consulta às múltiplas fontes de dados via *Multidatabase* [DCB02, BCD+98, BDH+95, DOB95, BBB+98, BGB+99, PSB+99, BSN+99, SBB00]. Nesta abordagem, as fontes de dados componentes da integração são acessadas para atender a uma dada consulta. Essa proposta também não é adequada para tratar do problema pois as fontes de dados componentes (locais) podem estar indisponíveis no momento da consulta. Além disso, a pesquisa em biologia molecular exige alto desempenho que não é atendido pelo ambiente distribuído.

Outra abordagem trata a questão da integração utilizando-se a tecnologia de *Data Warehouse*, que implementa uma visão materializada do esquema das fontes de dados componentes da integração [MPH99, EBP+99, PKH+00, CPW+01, BBB+98,

BGB+99, PSB+99, BSN+99, SBB00]. Esta forma de integração também não atende à integração de informações biológicas com relação à atualização dos esquemas e também com relação à atualização das instâncias de dados (os pesquisadores precisam ter acesso aos dados mais recentes).

O contexto da biologia traz diversas questões para a integração de informações que exigem um tratamento mais aprofundado. Primeiramente, o número de fontes de dados é muito elevado, tornando problemática a integração de visões e a resolução de conflitos. Em segundo lugar, as fontes de dados têm comportamento dinâmico, tornando a inclusão e exclusão de fontes uma tarefa que deve ser tratada de forma a não provocar impactos sobre a visão integrada. Finalmente, as fontes de dados utilizam diferentes recursos de computação, que variam desde SGBD's completos até arquivos texto, que podem ser semi-estruturados, dificultando a integração de visões. Para tratar o problema de integração, a comunidade de pesquisas em bancos de dados revisou a solução via *multidatabases* passando a utilizar *wrappers* e mediadores [OV99].
Assim, para cada fonte de dados, o *wrapper* exporta informações de esquema, dados e sua capacidade de processamento de consultas.

O mediador é a ferramenta de *software* que fornece aos usuários uma interface de acesso uniforme para as diversas fontes de dados. A ferramenta "esconde" dos usuários a localização física das fontes de dados, efetua consultas isoladas às diversas fontes de dados, sem detalhar que fontes devem participar da consulta e combina os resultados, sem intervenção do usuário [KBT+01].

O acesso às fontes de dados através do mediador tem como pré-condição a disponibilização de metadados ou meta informação, isto é, descrições sobre os dados. Estes metadados podem ser: dicionários, vocabulários, ontologias e definições de esquemas (objetos, propriedades, comportamentos e relações entre os objetos) [KBT+01].

A heterogeneidade das fontes de dados se dá em diferentes níveis, a saber [Wie93]:

- quanto ao modelo de dados utilizado, a informação presente nas fontes de dados pode estar armazenada, por exemplo, em relações, em objetos, em documentos semi-estruturados, em arquivos textuais e em arquivos que contém informações classificadas como multimídia;
- quanto aos tipos de dados utilizados, por exemplo, o salário pode ser armazenado em um inteiro ou em uma cadeia de caracteres;
- quanto aos significados dos ítens de dados, por exemplo, salário mensal e salário/hora;
- quanto ao comportamento dos objetos envolvidos;
- quanto ao significado dos conceitos, por exemplo um aposentado pode não ser considerado como empregado para a folha de pagamentos mas ser considerado como tal para o sistema de benefícios;
- com relação ao esquema onde a informação irá se ajustar, que não pode ser rígido, de modo a permitir a sua evolução.

2.3.1 Abordagens de Implementação dos Mediadores

As abordagens envolvidas na construção de mediadores são:

- Visão global (Global as View - GAV)

  Nesta abordagem, a integração é feita utilizando-se fontes de dados pré-selecionadas de acordo com necessidades de informação pré-definidas. Neste caso, é utilizada uma abordagem procedimental (Tsimmis [Tsimmis02], Squirrel [HZ96], WHIPS [WHIPS02]) para a integração de informações de diferentes fontes de dados, via rotinas *ad-hoc*. Se as fontes de dados participantes da integração sofrerem alteração ou mesmo se houver nova necessidade de informação, um novo mediador deve ser gerado.

- Visão Local (Local as View - LAV)

  Nesta abordagem a integração é feita utilizando-se fontes de dados arbitrárias, também com necessidades de informação pré-definidas. A abordagem, neste caso, é declarativa (Carnot [Carnot02, SCH+97], SIMS [SIMS02], Information Manifold

[Lev98], Infomaster [Infomaster02]). Estes mediadores tem mecanismos capazes de reescrever as consultas de acordo com as descrições das fontes de dados participantes da integração.

- Visão Combinada (Global and Local as View – GLAV)
  Esta abordagem de integração é implementada através de uma combinação das abordagens anteriores.

2.3.2 O Processo de Mediação

Pode-se identificar duas fases distintas e independentes do processo de mediação. Uma fase trata da construção do esquema global e a outra da operação em si [KBT+01].

A fase de construção do esquema global está focada na definição e declaração da meta-informação. Esta fase é desenvolvida obrigatoriamente pelos pesquisadores de mais alto nível envolvidos neste processo. É assumido que a meta-informação produzida nesta fase é, de certa forma, estável por um certo período de tempo e que, a princípio, o esquema global sofre apenas extensões. Os esquemas das fontes de dados representativas do domínio do conhecimento, bastante conhecidas, são utilizados durante o processo de definição das meta-informações. As meta-informações criadas na fase da construção são, para o mediador, o nível de representação da federação.

Na fase de operação, as fontes de dados devem ser registradas pelo mediador (via captura de esquemas) e representadas no nível da federação. Esse processo de registro é autônomo e pode ser aplicado a uma dada fonte de dados, independente de outra fonte qualquer. Os usuários do mediador precisam conhecer as informações apenas no nível da federação e neste nível são formuladas as consultas aos dados. O mediador decide que fonte de dados são relevantes para uma dada consulta. A consulta pode se beneficiar de um vocabulário / dicionário para melhorar a qualidade de resposta.

2.3.3 Vantagens do uso de Mediadores

A adoção de mediadores traz as seguintes vantagens ao processo de integração [KBT+01]:

- Torna possível implementar a integração semântica das fontes de dados heterogêneas;
- Usuários precisam conhecer somente as definições dos conceitos, estruturas e métodos que constam no nível da federação;
- As fontes de dados existentes podem ser incorporadas (registradas) e, consequentemente, suas informações (relativas à integração) podem ser disseminadas, de forma independente em relação às outras fontes e a qualquer tempo;
- Usuários não precisam conhecer o processo de registro;
- Informações referentes à plataforma de implementação das fontes de dados (modelo de dados, linguagens, ambiente de *software* e de *hardware*) são absolutamente independentes do ambiente do mediador e das definições que contém;
- Ao formular consultas através de uma dada definição contida no mediador, os usuários têm acesso integrado a todas as informações registradas no mediador, no momento da consulta;
- Mediadores podem formar estruturas recursivas, onde cada mediador pode ser registrado em outro. Assim, múltiplos domínios de conhecimento podem ser integrados (em termos semânticos) através da definição de mediadores de alto nível;
- Visões personalizadas sobre as definições dos conceitos podem ser criadas para grupos específicos de usuários. Este processo é independente das fontes de dados.

2.3.4 Desvantagens no Uso de Mediadores

A utilização de mediadores tem as seguintes desvantagens [KBT+01]:

- Requer um nível apropriado de organização e maturidade da comunidade científica. As fontes de dados das pesquisas devem ser suficientemente abertas/documentadas para que possam ser registradas;
- O processo de registro não é simples e requer o desenvolvimento de ferramentas específicas de suporte.

2.3.5 Comentários Finais

Não existe nenhum consenso sobre como os *wrappers* descrevem informações sobre as fontes de dados. Entretanto, o modelo *wrapper*-mediador é uma abstração com ampla aceitação para o problema de integração tratado nesta seção.

Uma arquitetura baseada na abordagem *wrapper*-mediador difere fundamentalmente da abordagem *data warehouse* pelo fato de que os dados, na primeira, não são materializados. Porém, estas abordagens podem ser complementares pois o mediador pode ser usado como fonte de dados para a implementação do *data warehouse*.

2.4 Abordagens de Integração no Contexto de Biologia Molecular

Um enorme progresso tem ocorrido nos últimos anos no sentido de disponibilizar ferramentas que permitam integrar informações biológicas, presentes em inúmeras fontes de dados e que são acessadas via *Web*. Tais ferramentas, no entanto, ainda se mostram pouco funcionais porque implementam formas de integração parciais ou porque não implementam todas as funcionalidades necessárias. O ambiente dificulta a adoção de ferramentas baseadas em abordagens de formulação de consultas distribuídas porque as respostas são demasiadamente lentas. As abordagens de integração utilizadas pelas ferramentas disponíveis para a pesquisa em biologia molecular são apresentadas a seguir.

No primeiro caso, encontram-se as ferramentas de integração implementadas via navegação hipertexto, que permite que os usuários naveguem entre registros de

diferentes fontes de dados através *links* existentes entre eles (ex. Entrez [Entrez02]) ou através de sistemas de navegação que criam os *links* entre diferentes fontes de dados (ex. SRS [SRS02b], LinkDB [LinkDB02]). Assim, em uma primeira operação, o usuário acessa um registro da fonte de dados pesquisada e, na segunda, o usuário solicita um *link* para uma outra fonte de dados onde está a informação que procura.

No segundo caso, encontram-se as ferramentas de integração que implementam consultas às diferentes fontes de dados pré-existentes (como é o caso de sistemas *multidatabase*). Estas consultas podem ser formuladas através de um mediador baseado em uma linguagem de consulta própria (ex. CPL/Kleisli [CPL02]) que permite representar tipos de dados complexos, bastante comuns na biologia, onde, para cada fonte de dados a ser acessada, é implementado um tradutor da consulta específico a fim de possibilitar o acesso.

Outra estratégia consiste em se utilizar de um mediador que é responsável por determinar as fontes de dados que participam da consulta, criar e executar um plano de acesso, traduzir conceitos e sintaxe, formular as consultas ao ambiente distribuído e integrar os resultados [Kar95]. A implementação destas estratégias apresenta resultados bastante lentos quando operam em um ambiente distribuído como a *Web* porém os tempos de resposta são adequados quando a operação é realizada em um ambiente local.

As ferramentas de integração que se encontram no terceiro caso tratam da implementação de uma instância de dados que engloba as informações biológicas disponíveis em diversas fontes (*data warehouse* (IGD [IGD02a, IGD02b], GIMS [GIMS02a]). Ao contrário das ferramentas existentes na estratégia anterior, estas apresentam boa performance no atendimento a consultas porém falham no aspecto de atualização das instâncias de dados e também na atualização dos esquemas das fontes de dados, tarefas esta que são bastante comuns e que exigem constante manutenção das ferramentas.

Uma outra possível forma de integração de informações é baseada na implantação de uma federação das fontes de dados de biologia molecular. A federação pressupõe um esquema global e mapeamentos entre este esquema e o esquema local de cada fonte de dados participante da federação. Este esquema de integração não é utilizado pela comunidade de bioinformática, possivelmente devido à dificuldade e complexidade de modelar o esquema global pois os esquemas locais estão em constante evolução. Outra explicação reside no fato de que diversos laboratórios implementaram bases de dados próprias, adaptadas às suas necessidades. Além disso, muitas ferramentas de *software* já foram desenvolvidas para as necessidades específicas de cada laboratório. As mudanças e adaptações, que certamente viriam com a federação, apresentam custos elevados (por exemplo, para a reescrita do software), sem que significativos benefícios locais sejam alcançados [Kar95, BDH+95].

2.5 Discussão das Abordagens de Integração de Biologia Molecular

De forma geral, as ferramentas que visam a integração de fontes de dados de biologia molecular apresentam limitações que serão descritas a seguir.

Em termos de flexibilidade para adaptação à evolução da pesquisa na área de biologia molecular algumas ferramentas de integração podem ser consideradas pouco flexíveis por diversos fatores.

- Em primeiro lugar porque adotam um esquema próprio, dificultando (ou não facilitando) a compreensão dos esquemas utilizados nas diversas fontes de dados;
- Em segundo lugar, as ferramentas disponíveis têm dificuldades para se adaptar às alterações de esquemas, operação bastante comum nesta área, cujo conhecimento está em constante evolução;
- Finalmente, tais ferramentas não permitem o uso de aplicações que são essenciais para a pesquisa nesta área.

Muitas das ferramentas de integração apresentam baixo desempenho, dificultando a sua utilização prática em um ambiente distribuído.

Em termos de extensibilidade, podem ser citadas as seguintes limitações:

- As ferramentas frequentemente limitam o uso das fontes de dados existentes;
- Em geral, as ferramentas admitem um reduzido conjunto de aplicações.

Além disso, seria bastante útil que as ferramentas de integração permitissem a instanciação de fontes de dados apropriadas para uma dada pesquisa [DCB01], onde os pesquisadores envolvidos conhecessem profundamente os atributos que a compõem, facilitando a formulação de consultas, o acesso aos dados e a análise dos resultados.

Outro aspecto trata da atualização das instâncias de dados nas ferramentas que implementam mecanismos de materialização dos dados. É importante que os dados sejam atualizados sistematicamente de forma a disponibilizar, para os pesquisadores, as informações mais recentes.

2.6 Comentários Finais

Este capítulo tem como objetivo motivar o problema. Inicialmente é fornecida uma visão global da pesquisa em bioinformática. Para tanto, foram apresentadas as principais fontes de dados com informações relativas à biologia molecular e as principais aplicações utilizadas na área.

Em seguida foram relacionados os requisitos de uma ferramenta de integração de forma a atender às necessidades da pesquisa na área de biologia molecular.

Foram ainda discutidos aspectos relativos às abordagens de integração de informações distribuídas e heterogêneas, em especial foi discutido o uso de mediadores como forma de solução.

Foram também apresentadas as abordagens de integração existentes no contexto da biologia molecular e foram discutidas estas abordagens em termos de vantagens, desvantagens e limitações de cada uma.

O capítulo seguinte apresenta a solução de integração que é proposta neste trabalho.

3 A Solução Proposta

Este capítulo apresenta inicialmente a tecnologia de *frameworks*, de forma a fundamentar os conceitos que serão utilizados ao longo do texto. O termo *framework*, bastante genérico na língua inglesa, é utilizado neste trabalho conforme definido em engenharia de *software*, isto é, como uma arquitetura flexível e extensível para a construção de uma família de sistemas, aplicadas a um determinado domínio.

Em seguida, é apresentada a proposta para a solução do problema de integração, que é baseada na tecnologia de *frameworks*. O *framework* proposto é descrito através dos diagramas da linguagem UML [BRJ99]. São apresentados os diagramas de classes (modelo estático) e os diagramas de sequência (modelos dinâmicos) e sempre que possível, são identificados na especificação padrões de projeto (*design patterns)* [GHJ+95, GHJ+93].

3.1 A Tecnologia de *Frameworks*

Framework é um (sub)sistema de software parcialmente completo, criado com objetivo de ser instanciado [BMR+96].

Um *framework* define uma arquitetura para uma família de sistemas e fornece blocos básicos para a sua construção. Também define as partes que devem ser adaptadas para realizar uma funcionalidade específica. Em um ambiente orientado a objetos, um *framework* é composto de classes abstratas e concretas e sua instanciação consiste de composição e herança de classes. As classes concretas são invisíveis para o usuário [WJ90].

Um *framework* consiste de *frozen spots* e *hot spots*. Os *frozen spots* definem a arquitetura global de um sistema de *software*, os seus componentes básicos e o relacionamento entre eles, e devem permanecer imutáveis em qualquer instanciação

do *framework*. Os *hot spots* representam as partes do *framework* que são específicas para cada instanciação [Pre94].

Frameworks são geradores de aplicações que estão relacionadas com uma determinada classe de problemas e soluções, ou seja, estão diretamente relacionadas com um domínio específico. Como *frameworks* são criados para gerar aplicações para um dado domínio, devem existir pontos de flexibilização (*hot spots*) que são customizados de acordo com aplicações específicas, onde cada uma soluciona um problema particular. Em virtude destes *hot spots*, os *frameworks* não são sistemas, já que não são executáveis. Para se gerar um sistema o *framework* deve ser instanciado, ou seja, deve ser implementado, no *framework*, o código apropriado para uma determinada aplicação, dentro do seu domínio. Na maior parte dos casos, os *hot spots* são classes abstratas, isto é, não tem implementação e, ao serem instanciadas, exigem a implementação dos métodos abstratos, fato que que pode resultar em alguma adaptação nas classes [ML00].

Essa adaptação pode ser exemplificada em um framework de integração de dados, onde os *hot spots* são constituídos pelos *wrappers* para acesso às fontes de dados. Assim, pode ser necessário criar um *wrapper* para uma fonte do tipo banco de dados, onde é necessário criar uma conexão com o banco. Já em um *wrapper* para uma fonte de dados do tipo arquivo texto, essa conexão não é necessária.

3.1.1 Benefícios do uso de Frameworks

A adoção de um *framework* traz os seguintes benefícios [ML00]:

- Maior modularidade, obtida pelo encapsulamento de implementações flexíveis através de uma interface estável, facilitando a compreensão e a manutenção do *software*;

- Reuso de projeto e implementação, obtido na determinação dos *frozen spots*;

- Extensibilidade, obtida na determinação de métodos explícitos que possibilitam às aplicações estenderem suas interfaces (variações requeridas pelas instanciações de uma aplicação).

3.1.2 Classificação de *Frameworks* por escopo

Com base na literatura existente sobre *frameworks*, podemos classificá-los em [FS97, FSJ97]:

- *Frameworks* de Infra-estrutura

Os *frameworks* de infra-estrutura são aqueles utilizados para simplificar o desenvolvimento de ferramentas de infra-estrutura de *software* eficientes e portáveis, como por exemplo, em bancos de dados [BP01], sistemas operacionais [CRJ87], em ferramentas de comunicação de dados [Sch97] e na construção de interfaces com o usuário [KP88]. Os *frameworks* de infra-estrutura podem ser desenvolvidos e utilizados por empresas de criação de *software* básico e estão envolvidas com o processo de produção de *software*.

- *Frameworks* de Integração

  São utilizados para integrar aplicações e componentes distribuídos. São projetados para aumentar a capacidade de modularização e reuso de ferramentas de *software*. Também conhecidos como *middleware*, os *frameworks* de integração têm sido desenvolvidos no sentido de estender as funcionalidades do *software* em questão para trabalhar em ambiente distribuído, por exemplo frameworks ORB [RBP+99] e em bancos de dados [Uch99].

- *Frameworks* de Aplicação

Referem-se a grandes domínios de aplicações, como por exemplo, em telecomunicações [CHS+97], finanças [Yod97], entre outros. Formam, de fato, a base das atividades do negócio.

Atualmente o desenvolvimento de *frameworks* tem contemplado a junção dos domínios de aplicação e integração, via composição. Isto tem ocorrido, por exemplo, para dotar os sistemas de facilidades de operação em ambientes distribuídos, ou seja, de facilidades de comunicação entre objetos locais e remotos.

3.1.3 O processo de Desenvolvimento de *Frameworks*

É dividido nas seguintes etapas [ML00]:

- Análise do domínio

  Consiste de um amplo aprendizado e uma análise completa do domínio de problemas que o *framework* se propõe solucionar. Tal aprendizado deve antecipar os possíveis requisitos futuros. A captura de requisitos se baseia em experiências anteriores publicadas ou obtidas através de levantamentos junto a especialistas, em sistemas pré-existentes e em padrões de comportamento a serem considerados. Nesta fase os *hot spots* e os *frozen spots* devem ser definidos.

- Projeto do framework

  Consiste na criação da abstração do *framework*. Nesta fase, a arquitetura (*frozen spots* e *hot spots*) deve ser modelada, utilizando-se, por exemplo, os diagramas da UML, e devem ser explicitadas a flexibilidade e extensibilidade que são admitidas. Os *hot spots* devem ser projetados tendo como foco a extensibilidade. Soluções que propiciem reuso, como a aplicação de padrões de projeto (*Design Patterns*), também são aplicadas nesta fase [Boo94].

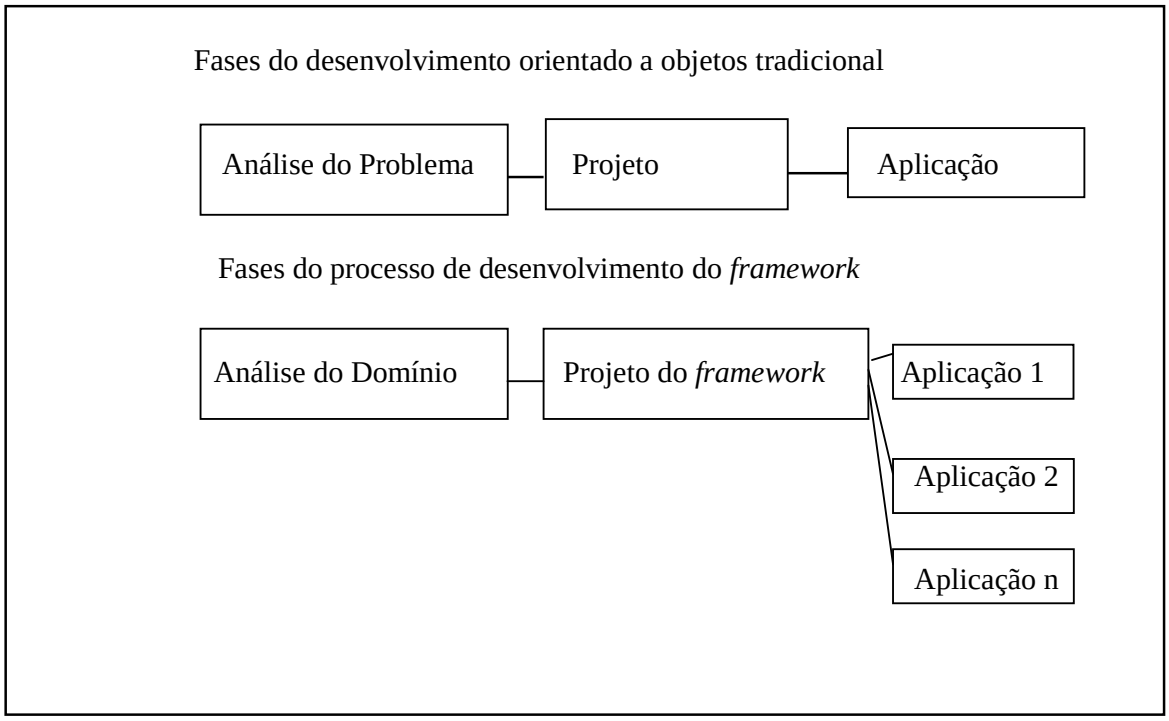

Figura 2 - Fases do processo de desenvolvimento de um *framework*

- Instanciação do framework

  A fase de instanciação consiste da implementação dos *frozen spots* e *hot spots*. A implementação dos *hot spots* gera um sistema de aplicações que é executável. É importante ressaltar que estas aplicações têm em comum os *frozen spots*.

As fases do processo de desenvolvimento do *framework* são comparadas na Figura 2 com as fases tradicionais de desenvolvimento de um projeto orientado a objetos. A análise do domínio procura uma solução geral para vários problemas de um mesmo domínio que, quando encontrada, gera uma família de soluções.

3.1.4 Questões a considerar ao se desenvolver um *Framework*

O desenvolvimento de um *framework* orientado a objetos tem como objetivo prover alta produtividade e um menor tempo entre o desenvolvimento e a disponibilização das aplicações, através do reuso que é aplicado ao projeto e à codificação. Estes objetivos são alcançados através da arquitetura flexível dos *frameworks*, que viabiliza o reuso do núcleo da arquitetura em suas diferentes instanciações [ML00].

A seguir são apresentadas algumas questões a serem consideradas na escolha do desenvolvimento via *frameworks* [ML00].

3.1.4.1 Desenvolvimento de um gerador de aplicações versus desenvolvimento da aplicação

Frameworks são em geral mais complexos de se modelar e construir do que uma aplicação. O custo de desenvolvimento do *framework* é no mínimo igual ao do desenvolvimento da aplicação e, em geral, é bem maior. Assim, deve-se pesar com cuidado a necessidade de se construir a aplicação através de uma arquitetura flexível pois, neste caso, a arquitetura deve contemplar todos os requisitos para atender necessidades futuras.

Por outro lado, o esforço de se construir uma arquitetura flexível pode se justificar, simplesmente pela economia de codificação, ao serem desenvolvidas diversas aplicações de um mesmo domínio.

3.1.4.2 Composição de *Frameworks*

É comum a composição de *frameworks* de forma a completar os requisitos da aplicação. No entanto esta tarefa não é simples. O processo de desenvolvimento de *frameworks* sem a intenção de uma futura composição, pode gerar uma série de problemas, como por exemplo:

- falta de aderência de interface,
- diferenças de cobertura do domínio,

- conflitos de pré-requisitos e objetivos de projeto,
- ausência de padrões.

3.1.4.3 Instanciação do *Framework*

Um *framework* pode ser classificado de acordo com as técnicas de extensão (criação dos *hot spots*) a serem utilizadas, que são denominadas *white box*, *black box* e *gray box*.

Na técnica *white box*, a instanciação do *framework* só é possível através da codificação e criação de novas classes, que podem ser introduzidas na arquitetura via herança e amarração dinâmica. Os desenvolvedores devem conhecer bem a arquitetura de modo a produzir uma nova instância.

Nos *frameworks* do tipo *black box*, as instâncias são produzidas por parâmetros de configuração, que escondem do desenvolvedor os detalhes do projeto e da implementação. *Frameworks black box* são mais difíceis de desenvolver já que requerem que os desenvolvedores do *framework* definam interfaces que antecipem uma grande variedade de casos potenciais de uso. São estendidos através da definição de interfaces para componentes que podem ser encaixados (*plug-in*), no framework, via composição de objeto.

Os *frameworks* que tem as duas características descritas anteriormente são denominados *gray box*.

3.1.4.4 Documentação do *Framework*

A habilidade de se criar sistemas executáveis a partir de uma arquitetura abstrata depende não só de ferramentas que facilitem a instanciação das aplicações mas principalmente da documentação, que deve conter um guia informando como instanciar os pontes de flexibilização da arquitetura [CI93].

Diferentes documentos podem ser gerados como documentação do *framework* para um dado público alvo. Para atender a todos os tipos de usuário, a documentação deve conter diferentes níveis de abstração, que descrevem:

- Propósito do *framework* (deve indicar o domínio do problema e soluções);
- Como usar os fundamentos do *framework* (um guia de uso propriamente dito);
- Propósito das aplicações e exemplos (deve propiciar o entendimento da arquitetura);
- Projeto do *framework* (descrição técnica da arquitetura, incluindo classes, relacionamentos e colaborações).

O primeiro documento serve para se tomar a decisão de utilização do *framework* ou não. O segundo e o terceiro documentos explicam como utilizar a arquitetura e como garantir o reuso. Já o quarto documento contém detalhes que permitem efetuar manutenções na arquitetura. Este último documento deve apresentar uma descrição formal da arquitetura e detalhes sobre onde e como o *framework* pode ser adaptado ou especializado.

Como descrito anteriormente, na construção de *frameworks* frequentemente são utilizados padrões de projeto. Embora estes padrões (fragmentos da arquitetura) compreendam visões parciais do *framework*, são bem conhecidos e podem ajudar no processo de instanciação, por serem bem documentados e compreendidos.

3.1.4.5 Custo da Análise do Domínio

Como descrito anteriormente, *frameworks* são criados com a intenção de gerar aplicações em um domínio específico. Assim, uma das fases do desenvolvimento de um *framework* consiste na análise do domínio, ou seja, no estudo de requisitos de uma classe de problemas.

Diferentemente do estudo de requisitos de um sistema de *software* , os requisitos de um *framework* devem cobrir todo o domínio, tornando-o complexo e com elevado custo de projeto e manutenção. A questão central trata de dimensionar corretamente o domínio. Quanto maior for o domínio, mais custoso e difícil de projetar e manter, enquanto que a escolha de um domínio muito estreito pode implicar em uma aplicabilidade (do *framework*) aquém da necessária.

3.1.4.6 Flexibilidade versus performance e complexidade

Um *framework* é mais complexo e tem mais pontos de flexibilização do que sistemas de *software* tradicionais. A flexibilização é obtida através do uso de herança e de amarrações dinâmicas (*binding*), estruturas comuns em linguagens orientadas a objetos. Como as amarrações dinâmicas normalmente introduzem custos adicionais (*overhead*), há um compromisso entre flexibilidade e desempenho.

O uso abusivo de *hot spots* para introduzir soluções genéricas na arquitetura irá, sem dúvida, aumentar a sua complexidade. Assim, há também um compromisso entre flexibilidade e complexidade.

3.1.5 Comentários Finais

O desenvolvimento baseado em *frameworks* deve ser considerado quando os requisitos do sistema evoluírem rapidamente e for possível construir a arquitetura considerando esta evolução. A construção do *framework* também deve ser considerada quando houver necessidade de se implementar o desenvolvimento dos *hot spots* de forma incremental. Isto pode ser obtido ainda através da criação de *hot spots* inicialmente mais simples, que serão substituídos posteriormente por versões incrementais [FSJ97].

A tecnologia de *frameworks* é bastante recente em termos de pesquisa e também do próprio desenvolvimento de aplicações. Assim, existem ainda muitas questões em aberto, que foram mencionadas neste documento.

3.2 A Proposta de Arquitetura

Esta seção descreve a especificação da arquitetura proposta para solução do problema estudado (que se refere à integração de dados e aplicações de biologia molecular) utilizando-se a tecnologia de *frameworks*. O *framework* é apresentado e formalizado utilizando-se a linguagem UML [BRJ99]. São relacionados os diagramas de classes (modelo estático) e os diagramas de sequência (modelos dinâmicos). Na especificação, sempre que possível, são identificados padrões de projeto (*design patterns).*

3.2.1 Descrição Geral da Arquitetura

As limitações das arquiteturas de integração existentes, observadas no capítulo anterior, nos levaram a propor uma arquitetura de *software* baseada em um *framework* orientado a objetos, de forma a prover flexibilidade e extensibilidade.

O *framework* se propõe a integrar dados de qualquer fonte de dados de biologia molecular. Para tal, o projeto inclui *wrappers* que capturam o esquema e os dados das fontes de dados que participam da integração. Os *wrappers* têm ainda a função de traduzir os esquemas obtidos nas fontes de dados para XMLSchema e os dados para XML. Tanto os esquemas como os dados são armazenados em um repositório da arquitetura. O repositório utiliza o modelo de dados semi-estruturado. Assim, os esquemas são armazenados em XMLSchema [XMLSchema02] e os dados em XML [XML02].

Muito tem sido questionado com relação ao modelo de dados adequado para a representação dos dados da biologia molecular. O modelo relacional não facilita a

compreensão do objeto biológico (o que dizer do pleno entendimento, por um biólogo, de uma base de dados cujo esquema tem mais de 200 tabelas ?). Os modelos orientados a objeto tem dificuldades de adaptação às necessidades de alteração de esquemas. Assim, a maioria das fontes de dados representam os dados sob a forma de texto, onde marcadores especiais indicam o início e o fim dos atributos. Essa forma, além de facilitar a transferência de dados entre laboratórios, facilita a compreensão dos usuários. Por este motivo, a arquitetura adota o modelo de dados semi estruturado.

Um usuário especial, denominado administrador do sistema, é responsável pela compreensão dos esquemas capturados das fontes de dados e pela montagem do esquema global das informações biológicas. Para tal, o administrador conta com uma ferramenta de visualização dos objetos dos esquemas capturados, onde ele deve incorporar as descrições dos objetos (nome do objeto, tipo de dado, descrição, domínio) de forma a possibilitar o seu entendimento e uso por outros pesquisadores. Após a descrição completa dos objetos, o administrador deve acrescentá-los ao esquema global. Isto implica na definição de relacionamentos com outros objetos do esquema global, com a descrição das cardinalidades, na identificação de sinônimos e na definição de regras de conversão (de unidades, ou de valores, neste caso, por exemplo, o atributo sexo pode ser armazenado como 0 / 1 em uma fonte de dados e como “m” / ”f” em outra).

De posse desta ferramenta, o administrador pode definir um esquema específico para uma dada pesquisa (por exemplo, o esquema utilizado em GUS [DCB01]) para posterior instanciação. A instanciação dos dados (criação do *data warehouse*) facilita a compreensão dos usuários com relação ao esquema utilizado, facilita a formulação de consultas e diminui o tempo de resposta.

A partir deste ponto, o administrador pode decidir por instanciar o *data warehouse*. O mediador da arquitetura é então acionado e decide, a partir das definições do esquema global, que fontes de dados devem ser consultadas para obter os objetos do esquema a ser instanciado e como fazer as associações entre as instâncias de objetos. As regras

de conversão que foram definidas sobre os objetos são executadas no nível do mediador, bem como as conversões de tipo necessárias.

De posse do esquema instanciado, os usuários podem fazer consultas ao *data warehouse* via linguagem de consulta apropriada, como XML-QL [QL02] ou Xquery [Xquery02], ou mesmo em SQL, em bancos de dados que admitem um tipo de dados específico para XML, como o tipo XMLType definido no SGBD Oracle 9i [Oracle9i02] . Isso é função da tecnologia utilizada no repositório.

Cabe ressaltar que o esquema global documenta a ontologia que é efetivamente utilizada nas fontes de dados de biologia molecular. O maior detalhamento e completude da ontologia é função do número de fontes de dados capturadas. Observa-se ainda que não é necessário construir as instâncias dos esquemas capturados. Somente devem ser instanciados os esquemas necessários.

O administrador pode também definir *drivers* (conversores de dados) para possibilitar o uso, via arquitetura, das diversas aplicações disponíveis para a pesquisa em biologia molecular. Estes *drivers* são responsáveis pela conversão de formato dos dados, presentes na arquitetura, para aquele que é esperado como entrada para uma dada aplicação.

O esquema global da arquitetura contempla os métodos das classes utilizadas na biologia molecular. Assim, a ferramenta permite que os usuários executem estes métodos diretamente sobre os dados instanciados.

A arquitetura prevê ainda o constante monitoramento dos esquemas das fontes de dados capturadas, de forma a informar ao usuário administrador a existência de alterações (evolução do esquema) através de agentes de *software*, que agem no mesmo nível dos *wrappers*.

Um mecanismo semelhante de monitoramento das fontes de dados baseado em agentes é utilizado para realizar a atualização de instâncias de dados que foram

alteradas na fonte original. A escolha de XML para armazenamento dos dados facilita a tarefa de atualização pois muitas fontes de dados utilizam esta forma de armazenamento. Além disso, é tarefa de fácil execução a comparação de documentos XML ou mesmo de *tag's*, a inclusão de novas *tag's* ou a alteração das *tag's* existentes. A tarefa de atualização dos dados poderia ser facilitada se as fontes de dados contassem com arquivos independentes contendo somente os dados de atualização, com *timestamps* para cada registro. Algumas fontes de dados já começam a implementar mecanismos de atualização deste tipo.

A arquitetura é flexível no sentido de permitir a incorporação de *wrappers* para as fontes de dados existentes na biologia molecular e de *drivers* para as aplicações disponíveis, sem qualquer restrição ou limitação. A captura dos esquemas permite a definição de um esquema global e possibilita a definição de uma ontologia que é baseada nas fontes de dados capturadas. O problema do baixo desempenho nas consultas, bastante comum nas arquiteturas estudadas, é resolvido através da instanciação dos dados no *data warehouse*.

A arquitetura conta ainda com mecanismos de atualização de esquemas e de atualização dos dados instanciados.

A arquitetura implementada pretende ser uma importante ferramenta de suporte à pesquisa em biologia molecular, visto que apresenta soluções inovadoras frente às limitações descritas nas ferramentas de integração existentes.

O *framework* proposto trata da integração de esquemas baseada em um meta-modelo, o que faz com que a proposta seja diferente daquelas existentes na literatura. A arquitetura é baseada em um framework de aplicação, sendo que, na área de pesquisa de *frameworks* com domínio específico, não foi encontrada na literatura qualquer referência a estudo semelhante.

A integração é feita através de um mediador que captura os esquemas e dados das fontes, faz as conversões necessárias, semânticas e entre modelos (para XMLSchema

e XML) e materializa as informações no repositório. A instanciação de um *data warehouse* específico para uma dada pesquisa não foi também tratada nos projetos encontrados na literatura, porém sua necessidade é indiscutível.

Apesar de existirem publicações com o objetivo de se criar e estender uma ontologia para a área [GO02, KRP+98, SK98, XOL02, BGB+99], muitas são limitadas, tratam apenas de sub-conjuntos de termos utilizados no domínio do conhecimento e outras são definidas por um ou vários pesquisadores, sem que haja correspondência com as definições utilizadas nas fontes de dados. O trabalho proposto visa também contribuir para a representação e definição de uma ontologia específica para a área, efetivamente utilizada nas fontes de dados existentes.

A questão da atualização dos dados também não foi tratada de forma conveniente pelos projetos na área, sendo relegada a segundo plano ou utilizada como argumento contrário à criação do *data warehouse*. A questão de atualização dos esquemas também não foi tratada de forma conveniente. Na arquitetura proposta, agentes de *software* se encarregam de monitorar a evolução dos esquemas das fontes de dados capturadas e de informar ao administrador a existência de alterações. Este, por sua vez, deverá efetuar uma nova captura dos esquemas informados, de forma a incorporar à arquitetura a evolução verificada. Outros agentes de software se encarregam de monitorar as alterações verificadas nas instâncias de dados e de incorporar as alterações ao *data warehouse*.

3.2.2 Apresentação da Arquitetura

O *framework* contém as seguintes funcionalidades:

- Capturar o esquema de uma fonte de dados previamente existente, dado que já exista um conversor (*wrapper*) construído para aquela fonte;

- Efetuar o casamento dos objetos de um esquema global da biologia molecular (vide capítulo 5) inicialmente proposto, que é utilizado na arquitetura, e os

existentes nas fontes de dados capturadas. Podem ser criados novos objetos no modelo da biologia e devem ser criadas as associações pertinentes. O casamento pode ainda estabelecer relações do tipo “é_sinônimo_de” e regras de conversão;

- Definir um esquema novo, próprio de uma pesquisa específica, e para o qual se deseje instanciar dados;

- Capturar os dados das fontes de dados, via conversores (*wrappers*), envolvidas com o esquema que se deseja instanciar;

- Gerar dados em um formato exigido por uma aplicação da biologia molecular, externa à arquitetura;

- Executar algoritmos que estão instanciados no *framework* como métodos das classes do modelo de biologia molecular;

- Informar a atualização de um esquema, verificado em uma dada fontes de dados;

- Atualizar as instâncias de dados conforme estas forem sendo alteradas em sua fonte original.

A arquitetura do *framework* está subdividida em quatro módulos, cuja interdependência está esquematizada na Figura 3.

O módulo Administrador realiza a interface com os usuários, de forma a prover as seguintes funções: permitir gerenciar o modelo da biologia e esquemas específicos, solicitar a captura de esquemas e/ou de dados, permitir a formulação de consultas, permitir a execução dos algoritmos instanciados no próprio *framework* e a execução de aplicações externas. Este módulo contém um repositório, que armazena o modelo global da biologia.

O módulo Capturador administra o repositório de dados e de esquemas da arquitetura.

Os *Wrappers* implementam o acesso às fontes de dados de biologia, efetuando a tradução dos esquemas das fontes de dados para XML Schema e dos dados para XML;

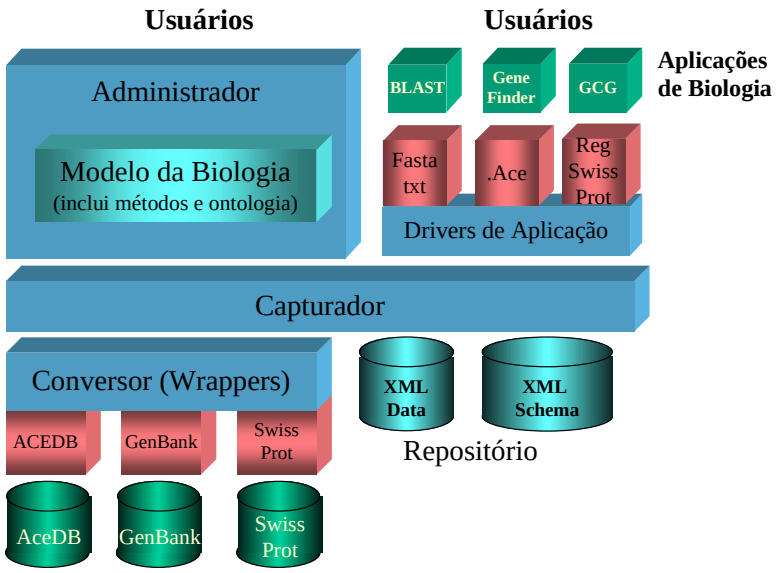

Figura 3 - Arquitetura do *framework*

Os *Drivers* implementam a conversão dos dados do formato interno do *framework* (XML) para o formato esperado pelas aplicações.

Por exemplo, um usuário especial denominado administrador solicita ao módulo Administrador a captura de um esquema, este repassa a solicitação para o módulo Capturador, que a repassa ao *Wrapper* da fonte de dados. A captura do esquema de uma fonte de dados só pode ser efetuada se o *wrapper* tiver sido previamente desenvolvido e instanciado, no *framework*. O *wrapper* implementa a conversão do esquema da fonte de dados para XML Schema. Os esquemas capturados são armazenados nos respectivos repositórios.

O usuário administrador passa então a reconhecer o esquema capturado e a comparar os objetos do esquema capturado com o esquema global (modelo da biologia),

enriquecendo este modelo com novos objetos capturados e identificando e documentando relacionamentos e sinônimos. O modelo da biologia passa, assim, a descrever os novos objetos (identificando a fonte de origem) e os relacionamentos entre os novos objetos incorporados ao esquema com os objetos ali existentes.

A partir do esquema global, o usuário administrador pode definir um sub-esquema do esquema global, de interesse específico de uma pesquisa. Informações do sub-esquema são armazenadas no repositório.

O usuário administrador pode também solicitar ao módulo Administrador a instanciação de um esquema do repositório. Neste caso, o módulo Administrador repassa a solicitação para o módulo Capturador, que solicita dados ao *Wrapper* da Fonte de dados de Biologia. A captura de dados de uma fonte da biologia só pode ser efetuada se o seu *wrapper* tiver sido previamente desenvolvido e instanciado, no *framework*. O *wrapper* implementa a conversão dos dados para XML. O módulo Capturador exerce a função de um mediador, distribuindo solicitações de dados aos *wrappers* de forma a compor o esquema a ser instanciado. Os dados são também armazenados no repositório.

O usuário administrador pode também acionar o módulo Administrador para solicitar que um arquivo, para uma dada aplicação de biologia, seja gerado. Da mesma forma, tal solicitação só pode ser efetuada se o respectivo *driver* tiver sido previamente desenvolvido e instanciado na arquitetura. O módulo Administrador aciona o *Driver* para que seja instanciado um arquivo para uma dada aplicação. O *Driver* por sua vez solicita os dados ao Capturador, que gerencia o repositório dos dados, recebe os dados e efetua a transformação necessária. Esta tarefa pode ser feita para qualquer esquema armazenado no repositório, global ou específico.

A arquitetura permite ainda que, via módulo Administrador, seja acionado um método da biologia que está instanciado no modelo e que opera sobre os dados disponíveis no repositório.

O *Driver de Aplicação* permite também a construção de interfaces entre o *framework* e as aplicações existentes da biologia, ou seja, podem ser instanciados pontos de entrada para solicitação de dados em formatos específicos para uma classe de aplicações. Assim, pode-se construir serviços de dados para aplicações desenvolvidas externamente.

3.2.3 Modelagem Estática: Diagrama de Classes da Arquitetura

Esta seção irá detalhar a arquitetura de cada módulo, apresentando suas classes, os métodos de cada uma e seus relacionamentos. Cada módulo tem uma classe (fachada) que representa a sua interface com os outros módulos, sendo utilizado o *pattern Facade* na sua implementação.

3.2.3.1 Administrador

Figura 4 - Diagrama de classes do módulo Administrador

O módulo Administrador (Figura 4) é composto pelas seguintes classes:

- FachadaAdm, provê todas as funcionalidades do *framework* do usuário administrador. Os usuários interagem com esta classe para realizar:

    1. a captura dos esquemas de fontes de dados cujos conversores (*wrappers*) foram previamente instanciados na arquitetura (e respectiva exclusão);

    2. o casamento (*matching*) entre o modelo da biologia que está presente na arquitetura e o esquema que foi capturado de uma fonte de dados (permitindo, caso não haja tal casamento, que novos objetos sejam adicionados ao modelo ou suprimidos, que os objetos sejam associados ou que sejam desfeitas associações, e que objetos sejam reconhecidos ou não como sinônimos de outros com as respectivas regras de conversão de tipos e de dados). Esta funcionalidade permite que novos conceitos da biologia sejam inseridos no modelo. As definições de novos objetos e de associações e sinônimos entre objetos do modelo devem ser representadas em XML Schema;

    3. a criação de um novo esquema, apropriado para uma dada pesquisa, cujos dados se deseje instanciar. O novo esquema vai ser criado a partir dos objetos já definidos no modelo da biologia (é de fato uma visão);

    4. a captura dos dados para instanciar um esquema específico ou mesmo o esquema global;

    5. a geração de dados para aplicações externas;

    6. a consulta aos repositórios de esquemas e de dados;

    7. a execução de algoritmos da biologia.

- ModeloBiologia, manipula o modelo de dados de biologia, permitindo a sua expansão.

- RepositórioModelo, permite que os objetos do modelo da biologia sejam persistidos e recuperados do Repositório.

3.2.3.2 Capturador

O módulo Capturador (Figura 5) é composto pelas seguintes classes:

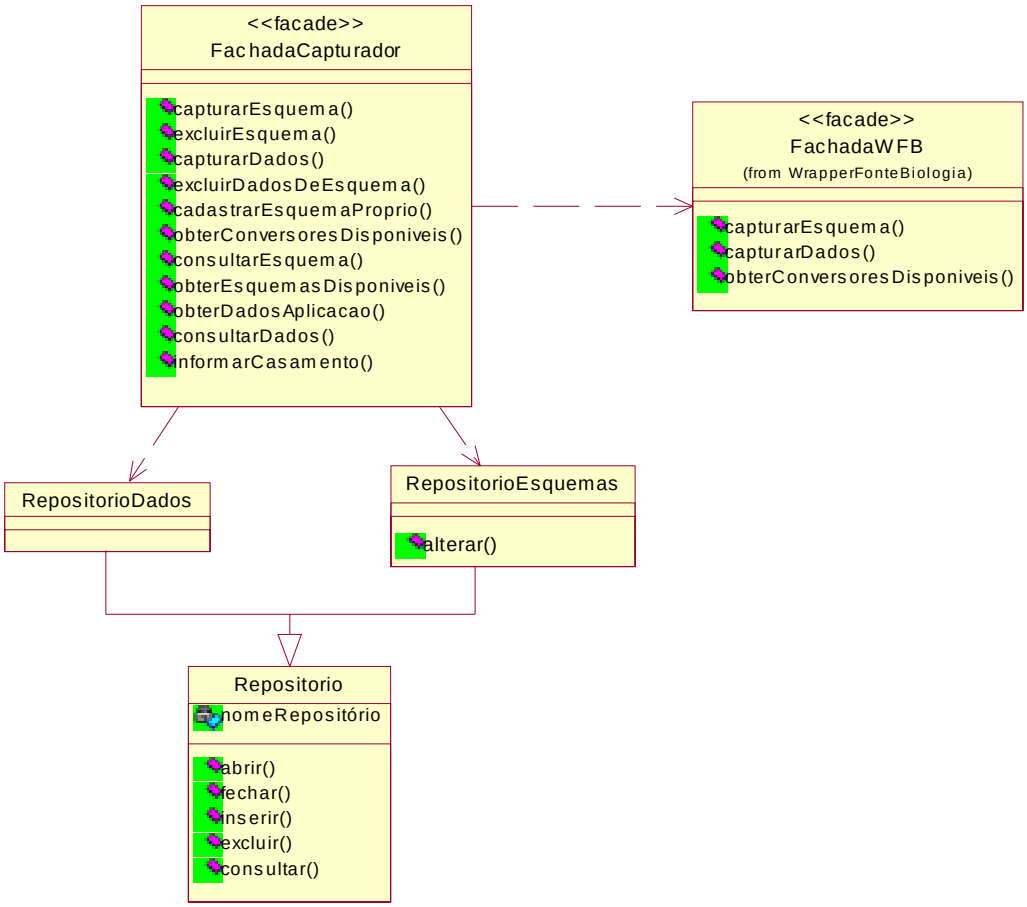

Figura 5 - Diagrama de classes do módulo Capturador

- FachadaCapturador, provê as seguintes funcionalidades:

1. a captura e o armazenamento de esquemas de fontes de dados de biologia;

2. o armazenamento de esquemas específicos, definidos a partir dos objetos do modelo da biologia que está disponível no *framework*;

3. a exclusão de um esquema, seja ele capturado ou específico;

4. a captura e o armazenamento dos dados associados a um esquema;

5. a exclusão dos dados que foram associados a um esquema;

6. a execução de consultas sobre o repositório.

Cabe observar que os esquemas são armazenados em XML Schema e os dados em XML. É utilizada uma ferramenta que permite a formulação de consultas ao Repositório e às fontes de dados.

- Repositório, permite que os esquemas e dados sejam persistidos e recuperados. Pode contar com linguagens de acesso e manipulação a dados em XML (Xquery [Xquery02] ou XML-QL [QL02]) ou mesmo SQL (se a ferramenta utilizada o permitir).

3.2.3.3 *Wrappers* de Fontes de Biologia (WFB)

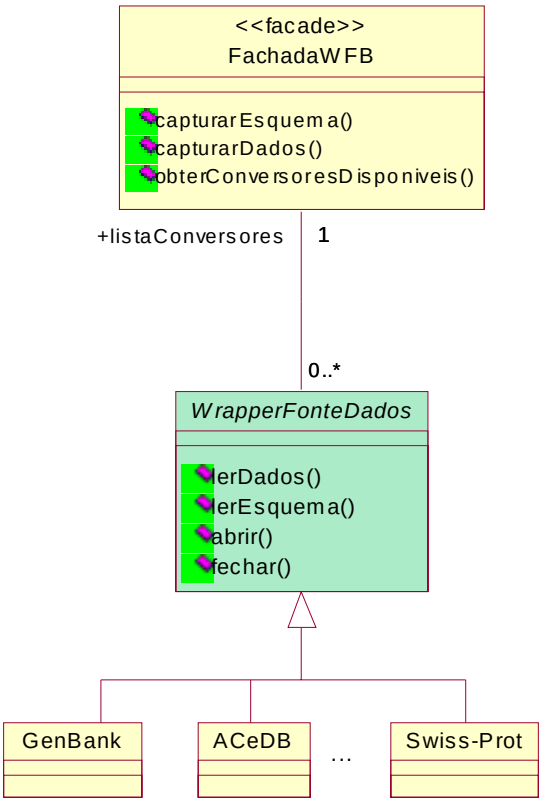

Figura 6 - Diagrama de classes do módulo WrapperFonteBiologia

O módulo *WrapperFonteBiologia* (Figura 6) é composto pelas seguintes classes:

- FachadaWFB, classe de interface entre os módulos do *framework* e os conversores de dados de biologia. Existirão vários conversores de dados instanciados na arquitetura, um para cada fonte de dados. Assim, o relacionamento entre a FachadaWFB e os Conversores é do tipo um-para-muitos.

- WrapperFonteDados, que representa a implementação de cada *wrapper*. Um *wrapper* deverá conter duas funcionalidades distintas, a primeira, de captura do esquema da fonte de dados de biologia e, a segunda, de captura dos dados daquela fonte. O WrapperFonteDados é um ponto de flexibilização da arquitetura.

3.2.3.4 *Drivers* para Aplicações de Biologia

O módulo *DriverAplicaçãoBiologia* (Figura 7) é composto pelas seguintes classes:

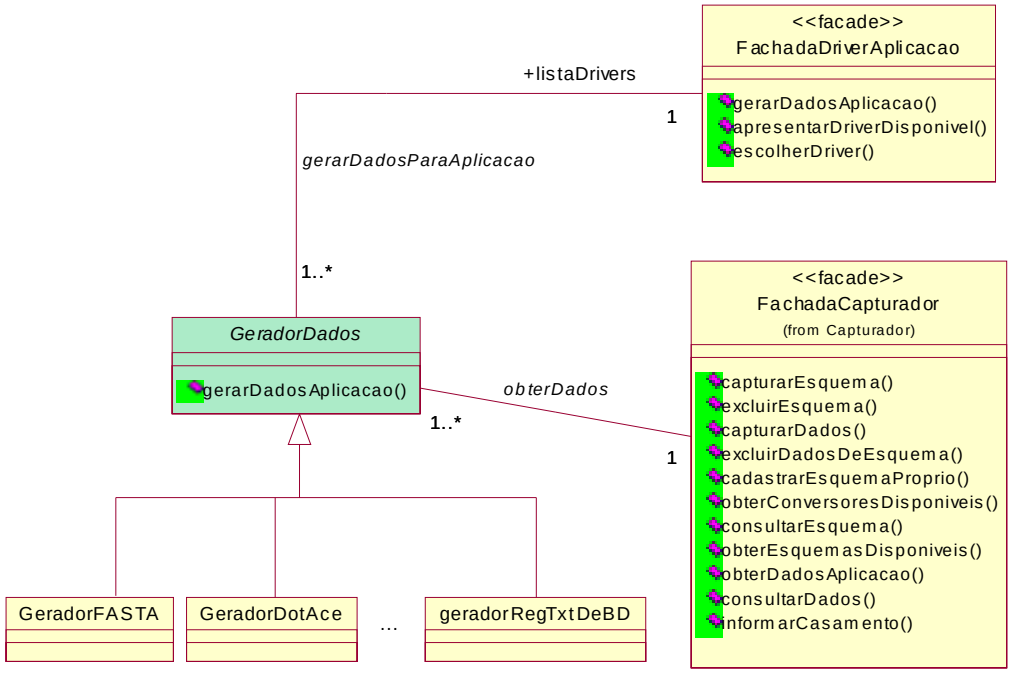

Figura 7 - Diagrama de classes do módulo DriverAplicaçãoBiologia

- *FachadaDriverAplicação*, classe de interface entre os módulos do *framework* e os *drivers* que geram dados para aplicações de biologia. Existirão vários *drivers* de aplicação na arquitetura, um para cada aplicativo a ser utilizado. Assim, o relacionamento entre a *FachadaDriverAplicação* e os *drivers* é do tipo um-para-muitos.

- GeradorDados, que representa a implementação de cada *driver*. O *driver* é um ponto de flexibilização da arquitetura. Por exemplo, um *driver* pode gerar dados no formato texto, segundo a sintaxe utilizada no GenBank ou no Swiss-Prot ou mesmo em formato FASTA (específico dos algoritmos da família Blast), a serem utilizados na execução de algoritmos que operem sobre eles. Além disso, um *driver* pode ser a implementação de uma interface com um sistema disponível via *Web*. Este *driver* pode enviar os dados disponíveis nos repositórios do *framework*

para que este sistema os processe. O driver pode ser também um serviço de dados, permitindo que uma aplicação se conecte ao *framework* e receba os dados ali armazenados.

3.2.4 Modelagem Dinâmica: Diagramas de Sequência das Funcionalidades da Arquitetura

Esta seção irá detalhar as macro funcionalidades do *framework*, apresentando a interação existente entre as classes. As macro-funcionalidades do *framework* são:

- Capturar esquema;
- Casar o esquema da fonte de dados com o modelo da biologia;
- Criar esquema próprio;
- Instanciar dados de um determinado esquema (capturar dados);
- Gerar dados para uma dada aplicação;
- Executar método do modelo da biologia.

3.2.4.1 Capturar esquema

A Figura 8 ilustra o diagrama de sequência da execução da função “Capturar esquema”. Inicialmente são apresentados para o usuário os conversores (*wrappers*) disponíveis. O usuário então seleciona um *wrapper* para ativação da captura de esquema que é processada pela *FachadaAdm*, que é repassada para as demais. A classe *FachadaWFB* ativa então o *wrapper* correspondente que retorna o esquema da fonte de dados, já em XML Schema, e que está em conformidade com o modelo da biologia existente na arquitetura.

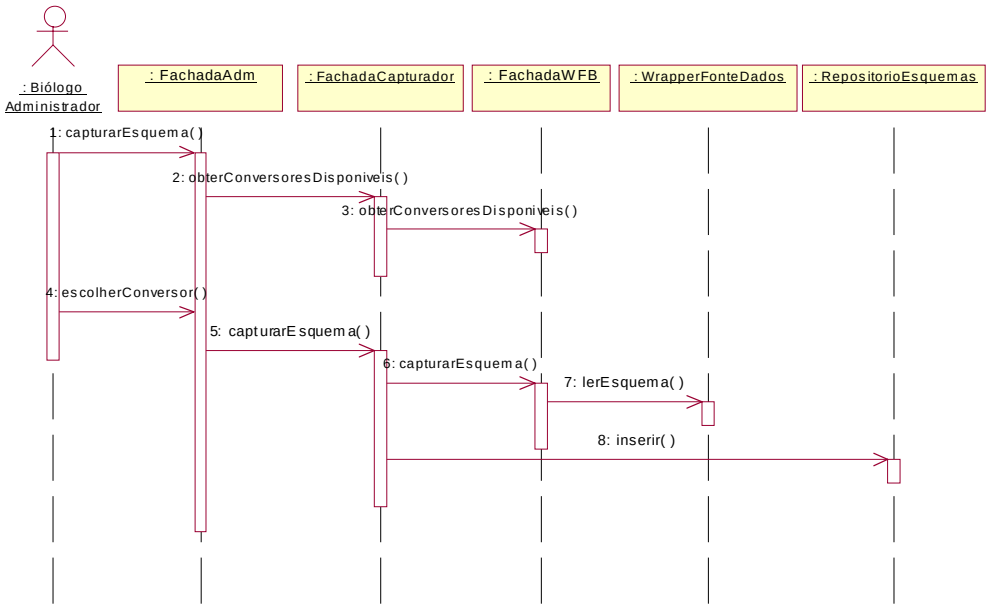

Figura 8 - Diagrama de sequência da funcionalidade "Capturar esquema existente"

3.2.4.2 Casar o esquema da fonte de dados com o modelo da biologia

Na Figura 9 é apresentado o diagrama de sequência da funcionalidade “Casar o esquema da fonte de dados com o modelo da biologia”. Inicialmente são consultados o modelo de dados da biologia e o esquema da fonte de dados que se deseja casar com o modelo. Se a fonte já estiver cadastrada, o casamento é anotado no esquema, caso contrário, o usuário poderá efetuar transformações no modelo da biologia de forma a que o casamento ocorra. As transformações possíveis são: inclusão / exclusão de objetos, inclusão / exclusão de associações entre objetos, criação / eliminação de sinônimos e regras de conversão. Cabe ressaltar que a manutenção criteriosa do modelo da biologia fornece elementos para a geração de uma ontologia que é efetivamente utilizada nas fontes de dados de biologia molecular (descrições dos objetos, propriedades e comportamentos, definição dos tipos de dados envolvidos, dos domínios permitidos para cada propriedade, descrição dos relacionamentos entre objetos e cardinalidades, definição de sinônimos e regras de conversão).

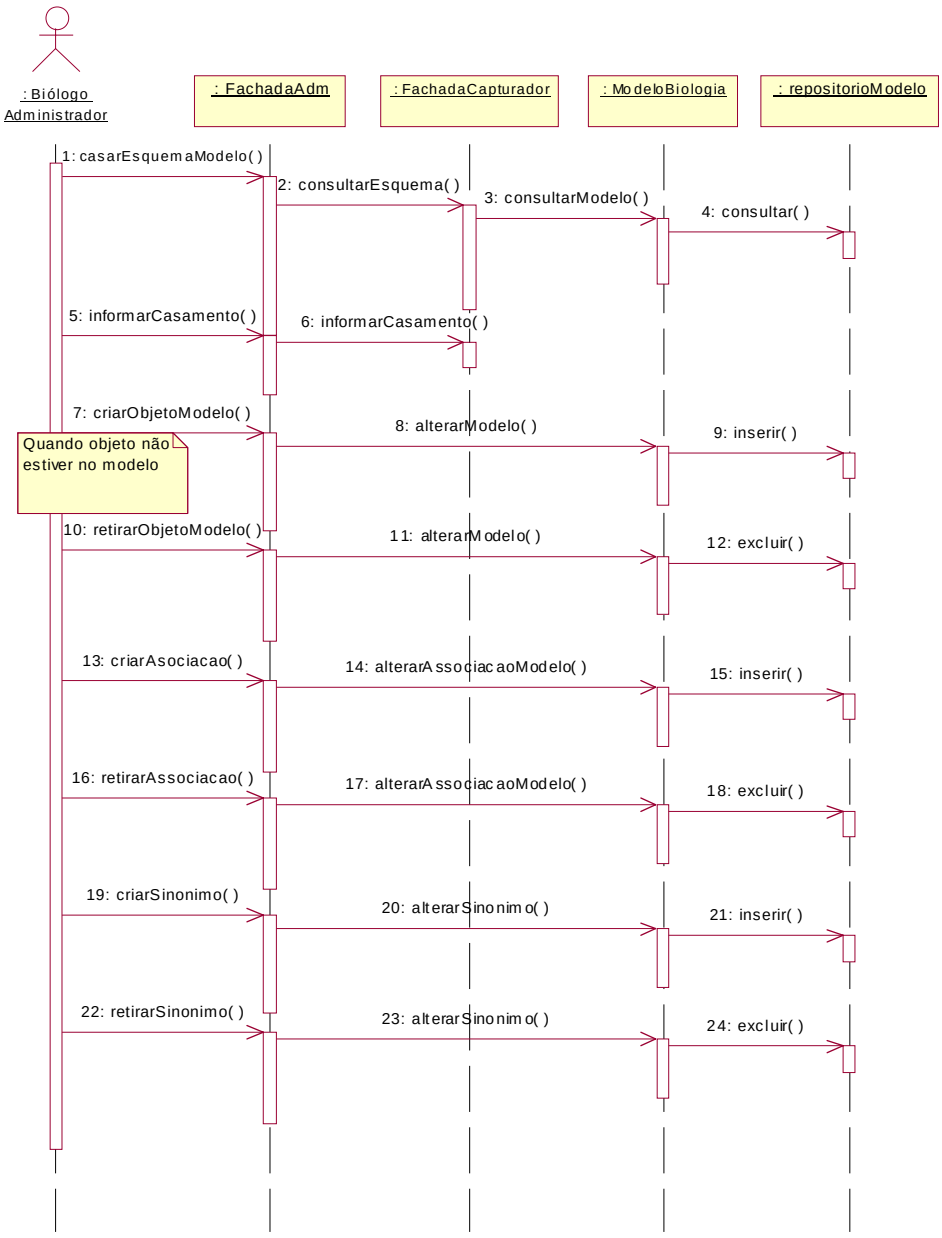

Figura 9 - Diagrama de sequência da funcionalidade "Casar o esquema da fonte de dados com o modelo da biologia"

3.2.4.3 Criar esquema próprio

A Figura 10 apresenta o diagrama de sequência da função “Criar esquema próprio”, que consulta o modelo de dados da biologia e permite que o usuário selecione objetos do modelo de forma a construir um esquema próprio. Ao término da seleção, o esquema construído é armazenado no Repositório.

3.2.4.4 Capturar dados

A Figura 11 ilustra o diagrama de sequência da execução da função “Capturar dados”. Inicialmente são apresentados para o usuário administrador os esquemas disponíveis. O usuário escolhe então o esquema a ser instanciado, que pode ser qualquer um daqueles armazenados no repositório. A ação de captura de dados é repassada ao *Capturador*, que atua como um mediador. O mediador pode agir, neste ponto, de diferentes formas:

- Verifica se o esquema a ser instanciado refere-se a uma única fonte de dados ainda não instanciada. Neste caso, o *wrapper* correspondente é acionado e os dados são armazenados no repositório.

- Verifica se o esquema a ser instanciado é composto de objetos que já constam do repositório. Neste caso, o repositório é acessado e o esquema é instanciado via SQL (operações de seleção e inserção).

- No caso geral, que se refere à instanciação de um esquema cujos objetos ainda não estão no repositório e que residem em fontes de dados diferentes, duas políticas distintas podem ser aplicadas. Uma trata de instanciar previamente as fontes de dados que participam do esquema (a ser instanciado) e o problema recai nos casos anteriores. A outra trata de:

1. acionar os wrappers correspondentes às fontes de dados que participam do esquema, para obtenção dos dados ainda não instanciados;

2. acionar o repositório para obtenção dos dados instanciados;

3. construir as instâncias do esquema a partir dos dados obtidos (podem ser necessárias conversões de dados e de tipos);

4. armazenar as instâncias no repositório.

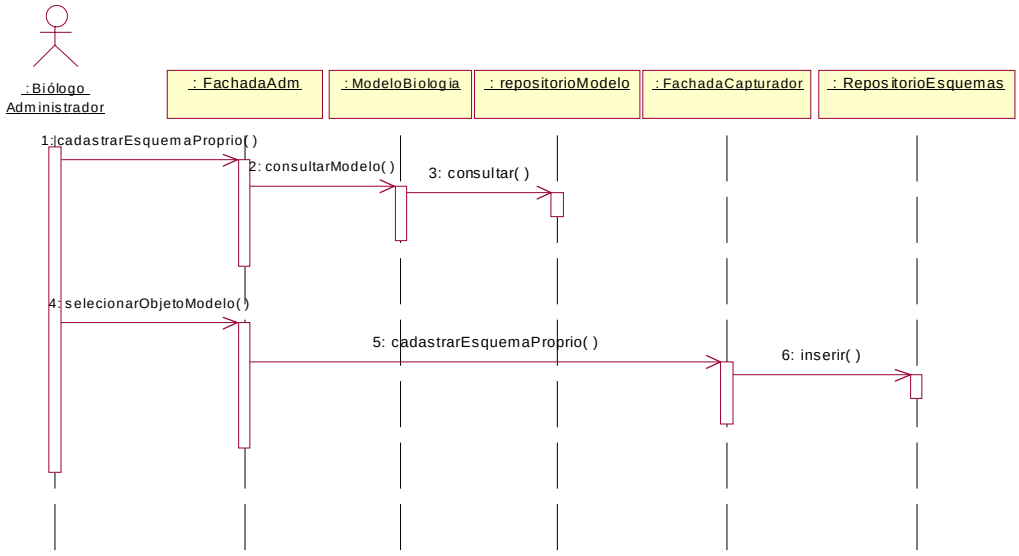

Figura 10 - Diagrama de sequência da funcionalidade "Criar esquema próprio"

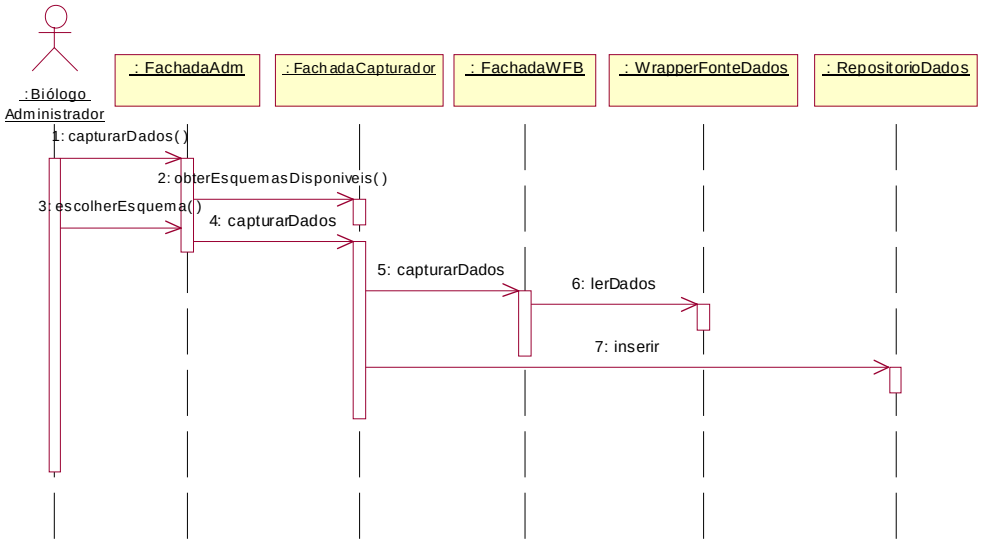

Figura 11 - Diagrama de sequência da funcionalidade "Capturar dados"

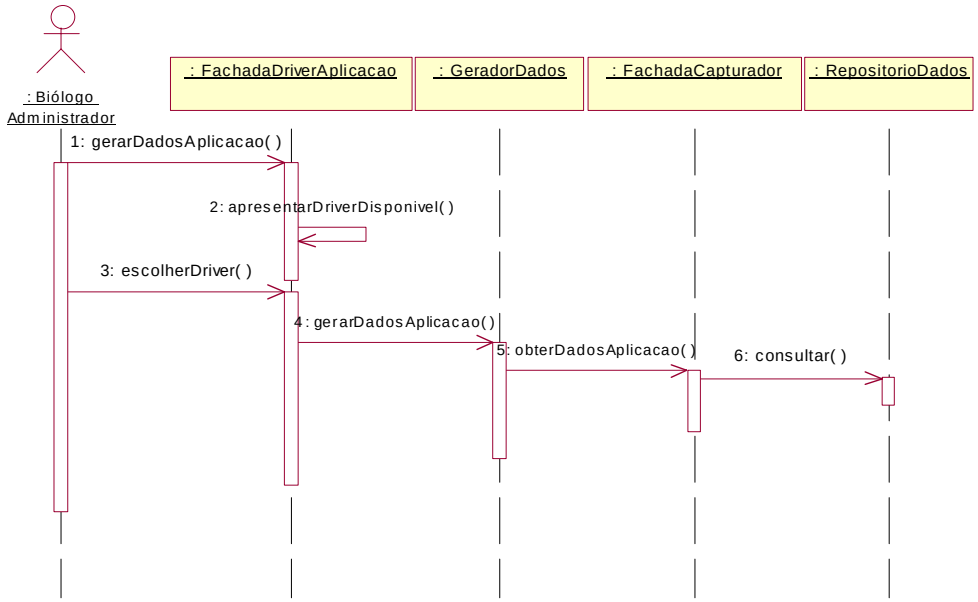

Figura 12 - Diagrama de sequência da funcionalidade "Gerar dados para aplicação"

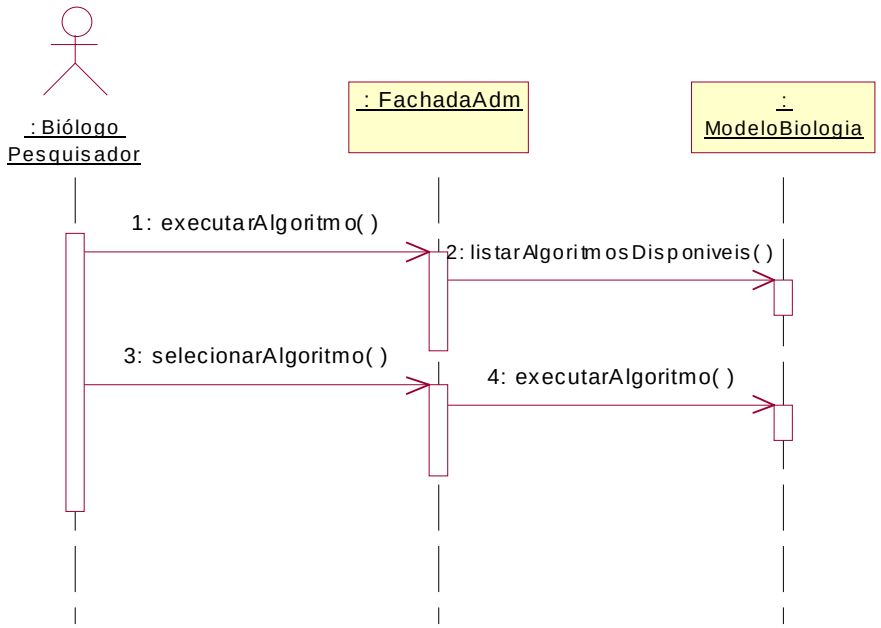

Figura 13 - Diagrama de sequência da funcionalidade "Executar método instanciado no modelo da biologia"

3.2.4.5 Gerar dados para uma dada aplicação

A Figura 12 apresenta o diagrama de sequência da funcionalidade “Gerar dados para aplicação”. Inicialmente são apresentados os *drivers* disponíveis na arquitetura. O usuário seleciona o driver que irá gerar um arquivo a ser utilizado na execução da aplicação. Os dados são obtidos do repositório e entregues para a aplicação externa que está em execução. É comum encontrar uma aplicação na *Web* que questione a localização e o nome do arquivo a ser processado, arquivo este que tem um formato pré-estabelecido. Essa funcionalidade propicia que tais aplicações operem sobre os dados disponíveis na arquitetura.

3.2.4.6 Executar método do modelo da biologia

A Figura 13 apresenta o diagrama de sequência da funcionalidade “Executar método do modelo da biologia”. Inicialmente são apresentados os métodos disponíveis no

modelo. O usuário seleciona o método a ser executado e essa informação é repassada à classe que representa o modelo da biologia, que ativa a sua execução. O método opera sobre as instâncias de dados da classe a ele associada.

3.3 Comentários Finais

Neste capítulo foi apresentada a tecnologia de *frameworks* e a especificação da arquitetura de integração proposta, através dos diagramas de classe e de sequência da UML. Todas as funcionalidades do *framework* foram aqui detalhadas. A especificação completa engloba ainda o modelo conceitual de informações biológicas, ou simplesmente modelo da biologia, que está apresentado no capítulo 5.

No próximo capítulo serão descritos os diversos projetos de integração existentes. Em seguida será feita uma comparação destes projetos com a proposta de integração descrita neste trabalho. Esta comparação está fundamentada em critérios baseados nas funcionalidades necessárias, na sua maioria mandatórias para a pesquisa. Ao final do capítulo é apresentado um quadro comparativo das ferramentas, baseado nos critérios estabelecidos.

4 Comparação entre as Arquiteturas de Integração

Este capítulo apresenta os principais trabalhos relacionados com a tese. Nele são apresentadas as arquiteturas de integração mais importantes na área de biologia molecular. Para cada arquitetura é apresentado o modelo de integração que ela utiliza, uma descrição sucinta de suas principais funcionalidades e são discutidas as vantagens e desvantagens de sua utilização. Em seguida são apresentados critérios para que se possa comparar as ferramentas, baseados em funcionalidades que as ferramentas de integração devem dispor. No final do capítulo é apresentado um quadro comparativo de conformidade entre as ferramentas e as funcionalidades necessárias. A comparação inclui a ferramenta proposta neste trabalho.

4.1 Apresentação dos Projetos de Integração de Informações da Biologia Molecular

Serão descritos, de forma sucinta, os seguintes projetos / produtos de integração na área de biologia molecular [Let99], e outros deles derivados ou a eles relacionados:

- SRS (*Sequence Retrieval System*) [SRS02a, SRS02b]
- OPM (*Object-Protocol Model*) [OPM02a, OPM02b, CM95a, CM95b, CM96a, CM96b, CM96c, CKM+96, KSC+96, MR95, Mar95, CKM+95]
- CPL/Kleisli (*Collection Programming Language*) [CPL02, Penn02, DCB02, BCD+98, BDH+95, DOB95, K202a, K202b, DCB01]
- IGD (*Integrated Genomic Databases*) [IGD02a, IGD02b, Rit94]
- TAMBIS (*Transparent Access to Multiple Bioinformatics Information Sources*) [TAMBIS02a, TAMBIS02b, TAMBIS02c, TAMBIS02d, SBS02, IMG02, BBB+98, BGB+99, PSB+99, BSN+99, SBB00, SGB+01].

4.1.1 SRS

O sistema SRS e trabalhos integra diversas fontes de dados textuais contendo informações biológicas através da implementação de *links* entre objetos que compõem estas fontes. A ferramenta foi desenvolvida originalmente no *European Molecular Biology Laboratory* (EMBL) a partir de 1989 e atualmente vem sendo atualizada e mantida no *European Bioinformatics Institute* (EBI). Em 1999 já integrava dados de aproximadamente 250 (duzentas e cinquenta) fontes de dados e contava com uma infraestrutura de mais de 35 (trinta e cinco) *sites* espalhados por todo o mundo, com cerca de 10.000 (dez mil) acessos diários. A lista completa das fontes de dados que estão contempladas no sistema está em http://srs.ebi.ac.uk/srs6bin/cgi-bin/wgetz?-page+databanks+-newId. A ferramenta é, sem dúvida, a mais popular entre os pesquisadores da área.

Os *links* são operadores que permitem a seleção de registros de uma fonte de dados e os associam a registros de uma outra fonte. Assim, o operador de ligação pode ser associado com o operador de junção da álgebra relacional, sendo que este é utilizado para combinar tuplas de relações em uma única tupla, enquanto aquele faz a ligação de registros de uma fonte de dados com registros de outra.

O sistema tem ainda um mecanismo de indexação de campos de registros. Os campos são analisados de forma a compor um índice de acordo com as suas ocorrências. Os índices são utilizados para atender de forma eficiente às consultas.

O sistema apresenta uma interface de acesso a diversas fontes de dados de biologia molecular, que são escolhidas pelo usuário, para formulação de consultas, de acordo com palavras-chave, sendo admitidos operadores lógicos. Na resposta, são apresentados os registros que atendem à consulta feita.

O sistema SRS permite também a formulação de consultas através de uma linguagem própria. A sintaxe da linguagem é descrita em camadas. Inicialmente são descritos os registros de um banco, em seguida as estruturas de dados de cada registro e finalmente os ítens de dados que compõem as estruturas. Assim, pode-se especificar que bancos procurar e sobre que atributos efetuar as consultas.

Os comandos da linguagem podem ser embutidos na linguagem C, através de uma API especialmente desenvolvida, tornando a ferramenta bastante útil pois, desta forma, podem ser feitas consultas que envolvem mais de uma fonte de dados e o uso de conectores lógicos.

O sistema permite ainda o uso de algumas poucas aplicações para análise de dados (por exemplo, para comparação de sequências via BLAST, para construção de múltiplos alinhamentos via ClustalW), todas elas referentes aos objetos: sequência de nucleotídeos e sequências de aminoácidos.

A ferramenta não fornece transparência de esquema e nem de localização, obrigando os usuários a formular consultas sobre atributos de uma dada fonte de dados. O sistema SRS não permite que sejam feitas consultas às fontes de dados utilizando o operador de junção. O sistema disponibiliza um reduzido conjunto de aplicativos de suporte à pesquisa.

4.1.2 OPM

O sistema integra dados armazenados em bancos de dados relacionais (Sybase e Oracle) e em arquivos texto semi-estruturados, como o GenBank, fornecendo aos usuários uma interface orientada a objetos que facilita a compreensão do objeto biológico. A ferramenta foi desenvolvida no *Lawrence Berkeley National Laboratory* [LBNL02] a partir de 1992. A importância da ferramenta deve-se à sua utilização no projeto do genoma humano, pelo *Genome Data Base* (GDB) mantido pela *Johns Hopkins School of Medicine* [JHU02], e *pelo Primary Database* mantido pelo *German Human Genome Resource Center* (RZPD [RZPD02]).

O sistema é dotado de ferramenta de consulta especial (OQL com interface gráfica) onde, através da visualização do objeto, é permitido aos usuários informar valores de atributos dos objetos a serem selecionados.

O sistema conta ainda com mecanismo de construção de visões sobre outros bancos de dados, que não foram construídos através da própria ferramenta, e que, posteriormente, incorporou também as fontes de dados que estão em arquivos texto semi-estruturados. Atualmente é possível formular consultas também ao GSDB, que é mantido pelo *National Center of Genome Resources* (NCGR) e ao GenBank.

Em 1996 foi desenvolvido o módulo *OPM Multidatabase Tools* para consultas a múltiplas fontes de dados, via esquemas nativos e visões sobre outras fontes de dados. Estas ferramentas foram aplicadas para construir o que o laboratório denomina de federação de bancos de dados de biologia molecular, que inclui o GDB, o GSDB e o GenBank. O módulo permite a formulação de consultas ao ambiente distribuído formado pelas bases que o compõem. Cada base possui esquema próprio, sendo que a formulação de consultas ao ambiente distribuído se utiliza de links entre classes das diferentes bases de forma a construir o plano de acesso.

O sistema propicia transparência de localização e dispõe de ferramentas capazes de lidar com a evolução dos esquemas locais através da manutenção de visões. As consultas em OPM são elaboradas através de uma linguagem semelhante à OQL.

OPM opera sobre um conjunto limitado de fontes de dados. A ferramenta não disponibiliza uma interface que permita a execução de aplicativos, se limitando a buscar e agrupar os dados. As aplicações devem ser executadas à parte, sobre o conjunto de dados resultante das consultas, e devem sofrer alteração de formato para aquele específico da ferramenta que se deseja executar.

4.1.3 CPL/Kleisli

O sistema CPL/ Kleisli é resultante de um projeto acadêmico que consiste de um sistema de consultas a um *multidatabase*. Os resultados do projeto contribuíram fortemente para a tecnologia de bancos de dados, especialmente para o estudo de linguagens de consulta a bancos de dados e para a construção de mediadores.

O sistema CPL/Kleisli foi desenvolvido na *University of Pennsylvania*, *no Institute for Research in Cognitives Sciences* (IRCS), para atender ao desafio lançado, em 1993, pelo Departamento de Energia americano (DOE) à comunidade científica [DOE02a, DOE02b, DOE02c]. O desafio (que foi vencido pelo sistema) consistia na formulação de uma série de consultas, consideradas impossíveis de serem respondidas, às fontes de dados de biologia molecular (distribuídas e heterogêneas), para atender à necessidades do projeto genoma humano. O sistema tem sido atualmente aplicado na integração de fontes de dados de patentes de proteínas [CWZ98].

O sistema CPL/Kleisli é baseado em uma poderosa linguagem de consulta (CPL), que modela tipos de dados complexos a partir de elementos simples (registros ou estruturas) mas com diferentes modos de organização, como:

- lista (formado por elementos simples ou compostos onde a ordem dos elementos é relevante);
- sacola (formado por elementos simples ou compostos onde a ordem é irrelevante);
- registro (formado por elementos rotulados, simples ou compostos);
- variante (formado por elementos rotulados, cujas instâncias são formadas por um destes elementos; os elementos podem ser simples ou compostos).

A linguagem permite a formulação de consultas utilizando os tipos complexos de dados, que são bastante utilizados na representação de informações da biologia molecular. O sistema suporta a formulação de consultas *ad-hoc* sobre bases de dados que estão distribuídas e são heterogêneas e tem sido utilizado para integração de fontes de dados autônomas, via mediadores (que implementam visões de usuários), com objetivo de formulação de consultas.

O CPL/Kleisli tem primitivas que implementam chamadas ao *wrapper* do banco de dados a ser acessado, no sentido de executar operações do tipo seleção, projeção e junção. O acesso a fontes de dados textuais é bem mais difícil de ser realizado uma vez que não há uma linguagem para tal. Assim, foi criada uma sintaxe para extração

de dados através de sucessivos acessos a registros de forma a implementar as operações.

O sistema soluciona o problema de consultas às fontes de dados através de uma única linguagem. As fontes de dados envolvidas podem residir em arquivos texto, em SGBD's (Oracle ou Sybase), nos arquivos indexados do sistema SRS, e em outras, bastando construir o *wrapper* adequado para a respectiva fonte.

De forma resumida, o sistema CPL/Kleisli oferece as seguintes funcionalidades:

- disponibiliza os tipos de dados necessários para a integração de recursos de dados distribuídos e heterogêneos da biologia molecular;
- provê uma interface uniforme para os recursos de dados heterogêneos;
- permite a formulação de consultas complexas ao ambiente distribuído e heterogêneo;
- provê a otimização de consultas distribuídas, incluindo paralelismo;
- provê tratamento uniforme para os algoritmos de análises empregados na biologia (por exemplo, BLAST);
- sua arquitetura permite a modularização dos *wrappers* para acesso aos sistemas distribuídos.

O projeto se ressente da lentidão do ambiente de rede (Internet) e não fornece tempos de resposta adequados. Porém isto é função do recurso que se deseja acessar e da forma de comunicação utilizada. As consultas feitas em ambientes distintos formados por bancos de dados relacionais ligados por rede local puderam se beneficiar do paralelismo característico do ambiente heterogêneo e apresentaram melhoras de desempenho significativas. A arquitetura tem dificuldades para lidar com a atualização dos esquemas das fontes de dados externas.

A Universidade da Pensilvânia tem até hoje importante participação em pesquisas para integração de dados de biologia molecular e também em diversos projetos genoma. Este fato levou o grupo de pesquisas da Universidade a criar um Centro de Pesquisas específico para a Bioinformática (Penn Center for BioInformatics - PCBI

[Penn02]) e a desenvolver outros projetos de integração com finalidades diferentes, como:

- K2 [K202a, K202b, DCB01]

  Trata-se da mais recente versão do sistema CPL/Kleisli, onde foram feitas algumas transformações, como a alteração da linguagem de consulta (CPL) por uma linguagem de consulta padronizada para acesso a dados (OQL), além da inclusão do tipo de dados "dicionário", que procura representar uma fonte de dados acessível via *Web*, que pode ser consultada a partir de um formulário com campos previamente definidos.

- GUS [DCB01]

  Consiste de um *datawarehouse* formado a partir dos dados extraídos de um conjunto limitado de fontes de dados, via *scripts* escritos em linguagem Perl. Contém sequências de nucleotídeos e de proteínas provenientes do GenBank e do Swiss-Prot e também de registros do dbEST [dbEST02] (contém informações de regiões contendo genes). O objetivo de armazenar dados destas três fontes é o de organizar no *data warehouse* informações disponíveis sobre o dogma central da biologia (DNA -> RNA -> Proteína), ou seja, dispõe de sequências e respectivas anotações (genes), RNA derivado destes genes e proteínas obtidas a partir deste RNA. A ferramenta GUS foi criada para resolver o problema de performance existente no sistema CPL/Kleisli e K2. O *data warehouse* foi construído utilizando o SGBD Oracle. O sistema prevê a atualização dos esquemas das fontes de dados externas através da atualização de visões sobre o esquema destas fontes.

CPL/Kleisli e K2 não dispõem de um esquema global que explique, para os usuários, quais objetos e relacionamentos estão envolvidos. As consultas contam com poderosa linguagem de consulta, posteriormente abandonada por não ser padronizada, se mostrou de difícil uso mesmo para especialistas em bancos de dados, mais acostumados com SQL.

O *data warehouse* criado no GUS é de uso específico e portanto não contempla as diversas fontes de dados biológicas, que são necessárias para pesquisas mais abrangentes. O sistema não disponibiliza uma interface que permita a execução de aplicativos, se limitando a buscar e agrupar os dados. As aplicações devem ser executadas à parte, sobre o conjunto de dados resultante das consultas, que devem sofrer alteração de formato para aquele específico da ferramenta que se deseja executar.

4.1.4 IGD

O projeto IGD é um exemplo de integração bem sucedida e está baseado na materialização dos dados do projeto genoma humano em um SGBD comercial (Sybase).

Em termos de integração de dados o IGD teve como objetivo inicial a criação de um banco fisicamente materializado (*data warehouse*) integrando as informações disponíveis nas diversas bases de dados (mais de vinte) que compõem o genoma humano, complementados com outras informações vindas de bases de dados de animais mamíferos (como por exemplo, a do genoma do rato). No entanto, o esquema implementado é genérico o suficiente para suportar dados de outros organismos.

As diretivas do projeto IGD tiveram como meta:

- integrar os dados do genoma humano e materializar estas informações segundo um modelo de dados comum;
- implementar o banco de dados em um ambiente cliente-servidor;
- disponibilizar nos clientes um gerenciador de banco de dados extensível e flexível;
- integrar dados e programas através de *links* entre objetos e métodos nos clientes;
- disponibilizar formas padronizadas de acesso aos dados;
- minimizar o uso de *software* proprietário na implementação.

A estratégia de implementação do IGD contempla as ferramentas / módulos de software descritos a seguir:

- No *front-end* a utilização do sistema AceDB [ACEDB02], que é um sistema orientado a objetos, não proprietário, construído especialmente para aplicações da área de biologia, e que possui linguagem própria para a definição e manipulação dos dados. O usuário tem a compreensão completa do objeto biológico. O AceDB usa o modelo de dados semi-estruturado e o armazenamento dos dados é feito em formato texto. A ferramenta disponibiliza para os usuários uma poderosa e adequada interface gráfica que muito auxilia os pesquisadores da área pois permite a exibição do mapa do genoma do organismo pesquisado e, através de cliques do *mouse* sobre o mapa, exibe informações detalhadas sobre a região em estudo.

- No *back-end* o uso do SGBD relacional Sybase.

Como o ACEDB é orientado a objetos e o Sybase é relacional, foi introduzido um nível de esquema intermediário que contempla o modelo de entidades e relacionamentos estendido (EER) [MS93a]. Foram ainda desenvolvidos aplicativos de conversão automática de esquemas entre o ACEDB e o EER, e foram empregadas as ferramentas ERDRAW [SM93]e SDTTools [MWF93] para para tradução de esquemas entre o EER e o esquema implementado no banco relacional.

Os usuários formulam as consultas à base de objetos através do uso de uma ferramenta gráfica, de alto nível, denominada *Concise Object Query Language* (COQL) [MS93b]. As consultas em COQL são automaticamente traduzidas para SQL.

Existem ainda ferramentas para gerar tradutores entre os dados formatados em Sybase para dados no formato ACEDB.

O sistema dispõe de um esquema parcial do modelo de dados da biologia molecular. As consultas contam com uma poderosa linguagem de consulta, porém não

padronizada. O sistema disponibiliza um conjunto limitado de aplicativos. O *data warehouse* criado é de uso específico e portanto não contempla as diversas fontes de dados biológicas, que são necessárias para pesquisas mais abrangentes, e não dispõe de mecanismo apropriado para lidar com a atualização dos esquemas das fontes de dados envolvidas na integração e nem com as atualizações das instâncias de dados.

4.1.5 TAMBIS

O projeto TAMBIS desenvolvido na Universidade de Manchester, consiste de um sistema de recuperação de informações biológicas que é orientado por uma ontologia construída para o domínio da biologia molecular e bioinformática. A ontologia é baseada em descrições lógicas de forma a permitir inferências e é gerenciada por um servidor de termos. A ontologia é utilizada para orientar a formulação de consultas pelos usuários, e para representar o esquema global, sobre o qual são elaboradas as consultas.

O sistema tem as seguintes características:

- Provê total transparência para os usuários na formulação de consultas, ou seja, não é exigido dos usuários o conhecimento dos esquemas e da localização das fontes de dados;
- Provê acesso de leitura às fontes de dados;
- Permite a formulação de consultas complexas sobre múltiplas fontes de dados heterogêneas. Os usuários dispõem de ferramenta gráfica de forma a facilitar a formulação de consultas por usuários não especialistas. A ferramenta transforma a consulta para a linguagem CPL;
- Dispõe de uma ferramenta visual para consulta aos dados;
- Dispõe de uma ontologia composta de mais de mil e oitocentos conceitos biológicos e respectivos relacionamentos e que é capaz de inferir outros. A ontologia engloba conceitos referentes a ácidos nucleicos, proteínas, estruturas de proteínas, funções biológicas e processos biológicos.

O sistema foi construído em camadas: a primeira é a conceitual, a segunda trata do mapeamento entre as camadas conceitual e física e a terceira, a camada física.

A camada conceitual (declarativa) compreende a ontologia, que representa o esquema global. As consultas são formuladas neste nível, utilizando a mesma linguagem de descrição dos conceitos e escondendo as fontes de dados dos usuários.

A segunda contém mapeamentos entre a camada conceitual e os esquemas das fontes de dados envolvidas na integração.

A terceira (procedural) compreende os *wrappers* para acesso às fontes de dados. Nesta última camada é utilizado o sistema CPL/Kleisli para acesso às fontes de dados e para descrição dos esquemas locais. Nesta camada são feitas também transformações de tipos entre as funções de acesso (*wrappers*).

O sistema TAMBIS, como já mencionado anteriormente, está baseado em uma ontologia que utiliza descrições lógicas para representação do esquema conceitual global. No entanto, descrições lógicas apresentam limitações para lidar com restrições semânticas como cardinalidade de relacionamentos e domínios de valores. Assim, um mecanismo para representar estas restrições foi implementado à parte, dentro da aplicação.

O sistema também se ressente da lentidão inerente ao sistema CPL/Kleisli, que é utilizado no acesso às fontes de dados, que estão distribuídas. Assim, uma outra ferramenta, baseada em um *data warehouse*, foi criada no sentido de atender ao problema de performance, denominada GIMS (*Genome Information Management System*) [GIMS02a, GIMS02b, GIMS02c, GIMS02d, SBS02, IMG02, MPH99, EBP+99, PKH+00, CPW+01].

O sistema GIMS foi disponibilizado em 2000 utilizando um SGBD orientado a objetos (Poet) que dispõe da linguagem de consulta OQL. O esquema global utilizado na ferramenta foi construído a partir do entendimento dos esquemas de um conjunto

de fontes de dados de biologia molecular. Este esquema global é bastante completo em termos das classes descritas mas não casa com os esquemas das fontes de dados (dificultando a sua compreensão) e não contempla os comportamentos dos objetos envolvidos, que se constituem, de fato, nos aplicativos necessários à pesquisa na área. O *data warehouse* criado não dispõe de mecanismo apropriado para lidar com a atualização dos esquemas das fontes de dados envolvidas na integração e nem com as atualizações das instâncias de dados.

4.2 Comparação entre as ferramentas de suporte à pesquisa em biologia molecular

Os critérios para comparação entre as ferramentas de suporte à biologia molecular são descritos a seguir.

1. Capacidade de formulação de consultas complexas. Por consulta complexa entenda-se uma consulta feita a diversas fontes de dados, com operadores de junção nos atributos de relacionamento entre entidades que pertençam a diferentes fontes de dados, e com operadores lógicos AND, OR e NOT;

2. Capacidade de formulação de consultas, via interface *Web*;

3. Possibilidade de formulação de consultas sofisticadas, porém de simples utilização, de forma a permitir o uso por pessoas não especialistas em informática;

4. Facilidade de acesso a todas as fonte de dados existentes;

5. Capacidade de lidar com o ambiente heterogêneo e fazer as transformações necessárias;

6. Transparência de localização física e dos mecanismos de acesso às fontes de dados utilizadas em uma dada consulta;

7. Não obrigar o conhecimento dos esquemas das diversas fontes de dados de biologia molecular;

8. Capacidade de tratar a atualização dos esquemas das fontes de dados;

9. Adoção de um esquema coerente com os esquemas utilizados pelas fontes de dados;

10. Possibilidade de instanciação de uma fonte de dados específica e adequada para uma dada pesquisa;

11. Detecção de atualização dos esquemas das fontes de dados;

12. Capacidade de tratar a atualização dos dados;

13. Execução dos aplicativos disponíveis para a área;

14. Facilidades para o entendimento dos objetos biológicos através de um conjunto de definições (ontologia) efetivamente utilizada na pesquisa.

4.3 Quadro Comparativo

De forma a permitir a comparação entre as ferramentas de integração existentes e a proposta na tese, é apresentado a seguir um quadro contendo, para cada ferramenta e para cada critério, uma avaliação indicando se a ferramenta atende ou não àquele critério. Quando o critério for possível de ser atendido, uma gradação através do uso de estrelas (de uma a quatro) é utilizada para diferenciar o grau de dificuldade de adequação ao critério (quanto menos estrelas, menos adequado é). Quando um determinado critério não fizer sentido na avaliação de uma determinada ferramenta, será utilizada a representação ‘----‘. Cabe observar que os graus representam uma

avaliação subjetiva, baseada na experiência e no conhecimento pessoal. O quadro comparativo é apresentado a seguir.

Ferramenta / Critério	SRS	OPM	CPL/Kleisli	K2	GUS	IGD	TAMBIS	GIMS	Bio-AXS
1	**	****	****	****	****	****	****	****	****
2	Sim	Não	Não	Não	Não	Sim	Não	Não	Sim
3	***	**	*	**	**	***	*	**	****
4	**	**	***	***	----	----	***	***	****
5	Não	***	****	****	**	**	****	****	****
6	Não	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim
7	Não	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim
8	*	***	*	*	***	*	*	****	****
9	----	****	****	****	****	****	****	***	****
10	Não	Não	Não	Não	Não	Não	Não	Não	Sim
11	*	*	*	*	*	*	*	*	****
12	**	****	****	****	**	**	****	**	***
13	*	*	*	*	*	*	*	*	****
14	----	***	***	***	*	*	****	*	****

Quadro comparativo entre as arquiteturas de integração.

No primeiro critério, a ferramenta SRS teve uma avaliação menor porque não implementa o operador de junção.

O segundo critério indica se a ferramenta disponibiliza ou não uma interface Web.

O critério de formulação de consultas sofisticadas, feitas de forma amigável, leva em consideração se a ferramenta disponibiliza uma interface do tipo QBE (*Query By Example*) ou mesmo formulários e também se a linguagem de consultas é do tipo SQL padrão ou se é necessário o aprendizado de uma linguagem específica.

No quarto critério, as ferramentas são avaliadas de acordo com a capacidade de incorporação de novas fontes de dados. Neste critério, a avaliação é bastante subjetiva em virtude de não se conhecer em detalhe as ferramentas disponíveis no ambiente de desenvolvimento.

O quinto critério, também de avaliação bastante subjetiva, considera a adequação da ferramenta ao ambiente heterogêneo. Neste caso, as ferramentas que integram um reduzido número de fontes de dados foram mal avaliadas pois tratam a

heterogeneidade de forma limitada em relação àquelas que lidam com o ambiente completo. O sistema SRS que implementa *links* não atende ao critério.

O sexto e o sétimo critérios indicam se a ferramenta que está sendo avaliada atende ou não ao critério que está sendo avaliado. Neste caso, apenas o sistema SRS não "esconde" dos usuários a fonte de dados a ser acessada e o respectivo esquema.
O oitavo critério foi avaliado de acordo com as informações disponíveis. Algumas ferramentas foram avaliadas com uma estrela quando não foi possível saber se implementam a funcionalidade específica, de forma mais automática do que simplesmente reconstruir os *wrappers* correspondentes.

No nono critério, o sistema SRS não adota um modelo de dados e portanto não faz sentido a sua avaliação. Já o sistema GIMS teve avaliação pouco inferior aos demais porque apresenta um modelo conceitual de informações biológicas próprio.

Quanto ao décimo critério, apenas a ferramenta proposta na tese (Bio-AXS) implementa a funcionalidade de instanciar fontes de dados específica para uma pesquisa.

A avaliação do critério de detecção da evolução dos esquemas das fontes locais foi bastante difícil. Algumas ferramentas foram avaliadas com uma estrela porque, na sua documentação, a funcionalidade não é tratada, o que evidencia, de certa forma, a pouca importância dada ao assunto.

Com relação a atualização dos dados, os sistemas que implementam consultas diretamente às fontes locais (*multidatabase*) foram melhor avaliadas do que os que implementam um *data warehouse*, ou mesmo o sistema SRS que contém *links* que também devem ser atualizados. Entre estas, a arquitetura proposta na tese foi melhor avaliada por dispor, na documentação, de esquemas próprio para a atualização das instâncias de dados.

No critério de execução dos aplicativos disponíveis para a área de biologia molecular, apenas a ferramenta proposta na tese possibilita a execução de aplicativos externos sobre os dados da arquitetura, de forma planejada para tal. Além disso, os aplicativos podem também ser incorporados à arquitetura (como métodos do modelo de dados) e disponibilizados para a execução pelos usuários.

O último critério, considera a operação da ferramenta de acordo com uma ontologia para a área. Aquelas que limitam o acesso às fontes de dados tiveram pior avaliação. As ferramentas OPM, CPL/Kleisli e K2 tratam a funcionalidade via registro das fontes de dados, porém, não de forma tão completa quanto TAMBIS e Bio-AXS. Não faz sentido avaliar o sistema SRS com relação ao critério.

4.4 Comentários Finais

Neste capítulo foram apresentadas as principais ferramentas de integração de informações da biologia molecular. Foi apresentada, de forma sucinta, as técnicas envolvidas em cada uma e foram discutidas as suas funcionalidades. Em seguida, foram listados critérios de comparação que atendem às funcionalidades necessárias a uma ferramenta de integração para a área de biologia molecular. Finalmente foi apresentado um quadro comparativo entre as ferramentas estudadas, que mostra a aderência da arquitetura proposta às funcionalidades exigidas pela pesquisa.

O capítulo seguinte discute os diferentes modelos de dados utilizados na biologia molecular. Propõe um modelo conceitual de informações biológicas e também um modelo lógico que é adequado para representação dos dados, que foram implementados na arquitetura. Finalmente são apresentadas e discutidas as principais ontologias existentes para a área e, novamente, é apresentada a proposta que é utilizada na arquitetura.

5 Modelos de Informações Biológicas

Neste capítulo são discutidos aspectos referentes ao modelo conceitual da biologia molecular, as suas classes e respectivos métodos, heranças, polimorfismos e relações entre elas, isto porque este trabalho propõe o uso de um modelo conceitual inicial, que é revisto e complementado à medida que a arquitetura toma conhecimento de (se relaciona com) novas fontes de dados utilizadas na área.

Aqui é também discutido que modelo lógico é adequado para melhor representar os dados de biologia molecular que contemple as suas características de alteração frequente de esquema e constante atualização.

Finalizando o capítulo, são apresentadas as diversas ontologias propostas para a área, já que o presente trabalho tem também como objetivo gerar uma ontologia acerca dos elementos de dados efetivamente utilizados nas fontes de dados existentes.

5.1 O Modelo Conceitual de Informações Biológicas

Existem tentativas localizadas no sentido de se definir o modelo conceitual das informações biológicas. Tais modelos, no entanto, ora são tratados de forma adequada em termos de detalhamento de propriedades e comportamentos mas são bastante específicos, isto é, tratam de um subconjunto reduzido e bem delimitado das informações, ora são modelos genéricos porém incompletos, que apenas exemplificam um modelo que é utilizado em uma dada ferramenta de integração.

No primeiro caso, encontram-se os padrões propostos pela OMG (Object Management Group) [OMG02]. A OMG tem modelos bastante detalhados para determinadas informações utilizadas pela pesquisa em biologia molecular. Um exemplo é a proposta de padrão para a representação de dados de expressão de genes e anotações, consideradas relevantes.

As especificações de padrões propostas pela OMG para dados biológicos (expressão gênica, análise de sequências, mapas genômicos) incorporam muitos detalhes do padrão CORBA no modelo proposto. Esta abordagem dificulta o entendimento da proposta adotada e torna pouco transparentes os conceitos biológicos. Além disso, mudanças no padrão CORBA irão afetar as definições apresentadas neste modelo.

A dificuldade na preparação de um modelo padrão para a biologia se dá pela falta de um consenso sobre os conceitos biológicos. Acrescentar aspectos de implementação (por exemplo CORBA) em uma proposta de padronização dificulta o entendimento do modelo além de impedir o entendimento, pelos biólogos, dos conceitos biológicos.

Os modelos da OMG tem como foco as anotações sobre o genoma, não sendo extensivos a todos conceitos biológicos relacionados com estas anotações. Por exemplo, os modelos de análises de sequências biomoleculares, de expressão gênica e de mapas genômicos não relacionam os dados apresentados com as características e funções protéicas . A definição de um modelo incompleto inviabiliza o acesso aos dados que são relacionados a conceitos que não estão incorporados nos modelos.

A definição de um modelo puramente biológico, ausente de aspectos de implementação, ajuda na sua transparência, independência e evolução. Além disso, os aspectos tecnológicos tais como: experimentos com *microarrays*, sequenciamento ou montagem de DNA, entre outros, podem "poluir" o modelo da biologia visto que eles não representam fenômenos biológicos. Por exemplo, fragmentos de DNA não representam um fenômeno biológico pois são resultado de um processo tecnológico durante o sequenciamento e montagem de um genoma. Adicionar aspectos de implementação ou tecnológico no modelo da biologia não agrega nenhum valor aos conceitos apresentados, dado que esses aspectos são mais voláteis do que os conceitos biológicos.

O segundo caso pode ser exemplificado pelo modelo da biologia utilizado no projeto GIMS. O modelo conceitual de informações biológicas que é proposto neste projeto é

bastante incompleto. Por exemplo, a modelagem das informações de genoma não inclui os elementos extra-cromossomais e nem as regiões de DNA sem função conhecida (pseudo-genes, regiões repetitivas, entre outras) (vide modelo da biologia proposto na seção seguinte). Nas informações de proteoma não são apresentadas as informações acerca da estrutura das proteínas.

É fundamental que uma proposta de modelo conceitual para a área seja a mais completa possível em termos dos objetos representados. Outro aspecto a considerar consiste na separação de conceitos biológicos daqueles estritamente tecnológicos. Novamente, o fragmento de cromossomo, que está representado em muitos modelos, tem cunho tecnológico pois é diretamente dependente da tecnologia de sequenciamento utilizada. Esta classe representa a divisão dos cromossomos em fragmentos (ver Figura 14) para possibilitar a leitura do DNA ali presente. Tal classe provavelmente não seria modelada se fosse tecnicamente possível realizar a leitura do cromossomo completo. Em termos biológicos, a classe que representaria de forma adequada uma parte do cromossomo seria, por exemplo, *região*, que poderia ser dividida em *região com informação biológica* e *região sem informação biológica* anotada.

A falta de um modelo conceitual de informações biológicas motivou a elaboração de um estudo específico nesse sentido, que contemplasse as informações biológicas armazenadas em diversas fontes de dados (no momento de elaboração do modelo), e também outras informações ainda ausentes das fontes de dados mas consideradas importantes como representação de fenômenos biológicos conhecidos. O estudo foi feito com base nas informações do dogma central da biologia. O modelo conceitual referente a estas informações é apresentado a seguir.

5.1.1 Modelo das Informações de Genoma

O modelo apresenta uma parte dos objetos do domínio da aplicação, especificamente as informações relativas ao dogma central da biologia. Outras informações referem-se, por exemplo, ao transcriptoma e metabolona. Este modelo deverá ser estendido de

forma a contemplar as informações biológicas, especialmente aquelas disponíveis nas fontes de dados de biologia.

No modelo apresentado na Figura 14 pode-se observar que o *Genoma* é formado por *Cromossomo* e que cada um pode conter *Região*. Uma *Região* pode estar presente em um *GrupoRegião*. Cada *GrupoRegião* pode ser uma *RegiãoComplexa* ou *FragmentoCromossomo*. Fragmentos são auto-relacionados, assim sabe-se que um *FragmentoCromossomo* segue-se ou é anterior a outro. *RegiãoComplexa* pode ser *Operon, Transposon* ou *Profago*. Uma *Região* pode estar relacionada a outras ou não. Uma *Região* pode ser uma *RegiãoInformativa* ou *RegiãoNãoInformativa*.

O *Genoma* é também formado por *ElementoExtraCromossomal*, que por sua vez pode ser *Plasmídeo* ou *DNA_Organelas*.

Uma *RegiãoNãoInformativa* pode se um *PseudoGene, RegiãoRepetitiva* ou *RegiãoDesconhecida*. A *RegiãoDesconhecida* é aquela considerada por alguns pesquisadores como "lixo" de DNA. A *RegiãoRepetitiva* por sua vez pode ser *RepetiçãoDireta, RepetiçãoInversa, Palíndromo* e *RegiãoBaixaComplexidade*.

A Figura 15 apresenta o modelo de *RegiãoInformativa*, ou seja, a *Região* que tem informação biológica relevante. Uma *RegiãoInformativa* pode ser *Transcrito (também TranscritoPrimário)* ou *RegiãoNãoTranscrita*. Esta pode ser *SequênciaRegulatória* ou *ElementoCromossomal*. A *SequênciaRegulatória* pode ser *Promotor* ou *Terminador*. O *ElementoCromossomal* pode ser *Centrômero, Telômero* ou *ORI*.

O *Transcrito* é composto de *Éxon, Íntron* e *UTR* e estes são *Componentes*, que estão auto-relacionados. A *UTR* pode ser *UTR5_l* (*UTR 5'*) ou *UTR3_l* (*UTR 3'*). A *UTR5_l* é composta de *Promotor_l* e *RBS*. A *UTR3_l* é composta de *Terminador_l* e *SítioPoliA*.

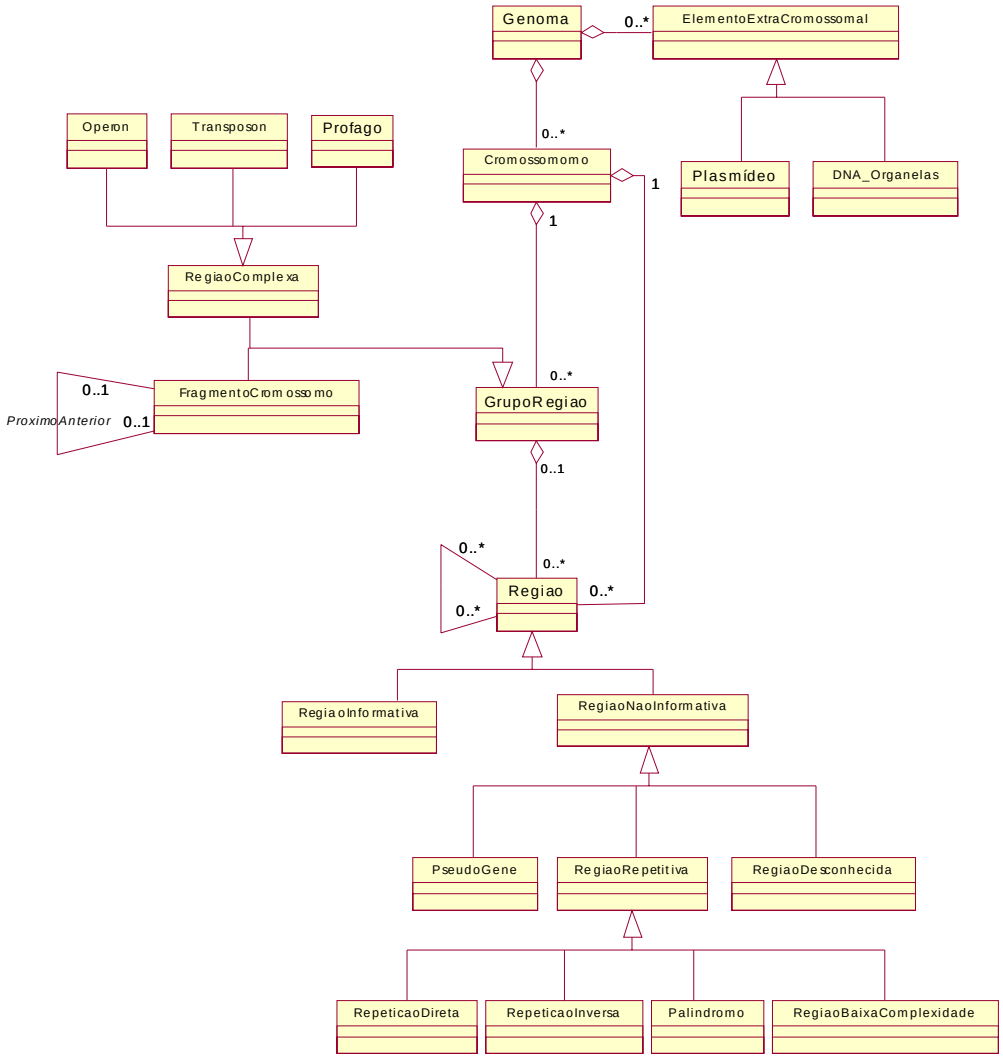

Figura 14 - Diagrama de classes de genoma

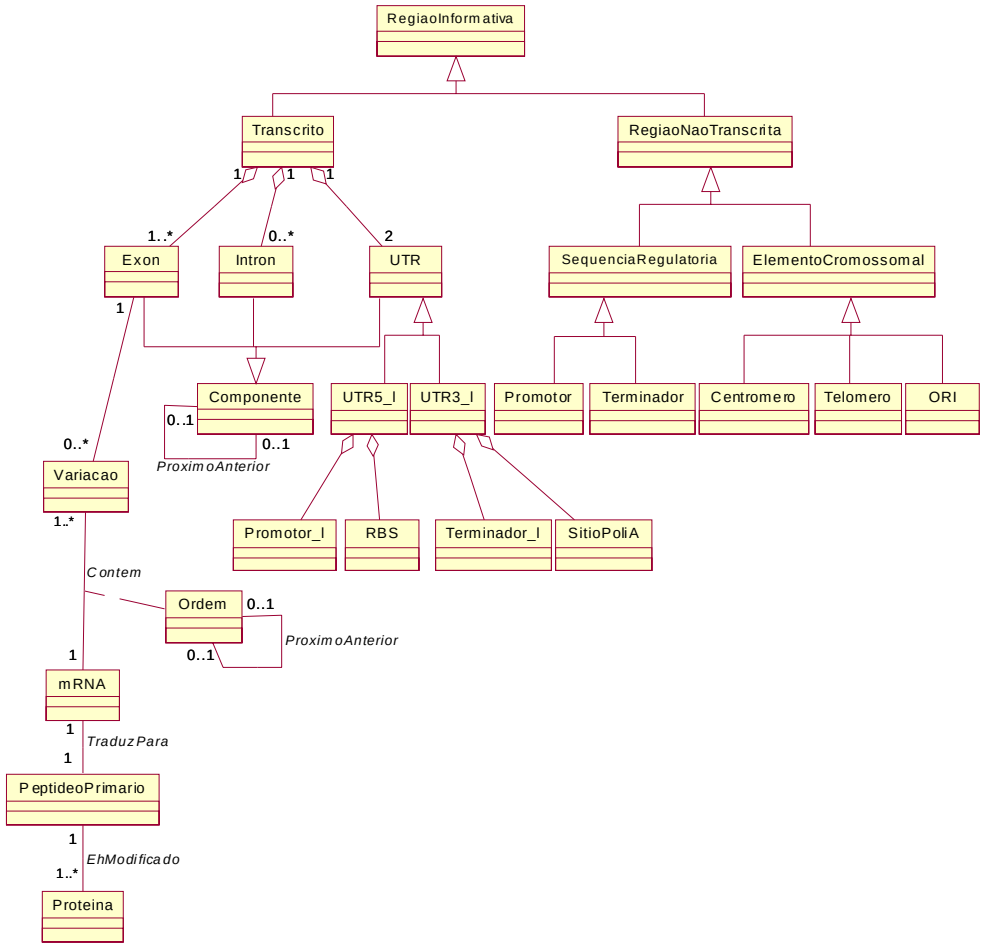

Figura 15 - Diagrama de classes de genoma (continuação referente a Região Informativa)

Um *Éxon* pode possuir diversas *Variação*(ões), que por sua vez está(ão) relacionada(s) ao *mRNA* em uma determinada *Ordem*.

O *mRNA* traduz para um *PeptídeoPrimário*, que por sua vez é modificado para gerar diversas *Proteína*(s).

Algumas fontes de dados da biologia molecular são exemplos de repositórios de instâncias de classes do Genoma. As fontes de dados GenBank, DDBJ, EMBL e

GSDB (entre outras) armazenam instâncias da classe *Genoma*. Da mesma forma, instâncias da classe *SequênciaRegulatória* estão armazenadas na fonte de dados TRANSFAC [TRANSFAC02] e as da classe *Palíndromo* estão na fonte REBASE [REBASE02, RM00].

5.1.2 Modelo das Informações de Proteoma

O modelo das informações de proteoma é apresentado na Figura 16.

Uma *Proteína* pode ser parte de uma *Família* e é composta de *Sítio*, *Domínio* e pode ter *EstruturaSecundária* e *EstruturaTerciária*. *Proteína*(s) podem ainda ter interações com outras.

Algumas fontes de dados da biologia molecular são exemplos de repositórios de instâncias das classes do proteoma. Assim, instâncias de *Família* estão armazenadas na fonte de dados PROFAM [PROFAM02], de *Sítio* na fonte PROSITE [PROSITE02], de *Domínio* na fonte ProDom [ProDom02], de *EstruturaSecundária* na fonte DSSP [DSSP02a, DSSP02b] e de *EstruturaTerciária* na fonte PDB [PDB02a, PDB02b].

O glossário dos termos utilizados nos modelos de genoma e de proteoma é apresentado no Anexo II.

5.2 Modelos Lógicos Utilizados em Biologia Molecular

Diversos modelos lógicos de dados têm sido utilizados para representação das informações biológicas. Esta seção discute os modelos lógicos existentes e as vantagens e desvantagens de cada um em termos de representação de fatos biológicos e de facilidades para os usuários.

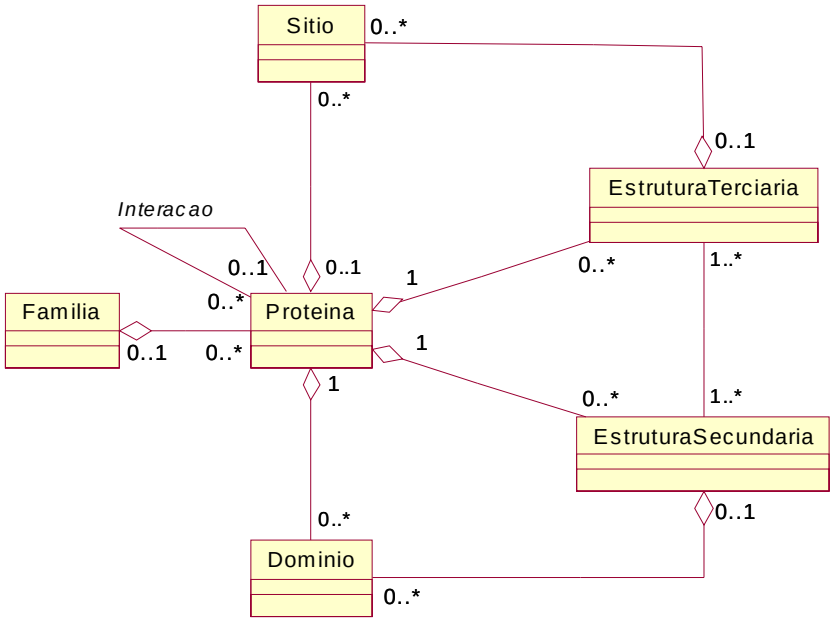

Figura 16 - Diagrama de classes do proteoma

5.2.1 Modelo Relacional

Como visto na seção anterior, diversas fontes de dados de biologia molecular são implementadas em bancos de dados relacionais disponíveis no mercado. Tal tecnologia, no entanto, apresenta vantagens e desvantagens [NK99] para esta aplicação, que serão resumidas a seguir.

O modelo relacional é baseado em normalizações, e agrega a informação em tuplas, onde cada tupla (ou linha da tabela relacional) representa uma coleção de valores correlacionados, que não podem mais ser separados em relações mais simples. A normalização serve para eliminar problemas inerentes à duplicação de dados, que são, por exemplo, múltiplas atualizações e geração de tuplas espúrias na operação de junção.

Nos bancos de dados de biologia molecular é freqüente a ausência de informações (que seriam representados por atributos que assumiriam valor NULL), fato que aumenta a decomposição dos dados em tabelas menores. Além disso, as frequentes exceções feitas às estruturas relacionais tendem a aumentar a decomposição, gerando novas tabelas. Assim, enquanto proliferam as tabelas do banco de dados, tornando os ítens de dados mais simples e de fácil entendimento de forma isolada, uma nova dificuldade aparece na compreensão e manutenção da estrutura dos dados, bem como no domínio completo do esquema.

Por exemplo, na implementação relacional do MITOMAP [MITOMAP02], a relação *genetic locus* sofreu os seguintes desmembramentos como conseqüência da evolução do esquema:

Fase 1: Genetic locus (nome, start, stop, dados_mutação, etc.)

A fase 1 apresentou dados de mutações dentro da tabela “Genetic locus”.

Fase 2: Genetic locus (nome, start, stop, id_mutação, etc.)
Mutação (id_mutação, tipo_mutação, dados_tipo_mutação, etc.)

Na fase 2 os dados de mutações estão em tabela à parte.

Fase 3: Genetic locus (nome, start, stop, id_mutação, etc.)
Mutação (id_mutação, tipo_mutação, etc.)
Mutação_inserção (dados_mutação_inserção)
Mutação_exclusão (dados_mutação_exclusão)
Mutação_alteração (dados_mutação_alteração)

Na fase 3 os dados de mutações foram desmembrados em três outras tabelas, de acordo com o tipo de mutação verificada (se por inserção, exclusão ou alteração de

uma ou mais bases de nucleotídeos). Assim, o objeto biológico se torna menos claro a cada decomposição.

Dado o grande tamanho destas bases de dados e ao elevado número de tabelas, rapidamente estas bases de dados se tornam difíceis de gerenciar e mesmo incompreensíveis pelos próprios administradores.

O modelo de entidades e relacionamentos é ideal para representar relacionamentos bem definidos entre as entidades. No entanto, os dados biológicos nem sempre se encaixam nessa categoria devido às transformações existentes em virtude, por exemplo, de novas classificações ou de novas descobertas biológicas. Assim, é necessário que o modelo que represente tais dados seja mais flexível, de forma a facilitar a sua adequação ao mundo real. O modelo relacional não fornece tal flexibilidade.

A formulação de consultas ao modelo implementado exige o conhecimento da sua estrutura, limitando o tipo de consultas que poderiam ser feitas, desencorajando a exploração da base por usuários comuns. Ou seja, apenas especialistas em bancos de dados poderiam fazer tais consultas, fato que reforça a ênfase a ser dada na simplicidade do modelo de dados, para que possa ser compreendido pelos usuários.

Existem, no entanto, benefícios na adoção de um modelo relacional. A teoria da normalização, baseada em dependências funcionais, garante a ausência de anomalias na base. A implementação relacional é responsável ainda pela obtenção de respostas rápidas às consultas, e por simplificar a tarefa de programação.

Em oposição a estes fatos, a validade da normalização se torna irrelevante se a tupla não pode representar o dado em questão (o objeto biológico completo). Adicione-se a isso o fato de que o modelo relacional não se ajusta facilmente à natureza dos dados biológicos. Por exemplo, não é possível representar um atributo com múltiplos tipos de dados, mas isso pode acontecer na natureza.

Em resumo, o modelo relacional representa o mapeamento incompleto do mundo real para o conjunto de informações necessárias ao estudo da biologia molecular, tornando a compreensão e atualização dos dados bastante difíceis. Tais alterações só poderiam ser feitas com o completo conhecimento do esquema do banco, e não necessariamente com o completo conhecimento dos dados biológicos e de suas relações.

Pelas razões descritas acima, diversas implementações usando outros modelos têm sido desenvolvidas.

5.2.2 Modelo Orientado a Objetos

Algumas fontes de dados biológicas foram implementadas no modelo de dados orientado a objetos (OO). O modelo OO traz vantagens em relação ao modelo relacional, pois permite o mapeamento direto de conceitos complexos do mundo real em estruturas de dados do modelo.

O projeto dos objetos do modelo permite determinar o grau de normalização / simplificação de cada entidade / objeto envolvido (tal fato não está ligado às regras do modelo).

Com a adoção do modelo OO, o usuário final recebe o benefício do conhecimento do objeto de forma completa (via sua estrutura de dados). Tal modelo também provê uma coleção de métodos e de estruturas para modelar, manter e consultar os dados.

Porém, o modelo OO também apresenta problemas. Objetos são representados em estruturas de dados fixas, têm métodos próprios e se relacionam através de ponteiros. Isso implica em que uma alteração no esquema do banco de dados pode acarretar na alteração da estrutura utilizada e mesmo na reprogramação dos métodos já implementados. Outro ponto problemático é a utilização de ponteiros para os objetos e do identificador único do objeto (OID) que, embora relevantes para o modelo, não são necessariamente relevantes em termos biológicos. Este fato pode dificultar a compreensão da referência aos dados por um usuário comum.

Outro complicador refere-se ao termo herança pois o seu uso na biologia difere do uso no modelo OO. As estruturas da biologia são representadas em uma enorme variedade de classes, que frequentemente não têm qualquer relação entre si. Assim, não existe nenhum benefício em herdar atributos de outras classes de objetos.

Embora o modelo orientado a objetos favoreça o mapeamento do mundo real, ainda existem inúmeras deficiências a serem resolvidas, que favoreceram o surgimento de novas implementações utilizando outros modelos [NK99].

5.2.3 Modelo Relacional-Objeto

O modelo relacional-objeto é bastante adequado para aplicações de biologia molecular pois estas aplicações são orientadas a consultas e requerem o uso de dados complexos.

De fato os bancos de dados que utilizam o modelo de dados relacional-objeto têm sido utilizados para o armazenamento de dados de biologia molecular, uma vez que aliam a facilidade de consulta inerente ao modelo relacional com o tratamento de dados complexos.

Os bancos de dados que adotam este modelo permitem a formulação de consultas utilizando-se funções e operadores definidos pelos usuários. Tais requisitos não existem na definição da linguagem SQL-2 [ANSI02], utilizada nos bancos que adotam o modelo relacional. Porém, são utilizados nas linguagens de consulta dos bancos de dados que adotam o modelo relacional-objeto [Informix02] (estes requisitos estão incluídos no padrão SQL-3 [ANSI02]).

O Tair [Tair02] (banco de dados do genoma da Arabidopsis thaliana) pode ser citado como exemplo de implementação neste modelo, utilizando a ferramenta Illustra.

5.2.4 Modelo de Dados Semi-Estruturados

Diversas fontes de dados biológicos implementam o arquivamento das informações utilizando dados semi-estruturados. É o caso do ACeDB e do GenBank. Outras fontes de dados se utilizam do código do AceDB e, portanto, utilizam o mesmo modelo. O AceDB optou por este tipo de implementação pelas facilidades inerentes à alteração dos objetos, sem necessáriamente exigir a alteração dos métodos já utilizados.

Para o AceDB, os objetos são definidos de acordo com uma linguagem cuja sintaxe é semelhante à XML. A representação dos dados pode ser vista como uma árvore, cujos nodos podem estar presentes ou não e onde existem facilidades (inerentes da estrutura) no sentido de adicionar, excluir e alterar nós ou sub-árvores. Assim, o AceDB armazena os dados nessa estrutura (árvore), em formato binário.

Outras fontes de dados de biologia têm arquivos semi-estruturados, de forma a facilitar a troca de informações com outros bancos. É o caso do Genbank, que utiliza o padrão ASN.1.

O modelo de dados semi-estruturado é bastante utilizado na área de bioinformática para troca de informações entre fontes de dados.

XML tem características voltadas para a solução de alguns problemas da comunidade de bioinformática [GJ00, AVB01, MDK+00, MH01, BH01], tais como:

- é flexível pois é simples modificar a DTD ou o XMLSchema, permitindo adicionar novos elementos ou atributos sem que seja necessário modificar os dados existentes.

- é orientada à Internet, o que significa que tem facilidades para interconexão de fontes de dados através de *links* entre as instâncias de dados (por exemplo na construção de referências cruzadas entre fontes de dados).

- é utilizada para definições e especificações padronizadas de dados.

- o arquivo pode ser lido por um editor de texto qualquer e pode ser compreendido, o que não ocorre, por exemplo, com o formato ASN.1.

Os dados podem ser também armazenados à parte em formatos próprios a fim de permitirem manipulação por algoritmos (programas) especiais. É o caso do formato FASTA, que é utilizado na execução dos algoritmos da família BLAST, para verificação de pré-existência de uma dada sequência no banco. O próprio GenBank, além de outras fontes de dados, implementa também este tipo de arquivamento.

5.3 Ontologias para Biologia Molecular

A integração das fontes de dados heterogêneas de biologia molecular é uma funcionalidade fundamental para uma ferramenta de suporte à pesquisa na área, pois essas fontes de dados armazenam informações complementares do domínio do conhecimento. Neste contexto, a definição de uma ontologia é extremamente importante para facilitar o entendimento e a interpretação dos esquemas e, conseqüentemente, a integração dos mesmos.

Para a comunidade de pesquisadores em Inteligência Artificial, uma ontologia é um sistema que descreve conceitos e relações entre eles. Essa descrição pode ser feita via redes semânticas, via *frames* ou via descrições lógicas, de forma a permitir a inferência de novos conhecimentos. Através da descrição dos conceitos, a ontologia trata da semântica dos elementos de um domínio e de suas relações [Big90, RK83].

A importância da adoção de uma ontologia tem sido reconhecida na comunidade de bioinformática [SK98] e alguns trabalhos têm sido feitos para o desenvolvimento de ontologias em biologia molecular [KRP+98, GO02, XOL92, BGB+99].

Representações através de *frames* provêem um *framework* preciso para a captura de conceitos e dos relacionamentos entre eles. [KRP+98] utiliza este formalismo.

As descrições lógicas (DL's) [Bor95] formam um outro exemplo de representação de conhecimento. DL's são feitas através de uma linguagem que é capaz de capturar conhecimentos que foram declarados sobre um determinado domínio e um classificador que permite fazer inferências sobre esse conhecimento. A informação capturada via DL's é estruturada em um formato hierárquico de conceitos e relacionamentos. Isto significa que novos conceitos podem ser construídos a partir de conceitos existentes e podem ser automaticamente encaixados na hierarquia. [BGB+99] utiliza este formalismo. DL's tem algumas limitações conhecidas, por exemplo existem limitações para restrições de cardinalidade e para representações de domínios (valores numéricos).

Para a comunidade de pesquisadores em Bancos de Dados, ontologias são tratadas no nível de linguagens de definição de esquemas. Essa forma de representação do conhecimento é considerada estática visto que não é possível fazer inferências (descoberta automática de novos conhecimentos). Porém, permite que se façam consultas aos dados de forma eficiente. Nessa abordagem, os relacionamentos ficam a cargo do projetista.

Assim, o termo ontologia é definido e tratado nesta tese como todo tipo de informação e conhecimento atribuídos a um objeto para facilitar seu entendimento e interpretação. A ontologia é definida através de um vocabulário associado ao objeto e de seus relacionamentos com outros objetos. O vocabulário de um objeto é composto pelo seu nome, descrição, tipo de dado, domínio de valores, além de outras informações como atributos e objetos que contém. O vocabulário dos relacionamentos é composto, adicionalmente, por restrições de integridade do tipo obrigatoriedade e cardinalidade. [SK98] também utiliza este formalismo.

5.4 Comentários Finais

Este capítulo procurou apresentar uma proposta de modelo conceitual das informações biológicas, modelo este que contempla o dogma central da biologia

molecular. Em seguida foram discutidos os modelos lógicos utilizados nas fontes de dados. Finalmente foram discutidas diferentes abordagens para definição de uma ontologia para a área.

No próximo capítulo é descrita a implementação do *framework*. São descritos os princípios que nortearam a sua construção, o ambiente de desenvolvimento e de testes, os componentes de reuso, os componentes desenvolvidos e a sistemática de uso e extensibilidade. Também são apresentados diversos problemas e dificuldades referentes à pesquisa em biologia molecular que poderiam ser resolvidos ou facilitados com o uso da ferramenta e estudos de caso práticos e reais que demonstram a utilidade da ferramenta e a sua aplicabilidade para suporte à pesquisa na área.

6 A Implementação do Framework

Inicialmente, o capítulo descreve as premissas adotadas para o desenvolvimento do framework.

Em seguida, é apresentada a forma de instanciação do *framework*, os componentes de *software* que foram re-utilizados e os que foram desenvolvidos.

Finalmente, são descritos os casos de uso exercitados de forma a demonstrar a sua aplicabilidade.

6.1 Premissas para o desenvolvimento da ferramenta

Nesta seção são discutidos aspectos de implementação dos módulos do *framework*. São discutidas as diferentes abordagens e são relatadas as decisões que foram tomadas para a implementação de cada módulo.

Em linhas gerais, o *framework* está implementado na linguagem Java [JAVA02], por ser uma linguagem orientada a objetos e por possibilitar a portabilidade de código entre diferentes plataformas. Existe ainda um grande conjunto de bibliotecas de classes disponível, especialmente para a área de bioinformática, que possibilitam alto grau de reuso. . Além disso, a ferramenta estará disponível para utilização através de uma interface *Web*.

6.1.1 Módulo Administrador

Este módulo requer uma interface amigável pois é o módulo de interação entre os usuários e a arquitetura. A especificação do Módulo Administrador passou pela discussão de diversas estratégias para a implementação da funcionalidade de integração de esquemas.

Inicialmente, a estratégia especificada foi a de se utilizar o auxílio de uma ferramenta *Case* orientada a objetos disponível no mercado, pois além da interface gráfica já ser consagrada, muitas das funcionalidades deste módulo são usualmente encontradas em tais ferramentas. A dificuldade verificada foi a de se encontrar uma ferramenta que utilizasse o recente padrão XMLSchema. As ferramentas de mercado mais conhecidas que utilizam o modelo orientado a objetos usam a linguagem UML para representação dos esquemas.

Assim, a primeira idéia foi a de se implementar um conversor entre as linguagens XMLSchema e UML, nos dois sentidos. No entanto, de forma a disponibilizar rapidamente a arquitetura, optou-se por um caminho diferente, uma via mais rápida, pois a implementação de um conversor completo seria bastante complexa a curto prazo. Além disso, seria difícil estabelecer um sub-conjunto da linguagem para facilitar a tarefa de conversão pois isso seria dependente da ferramenta escolhida. Optou-se então pelo desenvolvimento completo do módulo.

6.1.2 Módulo Capturador

Uma das funcionalidades deste módulo trata da persistência de esquemas e de dados. A especificação da implementação desta funcionalidade também passou pela discussão de que estratégia adotar.

A idéia inicial seria a de utilizar um gerenciador acadêmico que permitisse acrescentar novas funcionalidades, como por exemplo o gerenciador de objetos armazenáveis GOA [GOA02], cujos fontes foram disponibilizados por seus desenvolvedores. Porém, a ferramenta não conta com o tratamento de dados em XML e de esquemas em XMLSchema, o que dificultaria a sua aplicabilidade.

Outra estratégia no sentido de disponibilizar rapidamente uma primeira versão da arquitetura, seria a de se utilizar um gerenciador de banco de dados de mercado. Neste caso, apesar de não se dispor dos códigos-fonte para a implementação de novas

funcionalidades, já existem facilidades no tratamento de dados em XML e de esquemas em XMLSchema (por exemplo, na ferramenta Oracle9i).

O Oracle9i possui um tipo de dados especial, chamado XMLType, que pode ser utilizado para consultar e armazenar dados em XML no banco de dados. O tratamento do tipo de dados XMLType inclui funções para acessar, extrair e consultar dados em XML usando expressões em Xpath [Xpath02] (linguagem desenvolvida pelo comitê da W3C [W3C02] para acesso a partes de um documento XML). A pesquisa em documentos XMLType pode tratar a hierarquia entre elementos e seus atributos, e pode ser feita via SQL padrão. Estão ainda disponíveis outras formas de pesquisa em textos. Além disso a ferramenta Oracle9i permite criar índices textuais (Oracle Text Indexing) em uma coluna do tipo XMLType ou mesmo em elementos XML.

Por estas razões, decidiu-se pela utilização do SGBD Oracle 9i, embora outros gerenciadores possam ser posteriormente analisados e utilizados, se demonstrarem melhor adequação à ferramenta.

6.1.3 Módulo Conversor

Este módulo tem como funcionalidade a conversão dos esquemas utilizados nas diversas fontes de dados para o padrão XMLSchema e dos dados para XML. Decidiu-se que esta tarefa seria realizada nos *wrappers*. Assim, cada *wrapper* é conhecedor da fonte de dados a ser convertida (esquema e dados). Os programadores encarregados da escrita de cada *wrapper* são os responsáveis por implementar a conversão.

Existe a possibilidade de um *wrapper* implementar a conversão de esquemas e dados de mais de uma fonte de dados biológicos, como no caso daquelas que adotarem um padrão (por exemplo o AceDB) onde um modelo descreve o esquema de cada fonte de dados cujos dados estão armazenados em XML. O mesmo pode ocorrer para as fontes de dados que utilizarem um mesmo produto, por exemplo, um SGBD relacional de mercado, onde as definições de esquemas e formatos de dados constam do catálogo do sistema.

6.1.4 Módulo de Interface com os Drivers de Aplicação

Este módulo é implementado da mesma forma que o Módulo Conversor, ou seja, cada *driver* é conhecedor do formato de saída dos dados e é responsável pela implementação das conversões necessárias. No entanto, como famílias de aplicações utilizam um mesmo formato, por exemplo o formato FASTA ou mesmo o formato GenBank (ASN.1), o número de *drivers* não deve ser elevado, reduzindo-se o esforço de implementação.

6.2 Descrição da Implementação do *Framework*

Nesta seção é descrita a implementação de cada funcionalidade da arquitetura. Para cada funcionalidade são relacionadas as classes reutilizadas e são descritas aquelas desenvolvidas.

As classe abstratas definem os pontos de flexibilização da arquitetura (*hot spots*). Um novo *wrapper* ou *driver* a ser construído deve estender essas classes e implementar os métodos definidos pela sua superclasse. Os *wrappers* e *drivers* devem estar instanciados no *framework* de forma a se gerar o sistema.

6.2.1 Captura do esquema de uma fonte de dados

A implementação da captura de um esquema de uma fonte de dados é dependente do modelo de dados utilizado pela fonte. Para uma fonte de dados que utiliza o modelo semi-estruturado, a captura do esquema pode ser feita da seguinte forma:

- Primeiramente é definida uma gramática para a linguagem utilizada;

- Em seguida é executado um gerador de analisadores sintáticos (por exemplo, YACC [YACC02, LMB92] ou JAVACC [JAVACC02]), indicando como entrada de dados a gramática anteriormente definida;

- O analisador sintático é alterado para gerar código XML (no JAVACC é possível definir, na gramática, métodos e chamadas a métodos de classes. Pode-se, assim, gerar automaticamente uma saída no formato XML);

- Finalmente é utilizado o programa SPY [W3C02] para, a partir do código XML, gerar o XMLSchema correspondente.

Existem ainda diversos *parsers* disponíveis na *Web*, escritos em linguagem Perl e/ou em Java, que são capazes de efetuar a conversão dos dados de diversas fontes para XML. Estes *parsers* podem também ser utilizados. De posse dos dados em XML, é possível gerar o correspondente esquema em XMLSchema, usando o aplicativo SPY;

Para as fontes de dados implementadas via SGBD relacional ou orientado a objetos, pode ser gerado o esquema do banco, em XMLSchema, através de operações de consulta ao catálogo do banco.

O protótipo efetivamente implementado utilizou apenas fontes de dados que utilizam o modelo semi-estruturado porque são a maioria na área de biologia molecular e também porque, mesmo aquelas implementadas via SGBD, disponibilizam os dados em arquivos semi-estruturados.

No protótipo foram implementados *wrappers* para uma fonte de dados de sequências de nucleotídeos (GenBank) e para duas fontes de sequências de proteínas (PIR e Swiss-Prot).

Para o Swiss-Prot, o esquema foi definido manualmente a partir da documentação disponível em [ExPASy02b].

O PIR disponibiliza os dados em XML bem como o correspondente DTD. O aplicativo SPY é capaz de gerar a descrição do esquema em XMLSchema, a partir do DTD. Essa estratégia foi utilizada para gerar o esquema do PIR.

Para a geração do esquema do GenBank, foi utilizado um parser de domínio público denominado READSEQ [READSEQ02], que é capaz de transformar os dados para XML e de gerar o correspondente DTD. Mais uma vez foi utilizado o SPY para converter o DTD para a correspondente representação em XMLSchema.

Cabe ressaltar que o documento XMLSchema que é gerado a partir do SPY é pouco claro pois as definições de dados são geradas automaticamente a partir das instâncias ou do DTD. Assim, foi necessário editar o documento XMLSchema gerado pelo SPY, de forma simplificá-lo e a facilitar a sua compreensão.

6.2.2 Criação / extensão do esquema global da biologia

Em linhas gerais, pode-se descrever esta funcionalidade como aquela capaz de criar e de estender o esquema conceitual global de informações biológicas. A forma natural de implementar esta funcionalidade parte de um esquema inicial previamente definido, que é incrementado com os esquemas capturados das fontes de dados.

Este incremento pode ser feito através da construção de uma ferramenta visual que exibe o esquema global e um esquema capturado e permite que o usuário administrador manipule os esquemas de forma a representar a integração das informações.

A integração pode identificar novos objetos a serem incorporados ao esquema global e novos relacionamentos. A cada objeto do esquema global podem ser atribuídas propriedades com suas respectivas informações (nome, tipo, domínio, valor *default* e indicação de obrigatoriedade), métodos e relacionamentos com outros objetos (com as correspondentes cardinalidades).

Além disso, podem ser atribuídos conceitos aos objetos de forma que eles possam ser melhor compreendidos e interpretados. É importante que o objeto contenha referências às fontes de dados que armazenem suas instâncias para que seja possível, futuramente, instanciar o esquema global. Podem ainda ser identificados sinônimos entre os objetos e a necessidade de conversão de tipos de dados ou mesmo de domínio. Os objetos podem ser simples ou compostos, sendo que neste caso, poder ser organizados de diferentes modos. Essa organização deve contemplar aquelas tipicamente utilizadas em biologia (por exemplo, os tipos de CPL, lista, sacola, registro e variante). A integração de esquemas, feita desta forma, resulta de fato, na geração de uma ontologia que é efetivamente utilizada nas fontes de dados.

O protótipo implementado contempla a forma de integração descrita. A linguagem XMLSchema possibilitou a representação dos metadados supra-mencionados.

Assim, por exemplo, para representar informações de um objeto e respectivas propriedades (nome, tipo, domínio, valor *default* e indicação de obrigatoriedade), foram utilizadas cláusulas com correspondência direta em XMLSchema (*name*, *type* de *element*; *restriction*; *use*, *value*, *required* / *optional* de *attribute*).

Os relacionamentos com outros objetos, e as correspondentes cardinalidades, foram mapeadas em XMLSchema nas cláusulas *ref*, *minoccurs* / *maxoccurs* de *element.*

Para a definição de conceitos relativos a um objeto, em XMLSchema foi acrescentado um novo atributo ao objeto, implementado através da cláusula *name* = “desc” e *value* = “descrição do conceito” de *attribute*.

Analogamente, para a especificação da fonte de dados à qual o objeto pertence, foi acrescentado um novo atributo ao objeto, implementado através da cláusula *name* = “source” e *fixed* = “nome da fonte de dados” de *attribute*.

Da mesma forma, para identificar sinônimos entre os objetos e a conversão de tipos de dados e de domínio, foram acrescentados atributos ao objeto.

Para representar os objetos simples ou compostos foram utilizadas as cláusulas *simpleType* e *complexType*.

Os diferentes modos de organização dos objetos também podem ser representados utilizando cláusulas especiais de XML Schema. Por exemplo para representar a lista de CPL deve ser usada a cláusula *sequence* de *element* e para representar a variante de CPL deve ser usada a cláusula *choice*.

O protótipo possui uma interface amigável na *Web* através da qual é possível construir um esquema global a partir de outros esquemas de fontes de dados existentes. Os esquemas são representados através de árvores, o que faz com que os usuários não tenham que aprender XML Schema.

Para implementar esta tarefa foram utilizados pacotes Oracle encontrados em *http://technet.oracle.com/docs/tech/xml/xdk_java/doc_library/Production9i/index.html*, em particular *oracle.xml.parser.schema*, *oracle.xml.parser.v2* e *oracle.xml.treeviewer*. Estes pacotes permitem realizar um *parsing* no XML Schema e representá-lo sob a forma de árvore.

Para o controle dos eventos e das regras de negócio da aplicação foram construídas classes que formam uma camada acima dos pacotes Oracle. Estas classes realizam, por exemplo, as seguintes tarefas:

- definição dos conceitos dos objetos;

- implementação dos sinônimo;

- implementação das regras semânticas que governam a construção da árvore do esquema global (criar, remover, renomear objeto(s) da árvore);

- construção do XML Schema a partir da árvore do esquema global;

- encapsulamento das informações sobre o SGBD e sobre os esquemas que são utilizados pela ferramenta (é possível ler esquema de qualquer SGBD e de arquivos texto);

- interface gráfica da aplicação que trata da exibição das árvores a partir do XML Schema. Esta classe se utiliza do pacote Oracle para fazer o *parsing* do esquema (em XMLSchema) e para exibir seu conteúdo na forma de árvore (Jtree).

6.2.3 Criação de um esquema específico para uma pesquisa

O *framework* permite criar esquemas específicos para uma dada pesquisa a partir do esquema global da biologia. Esta funcionalidade é um caso particular do caso de criação e extensão do esquema global da biologia, tratado anteriormente.

No caso anterior, o esquema da fonte de dados externa era o esquema origem e o esquema geral, o destino. Neste caso o esquema origem é o esquema geral e o esquema destino é o esquema específico.

Existe apenas uma classe que trata a interface inicial do sistema para definir se o modo de operação é o de criação/extensão de um esquema geral ou de um esquema específico. Após a escolha da operação, feita pelo usuário administrador, é sabido quais são os esquemas que devem ser apresentados e, a partir daí, as mesmas classes (que independem da operação escolhida) são acionadas.

6.2.4 Instanciação de dados relativos a um esquema específico (captura de dados)

A tarefa de instanciar dados relativos a um esquema específico é função do esquema a ser instanciado (refere-se a objetos de apenas uma fonte de dados, ou refere-se a objetos de mais de uma fonte de dados porém os objetos já estão instanciados no

repositório, ou ainda refere-se a objetos de mais de uma fonte de dados porém os objetos podem não estar instanciados no repositório) e é também função do modelo de dados utilizado na fonte.

6.2.4.1 Instanciar objetos de uma única fonte de dados

Para uma fonte de dados que utiliza o modelo semi-estruturado, a captura de dados pode ser feita da seguinte forma:

- Se os dados já estiverem em XML, basta verificar a sintaxe dos dados capturados e armazená-los no repositório;

- Se os dados não estiverem em XML, primeiramente é definida uma gramática para a linguagem utilizada, em seguida é executado um gerador de analisadores sintáticos (YACC ou JAVACC), indicando como entrada de dados a gramática anteriormente definida e finalmente o analisador sintático é alterado para gerar código XML (ou pode ser criado já prevendo a geração de código em XML, via JAVACC).

Existem ainda diversos *parsers* disponíveis na Web, escritos em Perl e/ou em Java, que são capazes de efetuar a conversão dos dados de diversas fontes para XML. Estes *parsers* podem também ser utilizados.

Para as fontes de dados implementadas via SGBD relacional existem diversas formas de se gerar os dados em XML. A forma conveniente para a arquitetura requer a montagem do objeto biológico e não a simples conversão dos dados de cada tabela para XML.

Para as fontes de dados implementadas via SGBD orientado a objetos, os dados no formato XML podem ser gerados através de operações de consulta ao banco.

O protótipo implementado utilizou apenas fontes de dados que utilizam o modelo semi-estruturado porque são a maioria na área de biologia molecular e também porque, mesmo aquelas implementadas via SGBD, disponibilizam os dados em arquivos semi-estruturados.

No protótipo foram implementados *wrappers* para uma fonte de dados de sequências de nucleotídeos (GenBank) e para duas fontes de sequências de proteínas (PIR e Swiss-Prot).

Para o Swiss-Prot, os dados forma capturados via *parser* específico, construído a partir da documentação disponível.

O PIR disponibiliza os dados em XML, os quais foram carregados diretamente para o repositório passando antes por uma validação de sintaxe, implementada via *trigger*.

Para a transformação dos dados do GenBank para XML, foi utilizado um *parser* de domínio público denominado READSEQ.

6.2.4.2 Instanciar dados de um esquema a partir de objetos já existentes no repositório

Neste caso, a partir do esquema global, que define objetos, relacionamentos e localização (no caso, no *data warehouse*), deve-se consultar as instâncias dos objetos armazenados no repositório para construir as instâncias de dados do esquema. O uso do SGBD Oracle facilita a execução desta tarefa.

6.2.4.3 Instanciar dados de um esquema a partir de objetos ainda não disponíveis no repositório

Neste caso, a partir do esquema global, que define a localização dos objetos nas fontes de dados, deve-se recorrer aos *wrappers* das fontes de dados para a captura das instâncias dos objetos que participam do esquema.

6.2.5 Geração de dados em um formato esperado por um aplicativo

A implementação desta funcionalidade requer, como mencionado anteriormente, que o *driver* a ser construído conheça o formato dos dados, que é exigido pelo aplicativo. Assim, para cada aplicativo, ou família de aplicativos, deve ser construído um *driver* específico.

No protótipo implementado foi construído um *driver* para a aplicação BLAST. Sendo assim, o *driver* lê os dados de sequências (de nucleotídeos ou de aminoácidos) do repositório e monta um arquivo no formato FASTA, que é o formato esperado pela aplicação BLAST.

6.2.6 Execução de aplicativo próprio da arquitetura

As classes do esquema global da biologia, ou mesmo dos esquemas capturados, podem estar associadas a métodos que são instanciados na arquitetura através do uso de componentes (peça de *software* pequena o suficiente para criar e grande o bastante para desenvolver e manter, possuindo uma interface padronizada de forma a permitir a interoperabilidade), implementados através do mecanismo de Java Beans [JAVABEANS02].

No protótipo há um repositório de componentes, que é lido pelo Módulo Administrador, para exibição aos usuários. Os usuários podem assim escolher o componente a ser executado sobre os dados que estão no repositório.

No protótipo, a classe sequência do esquema global tem o método *Alinhamento,* que executa o algoritmo de alinhamento global ótimo [NW70] entre pares de sequências, sendo que uma é obtida do repositório e a outra é dada como parâmetro.

6.2.7 Monitoramento da alteração do esquema de uma fonte de dados

A implementação desta funcionalidade é feita através do uso de um agente de *software,* que é construído dentro do *wrapper* da fonte de dados. O agente monitora constantemente o esquema daquela fonte, informando ao usuário administrador a ocorrência de alterações, de forma a que este proceda a uma nova captura de esquema. Esta funcionalidade permite o tratamento da evolução de esquemas.

6.2.8 Atualização das instâncias de dados no repositório

A implementação desta funcionalidade também é feita através do uso de uma agente de software, construído dentro do *wrapper* da fonte de dados. O agente percebe mudanças nas instâncias de dados e atualiza o repositório.

6.3 Exemplos de aplicação da arquitetura e propostas de pesquisas

Nesta seção são exemplificados casos reais de aplicação da arquitetura e também são propostas pesquisas em biologia molecular a serem realizadas através da ferramenta.

6.3.1 Exemplos de uso da arquitetura em uma pesquisa real

Neste ítem será exemplificado o uso do *framework* para suportar uma pesquisa relativa à evolução molecular. A questão envolvida trata de analisar que fatores são envolvidos na escolha de códons. As tarefas a serem realizadas são descritas a seguir e são apresentados os passos do *framework* que deverão ser executados de forma a que

este realize cada tarefa. A pesquisa, sem o uso da ferramenta, levou seis meses para ser realizada [MAJ+00]. Na maior parte do tempo, os biólogos estavam construindo e utilizando ferramentas de software que não são compatíveis entre si, dificultando ainda mais o trabalho. Estima-se que, com a ferramenta, este tempo seja menor, mesmo sem que a infra-estrutura necessária esteja desenvolvida (*wrappers* e *drivers* implementados e algoritmos incorporados à arquitetura) pois a arquitetura pressupõe a compatibilidade entre as aplicações.

Tarefa 1: Coletar sequências codificantes em uma ou mais fontes de dados (por exemplo, no GenBank) de um conjunto de organismos selecionados para a pesquisa (por exemplo, os bacilos da lepra e da tuberculose).

Passo 1: Construir *wrapper* para o GenBank (captura de esquemas e de dados);
Passo 2: Casar o esquema do GenBank com o esquema do modelo da Biologia;
Passo 3: Instanciar dados do GenBank, relacionando-os ao esquema correspondente;
Passo 4: Selecionar da base instanciada os registros relevantes (lepra e tuberculose), via linguagem de consulta;
Passo 5: Construir *driver* para formato de dados do GenBank e instanciar arquivo com a base resultante do Passo 4;
Passo 6: Selecionar regiões codificantes da base resultante do Passo 5, via aplicativo ACNUC [ACNUC02], que reconhece formato de dados do GenBank;

Tarefa 2: Eliminar desvios nas sequências codificantes selecionadas na tarefa anterior.

Para cada organismo da pesquisa, executar os seguintes passos:
Passo 7: Eliminar regiões codificantes redundantes de forma a evitar desvios estatísticos, via aplicativo FASTA do pacote GCG [GCG02] , que reconhece formato de dados do GenBank;
Passo 8: Eliminar genes com poucos códons (menos do que 100, por exemplo), via aplicativo próprio que reconhece formato de dados do GenBank;

Tarefa 3: Construir *clusters* entre genes dos organismos em estudo.

Passo 9 : Construir *driver* para formato FASTA;
Passo10: Casar genes semelhantes entre os organismos em estudo, via aplicativo BLAST, que reconhece formato de dados do GenBank ou formato FASTA;
Passo11: Criar alinhamentos entre os *clusters*, via aplicativo ClustalW [ClustalW02];

Tarefa 4: Retirar sequências com possíveis erros de leitura ou sequenciamento (*frame shift*).

Passo 12: Executar algoritmo para identificar desvio composicional na terceira posição do códon, via aplicativo próprio, e retirar as sequências identificadas;
Passo13: Executar algoritmo para exibir o perfil de substituições sinônimas e não-sinônimas e retirar as sequências identificadas;

Tarefa 5: Construir a tabela de códons para cada gene e organismo.

Passo 14: Executar algoritmo que gera a tabela de códons, via aplicativo próprio.

Neste ponto, a análise da tabela de códons poderá levar a decisões do tipo:

a) Obter genes de outros organismos de forma a expandir os resultados (esta tarefa será bastante simplificada já que toda a infra-estrutura para a sua execução estará pronta).

b) Relacionar os resultados com fenômenos biológicos, como por exemplo: a estrutura da proteína que é sintetizada, a expressão gênica, a localização celular, o tempo, entre outros.

Tarefa 6: Continuando a demonstrar a funcionalidade do *framework*, pode-se supor que os pesquisadores decidam investigar a relação entre a tabela de códons de cada gene e a estrutura da proteína que é sintetizada, procurando identificar a composição

de regiões da estrutura da proteína (α-hélice, β-pregueada ou mesmo alças), ou seja, trata-se de investigar que códons são mais utilizados em cada estrutura. Neste caso, seriam necessários os seguintes passos:

Passo 15: Construir *wrapper* para o PDB (banco de dados de estruturas de proteínas);

Passo 16: Casar o esquema do PDB com o modelo da biologia (neste caso seria identificado um relacionamento entre a estrutura da proteína no PDB e o gene no GenBank, via identificador do gene);

Passo 17: Executar algoritmo para relacionar regiões de estruturas e códons, identificando que códons são mais utilizados em cada estrutura, via aplicativo próprio;

Outra forma de fazer o mesmo estudo, no caso de não se ter no PDB as estruturas necessárias, seria executar algoritmos que fornecem a previsão de estruturas de proteínas e extrair um consenso entre os resultados.

Tarefa 7: Outra pesquisa a ser feita seria obter um refinamento da relação entre a expressão gênica, com dados obtidos a partir de experimentos de *microarray*, e os códons mais utilizados em função de outros fenômenos como o tempo e a localização da proteína. Cabe observar que os dados de experimentos de *microarray* (padrões de expressão gênica) seriam obtidos através da instanciação de um *wrapper* para o banco de dados ArrayExpress [ArrayExpress02]. Estes dados seriam relacionados aos genes via identificador da sequência ou mesmo identificador da proteína. Os passos utilizados para realizar o experimento via *framework* são:

Passo 18: Construir *wrapper* para o ArrayExpress;

Passo 19: Casar modelo do banco ArrayExpresss com o modelo da biologia, identificando os relacionamentos existentes;

Passo 20: Executar aplicativo que extraia a informação desejada, ou seja, que códons são mais utilizados em função do fenômeno biológico em estudo.

6.3.2 Propostas de pesquisas realizadas via arquitetura

São relacionadas a seguir as pesquisas a serem realizadas utilizando-se o *framework*.

6.3.2.1 Alinhamentos de sequências de nucleotídeos a partir dos alinhamentos do COG (*Cluster of Orthologous Groups*)

A fonte de dados COG [COG02] fornece alinhamentos de todos os genes de um grupo de cerca de quarenta espécies de bactérias. Esses alinhamentos são feitos nas sequências de proteínas. O problema é que um aminoácido é formado por três nucleotídeos (que pode assumir os valores A, T, C, G), assim, deveriam existir 4^3 (64) possibilidades de aminoácidos. Porém como existem apenas vinte aminoácidos, códigos repetidos formam um mesmo aminoácido, mas a representação em termos de nucleotídeos pode ser diferente.

Em termos biológicos é útil implementar tais alinhamentos em sequências de nucleotídeos que correspondam às sequências de aminoácidos, por exemplo, para estudos em evolução molecular.

6.3.2.2 *Data Mining* em regiões promotoras

Os *clusters* de genes que são expressos de uma mesma forma em uma lâmina de *Array* de DNA (*in vitro*) podem ser analisados no sentido de descobrir se podem ser formados via análise computacional. Isto pode ser feito analisando-se as regiões promotoras de cada gene do *cluster* e, buscando-se padrões de regiões promotoras na base de dados TRANSFAC [TRANSFAC02, WCH+00], pode-se averiguar se as regiões promotoras têm um padrão que explique a expressão gênica que foi analisada. Se isso ocorrer, os *clusters* poderão ser formados *in silico*.

6.3.2.3 Análises de anotações nas fontes de dados de genoma, de forma a apontar inconsistências

Um dos problemas importantes da bioinformática hoje, que deve ser resolvido rapidamente, consiste em promover uma limpeza de dados inconsistentes que foram armazenados nas fontes de dados, em especial, as anotações biológicas. Muitas dessas anotações resultam de erros de sequenciamento e pouco cuidado na submissão de sequências para armazenamento, ou mesmo pouco cuidado no relacionamento de informações, que geraram anotações erradas. A solução para este problema pode ser tratada através da comparação das anotações existentes nas diversas fontes de dados e o relato das inconsistências verificadas. Por exemplo, nas anotações em genes, pode-se expor erros de classificação (se um gene em uma dada fonte estiver classificado como uma kinase e em outra como uma álcool-desidrogenase).

6.4 Casos de Uso da Arquitetura

Nesta seção são apresentados os diversos casos de uso que foram construídos de forma a comprovar a funcionalidade da arquitetura.

6.4.1 Execução de um algoritmo (BLAST) externo à arquitetura

Para demonstrar esta funcionalidade, foi necessária a implementação de uma aplicação cliente-servidor na *Web* onde um usuário pode comparar uma dada sequência contra sequências armazenadas no repositório do *framework* ou contra um banco de dados de sequências de sua propriedade. A comparação é feita utilizando o algoritmo BLAST, que é externo ao *framework*.

Para realizar esta operação, o usuário precisa fornecer um arquivo com a sequência de entrada e indicar se deseja que a comparação seja feita contra um banco de dados pessoal ou contra sequências do repositório do *framework*.

No primeiro caso, o usuário precisa fornecer o banco de dados pessoal no formato de um arquivo FASTA.

No segundo caso, o usuário deve selecionar qual esquema no repositório do *framework* armazena as sequências contra as quais se deseja realizar a comparação. De posse desta informação, o *driver* deve ler do repositório as instâncias do esquema especificado pelo usuário e gerar o arquivo no formato FASTA necessário para a execução do BLAST.

Em seguida, para os dois casos, tanto o arquivo no formato FASTA como o arquivo com a sequência de entrada são enviados para a máquina onde o BLAST está instalado. Após esta operação, um programa auxiliar dispara a execução do aplicativo BLAST, passando como parâmetro os arquivos enviados. A resposta do BLAST é enviada à máquina cliente, que disparou o processo.

6.4.2 Execução de um algoritmo (Alinhamento Global Ótimo) interno à arquitetura

Os esquemas global e específico têm métodos associados a seus objetos. Um exemplo de método é o alinhamento global ótimo, que foi implementado para demonstrar a funcionalidade de execução de algoritmos internos à arquitetura.

No protótipo, o usuário indica que deseja realizar a operação de alinhamento com, por exemplo, uma sequência obtida através de uma consulta e uma outra dada como entrada.

Para implementar esta funcionalidade foram reutilizados pacotes Java encontrados na *Web* que executam transferência de arquivos, são eles:

- FtpBean - http://www.geocities.com/SiliconValley/Code/9129/javabean/ftpbean/
- Servlets.com - http://www.servlets.com/cos/index.html

6.4.3 Execução de consultas

Existem diversos tipos de consultas que podem ser realizadas sobre o repositório implementado no protótipo. Elas podem ser do tipo simples, onde o usuário fornece palavras-chave e tem a possibilidade de utilizar conectores lógicos (*and*, *or*, *not*) e complexas, onde é utilizado o operador de junção.

Estas consultas podem ser realizadas em instâncias de um mesmo esquema ou em instâncias de esquemas diferentes.

6.4.3.1 Consultas simples

O protótipo possibilita a consulta a instâncias de esquemas armazenados no repositório. Assim, foram capturados os esquemas das fontes de dados PIR, Swiss-Prot e GenBank, que foram também instanciados.

Para exemplificar a funcionalidade, foram formuladas consultas aos campos das fontes de dados (por exemplo os atributos nid, accession, keyword e seq-data das instâncias do Genbank, ou os campos uid, accession, keyword e sequence do PIR). Para tal o usuário deve indicar sobre qual fonte de dados / campo deve ser feita a consulta e o predicado da cláusula *where*. São permitidas ainda consultas que utilizam conectores lógicos.

6.4.3.2 Consultas utilizando o operador de junção

O protótipo implementado permite a formulação de consultas utilizando-se o operador de junção. Um exemplo implementado compara anotações feitas em duas fontes de dados de proteínas (PIR e Swiss-Prot) no campo *keyword*. Um registro do Swissprot pode conter referências ao PIR e vice-versa, esse campo é utilizado como operador de junção.

Assim, um exemplo de consulta consiste em descobrir quais são as *keywords* dos registros do PIR e Swissprot que são diferentes, para uma mesma proteína.

Este tipo de consulta é interessante porque permite descobrir anotações diferentes que estão associadas à mesma sequência, em diferentes fontes de dados, ou seja, detecta inconsistências nos dados.

6.5 Comentários Finais

Neste capítulo foi apresentada a implementação do *framework*. Foi apresentada uma descrição em alto nível de cada módulo de *software* desenvolvido. Foram ainda relacionados, para cada módulo, os componentes reutilizados. Ao final do capítulo foram apresentados casos de uso que comprovam a utilidade da ferramenta para suporte à pesquisa, tendo sido inclusive verificadas, na prática, inconsistências de anotações em diferentes fontes de dados de proteínas.

A implementação do protótipo da arquitetura demonstrou a sua viabilidade técnica e de custos.

O capítulo seguinte apresenta as conclusões do trabalho.

7 Conclusão

Este capítulo apresenta as conclusões do trabalho.

Inicialmente é apresentado uma síntese do estudo, detalhando-se as etapas executadas, as direções de projeto e a implementação da arquitetura.

Em seguida são relacionadas as principais contribuições da tese.

Finalmente são apresentados os trabalhos futuros, que prevêem estudos e extensões na arquitetura, de forma a disponibilizar uma ferramenta dinâmica e atualizada para os pesquisadores em biologia molecular e bioinformática.

7.1 Síntese do trabalho

Este trabalho tem como objetivo contribuir para uma melhor compreensão dos fenômenos relativos à biologia molecular, e de especificar e construir uma ferramenta que forneça o suporte adequado à pesquisa nesta área. Nesta direção foram concentrados os esforços do estudo. Assim, o foco do trabalho foi direcionado para o estudo da integração de informações biológicas, percebido como o principal problema atual da pesquisa. Seguem-se os passos dados para a definição, projeto e implementação da ferramenta.

Inicialmente foram analisadas as principais fontes de dados utilizadas em termos das informações que armazena, da tecnologia utilizada e das formas de acesso aos dados que disponibiliza. Neste ponto, percebeu-se imediatamente o problema de integração de esquemas, pois cada fonte de dados tem esquema e modelo de dados próprio, além de conter informações complementares sobre o domínio do conhecimento. Passou-se então a privilegiar a construção de um esquema global de modo a se compreender melhor os elementos de informação e seus relacionamentos. Ao se verificar a

literatura notou-se que muito pouco havia sido explorado neste sentido. Esforços de pesquisa individuais foram estudados, especialmente o modelo conceitual de informações biológicas proposto no projeto GIMS e a ontologia proposta pelo mesmo grupo de pesquisa, documentada no projeto TAMBIS, além de esforços de grupos de pesquisas e empresas concentrados na OMG, que resultaram em documentos de especificação de objetos relativos a mapas de genoma, a análise de sequências e a expressão de genes. Assim, foi aberta uma orientação de pesquisa que resultou em uma proposta de esquema conceitual para informações da biologia molecular referentes ao dogma central.

Outro aspecto relevante verificado no estudo das fontes de dados é o fato da necessidade de cada uma estar associada a aplicativos que formam o ferramental básico dos pesquisadores da área. Verificou-se que cada fonte de dados é formada por um sistema composto por um repositório de dados aliado a um conjunto de aplicativos que disponibiliza. Estes aplicativos são portanto associados às fontes de dados, e, embora tenham pontos em comum, são bastante diversificados. Neste aspecto, ficou evidenciada a necessidade de construção de conversores de formatos entre fontes de dados distintas de forma a possibilitar a execução dos aplicativos associados a cada fonte. Foram encontrados inúmeras ferramentas e famílias de ferramentas, inclusive com código aberto, que implementam estas conversões. Algumas foram analisadas e foi verificado que não são ferramentas genéricas, muitas foram desenvolvidas para tratar de um problema específico. Foram encontrados ainda aplicativos mal construídos, pobres de recursos e sem capacidade de generalização. Esses aspectos foram anotados para incorporar à ferramenta de integração que se desejava construir a funcionalidade de converter formatos de dados.

Em seguida foram estudadas as principais arquiteturas de integração existentes. Foram relacionados os principais problemas verificados em cada uma e as propostas de solução possíveis. Foram relacionadas também as funcionalidades que deveriam implementar. Neste momento, ficou evidente a necessidade de possibilitar o acesso a todas as fontes de dados. A ferramenta deveria ter *plug-ins* para a implementação do acesso. Neste aspecto foi mandatório o estudo da tecnologia de *frameworks* orientados

a objetos de forma a dotar a ferramenta da flexibilidade necessária. O desenvolvimento baseado em *framework* foi considerado pois os requisitos do sistema evoluem rapidamente (novas descobertas ocorrem com frequência). A construção do *framework* foi considerada também devido à necessidade de se implementar o desenvolvimento incremental dos *hot spots*. A mesma idéia (*hot spots*) foi adotada para que a ferramenta possibilitasse o uso de qualquer aplicativo disponível, isto é, deveria prever *plug-ins* para os aplicativos também.

O estudo das ferramentas disponíveis demonstrou a importância de se instanciar um repositório próprio no sentido de atender ao requisito de desempenho na respostas às consultas. Todas as ferramentas caminharam para este tipo de implementação, em detrimento da proposta inicial de algumas que tratava da formulação de consultas a um ambiente distribuído. Os complexos algoritmos utilizados na área favoreceram esta decisão.

Para completar a definição da arquitetura, foi estudada a tecnologia de mediadores, já que se desejava inserir dados em um repositório formado a partir de instâncias de fontes locais, que deveriam ser acessadas. O repositório conta com um esquema global, obtido a partir do estudo dos esquemas das fontes de dados locais. A tecnologia utilizada na construção de mediadores com visão global foi portanto utilizada.

Com o arcabouço da ferramenta definido, passou-se então ao detalhamento do repositório e do modelo de dados a ser adotado. Os recentes estudos de aplicação de dados semi-estruturados em bioinformática aliado às facilidades inerentes à sua estrutura (troca de informações entre instituições de pesquisa; facilidade para alterar e adicionar novos elementos ou atributos; facilidade de leitura e compreensão do conteúdo do arquivo; facilidades para construção de *links* entre arquivos; adoção de padrões de documentos, entre outros) levaram a utilizar o padrão XML como modelo de dados da arquitetura. Analogamente foi utilizado o padrão XMLSchema para a representação dos esquemas que, por exemplo, é capaz de representar o mecanismo de

herança entre objetos, tipos de dados complexos, relacionamentos entre os objetos e as descrições semânticas necessárias, inclusive implementadas através de métodos.

Neste ponto, o repositório poderia ser implementado por um gerenciador de objetos que permitisse armazenar os dados em XML e cujo catálogo armazenasse as descrições dos dados em XMLSchema. O SGBD Oracle 9i foi então escolhido pois disponibiliza um tipo de dado específico para XML, permite que seja verificado o conteúdo do documento XML a ser armazenado (via trigger), permite que sejam formuladas consultas em SQL sobre dados XML (inclusive utilizando índices sobre o texto), entre outros aspectos favoráveis a esta decisão. Além disso, a adoção desta ferramenta facilita o desenvolvimento de um protótipo que possibilita exercitar o modelo de dados semi-estruturado, permitindo comparações com outras soluções (por exemplo, a adoção de um modelo orientado a objetos) para avaliação da sobrecarga que o formato dos dados baseado em texto pode causar (por exemplo, necessidade de *parsing* e sua influência nos termos de tempos de resposta a consultas; maior necessidade de armazenamento pois utilizando os dados em formato texto aumenta-se o tamanho da base de dados e dos objetos, e consequentemente é maior o tempo de transmissão de dados).

Após a fase de estudos e definições, passou-se ao projeto e implementação da arquitetura. O projeto foi feito utilizando-se uma ferramenta *Case* que permite a construção e manipulação de modelos orientados a objetos. Foram utilizados no projeto os diagramas de classes e de sequência da UML. A implementação da arquitetura foi desenvolvida na linguagem JAVA devido ao requisito de portabilidade. Alguns componentes obtidos via Web foram reutilizados (estão documentados no texto) que auxiliaram na construção de *wrappers*, *drivers* e no módulo de criação e incremento do esquema global, especialmente na visualização e *parsing* de documentos XML.

Com a ferramenta implementada foram realizados estudos de casos que permitiram demonstrar as principais funcionalidades da arquitetura, como por exemplo: a captura de esquemas, a integração de esquemas, a criação e instanciação de um esquema

próprio, a consulta de dados simples (com projeção) e utilizando o operador de junção e a execução de algoritmos externos e internos à arquitetura. Foi ainda exercitado um caso prático da pesquisa em biologia molecular, no sentido de faciltar a análise das anotações feitas nas fontes de dados de proteínas PIR e Swiss-Prot (em especial no atributo *keywords*).

A implementação da ferramenta tem demonstrado ser adequada para o suporte à pesquisa na área de biologia molecular. Diversos outros casos práticos foram documentados neste estudo, sendo que as respectivas implementações estão em curso de forma a se investigar os fenômenos e relações biológicas existentes entre os objetos.

7.2 Principais Contribuições

A tese apresenta uma arquitetura que engloba soluções para as questões referentes à integração de informações e de aplicativos da biologia molecular. A arquitetura, cuja especificação, projeto e implementação constam deste trabalho, propõe a integração de qualquer fonte de dados previamente existente e dos aplicativos disponíveis. A arquitetura permite ainda que sejam instanciadas novas fontes de dados e que sejam criados novos aplicativos, que são a ela incorporados.

A integração é feita via captura de esquemas para a construção e/ou alteração incremental (à medida que um novo esquema é capturado) de um esquema global. A captura dos esquemas é manual e deve ser feita por um administrador da arquitetura que, obrigatoriamente, deve conhecer os conceitos biológicos pois são necessários conhecimentos semânticos acerca dos objetos que estão sendo incorporados ao esquema global, além de definições precisas da ontologia efetivamente utilizada pela fonte de dados local.

Essa ontologia (que é incorporada ao esquema global) é utilizada pelo mediador de acesso aos dados, com objetivo de instanciar uma base de dados específica para uma dada pesquisa. Essa abordagem tem vantagens em termos de desempenho pois utiliza

um *data warehouse* integrador, ao invés de realizar consultas ou executar algoritmos sobre as fontes de dados distribuídas. Cabe ressaltar ainda que, se a fonte de dados não estiver disponível, o acesso a ela não pode ser feito, o que não ocorre com o uso da abordagem de *data warehouse*.

A evolução de um esquema local é identificada automaticamente via agentes de *software* que acompanham permanentemente a sua evolução. Ao ser identificada a alteração de um esquema, esta é informada ao administrador, que irá proceder a uma nova captura do esquema local. Agentes de *software* são também utilizados para atualização das bases de dados presentes na arquitetura, a partir da identificação de alterações nas bases locais.

De posse do protótipo que foi desenvolvido, estudos estão sendo realizados por pesquisadores da área de biologia molecular com objetivo de resolver os problemas de integridade e de qualidade dos dados e também com intuito de descobrir novos conhecimentos. Um exemplo destes estudos, que foi implementado, trata da verificação de inconsistências das anotações biológicas que possam existir entre as fontes de dados de proteínas Pir e Swiss-Prot.

De forma resumida, as principais contribuições da tese referem-se a:

- proposta de arquitetura de integração baseada no uso de mediadores e na implementação de *data warehouses* (não usual na literatura), projetada e implementada utilizando conceitos de engenharia de software, no caso, *frameworks* orientados a objetos, até então não utilizados nos projetos de integração descritos na literatura. A arquitetura proposta é capaz de tratar a evolução dos esquemas das fontes de dados componentes da integração.

- proposta de um esquema conceitual de informações do dogma central da biologia, contendo a descrição dos objetos envolvidos e seus relacionamentos;

- construção de um protótipo da ferramenta, que demonstrou ser útil para a pesquisa na área de biologia molecular, além de eficaz na solução dos problemas de integração das fontes de dados e dos aplicativos. O protótipo vem sendo utilizado para verificar hipóteses biológicas e contribuir na busca de novos conhecimentos para a área;

- proposta de definição de uma ontologia que é efetivamente utilizada nas fontes de dados da biologia molecular, que pode ser confrontada com as propostas para a área;

- comparação entre as arquiteturas de integração existentes;

7.3 Trabalhos Futuros

Atualmente estão sendo pesquisados e construídos *wrappers* para outras fontes de dados biológicas, de forma a aumentar a capacidade de expressão e, consequentemente, a utilidade da ferramenta. Outros *drivers* também estão sendo pesquisados ou com implementação prevista com o mesmo objetivo, sendo prioritários os que permitem a exibição de mapas de cromossomos e de dados de *microarray* de DNA.

Ainda com objetivo de tornar a ferramenta robusta, pretende-se empreender uma busca de aplicativos na *Web* que tenham código aberto, com objetivo de implementar os algoritmos envolvidos como métodos do esquema global.

Caso o desempenho das consultas não atenda às exigências da pesquisa, pode-se introduzir uma camada de transformação do formato do objeto (texto para binário) antes de seu armazenamento no repositório. Desta forma, evitar-se-ia a necessidade de *parsing* ao objeto no formato texto e seria reduzido o tamanho da base de dados com objetivo de agilizar as consultas.

No código fixo da arquitetura (*frozen spots*) o mediador exige grande esforço de implementação de forma a permitir que sejam reconhecidas as fontes de dados e os objetos envolvidos e que as instâncias sejam capturadas (no repositório ou na própria fonte de dados) de modo a instanciar um esquema específico para uma dada pesquisa.

Está em andamento o projeto de uma QBE (*query by example*) inserida em uma página *Web*, no sentido de facilitar a formulação de consultas sofisticadas, por usuários não especialistas em informática.

A arquitetura pode se beneficiar de técnicas de otimização de consultas baseadas em paralelismo, sendo que estas técnicas estão implementadas na ferramenta atualmente em uso no repositório (Oracle 9i).

Uma linha de pesquisa a ser adotada trata da transformação automática do esquema global para descrições lógicas, de forma a dotar a arquitetura da capacidade de realizar inferências sobre os dados. Essas descrições lógicas irão formar uma ontologia de mais alto nível com relação àquela anterior, baseada na descrição do esquema global. De posse desta ferramenta poder-se-á exercitar questionamentos aos dados para investigar comportamentos biológicos.

Em termos de continuidade da pesquisa, pretende-se ainda investigar a formulação de hipóteses biológicas sobre as bases de dados resultantes do processo de integração.

Anexo I - Conceitos Básicos de Biologia

Este documento trata apenas de uma parte da biologia computacional, aquela relativa ao armazenamento, tratamento, recuperação e disponibilização das informações relevantes à pesquisa em biologia molecular. As informações e definições tratadas neste anexo são baseadas em [Doo90, Let99, GJ02, BO98].

As informações básicas de biologia referem-se a sequências de caracteres que representam ácidos nucléicos (DNA e RNA) e proteínas. Os ácidos nucléicos são responsáveis pela hereditariedade e as proteínas por reações bioquímicas que ocorrem nos seres vivos. Estão localizados nas células, que são as responsáveis pela estrutura e pelas funções básicas dos organismos. A Figura 17 apresentada a seguir mostra os componentes de uma célula.

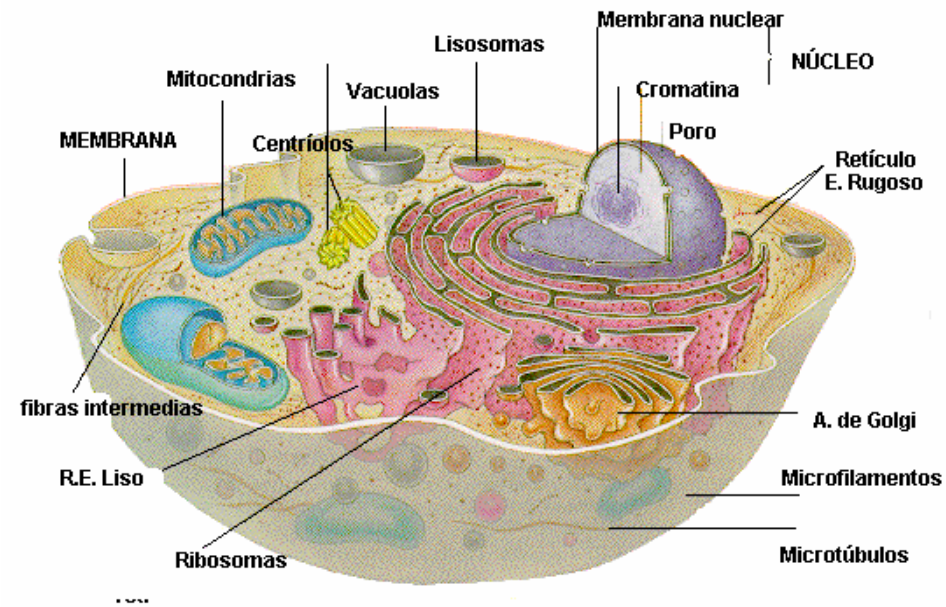

Figura 17 - Componentes de uma célula

O DNA (ácido desoxirribonucléico) encontra-se principalmente no núcleo das células e é composto de nucleotídeos dispostos em dois filamentos unidos, antiparalelos, com forma helicoidal. Estes dois filamentos estão unidos pela formação de pontes de hidrogênio (um tipo de ligação química) entre os nucleotídeos das duas fitas, como mostra a Figura 18 apresentada a seguir. A função principal do DNA é o armazenamento de informaçao genética, pois esta molécula é bastante estável nas células.

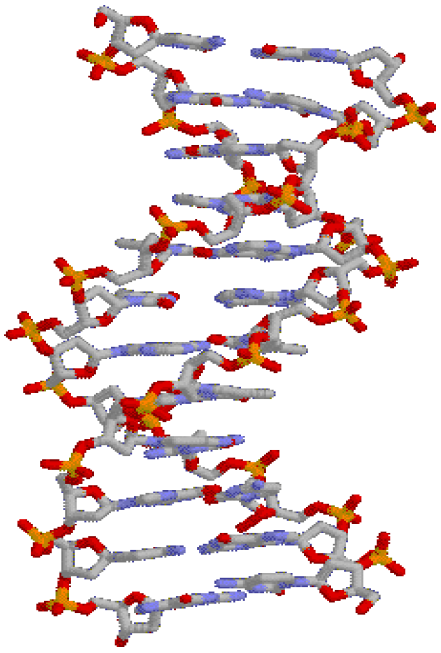

Figura 18 - Estrutura do DNA

Um nucleotídeo é formado por três diferentes tipos de moléculas: uma pentose (açúcar de 5 carbonos), um grupo fosfato e uma base nitrogenada. Os nucleotídeos são: Adenina, Timina, Citosina e Guanina e são representados pelas letras A, T, C e G respectivamente. A composiçào de cada nucleotídeo é apresentada na Figura 19.

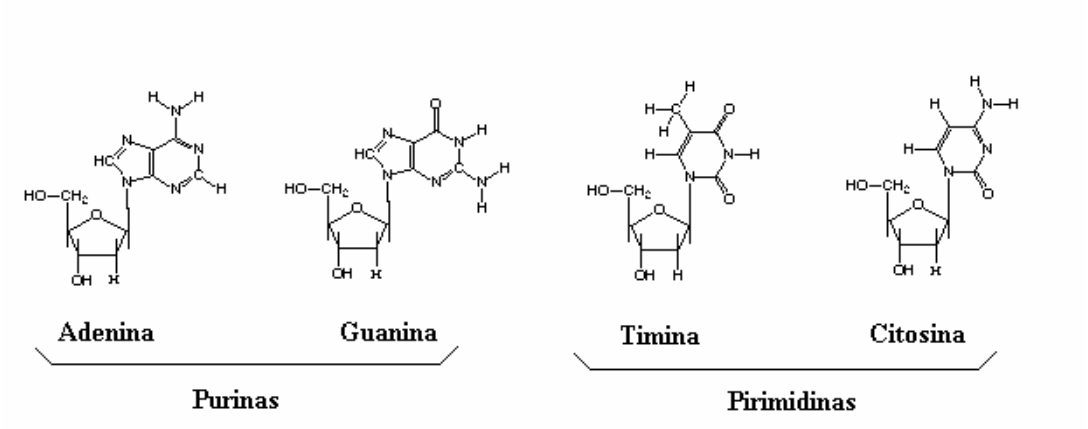

Figura 19 - Os nucleotídeos do DNA

As bases A e T são complementares, em virtude de que uma base A em um filamento (também chamado de fita) corresponde a uma base T na outra e vice-versa. O mesmo ocorre com as bases C e G, sendo que um par A-T forma somente duas pontes de hidrogênio enquanto um par G-C forma três pontes de hidrogênio, como mostrado na Figura 20.

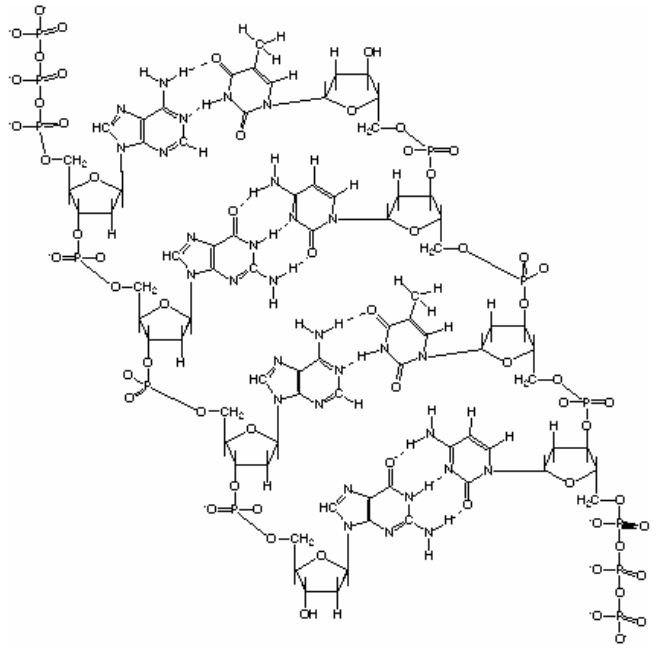

Figura 20 - A ligação dos filamentos via pontes de hidrogênio

O RNA (ácido ribonucléico) é formado a partir do DNA em um processo denominado transcrição. No RNA a Timina é substituída pela Uracila, representada pela letra U. O RNA possui uma pentose chamada ribose e encontra-se no núcleo e no citoplasma. As moléculas de RNA são intermediários na transmissão da informação armazenada no DNA para a sua forma ativa (as proteínas). Participam diretamente de vários processos catalíticos.

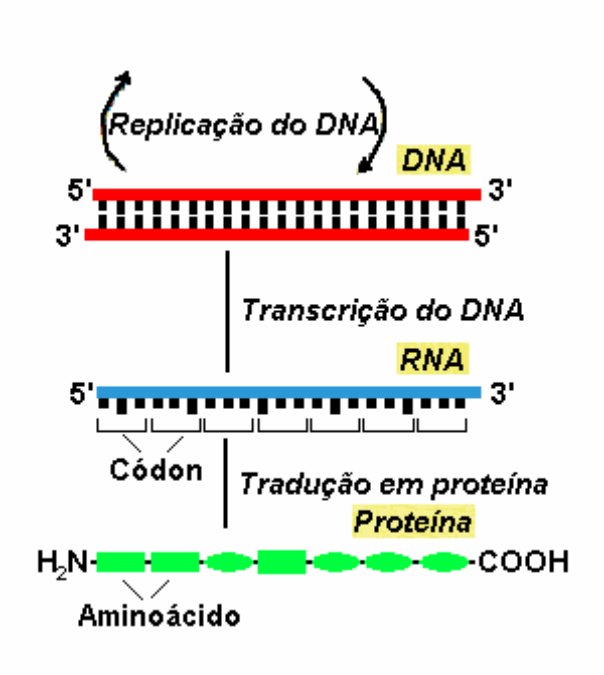

Figura 21 - Dogma central da biologia molecular

Uma proteína é formada a partir do RNA em um processo chamado de tradução e é composta por aminoácidos, cada um especificado no DNA por três nucleotídeos. A esse trio de nucleotídeos é dado o nome de códon, que são unidades contendo o código de informação para um aminoácido. A Figura 21 apresenta um esquema dos processos de transcrição e tradução.

Existem trios de nucleotídeos que especificam o mesmo aminoácido. Assim, todas as combinações possíveis de nucleotídeos formam 64 possibilidades de códons mas normalmente apenas 20 aminoácidos são utilizados, que podem ser representados por

letras do alfabeto. A tabela que mostra a correspondência entre os códons e os aminoácidos é denominada código genético e é apresentada na Figura 22.

Figura 22 - Tabela do código genético

Todas as regiões do DNA contendo algum tipo de informação são chamadas de gene. Existem três tipos de genes: os transcritos e traduzidos, que dão origem às proteínas, os transcritos mas não traduzidos, que dão origem a RNAs (RNA ribossomal, RNA transportador, etc), e finalmente os não transcritos, que podem ser regiões promotoras ou reguladoras da expressão gênica por regularem sua produção (vide Figura 23).

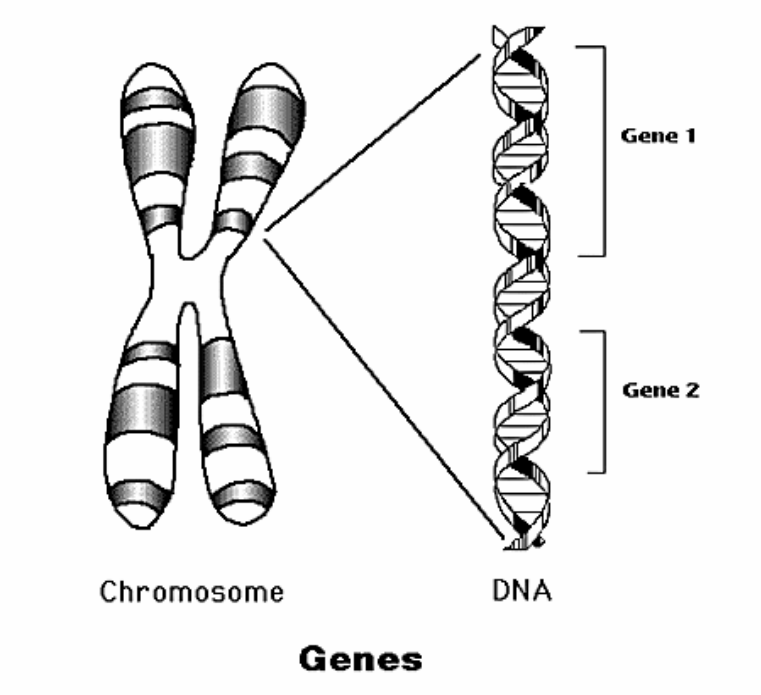

Figura 23 - Genes

Os genes que contém as instruções para a construção de uma proteína são do primeiro tipo. Dependendo do gene e/ou do organismo em questão, o gene pode ser dividido mais uma vez em duas regiões distintas: os éxons e os íntrons. As regiões codificantes (ou seja, as regiões contendo informação relevante para a produção de uma determinada proteína) são chamadas éxons e as não-codificantes são os íntrons, os quais são retirados em etapa posterior do processamento da informação pela célula (vide Figura 24).

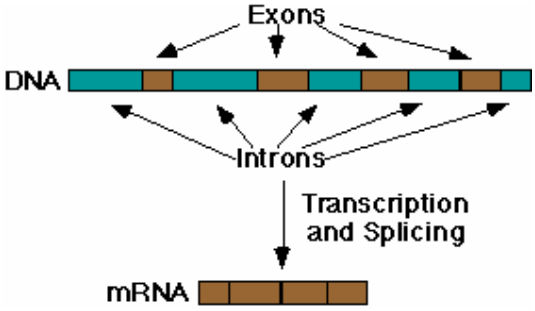

Figura 24 - Região Codificante: Éxons e Introns

O genoma é o conteúdo de todo o DNA presente em uma célula, incluindo os genes e as regiões intergênicas (as quais também são não-codificantes). A determinação da sequência exata dos nucleotídeos que formam o DNA, ou dos aminoácidos que formam uma proteína, é chamada de sequenciamento.

A técnica de sequenciamento de DNA consiste em se partir a molécula de ácido nucléico em partes e realizar experimentos físico-químicos nestas partes de modo a se conhecer os nucleotídeos que a formam. O sequenciamento de proteínas segue a mesma idéia básica. Esta técnica tem seus limites, sendo possível o sequenciamento de uma molécula de DNA de até aproximadamente 600 nucleotídeos. Portanto, para que o sequenciamento completo de um genoma seja possível, é necessário o sequenciamento de milhares de partes do mesmo em separado.

Em seguida essas partes são fornecidas como entrada para um algoritmo especial que visa reunir novamente as partes de forma a se conhecer o todo. Esse algoritmo é denominado *sequence assembly* ou "montagem de fragmentos" [TIGR02]. Existem também algoritmos especiais para investigação de regiões codificantes [SS97, GS93], para análises de mutações (substituições, remoções ou inserções pontuais de nucleotídeos, e também rearranjos ou trocas na ordem de grandes segmentos de DNA) [KTN94] e de filogenias (estudos visando a determinação do grau de parentesco entre os seres vivos) [KTN94].

Anexo II - Glossário dos termos utilizados nos modelos de genoma e de proteoma (dogma central da biologia)

- Centrômero: Ponto de contato com o fuso mitótico (conjunto de fibras); a partir deste ponto o cromossomo é puxado por fibras para o centrossomo, durante a mitose.
- Cromossomo: Estrutura nuclear, composta em sua maior parte de cromatina, na qual os genes eucarioticos estão organizados.
- DNAOrganelas: DNA contido em inclusões delimitadas por membranas no citoplasma celular.
- Domínio: Região da proteína, com aminoácidos contíguos, com características estruturais distintivas.
- ElementoCromossomal: Região característica da estrutura do cromossomo.
- ElementoExtraCromossomal: DNA não localizado no cromossomo.
- EstruturaSecundária: Enovelamento da proteína em padrões como α-hélice, cadeias β, etc.
- EstruturaTerciária: Estrutura tri-dimensional da proteína.
- Éxon: Parte de um gene que é traduzida em proteína.
- Família: Grupo de proteínas relacionadas evolutivamente.
- Genoma: conteúdo total de DNA em uma célula de um organismo.
- Íntron: Parte de um gene que não é traduzida em proteína, mas é transcrita.
- mRNA: RNA mensageiro maduro.
- Operon: Unidade completa de regulação e expressão de genes bacterianos incluindo gene(s) estruturais, regulatórios e elementos controladores no DNA.
- ORI: Local no DNA onde se inicia o processo de replicação.
- Palíndromo: Sequência de nucleotídeos que é a mesma quando lida em qualquer orientação.
- PeptídeoPrimário: Sequência protéica imediatamente produzida após a tradução.

- Plasmídio: DNA dupla fita, normalmente circular, extracromossomal, capaz de replicação dentro de uma célula, contendo normalmente genes para resistência a antibióticos.
- Profago: Sequência viral inativa, inserida no genoma do hospedeiro.
- Promotor: Sequência específica no DNA reconhecida pela RNA polimerase.
- Promotor_l: Região regulatória do processo de tradução.
- Proteína: Sequência protéica madura (já com suas modificações pós-traducionais).
- PseudoGene: Componentes inativos do genoma derivados por duplicação e mutação de um gene ancestral.
- RBS: Local de ligação ao Ribossoma.
- RegiãoRepetitiva: Sequência repetitiva de DNA.
- RepetiçãoDireta: Repetição de uma porção de DNA na mesma orientação.
- RepetiçãoInversa: Repetição de uma porção de DNA em orientações distintas.
- SequênciaRegulatória: Região do DNA que controla a expressão de um determinado gene.
- Sítio: Região de uma proteína, não necessariamente formada por aminoácidos contíguos, que irá interagir com outra molécula.
- SítioPoliA: Local onde uma cadeia de adeninas é adicionada ao mRNA.
- Telômero: Estrutura repetitiva encontrada na extremidade dos cromossomos.
- Terminador: Sequência cuja função é terminar a transcrição pela RNA polimerase.
- Terminador_l: Região regulatória da estabilidade do mRNA.
- Transcrito: Molécula de RNA produzida a partir do DNA (Transcrito Primário).
- Transposon: Elemento genético transponível, ou seja, capaz de se mover de um local para outro em uma molécula de DNA.
- UTR: Regiões não traduzidas das extremidades de um transcrito.
- UTR3_l: Regiões não traduzidas da extremidade 3’ de um transcrito.
- UTR5_l: Regiões não traduzidas da extremidade 5’ de um transcrito.

Referências Bibliográficas

A

[AceDB02] "ACeDB -- A C elegans Database".
Disponível: http://elegans.swmed.edu/genome.shtml, 2002.

[ACNUC02] "The ACNUC Database Management System".
Disponível: http://pbil.univ-lyon1.fr/databases/acnuc.html, 2002.

[AG97] Ashburner, M., Goodman, N.. "Informatics – Genome and Genetics Databases". *Current Opinion in Genetics & Development*, 7, pp. 750-756, 1997.

[AGM+90] Altschul, S. F., Gish,W., Miller, W., Myers, E. W., Lipman, D. J.. "A Basic Local Alignment Search Tool". *J. of Molecular Biology*, 215, pp. 403-410, 1990.

[AGRICOLA02] "AGRICOLA".
Disponível: http://www.nal.usda.gov/ag98/, 2002.

[AMS+97] Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller,W., Lipman, D. J.. "Gapped blast and psi-blast: a new generation of protein database search programs". *Nucleic Acids Research*, 25(17), pp. 3389-3402, 1997.

[ANSI02] "American National Standards Institute".
Disponível: http://www.ansi.org, 2002.

[ArrayExpress02] "ArrayExpress".
Disponível:
http://www.ebi.ac.uk/microarray/ArrayExpress/arrayexpress.html, 2002.

[ASN02] "Abstract Syntax Notation number One (ASN.1)". Disponível: http://asn1.elibel.tm.fr/en/introduction/index.htm, 2002.

[AVB01] Achard, F., Vaysseix, G., Biríllot, E.. "XML, bioinformatics and data integration". *Bioinformatics*, 17, pp. 115-125, 2001.

B

[BA00] Bairoch, A., Apweiler, R.. **"**The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000"**.** *Nucleic Acids Research,* 28, pp. 45-48, 2000.

[BBB+98] Baker, P.G., Brass, A., Bechhofer, S., Goble, C., Paton, N., Stevens, R.. "TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources". *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology* (ISMB98), 1998.

[BBC+00] Baker W., Van den Broek A., Camon E., Hingamp P., Sterk P., Stoesser G., Tuli M.A., The EMBL Nucleotide Sequence Database, *Nucleic Acids Research,* 28(1), pp. 19-23, 2000.

[BCD+98] Buneman, P., Crabtree, J., Davidson, S., Tannen, V., L. Wong.. "BioKleisli". *Bioinformatics*, ed. S. Letovsky, Kluwer Academic Publishers, 1998.

[BCM02] "BCM Search Launcher: Multiple Sequence Alignments." Disponível: http://dot.imgen.bcm.tmc.edu:9331/multi-align/multi-align.html, 2002.

[BDH+95] Buneman, P., Davidson, S., Hart, K., Overton, C., L. Wong. "A Data Transformation System for Biological Data Sources". *Proceedings of 21th International Conference on Very Large Data Bases*, pp.158-169, 1995.

[Ben99] Benson, G.. “Tandem repeats finder: a program to analyze DNA sequences”. *Nucleic Acids Research*, 27(2), pp. 573-80, 1999.

[BGB+99] Baker, P.G., Goble, C.A., Bechhofer, S., Paton, N.W., Stevens, R., Brass, A.. “An Ontology for Bioinformatics Applications”. *Bioinformatics*, 15(6), pp. 510-520, 1999.

[BGH+00] Barker W.C., Garavelli J.S., Huang H., McGarvey P.B., Orcutt B.C., Srinivasarao G.Y., Xiao C., Yeh L.L., Ledley R.S., Janda J.F., Pfeiffer F., Mewes H., Tsugita A., Wu C., The Protein Information Resource (PIR), *Nucleic Acids Research* 28(1), pp. 41-44, 2000.

[BH01] Braganholo, V. P., Heuser, C.A.. “XML Schema, RDF(S) e UML: uma comparação”. *IDEAS'2001 (IV Worksho Iberoamericano de Ingeniería de Requisitos y Ambientes de Software)*, 2001.

[Big90] Bigonha, M.. “O papel da representação do conhecimento na construção de Sistemas Especialistas”. *Monografia em Ciência da Computação n° 4/90*. Departamento de Informática PUC-Rio, 1990.

[BKL02] Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., Rapp, B. A., Wheeler, D.L.. “GenBank”. *Nucleic Acids Research*, 30, pp. 17-20, 2002.

[BLOCKS02] “Blocks”.
Disponível: http://www.blocks.fhcrc.org/, 2002.

[BML+00] Benson D.A., Karsch-Mizrachi I., Lipman D.J., Ostell J., Rapp B.A., Wheeler D.L.. “GenBank”. *Nucleic Acids Research*, 28(1), pp. 15-18, 2000.

[BMR+96] Bushmann, F., Meunier, R., Rohnert, H., Sommerlad, P, Stal, M..

"Pattern-Oriented Software Architecture: A System of Patterns". John Willey & Sons, pp.457, 1996.

[BO98] Baxevanis, A., Ouellette, B.. "Bioinformatics: A pratical guide to the analysis of genes and proteins". *Editora Wiley-Interscience*, 1998.

[Boo94] Booch G., Designing an Application Framework, Dr. Dobb´s Journal 19(2), 1994.

[Bor95] Borgida, A.. "Description Logics in Data Management". *IEEE Transactions on Knowledge an Data Engineering*, vol. 5, n. 5, 1995.

[BP01] Barbosa, A., Porto, F.. "Configurable Data Integration Middleware System". *Proc. of the International Workshop on Information Integration on the Web - WIIW*, pp. 74-80, 2001.

[BRJ99] Booch, G., Rumbaugh, J., Jacobson, I.. "The Unified Modeling Language User Guide". Addison-Wesley Longman, 1999.

[BSN+99] Bechhofer, S., Stevens, R., Ng, G., Jacoby, A., Goble, C.. "Guiding the User: An Ontology Driven Interface. *Proc. User Interfaces to Data Intensive Systems* (UIDIS99), ed. Norman W. Paton and Tony Griffiths, IEEE Press, pp. 158-16, 1999.

[BWF+00] Berman, H.M., Westbrook, J., Feng, Z., Gillil, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.. "The Protein Data Bank". *Nucleic Acids Research*, 28(1), pp. 235-242, 2000.

C

[Carnot02] "Carnot Project".
Disponível: http://www.mcc.com/projects/carnot/carnot-paper.html, 2002.

[CGK98] Corpet, F., Gouzy, J., Kahn, D.. “The ProDom database of protein domain families”. *Nucleic Acids Research*, 26, pp. 323-326, 1998.

[CHS+97] Codenie, W., Hondt, K.D., Steyaert, P., Vercammen, A.. “From Custom Applications to Domain-Specific Frameworks”. *Communications of the ACM*, vol. 40, n.10, pp. 71-77, 1997.

[CI93] Campbell, R.H., Islam, N.. “A Technique for Documenting the Framework of an Object-Oriented System”. *Computing Systems*, vol. 33, N. 1, pp. 70-80, 1993.

[CKM+95] Chen, I.A., Kosky, A., Markowitz, V.M., Szeto, E. “OPM*QS: The Object-Protocol Model Multidatabase Query System”. *Technical Report* LBNL-38181, 1995.

[CKM+96] Chen, I.A., Kosky, A., Markowitz, V.M., Szeto, E.. “OPM Query Translator”. *Technical Report* LBNL-38180, 1996.

[ClustalW02] “ClustalW - Multiple Sequence Alignment”.
Disponível:
http://dot.imgen.bcm.tmc.edu:9331/multi-align/Options/clustalw.html, 2002.

[CM95a] Chen, I.A., and Markowitz, V.M.. “Modeling Scientific Experiments with an Object Data Model”. Proc. of the 11th Int. Conference on Data Engineering, 1995.

[CM95b] Chen, I.A., Markowitz, V.M.. “An Overview of the Object-Protocol Model (OPM) and OPM Data Management Tools”. *Information Systems*, 20(5), 1995.

[CM96a] Chen, I.A., Markowitz, V.M.. “The Object-Protocol Model (Version

4)". *Technical Report* LBNL-32738, 1996.

[CM96b] Chen, I.A., Markowitz, V.M.. "OPM Schema Translator (OPM Version 4)". *Technical Report* LBNL-35582, 1996.

[CM96c] Chen, I.A., Markowitz, V.M.. "Mapping OPM Schemas into Relational Database Schemas and Procedures". *Technical Report* LBNL-33048, 1996.

[COG02] "Cluster of Orthologous Groups".
Disponível: http://www.ncbi.nlm.nih.gov/COG/, 2002.

[Cos02] Costa, R.."Alocação de Dados e Distruição de Carga para execução Paralela da Estratégia de BLAST para comparação de sequências". *Dissertação de Mestrado*. Departamento de Informática PUC-RIO, 2002.

[CPL02] "Collection Programming Language - PENN Database Research Group". Disponível: http://db.cis.upenn.edu/, 2002.

[CPW+01] Cornell, M., Paton, N.W., Wu, S., Goble, C.A., Miller, C.J., Kirby, P., Eilbeck, K., Brass, A., Hayes, A. and Oliver, S.G.. "GIMS - A Data Warehouse for Storage and Analysis of Genome Sequence and Functional Data". *Proc. 2nd IEEE International Symposium on Bioinformatics and Bioengineering* (BIBE), IEEE Press, pp.15-22, 2001.

[CRJ87] Campbell, R.H., Russo, V.F., Johnston, G.M.. "The Design of a Multiprocessor Operating System". *Proc. Of the USENIX C++ Workshop*, 1987.

[CSG00] Corpet, F., Servant, F., Gouzy, J., Kahn, D.. "ProDom and ProDom-

CG: tools for protein domain analysis and whole genome comparisons". *Nucleic Acids Research*, 28, pp.267-269, 2000.

[CWZ98] Chen, J., Wong, L., Louxin, Z.. "A Protein Patent Query System Powered By Kleisli". *Communications of ACM*, pp.593-595, 1998.

D

[dbEST02] "dbEST: database of Expressed Sequence Tags". Disponível: http://www.ncbi.nlm.nih.gov/dbEST/index.html, 2002.

[DCB01] Davidson, S. B., Crabtree, J., Brunk, B. P., Schug, J., Tannen, V., Overton, G. C., C. J. Stoeckert, Jr.. "K2/Kleisli and GUS: Experiments in integrated access to genomic data sources".IBM Systems Journal, 40(2), 2001.
Disponível: http://www.research.ibm.com/journal/sj/402/davidson.html, 2002.

[DCB02] Davidson, S., Crabtree, J., Brunk, B., Schug, J., Tannen, V., Overton, C., Stoeckert, C.. "K2/Kleisli and GUS: Experiments in integrated access to genomic data sources". *IBM Systems Journal*, 2001.

[DOB95] Davidson, S., Overton, C., Buneman, P.. "Challenges in Integrating Biological Data Sources". *Journal of Computational Biology*, 2(4), pp. 557-572, 1995.

[DOE02a] "DOE - U.S. Department of Energy. Human Genome Research". Disponível: http://www.er.doe.gov, 2002.

[DOE02b] "DOE - U.S. Department of Energy. Human Genome Project Information". Disponível: http://www.ornl.gov/hgmis/, 2002.

[DOE02c] "DOE - U.S. Department of Energy. Human Genome Research".

Disponível: http://www.science.doe.gov/ober/hug_top.html, 2002.

[Doo90] Doolittle, R.F.. Ed. Methods in Enzymology, Academic Press, 1990.

[DSSP02a] "DSSP: Database of Secondary Structure in Proteins".
Disponível: http://www.sander.ebi.ac.uk/dssp/, 2002.

[DSSP02b] "The DSSP database".
Disponível: http://www.cmbi.kun.nl/gv/dssp/,2002.

[Dub00] Dubuisson, O. "ASN.1 - Communication between heterogeneous systems". Morgan Kaufmann Publishers.
Disponível: http://www.oss.com/asn1/dubuisson.html, 2002.

E

[EBI02] "European Bioinformatics Institute (EBI) - SWISS-PROT and TrEMBL". Disponível: http://www.ebi.ac.uk/swissprot/, 2002.

[EBP+99] Eilbeck, K., Brass, A., Paton, N.W., Hodgman, C.. "INTERACT: an object-oriented protein-protein interaction database". *Proc. 7th Int. Conference on Intelligent Systems for Molecular Biology* (ISMB), AAAI Press, pp. 87-94, 1999.

[EGAD02] "EGAD". Disponível: http://www.tigr.org/tdb/egad/egad.shtml, 2002.

[EMBL02] "The EMBL Nucleotide Sequence Database - European Bioinformatics Institute". Disponível: http://www.ebi.ac.uk/embl/, 2002.

[EN89] Elmasri, R., Navathe, S.. "Fundamentals of Database Systems". Addison--Wesley World Student Series. The Benjamin/Cummings Publ., Redwood City, CA, 1989.

[Entrez02] "Entrez Nucleotide".
Disponível: http://www.ncbi.nlm.nih.gov/Entrez/, 2002.

[ExPASy02a] "Enzyme nomenclature database".
Disponível: http://www.expasy.ch/enzyme/, 2002.

[ExPASy02b] "ExPASy - SWISS-PROT and TrEMBL".
Disponível: http://www.expasy.ch/sprot/, 2002.

F

[FioCruz02] "Fundação Inst. Oswaldo Cruz".
Disponível: http://www.dbbm.fiocruz.br/, 2002.

[FGM+97] Fujibuchi, W., Goto, S., Migimatsu, H., Uchiyama, I., Ogiwara, A., Akiyama, Y., Kanehisa, M.. "DBGET/LinkDB: an Integrated Database Retrieval System. Pacific Symp". *Biocomputing*, 3, pp. 683-694, 1997.

[FPB+02] Falquet,L., Pagni, M., Bucher, P., Hulo, N., Sigrist, C. J. A., Hofmann, K., Bairoch, A.. "The PROSITE database, its status in 2002". *Nucleic Acids Research*. 30, pp. 235-238, 2002.

[FS97] Fayad, M.E., Schmidt, D.C.. "Object-Oriented Application Frameworks". *Comunications of the ACM*, vol 40, n. 10, pp.32-38, 1997.

[FSJ97] Fayad, M.E., Schmidt, D.C., Johnson, R.E.. "Object-Oriented Application Frameworks: Problems and Perspectives". Wiley, 1997.

[FSJ99] Fayad M.E., Schmidt D.C. e Johnson R.E., Building Application Frameworks, Addison-Wesley, 1999.

G

[Gbrel02] "GenBank User Manual".
Disponível: http://ncbi.nlm.nih.gov/genbank/gbrel.txt, 2002.

[GCG02] "Genetics Computer Group".
Disponível: http://www.gcg.com, 2002.

[GDB02a] "The Genome Database (GDB) Blast Search".
Disponível: http://www.gdb.org/gdb/seqBlast.html, 2002.

[GDB02b] "The Genome Database". Disponível: http://www.gdb.org/, 2002.

[GenBank02] "GenBank Sequence Database".
Disponível: http://www.ncbi.nlm.nih.gov/GenBank/index.html, 2002.

[GHJ+93] Gamma, E., Helm, R., Johnson, R., Vlissides, J.. "Design Patterns: Abstractions and reuse of object-oriented design". *Proc. of European Conference on Object-Oriented Programming (ECOOP'93)*, Lecture Notes in Computer Science, n. 707, Spring-Verlag, 1993.

[GHJ+95] Gamma E., Helm R., Johnson R. and Vlissides J., Design Patterns: Elements of reusable object-oriented software, Addison-Wesley Longman, 1995.

[GIMS02a] "Genome Information Management System".
Disponível: http://www.cs.man.ac.uk/~norm/gims/, 2002.

[GIMS02b] "School of Biological Sciences – UK".
Disponível: http://www.biomed.man.ac.uk/, 2002.

[GIMS02c] "The Information Management Group".
Disponível: http://img.cs.man.ac.uk, 2002.

[GIMS02d] "Computer Science - University of Manchester". Disponível: http://www.cs.man.ac.uk/, 2002.

[GJ00] Guerrinia, V, Jackson D.. "Bioinformatics and XML". *On Line Journal of Bioinformatics*, 1(1), pp. 1-13, 2000.

[GJ02] Gibas, C., Jambeck, P.. "Desenvolvendo Bioinformática". *Editora Campus e O´Reilly*, 2002.

[GO02] "Geneontology". Disponível: http://www.geneontology.org/, 2002.

[GOA02] "GOA". Disponível: http://www.cos.ufrj.br/~goa/, 2002.

[Goo94] Goodman, N. (1994). An object oriented DBMS war story: Developing a genome mapping database in C++. In Kim, W., editor, *Modern Database Management: Object-Oriented and Multidatabase Technologies*. ACM Press.

[GS93] Gish, W., States, D.J.. "Identification of protein coding regions by database similarity search". *Nature Genetics*, 3(3), pp. 266-72, 1993.

H

[HCF+00] Harger, C., Chen, G., Farmer, A., Huang, W., Inman, J., Kiphart, D., Schilkey, F., Skupski, M. P., Weller, J.. "The Genome Sequence DataBase"**.** *Nucleic Acids Research*, 28, pp. 31-32, 2000.

[HGD02] "Welcome to the GenomeWeb - Human Genome Databases". Disponível: http://www.hgmp.mrc.ac.uk/GenomeWeb/human-gen-db-genome.html, 2002.

[HMD02] "GenomeWeb - Human Mutation Databases". Disponível:

http://www.hgmp.mrc.ac.uk/GenomeWeb/human-gen-db-mutation.html, 2002.

[HSB+98] Harger, C., Skupski, M., Bingham, J., Farmer, A., Hoisie, S., Hraber, P., Kiphart, D., Krakowski, L., McLeod, M., Schwertfeger, J., Seluja, G., Siepel, A., Singh, G., Stamper, D., Steadman, P., Thayer, N., Thompson, R., Wargo, P., Waugh, M., Zhuang, J.J., Schad, P.A.. "The Genome Sequence DataBase (GSDB): improving data quality and data access". *Nucleic Acids Research*, 26, pp.21-26, 1998.

[HZ96] Hull R., Zhou G., "A Framework for Supporting Data Integration Using the Materialized and Virtual Approaches".SIGMOD Conference 1996: 481-492.

I

[IGD02a] "Integrated Genomic Databases".
Disponível: http://genome.dkfz-heidelberg.de/igd-gis/, 2002.

[IGD02b] "Integrated Genomic Databases".
Disponível:http://www-leibniz.imag.fr/GDR-INFOGENOMES/JC-ritter.html, 2002.

[IMG02] "The Information Management Group".
Disponível: http://img.cs.man.ac.uk, 2002.

[Infomaster02] "Infomaster".
Disponível: http://infomaster.stanford.edu/infomaster-info.html, 2002.

[Informix02] "Informix".
Disponível: http://www-3.ibm.com/software/data/informix/, 2002.

[INSD02] "International Nucleotide Sequence Databank". Disponível:

http://www.ncbi.nlm.nih.gov/Genbank/GenbankOverview.html, 2002.

[IOPI02] "International Organization for Plant Information". Disponível: http://iopi.csu.edu.au/iopi/iopigpc1.html, 2002.

[ITIS02] "Integrated Taxonomic Information System". Disponível: http://www.itis.usda.gov/, 2002.

J

[JAVA02] The Source for Java(TM) Technology. Disponível: http://java.sun.com/, 2002.

[JAVABEANS02] JavaBeans(TM). Disponível: http://java.sun.com/products/javabeans/, 2002.

[JAVACC02] WebGAIN(TM). Disponível: http://www.webgain.com, 2002.

[JHU02] "The Johns Hopkins University, School of Medicine". Disponível: http://infonet.welch.jhu.edu/som/, 2002.

K

[K202a] "K2". Disponível: http://db.cis.upenn.edu/K2/, 2002.

[K202b] "K2". Disponível: http://db.cis.upenn.edu/K2/papers.html, 2002.

[Kar95] Karp, P.D.. "A Strategy for Database Interoperation". *Journal of Computational Biology*, 2(4), pp. 573-586, 1995.

[Kar98] Karp P., "What We Do Not Know About Sequence Analysis and Sequence Databases", Bioinformatics vol.14, No. 9, pp 753-754, 1998.

[KBT+01] Kalinichenko L.A., Briukhov D.O., Tyurin I.N., Skvortsov N.A., ”Intermediator Framework Protocol for Information Sources Registration at Heterogeneous Mediators”, In proceedings of the DELOS Workshop Interoperability in Digital Libraries, September 8-9,2001, GMD-IPSI, Darmstadt, Germany.

[KLB+02] Kogelnik, A.M., Lott, M.T., Brown, M.D., Navathe, S.B., Wallace, D.C.. “MITOMAP: a human mitochondrial genome database--1998 update”. *Nucleic Acids Research,* 26, pp. 112-115, 1998.

[KP88] Krasner, G.E., Pope, S.T.. “A Cook Book for Using the Model-ViewController User Interface Paradigm in SmallTalk-80”. *Journal of Object-Oriented Programming*, Vol. 1, N. 3, pp. 26-49, 1988.

[KRP+98] Karp, P., Riley, M., Paley, S., Pellegrini-Toole, A.. “Ecocyc: Eletronic Encyclopedia of E.coli genes and metabolism”. Nucleic Acids Res., 26 (50), 1998.

[KRT96] Karp, R., Ruzzo, L., Tompa, M.. “Algorithms in Molecular Biology”. Disponível: http://www.cs.washington.edu/education/courses/590bi/96wi/, 1996.

[KS83] Kabsch, W., Sander, C.. “Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features”. *Biopolymers*, 22, pp. 2577-2637, 1983.

[KSC+96] Kosky, A., Szeto, E., Chen, I.A., Markowitz, V.M.. “OPM Data Management Tools for CORBA Compliant Environments”. *Technical Report* LBNL-38975, 1996.

[KTN94] Kumar, S., Tamura, K., Nei, M.. “MEGA: Molecular Evolutionary Genetics Analysis software for microcomputers”. *Computer*

Applications in Biosciences, 10(2), pp.189-91, 1994.

L

[LBNL02] "Lawrence Berkeley National Laboratory". Disponível: http://www.lbl.gov/, 2002.

[LCP98] Letovsky, S.I., Cottingham, R.W., Porter, C.J., Li, P.W.D.. "GDB: the Human Genome Database". *Nucleic Acids Research*, 26, pp. 94-99, 1998.

[Lem00] Lemos, M.. "Gerenciamento de Memória para Comparação de Biossequências", Dissertação de Mestrado, Departamento.de Informática, PUC-Rio, 2000.

[Lev98] Levy A., "The Information Manifold Approach to Data Integration", Dept. of Computer Science and Engineering, University of Washington, 1998.

[Let99] Letovsky, S.."Bioinformatics: Databases and Systems". ed. S. Letovsky, Kluwer Academic Publishers, 1998.

[LIGAND02] "Ligand-Protein Docking". Disponível: http://bioinfo.weizmann.ac.il:8500/oca-bin/lpccsu, 2002.

[LinkDB02] **"**DBGET/LinkDB - Integrated database retrieval system". Disponível: http://www.genome.ad.jp/dbget/dbget.html , 2002.

[LKS+92] Lamperti, E.D., Kittelberger, J.M., Smith, T.F., Villa-Komaroff, L.. "Corruption of genomic databases with anomalous sequence". *Nucleic Acids Research*, 20(11):2741-2747, 1992

[LMB92] Levine, J.R., Mason, T., Brown, D.. "lex & yacc, 2nd Edition". O'Reilly

& Associates, ISBN: 1565920007, 1992.

[LSU01] Lifschitz, S., Seibel, L.F.B., Uchôa, E.M.A.. "A Framework for Molecular Biology Data Integration", *Procs. Workshop on Information Integration on the Web* (WIIW), pp. 27-34, 2001.

M

[MAJ+00] Miranda A.B., Alvarez-Valin, F., Jabbari, K., Degrave, W., Bernardi, G.. "Gene Expression, Amino Acid Conservation, and Hydrophobicity Are The main Factors Shaping Codon Preferences in *Mycobacterium tuberculosis* and *Mycobacterium Leprae*". *Journal of Molecular Evolution*, 50:45-55, 2000.

[Mar95] Markowitz, V.M.. "Coping with Data Modeling Diversity". *The 2nd Meeting on Interconnection of Molecular Biology Databases*, 1995.

[MDK+00] Mello, R. S., Dorneles, C.F., Kade, A., Braganholo, V.P., Heuser, C.A.. "Dados Semi-Estruturados". *SBBD*, pp. 475-513, 2000.

[MH01] Mello, R., Heuser, C.. "A Bottom-up Approach for Integration of XML Sources". *Proc. of the International Workshop on Information Integration on the Web - WIIW*, pp. 118-124, 2001.

[MITOMAP02] "MITOMAP - A Human Mitochondrial Genome Database". Disponível: http://www.gen.emory. edu/MITOMAP, 2002.

[ML00] Markiewicz, M., Lucena, C.. "Understanding Object-Oriented Framework Engineering". *Monografia em Ciência da Computação n° 38/00*. Departamento de Informática PUC-RIO, 2000.

[MLD95] Mougenot I., Libourel T., Déhais P., "Genetic Sequence Annotation within Biological Databases", Proc. of the Fourth International

Conference on Database Systems for Advanced Applications (DASFAA'95), pp. 333-341, 1995.

[MPH99] Moussouni, F., Paton, N.W., Hayes, A., Oliver, S., Goble, C.A. and Brass, A.. "Database Challenges for Genome Information in the Post Sequencing Phase". *Proc. 10th Database and Expert Systems Applications* (DEXA), ed. T. Bench-Capon et al., Springer-Verlag, pp. 540-549, 1999.

[MR95] Markowitz, V.M., Ritter, O. "Characterizing Heterogeneous Molecular Biology Database Systems". *Journal of Computational Biology*, 2(4), 1995.

[MS93a] Markowitz, V.M., Shoshani, A., "An Overview of the Lawrence Berkeley Laboratory Extended Entity-Relationship Database Tools, TR LBL-34932, Lawrence Berkeley National Laboratory, 1993.

[MS93b] Markowitz, V.M., Shoshani A., "Object Queries over Relational Databases: Language, Implementation, and Applications, *Proc. of the 9th Int. Conference on Data Engineering, pp. 71-80, 1993.*

[MS97] Meidanis, J., Setúbal, J.C.. "Introduction to Computational Molecular Biology". *PWS Publishing Company*, 1997.

[MSAP02] "Multiple Sequence Alignment for Proteins".
Disponível: http://stateslab.bioinformatics.med.umich.edu/msa/, 2002.

[NCBI02a] "National Center for Biotechnology Information (NCBI)".
Disponível: http://www.ncbi.nlm.nih.gov/, 2002.

[NCBI02b] "National Center for Biotechnology Information (NCBI) Blast".
Disponível: http://ww.ncbi.nlm.nih.gov/BLAST/, 2002.

[NCGR02] **“**National Center of Genome Resources (NCGR) - The Genome Sequence DataBase”. Disponível: http://www.ncgr.org/, 2002.

[NK99] Navathe, S., Kogelnik, M.. “The Challenges of Modeling Biological Information for Genome Databases”, *Conceptual Modeling*, LNCS 1565, Springer-Verlag, pp. 168-182, 1999.

[NW70] Needleman, S.B., Wunsch, C.D.. “A general method applicable to the Search for Similarities in the Amino Acid Sequence of TwoProteins”. *Journal of Molecular Biology*, 48, pp. 443-453, 1970.

O

[OMG02] Object Management Group. Disponível: www.omg.org, 2002.

[OPM02a] “The OPM Project”.
Disponível: http://gizmo.lbl.gov/DM_TOOLS/DMTools.html, 2002.

[OPM02b] “The OPM Project”.
Disponível: http://gizmo.lbl.gov/DM_TOOLS/OPM/New.html, 2002.

[Oracle9i02] “Oracle 9i”. Disponível: http://www.oracle.com/, 2002.

[ORP+99] Oliver, J.L., Roman-Roldan, R., Perez, J., Bernaola-Galvan, P.. “SEGMENT: identifying compositional domains in DNA sequences”. *Bioinformatics*, 15(12), pp. 974-9, 1999.

[OTG+98] Okayama, T., Tamura, T., Gojobori, T., Tateno, Y., Ikeo, K., Miyasaki, S., Fukami-Kobayashi, K., Sugawara, H.. “Formal Design and Implementation of an Improved DDBJ DNA Database with a New Schema and Object-oriented Library”, *Bioinformatics*, 14, pp. 472-478, 1998.

[OV99] Özsu, M., Valduriez, P.. “Distrbuted Database Systems”. Second Edition, Prentice Hall, 1999.

P

[PDB02a] “The Protein Data Bank”. Disponível: http://www.pdb.org/, 2002.

[PDB02b] “The Protein Data Bank”. Disponível: http://www.rcsb.org/pdb/, 2002.

[Pea90] Pearson,W.R.. “Rapid and sensitive sequence comparison with FASTP and FASTA”. *Methods Enzymology,* 183, pp. 63-98, 1990.

[Pea91] 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.

[Pea94] Pearson,W.R.. “Using the FASTA program to search protein and DNA sequence databases”. *Methods in Molecular Biology,* 24, pp.307-31, 1994.

[Penn02] “Penn Center for BioInformatics”.
Disponível: http://www.pcbi.upenn.edu/, 2002.

[Pfam02] “Pfam”. Disponível: http://pfam.wustl.edu/, 2002.

[PIR02] “Protein Information Resource”.
Disponível: http://pir.georgetown.edu/, 2002.

[PKH+00] Paton, N.W., Khan, S.A., Hayes, A., Moussouni, F., Brass, A., Eilbeck, K., Goble, C.A., Hubbard, S. and Oliver, S.G., Conceptual Modelling of Genomic Information, *Bioinformatics*, 16(6), pp. 548-558, 2000.

[PL88] Pearson, W.R., Lipman, D.J.. “Improved tools for biological sequence

comparison". *Proceedings of the National Academy of Sciences U S A*, 85(8), pp 2444-8,1988.

[Pre94] Pree, W.. "Meta-Patterns – A means for capturing the essentials of reusable objected-oriented design". *Proc. of European Conference on Object-oriented Programming (ECOOP'94)*, pp. 150-162, 1994.

[ProDom02] "The ProDom protein domain database".
Disponível: http://prodes.toulouse.inra.fr/prodom/doc/prodom.html, 2002.

[PROFAM02] "PROFAM - Protein families and domains".
Disponível:http://genius.embnet.dkfz-heidelberg.de/menu/cgi-bin/xml/datdoc.cgi?xml=datdoc.xml&xsl=datdoc.ebene2.xsl#PROFAM, 2002.

[PROSITE02] "PROSITE - Database of protein families and domains".
Disponível:http://www.expasy.org/prosite/, 2002.

[ProtFam02] "ProtFam".
Disponível: http://mips.gsf.de/proj/protfam/, 2002.

[PSB+99] Paton, N.W., Stevens, R., Baker, P.G., Goble, C.A., Bechhofer, S., Brass, A.. "Query Processing in the TAMBIS Bioinformatics Source Integration System", *Proc. 11th Int. Conf. on Scientific and Statistical Databases (SSDBM)*, IEEE Press, pp.138-147, 1999.

Q

[QL02] "XML-QL: A Query Language for XML" .
Disponível: http://www.w3.org/TR/NOTE-xml-ql/, 2002.

R

[RBP+99] Ramfos A., Busse R., Platis N., et all, “An Integration Framework for Corba Objects”, Trnsactions of the SDPS, vol. 3, No. 1, pp. 27-41, 1999.

[RE01] Rivas, E., Eddy, S.R.. “Noncoding RNA gene detection using comparative sequence analysis”. *BMC Bioinformatics*, 2(1), pp. 8, 2001.

[READSEQ02] “Readseq:Read and reformat biosequences”. Disponível: http://iubio.bio.indiana.edu/soft/molbio/readseq/java/, 2002.

[REBASE02] “REBASE - The Restriction Enzyme Database”. Disponível: http://www.neb.com/rebase, 2002.

[Red02] “Accessing Databases - What is Redundancy?”. Disponível: http://twod.med.harvard.edu/seqanal/db.html , 2002.

[Rit94] Ritter, O.. “The Integrated Genomic Database”. *Computational Methods in Genome Research,* ed. S.Suhai, *Plenum*, pp. 57-73, 1994.

[RK83] Rich, E., Knight, K.. “Artificial Inteligence”. *Editora McGraw-Hill Companies*, 1983.

[RM00] Roberts, R. J., Macelis, D.. “REBASE – restriction enzymes and methylases”. *Nucleic Acids Research.* 28, pp. 306-307, 2000.

[RSG95] Rozen, S., Stein, L., and Goodman, N., (1995). Labbase: A database to manage laboratory data in a large-scale genome-mapping project. *IEEE Computers in Medicine and Biology*, 14 702-709 (Nov./Dec. 1995).

[RZPD02] “Resource Center/Primary Database (RZPD)”.

Disponível: http://www.rzpd.de/, 2002.

S

[SAC02] "Sequences Alignments and Comparisons". Disponível: http://bioweb.pasteur.fr/seqanal/alignment/intro-uk.html, 2002.

[Sanger02] "The Wellcome Trust Sanger Institute". Disponível: http://www.sanger.ac.uk/, 2002.

[SBB00] Stevens, R., Baker, P., Bechhofer, S., Ng, G., Jacoby, A., Paton, N.W., Goble, C.A., Brass, A.. "TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources". *Bioinformatics,* 16(2), pp.184-186, 2000.

[SBB02] Stoesser, G., Baker, W., Broek, A.v.d., Camon, E., Garcia-Pastor, M., Kanz, C., Kulikova, T., Leinonen, R., Lin, Q., Lombard, V., Lopez, R., Redaschi, N., Stoehr, P., Tuli, M.A., Tzouvara, K., Vaughan, R.. "The EMBL Nucleotide Sequence Database". *Nucleic Acids Research*, 30: 21-26, 2002.

[SBS02] "School of Biological Sciences – UK". Disponível: http://www.biomed.man.ac.uk, 2002.

[Sch97] Schmidt, D.G., "Applying Design Patterns and Frameworks to Develop Object-Oriented Communication Software". *Handbook of Programming Languages*, V. 1, Mac Millan Computer Publishing, 1997.

[SCH+97] Singh M., Cannata P., Huhns M., et all "The Carnot Heterogeneous Database Project: Implemented Applications", Distributed and parallel Databases 5, 207-225 (1997).

[SGB+01] Stevens, R., Goble, C., Baker, P., Brass, A.. “A Classification of Tasks in Bioinformatics”. *Bioinformatics*, 17(2), pp. 180-188, 2001.

[SIMS02] “SIMS”.
Disponível: http://www.isi.edu/info-agents/index.html, 2002.

[SK92] Sheth, A., Kashyap, V.. “So Far (Schematically) yet so Near (Semantically)”. *IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems*, Elsevier Scientific Publisher B. V., 1992.

[SK98] Schulze-Kremer, S. “Ontologies for Molecular Biology. *Proc. of the Third Pcific Symposium on Biocomputing*, AAAI Press, pp. 693-704, 1998.

[SKS97] Silberschatz, A., Korth, H.F., Sudarshan, S.. “Database System Concepts”. *Editora McGraw-Hill Companies*, 1997.

[SL01] Seibel, L.F.B., Lifschitz, S.. “A Genome Database Framework”. *Proc. 12th Database and Expert Systems Applications* (DEXA), ed. T. Bench-Capon et al., Springer-Verlag, pp. 319-329, 2001.

[SL90] Sheth, A.P., Larson, J.A.. “Federated Database Systems for Managing Distributed, Heterogeneous, and Autonompus Databases”. *ACM Computing Surveys*, vol. 22, n. 3, pp. 183-236, 1990.

[SLL00] Seibel, L.F.B., Lemos, M., Lifschitz, S.. “Bancos de Dados de Genoma”. *Procs. of the Brazilian Databases Symposium Tutorials*, pp. 514-553, 2000.

[SM93] Szeto, E., Markowitz, V.M., “ERDRAW 5.3. A Grafical Editor for Extended Entity-Relationship Schemas”. *TR LBL-PUB-3084*, Lawrence

Berkeley National Laboratory, 1993.

[Species02] "Species 2000".
Disponível: http://www.sp2000.org/sp2000bg.html, 2002.

[SPY02] "XML Spy".
Disponível: http://www.xmlspy.com/, 2002.

[SRS02a] "Sequence Retrieval System".
Disponível: http://www.expasy.org/srs5/, 2002.

[SRS02b] "Sequence Retrieval System".
Disponível: http://srs.ebi.ac.uk/, 2002.

[SS97] Solovyev, V., Salamov, A.. "The Gene-Finder computer tools for analysis of human and model organisms genome sequences". *Proceedings of the International Conference of Intelligent Systems in Molecular Biology*, 5, pp. 294-302, 1997.

[STACK02] "STACK".
Disponível: http://www.sanbi.ac.za/Dbases.html, 2002.

T

[TAIR02] The Arabidopsis Thaliana Information Resources.
Disponível: http://www.arabidopsis.org/, 2002.

[TAMBIS02a] "Transparent Access to Multiple Bioinformatics Information Sources".
Disponível: http://img.cs.man.ac.uk/tambis/, 2002.

[TAMBIS02b] "School of Biological Sciences – UK".
Disponível: http://www.biomed.man.ac.uk/, 2002.

[TAMBIS02c] "The Information Management Group".
Disponível: http://img.cs.man.ac.uk, 2002.

[TAMBIS02d] "Computer Science - University of Manchester".
Disponível: http://www.cs.man.ac.uk/, 2002.

[TDZ01] Tabaska, J.E., Davuluri, R.V., Zhang, M.Q.. "Identifying the 3'-terminal exon in human DNA". *Bioinformatics*, 17(7), pp. 602-7, 2001.

[THG94] Thompson, J.D., Higgins, D.G., Tgibson, T.J.. "CLUSTALW: Improving the Sensitivity of Progressive Multiple Sequence Alignment through Sequence Weighting, Position-Specific Gap Penalties and Weight Matrix". *Nucleic Acids Research*, pp.4673-4680, 1994.

[TIGR02] "The Institute for Genomic Research".
Disponível: http:www.tigr.org/, 2002.

[TLP02] "The Tree of Life Project".
Disponível: http://tolweb.org/tree/phylogeny.html, 2002.

[TRANSFAC02] "TRANSFAC - The Transcription Factor Database, Research Group Bioinformatics/AG Bioinformatik".
Disponível:http://transfac.gbf.de/TRANSFAC/, 2002.

[Tsimmis02] "Tsimmis".
Disponível: http://www-db.stanford.edu/tsimmis/tsimmis.html, 2002.

U

[Uch99] Uchôa, E.. "Framework para integração de sistemas de banco de dados heterogêneos". *Tese de Doutorado*. Departamento de Informática PUC-RIO, 1999.

[ULM97] Uchôa, E., ., Lifschitz, S.Melo, R.. “HEROS: um Sistema de Banco de Dados Heterogêneos usando CORBA”. *Monografia em Ciência da Computação n° 42/97 - ISSN 0103-9741*. Departamento de Informática PUC-RIO, 1997.

[UML96] Uchôa, E., Melo, R., Lifschitz, S..“Interoperabilidade de Objetos em Sistemas de Banco de Dados Hetrogêneos”. *Monografia em Ciência da Computação n° 45/96 - ISSN 0103-9741*. Departamento de Informática PUC-RIO, 1996.

[UniGene02] “UniGene”. Disponível: http://www.hgmp.mrc.ac.uk/Bioinformatics/Databases/unigene-help.html, 2002.

V

[Vec02] “VecScreen - Contamination In Sequence Databases”. Disponível: http://www.ncbi.nlm.nih.gov/VecScreen/contam.html, 2002.

W

[W3C02] “The World Wide Web Consortium (W3C)”. Disponível: http://www.w3.org/, 2002.

[WB98] Wheelan S., Boguski M., “Late-Night Thoughts on the Sequence Annotation Problem”, Genome Research, vol. 8, pp. 168-169, 1998.

[WJ90] Wirfs-Brock, R.J., Johnson, R.E.. “Surveying current research in object-oriented design”. Communications of the ACM, vol. 33, n. 9, pp. 104-124, 1990.

[WCH+00] Wingender, E., Chen, X., Hehl, R., Karas, H., Liebich, I., Matys, V., Meinhardt, T., Prüß, M., Reuter, I., Schacherer, F.. “TRANSFAC: an

integrated system for gene expression regulation". *Nucleic Acids Research*, 28, pp. 316-319, 2000.

[WFJ+02] Westbrook, J., Feng, Z., Jain, S., Bhat, T. N., Thanki, N., Ravichandran, V., Gilliland, G. L., Bluhm, W., Weissig, H., Greer, D. S., Philip, Bourne, E., Berman, H. M.. "The Protein Data Bank: Unifying the Archive". *Nucleic Acids Research*, 30, pp. 245-248, 2002.

[Wie92] Wiederhold, G.. "Mediators in the Architecture of Future Information Systems". *IEEE Computer*, pp. 38-49, 1992.

[Wie93] Wiederhold, G.. "Intelligent Integration of Information". *Proc. of the ACM Conference on Management of Data*, pp. 434-437, 1993.

[WHA+02] Wu, C.H., Huang, H., Arminski, L., Castro-Alvear, J., Chen, Y., Hu, Z., Ledley, R.S., Lewis, K.C., Mewes, H., Orcutt, B.C., Suzek, B.E., Tsugita, A., Vinayaka, C. R., Yeh, L.L., Zhang, J., Barker, W.C.. "The Protein Information Resource: an integrated public resource of functional annotation of proteins". *Nucleic Acids Research*, 30, pp. 35-37, 2002.

[WHIPS02] "WHIPS".
Disponível: http://www-db.stanford.edu/warehousing/warehouse.html, 2002.

X

[XML02] "XML".
Disponível: http://www.w3.org/XML/, 2002.

[XMLSchema02] "XML Schema".
Disponível: http://www.w3.org/XML/Schema, 2002.

[XOL02] "XOL Ontology Exchange Language".
Disponível: http://www.ai.sri.com/pkarp/xol/, 2002.

[Xpath02] "XML Path Language (XPath)".
Disponível: http://www.w3.org/TR/xpath, 2002.

[Xquery02] "Xquery".
Disponível: http://www.w3.org/TandS/QL/QL98/pp/xquery.html, 2002.

Y

[YACC02] "The LEX & YACC Page".
Disponível: http://dinosaur.compilertools.net/, 2002.

[Yod97] Yoder, J.W.. "Patterns for Developing Successful Object Oriented Frameworks". *OOPSLA Conference on Developing Successful Object oriented frameworks*, pp.72-76, 1977.

Z