5 Um *Framework* para Sistemas de Gerência de Análises em Biossequências

5.1 Introdução

Um sistema de gerência de análises (SGABio) em biossequências é composto por dois sub-sistemas. O primeiro é um sistema de gerência de workflows de Bioinformática (SGWBio), que oferece automatização dos processos necessários para se realizar as análises. O segundo é um sistema de gerência de dados em Bioinformática (SGDBio), que trata do armazenamento e da manipulação dos dados envolvidos nestas análises. Esta tese dará ênfase ao componente SGWBio.

Um SGABio pode ser construído em diversos tipos de ambientes de trabalho dos pesquisadores, desde um ambiente pessoal que possui apenas uma máquina até ambientes mais complexos que possuem um parque de máquinas, como os ambientes de laboratório e de comunidade, que serão apresentados em detalhes no Capítulo 6.

Este capítulo apresenta uma proposta de um *framework* para um SGABio. Um *framework* é "uma arquitetura desenvolvida com o objetivo de se obter a máxima reutilização, representada como um conjunto de classes abstratas e concretas, com grande potencial de especialização" [Mattsson, 1996].

5.2 *Framework* para Sistemas de Gerência de Análises em Biossequências

O *framework* proposto está dividido em módulos (também chamados de pacotes na literatura), definidos de acordo com as responsabilidades que tal sistema deve atender. Ao longo deste capítulo serão apresentados os conceitos destes módulos pertinentes a qualquer ambiente de trabalho em que o *framework* for instanciado. No próximo capítulo serão apresentados os detalhes do sistema em cada um dos ambientes de trabalho mais comuns dos pesquisadores.

A Figura 8 e a Figura 9 apresentam um diagrama esquemático e um diagrama de classes, respectivamente, representando os principais módulos do sistema.

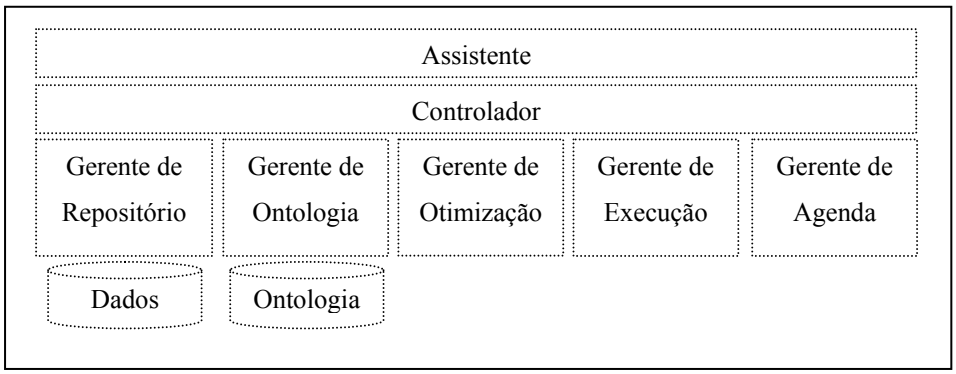

Figura 8. Diagrama esquemático do SGABio.

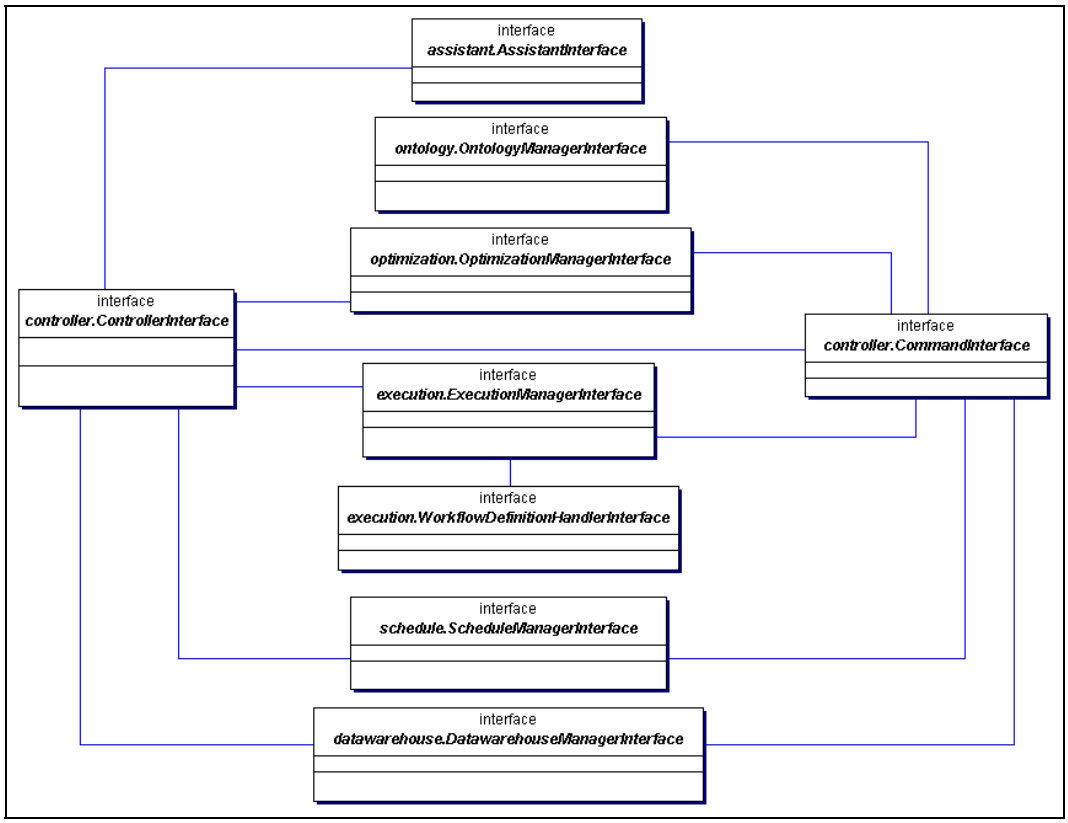

Figura 9. Diagrama de Classes do SGABio.

Segue uma descrição sucinta de cada um destes módulos do SGABio.

Assistente

O módulo assistente é responsável pela interface gráfica que permite ao pesquisador interagir com o SGABio. Ele está sendo representado no diagrama de classes da Figura 9 pela interface *AssistantInterface*.

Este módulo é comum aos dois sub-sistemas do SGABio. No SGWBio, ele fornece ferramentas que permitem ao pesquisador definir, validar, otimizar, agendar e executar um workflow. No SGDBio, ele permite, por exemplo, que o pesquisador defina relatórios sobre os resultados do workflow. Nos dois sub-sistemas, ele utiliza informações contidas na ontologia.

Gerente de Ontologia

O módulo gerente de ontologia é responsável por acessar e manipular todas as informações da ontologia. Ele está sendo representado no diagrama de classes da Figura 9 pela interface *OntologyManagerInterface*.

Este módulo está presente nos dois sub-sistemas do SGABio. No SGWBio, ele fornece, por exemplo, informações que permitem que o assistente ajude o pesquisador a definir um workflow mais adequado para a tarefa que deseja realizar. No SGDBio, ele fornece, por exemplo, informações que o assistente utiliza para ajudar o pesquisador a definir relatórios sobre os resultados do workflow.

Gerente de Otimização

O módulo gerente de otimização é responsável pela validação, otimização e criação do documento de especificação do workflow de acordo com a definição feita pelo pesquisador. Ele utiliza informações contidas na ontologia do SGABio.

Este módulo faz parte do SGWBio, mas é importante para os dois sub-sistemas do SGABio. Por exemplo, no SGWBio, ele aplica as regras de otimização e inclui, no workflow definido pelo pesquisador, processos responsáveis pelo armazenamento dos resultados no *data warehouse* que será acessado pelo SGDBio.

Este módulo está sendo representado no diagrama de classes da Figura 9 pela interface *OptimizationManagerInterface*.

Gerente de Execução

O módulo gerente de execução é responsável pela execução do workflow definido pelo pesquisador, utilizando informações definidas no documento de especificação do workflow criado pelo gerente de otimização. Ele está sendo representado no diagrama de classes da Figura 9 pela interface *ExecutionManagerInterface* que se relaciona com a interface *WorkflowDefinitionHandlerInterface,* representante do documento de especificação do workflow.

Este módulo faz parte do SGWBio, mas é importante para os dois sub-sistemas do SGABio. No SGWBio, ele gerencia a execução e executa processos responsáveis pelo armazenamento dos resultados no *data warehouse* que será acessado pelo SGDBio. Desta forma, há uma comunicação entre os dois sub-sistemas através deste módulo.

Gerente de Agenda

O módulo gerente de agenda, presente apenas no sub-sistema SGWBio, é responsável pelo agendamento das execuções dos workflows. Ele está sendo representado no diagrama de classes da Figura 9 pela interface *ScheduleManagerInterface*.

Controlador

O módulo controlador, comum aos dois sub-sistemas do SGABio, é responsável pela comunicação entre os diversos módulos do SGABio. Ele está sendo representado no diagrama de classes da Figura 9 pela interface *ControllerInterface*. O controlador cria comandos para atender os pedidos de todos os módulos do sistema. Os comandos são representados pela interface *CommandInterface.*

Gerente de Repositório

O módulo gerente de repositório é responsável pelo acesso e manipulação de todos os dados armazenados no *data warehouse* do SGABio. Ele está sendo representado no diagrama de classes da Figura 9 pela interface *DataWarehouseManagerInterface*.

Este módulo é comum aos dois sub-sistemas do SGABio, acessando informações contidas no *data warehouse* em ambos os casos. No SGWBio, ele

armazena, por exemplo, os resultados dos processos dos workflows que foram executados e as biossequências que são analisadas. No SGDBio, ele, por exemplo, acessa o *data warehouse* e fornece todos os dados dos relatórios que o pesquisador definiu.

Todos os módulos do SGABio são apresentados de forma detalhada nas próximas seções deste capítulo, dando ênfase às suas responsabilidades no SGWBio. O SGDBio será considerado na apresentação do gerente de repositório, módulo principal deste sub-sistema. Durante esta apresentação, serão descritas as tarefas dos outros módulos, como gerente de ontologia e assistente, que pertencem também ao SGDBio.

Cada um destes módulos possui *hot spots* e *frozen spots*. Os *hot spots* são partes do *framework* que são abertas para customização e extensão, chamados também de *pontos de flexibilização*. Os *frozen spots* compõem as outras partes do *framework*.

Em conjunto com estes módulos, existe ainda o módulo *compartilhamento de objetos* (também chamado de *blackboard* na literatura), que não foi apresentado no diagrama da Figura 9 para simplificá-lo. Este módulo representa uma área em que os objetos podem ser acessados por todos os módulos do sistema. Entre os vários casos, destacam-se as classes:

- *SProcess*: representante da classe de processos;
- *Input*: representante da classe de dados de entrada dos processos;
- *Output*: representante da classe de dados de saída dos processos;
- *Resource*: representante da classe de recursos utilizados pelos processos;
- *Project*: representante da classe de projetos;
- *WProcess*: representante da classe de processos definidos pelo pesquisador para compor o seu workflow.

Todas estas classes possuem um auto-relacionamento que caracteriza a hierarquia que existe entre os objetos da classe. Por exemplo, as classes de processos, representada por *SProcess*, possui uma hierarquia esquematizada no Quadro 2.

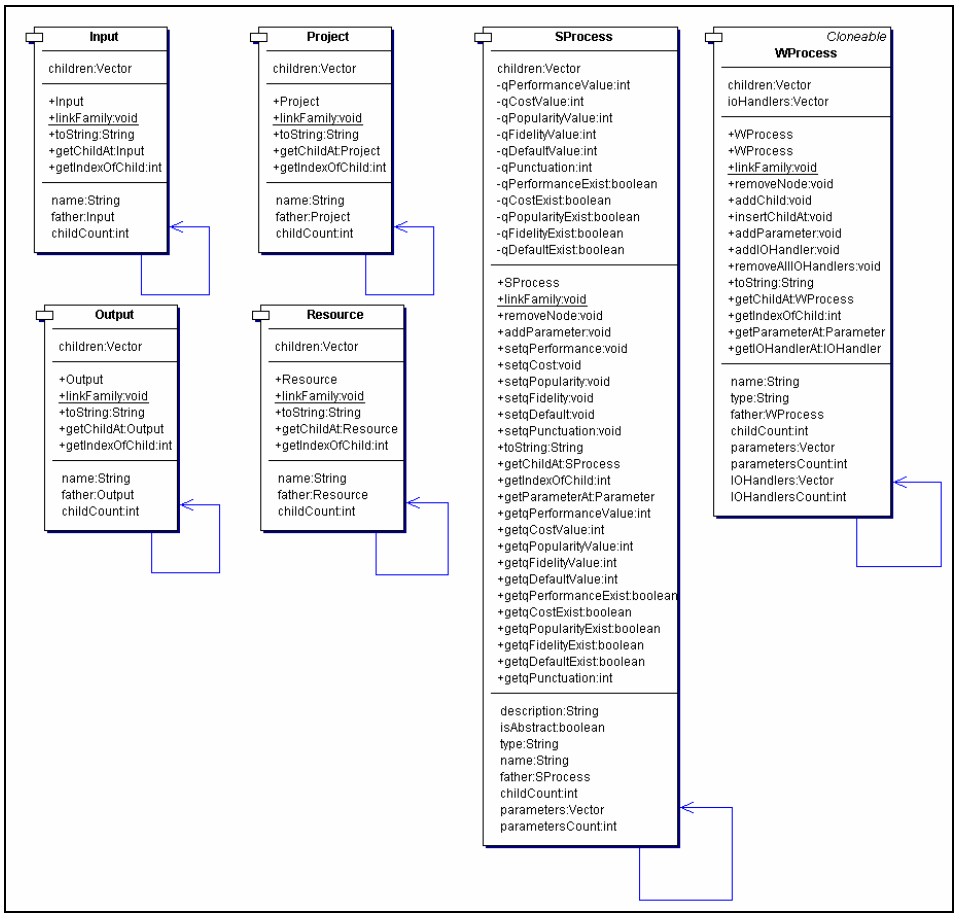

Figura 10. Diagrama de classes: módulo compartilhamento de objetos.

5.3 Controlador

A interface *ControllerInterface* (Figura 11) é a representante do módulo controlador, que é responsável por gerenciar a comunicação entre todos os módulos existentes do SGABio, nos dois componentes: SGWBio e SGDBio.

Quando um módulo necessita realizar uma tarefa que depende de um outro módulo do sistema, ele prepara um pedido e entrega-o ao controlador. O controlador examina o pedido, cria um comando responsável pelo atendimento do pedido. O comando acessa todos os módulos do sistema necessários para atender ao pedido. Cada módulo executa as tarefas pedidas pelo comando e devolve suas respostas. O comando recebe as respostas e entrega-as ao controlador, que repassa-as ao módulo que preparou o pedido.

A interface *ControllerInterface* é um *hot spot* do *framework*, o que permite que o SGABio possa se adaptar a diversos tipos de ambientes de trabalhos. Na

Figura 11, a classe *PersonalController* representa um ambiente de trabalho em uma máquina pessoal do pesquisador, que será detalhado no Capítulo 6.

A interface *CommandInterface* é um *hot spot* do *framework*, o que permite que o SGABio possa estender suas capacidades criando novos comandos sempre que necessário.

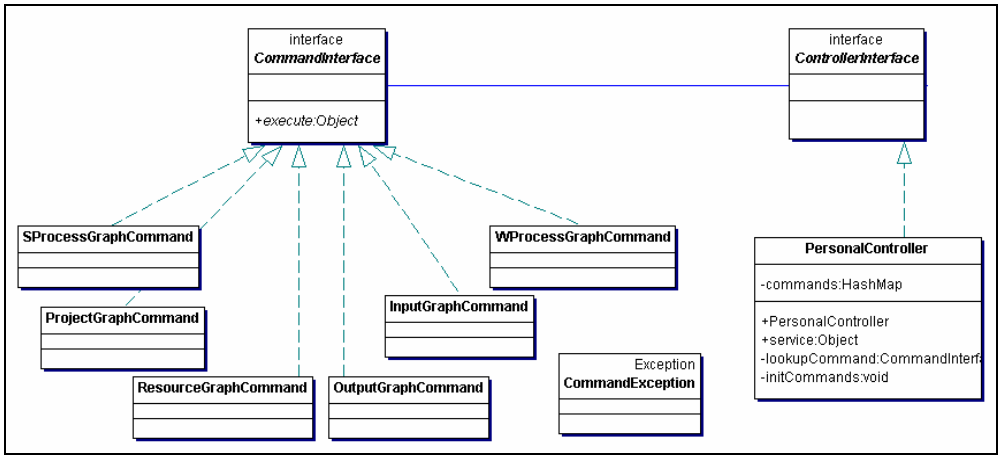

Figura 11. Diagrama de classes: módulo controlador.

5.4 Assistente

O assistente é a interface do SGABio com o pesquisador. Esta seção apresentará as suas responsabilidades no componente SGWBio, que trata da ajuda ao pesquisador para a definição, validação, otimização e execução do workflow. Para tanto, este módulo se comunica com todos os outros módulos do sistema através do módulo controlador (Figura 9). As responsabilidades do assistente no componente SGDBio serão discutidas no final deste capítulo.

A interface *AssistantInterface* é um *hot spot* do *framework* pois permite que o sistema interaja com o pesquisador de diversas maneiras, por exemplo através de aplicações web ou *desktop*, ilustrada pela classe *DesktopAssistant*.

Pelos atributos e métodos da classe *DesktopAssistant* e das classes auxiliares, como *ParameterDialog* e *QualityDialog*, apresentadas na Figura 12, nota-se que este módulo trata de elementos de interface, como botões, painéis e *pop-ups*, com o objetivo de criar uma interface amigável do SGWBio com o pesquisador.

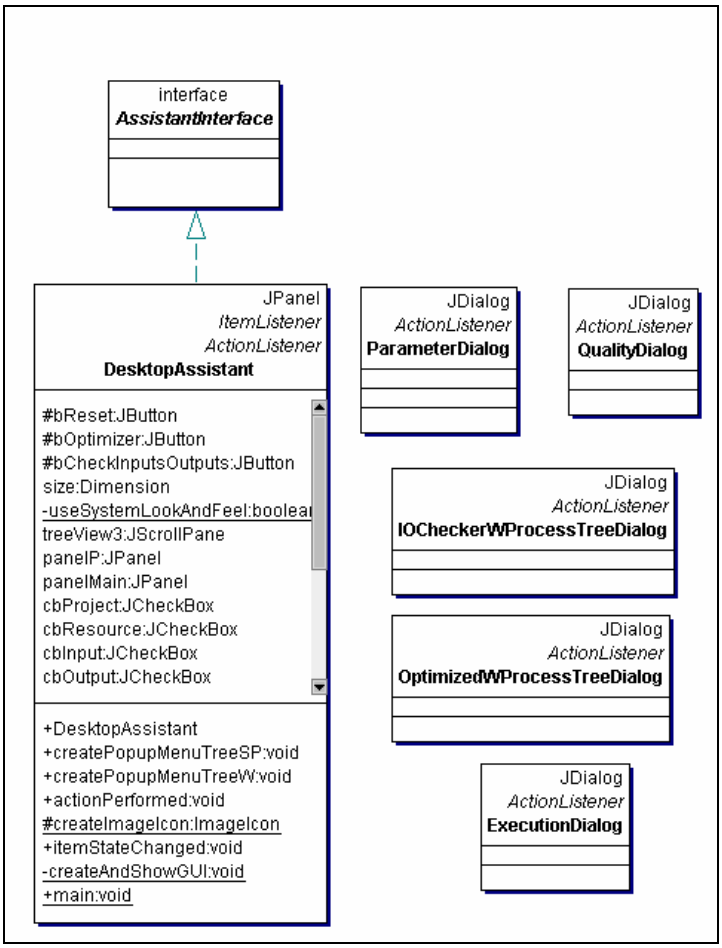

Figura 12. Diagrama de classes: módulo assistente.

O módulo assistente também permite que os usuários (pesquisadores ou profissionais de informática) possam interagir com a ontologia, sendo capazes de cadastrar novos conhecimentos ou atualizar (controlando versões da ontologia) os que já existem. Atualmente já existem ambientes, como o Protégé [Stanford Medical Informatics, 2004], que podem auxiliar nesta tarefa.

Grande parte da responsabilidade do assistente diz respeito à definição do workflow. Segue uma explicação detalhada do auxílio que o sistema fornece ao pesquisador nesta etapa, que é independente do ambiente de trabalho em que o sistema está sendo construído. Contudo, antes é importante ressaltar que esta etapa está totalmente baseada em informações extraídas da ontologia. Sendo assim, há diversas comunicações realizadas entre os módulos assistente, controlador e gerente de ontologia.

Por exemplo, a interface do sistema apresenta o workflow corrente, que está sendo definido pelo pesquisador, e a hierarquia de classes e instâncias de processos, projetos, recursos e dados de entrada e saída, que são extraídas da ontologia.

Suponha o caso da hierarquia de classes e instâncias de processos. O assistente cria um pedido para obter todas as classes e instâncias de processos e envia-o ao controlador (Figura 9). O controlador examina o pedido, descobre que o comando *SprocessGraphCommand* (Figura 11) é responsável por ele, e cria então uma instância da classe *SprocessGraphCommand*. O comando faz acesso ao módulo gerente de ontologia, que captura as informações requisitadas e entrega-as ao comando, que repassa-as para o controlador, que por sua vez, entrega ao assistente.

Nenhum módulo do sistema necessita da ajuda do assistente em suas tarefas. Desta forma, nenhum módulo cria pedido que o assistente deva resolver e, consequentemente, o módulo controlador não cria comandos para serem submetidos ao assistente. Isto pode ser visto na Figura 9, pois não existe relacionamentos entre *CommandInterface* e *AssistantInterface.*

Definição do Workflow

Na criação de um workflow, o pesquisador escolhe as instâncias de processo das classes construtiva, filtro ou controle externo. Em caso de dúvidas, o assistente possui várias formas de auxílio.

Uma ajuda é a restrição das possibilidades de escolha de instâncias de processos através da seleção de classe(s) ou instância(s) da classe de projeto, recurso, entrada ou saída. O assistente apresenta para o pesquisador somente as classes e as instâncias dos processos que estão de acordo com suas restrições.

Outra forma de ajuda é através das propriedades de qualidade dos processos. O pesquisador pede para o assistente apresentar as propriedades de qualidade definidas para uma classe de processo. Estas qualidades diferenciam as instâncias de uma classe de processo. Ao analisar as qualidades, o pesquisador define pesos para cada uma delas. O assistente calcula pontuações para cada instância de processo de acordo com seus pesos e apresenta uma classificação ordenada pelas pontuações. Assim, o assistente ajuda o pesquisador a definir qual é a instância de processo mais adequada para ser incluída no workflow corrente.

O assistente também pode apresentar para o pesquisador a descrição das classes e das instâncias dos processos, projetos, recursos, entrada e saída.

Ao pedir para incluir uma instância de processo no workflow corrente, o pesquisador precisa definir as instâncias de entradas, recursos e saídas associadas ao processo.

Para cada instância de entrada ou recurso, o pesquisador indica se é um arquivo externo ou se é saída de um outro processo. No primeiro caso, ele indica o nome e caminho do arquivo. No segundo caso, o assistente apresentará uma lista com todos os nomes das instâncias de classe de saída de processos definidos anteriormente no workflow, e o pesquisador seleciona qual deve ser utilizada.

Para cada instância de saída, o assistente apresentará um nome padrão para identificação da saída e o pesquisador pode aceitá-la ou modificá-la para um nome mais intuitivo. Estes nomes serão utilizados nas definições das instâncias de entradas ou recursos de outros processos.

Depois da inclusão de uma instância de processo no workflow corrente, o pesquisador pode configurar os seus parâmetros para ajustar a qualidade dos resultados gerados pelo processo. Para tanto, o assistente apresentará os parâmetros de uma instância da classe processo e suas propriedades: nome e valor. A princípio, o valor estará configurado como o padrão, mas o pesquisador poderá alterá-lo e, se desistir, retornar para o padrão.

O assistente é capaz de validar o preenchimento dos valores definidos pelo pesquisador de acordo com os tipos de dados esperados pelos processos. Por exemplo, caso um parâmetro seja do tipo inteiro, o pesquisador não poderá entrar com uma cadeia de caracteres.

Além disso, o assistente permite que o pesquisador redefina as entradas, saídas e recursos de uma instância de processo presente no workflow corrente.

O assistente também apresenta os filtros das instâncias da classe processos de Bioinformática. Foi notado que algumas instâncias de processos de Bioinformática já possuem parâmetros de entrada que restringem sua saída e, consequentemente, não necessitam explicitamente da definição de uma instância de processo filtro.

O pesquisador pode incluir uma instância de processo filtro no workflow, associando-a a uma instância de processo de Bioinformática. O assistente não

permitirá tal inclusão caso não exista a definição de relacionamento entre as duas instâncias na ontologia.

O assistente também apresenta a instância da classe controle "!", que permite que o pesquisador interrompa a execução do workflow para uma análise dos resultados intermediários e retorne a execução caso queira. Esta instância deve estar associada à última instância de processo que o pesquisador deseje que execute antes da interrupção.

Além disso, o assistente também deve fornecer ferramentas para que o pesquisador possa incluir as outras instâncias da classe controle externo, como a de condição (*if*) e a de parada (*exit*).

Redefinição do Workflow

No caso do pesquisador não ficar satisfeito com os resultados (intermediários ou finais) do workflow, o assistente pode ajudá-lo a redefinir seu workflow. Na análise dos resultados intermediários, o pesquisador pode decidir redefinir o restante do workflow ainda não executado, além de somente dizer se quer ou não continuar a sua execução.

Considerando que existem vários bancos de dados que armazenam dados similares, diversos programas que executam tarefas similares e que possuem parâmetros de entrada que limitam a qualidade de seus resultados, o assistente pode indicar instâncias similares da classe contêiner ou da classe processo e modificações em parâmetros a fim de restringir ou relaxar a qualidade dos resultados.

Como exemplo de bancos de dados similares, tem-se o Swiss-Prot e o TrEMBL, que armazenam sequências de proteínas. As sequências de proteínas do Swiss-Prot são adicionadas depois de uma cuidadosa avaliação de pesquisadores e, portanto, possui anotações com ótima qualidade, mínima redundância e uma alta integração com outros bancos de dados. Em contraste, as sequências de proteínas do TrEMBL são traduções, feitas de forma automática, das sequências presentes no banco de dados de sequências de nucleotídeos EMBL e que ainda não foram integradas ao Swiss-Prot.

O TEIRESIAS e o Pratt são exemplos de programas semanticamente similares, ou seja, ambos tem como objetivo encontrar padrões desconhecidos em sequências.

Validação e Otimização *a priori*

Depois da definição de um workflow, o pesquisador pode pedir ao assistente para validá-lo. Para a validação, o assistente cria um pedido e o envia, em conjunto com o workflow definido pelo pesquisador, ao controlador. O controlador descobre o comando responsável e repassa o pedido. O comando acessa o módulo gerente de otimização, responsável pela validação. Se o resultado da validação for positivo, o gerente de otimização cria um documento de especificação do workflow, incluindo aspectos que auxiliarão o gerente de execução a executar o workflow de forma otimizada. Este documento é entregue ao controlador, que o repassa para o assistente. O assistente apresenta-o ao pesquisador, que pode gravar este arquivo em disco e pedir para executá-lo através do menu principal do assistente.

Execução, Monitoramento e Otimização *a posteriori*

A execução é responsabilidade do gerente de execução. Neste caso, o assistente cria o pedido e o envia, em conjunto com o documento de especificação do workflow, para o gerente de execução. A execução é realizada e a resposta é devolvida ao controlador que repassa-a ao assistente.

O assistente deve permitir que o pesquisador monitore a execução do workflow. Neste monitoramento o pesquisador saberá, por exemplo, quais processos estão sendo executados, quais máquinas (quando for o caso) estão sendo utilizadas, os tempos de execução e os resultados obtidos pelos processos que já terminaram a execução. Estas informações devem ser passadas pelo gerente de execução para o assistente. A questão de monitoramento não será tratada nesta tese.

O gerente de execução utiliza informações no documento de especificação do workflow em conjunto com as informações do ambiente em que está sendo executado para fazer otimizações durante a execução.

Agendamento

O pesquisador conta com a ajuda do assistente para agendar a execução do workflow. O agendamento é de responsabilidade do módulo gerente de agenda. O gerente de agenda deve chamar o gerente de execução no momento em que foi

especificada a execução do workflow. A questão de agendamento não será tratada nesta tese.

As etapas de validação, otimização e execução são detalhadas nas próximas seções.

5.5 Gerente de Ontologia

O gerente de ontologia é responsável pelo acesso e manipulação da ontologia. Entende-se como manipulação, consulta, adição, remoção e atualização de conhecimentos na ontologia. A atualização é importante para acomodar novas classes, propriedades e regras pertinentes aos processos, recursos, dados e projetos de Bioinformática.

Este gerente possui um agente de software que inspeciona os acessos à ontologia e as escolhas das instâncias de processos de Bioinformática e atualiza as propriedades de qualidade.

A Figura 13 mostra a interface *OntologyManagerInterface* que representa o módulo gerente de ontologia.

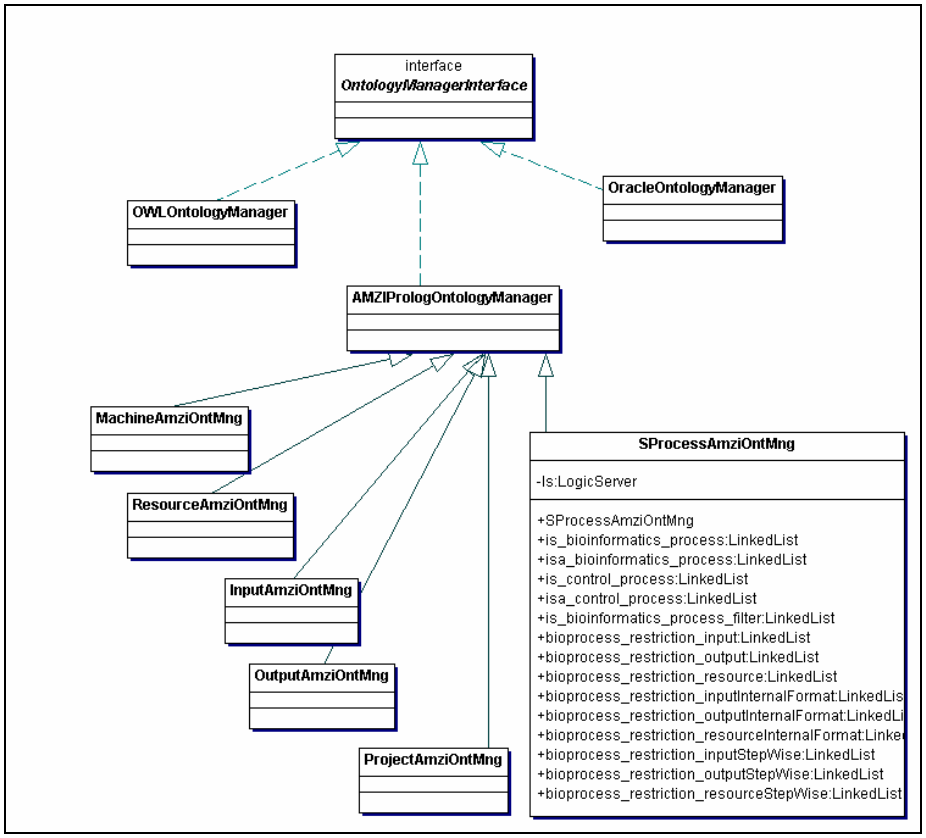

Figura 13. Diagrama de classes: módulo gerente de ontologia.

Esta interface é um *hot spot* do *framework* porque permite que a ontologia seja definida e gerenciada de diversas maneiras.

No diagrama da Figura 13 estão representados os casos em que a ontologia foi definida em um sistema gerenciador de banco de dados relacional, no caso o Oracle [Oracle, 2004], através da classe *OracleOntologyManager*, em OWL-S [DAML, 2004], através da classe *OWLSOntologyManager*, e em Prolog [Crookes, 1988], no caso utilizando a versão do Amzi-Prolog [Amzi!, 2004], através da classe *AMZIPrologOntologyManager*.

O módulo gerente de ontologia não necessita da ajuda de outros módulos para realizar suas tarefas. Desta forma, ele não cria pedidos e, consequentemente, não os envia ao módulo controlador. Isto pode ser visto na Figura 9, pois não existem relacionamentos entre *OntologyManagerInterface* e *ControllerInterface.*

5.6 Gerente de Otimização

O gerente de otimização é responsável pela validação do workflow, pela inclusão de contêineres, de conexões, de processos de controle interno (como de transformação de formatos e de inspeção), pela definição destes objetos de forma a identificar otimizações que podem ser feitas no workflow. Todas estas informações são refletidas no documento de especificação do workflow, que o gerente cria.

A Figura 14 mostra a interface *OptimizationManagerInterface* que representa o gerente de otimização. A *OptimizationManagerInterface* é um *hot spot* do *framework* pois permite que o gerente de otimização seja definido de formas particulares para ambientes de trabalho diferentes. No diagrama há três classes, cada uma para um tipo de ambiente diferente: a classe *PersonalOptimizationManager* para um gerente de otimização em um ambiente pessoal, *LaboratoryOptimizationManager* para um gerente em um ambiente de laboratório, *CommunityOptimizationManager* para um gerente em um ambiente de comunidade.

Esta seção apresentará o gerente de otimização de forma geral, sem os aspectos específicos de cada ambiente. Estes aspectos serão descritos no próximo capítulo.

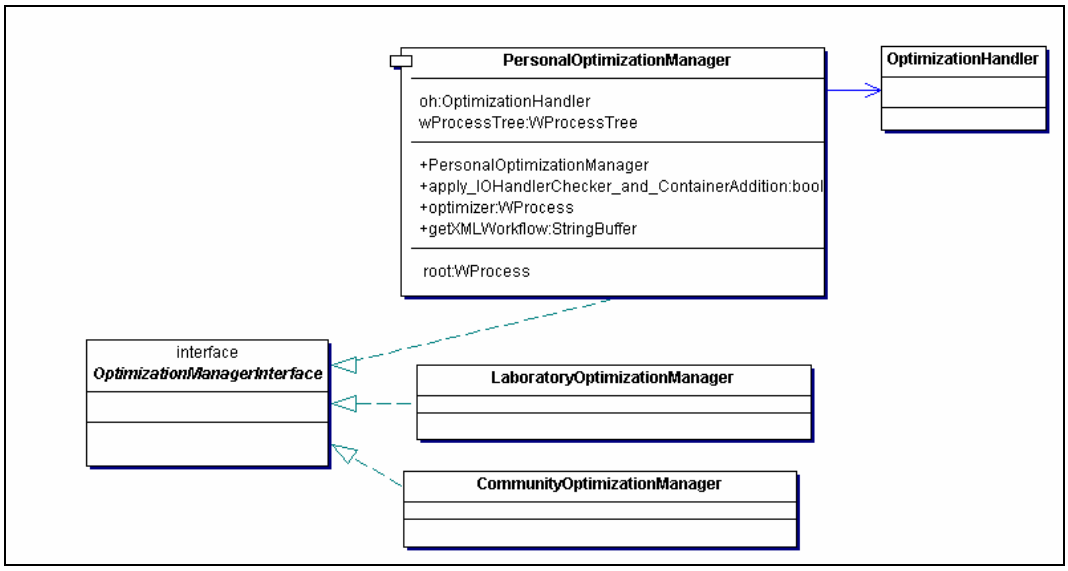

Figura 14. Diagrama de classes: módulo gerente de otimização.

5.6.1 Validação

Na etapa de validação, o otimizador verifica a coerência das instâncias de dados de entradas, saídas e recursos definidos para as instâncias dos processos.

Caso os formatos dos dados compartilhados entre as instâncias de processos sejam diferentes, o gerente de otimização tornará, se possível, a composição viável. Isto é feito através da inclusão de instâncias de processos de transformação de formato ou, através da configuração de parâmetros das instâncias de processos que definem o formato de saída dos seus resultados. Após este tratamento, o gerente de otimização inclui instâncias de processos de inspeção.

A validação é feita pelo gerente de otimização em conjunto com o gerente de ontologia. Por exemplo, conhecendo os formatos internos de entradas, recursos e saídas (informações armazenadas na ontologia), o assistente pode determinar as instâncias de processos de transformação de formato, caso existam.

Neste caso, o gerente de otimização cria pedidos e entrega-os ao controlador. O controlador instancia comandos que acessam o gerente de ontologia, que consulta as informações na ontologia e devolve as respostas aos comandos. Estes repassam as respostas ao controlador, que por sua vez, retornam-as ao gerente de otimização (Figura 9).

Além da coerência entre entradas e saídas dos processos, a validação pode ajudar o pesquisador na definição do seu workflow. Considerando um cenário

com muitos bancos de dados e programas semanticamente equivalentes, durante a validação, o sistema pode avisar ao pesquisador que existe chance de ele estar cometendo engano na definição do seu workflow, caso ele peça a inclusão de várias instâncias de processos e de recursos similares.

5.6.2 Otimização

A experiência de trabalhar com pesquisadores de Bioinformática nos mostrou que, no contexto de sistemas de processamento de biossequências, os workflows não são muito complicados, e consequentemente, não há necessidade de procedimentos de otimização complexos, tais como os encontrados em sistema de bancos de dados objeto-relacional. Contudo, há vantagens no emprego de vários procedimentos simples de otimização.

Desta forma, quando o resultado da validação for positivo, o gerente de otimização cria um documento de especificação do workflow, incluindo aspectos que auxiliarão o gerente de execução a executar o workflow de forma otimizada. Este documento é entregue ao controlador, que o repassa para o assistente.

Os aspectos referentes à otimização estão de acordo com a ontologia definida anteriormente. Sendo assim, da mesma forma que na etapa de validação, há uma comunicação entre o gerente de otimização, o controlador e o gerente de ontologia. Os detalhes referente à otimização serão tratados no próximo capítulo.

5.6.3 Documento de Especificação do Workflow

Na etapa final, o gerente de otimização cria um documento de especificação do workflow que permite que o gerente de execução possa construir um grafo bipartido para representar o workflow de Bioinformática, e gerenciar sua execução de acordo com a otimização pertinente a cada ambiente.

O grafo bipartido possui vértices chamados de *nós-processo*, representantes de processos, vértices chamados *nós-contêiner*, representantes dos contêineres, e arcos que conectam nós-processo e nós-contêiner. A Figura 4 apresentou um exemplo onde o vértice rotulado por p_p representa um nó-processo produtor, o vértice rotulado por p_c representa um nó-processo consumidor e c_1 representa o nó-contêiner que armazena os dados compartilhados por p_p e p_c.

5.7 Gerente de Execução

O gerente de execução é responsável pela execução do workflow utilizando o documento de especificação do workflow elaborado pelo gerente de otimização.

A Figura 15 apresenta a interface *ExecutionManagerInterface* que representa o gerente de execução. Esta interface é um *hot spot* do *framework* pois permite que o gerente de execução seja definido de formas particulares para ambientes de trabalho diferentes. No diagrama há três classes, cada uma para um tipo de ambiente diferente: a classe *PersonalExecutionManager* para um gerente de execução em um ambiente pessoal, *LaboratoryExecutionManager* para um gerente em um ambiente de laboratório, *CommunityExecutionManager* para um gerente em um ambiente de comunidade.

Como apresentado na Figura 9 e na Figura 15, a interface *WorkflowDefinitionHandlerInterface* é um *hot spot* do *framework* e permite que o gerente de execução interprete o documento de especificação do workflow de acordo com sua definição. Na Figura 15 há a classe *XMLWorkflowDefinitionHandler* como exemplo de uma instanciação de *WorkflowDefinitionHandlerInterface*, que trata de um documento de especificação do workflow escrito XML em um ambiente de trabalho pessoal. Este documento segue o esquema definido no XML Schema do Quadro 7.

Em alguns ambientes de trabalho, o gerente de execução necessitará de apoio do gerente de ontologia, o que, pelo *framework* proposto para o SGABio, só será possível através da comunicação com o controlador. Isto justifica o relacionamento da interface *ExecutionManagerInterface* com a classe *Controller* na Figura 9.

Esta seção apresentará o gerente de execução de forma geral, sem os aspectos específicos de cada ambiente de trabalho dos pesquisadores. Estes aspectos serão descritos no próximo capítulo.

O procedimento de criação de uma *thread* para o tratamento de execução de um processo é dependente do ambiente em que o SGWBio está sendo implementado, e portanto, será detalhado no próximo capítulo. Entretanto, em todos os casos, o gerente de execução fornece à *thread* as listas dos parâmetros e dos contêineres de entrada e de saída do processo, de acordo com a definição do documento de especificação do workflow. Esta *thread* é responsável por um

procedimento adaptador, que sabe interpretar as informações obtidas do gerente de execução e adaptá-las de forma a executar o processo.

A definição da forma de implementação dos contêineres pode ser feita de forma a otimizar o workflow. Esta otimização depende do ambiente em que o SGWBio estiver sendo instanciado. Consequentemente, os aspectos referentes ao contêiner durante a execução do workflow também dependem do ambiente, e portanto, serão apresentados no capítulo seguinte.

O gerente de execução deve disponibilizar os resultados dos processos em uma área (permanente ou temporária) do *data warehouse* do SGABio de forma a permitir que o assistente apresente-os ao pesquisador. Não é necessário que o workflow seja completamente executado para que os resultados estejam disponíveis para o pesquisador acessar.

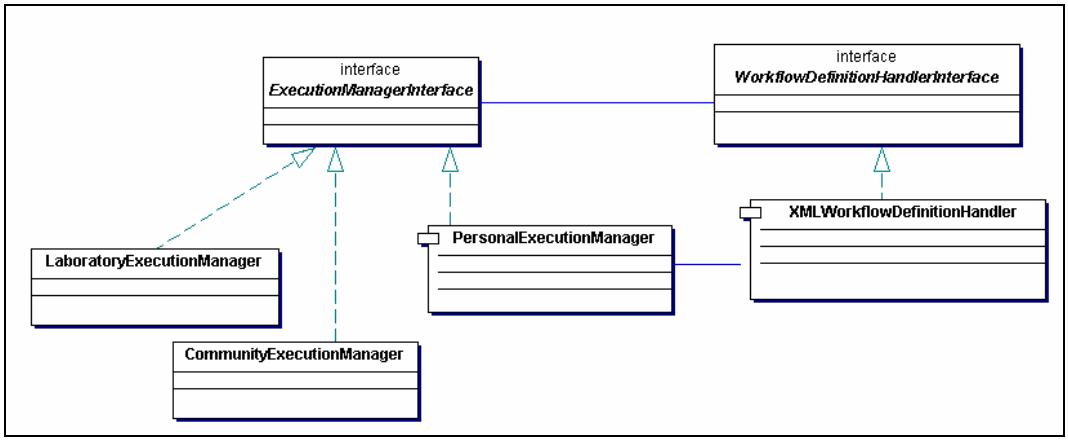

Figura 15. Diagrama de classes: módulo gerente de execução.

5.8 Gerente de Repositório

O módulo gerente de repositório é responsável pelo armazenamento, acesso e manipulação de todos os dados do *data warehouse* do SGABio. Ele representa o principal módulo do componente SGDBio e conta com o auxílio de outros módulos, como gerente de ontologia, assistente e controlador, para realizar suas tarefas.

Modelo do *Data Warehouse*

No *data warehouse* existem informações dos pesquisadores e dos workflows. A Figura 16 apresenta um modelo lógico simplificado que representa algumas entidades e relacionamentos que devem ser contemplados pelo *data*

warehouse. Um diagrama de um *data warehouse* mais completo é o utilizado pelo sistema de anotação BioNotes, definido em [Lemos, 2003d, e].

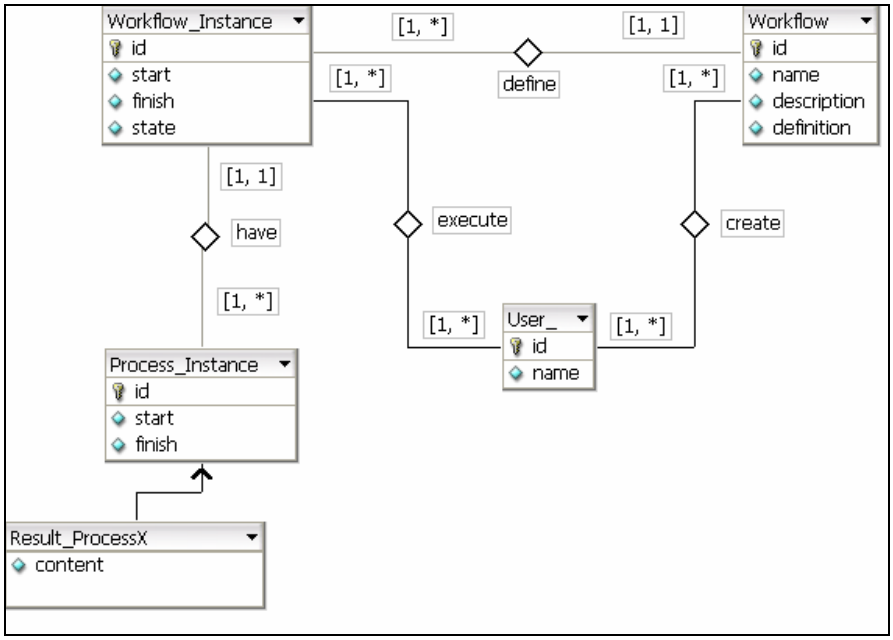

Figura 16. Modelo lógico simplificado do *data warehouse*.

No *data warehouse* estão armazenados os pesquisadores e os vários workflows criados por eles. Um dado importante do workflow é o seu documento de especificação do workflow, que define os processos, os parâmetros dos processos, os contêineres, os recursos e os dados de entrada e saída que são utilizados.

Uma instância do workflow é criada quando um pesquisador pede para executá-lo. Os atributos de uma execução de um workflow incluem: tempo inicial e tempo final da execução; e estado final da execução (terminado com sucesso, terminado com erros, em execução, não iniciado).

A instância do workflow está relacionada aos resultados dos processos que foram executados, definidos na Figura 16 através de entidades de acordo com o nome do processo. Por exemplo, a entidade *Result_ProcessX* representa os resultados gerados pelo processo genérico *X*. Os resultados são criados de acordo com um tipo de dados (em XML Schema), definido na ontologia através da propriedade *formato de dados* da classe contêiner. Eles também possuem como atributos os momentos em que o processo começou e terminou sua execução.

Outros dados, como os dos recursos utilizados pelos processos, também fazem parte do *data warehouse*, e não foram apresentados neste modelo para simplificá-lo.

A Ontologia para o SGDBio

A ontologia inclui classes contêineres correspondendo aos resultados dos processos, que podem ser descritos de diversas maneiras.

Os administradores do SGABio são responsáveis pela definição destas classes, uma tarefa que geralmente não é simples pois depende das documentações dos processos que, nem sempre, são de boa qualidade.

Caso o processo não gere resultados no formato esperado pelo *data warehouse*, um processo de controle interno transformador de formato deve existir para tratar de tal transformação. Além disso, deve existir um outro processo de controle interno para tratar do armazenamento dos resultados no *data warehouse*.

O gerente de otimização é responsável pelo acréscimo automático destes processos no workflow definido pelo pesquisador. Durante esta tarefa, o gerente de otimização conta com a ajuda do gerente de ontologia que o informa sobre os processos de controle interno para transformação de formato e para armazenamento dos resultados no *data warehouse*.

Relatórios

O *data warehouse* permite que o SGDBio preste diversos serviços importantes para os pesquisadores. Um dos mais importantes é o de geração de relatórios das informações armazenadas. Estes relatórios podem ser produzidos através de consultas simples em atributos das entidades como, por exemplo, o relatório dos tempos de execução dos workflows definidos por um determinado pesquisador. Entretanto, várias consultas mais complexas também podem ser feitas para auxiliar os pesquisadores em tarefas como a definição de novos workflows, o uso de resultados gerados em execuções de workflows anteriores como entrada de processos de workflows subsequentes, e a descoberta de informações relacionadas entre diversas execuções de diferentes workflows. Para isso, é importante que o gerente de repositório possa encontrar relacionamentos entre os documentos de especificação dos workflows e, também, entre os resultados das instâncias de workflows.

Por exemplo, neste último caso, suponha o caso do BioNotes que está sendo utilizado pelos pesquisadores do Riogene para analisar a bactéria *Gluconacetobacter diazotrophicus*. O workflow da Figura 1 é executado e os pesquisadores precisam anotar as ORFs geradas pelo Glimmer. No primeiro passo, ele analisa uma ORF, considerando, por exemplo, os resultados do Glimmer e do BLAST, e indica se a ORF gerada pelo Glimmer é ou não válida, ou seja, se a ORF corresponde ou não a um gene. No caso da ORF válida, o pesquisador determina suas características através da seleção de itens em listas com vocabulário controlado e armazenado no *data warehouse* do BioNotes. Entre estas características estão a categoria para classificar o gene, o nome do gene e o nome de um outro organismo em que este gene também é encontrado.

Caso cada pesquisador pudesse definir e executar seus próprios workflows, as anotações das ORFs poderiam ser feitas de forma mais rápida e com mais qualidade, já que os pesquisadores teriam outras maneiras de analisar a ORF, e não somente as oferecidas pelo workflow básico definido na Figura 1.

Uma maneira de auxiliar os pesquisadores neste caso seria apresentando uma lista de todos os metadados dos workflows que tiveram processos em que uma determinada ORF foi entrada ou saída. Desta forma, ele analisaria a ORF utilizando os resultados do workflow definido na Figura 1 e definidos por todos os pesquisadores da comunidade.

Além disso, no caso de se ter um vocabulário controlado para anotações de sequências, é possível relacionar as sequências e os metadados dos workflows em que ela esteve envolvida, como entrada ou saída de algum processo, com outras sequências e outros workflows. No BioNotes seria possível relacionar as ORFs e os metadados dos seus workflows, o que facilitaria ainda mais as análises que precisam ser feitas.

Definição de Relatórios

O módulo assistente pode ajudar o pesquisador a definir e a visualizar os seus relatórios. Na verdade, o assistente ajudará o pesquisador a criar consultas sobre os dados armazenados no *data warehouse*, sem exigir o seu conhecimento em Informática.

Para que o módulo assistente preste este serviço ao pesquisador, ele conta com ajuda do gerente de ontologia, já que a ontologia possui conhecimento sobre as classes de dados de saída dos processos.

Alguns relatórios podem ser pré-definidos pelos administradores do SGABio e disponibilizados aos pesquisadores. Estes relatórios também farão parte da ontologia.

Apresentação dos Relatórios

Além da criação do relatório, o assistente pode dar sugestões de informações presentes nos resultados dos processos, que são importantes de serem consultadas.

Os processos de Bioinformática geralmente produzem como saída parâmetros que indicam a qualidade dos seus resultados. Os valores destes parâmetros ajudam os usuários analisar e conhecer o quanto os resultados são confiáveis.

O assistente pode ajudar os pesquisadores a analisar os resultados do workflow apresentando-os em uma tabela baseada nas qualidades destes resultados. Cada linha da tabela é representante de um dado de entrada, que pode ter sido processado por diversos processos. A coluna é representante de um processo. Desta forma, dado uma linha e coluna, tem-se a qualidade do resultado gerado por um processo para um determinado dado de entrada. Ao analisar todas as células de uma linha, tem-se todas as qualidades geradas por diversos processos para um determinado dado. Se todos os valores forem altos, o pesquisador confiará mais nos resultados. Ao analisar todas as células de uma coluna, tem-se todas as qualidades geradas por um mesmo processo para diferentes dados de entrada. O pesquisador será capaz de conhecer qual entrada de dados tem melhores resultados e são mais confiáveis.

Por exemplo, em um projeto genoma de uma bactéria na fase de anotação como o caso do projeto da *Gluconacetobacter diazotrophicus* descrito anteriormente, os pesquisadores precisam identificar os genes e descobrir suas funções com o auxílio do programa Glimmer, que gera as ORFs, e da comparação das ORFs geradas com vários bancos de dados, por exemplo, utilizando o programa BLAST.

Neste caso é interessante que os resultados da execução do BLAST contra todos os bancos de dados e do Glimmer sejam organizados de uma forma

adequada para que o pesquisador possa comparar as diferentes ORFs e descobrir, mais facilmente, quais delas podem ser consideradas genes e, sendo gene, qual a sua função. No caso de uma tabela, as colunas poderiam ser os resultados dos programas BLAST e Glimmer, e as linhas poderiam ser as ORFs.

Como exemplo de parâmetro de saída do BLAST que pode ser usado como medida de qualidade tem-se o *e-value* (do inglês, *expectation value*). Este parâmetro indica a chance do alinhamento obtido ser relevante ou não. Quanto menor o *e-value*, mais significativo é o alinhamento obtido. Da mesma forma, o Glimmer produz uma medida que indica a probabilidade de que a ORF gerada seja realmente um gene.

Outros programas, como o Transterm, que encontra terminadores no genoma, ou o tRNAScan, que encontra RNAs transportadores, também possuem medidas que indicam a chance do resultado obtido ser real ou um falso positivo. Similarmente, os programas que fazem reconhecimento de bases, como Phred, e os de montagem de fragmentos, como Phrap e CAP3, atribuem medidas de qualidade para cada base da sequência gerada (*read* no primeiro caso, e contig no segundo caso). Estas qualidades também são usadas para se ter segurança do resultado obtido.

Geralmente estes programas aceitam que o pesquisador defina um valor de corte para filtrar o resultado da saída, caso estas medidas estejam abaixo de um determinado valor. Além de medidas numéricas, caso exista um vocabulário controlado nos resultados dos programas, o texto também pode ser considerado.

Para que o módulo assistente possa auxiliar o pesquisador a definir uma melhor forma de apresentação do relatório do workflow, a ontologia deve possuir além dos formatos dos resultados dos processos, informações acerca de quais parâmetros nestes formatos são mais importantes e influenciam mais na análise dos resultados.

Escopo de Visualização

O pesquisador que define um workflow pode liberar o acesso aos metadados e aos dados do workflow para nenhum, para vários ou para todos os pesquisadores da comunidade que pertence. É importante que o SGABio possa tratar estes diferentes escopos de acesso.

Quando o pesquisador criar o workflow, o assistente deve permitir que ele defina o escopo do workflow, e o gerente de repositório deve disponibilizar os dados e metadados do workflow para os outros pesquisadores de acordo com o escopo definido. Esta informação pode ser acrescentada no documento de especificação do workflow.

Ambiente de Colaboração

Ao criar relacionamentos entre os workflows e os pesquisadores, e definir escopos de acesso aos workflows, o gerente de repositório permite que o SGWBio seja capaz de oferecer aos pesquisadores, um ambiente de desenvolvimento colaborativo de workflows. Este ambiente permitirá que os pesquisadores possam conhecer e aprender com os workflows de outros pesquisadores, e consequentemente, criar seus próprios workflows de forma mais eficiente e com maior qualidade.

Para tornar este ambiente ainda melhor, o gerente de repositório pode classificar os workflows de tal forma que os pesquisadores possam avaliá-lo, mesmo sem analisar em detalhes seu documento de especificação e os resultados gerados por suas instâncias.

Esta classificação pode ser feita de diversas maneiras. Um exemplo simples seria através da *popularidade* do workflow, i.e., quanto mais pesquisadores executarem um determinado workflow, maior é a sua popularidade. Um outro exemplo, seria através de notas atribuídas ao workflow pelos pesquisadores que já o executaram.

Além da classificação dos workflows, os pesquisadores também poderiam ser classificados de acordo com os workflows que definem. Os pesquisadores que tivessem definido workflows com popularidade ou notas altas, seriam melhor classificados. Isto ajudaria os pesquisadores a avaliarem um workflow recentemente definido, pois se ele foi definido por um pesquisador com classificação elevada, provavelmente terá uma boa qualidade.

5.9 Comentários Finais

Este capítulo apresentou uma proposta de um *framework* para um sistema de gerência de análises em biossequências, enfatizando o sistema de gerência de workflows de Bioinformática (SGWBio).

Foram apresentados os módulos do *framework*, seus pontos de flexibilização e suas responsabilidades de acordo com os requisitos levantados para este sistema nas seções 2.9 e 2.10.

Como o *framework* pode ser instanciado em diferentes ambientes de trabalho dos pesquisadores, este capítulo mostrou as características gerais do sistema pertinentes a todos os ambientes.

O módulo assistente foi totalmente definido neste capítulo pois ele é responsável pela iteração do sistema com o usuário. Como esta iteração não deve exigir que o pesquisador conheça características de ambiente, este módulo torna-se completamente independente dos tipos de ambientes em que o sistema será implementado.

O Capítulo 6 mostrará instanciações do *framework* para o sistema SGWBio, detalhando as características particulares do sistema para cada ambiente de trabalho em que ele for implementado. O Capítulo 7 apresentará o protótipo de implementação do SGWBio.