Universidade Católica de Santos

Mestrado em Informática

GISE: Uma Arquitetura para a Integração de Múltiplas Fontes de Dados na Grade Computacional

EDUARDO GALLO

Santos

2006

Universidade Católica de Santos

Mestrado em Informática

GISE: Uma Arquitetura para a Integração de Múltiplas Fontes de Dados na Grade Computacional

EDUARDO GALLO

Dissertação apresentada ao Programa de Mestrado em Informática da Universidade Católica de Santos, como requisito parcial para obtenção do grau de Mestre em Informática.
Área de concentração: Ciência da Computação.
Linha de Pesquisa: Sistemas Distribuídos.
Orientador: Prof. Dr. Fabrício Alves Barbosa da Silva.

Santos

2006

Comissão Julgadora

_____

Prof. Fabrício Alves Barbosa da Silva, Ph.D.

_____

Prof. Dr. Manoel Mendes, Ph.D.

_____

Prof[a]. Dr[a]. Júlia Célia Mercedes Strauch, D.Sc.

Aos meus pais:

Salvatore Gallo (*in memorian*) e
Geny Camargo Gallo.

AGRADECIMENTOS

Neste momento tão especial e de realização um grande sonho, quero primeiramente agradecer a Deus por me permitir a vida. Tenho certeza que em todos os momentos que foram dedicados nesta dissertação, Deus protegeu e iluminou meu caminho.

Quero muito agradecer ao Prof. Fabrício Alves Barbosa da Silva que de forma brilhante orientou – me em todos os momentos na realização deste trabalho e através de sua competência, dedicação, técnica e ética viabilizou esta Dissertação. Agradeço muito a este fantástico orientador e excepcional amigo.

Agradeço muito à Prof[a.] Júlia Célia Mercedes Strauch por ter aceitado compor a banca examinadora deste trabalho, cedendo uma parcela de seu precioso tempo e de seu grande conhecimento. Agradeço também a Prof[a.] Júlia por ter me incentivado e disponibilizado referências fundamentais para criação deste trabalho.

Agradeço muito aos Professores Hermes Senger e Manoel Mendes que além de participarem com seus grandes e valiosos conhecimentos da banca examinadora deste trabalho, são extraordinários Professores e sempre estiveram dispostos a esclarecerem minhas questões.

Agradeço também a todos os Professores que participam desta empreitada. Especialmente agradeço muito ao Professor Eduardo Raul Hruschka por sempre ter me disponibilizando grande ajuda em momentos de dúvidas. Suas admiráveis orientações e sua competência profissional foram fundamentais para a Conclusão de diversas disciplinas da Grade Curricular deste Mestrado.

Aos meus colegas de Projeto IntegraEPI: Henrique Gagliardi, Alexandre Resende e Maria Madope. Agradeço pela amizade, apoio, discussões altamente produtivas e bons momentos de companheirismo. Agradeço também o apoio de diversos colegas que tive a oportunidade de conhecer nesta fase da minha vida: Bianca, Alexandre Fukaya, Armando, entre outros.

Quero muito agradecer aos meus grandes primos: Sergio Lamelas e Fernando José Lamelas. Estes que com muito companheirismo e profissionalismo através da Datacomp-Informática foram os propulsores de minha carreira acadêmica e profissional.

Finalmente, quero agradecer a toda minha Família que sempre me apóia e me ajuda em todos os momentos, dificuldades e decisões.

Resumo da Tese apresentada à UNISANTOS como parte dos requisitos necessários para a obtenção do grau de Mestre em Informática (M.Sc.).

GISE: UMA ARQUITETURA PARA INTEGRAÇÃO DE MÚLTLPLAS FONTES DE DADOS NA GRADE COMPUTACIONAL.

Eduardo Gallo

Junho / 2006

Orientador: Prof. Dr. Fabrício Alves Barbosa da Silva.

Programa: Sistemas Distribuídos.

Palavras Chave: GISE, Integração de Fontes de Dados, Grade Computacional.

Este trabalho apresenta uma arquitetura flexível para a integração de múltiplas fontes de dados na Grade Computacional denominada GISE (Grid Integration SErvice). A arquitetura proposta faz uso de Mediadores, Tradutores e Modelos de Dados Canônicos para disponibilizar a execução operações de consultas em fontes de dados heterogêneas e geograficamente distribuídas, resolvendo conflitos sintáticos e semânticos causados pelo processo de integração de múltiplas fontes de dados. O Modelo de Dados Canônico do GISE permite a representação de fontes de dados locais e fontes de dados integradas através de XML. O GISE possui um conjunto de componentes agrupados nas camadas Cliente, Serviço de Metadados, Serviços de Integração (Mediador), Serviços do Globus Toolkit, Serviço de Acesso as Fontes de Dados e as Fontes de Dados. Neste trabalho foi desenvolvido e testado um protótipo da Arquitetura GISE, sendo apresentado um estudo de caso na área de saúde pública.

Abstract of Thesis present to UNISANTOS as a partial fulfillment of the requirements for the degree of Master Computer Sciente (M.Sc.).

GISE: AN ARCHITECTURE FOR INTEGRATION OF MULTIPLE DATASOURCES IN THE GRID.

Eduardo Gallo

June / 2006

Advisor: Prof. Dr. Fabrício Alves Barbosa da Silva.

Departament: Distributed Systems.

Keywords: GISE; Data Integration; Grid Computing.

This work presents flexible data integration architecture on the Grid called GISE (Grid Integration SErvice). The proposed architecture is composed of Mediators, Wrappers and Canonic Data Models to run query operations on heterogeneous and geographic distributed data sources and to resolve syntactic and semantic conflicts caused by the integration process of multiple data sources. The Common Model Data of GISE allows for the representation of local and integrated data sources through XML. GISE have layers of components grouped by Client Layer, Metadata Services Layer, Integration Services Layer (Mediator), Globus Toolkit Services Layer, Data Source Access Service Layer and Data Sources Layer. In this work, a prototype was developed and tested. We also present a case study using data obtained from public health agencies.

SUMÁRIO

Ambos os projetos foram inovadores e alcançaram sucesso. Estes projetos fazem parte da vanguarda da super-computação e serviram como base para muitos projetos sucessores na Grade Computacional. O FAFNER foi o precursor de projetos com o SETI@Home e o I-WAY de projetos como o *Globus* (GLOBUS, 2004) e o *Legion* (GRIMSHAW & WULF, 1997).

2.2.2 Segunda Geração

Na segunda geração da evolução das Grades Computacionais aconteceram projetos de pesquisa voltados para suprir os desafios dos ambientes até então propostos. Os principais desafios encontrados na segunda geração da evolução das Grades Computacionais focam nos problemas de heterogeneidade, utilização em múltiplos domínios administrativos, escalabilidade, adaptabilidade e controle distribuído.

A Grade Computacional possui um grande número de recursos interconectados. Para que seja possível disponibilizar um ambiente coordenado e cooperativo é necessário o suporte aos aspectos envolvidos na construção desta infra-estrutura computacional, como por exemplo, (ROURE et al., 2003):

- **Hierarquia Administrativa:** A hierarquia administrativa determina como as informações serão organizadas na Grade Computacional com a introdução de novas organizações;
- **Serviços de Comunicação:** As aplicações devem fornecer suporte para trabalhar com diferentes formas de comunicação na Grade Computacional. As formas de comunicação variam desde transferências confiáveis ponto-a-ponto até múltiplas transferências simultâneas para múltiplos pontos não confiáveis (*multicast*). Os serviços de rede utilizados pela Grade Computacional devem fornecer suporte adequado para que seja possível oferecer qualidade de serviço ao usuário;

- **Serviços de Informação:** Na Grade Computacional os tipos e disponibilidade dos serviços variam constantemente. Para resolver este problema existe a necessidade que seja disponibilizada ao usuário mecanismos capazes de retornar as informações sobre os serviços disponíveis de forma fácil, confiável e precisa;
- **Serviços de Nomes:** Os nomes são utilizados na Grade Computacional e em qualquer outro ambiente computacional para se referir a um objeto em particular dentro de um determinado conjunto. Na Grade Computacional, o serviço de nomes deve fornecer, em todos os pontos do ambiente distribuído, uma forma padrão para a nomeação. Podemos citar como exemplo de serviços de nomes, o conhecido DNS (*Domain Name Service*), utilizado na Internet;
- **Segurança e Autorização:** A infra-estrutura de segurança é um aspecto de grande importância da Grade Computacional. A segurança é um assunto complexo que necessita de diversos recursos administrados de forma autônoma, que interagem de maneira que não causem impacto no uso dos recursos compartilhados e também não permitam falhas de segurança nos sistemas individuais ou distribuídos; e
- **Interface e Forma de Acesso:** É altamente desejável o acesso simples e intuitivo aos diversos serviços, recursos e aplicações heterogêneas disponíveis na Grade Computacional.

A segunda geração foi contemplada com o surgimento de softwares para a construção de Grades Computacionais. Muitos destes, além de oferecer serviços e protocolos básicos, deram suporte aos desafios e requisitos computacionais identificados nesta geração.

Entre os projetos conhecidos na segunda geração, podemos destacar: o *Globus Toolkit*, o *Legion* e o *Condor*. Estes projetos são discutidos respectivamente nas sessões 2.4.1, 2.4.2 e 2.4.3.

2.2.3 Terceira Geração

As pesquisas realizadas na segunda geração da evolução das Grades Computacionais, deram suporte a uma visão de uma infra-estrutura computacional preocupada com os requisitos e funções necessárias para o suporte de aplicações em um ambiente coordenado e geograficamente distribuído (FOSTER et al., 2002).

Na terceira geração, foram explorados diferentes aspectos de engenharia, tais como: o reuso do código–fonte de componentes existentes e a utilização de metadados. Também na terceira geração da evolução das Grades Computacionais, destacam-se aspectos de segurança e automação. A possibilidade de integrar inúmeros recursos culmina na necessidade de uma gerência especializada que pode ser realizada através de escalonadores. Os escalonadores decidem qual recurso do ambiente computacional deve ser alocado para execução de uma tarefa. Para realizar esta operação são utilizadas as informações contidas nos serviços de informação e metadados. Os metadados armazenam informações sobre o estado, condições e disponibilidade de utilização dos recursos da Grade Computacional (ROURE et. al., 2003).

As soluções na terceira geração envolvem a adoção de um modelo de Grade Computacional orientada a serviço, maximizando a importância dos metadados. Através desta combinação, os serviços na Grade Computacional podem ser construídos de maneira flexível e extensível, permitindo através da composição, serviços providos com maiores funcionalidades.

Os metadados podem ser utilizados para construir serviços capazes de descobrir automaticamente as funcionalidades e disponibilidades de recursos distribuídos, oferecendo aos usuários serviços com funcionalidades avançadas (SILVA & SENGER, 2004).

A visão de Grades Computacionais orientadas a serviço tem como base o sucesso da tecnologia de *Web Services* (FOSTER et al., 2002). Esta tecnologia foi criada inicialmente para resolver problemas de integração de softwares. Através dos *Web Services* todas as

aplicações são capazes de se comunicar, independente da arquitetura computacional, sistema operacional ou linguagem de programação.

Na terceira geração de evolução das Grades Computacionais uma iniciativa de pesquisa da IBM, em conjunto com o Projeto *Globus*, criou a especificação OGSA (*Open Grid Service Architecture*) (OGSA, 2003; FOSTER et al., 2002). O OGSA disponibiliza padrões e componentes necessários para a construção de Grades Computacionais orientadas a serviços. Neste novo conceito de Grade Computacional, os *Web Services* evoluíram para *Grid Services*. Os *Grid Services* são *Web Services* especializados para serem utilizados na Grade Computacional.

O *Globus Toolkit 3.x* (GT3) (SOTOMAYOR, 2005) foi construído para fornecer suporte à arquitetura OGSA através da especificação OGSI. A especificação e desenvolvimento destes padrões fornecem suporte para a interoperabilidade entre aplicações. A interoperabilidade permite a construção de sistemas de Grades Computacionais através das combinações de diversas implementações de Grade Computacional.

Com a evolução da tecnologia de *Web Services*, fez se necessário à atualização da especificação OGSI utilizado no GT3 para a nova versão do *Globus Toolkit* (GT4). Nesta versão do *Globus Toolkit*, a especificação OGSI foi substituída pela especificação WSRF (*Web Services Resource Framework*) (CZAJKOWSKI et al., 2005).

Os aspectos e características relacionadas à Grade Computacional baseada em serviços são expostos na próxima seção, tais como: tecnologias associadas, forma de descoberta e composição dos serviços. O Capítulo finaliza com uma discussão sobre as plataformas para Grade Computacional.

2.3 Grades Computacionais Baseadas em Serviços

Os requisitos que motivam os esforços de pesquisa na Grade Computacional são encontrados na área acadêmica e comercial. Portanto, prover uma infra-estrutura capaz de integrar recursos e serviços de forma coordenada, distribuída e eficiente são pontos de comum de interesse. Do ponto de vista acadêmico, a Grade Computacional possibilita a cooperação técnica e científica entre Centros de Pesquisas geograficamente distribuídos, com o objetivo da geração de novos conhecimentos. Do ponto de vista comercial, a Grade Computacional disponibiliza uma arquitetura capaz de executar serviços computacionais especializados sob demanda, através da integração de recursos e sistemas (FOSTER et al., 2004).

Os requisitos encontrados nestas áreas impulsionaram convergência das tecnologias utilizadas em ambas as comunidades. Desta forma, a Grade Computacional passou a utilizar uma metodologia voltada para execução de serviços especializados, utilizando como tecnologia padrão os *Web Services* (KREGER, 2004).

Um serviço na Grade Computacional é um recurso compartilhado, seja ele hardware ou software, que possa ser descrito através de uma interface comum, com objetivo de satisfazer as necessidades de um cliente (FOSTER et al., 2002).

2.3.1 Tecnologias Associadas

O conceito de arquitetura orientada a serviços encapsula o estado de um serviço e permite que o mesmo seja acessível através de uma interface bem definida. Através deste conceito, é possível fazer uso de diferentes serviços especializados, utilizando protocolos claramente definidos.

Além dos *Web Services*, existem diversas tecnologias para a computação distribuída, onde podemos citar como exemplo o RMI (*Remote Method Invocation*) (CHEDE, 2004) e o

CORBA (*Common Object Request Broker Access*) (CORBA, 2005). Porém, ambas as tecnologias não são adequadas para a utilização na Grade Computacional.

O RMI não é adequado para a utilização na Grade Computacional porque não ofornece interoperabilidade entre os serviços, obrigando que seu cliente seja implementado utilizando a linguagem *Java* e também conheça antecipadamente sua interface de acesso.

Por outro lado, o CORBA permite interoperabilidade entre seus clientes, pois são descritos utilizando uma interface de acesso independente da linguagem de programação. O CORBA utiliza como tecnologia para prover acesso transparente aos serviços um componente chamado ORB (*Object Request Broker*) que se comunicam através do protocolo IIOP.

Todavia, a aplicação de CORBA na Grade Computacional falha em sua camada de transporte. O protocolo IIOP não é um protocolo comum para a troca de mensagens e não é capaz de ultrapassar *Firewalls* sem configurações adicionais por seus administradores. Este aspecto inviabiliza seu uso em um ambiente altamente dinâmico e distribuído como a Grade Computacional.

A tecnologia de *Web Services* é ideal para a construção de sistemas computacionais distribuídos na Grade Computacional. Os *Web Services* permitem a interoperabilidade entre seus diversos clientes, não obrigando o usuário a utilizar uma linguagem de programação específica (o que acontece no RMI) e não obriga o usuário a conhecer antecipadamente a interface de acesso ao serviço, permitindo que a mesma seja conhecida em tempo de execução.

Os *Web Services* utilizam o protocolo SOAP (*Simple Object Access Protocol*) para a troca de mensagens. O SOAP é uma especificação que define uma gramática de mensagens que não são vinculadas a nenhuma arquitetura de hardware ou software. As mensagens SOAP podem ser encapsuladas e transmitidas pelo difundido protocolo http (*Hiper Text Transfer Protocol*), utilizado na Internet (POTTS & KOPACK, 2003).

A Figura 2 ilustra os passos necessários para o acesso a um serviço utilizando as tecnologias de *Web Services* (HASS, 2002). No primeiro instante, o cliente faz uma requisição a um catálogo denominado UDDI (*Universal Description, Discovery and Integration*) (passo 1). O UDDI é um repositório de informações que disponibiliza dados sobre a localização dos serviços desejados. A requisição aos registros do UDDI pode ser executada por um cliente ou aplicação em tempo de execução, informando apenas quais são as características do serviço desejado.

Após receber a requisição para localização do serviço, o UDDI informa ao solicitante qual o servidor que hospeda o Web Service (passo 2).

O cliente, após receber o endereço do UDDI, se comunica com o servidor para obter as informações necessárias sobre os parâmetros utilizados para acessar corretamente o serviço (passo 3). Após receber esta solicitação, o servidor envia ao cliente um documento WSDL (*Web Service Description Language*) (passo 4).

O documento WSDL descreve a interface de acesso do Web Service. Através do WSDL o cliente descobre quais são os métodos e os parâmetros necessários para utilizar o serviço. Em seqüência o cliente faz uma requisição SOAP ao servidor (passo 5) que após o processamento, retorna os resultados solicitados ao cliente (passo 6).

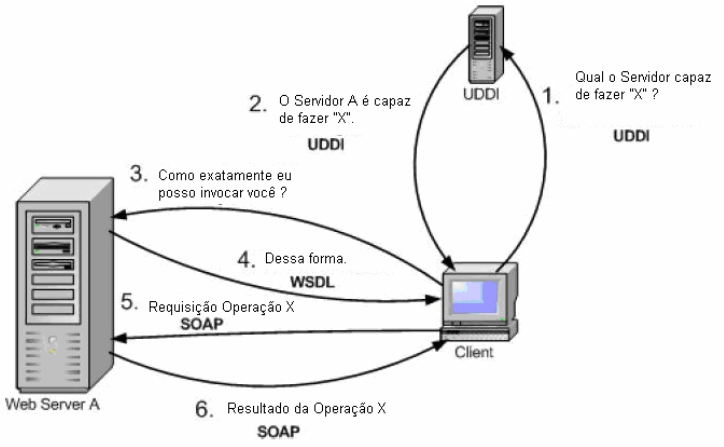


Figura 2: Acesso a um Web Service (HASS, 2002).

2.3.2 Descoberta de Serviços Computacionais

A Grade Computacional é uma infra-estrutura altamente dinâmica. Por este motivo, a mesma deve disponibilizar mecanismos capazes de descobrir dinamicamente serviços e recursos computacionais.

O requisito de um mecanismo de descoberta é que ele tenha a mesma utilidade de um catálogo telefônico de páginas amarelas. Desta forma, os usuários podem localizar, conforme suas necessidades, informações sobre diversos prestadores de serviços.

A Grade Computacional é uma infra-estrutura que pode disponibilizar inúmeros serviços aos seus usuários. Neste cenário, é de extrema importância a utilização de mecanismos eficientes e eficazes para a descoberta de serviços computacionais. Os serviços computacionais podem ser: ciclos de CPU, armazenamento em disco, etc (SOTOMAYOR, 2005).

Para disponibilizar um mecanismo para descoberta de dados, a tecnologia de *Web Services* utiliza o UDDI. O UDDI é um Web Service capaz de compor um catálogo global de todos os serviços disponíveis na Grade Computacional (POTTS & KOPACK, 2003).

O UDDI pode ser público ou privado. Nos registros públicos não existem restrições, qualquer serviço pode ser registrado e qualquer usuário pode consultar as informações armazenadas no registro. Os registros privados divulgam informações referentes apenas a um determinado domínio em um grupo seletivo de usuários, como por exemplo: a publicação dos serviços disponíveis de uma empresa para seus funcionários (UDDI, 2005).

O UDDI tradicional falha em alguns aspectos quando submetido aos requisitos existentes na Grade Computacional. O UDDI possui controle centralizado, caracterizando um ponto único de falha.

Na literatura, existem algumas alternativas para a construção de mecanismos de descoberta de serviços computacionais na Grade Computacional, como por exemplo, o WSPDS (*Web Service Peer– to-peer Discovery Service*).

O WSPDS se diferencia do UDDI formando uma rede ponto–a–ponto de registros. Na metodologia de trabalho do WSPDS, quando um ponto da rede (nó) recebe uma solicitação que não possa ser atendida, o nó se encarrega de transferir a solicitação a outro nó da rede, limitando–se a um número definido de transferências de consultas. O WSPDS é um mecanismo descentralizado e não é totalmente confiável. No WSPDS existe a possibilidade de um conjunto de nós não serem acessados no processo de consulta devido à sua limitação na propagação da consulta em um conjunto limitado de nós (BANAEI-KASHANI et al., 2004).

Uma forma de se resolver o problema da descoberta de serviços na Grade Computacional é utilizar uma metodologia que combina as vantagens do UDDI e do WSPDS.

Na Grade Computacional, é possível utilizar o MDS (*Monitoring and Discorey Service*) para a descoberta de serviços computacionais. O MDS é um serviço capaz de armazenar e disponibilizar informações sobre outros serviços disponíveis na Grade Computacional. Desta forma, uma aplicação pode obter dados do serviço antes de invocá-lo e determinar em tempo de execução, qual o serviço adequado para executar um processamento (MDS, 2005).

Com as informações disponibilizadas pelo MDS uma aplicação também é capaz em tempo de execução, realizar a composição de serviços especializados e de alto nível aos seus usuários (MDS, 2005).

O MDS possui mecanismos capazes de coletar informações sobre os serviços de uma Grade Computacional em múltiplas fontes de dados. O MDS é disponibilizado no *Globus Toolkit* versões 2.x, 3.x e 4.x (FOSTER, 2005).

2.3.3 Composição e Coordenação de Serviços

Os sistemas computacionais que executam algum tipo de processamento (serviço) desenvolvidos até pouco tempo, eram tipicamente fechados (sistemas legados), onde caso houvesse a necessidade de uma extensão as suas funcionalidades, era necessário substituir toda aplicação ou criar uma nova aplicação que pudesse, de alguma forma, integrar-se com o sistema principal (SZYPERSKI, 1996). Porém, um serviço pode ser composto de diversos outros serviços, como por exemplo: uma agência de viagens que comercializa pacotes turísticos. Esta agência de viagens não é a responsável pela execução das atividades como o transporte, hospedagem e passeios. A agência comercializa um serviço composto (pacote turístico) e as empresas participantes desta prestação de serviços são transparentes ao usuário.

Na Grade Computacional é utilizada a mesma analogia e um serviço pode ser composto de diversos outros serviços.

A utilização de serviços compostos na Grade Computacional permite que a complexidade do serviço seja ocultada ao usuário e o mesmo não precisa conhecer “a priori” os detalhes dos serviços menores que compõem o serviço composto.

Com a utilização de serviços compostos, podemos também reduzir o tempo para a construção de novos serviços através da reutilização do código–fonte (FOSTER, 2005).

Para a utilização de serviços compostos, é necessário que o serviço responsável pela invocação dos serviços menores possua informações que sejam capazes de determinar quais são os possíveis serviços e em qual momento eles podem ser utilizados.

A coordenação é um aspecto de grande importância. Através desta, determinamos à sincronização e temporização dos serviços menores que compõem o serviço composto. A coordenação de serviços permite a criação de serviços dinâmicos e adaptáveis, compatíveis com os requisitos da Grade Computacional.

Os *Web Services* e os *Grid Services* são tecnologias que suportam inúmeros processos de negócio e relações entre empresas. Desta forma, várias propostas têm surgido ao que se refere à modelagem de processos para *Web Services*, como por exemplo: O BPEL4WS (*Business Process Execution Language for Web Services*) (MACEDO et al., 2004).

O BPEL4WS provê uma linguagem para especificação formal de processos de negócio e protocolos de interação, estendendo o modelo de interação dos *Web Services.* O BPEL4WS define um modelo de integração que facilita a expansão da integração automatizada de processos entre empresas (ANDREWS et al., 2003).

Um processo BPEL4WS pode definir uma função na regra do negócio usando a notação de processo abstrato. Por exemplo, no fornecimento de produtos, o comprador e o vendedor são papéis distintos, cada um com seu processo abstrato. Em BPEL4WS o relacionamento entre os processos é tipicamente modelado como um processo associado. Os processos abstratos utilizam os conceitos de BPEL4WS e a metodologia no tratamento de dados reflete o nível de abstração requerido para descrever os aspectos públicos das regras de negócio. Especificamente, processos abstratos lidam somente com dados relevantes as regras do negócio (ANDREWS et al., 2003).

Pode-se utilizar BPEL4WS para definir um processo de negócio executável para *Web Services.* A lógica e o estado do processo determinam à natureza e a seqüência das interações dos *Web Services* conduzida para cada papel na regra de negócio. (ANDREWS et al., 2003).

Na Grade Computacional, os serviços de notificação permitem que serviços distribuídos possam se comunicar de forma assíncrona sobre as significantes alterações em seu estado. Desta forma, os serviços podem ser construídos através da composição de outros serviços.

A interface de descoberta do OGSA pode ser utilizada para que o serviço principal localize outros serviços capazes de executar o processamento necessário. A interface de

Criação Dinâmica de Serviços é utilizada para invocar automaticamente a execução de serviços de menor nível para geração do serviço composto (FOSTER, 2002). O OGSA também disponibiliza interface para o gerenciamento do Ciclo de Vida do Serviço. Desta forma, podemos realizar o encadeamento automático de serviços para a geração de um serviço composto.

2.4 Plataformas para Grade Computacional

A seguir, são identificadas e discutidas algumas plataformas para Grade Computacional. A seção inicia com uma introdução sobre ao *Globus* e na seqüência é exposta a arquitetura e serviços do *Globus Toolkit* 2.x, 3.x e 4.x. Finalmente, após a discussão sobre o *Globus*, é realizada uma breve exposição sobre o *Legion* e o *Condor*.

2.4.1 Globus

O *Globus* é um projeto criado inicialmente por Ian Foster da Divisão de Matemática e Ciência da Computação do Laboratório de Argonne (EUA) e Carl Kesselman do Instituto de Ciências da Informação da Universidade de Sul da Califórnia (EUA) (FOSTER & KESSELMAN, 1998). O *Globus* disponibiliza uma infra-estrutura de software capaz de permitir que as aplicações compartilhem recursos heterogêneos e geograficamente distribuídos (FOSTER, 2002).

O Projeto *Globus* iniciou engajado na resolução de problemas de configuração e otimização de desempenho em ambientes de metacomputação (FOSTER & KESSELMAN, 1997). O *Globus* é composto por uma arquitetura claramente definida de protocolos e serviços. Conforme ilustrado na Figura 3, o *Globus* emergiu em 1997 como um padrão de fato e hoje é a arquitetura de maior utilização no mundo para a construção de Grades Computacionais.

Em 1997 o *Globus* disponibilizou um *middleware*, que é um toolkit focado na interoperabilidade de recursos computacionais, denominado de *Globus Toolkit* 2.x (GT2). Atualmente, o *Globus* está disponibilizando o *Globus Toolkit* 4.x (GT4).

O *Globus Toolkit* é um software de código aberto, extensível, escrito na linguagem C (até a versão 3.2) e na linguagem *Java* (a partir da versão 4.x) que fornece serviços de segurança, gerenciamento de dados, recursos computacionais e serviços de informação. Atualmente, muitas empresas como a IBM e a Oracle disponibilizam seus produtos com suporte para o *Globus Toolkit* (FOSTER et al., 2002).

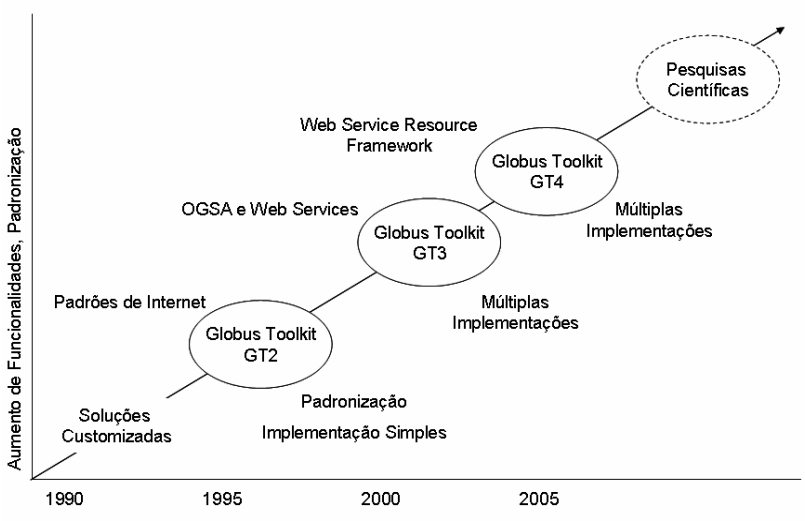


Figura 3: Evolução do Globus Toolkit.

Com o *Globus* é permitido que as aplicações utilizem recursos distribuídos compartilhados, como computadores, dispositivos de armazenamento, dados, sensores, etc. Desta forma, é possível a integração de recursos e serviços que não são encontrados localmente, como por exemplo: um cientista que necessita acessar diversos bancos de dados em diferentes locais, uma indústria que necessita alocar grande poder computacional por um determinado período de tempo ou até mesmo um astrônomo que necessita distribuir terabytes de dados em um determinado dia para diversos lugares no mundo (FOSTER, 2005).

2.4.1.1 Globus Toolkit 2.x

O *Globus Toolkit* 2.x (GT2) é dividido em camadas. As camadas são compostas por componentes que compartilham das mesmas características e se comunicam com outras camadas, disponibilizando o funcionamento da Grade Computacional (FOSTER et al., 2001). A Figura 4 ilustra as camadas da arquitetura do *Globus Toolkit* 2.x.

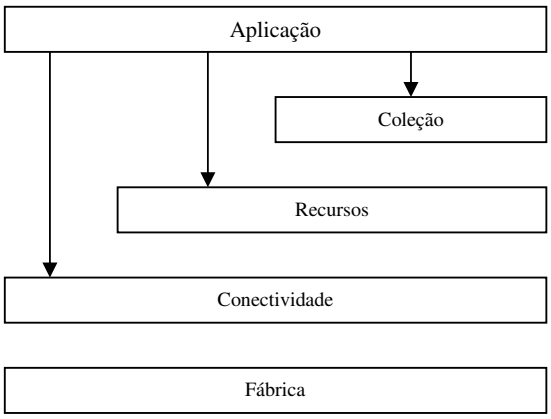


Figura 4: Camadas da Arquitetura do Globus Toolkit 2.x (FOSTER et al., 2001).

Camada *Fábrica*

O *Globus Toolkit* 2.x se concentrou na implementação de protocolos utilizados para o gerenciamento de recursos locais, forma de acesso e controle.

Os componentes desta camada são capazes de descobrir informações dos recursos locais compartilhados, tais como: estruturas e estados de computadores, softwares, rede, etc (FOSTER et al., 2001).

Embora o gerenciamento de recursos locais seja executado por administradores do domínio, o GT2 disponibiliza nesta camada um protótipo do GARA (*General-Purpose Architecture for Reservation and Allocation*) utilizado para realizar a reserva e alocação de recursos (FOSTER et al., 2000).

Camada de *Conectividade*

A camada de *Conectividade* se preocupa com a segurança na Grade Computacional. A segurança pode se tornar um problema crítico, uma vez que a arquitetura envolve múltiplos domínios administrativos e a autenticação do usuário em cada domínio é uma prática inviável (FOSTER et al., 2001).

A camada de conectividade é composta por componentes que provêem segurança no acesso aos serviços e comunicação na Grade Computacional. Esta camada é baseada nos protocolos do GSI (*Globus Security Infrastructure*) e disponibilizam o login único para toda a Grade Computacional utilizando a criptografia de chave pública, certificados X.509 e comunicação através de SSL (*Security Socket Layer*). Desta forma, é estabelecida e garantida a identidade de um usuário *Globus* (WELCH et al., 2003).

Camada de *Recursos*

A camada *Recursos* do *Globus Toolkit* 2.x disponibiliza componentes para o gerenciamento de recursos individuais. Devido a Grade Computacional ser um ambiente altamente distribuído, dinâmico e escalar, é inviável que a mesma possua um escalonador que controle todo o sistema, pois este seria um ponto único de falha e teríamos problemas de desempenho nos processos de consulta e atualização de dados (FOSTER et al., 2001).

Quando um usuário *Globus* submete uma aplicação para execução na Grade Computacional um escalonador de aplicação é acionado. O escalonador de aplicação tem como objetivo escolher entre os recursos computacionais disponíveis quais serão utilizados. Na seqüência, o escalonador de aplicação divide as atividades entre os recursos computacionais escolhidos e as encaminha diretamente para os escalonadores do recurso (CZAJKOWSKI et al., 2001).

Para executar a submissão e controle de tarefas, o *Globus* utiliza o GRAM (*Globus Resource Allocation Manager*) (CZAJKOWSKI et al., 1998). O GRAM também prove informações sobre o estado dos recursos ao MDS (*Monitoring and Discovery Service*). O MDS é um componente da camada *Coletiva.*

Conforme ilustrado na Figura 5, o GRAM é composto pelos componentes *Gatekeeper*, *Job Manager* e *GRAM Reporter*. O benefício de utilizar o GRAM é que ele funciona independente de qual escalonador de recurso local está sendo utilizado, pois as requisições enviadas a ele são feitas no formato RSL *(Resource Specification Language)*. O cliente GRAM submete uma requisição ao *Gatekeeper* e este se comunica diretamente com o GSI para obter informações sobre a identidade do usuário *Globus*. O Job Manager traduz a requisição para que seja entendida pelo escalonador local e a transmite para o mesmo. O *GRAM Reporter* consulta os escalonadores de recursos locais e disponibiliza informações ao MDS. O MDS é consultado pelo cliente antes de executar a requisição (CZAJKOWSKI et al., 1998).

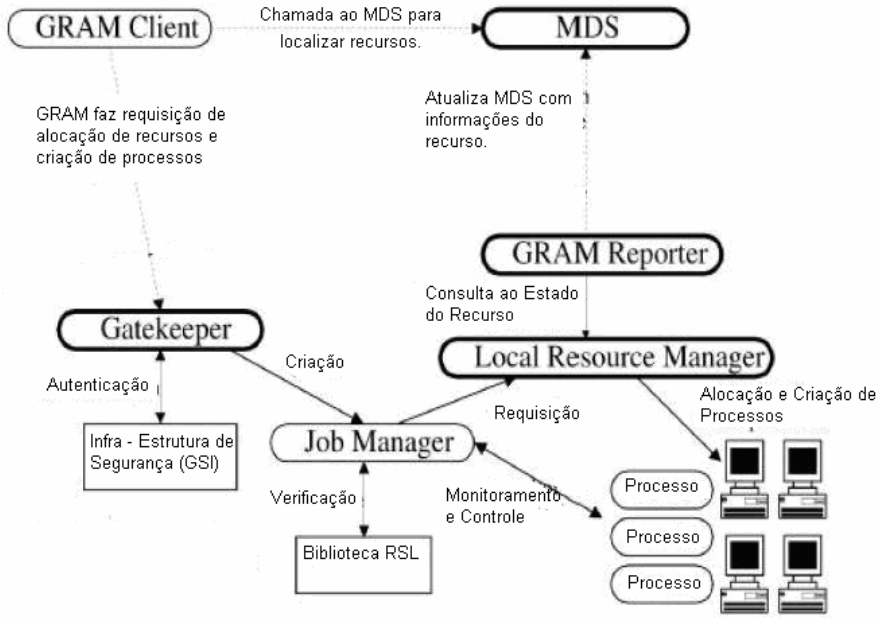


Figura 5: Funcionamento do GRAM (CZAJKOWSKI et al., 1998).

A camada de *Recursos* também possui o componente GridFTP. O projeto *Globus* sabendo das limitações do protocolo FTP (*File Transfer Protocol*) na Grade Computacional, desenvolveu o GridFTP. Este protocolo é basicamente uma extensão do protocolo FTP com suporte as necessidades encontradas na Grade Computacional (ALLCOCK, 2002).

O GridFTP possui suporte para a transferência paralela de dados, ou seja, várias conexões TCP/IP entre a fonte e a transferência de dados *striped*. A transferência de dados *striped* é uma conexão entre várias fontes e um destino ou vice – versa (ALLCOCK, 2002).

Camada *Coletiva*

Enquanto a camada de Recursos é focada em interações do usuário ou aplicação com um recurso computacional local, a camada *Coletiva* contém protocolos e serviços que não são associados a um recurso específico. A camada *Coletiva* é composta por serviços de natureza global e captura as interações entre os recursos locais (FOSTER et al., 2001).

A camada *Coletiva* possui o MDS (*Monitoring and Discovery Service*). O MDS obtém as informações distribuídas que são disponibilizadas em diferentes recursos ou serviços na Grade Computacional e as disponibiliza aos usuários. O MDS pode disponibilizar informações como: recursos, localização, propriedade, custo, entre outras (CZAJKOWSKI et al., 2001).

A camada *Coletiva* possui um serviço para o gerenciamento de réplicas de arquivos denominado RLS *(Replica Location Service)*. O RLS mantém e provêem acesso para mapeamento de nomes lógicos (apelidos) de arquivos para seus respectivos caminhos de acesso (*target name*).

A replicação de arquivos reduz a latência de acesso, aumenta a escalabilidade e o desempenho de aplicações distribuídas.

Para disponibilizar o Serviço de Réplicas, o RLS mantém uma lista de índice com todas as estruturas que estão replicadas na Grade Computacional. Para cada entrada na lista de índice, são armazenados conjuntos de pares de mapeamento de nomes lógicos (apelidos) para seu caminho de acesso (RLS-GT, 2005).

Camada *Aplicação*

A camada final da arquitetura do *Globus Toolkit* 2.x é compreendida pelas aplicações do usuário que operam na Grade Computacional. As aplicações podem fazer uso dos serviços da camada *Conectividade, Recursos e Coleção*. Estas camadas possuem protocolos para acesso aos serviços disponibilizados, como por exemplo: gerenciamento e alocação de recursos, serviços de informação, serviços de localização de réplicas, etc.

2.4.1.2 Globus Toolkit 3.x

Com o objetivo de se aproveitar as características e vantagens da Grade Computacional na execução de serviços especializados em comparação com outras arquiteturas distribuídas, houve uma convergência natural entre as tecnologias de Grade Computacional e os *Web Services.* Assim nasceu a tecnologia de *Grid Services*.

O *Globus Toolkit* 3.x (GT3) é um *middleware* que possui uma arquitetura claramente definida e um conjunto de bibliotecas de código aberto para a construção de Grades Computacionais (FOSTER & KESSELMAN, 1999). O GT3 corrige os problemas encontrados no GT2 e define uma arquitetura para construção de Grade Computacional baseada em serviços denominada OGSA (*Open Grid Service Architecture*). O OGSA propõe uma infra-estrutura de sistemas e aplicações que são requeridas para a integração e gerenciamento de serviços dento de uma organização virtual distribuída, heterogênea e dinâmica (FOSTER et al., 2002).

Os *Grid Services* são *Web Services* especializados pela especificação OGSA para serem utilizados na Grade Computacional. Com OGSA é possível a utilização de diferentes *Grid Services*, de forma transparente ao usuário. As interfaces possuem baixo acoplamento com o provedor do serviço e mudanças no código–fonte do serviço não refletem mudanças em sua interface e/ou forma de acesso (FOSTER et al., 2002).

A arquitetura OGSA define dois tipos distintos de serviços: Os serviços persistentes e os serviços transientes. Os serviços persistentes são os serviços ativados pelo usuário (ou outro serviço) e tem seu tempo de vida indefinido. O serviço persistente deixa apenas de responder as requisições quando é desativado. Podemos destacar como exemplo de serviço persistente na Grade Computacional o GSF (*Grid Service Factory*). O GSF tem como responsabilidade receber as requisições sob demanda aos serviços transientes e disponibiliza-los.

Os serviços transientes têm como objetivo executar os processamentos necessários para atender uma requisição. Seu tempo de vida é gerenciado pelo GSF responsável por sua criação e o *Grid Service* pode ser ou não destruído após completar seu processamento. A vantagem da utilização de serviços transientes é a liberação dos recursos computacionais para outros *Grid Services*. Os *Grid Services* são identificados por um GSH (*Grid Service Handle*) único, informado ao cliente através do *Grid Service Factory* e utilizado para fazer acesso ao serviço desejado (FOSTER et al., 2002).

A arquitetura OGSA também disponibiliza o GSR (*Grid Service Registry*). O GSR é um serviço persistente responsável por armazenar informações sobre os *Grids Services Factorys* e outros serviços persistentes. As informações armazenadas pelo GSR são disponibilizadas aos clientes da Grade Computacional para que os mesmos possam saber quais são os serviços que estão disponíveis para uso (FOSTER et al., 2002).

A arquitetura OGSA é implementada por mecanismos da especificação OGSI (*Open Grid Service Infrastructure*) (TUECKE et al., 2003). No OGSI são definidos protocolos, interfaces, propriedades e atributos para a construção de serviços suportando invocação confiável e segura, gerenciamento do tempo de vida, notificação, políticas de acesso, virtualização, etc (SILVA & SENGER, 2004).

A versão 1.0 do OGSI foi implementada no GT3 em julho de 2003 e foi utilizada até a versão 3.2 do *Globus Toolkit*.

A Figura 6 mostra o relacionamento entre OGSA, OGSI e o *Globus* (SOTOMAYOR, 2005). Os *Grid Services* são definidos pela especificação OGSA. O comportamento dos *Grid Services* é definido pela especificação OGSI e os *Web Services* são adaptados para serem *Grid Services* pela especificação OGSA.

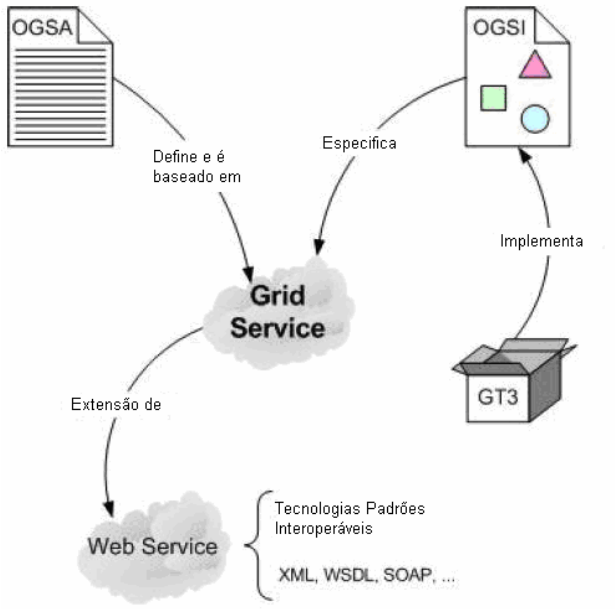


Figura 6: Relacionamento entre OGSA, OGSI e Web Services (SOTOMAYOR, 2005).

2.4.1.3 Globus Toolkit 4.x.

O *Globus Toolkit* 4.x (GT4) é um software para a construção de Grades Computacionais que disponibiliza um grande número de componentes utilizados no gerenciamento de infra-estrutura da Grade Computacional, ferramentas para a construção de novos *Web Services* e um forte padrão de segurança da infra-estrutura computacional distribuída. O conjunto destes componentes permite a formação de um rico ecossistema de componentes que trabalham em conjunto ou com aplicações clientes em diferentes domínios (FOSTER, 2005).

Para a implementação de *Grid Services* no GT4 é utilizada a especificação WSRF (*Web Services Resource Framework*). A especificação WSRF é uma versão melhorada da especificação OGSI (utilizada no GT3) e é compatível com os novos padrões de *Web Services* (OGSA-DAI-WSRF, 2005).

As interfaces dos serviços e as estruturas dos componentes no GT4 são definidas através da utilização dos mecanismos de *Web Services.* Desta forma, são providos mecanismos extensíveis e flexíveis para descrever, descobrir e invocar os serviços na Grade Computacional.

A maioria das aplicações construídas é geralmente focada em operações específicas de um determinado domínio, como por exemplo: análise de crédito, geração de folha de pagamento, etc. O GT4 foca seus esforços nas operações de gerenciamento da infra-estrutura computacional voltada para a distribuição, uso coordenado e cooperativo dos recursos computacionais.

Na arquitetura proposta neste trabalho, o GT4 é utilizado para disponibilizar a infra–estrutura necessária para a execução das atividades de integração de fonte de dados. Os componentes da arquitetura do GT4 são discutidos nos parágrafos a seguir.

Arquitetura do Globus 4.x

A arquitetura do *Globus* 4.x (GT4) é dividida principalmente em três conjuntos de componentes, conforme ilustrado Figura 7.

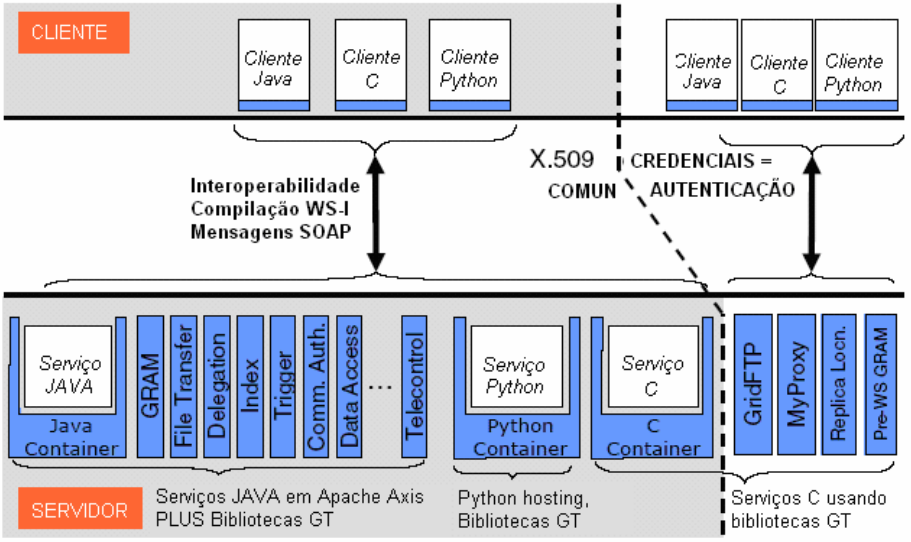
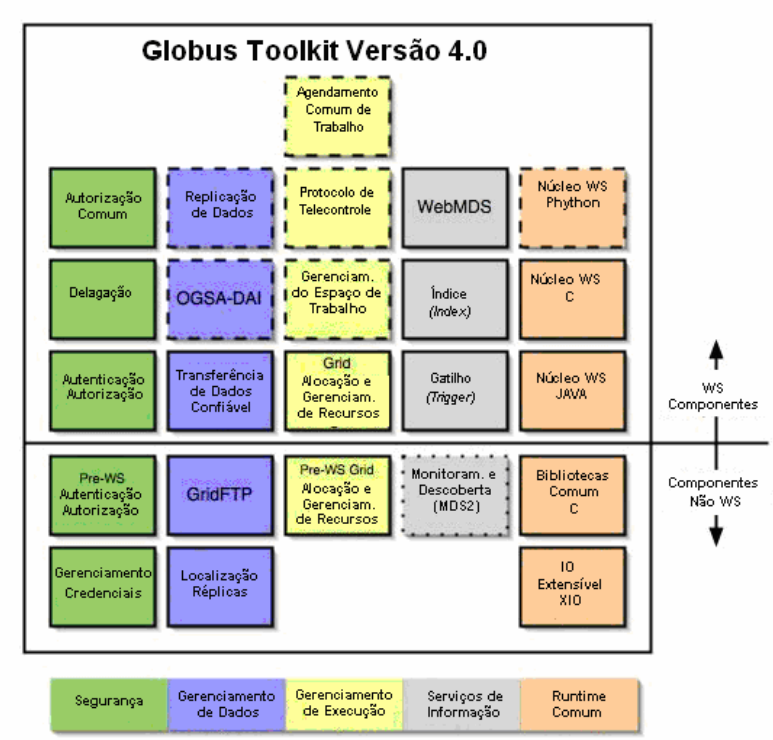


Figura 7: Arquitetura do Globus 4.x (FOSTER, 2005).

Inicialmente, identificamos na arquitetura do GT4 a implementação de um conjunto de serviços para o gerenciamento da infra-estrutura, tais como: serviços para o gerenciamento da execução de tarefas (GRAM), acesso e transferência de dados (GridFTP), gerenciamento de réplicas (RLS), ente outros. Podemos identificar também que são oferecidos três *containeres* para a construção de novos serviços pelo usuário. Estes serviços podem ser escritos nas linguagens *C*, *Java* ou *Python* e utiliza um conjunto de bibliotecas para a execução de suas operações (FOSTER, 2005).

Componentes do Globus 4.x (GT4)

A Figura 8 ilustra os componentes do GT4 agrupados através de suas funcionalidades que são discutidas a seguir:



As linhas pontilhadas caracterizam previsões técnicas

Figura 8: Componentes do GT4 (FOSTER, 2005).

Gerenciamento a Execução de Tarefas

A submissão, monitoramento e gerenciamento das tarefas no GT4 são realizados pelo GRAM (*Grid Resource Allocation Management*). O GT4 provê uma interface em *Web Services* para acesso a este serviço pelos usuários ou aplicações. Desta forma, é possível monitorar e gerenciar a execução, distribuição e submissão de tarefas nos diferentes nós da Grade Computacional.

O WMS (*Workspace Management Service*) é um serviço que disponibiliza uma alocação dinâmica de recursos para contas Unix e o GTCP (*Grid TeleControl Protocol*) é um protocolo para gerenciar a utilização remota de recursos na Grade Computacional. Ambos os serviços são uma previsão técnica do GT4 (FOSTER, 2005).

Acesso e Movimentação de Dados

Uma grande quantidade de dados distribuídos é acessada por diferentes aplicações na Grade Computacional. O GT4 resolve este problema com a utilização do GridFTP.

O GridFTP é um protocolo que permite uma transferência de dados rápida de disco para disco ou de memória para memória e é capaz de operar em conjunto com o FTP tradicional.

O RFT (*Reliable File Transfer*) é o serviço que disponibiliza uma transferência de dados segura entre múltiplas instâncias do GridFTP.

O RLS (*Reliable Location Service*) é o serviço responsável por disponibilizar o acesso a múltiplas réplicas de arquivos.

O DRS (*Data Replication Service*) é um serviço de alto nível que combina o componente RFT, RLS e GridFTP. O componente DRS é uma previsão técnica no GT4 provê a capacidade de replicação de conjuntos de arquivos na Grade Computacional.

A função do DRS é assegurar a replicação confiável de conjuntos de arquivos nos dispositivos de armazenamento. O componente DRS inicia a execução com uma consulta ao RLS para descobrir onde estão os arquivos desejados para replicação e após serem localizados, o DRS cria uma requisição de transferência que é executada pelo RFT. Quando a transferência estiver completada, o DRS registra as novas réplicas no RLS (DATA, 2005).

O DRS é implementado como um Web Service e compilado com a especificação WSRF. Quando uma requisição DRS é recebida, é mantido o estado sobre cada arquivo inicialmente replicado, incluindo quais operações nos arquivos foram realizadas com sucesso e quais operações falharam (DATA, 2005).

A implementação inicial do DRS contém um recurso chamado de Replicador. O Replicador aceita requisições de um cliente para localizar, transferir e registrar novas réplicas de arquivos na Grade Computacional. Uma ferramenta em linha de comando (`globus-`

`replication-create`) também é disponibilizada ao usuário para criar e iniciar as requisições de replicação na Grade Computacional (DRS, 2005).

O OGSA-DAI é o serviço responsável pelo acesso à fonte de dados na Grade Computacional. A *Arquitetura GISE* utiliza o OGSA-DAI para realizar acesso às fontes de dados envolvidas no processo de integração.

Monitoramento e Descoberta de Serviços e Recursos

Nos sistemas distribuídos, o monitoramento e descoberta de serviços e recursos é uma função essencial. O monitoramento permite detectar e diagnosticar diversos problemas e a descoberta permite identificar recursos ou serviços necessários pelos usuários ou aplicação. Estes serviços coletam informações sobre os serviços da Grade Computacional nas fontes de informação distribuídas.

O GT4 permite associar descrições baseadas em documentos XML com os recursos da infra-estrutura computacional. Desta forma, os metadados sobre os recursos computacionais podem ser recuperados através de consultas ou atualizados através de requisições.

O MDS4 (*Monitoring and Discovery Service*) é o componente do GT4 baseado na especificação WSRF que simplifica a tarefa de monitoramento e descoberta de recursos computacionais. O monitoramento e descoberta de recursos requerem a habilidade de coletar informações sobre os recursos na Grade Computacional de múltiplas e distribuídas fontes de dados. Para isso, o MDS4 provê os serviços de registro (*Index*) e o filtro de dados (*Trigger*) que coletam informações sobre os estados dos recursos nas diferentes fontes de informação. O GT4 também disponibiliza um serviço para consulta e acesso as fontes de informação (metadados) através de *browsers* ou linhas de comando denominado WebMDS (MDS, 2005).

A Figura 9 ilustra a infra-estrutura de monitoramento e descoberta de serviços. O MDS-Index se comunica com outros MDS-Index na Grade Computacional para obter informações sobre os recursos computacionais. Por outro lado, o MDS-Index se comunica diretamente com os serviços do GT4 para buscar informações sobre o serviço. Finalmente, o WebMDS acessa o MDS-Index para realizar consultas e gerar visualizações personalizadas das informações dos serviços disponíveis na Grade Computacional (FOSTER, 2005).

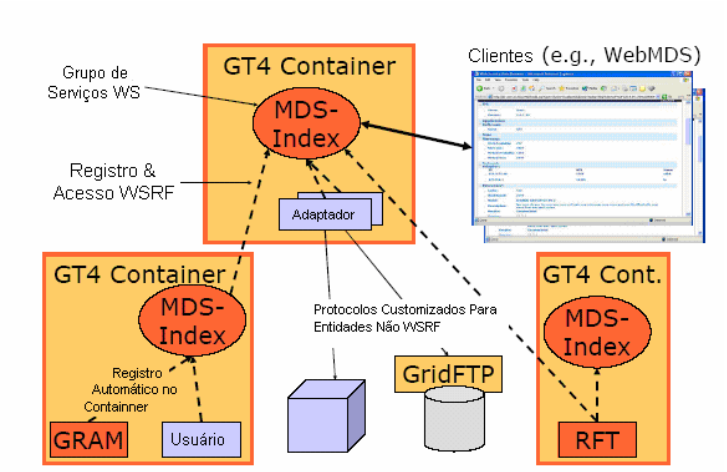


Figura 9: Infra-Estrutura de Monitoramento e Descoberta de Serviços (FOSTER, 2005).

Segurança

Quando os recursos computacionais e/ou usuários estão distribuídos em diferentes localizações, a segurança que deve ser oferecida pela infra-estrutura se torna um desafio. Disponibilizar recursos individuais para uso externo pode se tornar um problema se não forem definidas políticas claras para o acesso e uso de recursos compartilhados. A privacidade das informações transmitidas também deve ser um ponto coberto pela segurança da infra-estrutura, pois não é desejável que usuários tenham acesso a informações não autorizadas.

O GT4 possui uma infra–estrutura de segurança em nível de troca de mensagens, baseado nas credenciais X.509 e protocolos para a proteção de mensagens, autenticação, delegação e autorização de usuários.

Na Grade Computacional cada usuário recebe um certificado público X.509 (Proxy) para acesso os serviços. Este certificado é sempre validado pelo serviço que o usuário utilizar. Desta forma, o usuário terá uma identidade única no GT4 e realizará apenas um *login* para todos os recursos da Grade Computacional (FOSTER, 2005).

Atualização do Globus Toolkit 3.x para Globus Toolkit 4.x.

Os *Grid Services* implementados até as versões 3.x do *Globus Toolkit* nasceram da tecnologia de *Web Services*. Portanto, com a evolução dos *Web Services* se fez necessário à evolução da especificação OGSI. A nova especificação foi apresentada em Janeiro de 2004 e denominada como WSRF (*Web Services Resource Framework*).

Um dos pontos de falha do OGSI é o mecanismo de endereçamento GSH (*Grid Service Handle*) e o mecanismo de notificação. A nova especificação WSRF oferece um mecanismo de endereçamento e notificação compatível com a tecnologia de *Web Services* (*WS-Addressing*) e (*WS-Notification*) (CZAJKOWSKI, 2005).

A especificação WSRF possui vantagens em relação a sua antecessora. A especificação OGSI não funciona bem com ferramentas de Web Service, pois utiliza demasiadamente esquemas XML e documentos orientados a operações WSDL construídos sem padrões.

A especificação OGSI é muito voltada para orientação a objetos e modela um recurso persistente da mesma forma que um recurso transiente. Outra vantagem de WSRF em relação à OGSI é que esta especificação é muito longa, não possuindo separações claras. O WSRF resolve este problema dividindo sua especificação em cinco partes (CZAJKOWSKI, 2004), conforme descrito a seguir:

- *WS-ResourceProperties:* Descreve a associação de recursos e *Web Services* para produzir *WS-Resources*. Como as propriedades dos *WS-Resource* são publicamente visíveis, elas podem ser recuperadas, alteradas e apagadas;
- *WS-ResourceLifetime* Permite ao criador destruir imediatamente ou agendar a destruição de um *WS-Resource;*
- *WS-RenewableReferences* Anota uma referência de um *WS-Resource* com informações necessárias para gerar uma nova referência ao *WS-Resource* quando a mesma é invalidada*;*
- *WS-ServiceGroup* Cria e usa coleções heterogêneas de *Web Services;* e
- *WS-BaseFault* Descreve os erros bases utilizados para reportar erros na execução dos *WS-Resource.*

Por fim, o OGSI foi construído com base em um rascunho da versão 2.x do WSDL, sendo incompatível com o WSDL versão 1.0 (CZAJKOWSKI et al., 2005).

A comunidade de *Web Services* argumenta que um Web Service não deve manter seu estado ou ter instâncias, o que acontece com a especificação OGSI. O WSRF modifica este modelo e define diferentemente o serviço e as suas instâncias. Esta composição é denominada na especificação WSRF de *WS-Resource* (FOSTER et al., 2004).

Vale lembrar que com WSRF, as interfaces dos serviços OGSA não sofreram alterações.

A especificação WSRF está disponível de forma estável a partir da implementação da versão 4.x do *Globus Toolkit*.

2.4.2 Legion

O *Legion* é um projeto realizado pela Universidade da Virginia – EUA que começou nos anos 80 com uma linguagem de programação chamada MENTAT. *Legion* tem como objetivo a utilização de recursos distribuídos em múltiplas instituições (GRIMSHAW, 1997).

O *Legion* se diferencia de outras arquiteturas de Grades Computacionais. No ambiente *Legon,* todos os componentes são representados na forma de objetos, sejam eles recursos computacionais (Dados, Discos, CPU), dispositivos físicos (microscópios) e usuários. Estes objetos podem assumir dois estados: "Ativos" quando eles estão prontos para receberem chamados ou "Inativos" quando não estão disponíveis para uso imediato.

Todos os objetos *Legion* possuem um nome, um identificador *Legion* (LOID) e um endereço (LOA) composto pelo endereço IP do local físico do serviço e sua porta de acesso.

O projeto de *Legion* permite que ele seja utilizado em múltiplos domínios administrativos. Porém, ele não impõe política de uso de recursos aos usuários. Quem tem autonomia no *Legion* é o administrador local. Ele é responsável por especificar quais recursos quer compartilhar e como eles serão compartilhados aos usuários da Grade Computacional.

Como características principais, o *Legion* é independente de plataforma, permite o uso de múltiplas linguagens de programação, possui suporte para o aproveitamento de aplicações legadas, gerenciamento de falhas e fornece componentes para segurança e autenticação.

O núcleo de *Legion* é extensível e permite que partes deste possam ser substituídas ao aprimoradas. O *Legion* possui implementação padrão para os seguintes tipos de objetos:

Objetos de Contexto:	Mapeiam nomes para LOID´s;
Objetos de Associação:	Registro entre pares LOID, LOA;
Objeto Hospedeiro:	Representam um processador e podem receber pedidos;
Objeto Cofre:	Representam uma área de disco; e
Objetos de Implementação:	Representam um programa que será executado.

Em *Legion* cada objeto tem suas próprias políticas de acesso e indicam quais usuários poderão fazer chamadas aos seus métodos ou um conjunto deles. Quando um usuário tem acesso autorizado a um objeto, ele recebe um certificado criptografado indicando suas permissões de acesso.

2.4.3 Condor

O *Condor* é um projeto iniciado de 1985 como a evolução do Projeto Remote Unix. O *Condor* assim como o *Legion* e o *Globus* tem por objetivo gerenciar recursos ociosos na Grade Computacional. *Condor* pode oferecer grande poder de processamento para o usuário (LITZKOW, 1988).

A arquitetura *Condor* é dividida em seis camadas distintas, permitindo grande flexibilidade no uso. O *Condor* pode ser utilizado em um computador isolado ou no trabalho cooperativo com outras plataformas de Grade Computacional. A arquitetura *Condor* está especificada na Figura 10.

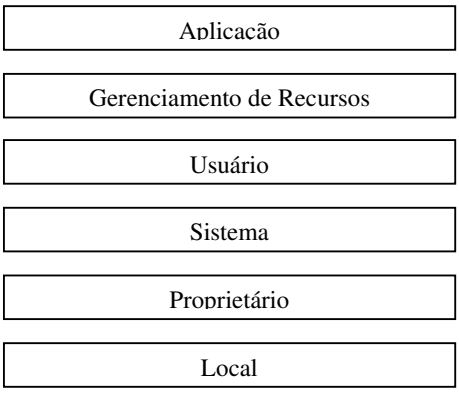


Figura 10: Arquitetura Condor (LITZKOW, 1988).

- A primeira camada é denominada de camada local. Ela gerencia os recursos locais, como dispositivos de armazenamento e processamento;

- A segunda é a camada do proprietário. Esta camada disponibiliza ao usuário os componentes para que possam ser especificadas as restrições do usuário e sua disponibilidade de recursos;
- A terceira é a camada do sistema. Esta camada tem como responsabilidade realizar transações entre os recursos que são disponibilizados e a necessidade de serviços dos usuários;
- A quarta é a camada do usuário. Esta representa suas necessidades, onde suas requisições são colocadas em filas;
- A quinta camada é responsável pelo gerenciamento de recursos da aplicação. Esta camada administra os recursos obtidos pelos escalonadores para a execução de tarefas na Grade Computacional; e
- Por fim, a última camada é a camada de aplicação, onde estão localizados os componentes de software que serão executados no ambiente *Condor*.

Um dos recursos mais importantes do *Condor* é a sua estrutura de *Checkpoints*. O *Checkpoint* é um recurso capaz de identificar o acontecimento de determinados eventos, seja eles temporais ou não e automaticamente gravar as informações referentes ao processamento em execução para um dispositivo não volátil. Este recurso permite que longos processamentos não sejam perdidos em uma falha de hardware ou software e que o processamento seja retomado o mais próximo possível do ponto de interrupção.

O recurso de *Checkpoint* é de muita importância em um ambiente distribuído, pois não é possível controlar a disponibilidade de recursos uma vez que a conexão de rede ou um computador pode vir a falhar ou não estar mais disponível pelas necessidades do usuário proprietário do recurso.

O *Condor* possui em sua estrutura um serviço que permite detectar a indisponibilidade de um recurso computacional e transferir suas atividades para outros computadores sem a necessidade de reiniciar o trabalho que já foi executado.

A execução do *Condor* pode acontecer em quatro perfis diferentes, conforme descrito a seguir:

- *Condor* pode ser executado isoladamente em um único computador pessoal, aproveitando–se do recurso de *Checkpoint.* Desta forma, é disponibilizado ao usuário um mecanismo bastante eficiente de tolerância à falhas;
- *Condor* pode ser executado em um conjunto de computadores dentro do mesmo ambiente administrativo para a execução das tarefas de compartilhamento de recursos computacionais;
- *Condor* pode ser executado interligando diversas outras redes *Condor*. Assim são ultrapassadas barreiras administrativas e geradas oportunidades de negócio e pesquisa; e
- *Condor* pode ser executado simultaneamente com o *Globus*. Desta forma, podemos usufruir dos serviços que as duas arquiteturas têm de melhor. Quando o *Condor* é utilizado em conjunto com o *Globus* o denominamos de *Condor-G*.

Capítulo 3-Integração de Fontes de Dados

Neste Capítulo são abordados aspectos referentes à integração de fontes de dados na Grade Computacional. Conforme pesquisado na literatura, a integração de fontes de dados pode ocorrer de diferentes maneiras, utilizando diversas metodologias e arquiteturas. A seção 3.1 faz uma abordagem sobre a visão de integração de fontes de dados e discute os conflitos encontrados no processo de integração de fontes de dados e o Modelo de Dados Canônico (MDC). A seção 3.2 discute o acesso à fonte de dados na Grade Computacional. A seção 3.3 expõe alguns tipos de arquiteturas para a integração de fontes de dados e finalmente na seção 3.4 são discutidas alguns exemplos de arquiteturas propostas na literatura para a integração de fontes de dados.

3.1. Visão Sobre a Integração de Fontes de Dados

A introdução da Sociedade da Informação culmina na necessidade de um completo acesso e processamento de informações. Neste cenário, encontram-se organizações que são fisicamente distribuídas, possuem divisões internas (setores) e em muitos casos, uma organização pode ser composta por um conjunto de outras organizações. Desta forma, a descentralização de informações é uma característica prevista e inevitável. A descentralização impõe desafios computacionais para um acesso transparente e integrado a diversas fontes de dados, tais como: distribuição dos dados, heterogeneidade das tecnologias computacionais, utilização de distintas metodologias de representação e freqüência de representações díspares das fontes de dados análogas.

O acesso integrado e transparente as fontes de dados é vital para o processo de tomada de decisões corporativas globais. As organizações que possuem aplicações capazes de executar transações globais em suas fontes de dados possuem um diferencial em relação aos

seus concorrentes. Porém, a maioria das aplicações falha por não oferecer suporte para o acesso integrado e transparente as fontes de dados (ATKINSON et al., 2004).

A integração de fontes de dados já vem sendo estudada por alguns anos e sua aplicação na Grade Computacional introduziu novos desafios, como o aumento de heterogeneidade, escalabilidade, distribuição de dados, etc (RAMAN et al., 2003). Na Grade Computacional a disponibilidade de acesso a fontes de dados heterogêneas e recursos computacionais, a facilidade de romper limites administrativos, fronteiras geográficas e suporte a escalabilidade torna a mesma um ambiente atrativo para o processo de integração de fontes de dados. Porém, para a realização da integração de fontes de dados é desejável o suporte da infra–estrutura computacional para disponibilizar: (RAMAN et al., 2003):

- **Transparências para o Acesso de Dados:**

  Heterogeneidade: As aplicações devem acessar diversas fontes de dados em múltiplos domínios administrativos e por sua vez, os dados podem estar armazenados de diferentes formas, como por exemplo: dados estruturados (tabelas), dados semi-estruturados (arquivos XML) ou dados sem estruturação (páginas web, textos, etc);

  Nomes: Para o acesso aos recursos das fontes de dados na Grade Computacional não é necessário que o usuário conheça exatamente quais são os nomes das fontes de dados de seu interesse. O acesso pode ser feito de uma forma semântica, onde o usuário informa os atributos que descrevam suas necessidades e a infra–estrutura computacional se encarrega de lhe fornecer o objeto desejado;

  Autoria e Custo: As aplicações devem estar preparadas para negociarem o acesso das fontes de dados localizadas em diferentes domínios, respeitando características como: propriedade dos dados, autenticação, políticas locais de segurança e custos de acesso e uso das fontes de dados;

- **Transparências para Processamento de Dados:**

  Paralelismo: Os benefícios da computação paralela oferecida na Grade Computacional devem ser explorados, permitindo desempenho escalável na manipulação e consulta de grandes fontes de dados; e

  Distribuição: Os dados podem estar distribuídos de maneira uniforme, não importando ao usuário em que local os dados estão armazenados. É necessário que os dados sejam visualizados e manipulados de maneira unificada, preservando sua integridade e consistência.

As tecnologias atuais para armazenamento e gerenciamento de dados possuem diversas limitações para a aplicação na Grade Computacional. Pode-se destacar como exemplo que as linguagens utilizadas para acesso as fontes de dados que especifiquem apenas o dado a ser mapeado, mas não como a consulta deve ser executada (ATKINSON et al., 2004).

Atualmente existem muitas interfaces para o acesso a fontes de dados, tais como: JDBC e OBDC. Ambas as interfaces suportam uma variedade de operações, como inserções, alterações e consultas. Elas provêm transparência de heterogeneidade, mas fornecem suporte a poucos tipos de fontes de dados (ATKINSON et al., 2004).

O processo de integração de fontes de dados é uma tarefa complexa que ocasiona diversos conflitos decorrentes da utilização de modelos díspares. Na próxima seção são identificados os conflitos decorrentes do processo de integração de fontes de dados.

3.1.1 Conflitos no Processo de Integração de Fontes de Dados

Integrar dados é tarefa complexa e na Grade Computacional este problema é maximizado pelas características de distribuição e heterogeneidade da infra-estrutura. A

seguir, estão distribuídos em três grupos distintos os conflitos identificados no processo de integração de fontes de dados (BOTELHO, 2004):

- **Conflitos Semânticos:**

- *Conflitos de Denominação:* Ocorrem na nomenclatura dos objetos, podendo ser sinônimos quando os atributos de objetos idênticos são denominados de formas diferentes ou homônimos, quando diferentes atributos são denominados de forma idêntica. Este conflito é resolvido gerando–se esquemas locais específicos das diferentes fontes de dados envolvidas no processo de integração e esquemas integrados que são utilizados para representar a integração de dois ou mais esquemas locais.

- **Conflitos Estruturais:**

- *Conflitos de Tipo:* Este tipo de conflito acontece quando elementos a serem integrados em diferentes estruturas de dados possuem definições de tipo incompatíveis. Este conflito é tratado armazenando informações pertinentes ao tipo do atributo permitido. Com estas informações, é realizado o tratamento adequado nos processos de integração de fontes de dados;

- *Conflitos de Formato:* Acontecem quando os atributos integrados são representados em diferentes formatos em seus esquemas locais. Este conflito é tratado semelhantemente ao conflito de tipo. É realizado o armazenamento de informações referentes ao formato das propriedades dos objetos, permitindo tratamento destas informações no processo de integração de fontes de dados;

- *Conflitos de Unidade:* Acontecem quando os atributos dos objetos são representados em unidades diferentes. Este conflito é tratado armazenando as informações referentes ao tipo de unidade nos esquemas locais da fonte de dados e uma função de conversão;

- *Conflitos de Restrição:* Acontecem quando um determinado atributo possui um conjunto limitado de valores que podem ser assumidos;

- *Conflitos de Código:* Acontecem quando os valores assumidos pelos identificadores únicos se sobrepõem no processo de integração de fontes de dados; e

- *Conflitos entre Chaves:* Acontecem quando os esquemas locais possuem diferentes campos chave.

- **Conflitos entre Valores:**

- *Conflitos de Valores Default:* Em muitos casos as aplicações ou até mesmo os sistemas gerenciadores de banco de dados atribui um valor padrão para uma determinada situação nos atributos das fontes de dados. Estes valores podem ser conflitantes para propriedades semanticamente iguais; e

- *Conflitos de Atualização:* Acontecem quando existem mudanças nas propriedades dos objetos envolvidos no processo de integração.

Este trabalho está limitado no tratamento de conflitos de denominação, tipo, formato e unidade, sendo os outros conflitos identificados objetos de pesquisa futura.

Para resolver os conflitos identificados no processo de integração de fontes de dados na Grade Computacional e permitir o acesso transparente às fontes de dados integradas é utilizado um Modelo de Dados Canônico (MDC). O Modelo de Dados Canônico, suas características e funcionalidades são discutidas na próxima seção.

3.1.2 Modelo de Dados Canônico

A integração de fontes de dados é um processo complexo e uma de suas diversas etapas é o mapeamento de fontes de dados heterogêneas para um Modelo de Dados Canônico (BOTELHO, 2004).

A modelagem de bancos de dados é uma metodologia capaz de abstrair uma realidade e representa-la através de Sistemas Gerenciadores de Banco de Dados (MACFARLANE et al., 1996). Porém, os responsáveis pela construção destas estruturas de dados criam seus

modelos de forma heterogênea através de seu próprio conhecimento sobre o domínio modelado, não seguindo nenhuma regra ou especificação previamente definida, gerando consequentemente em muitos casos, representações em diferentes formas semânticas.

O MDC é um modelo para o qual todas as fontes de dados envolvidas no processo de integração são transformadas, ocultando as diferenças sintáticas geradas no processo da modelagem de dados. (BOTELHO, 2004)

A criação de um MDC é um fator crítico. A utilização de um MDC semanticamente rico facilita a descoberta de similaridades entre as fontes de dados envolvidas no processo de integração. Porém, a utilização de um MDC que não expresse corretamente o modelo conceitual das fontes de dados utilizadas no processo de integração não proporciona resultados válidos.

O MDC também permite que os usuários realizem operações de consultas e manipulações de dados de forma transparente em múltiplas fontes de dados.

Existem algumas pesquisas na literatura que descrevem quais são os requisitos que um MDC deve conter (VENTRONE & HEILER, 1991). Nestes atributos, destacam-se:

- Expressar através de formas simples, semânticas geradas em diferentes níveis de abstração;
- Possibilidade de acrescentar metadados não armazenados nas fontes de dados;
- Representar facilmente diferenças entre os diversos esquemas; e
- Disponibilizar as informações semânticas a todos os usuários da infra-estrutura distribuída.

3.2. Acesso a Banco de Dados na Grade Computacional (OGSA-DAI)

O *Open Grid Service Architecture Data Access and Integration* (OGSA-DAI) é um projeto iniciado pelo *UK Database Task Force* e atualmente é de responsabilidade do grupo *Database Access and Integration Services Working Group* (DAIS-WG) do Global Grid Fórum (GGF). O OGSA-DAI é um projeto baseado em um consórcio de grandes empresas, como a EPCC, IBM e a Oracle. O OGSA-DAI tem como objetivo criar um conjunto de componentes padrões de código aberto compatíveis com a especificação OGSA para o acesso a fontes de dados na Grade Computacional (ATKINSON, 2003).

O OGSA-DAI é um *middleware* que permite que fontes de dados, possam ser acessadas na Grade Computacional. O OGSA-DAI provê as funcionalidades básicas requeridas para compor serviços de alto nível para integração de fontes de dados. Os serviços disponibilizados pelo OGSA-DAI podem ser utilizados como premissas básicas para oferecer suporte no processamento distribuído de consultas, provendo transparência de heterogeneidade (OGSA-DAI-WSRF, 2005).

As fontes de dados são representadas no OGSA-DAI como um *Grid Service* disponível para os usuários na Grade Computacional. A estrutura do OGSA-DAI é extensível, permitindo o acesso e atualização das fontes de dados através da exploração de operações de manipulação de dados na Grade Computacional, tais como: *inserts, updates* e *selects*. As fontes de dados heterogêneas são acessadas de maneira uniforme, disponibilizando ao usuário a entrega de dados em um arquivo XML.

As especificações mais conhecidas para a construção de Grades Computacionais são suportadas pelo OGSA-DAI, tais como: o OGSI (*Open Grid Service Infrastructure*) - disponível no *Globus* 3.x (GT3) e o WSRF (*Web Services Resource Framework*) - disponível no *Globus* 4.x (GT4).

O OGSA-DAI possui em sua implementação um conjunto de serviços responsáveis pela interação entre usuário e a fonte de dados. O usuário faz acesso ao serviço e este se responsabiliza pelo acesso ao sistema de armazenamento que a fonte de dados reside. O OGSA-DAI fornece acesso às fontes de dados na Grade Computacional através do *Data Service Resource* (ATKINSON et al., 2005).

O OGSA-DAI é a arquitetura de acesso e comunicação às fontes de dados utilizadas na arquitetura proposta por este trabalho. Porém, vale lembrar que o OGSA-DAI não resolve sozinho o problema de integração de fontes de dados. O OGSA-DAI não foi implementado com a capacidade de receber uma requisição integrada que acesse múltiplas fontes de dados simultaneamente na Grade Computacional.

Para que o OGSA-DAI seja capaz de acessar as fontes de dados, as requisições devem estar no modelo natural da fonte de dados.

O OGSA-DAI possui em seu conjunto de funcionalidades um serviço de processador de consultas distribuídas denominado OGSA-DQP (*Open Grid Service Architecture – Distributed Query Processor*). O OGSA-DQP é uma arquitetura para a execução de consultas distribuídas na Grade Computacional.

A arquitetura do OGSA-DQP utiliza como base os serviços oferecidos pelo OGSA-DAI para acessar de forma transparente as fontes de dados heterogêneas na Grade Computacional. O OGSA-DQP também se beneficia das vantagens de processamento paralelo, alocando seus serviços de consulta em diversos nós da Grade Computacional (ALPDEMIR et al., 2003).

A arquitetura do OGSA-DQP é formada por dois componentes: o GDQS (*Grid Distributed Query Service*) e o GQES (*Grid Query Evaluation Service*). O GDQS é responsável por receber as requisições de consulta do usuário e processa-las na Grade Computacional. O GQES é o serviço que recebe as sub-consultas do GDQS e acessa as fontes

de dados. As instâncias, gerenciamento e alocação dos GQES entre os nós da Grade Computacional são de responsabilidade do GDQS (ALPDEMIR et al., 2003).

Quando uma consulta é enviada ao OGSA-DQP, é criado pelo GDQS um plano de execução. Neste plano de execução são definidas quantas instâncias GQES serão necessárias para o processamento da consulta e com base nas informações dos escalonadores, definem-se quais serão os nós da Grade Computacional que serão responsáveis pelo processamento das consultas dos usuários (ALPDEMIR et al., 2003).

A arquitetura do OGSA-DQP por si só também não é capaz de resolver o problema de integração de fontes de dados. Devemos lembrar que no OGSA-DQP, nenhuma integração de esquemas e resolução de conflitos é permitida. Caso o usuário necessite realizar uma consulta integrada deverá implementar seus próprios mecanismos, preocupando–se com o tratamento de conflitos semânticos e estruturais decorrentes do processo de integração de fontes de dados.

O OGSA-DQP possui uma limitação no seu mecanismo de consulta distribuída. O plano de processamento gerado pelo GDQS é estático e não pode sofrer nenhuma alteração em tempo de execução. Desta forma, se por algum motivo o computador responsável por executar a consulta ou o computador que armazena a fonte de dados vier a falhar, o OGSA-DQP ficará impossibilitado de finalizar sua requisição e retornará um erro ao usuário requisitante.

O OGSA-DAI e o OGSA-DQP são serviços disponibilizados na Grade Computacional para o acesso e consulta distribuída a fontes de dados, respectivamente. Neste estudo, o OGSA-DQP não está sendo utilizado. Porém, a utilização deste componente pode estar condicionada a objeto de pesquisa futura visando à otimização dos Serviços de Integração de Fontes de Dados disponibilizadas pela arquitetura proposta.

A seguir são discutidos alguns tipos de arquiteturas utilizadas para a integração de fontes de dados.

3.3. Tipos de Arquiteturas Para Integração de Fontes de Dados

Há algum tempo, a integração de fontes de dados já vem sendo estudada. Com o avanço dos estudos, a forma que os dados são integrados passou por diversas variações e produziu diferentes tipos de arquiteturas. Nesta seção serão abordadas as características das arquiteturas de Bancos de Dados Múltiplos, Federação de Bancos de Dados e Mediadores.

3.3.1 Banco de Dados Múltiplos

A arquitetura de banco de dados múltiplos ou *multidatabase* é uma coleção de diversos bancos de dados. Nos Bancos de Dados Múltiplos, não existe o conceito de Esquema Integrado que representa a unificação dos esquemas globais. Para possibilitar o acesso integrado, é disponibilizada uma linguagem que oferece construtores para executar operações envolvendo vários bancos de dados. Nesta arquitetura, os sistemas gerenciadores de banco de dados possuem autonomia local e desconhecem outras fontes de dados disponíveis para a integração de dados (HURSON et al., 1994).

A Figura 11 ilustra a arquitetura de Bancos de Dados Múltiplos. Esta arquitetura é composta pelos seguintes componentes:

- Bancos de Dados Locais;
- Esquemas Locais;
- Esquemas Conceituais representados em um mesmo modelo de dados que define a parte do Esquema Local acessível para a integração;
- Esquema de Dependências que definem a dependência e restrições de integridade globais envolvendo os bancos de dados;

- Esquema Externo (que representa uma visão externa, envolvendo dados de um ou mais esquemas conceituais);
- Aplicações (realizam consultas através dos esquemas externos);

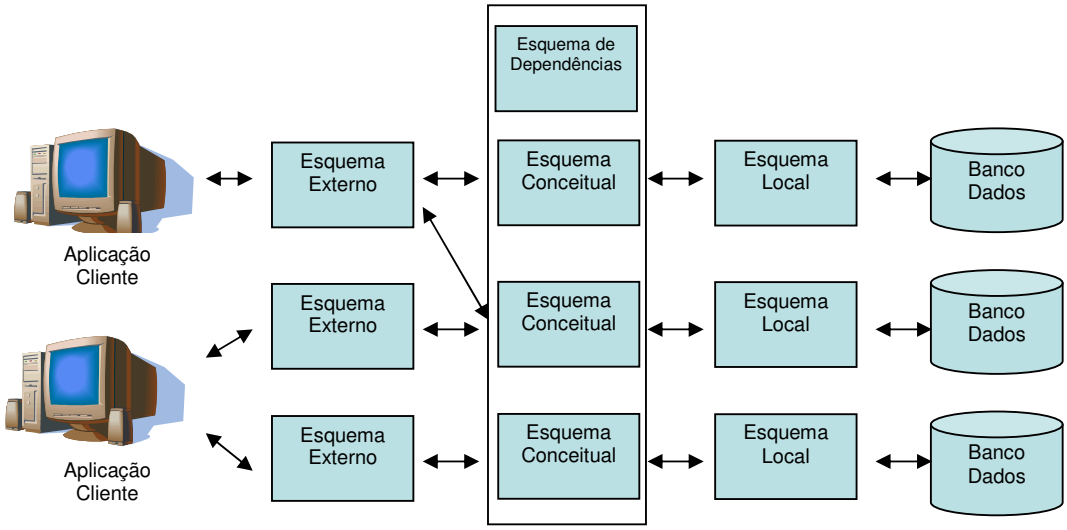


Figura 11: Banco de Dados Múltiplos.

Esta arquitetura possui inúmeras desvantagens quando aplicada a Grade Computacional, conforme descritas a seguir (RAMAN, 2003):

- Os Bancos de Dados Múltiplos não fornecem transparência de localização, precisando o usuário saber antecipadamente a localização exata das fontes de dados;
- Os Bancos de Dados Múltiplos possuem baixa escalabilidade, onde quaisquer mudanças nos componentes participantes da integração ocasionam a geração de novos esquemas locais, esquemas conceituais, esquemas de dependências e esquemas externos; e

- Nesta arquitetura não é possível proporcionar a integração automática dos dados. Portanto, existe a necessidade de um usuário que conheça a "priori" a semântica existente nas fontes de dados para executar o processo de integração de dados.

A integração de fontes de dados pela arquitetura de bancos de dados múltiplos não é recomendada para ambientes altamente dinâmicos, distribuídos e escalares, como acontece na Grade Computacional.

3.3.2 Federação de Banco de Dados

A Federação de Banco de Dados é uma arquitetura que provê a coleção de sistemas de bancos de dados autônomos, permitindo o compartilhamento controlado e parcial de seus dados. Com a Federação de Banco de Dados o usuário possui uma visão única de todas as fontes de dados que integram um esquema federado, disponibilizando ao mesmo, acessos e manipulações simultâneas em diversas fontes de dados (HURSON et al., 1994; ALPDEMIR et al., 2003).

A Figura 12 ilustra a arquitetura de Federação de Bancos de Dados. Esta arquitetura é composta pelos seguintes componentes:

- Bancos de Dados Locais;
- Esquemas dos Bancos de Dados Locais;
- Esquemas Componentes com representação do banco de dados em um modelo canônico;
- Esquemas de Exportação que permite definir parte do esquema componente disponível para acesso;
- Esquema Federado que viabiliza uma visão unificada de todos os esquemas de exportação disponíveis;

- Esquema Externo que abstrai o Esquema Federado quando ele é muito complexo; e
- Aplicações que acessam os bancos de dados da federação.

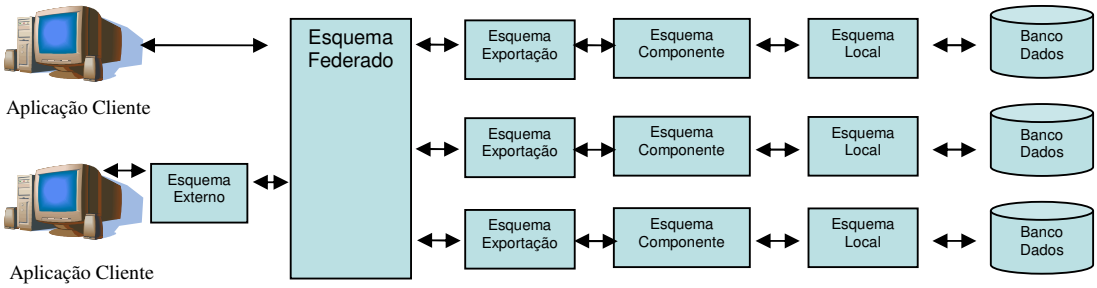


Figura 12: Banco de Dados Federado.

Uma das principais vantagens da Federação de Banco de Dados é a cooperação de sistemas independentes (BOTELHO, 2004). Nesta arquitetura, a integração das fontes de dados pode ser realizada apenas em conjuntos específicos de dados, diferentemente com o que ocorre na arquitetura de Bancos de Dados Múltiplos (PINTO, 2002).

A arquitetura de Bancos de Dados Federados resolve os problemas de transparência de heterogeneidade e distribuição de dados, sendo esta arquitetura mais eficaz no processo de integração de fontes de dados do que a arquitetura de Bancos de Dados Múltiplos.

Porém, a Federação de Banco de Dados possuem limitações quando aplicadas na Grade Computacional. O principal problema é na geração e manutenção dos esquemas federados, vistos que o número de fontes de dados envolvidas no ambiente pode ser muito grande.

3.3.3 Mediadores

A arquitetura de Mediadores permite a virtualização de fontes de dados heterogêneas e distribuídas. Os Mediadores são componentes de software capazes de encapsular a representação de múltiplas fontes de dados, explorarem seu conhecimento e produzir informações para outros componentes da arquitetura (BOTELHO, 2004).

Através do uso de Mediadores podem-se gerar modelos para a representação de fontes de dados de maneira uniforme e extensível, permitindo o acesso e manipulação transparente às múltiplas fontes de dados.

A utilização de uma arquitetura baseada em Mediadores ocasiona na utilização de um modelo de dados canônico. Todas as fontes de dados envolvidas no processo de integração são mapeadas de acordo com o Modelo de Dados Canônico para resolver os conflitos de denominação em diferentes modelos de dados (OZSU & VALDURIEZ, 1999).

Na arquitetura de Mediadores o acesso à fonte de dados é realizado através de um Tradutor. Porém, nesta arquitetura, as requisições de manipulação de dados são de responsabilidade do Mediador. Este é responsável por obter os resultados solicitados, comunicando-se diretamente com o Tradutor da fonte de dados acessada.

A arquitetura de Mediadores pode ser empregada através de duas formas distintas. Desta forma, a fonte de dados integrada pode ser virtual ou materializada.

Na abordagem de fonte de dados virtual, os dados permanecem armazenados nas fontes de dados de origem. Quando a consulta do usuário ou aplicação é recebida pelo Mediador, o mesmo fragmenta em quantas sub–consultas forem necessárias (divididas por fontes de dados) e as repassa para os Tradutores específicos que realizam o acesso local a fonte de dados. Depois de receber os diversos resultados das sub-consultas enviadas, os dados são desfragmentados, processados e enviados para o usuário.

Na abordagem materializada, inicialmente todos os dados são extraídos dos repositórios de informação pelos Tradutores e unificados pelo Mediador com base nas informações dos esquemas integrados em um repositório único. Por fim, a consulta integrada é processada (sem decomposições) na fonte de dados integrada e os resultados são retornados ao usuário (BOTELHO, 2004).

A Figura 13 ilustra a arquitetura de Mediadores. Esta arquitetura é composta pelos seguintes componentes:

- Aplicação Cliente
- Mediadores
- Tradutores
- Fontes de Dados

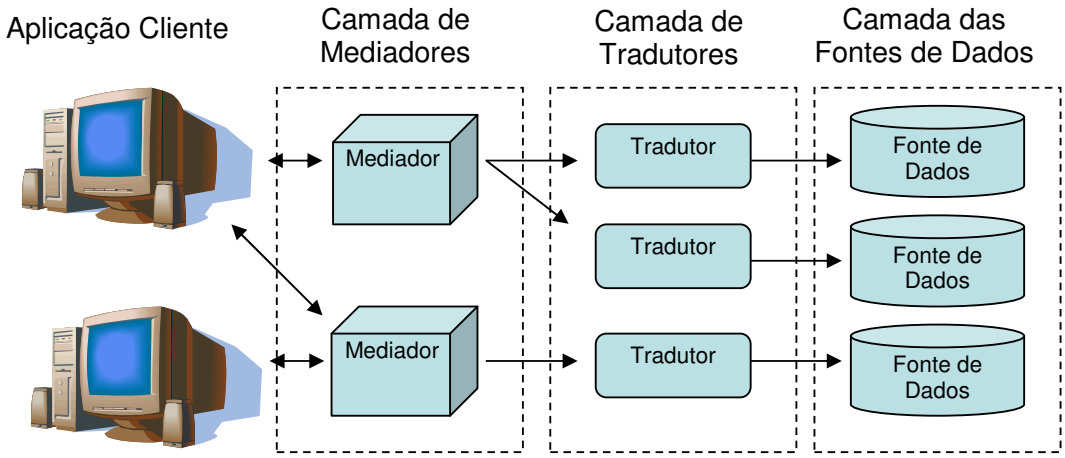


Figura 13: Arquitetura de Mediadores.

A arquitetura de Mediadores apresenta diversas vantagens em comparação às outras arquiteturas citadas neste trabalho. A arquitetura de Mediadores provê através do uso de Tradutores transparência de heterogeneidade. A transparência de nomes é realizada através da utilização de um modelo de dados canônico. À aplicabilidade desta arquitetura na Grade

Computacional, também pode destacar diversas vantagens em relação aos outros tipos de arquiteturas que foram discutidas neste trabalho, conforme segue (OZSU & VALDURIEZ, 1999):

- Os Mediadores são capazes de se especializar em conjuntos específicos de informações. Desta forma, podem ser definidos diferentes Mediadores para diferentes domínios; e
- Na arquitetura de Mediadores é permitido que se tenham componentes especializados no tratamento apropriado a diferentes requisitos encontrados no processo de integração de fontes de dados.

Devido às características e as vantagens identificadas, a implementação da arquitetura proposta neste trabalho utiliza a arquitetura de Mediadores. Desta forma, este trabalho propõe uma arquitetura de integração de fontes de dados extensível na Grade Computacional.

3.4 Exemplos de Arquiteturas de Mediação para a Integração de Fontes de Dados

A seguir serão discutidas algumas arquiteturas encontradas na literatura para a integração de fontes de dados. A discussão é composta pelas arquiteturas Garlic, MIX, Mocha, WISE e GDIS que ao final da seção são comparadas.

3.4.1 Garlic

O Garlic é um sistema motivado pela resolução dos problemas de acesso em fontes de dados diversificadas. Foi um projeto realizado pela IBM e utiliza como arquitetura de integração de fontes de dados a arquitetura de Mediadores. No Garlic é utilizado um modelo de dados canônico orientado a objetos e uma interface de programação que obedece a padrões estabelecidos pelo ODMG (*Object Database Management Group*). No Garlic também é disponibilizado interfaces para execução de consultas (LUNIEWSKI et al., 1995).

As consultas em Garlic são negociadas dinamicamente com o Tradutor e o Mediador do sistema. Neste momento são definidos quais serão as atividades do Tradutor no processamento das consultas. A responsabilidade da criação de planos de consulta no Garlic é do processador de consultas que também acessa diversos repositórios de informações para disponibilizar a decomposição de consultas. Depois de gerado o plano de consultas, o mesmo é transferido para os Tradutores da arquitetura e estes se responsabilizam por controlar sua execução (LUNIEWSKI et al., 1995).

A vantagem de Garlic é que ele resolve o problema de heterogeneidade encontrado em múltiplas fontes de dados. A desvantagem é que seu modelo de dados canônico não é extensível e, consequentemente, não é recomendado para aplicação em ambientes altamente dinâmicos.

3.4.2 MIX

O MIX (*Mediation of Information Using XML*) é um projeto entre o *UCDS Database Laboratory* e o DICE (*Data-Intensive Computing Enviroments*) que utiliza a arquitetura de Mediadores e um modelo de dados canônico flexível e uniforme escrito através da linguagem XML. A motivação do Projeto MIX é desenvolver sistemas de consulta sobre fontes de dados heterogêneas e distribuídas, tais como: fontes de dados estruturadas, páginas web, sistemas legados, etc (BARU et al., 1999).

Para permitir a integração das fontes de dados em MIX, é necessário que as fontes de dados possuam uma visão lógica em XML.

No MIX, os resultados de quaisquer consultas são fornecidos em um documento XML. O Mediador do sistema é o componente responsável por receber as consultas e decompô-las em sub-consultas. As sub-consultas são geradas com base em informações da visão lógica das fontes de dados armazenadas nos repositórios do sistema. O Mediador do

MIX também é responsável por receber os resultados das sub-consultas e disponibiliza-las aos usuários como um único resultado (BARU et al., 1999).

Na arquitetura MIX os Mediadores geram as sub-consultas em uma linguagem proprietária denominada XMAS (LUDASCHER et al., 1999). Os Tradutores nesta arquitetura recebem as requisições em XMAS e as transformam em uma linguagem de consulta que possa ser entendida pela fonte de dados. Os Tradutores também executam a função de transformar os resultados obtidos pelas fontes de dados em documentos XML.

Devido a MIX utilizar uma linguagem XML para a construção do seu modelo de dados canônico, ele se torna um sistema extensível de integração de fontes de dados. Porém, o MIX é uma arquitetura que não foi construída para ser utilizada na Grade Computacional.

3.4.3 MOCHA

O MOCHA (*Middleware Based On a Code SHipping Architecture*) é um middleware que tem como objetivo interconectar fontes de dados distribuídas em uma grande rede (RODRIGUEZ-MARTINEZ & ROUSSOPOULOS, 2000).

A arquitetura MOCHA é composta por quatro componentes. Um CP (*Client Application*) é utilizado para enviar consultas para processamento. O QPC (*Query Processor Coordinator*) é responsável pelos serviços de análise, validação, otimização, planejamento de decomposição, execução de consultas e gerenciamento de erros. O DAP (*Data Access Provider*) é responsável pelo acesso e tradução às fontes de dados e execução de consultas e finalmente o *Data Server* é a entidade que armazena as fontes de dados, conforme ilustrado na Figura 14 (RODRIGUEZ-MARTINEZ & ROUSSOPOULOS, 2000).

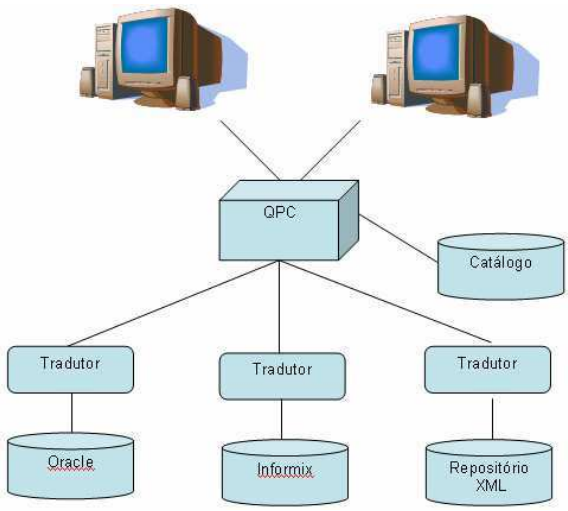


Figura 14: Arquitetura MOCHA 14 (RODRIGUEZ-MARTINEZ & ROUSSOPOULOS, 2000).

Na arquitetura MOCHA os DAP são Tradutores avançados, pois possuem a característica de execução de consultas. Mas não são capazes de executar o processamento paralelo de consultas como acontece no OGSA-DAI. Contudo, MOCHA tem a mesma limitação que o OGSA-DQP, pois também possui um plano de execução de consultas estático, onde os DAP são instalados diretamente nos *Data Server* ou em algum nó próximo ao *Data Server* na rede, não estando previsto na arquitetura caso seja necessário, a realocação do DAP em tempo de execução (RODRIGUEZ-MARTINEZ & ROUSSOPOULOS, 2000).

3.4.4 Web data Integration SystEm (WISE)

O Web data Integration SystEm *(WISE)* é um sistema desenvolvido em 2004 na Universidade Federal do Rio de Janeiro *(UFRJ)* que tem como objetivo a integração semântica e compartilhamento de dados convencionais e georeferrenciados na Internet (BOTELHO, 2004).

O WISE utiliza um MDC chamado XML Schema Semântico desenvolvido através das tecnologias XML, GML e um conjunto de Elementos Informativos. O XML Schema Semântico mapeia o conjunto dos esquemas das fontes de dados locais heterogêneas para um modelo de dados canônico facilitando o processo de integração de fontes de dados. O MDC do WISE está ilustrado na Figura 15 (BOTELHO, 2004).

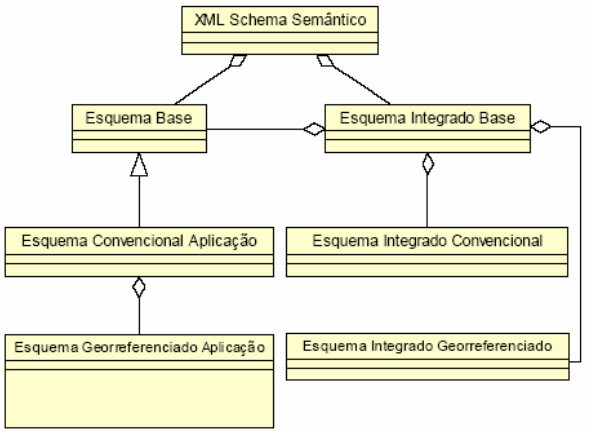


Figura 15. Modelo de Dados Canônico do WISE (BOTELHO, 2004).

A arquitetura WISE possui diversos componentes que atuam em suas quatro camadas: *Camada Cliente*, *Camada de Integração*, *Camada de Tradutores* e *Camada de Acesso a Dados.* Os componentes destas camadas são flexíveis e suportam diferentes formatos de dados de saída, bem como diversos tipos de fontes de dados heterogêneas (BOTELHO, 2004). A Figura 16 ilustra a arquitetura de camadas e o conjunto de componentes do WISE.

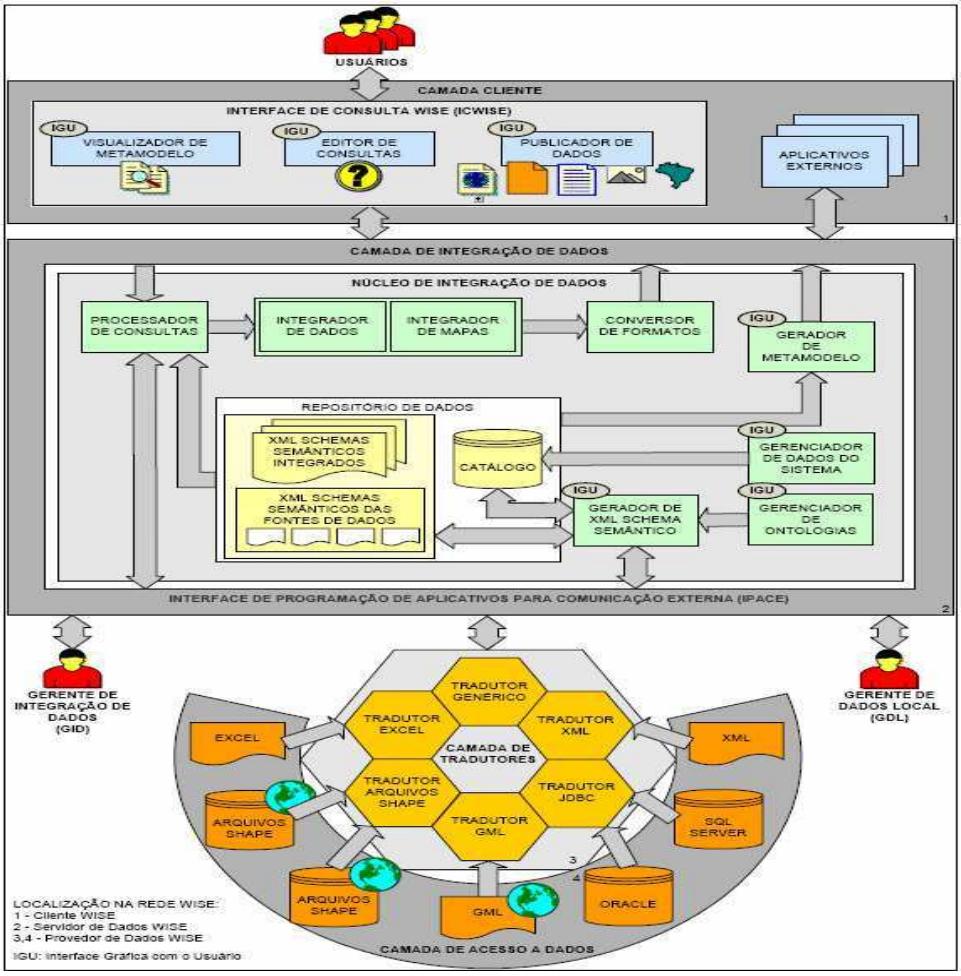


Figura 16: Camadas e Componentes da Arquitetura WISE (BOTELHO, 2004).

A arquitetura WISE é capaz de acessar através de seus Tradutores fontes de dados convencionais e georeferenciadas em diversos provedores de dados. O WISE também possui uma ferramenta automática para extrair o conteúdo de um esquema da fonte de dados local, bem como ferramentas semi-automáticas para a edição e integração de esquemas criados a partir dos esquemas locais. O Gerador de XML Schema Semântico realiza a integração de esquemas convencionais e georeferrenciados ou a combinação de ambos com o auxílio de ontologias do domínio. (BOTELHO, 2004).

Na arquitetura WISE podem existir servidores distribuídos, formando desta forma a rede WISE, aumentando seu poder de compartilhamento de integração de dados convencionais e georeferrenciados entre diversas instituições (BOTELHO, 2004).

A arquitetura WISE foi construída para ser utilizada na Internet e foi utilizada como arquitetura base para a construção da arquitetura proposta neste trabalho.

3.4.5 Grid Data Integration System (GDIS)

O GDIS é um projeto desenvolvido na Universidade da Calábria que tem como objetivo a integração de fontes de dados na Grade Computacional. O GDIS explora como *middleware* para a construção e execução de seus serviços o OGSA, OGSA-DAI e OGSA-DQP. O GDIS utiliza o *Globus* 3.x (GT3) como infra-estrutura para gerenciamento da Grade Computacional (COMITO et al., 2005).

O GDIS foi projetado para se utilizar dos benefícios da arquitetura ponto - a ponto *(P2P)* através de uma abordagem descentralizada, permitindo que consultas distribuídas possam ser executadas nas fontes de dados localizadas em diferentes *sites* (COMITO et al., 2005).

No GDIS pode haver um conjunto de nós que executam uma ou mais tarefas do sistema de integração de fontes de dados. As principais entidades e suas tarefas são:

- *Provedores de Dados* que disponibilizam fontes de dados com ou sem esquema;
- *Mediadores* que provém somente o mapeamento entre esquemas; e
- *Clientes* que processam suas consultas.

Desta forma, a proposta do GDIS é oferecer um Tradutor/Mediador descentralizado através da utilização do OGSA-DQP. A arquitetura de camadas do GDIS está ilustrada na Figura 17 (COMITO et al., 2005).

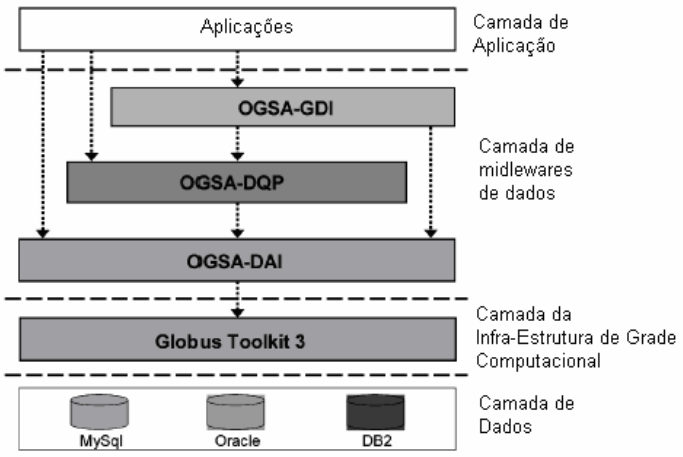


Figura 17: Arquitetura de Camadas do GDIS (COMITO et al., 2005).

No GDIS um ponto pode ser *Provedor de Dados* e/ou *Mediador* e/ou *Cliente.* O relacionamento semântico entre os esquemas é localizado usando uma *Peer Programming Language* (PPL). O PPL permite definir dois tipos de mapeamento: i) o *peer mappings* que estabelece as correspondências entre os esquemas em diferentes pontos, e ii) o *storage descriptions* que descrevem quais são os dados armazenados em um ponto para um ou mais pontos (COMITO et al., 2005).

O GDIS categoriza seus nós em quatro tipos, são eles (COMITO et al., 2005):

- Nós de Processamento: São nós capazes de executar consultas distribuídas;
- Nós de Integração de Dados: São nós que oferecem um conjunto de utilitários de integração que permitem estabelecer o mapeamento entre esquemas;
- Nós de Mapeamento: São nós que contém catálogos globais e todas as informações sobre o mapeamento entre esquemas.

- Nós Tradutores: São nós que permitem o acesso às fontes de dados.

No GDIS um nó pode executar qualquer serviço mencionado em conjunto com um ou mais outros serviços.

3.4.6 Comparação entre as Arquiteturas Propostas para a Integração de Fontes de Dados

As arquiteturas propostas apresentadas nas sessões 3.4.1, 3.4.2, 3.4.3, 3.4.4 e 3.4.5 são comparadas na Tabela 1 com base nas seguintes características: tipo de arquitetura, modelo canônico de dados, visão dos dados, flexibilidade do modelo de dados canônico, aplicabilidade na Grade Computacional e utilização de conversores de dados.

Características	**Garlic**	**MIX**	**MOCHA**	**WISE**	**GDIS**
Tipo de Arquitetura	Mediador	Mediador	Mediador	Mediador	P2P
Modelo de Dados Canônico	ODMG	XML	XML	XML	XML
Visão dos Dados	Virtual	Virtual	Virtual	Virtual	Virtual
Flexibilidade	Não	Sim	Sim	Sim	Sim
Aplicabilidade na Grade Computacional	Não	Não	Não	Não	Sim
Conversores de Dados	Não	Não	Não	Sim	Não

Tabela 1: Tabela de Comparações entre os Exemplos de Arquiteturas para a Integração de Fontes de Dados.

3.4.7 Arquitetura Proposta

O GISE é a arquitetura proposta neste trabalho para integrar fontes de dados heterogêneas e geograficamente distribuídas. O GISE permite que as fontes de dados sejam virtualizadas e acessadas de forma transparente na Grade Computacional.

O tipo de arquitetura utilizada pelo GISE é a arquitetura de Mediadores. Esta arquitetura permite a construção de componentes especializados capazes de resolver os conflitos encontrados no processo de integração de fontes de dados.

O modelo de dados canônico (MDC) do GISE foi construído através da linguagem XML. O XML possui a capacidade de representar tipos de dados estruturados e semi-estruturados, permitindo que a *Arquitetura GISE* seja altamente flexível.

A *Arquitetura GISE* proporciona aos seus usuários a abordagem virtual dos dados acessados. Desta forma, evitam-se a replicação das fontes de dados e consequentemente, problemas de atualização nas fontes de dados distribuídas. A *Arquitetura GISE* também disponibiliza conversores de formato, tipo e unidade por fonte de dados envolvida no processo de integração. Desta forma, a arquitetura pode disponibilizar os resultados aos usuários convertidos através de um padrão ou uma fórmula de conversão específica.

A única arquitetura comparada ao GISE nas sessões anteriores que possui a Grade Computacional como infra-estrutura para execução dos serviços de integração de fontes de dados foi o GDIS. A *Arquitetura GISE* e o GDIS utilizam como *middleware* para plataforma de Grade Computacional o *Globus Toolkit*. Porém, mesmo com a similaridade na infra-estrutura computacional as duas arquiteturas são heterogêneas.

O GDIS utiliza a arquitetura *P2P* onde um nó pode executar uma ou mais tarefas de integração simultaneamente e pode se comunicar com quaisquer outros nós para requisitar informações. Desta forma, o GDIS disponibiliza uma arquitetura totalmente distribuída. O problema deste tipo de arquitetura é garantir toda a consistência dos esquemas das fontes de

dados e desempenho nos processos de atualização. Com a utilização de uma abordagem *P2P,* o GDIS possui um *Serviço de Metadados* distribuído e as funcionalidades do *Mediador* estão limitadas ao armazenamento de informações sobre os mapeamentos entre esquemas locais.

Neste aspecto o GISE se diferencia do GDIS com a utilização da arquitetura de Mediadores. Desta forma, o GISE pode oferecer um conjunto de Mediadores para a execução das tarefas de integração de fontes de dados e também pode oferecer Mediadores especializados no tratamento de informações integradas de um domínio. Diferentemente do GDIS, o GISE possui um *Serviço de Metadados* integrado que não está fragmentado nos Mediadores da arquitetura.

Contudo, a *Arquitetura GISE* se difere das outras ferramentas discutidas, pois resolve os conflitos de denominação, tipo, formato e unidade decorrente do processo de integração de fontes de dados e proporciona uma arquitetura flexível, integrada aos serviços disponíveis na Grade Computacional. A *Arquitetura GISE,* através do uso de Mediadores, Tradutores e do Modelo de Dados Canônico, proporciona a virtualização de fontes de dados integradas na Grade Computacional. A arquitetura proposta neste trabalho, seus atributos, funcionalidades, aplicações e limitações serão exploradas nos próximos Capítulos.

Capítulo 4 – Descrição da Arquitetura GISE

Neste capítulo é apresentada a *Arquitetura GISE* (*Grid Integration SErvice*). Este Capítulo inicia com uma introdução à arquitetura, em seguida é apresentado o *Modelo de Dados Canônico* (MDC) utilizado para representar as fontes de dados locais e integradas. Na próxima seção são discutidos os principais componentes da arquitetura e as Camadas: *Cliente*, *Serviço de Metadados*, *Serviços de Integração* (Mediador), *Serviços do Globus Toolkit*, *Serviço de Acesso aos Dados* e *Fontes de Dados*.

4.1 Introdução à Arquitetura GISE

A *Arquitetura GISE* tem como objetivo fornecer aos usuários a virtualização e acesso integrado as fontes de dados heterogêneas e geograficamente distribuídas na Grade Computacional. A *Arquitetura GISE* integra fontes de dados resolvendo os conflitos semânticos e os conflitos estruturais de tipo, formato e unidade, identificados na seção 3.1.1.

A arquitetura proposta neste trabalho abrange uma grande variedade de usuários, uma vez que está sendo proposta como infra-estrutura de computação distribuída a Grade Computacional.

A utilização na Grade Computacional nesta arquitetura é um diferencial. Desta forma, é possível se aproveitar dos benefícios providos por esta infra-estrutura para um gerenciamento de recursos eficaz (com a utilização do GRAM), uma infra-estrutura eficaz de segurança e *login* único para múltiplas fontes de dados distribuídas (através da utilização do GSI), além de toda uma arquitetura para acesso transparente as fontes de dados heterogêneas (através da utilização do OGSA-DAI).

A Grade Computacional é um ambiente altamente dinâmico, por isso a *Arquitetura GISE* foi desenvolvida utilizando os benefícios da arquitetura de Mediadores, discutida na seção 3.3.3.

A *Arquitetura GISE* utiliza um Modelo de Dados Canônico capaz de representar múltiplas fontes de dados e diferentes formas de integração. O Modelo de Dados Canônico do GISE é apresentado a seguir.

4.2 Modelo de Dados Canônico do GISE

O Modelo de Dados Canônico da *Arquitetura GISE* (MDC) é denominado de GISE-MDC e utiliza a linguagem de marcação extensível XML (*eXtensible Markup Language*) para a representação das fontes de dados locais e integradas. O XML foi uma linguagem proposta pelo W3C como padrão para representação e troca de informações na Internet (GOLDMAN et al., 1999; CHRISTOPHIDES et al., 2000).

O XML é utilizado para fornecer uma representação uniforme e flexível de fontes de dados heterogêneas. Desta forma, possuímos um Modelo de Dados Canônico capaz de representar diversos modelos de dados. Vários sistemas de integração de fontes de dados têm utilizado o XML como base para a construção de um Modelo de Dados Canônico (GARDARIN et al., 1999; ABITEBOL et al., 2000).

O Modelo de Dados Canônico do GISE é composto por duas partes. A primeira é denominada como *GISE-LOCAL-SCHEMA* que é um esquema XML que define uma estrutura de agregação capaz de representar uma fonte de dados local. A segunda parte do Modelo de Dados Canônico é o *GISE-INTEGRATED-SCHEMA* que é um esquema XML responsável por representar a integração de um ou mais *GISE-LOCAL-SCHEMA*. Os dois esquemas utilizados na *Arquitetura GISE* são discutidos em detalhes a seguir.

4.2.1 Esquema Local (*GISE-LOCAL-SCHEMA*)

O *GISE-LOCAL-SCHEMA* é um arquivo XML construído através das informações contidas nas estruturas de dados para representar uma fonte de dados em particular na Grade Computacional.

Para permitir a integração de fontes de dados, o *GISE-LOCAL-SCHEMA* deve ser construído através de regras pré-estabelecidas. Para padronizar a criação do *GISE-LOCAL-SCHEMA* a *Arquitetura GISE* utiliza os esquemas XML. O esquema XML é um arquivo construído através do próprio XML para representar as regras e estruturas necessárias para a formação de um documento XML. O arquivo que contém um esquema XML é identificado com a extensão XSD. O arquivo que contém o esquema XML para a construção e validação do *GISE-LOCAL-SCHEMA* é identificado na *Arquitetura GISE* pelo arquivo *GISE-LOCAL-SCHEMA*.XSD e está ilustrado na Figura 18 (representação gráfica) e no Apêndice B.1.1.

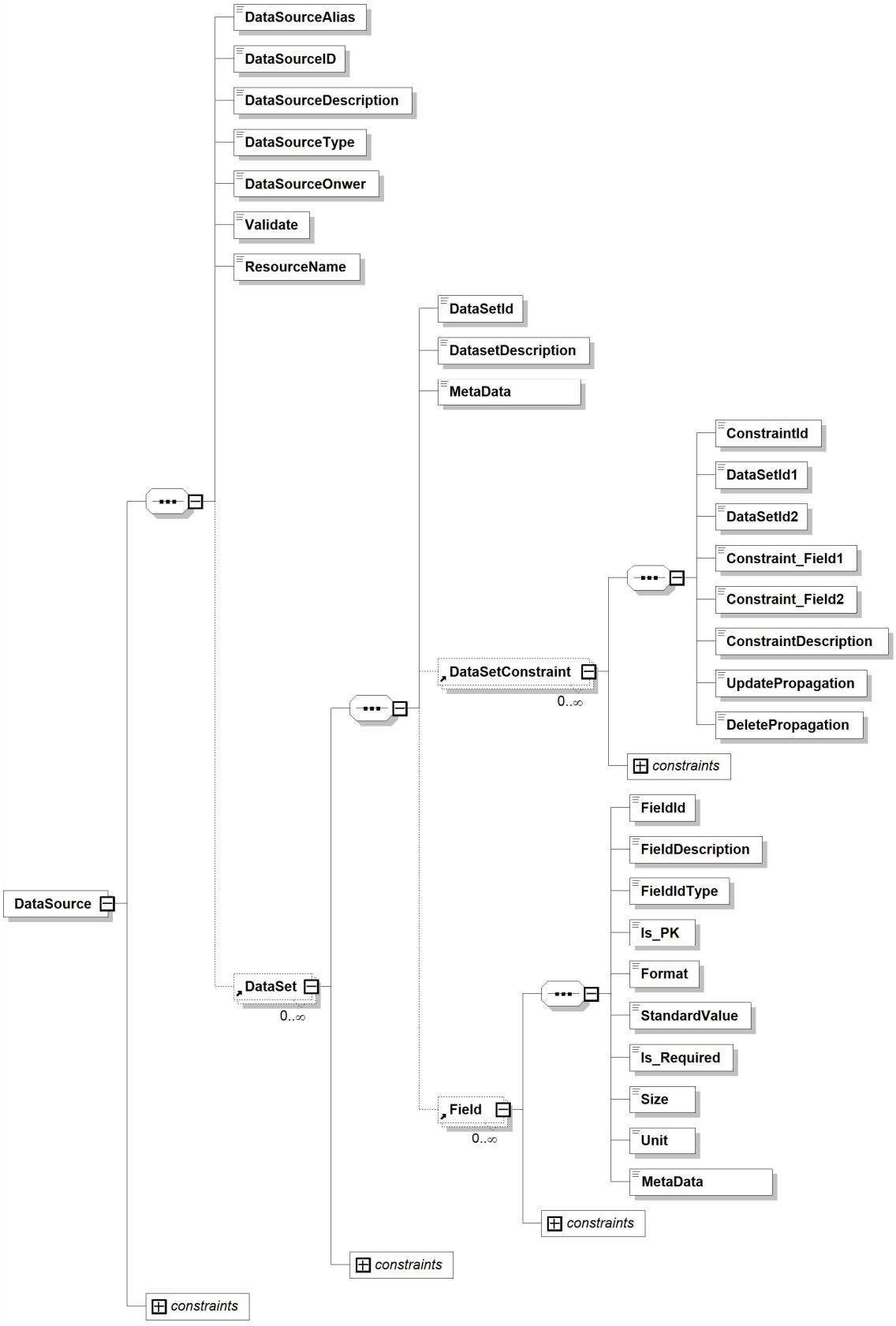


Figura 18: Representação Gráfica do *GISE-LOCAL-SCHEMA*.XSD.

O elemento DataSource é o elemento raiz do *GISE-LOCAL-SCHEMA* e armazena o nome da fonte de dados local. O DataSource é composto pelos seguintes elementos:

- DataSourceAlias: Elemento simples que armazena um apelido para a fonte de dados;
- DataSourceID: Elemento simples que armazena uma identificação para a fonte de dados;
- DataSourceDescription: Elemento simples que armazena uma descrição para a fonte de dados;
- DataSourceType: Elemento simples que armazena o tipo da fonte de dados;
- DataSourceOwner: Elemento simples que armazena o nome do proprietário da fonte de dados;
- Validate Elemento simples que informa se dos dados do Esquema Local mapeiam corretamente os atributos da fonte de dados;
- ResourceName: Elemento simples que armazena o nome do recurso. Este dado é utilizado para processamento de consultas pelo OGSA-DAI; e
- DataSet: Elemento composto que armazena informações sobre os conjuntos de dados disponíveis na fonte de dados.

Elementos Compostos e sua Composição no *GISE-LOCAL-SCHEMA*

O elemento DataSet é composto pelos seguintes elementos:

- DataSetId: Elemento simples que armazena uma identificação para um conjunto de dados;

- DatasetDescription: Elemento simples que armazena uma descrição para um conjunto de dados;
- MetaData: Elemento simples que armazena quaisquer informações adicionais sobre o conjunto de dados;
- DataSetConstraint: Elemento composto que armazena informações sobre as restrições existentes entre as fontes de dados; e
- Field: Elemento composto que armazena informações sobre os campos das fontes de dados.

O elemento DataSetConstraint do elemento DataSet são compostos pelos seguintes elementos:

- ConstraintID: Elemento simples que é uma identificação para a restrição da fonte de dados;
- DataSetId1: Elemento simples que identifica um conjunto de dados X que armazena o primeiro campo da restrição;
- DataSetId2: Elemento simples que identifica o conjunto de dados Y que armazena o segundo campo da restrição;
- Constraint_Field1: Elemento simples que identifica o primeiro campo da restrição no conjunto de dados X;
- Constraint_Field2: Elemento simples que identifica o segundo campo da restrição no conjunto de dados Y;
- UpdatePropagation: Elemento simples que identifica se o valor do primeiro campo sofrer uma alteração, o segundo deva ser automaticamente atualizado; e

- DeletePropagation: Elemento simples que identifica se o valor do primeiro campo seja excluído, o segundo deva ser automaticamente excluído.

O elemento Field do elemento DataSet é composto pelos seguintes elementos:

- FieldId: Elemento simples que fornece um identificador único para o campo mapeado;
- FieldDescription: Elemento simples que armazena a descrição do campo da fonte de dados;
- FieldIdType: Elemento simples que armazena informações sobre o tipo do campo da fonte de dados;
- Is_PK: Elemento simples que identifica se o campo é uma chave-primária;
- Format: Elemento simples que armazena informações sobre o formato do campo na fonte de dados;
- StandardValue: Elemento simples que armazena informações sobre o valor padrão do campo na fonte de dados;
- Is_Required: Elemento simples que identifica se o campo é sempre requerido em um processo de inclusão de dados;
- Size: Elemento simples que armazena informações sobre o tamanho do campo;
- Unit: Elemento simples que armazena a unidade do campo; e
- MetaData: Elemento simples que armazena quaisquer informações adicionais sobre o campo.

4.2.2 Esquema Integrado (*GISE-INTEGRATED-SCHEMA*)

O *GISE-INTEGRATED-SCHEMA* é um arquivo XML construído através das informações armazenadas no *GISE-LOCAL-SCHEMA*. O *GISE-INTEGRATED-SCHEMA* representa a integração de fontes de dados (duas ou mais) e está estruturado para fornecer suporte para o armazenamento das informações vitais para que seja possível a execução de um processo de consulta em fontes de dados distribuídas.

Para permitir a padronização na criação dos arquivos *GISE-INTEGRATED-SCHEMA*, assim como acontece com o *GISE-LOCAL-SCHEMA*, o *GISE-INTEGRATED-SCHEMA* deve ser construído através de regras pré-estabelecidas. O arquivo de esquema utilizado para a construção e validação do *GISE-INTEGRATED-SCHEMA* é identificado na *Arquitetura GISE* pelo arquivo *GISE-INTEGRATED-SCHEMA.XSD* e está ilustrado na Figura 19 (representação gráfica) e no Apêndice B.1.2.

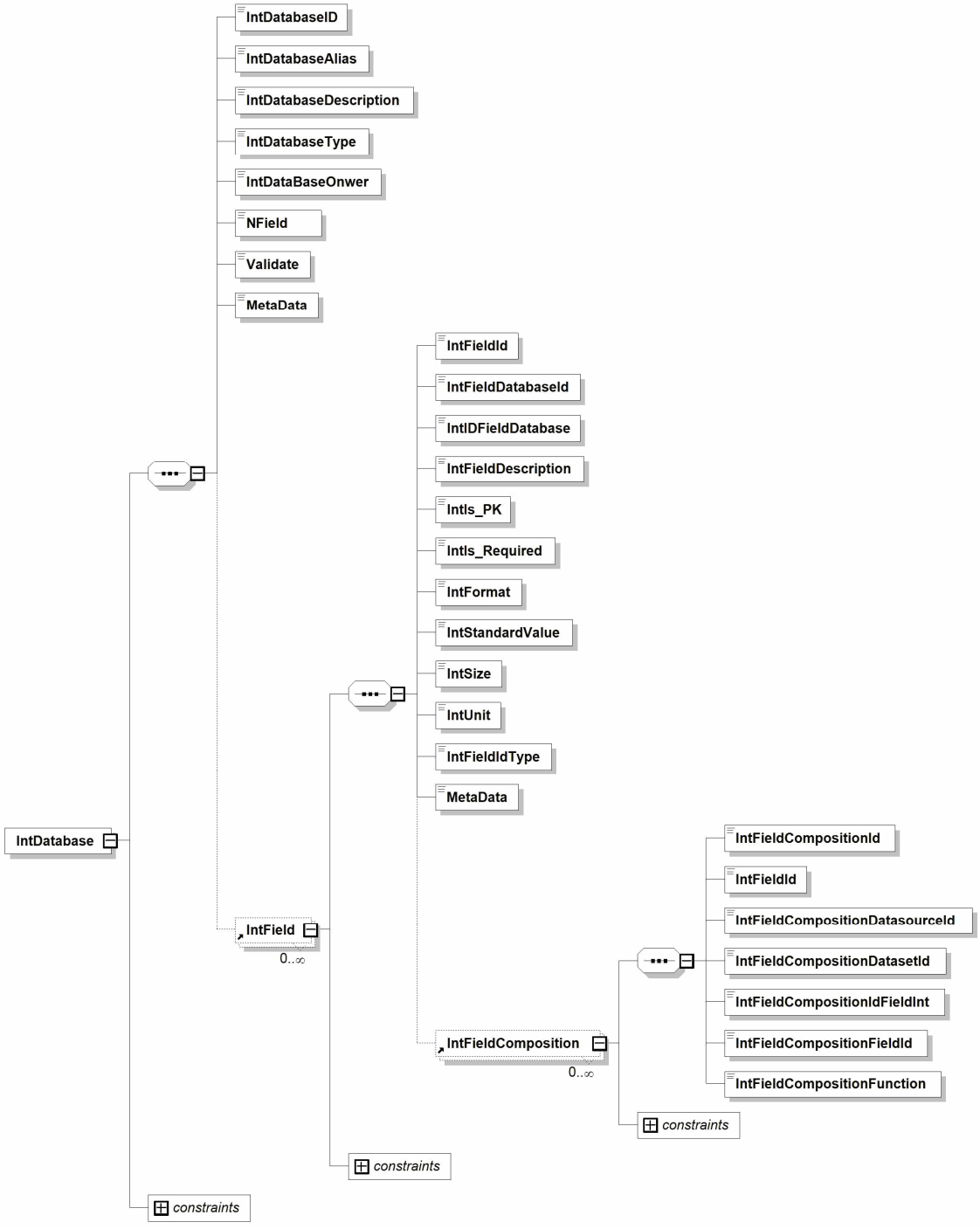


Figura 19: Representação Gráfica do Arquivo

***GISE-INTEGRATED-SCHEMA*.XSD.**

O elemento IntDatabase é o elemento raiz do *GISE-INTEGRATED-SCHEMA* e armazena o nome da fonte de dados integrada. O IntDatabase é composto pelos seguintes elementos:

- IntDatabaseId: Elemento simples que identifica a fonte de dados integrada;
- IntDatabaseAlias: Elemento simples que armazena um apelido para a fonte de dados integrada;
- IntDatabaseDescription: Elemento simples que armazena uma descrição para a fonte de dados integrada;
- IntDatabaseType: Elemento simples que identifica o tipo da fonte de dados integrada;
- IntDatabaseOwner: Elemento simples que identifica o proprietário da fonte de dados integrada;
- NFields: Elemento simples que armazena o número de campos existentes na fonte de dados integrada;
- Validate: Elemento simples que determina se o Esquema Integrado é válido;
- MetaData: Elemento simples que armazena quaisquer meta informações sobre a fonte de dados integrada; e
- IntField: Elemento composto que armazena informações sobre os campos integrados da fonte de dados.

O elemento IntField do elemento IntDatabase é composto pelos seguintes elementos:

- IntFieldId: Elemento simples que armazena uma identificação global para o campo integrado;
- IntFieldDataBaseId: Elemento simples que armazena a identificação da fonte de dados local para do campo composto;

- IntIDFieldDatabase: Elemento simples que armazena uma identificação local para o campo integrado;
- IntFieldDescription: Elemento simples que armazena uma descrição para um campo integrado da fonte de dados;
- IntIs_PK: Elemento simples que identifica se o campo integrado é uma chave primária;
- IntIs_Required: Elemento simples que identifica se o campo integrado é requerido em um processo de inserção de dados;
- IntFormat: Elemento simples que armazena o formato do campo integrado;
- IntStandardValue: Elemento simples que armazena o valor padrão para o campo integrado;
- IntSize: Elemento simples que armazena o tamanho do campo integrado;
- IntUnit: Elemento simples que armazena a unidade do campo integrado;
- IntFieldIdType: Elemento simples que armazena o tipo do campo integrado;
- MetaData: Elemento simples que armazena quaisquer meta informações sobre o campo integrado; e
- IntFieldCompostion: Elemento composto que armazena informações sobre a composição do campo integrado;

O elemento composto IntFieldComposition do elemento IntField é composto pelos seguintes elementos:

- IntFieldCompostionId: Elemento simples que armazena uma identificação para a composição do campo integrado;
- InFieldId: Elemento simples que armazena a identificação do campo na fonte de dados integrada;
- IntFieldCompositionDataSourceId: Elemento simples que armazena a identificação da fonte de dados que armazena o campo local;
- IntFieldCompositionDataSetId: Elemento simples que armazena a identificação do conjunto de dados que armazena o campo local;
- IntFieldCompositionIdFieldInt: Elemento simples que armazena o número de seqüência do campo composto;
- IntFieldCompositionFieldId: Elemento simples que armazena a identificação global de um campo local do *GISE-LOCAL-SCHEMA*; e
- IntFieldCompositionFunction: Elemento simples que armazena uma função de conversão do campo. Este elemento armazena informações necessárias para resolver o conflito de unidade encontrado no processo de integração de fonte de dados.

4.3 Camada de Serviços e Componentes da Arquitetura GISE

A *Arquitetura GISE* está dividida em camadas e suas funcionalidades são fornecidas através da especialização dos componentes que compõe cada camada. A Figura 20 ilustra a *Arquitetura GISE*, suas camadas e componentes.

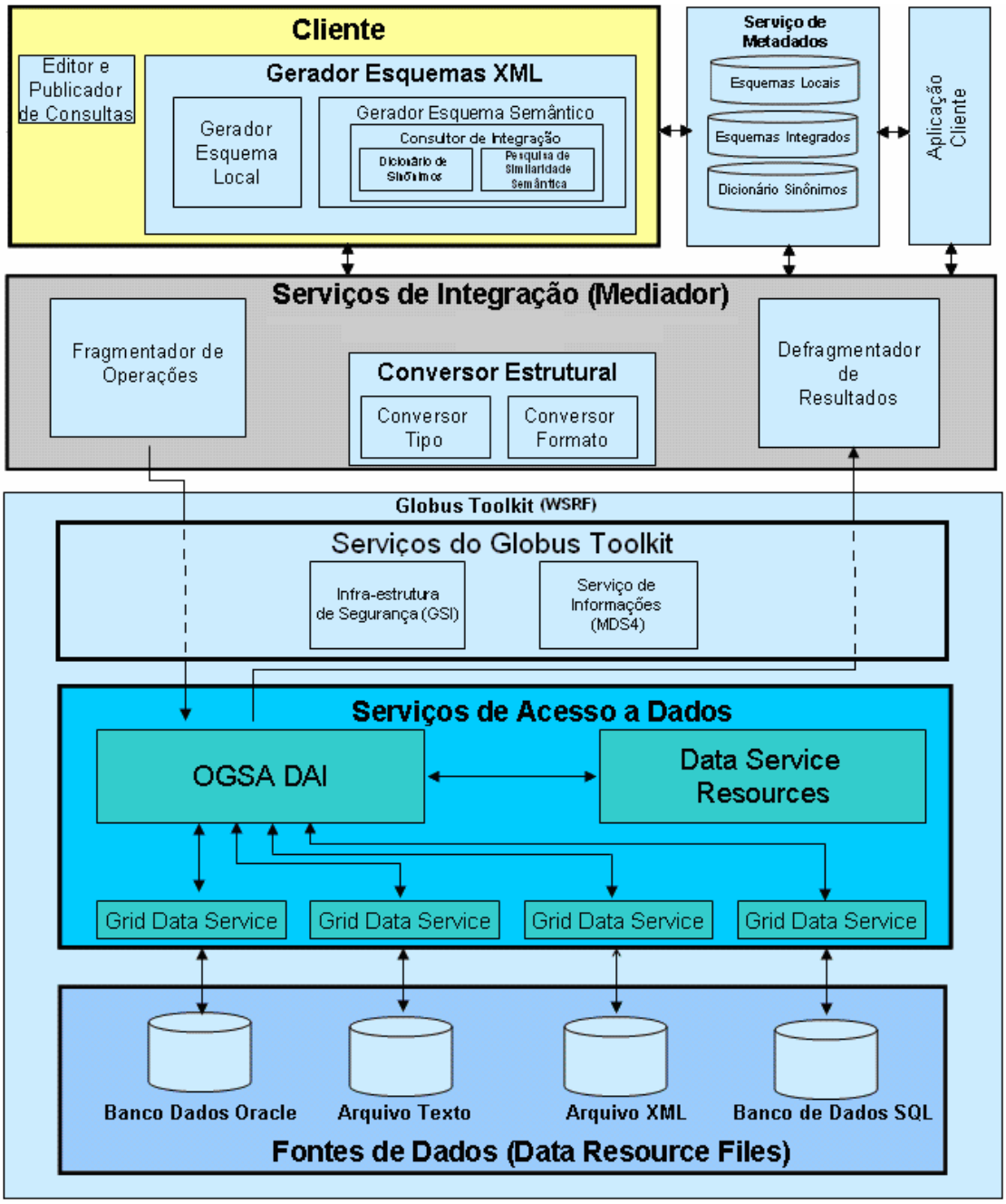


Figura 20: Arquitetura GISE, suas camadas e principais componentes.

Os componentes que compõem as camadas da *Arquitetura GISE* são implementados de forma independente através de classes. O Diagrama de Classes ilustrado na Figura 21 identificam as classes utilizadas pela arquitetura, seus atributos e os métodos disponíveis para cada classe.

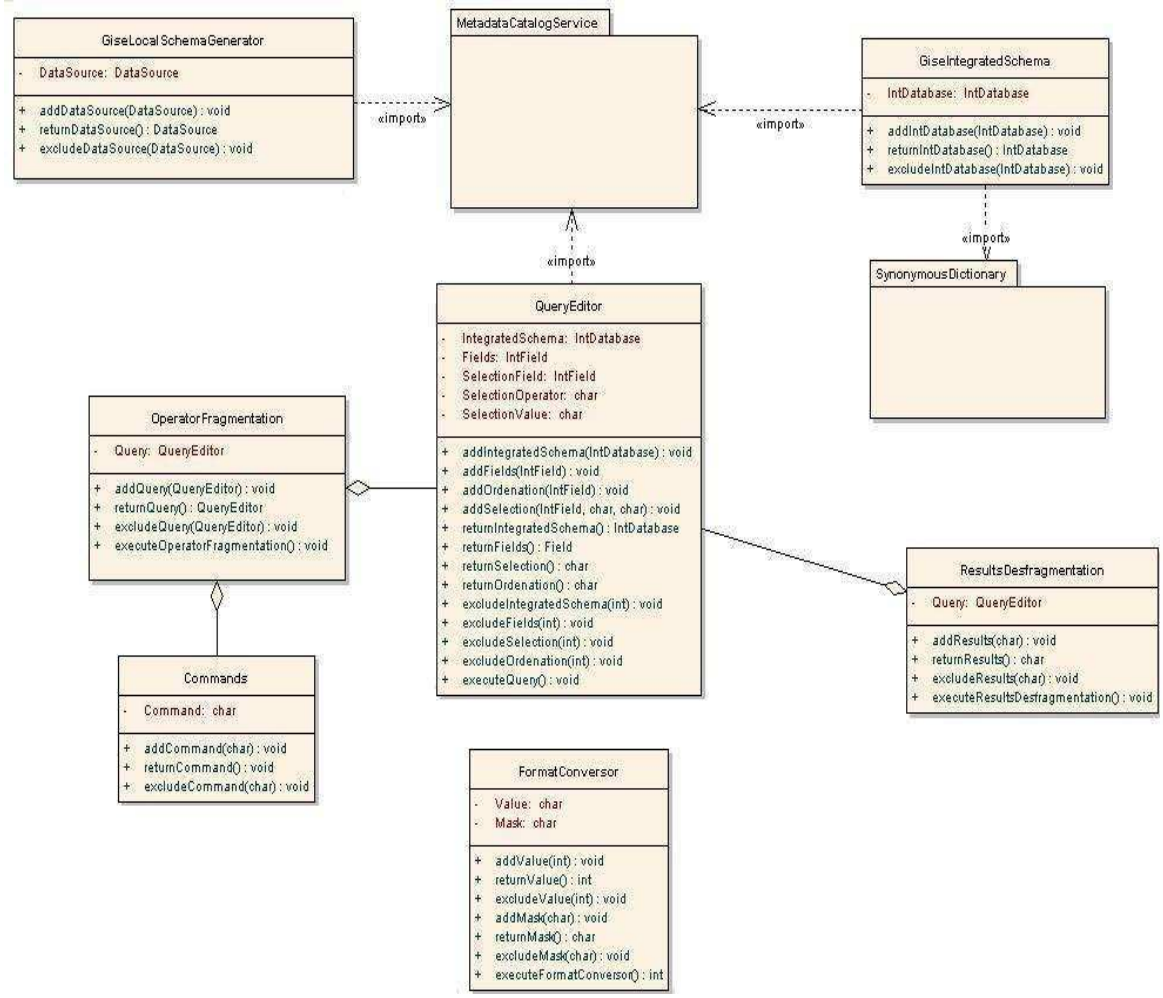


Figura 21: Diagrama de Classes da Arquitetura GISE.

Conforme identificado na Figura 21, a *Arquitetura GISE* possui seis classes e dois pacotes. A classe *GiseLocalSchemaGenerator* é a classe responsável pelo controle e armazenamento de informações do GISE-LOCAL-SCHEMA e a classe *GiseIntegratedSchema* é responsável pelas informações do GISE-INTEGRATED-SCHEMA. Ambas as classes utilizam o pacote *MetadataCatalogService.* O pacote *MetadataCatalogService* representa as classes que mapeiam as informações da *Camada Serviço de Metadados* e sua composição está ilustrada na Figura 48. A classe *GiseIntegratedSchema* utiliza-se das informações do pacote *SynonymousDictionary* para ajudar o usuário no processo de geração de esquemas integrados. O conteúdo do pacote *SynonymousDictionary* está ilustrado na Figura 49.

As consultas integradas em fontes de dados virtuais são disponibilizadas na *Arquitetura GISE* através da classe *QueryEditor*. Esta classe se responsabiliza pelo armazenamento das informações necessárias para o processamento de consultas através das classes *OperatorFragmentation* e *ResultsDesfragmentation.* A classe *OperatorFragmentation* com utilização da classe *Commands* gera e armazena respectivamente, os comandos necessários para acesso às fontes de dados e a classe *ResultsDesfragmentation* cria uma visão única dos resultados individualizados para acesso pelos usuários. A classe *ResultsDesfragmentation* utiliza a classe *FormatConversor* para realizar, quando necessário, a conversão de formatação dos diferentes valores nos atributos compostos.

As Figuras 22 até a Figura 28 ilustram as funcionalidades da *Arquitetura GISE* através de Diagramas de Caso de Uso, apresentando uma visão externa do sistema e suas interações com o mundo exterior.

A Figura 22 ilustra as interações necessárias na *Arquitetura GISE* para que um cliente possa cadastrar Esquemas Locais ou Esquemas Integrados das fontes de dados. Inicialmente o usuário envia o esquema a ser cadastrado para o *Serviço de Metadados* e o mesmo se responsabiliza pelo envio das informações aos seus respectivos repositórios. Após a informação ser gravada, o *Serviço de Metadados* envia uma notificação ao usuário informando o sucesso da operação.

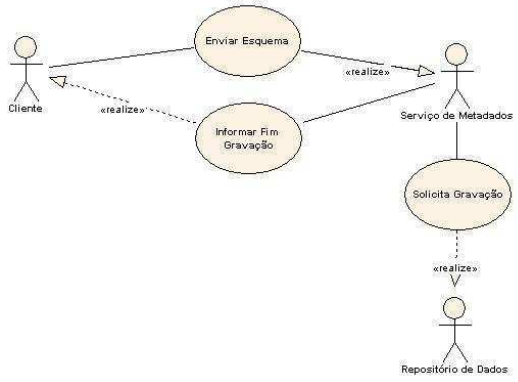


Figura 22: Diagrama de Caso de Uso da Inclusão de Esquemas no Serviço de Metadados.

A Figura 23 ilustra as interações necessárias para a alteração das informações residentes nos repositórios de dados da *Arquitetura GISE*. Inicialmente o usuário realiza uma busca no *Serviço de Metadados* do esquema que será atualizado e, após receber o esquema solicitado, o usuário deve alterar as informações necessárias e enviar-las novamente para o *Serviço de Metadados*. A atualização dos repositórios de informação é realizada pelo *Serviço de Metadados* que ao fim da operação, notifica o usuário.

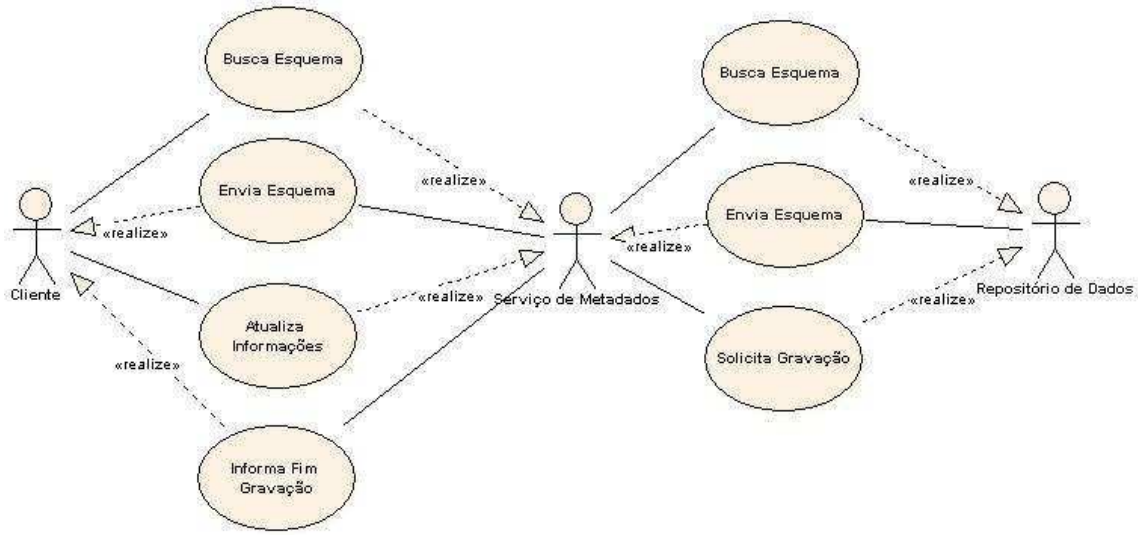


Figura 23: Diagrama de Caso de Uso da Alteração de Esquemas no Serviço de Metadados.

A consulta de esquemas no *Serviço de Metadados* é realizada conforme ilustrado na Figura 24. A consulta de esquemas, embora esteja representada pela entidade Cliente na Figura 24, também pode ser iniciada pelo *Fragmentador de Operações* e pelo *Desfragmentador de Resultados.* Para que o *Serviço de Metadados* retorne corretamente o esquema consultado, o cliente deverá informar a identificação única (chave primária) do esquema. Após realizar a busca nos repositórios de informação, o *Serviço de Metadados* retorna o esquema desejado ao cliente.

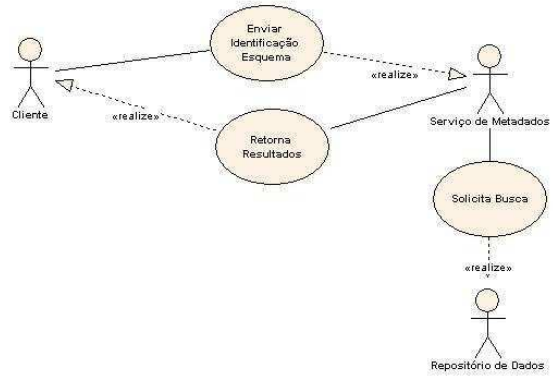


Figura 24: Diagrama de Caso de Uso da Consulta de Esquemas no Serviço de Metadados.

Na *Arquitetura GISE,* quando a estrutura de uma fonte de dados local é atualizada, todos os Esquemas Locais e Esquemas Integrados referentes à fonte de dados atualizada são invalidadas. A Figura 25 ilustra as interações que ocorrem no processo de invalidação de esquemas. O processo inicia com o cliente solicitando uma busca pelo *Serviço de Metadados* de um esquema de interesse. Após o retorno dos resultados, o cliente solicita ao *Serviço de Metadados* a invalidação do esquema e o mesmo se responsabiliza pela atualização dos repositórios de dados.

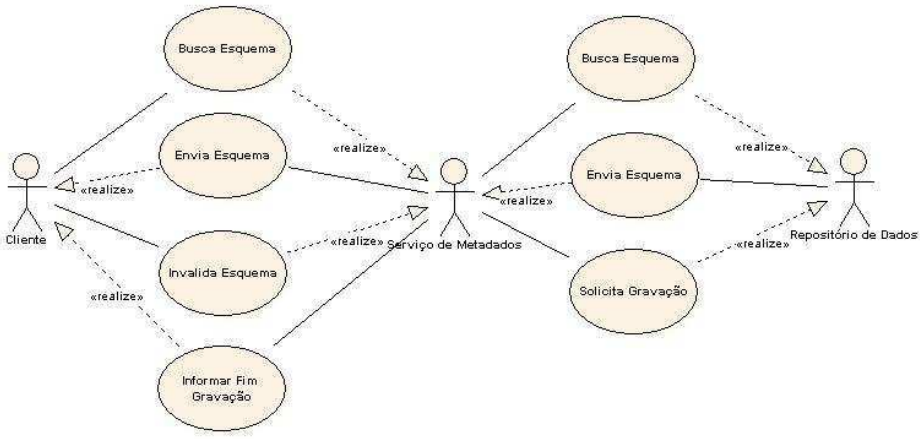


Figura 25: Diagrama de Caso de Uso da Invalidação de Esquemas no Serviço de Metadados.

O processo para cadastro de termos está ilustrado na Figura 26. Este processo é semelhante ao processo para cadastro de esquemas (Figura 22). A única variação é que o cliente, ao invés de enviar um Esquema Local ou Esquema Integrado para o *Serviço de Metadados*, envia um termo para ser adicionado ao repositório de dados do *Dicionário de Sinônimos* pelo *Serviço de Metadados.*

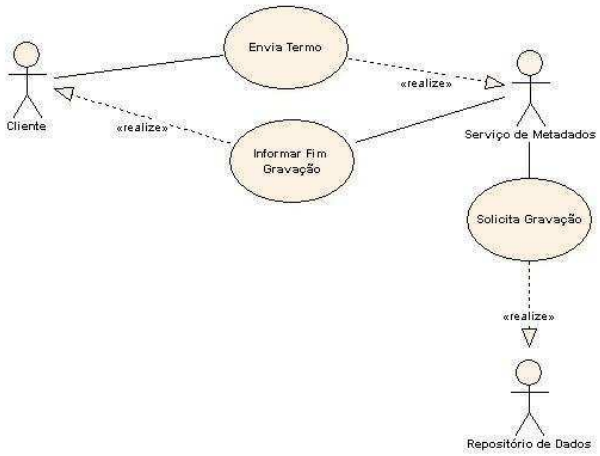


Figura 26: Diagrama de Caso de Uso do Cadastro de Termos.

Após o cadastramento dos termos, o cliente tem a possibilidade de cadastrar sinônimos para os termos informados. Desta forma, é disponibilizada uma ferramenta capaz de auxiliar o usuário no processo de geração de Esquemas Integrados. A Figura 27 ilustra o cadastramento de sinônimos da *Arquitetura GISE.* O cliente inicialmente executa uma busca do termo pelo *Serviço de Metadados* no repositório de dados. Uma vez recebido o termo pesquisado, o cliente imediatamente faz uma nova busca por outro termo. Depois que os dois termos são retornados, o usuário solicita ao *Serviço de Metadados* o cadastramento de sinônimos para os termos retornados. Finalmente o *Serviço de Metadados* atualiza os repositórios de dados pertinentes e informa ao cliente da confirmação da gravação dos dados.

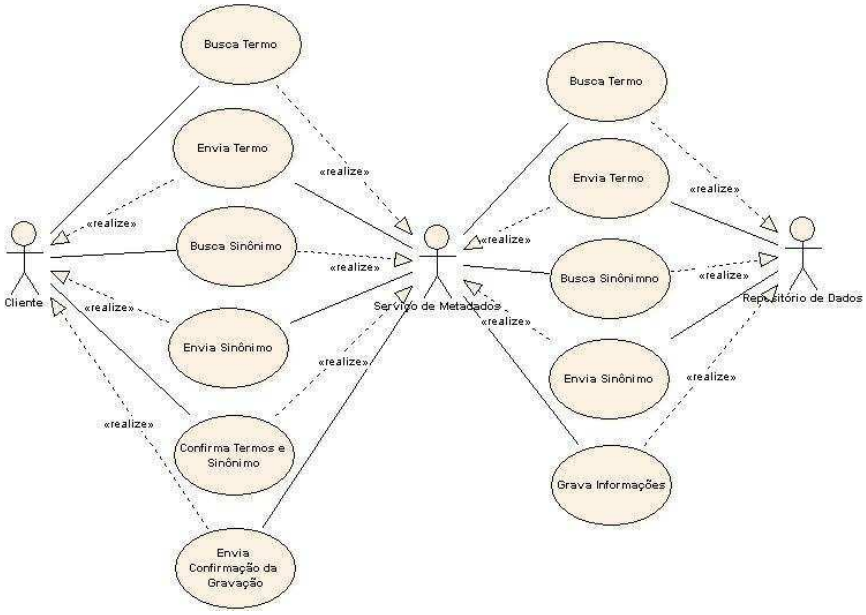


Figura 27: Diagrama de Caso de Uso do Cadastro de Sinônimos.

A Figura 28 ilustra todas as entidades e interações necessárias para o processamento de uma consulta integrada em uma fonte de dados virtual. O processo é iniciado por um cliente que solicita o Esquema Integrado de uma fonte de dados ao *Editor e Publicador de Consultas* que, por sua vez, busca o Esquema Integrado no *Serviço de Metadados.*

O *Serviço de Metadados* busca as informações do Esquema Integrado no repositório de dados e o transmite para o *Editor e Publicador de Consultas,* que por sua vez, disponibiliza o Esquema Integrado para que o cliente construa a sua consulta. Após criação da consulta, o *Editor e Publicador de Consultas* enviam as informações para o *Fragmentador de Operações.*

O *Fragmentador de Operações* recebe a consulta integrada e com informações obtidas no *Serviço de Metadados,* realiza a sua decomposição. O *Fragmentador de Operações* envia as solicitações individuais por fonte de dados para o *OGSA-DAI* que em seqüência, processa as solicitações na respectiva fonte de dados e envia os resultados para o *Desfragmentador de Resultados.*

O *Desfragmentador de Resultados* obtém informações no *Serviço de Metadados* e realiza a composição dos resultados recebidos em uma única resposta integrada. Caso necessário, o *Desfragmentador de Resultados* solicita ao *Conversor Estrutural* a execução das conversões necessárias nos atributos das fontes de dados para se disponibilizar resposta em um formato único ao cliente.

Finalmente, o *Desfragmentador de Resultados* envia a solicitação do cliente para o *Editor e Publicador de Consultas* e o mesmo disponibiliza a consulta integrada na fonte de dados virtual ao cliente.

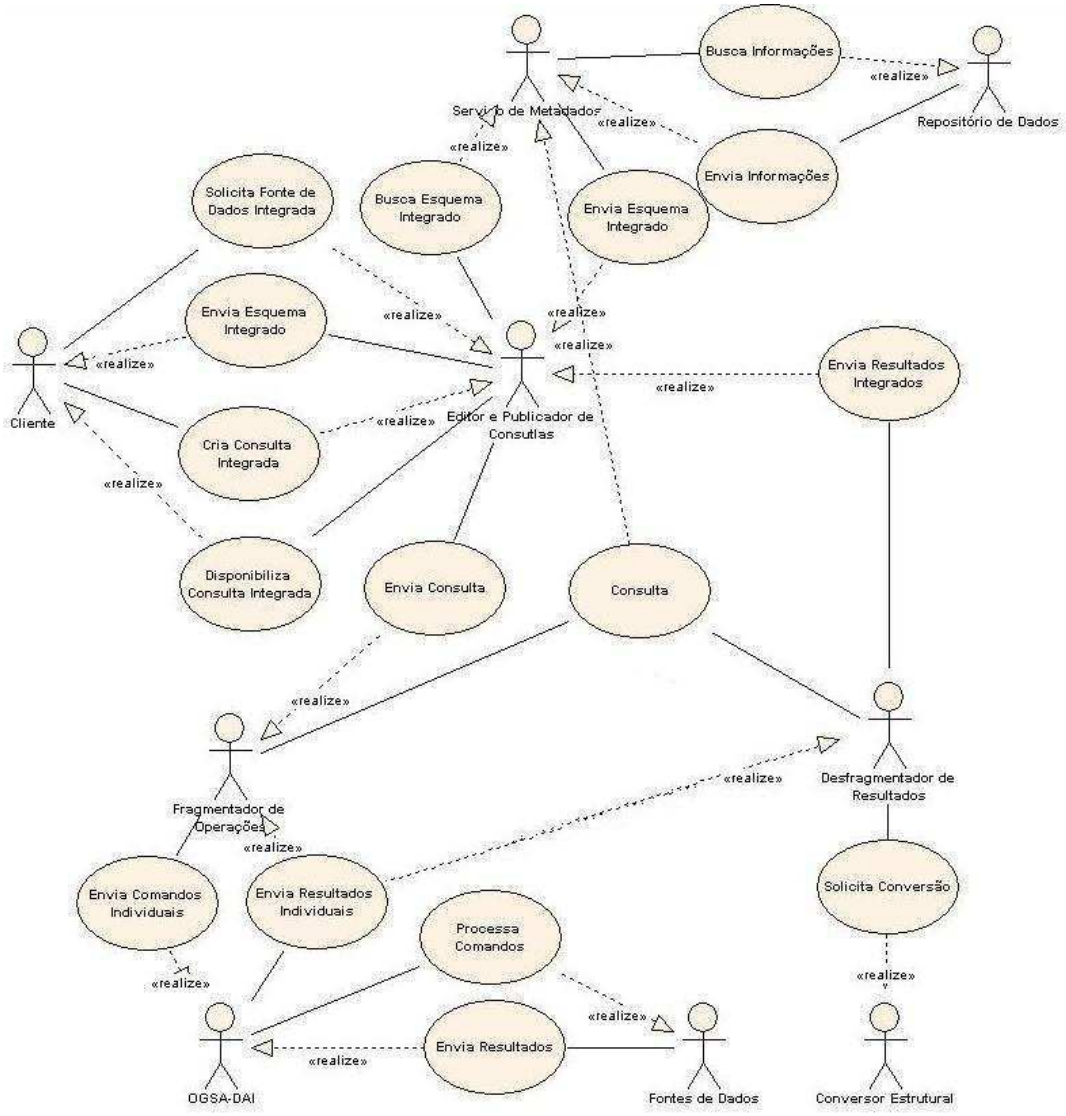


Figura 28: Diagrama de Caso de Uso da Execução de Consultas Integradas.

4.3.1 Camada Cliente

A *Camada Cliente* é a interface dos usuários com a *Arquitetura GISE*. Esta camada é composta pelos gerenciadores de esquemas locais (*GISE-LOCAL-SCHEMA*), esquemas integrados (*GISE-INTEGRATED-SCHEMA*) e a ferramenta para execução de consulta de dados distribuídos. Nesta camada também é implementado o Consultor de Integração que tem como objetivo facilitar o processo de integrações dos esquemas locais.

Esta camada recebe as requisições através do Editor e Publicador de Consultas e as encaminha para a *Camada Serviços de Integração* (Mediador), que se encarrega de enviá-las para os Tradutores específicos das fontes de dados envolvidas no processo de integração.

A *Camada Cliente* pode ser facilmente substituída na *Arquitetura GISE*. Uma aplicação com as mesmas funcionalidades pode executar suas atividades.

As funcionalidades providas pela *Camada Cliente* são implementadas na *Arquitetura GISE* através do *Sistema GISE*. O *Sistema GISE*, seus componentes e seu funcionamento são expostos a seguir.

4.3.1.1 Menu Principal

O *Menu Principal* é a porta de acesso às funcionalidades da *Camada Cliente*. O *Menu Principal* é dividido em duas opções principais, a *Gerador de Esquemas* e *Utilitários de Integração*, conforme ilustrado na Figura 29.

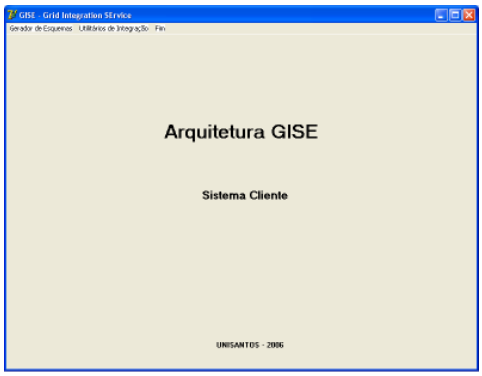


Figura 29: Menu Principal do Sistema.

Gerador de Esquemas

A opção de *Geração de Esquemas* do *Menu Principal* do GISE é composta pelas opções *Esquemas Locais* (*GISE-LOCAL-SCHEMA*) e *Esquemas Integrados* (*GISE-INTEGRATED-SCHEMA*), conforme ilustrado na Figura 30.

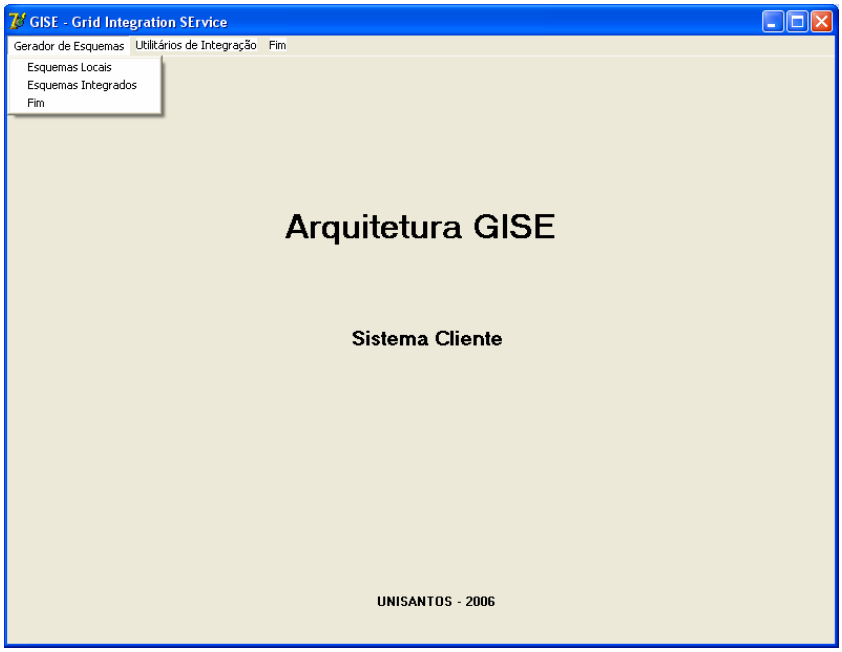


Figura 30: Opções do Gerador de Esquemas.

Utilitários de Integração

A opção de *Utilitários de Integração* do *Menu Principal* do GISE disponibiliza acesso ao *Editor de Consultas Integradas* e as ferramentas do *Consultor de Integração*, conforme ilustrado na Figura 31.

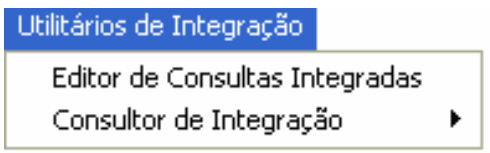


Figura 31: Opções do Menu Utilitários de Integração.

Consultor de Integração

O *Consultor de Integração* é a opção responsável por fornecer acesso às interfaces do *Dicionário de Sinônimos* e da *Pesquisa de Similaridade Semântica*. Estas interfaces auxiliam os usuários no processo de integração de esquemas locais (*GISE-LOCAL-SCHEMA*). As opções do *Consultor de Integração* estão ilustradas na Figura 32.

Figura 32: Opções do Consultor de Integração.

4.3.1.2 Gerador de Esquemas Locais

O *Gerador de Esquemas Locais* é a interface que recebe informações dos usuários para criação dos esquemas locais. No momento que o usuário escolher esta opção, é apresentada uma tela com todos os esquemas locais já presentes no *Serviço de Metadados* da arquitetura. Neste momento, o usuário poderá escolher entre cadastrar um novo Esquema Local ou gerar um arquivo XML com todas as informações sobre as fontes de dados locais presentes no *Serviço de Metadados*, conforme ilustrado na Figura 33.

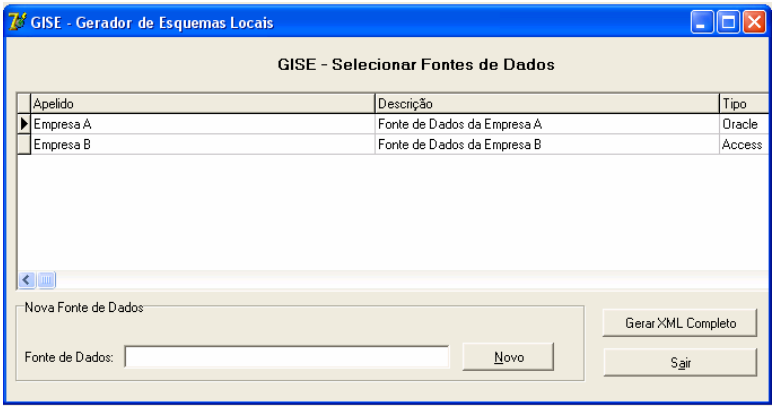


Figura 33: Tela Principal do Gerador de Esquemas Locais.

Caso o usuário preencha o nome da fonte de dados e clique na opção *Novo*, é apresentada uma tela com informações básicas sobre a fonte de dados a ser cadastrada. Caso o usuário efetue um duplo clique em um nome de fonte de dados pré-existente, é aberta a mesma tela com objetivo de alterar as informações básicas da fonte de dados, conforme ilustrado na Figura 34.

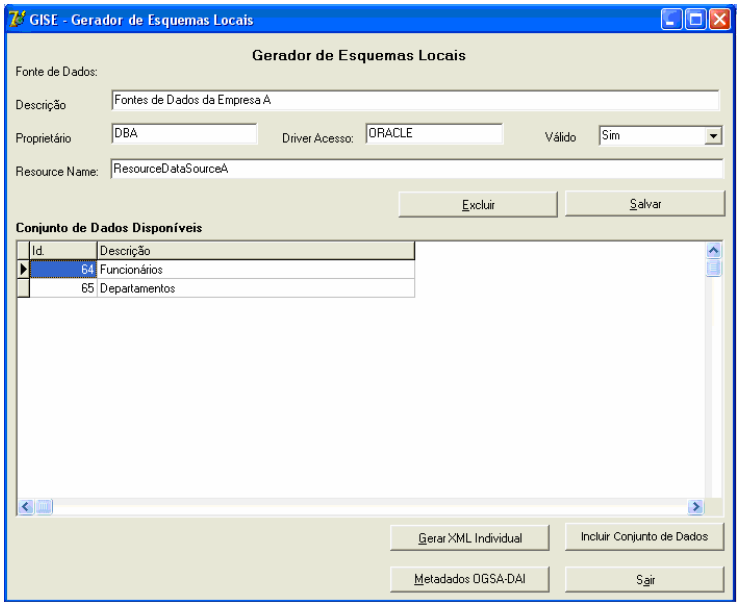


Figura 34: Tela com Informações Básicas da Fonte de Dados.

Na tela de *Informações Básicas da Fonte de Dados* o usuário poderá gerenciar o cadastro de sua fonte de dados. Estão disponíveis neste momento as seguintes informações: Descrição, proprietário, *driver* utilizado (Tradutor) para acesso a fonte de dados, validação e *resource name*. Nesta tela (Figura 34) o usuário também tem disponível uma opção para gerenciar os conjuntos de dados de sua fonte de dados, gerar e visualizar o *GISE-LOCAL-SCHEMA* ou atualizar as informações contidas no *Serviço de Metadados*.

Caso o usuário deseje gerar e visualizar as informações do *GISE-LOCAL-SCHEMA* poderá clicar no botão *Gerar XML Individual*. Desta forma, é apresentada uma tela conforme ilustrado na Figura 35.

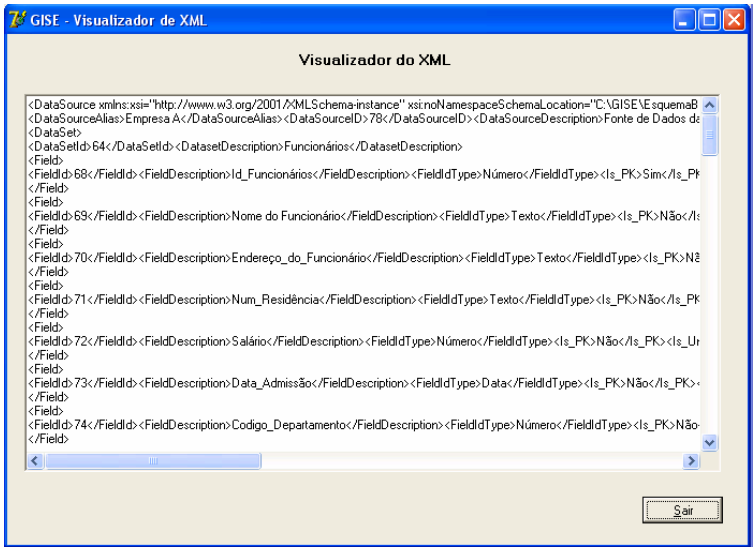


Figura 35: Visualizador do XML.

Para gerenciar os conjuntos de dados de uma fonte de dados, o usuário poderá efetuar um duplo clique no conjunto de dados de interesse ou pressionar o botão I*ncluir Conjunto de Dados,* conforme ilustrado na Figura 34. As informações de gerenciamento dos conjuntos de dados estão ilustradas na Figura 36.

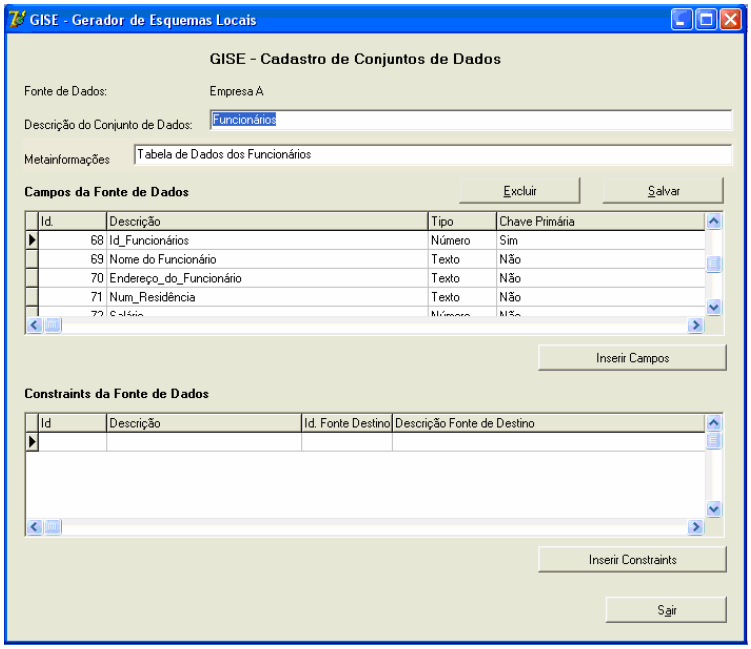


Figura 36: Tela para Cadastro de Conjunto de Dados.

Na tela para *Cadastro de Conjunto de Dados* (Figura 36) o usuário poderá informar uma descrição e informações adicionais para o conjunto de dados, incluírem campos ao conjunto de dados ou descrever um relacionamento para o conjunto de dados. Caso o usuário escolha a opção de *Inserir Campos*, será apresentada uma tela para cadastro das informações de campos, conforme ilustrado na Figura 37:

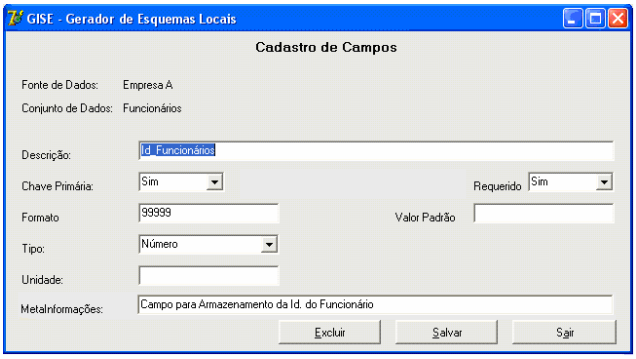


Figura 37: Tela para Cadastro de Campos.

Na tela para *Cadastro de Cam*pos (Figura 37), o usuário poderá informar: Descrição, chave primária (sim/não), requerido (sim/não), formato, tipo (número/texto/data), unidade e informações adicionais sobre o campo.

Caso o usuário optar pela opção *Inserir Constraint* (Figura 36), será apresentada uma tela para o *Cadastro de Relacionamento entre Conjuntos de Fontes de Dados*, conforme ilustrado na Figura 38.

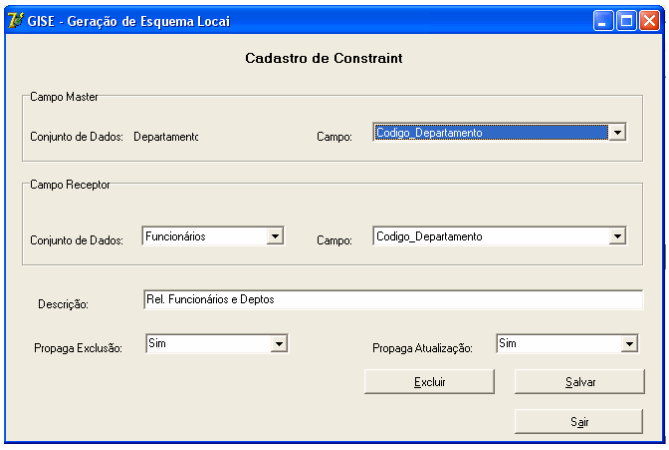


Figura 38: Tela para Cadastro de Relacionamentos.

Na tela para *Cadastro de Relacionamentos* (Figura 38) o usuário poderá informar quais são os conjuntos de dados envolvidos no relacionamento e seus respectivos campos. Neste momento, também é informado uma descrição para identificar o relacionamento e as suas características.

4.3.1.3 Gerador de Esquemas Integrados

A opção *Geração de Esquemas Integrados* fornece uma interface com os usuários para a geração de esquemas integrados. Caso o usuário escolha por esta opção no *Menu Principal* (Figura 30), será apresentada uma tela conforme ilustrado na Figura 39.

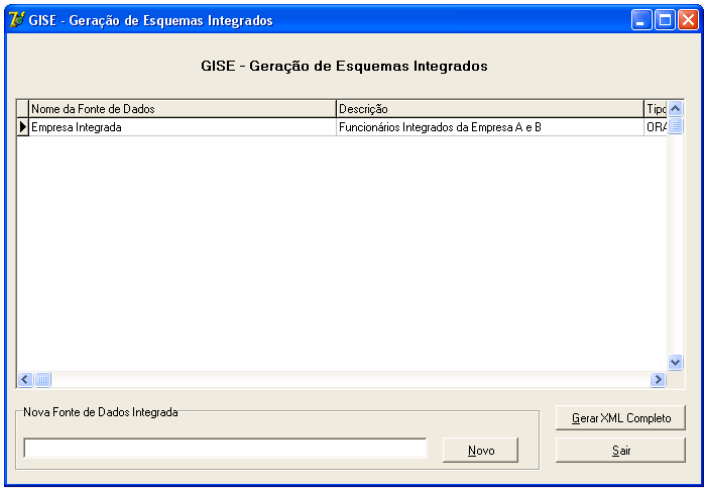


Figura 39: Tela Principal para Cadastro de Esquemas Integrados.

Na tela principal para *Cadastro de Esquemas Integrados* (Figura 39), caso o usuário preencha o nome da fonte de dados integrada e clique na opção *Novo*, é apresentada uma tela para entrar com as informações básicas sobre a fonte de dados integrada a ser cadastrada. Caso o usuário efetue um duplo clique em um nome de fonte de dados integrada pré-existente, é aberta uma tela para alteração das informações básicas da fonte de dados integrada conforme ilustrado na Figura 40.

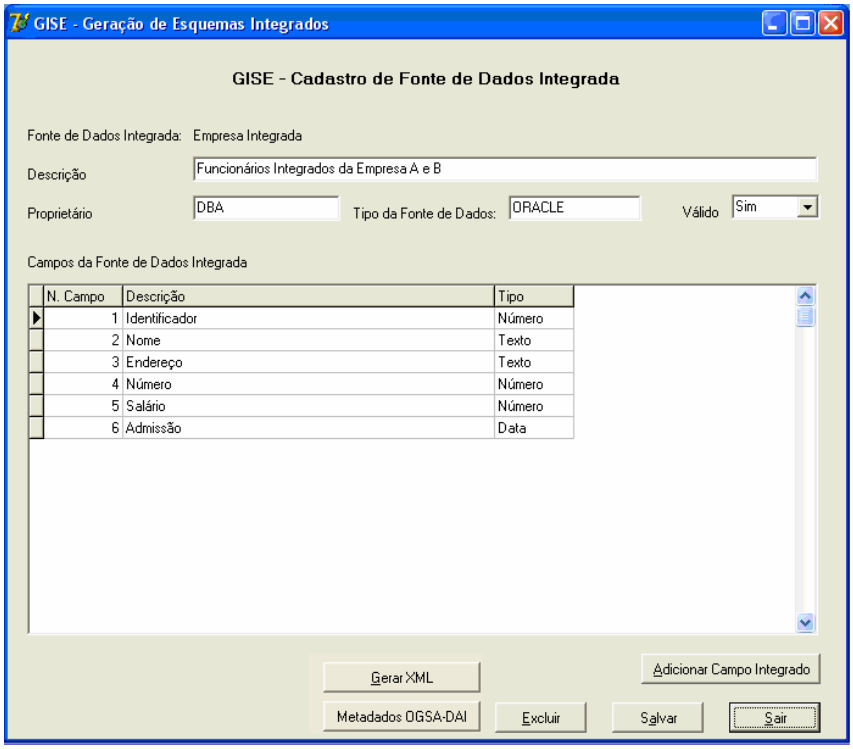


Figura 40: Tela Principal para o Cadastro de Fontes de Dados Integrada.

Na tela ilustrada na Figura 40 o usuário poderá gerenciar o cadastro de sua fonte de dados integrada. Estão disponíveis as informações de descrição, proprietário e driver utilizado (Tradutor) para acesso a fonte de dados. Neste momento, o usuário tem disponível uma opção para incluir um campo integrado à fonte de dados, gerar e visualizar as informações do *GISE-INTEGRATED-SCHEMA* e atualizar o *Serviço de Metadados*. A ilustração da ferramenta para visualização de esquemas XML está ilustrada na Figura 35.

Caso o usuário clique no botão *Adicionar Campo Integrado* será apresentada uma tela com informações básicas sobre o campo integrado, conforme ilustrado na Figura 41.

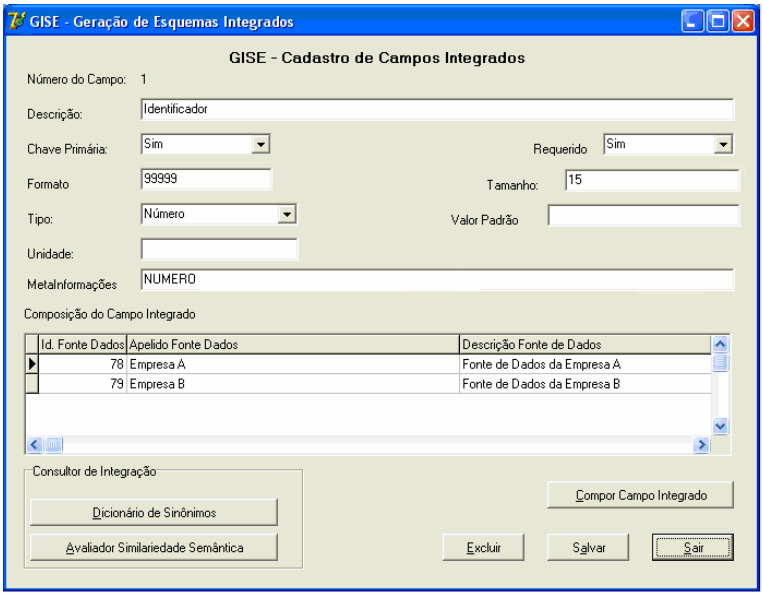


Figura 41: Tela de Cadastro para Informações Básicas de Campos em Fontes de Dados Integradas.

Na tela ilustrada na Figura 41 o usuário poderá cadastrar as seguintes informações: Descrição, chave primária (sim / não), requerido (sim / não), formato, tamanho, tipo, valor padrão, unidade e informações adicionais. Neste momento o usuário poderá clicar na opção *Compor Campo Integrado* para definir quais os campos dos esquemas locais (*GISE-LOCAL-SCHEMA*) irão compor o campo integrado cadastrado. Para facilitar o processo de cadastro de campos integrados, o *Sistema GISE* disponibiliza o *Consultor de Integração* que é discutido na seção 4.3.1.5.

Caso o usuário opte por *Compor Campo Integrado*, será apresentada uma tela com informações necessárias para a composição do campo, conforme ilustrado na Figura 42.

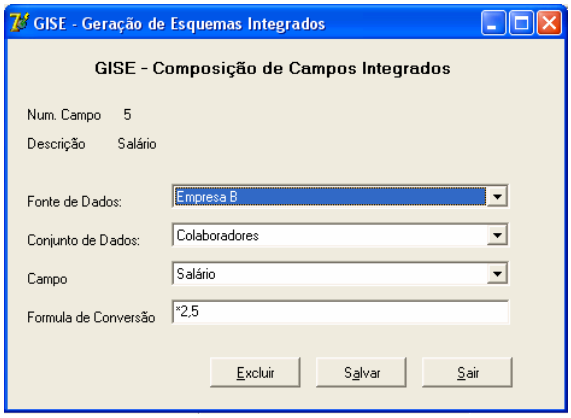


Figura 42: Tela para Composição de Campo Integrado.

Na tela para *Composição de Campo Integrado* (Figura 42) é apresentado ao usuário uma caixa de diálogo com todas as fontes de dados locais disponíveis no *Serviço de Metadados*. Após a escolha por uma fonte de dados, são apresentados na caixa de diálogo *Conjunto de Dados* todos os conjuntos de dados da fonte de dados escolhida. Neste momento o usuário deverá escolher um conjunto de dados e definir qual é o campo local que irá compor o campo integrado. O campo integrado pode ser composto de diversos campos locais, resolvendo desta forma, o problema de conflito de denominação identificado na seção 3.1.1.

Caso o campo integrado seja do tipo número, é possível também resolver o conflito de unidade através da caixa *Fórmula de Conversão*. Para utilizar este recurso o usuário deverá cadastrar uma fórmula de conversão. Por exemplo, se o campo integrado for definido na moeda *Real* na fonte de dados integrada e estiver cadastrado na moeda *Dólar* em alguma fonte de dados local, o usuário deverá informar a expressão "*2,5" na fórmula de conversão no momento da composição com a fonte de dados em *Reais*. Desta forma, é resolvido o problema de conflitos de unidade identificado na seção 3.1.1.

4.3.1.4 Editor de Consultas Integradas

O *Editor de Consultas Integradas* é responsável por fornecer uma interface com o usuário para a consulta de dados distribuídos. As informações necessárias para executar uma consulta de dados distribuídos estão ilustradas na Figura 43.

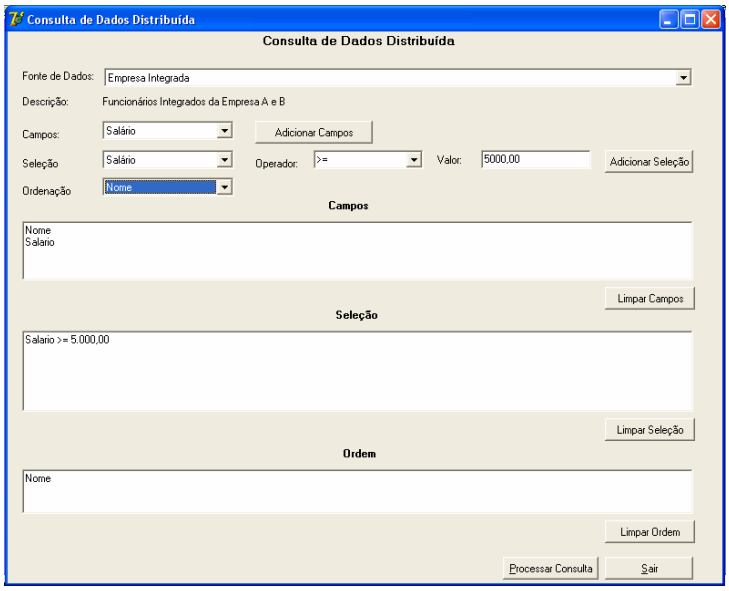


Figura 43: Processador de Consultas Distribuídas.

Para realizar uma consulta distribuída de forma transparente às fontes de dados na Grade Computacional, o usuário deverá informar o Esquema Integrado (*GISE-INTEGRATED-SCHEMA*) que será utilizado, definir quais serão os campos compostos que participará da consulta e pressionar o botão *Adicionar Campos*.

No *Sistema GISE*, é possível fornecer uma condição de consulta para o retorno dos dados. Para utilizar este recurso são disponibilizados: i) os campos seleção (que deve ser preenchido com os campos do Esquema Integrado); ii) operador (com as opções: =, <>, <, <=, >, >=, like); e iii) valor. Desta forma, é possível realizar um filtro no retorno das informações obtidas nas fontes de dados locais de forma integrada.

Finalmente, o *Processador de Consultas Distribuídas* permite que os resultados sejam obtidos em uma ordem específica. Desta forma, é disponibilizado o campo ordenação. O campo ordenação especifica a seqüência de organização dos dados obtidos.

O botão *Processar Consultas* é responsável por receber a solicitação de consulta distribuída e encaminha-lá para o componente *Fragmentador de Operações* da *Camada Serviços de Integração (Mediador)* da *Arquitetura GISE*.

A seguir, serão discutidos aspectos referentes ao *Consultor de Integração* e suas funcionalidades.

4.3.1.5 Consultor de Integração

O *Consultor de Integração* é o componente responsável por auxiliar os usuários no processo de confecção de esquemas integrados (*GISE-INTEGRATED-SCHEMA*). O *Consultor de Integração* é composto pelo componente *Dicionário de Sinônimos* e pelo componente *Pesquisa de Similaridade Semântica*. A forma de funcionamento destes componentes é descrita a seguir:

Dicionários de Sinônimos

O *Dicionário de Sinônimos* é uma ferramenta que auxilia o usuário no processo de integração de fontes de dados. O *Dicionário de Sinônimos* é inicialmente composto por um *Cadastro de Termos*. Os termos e seus sinônimos são utilizados como base de pesquisa nos esquemas locais na tentativa de encontrar possibilidades de integração entre os campos do Esquema Integrado e os campos do esquema base. O *Cadastro de Termos* está ilustrado na Figura 44.

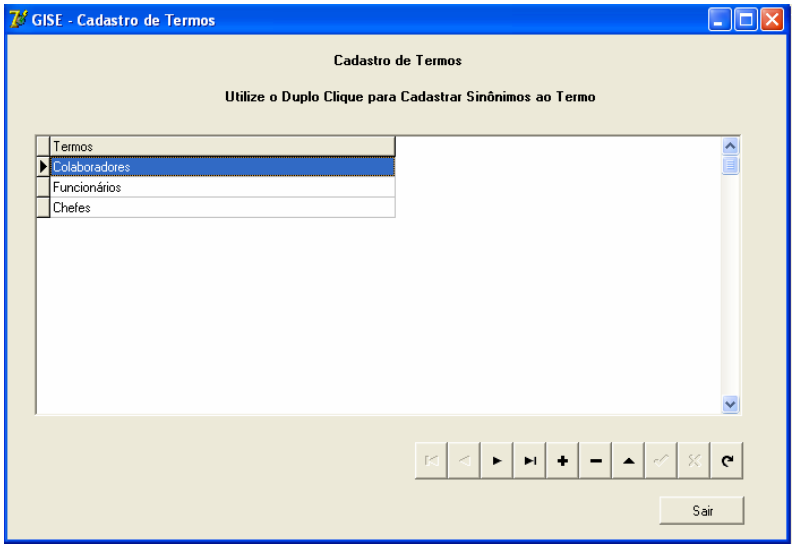


Figura 44: Cadastro de Termos.

Para a definição, consulta ou exclusão de um sinônimo ao termo, basta que o usuário efetue um duplo clique sobre algum termo do dicionário e informe a operação desejada, conforme ilustrado na Figura 45.

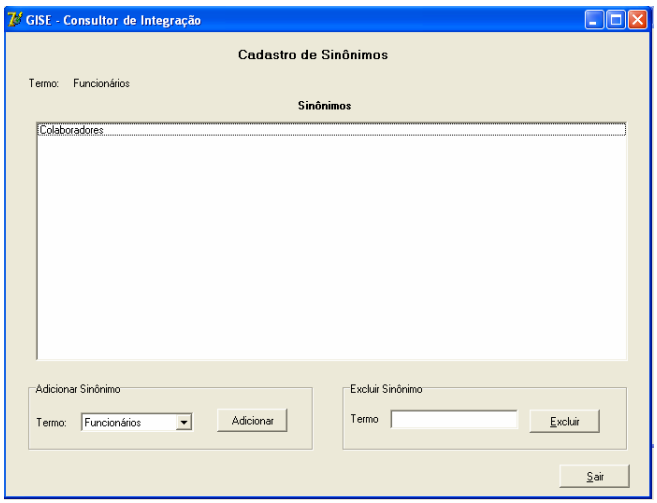


Figura 45: Cadastro de Sinônimos.

As informações do *Dicionário de Sinônimos* são utilizadas no processo de geração de campos integrados, onde é disponibilizado o botão *Dicionário de Sinônimos*, conforme ilustrado na Figura 41. Caso o usuário opte pela ajuda do *Dicionário de Sinônimos*, será

primeiramente executada uma busca dos termos da descrição do campo integrado no *Dicionário de Termos*. Caso o resultado da pesquisa retorne um valor diferente de nulo, serão pesquisados no *Dicionário de Sinônimos* os itens de sinônimos correspondentes ao termo pesquisado.

Quando o *Dicionário de Sinônimos* obtém todos os sinônimos da descrição do campo integrado é feita uma pesquisa nos esquemas locais para verificar a existência de algum item correspondente. Desta forma, o processo de integração de fontes de dados é facilitado. A estrutura das informações disponibilizadas pelo *Dicionário de Sinônimos* está ilustrada na Figura 46.

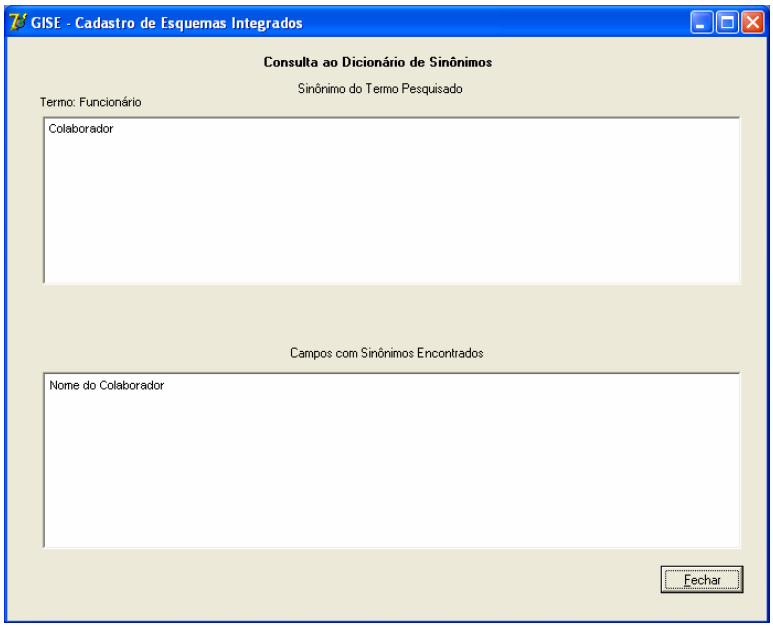


Figura 46: Atuação do Dicionário de Sinônimos.

O *Dicionário de Sinônimos* pode receber informações específicas de um determinado domínio. Desta forma, teremos um dicionário especializado que pode muito facilitar o processo de integração de fontes de dados.

Pesquisa de Similaridade Semântica

A *Pesquisa de Similaridade Semântica* é a segunda ferramenta do *Consultor de Integração*. Esta ferramenta pode localizar as similaridades das informações recebidas na descrição do campo integrado com as informações armazenadas na descrição de campos nos esquemas locais. A *Pesquisa de Similaridade Semântica* deve ser iniciada pela tela *Cadastro de Campos Integrados* (Figura 41) e retornará as informações que estão ilustradas na Figura 47.

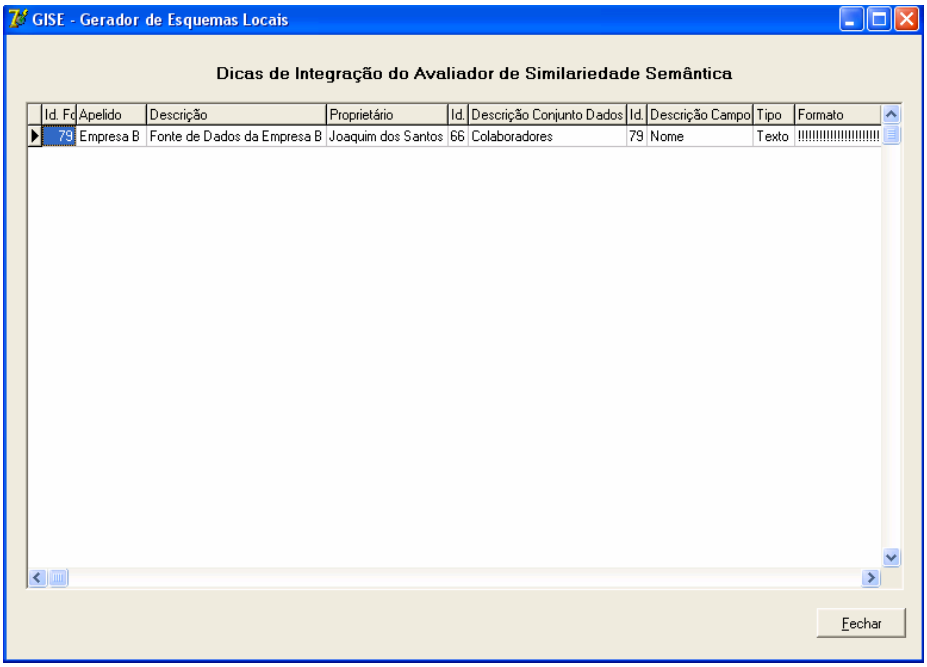


Figura 47: Informações Retornadas pela Pesquisa de Similaridade Semântica.

A *Pesquisa de Similaridade Semântica* em seu funcionamento executa uma busca dos termos informados pelo usuário na descrição do campo integrado. Após receber os termos, a *Pesquisa de Similaridade Semântica* faz uma busca na descrição de campos em todos os esquemas locais. Por exemplo, se possuirmos cadastrado um campo integrado com a descrição nome, a *Pesquisa de Similaridade Semântica* retornará como resultado todos os campos dos esquemas locais cadastrados cujas descrições contêm a palavra nome.

4.3.2 Camada Serviços de Metadados

Na Grade Computacional existe uma grande heterogeneidade quando nos referimos as fontes de dados disponíveis para acesso e consulta pelos usuários. Portanto, para permitir um processo de integração transparente e eficaz é de vital importância o armazenamento de informações referentes às fontes de dados locais, tais como, estrutura, forma de acesso, conjunto de dados, entre outros.

Para que seja possível realizar operações de consultas em fontes de dados integradas, é necessário que o *Serviço de Metadados* também armazene informações sobre os esquemas integrados. Desta forma, a definição dos dados que serão armazenados pelo *Serviço de Metadados* é o ponto inicial no processo de integração de fontes de dados.

Na arquitetura de *Mediadores* o *Serviço de Metadados* disponibiliza informações sobre as fontes de dados locais e integradas diretamente aos *Mediadores* que acessam, através de seus *Tradutores* específicos, as fontes de dados heterogêneas envolvidas no processo de integração.

A *Camada de Serviços de Metadados* da *Arquitetura GISE* foi implementada na Grade Computacional através da utilização de um serviço próprio de *Metadados* denominado GISE-MCA *(Grid Integration Service – Metadata CAtalog*). O GISE-MCA disponibiliza ao *Sistema GISE* ou a qualquer aplicação cliente, informações sobre as características e atributos das fontes de dados, esquemas locais, esquemas integrados e os dados necessários para a *Pesquisa de Similaridade Semântica*.

A estrutura de metadados da *Arquitetura GISE* é projetada para a representação de esquemas locais e esquemas integrados de fontes de dados. Esta abordagem é necessária, pois, através desta integração, o *Sistema GISE* garante que não existam esquemas integrados que referenciam elementos inexistentes em esquemas bases. O esquema de metadados relacionado

ao mapeamento de esquemas das bases de dados na *Arquitetura GISE* está ilustrada na Figura 48 e são armazenados como documentos XML.

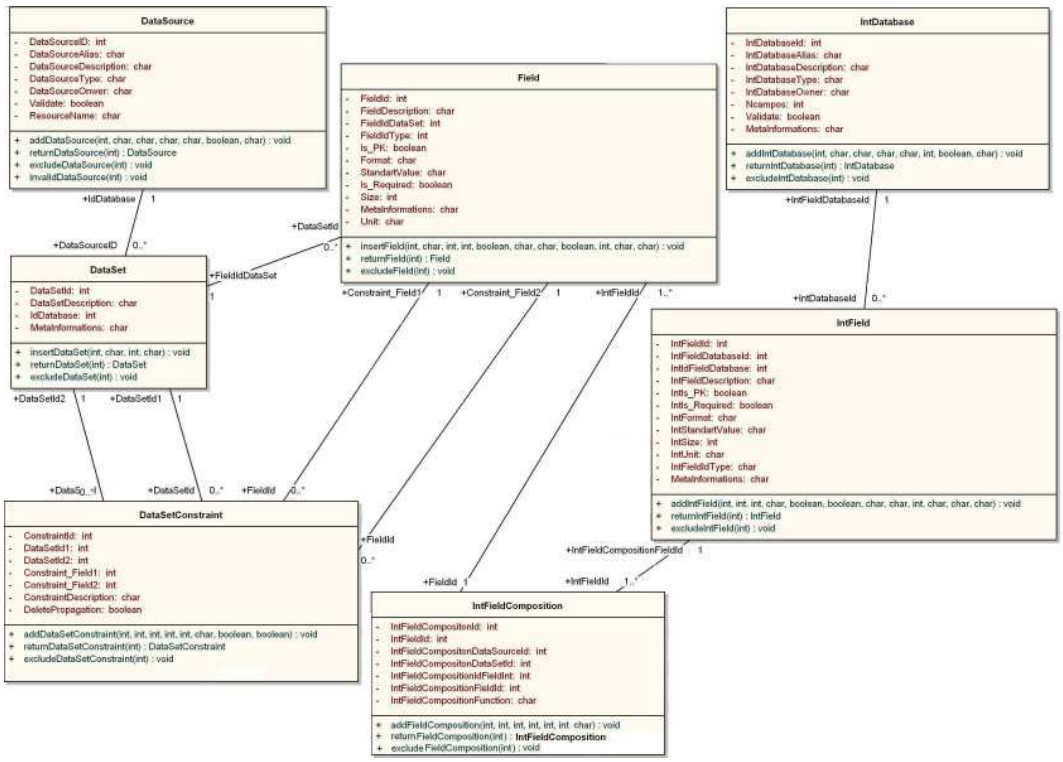


Figura 48: Estrutura de Metadados de Esquemas Locais e Integrados da Arquitetura GISE.

Conforme ilustrado na Figura 48, para que seja possível ao *Serviço de Metadados* representar os esquemas locais e esquemas integrados das fontes de dados, são utilizadas sete classes distintas. Destas, as classes DataSource, DataSet, DataSetConstraint e Field são utilizadas para representar esquemas locais e as classes IntDatabase, InfField, IntFieldComposition e Field são utilizadas para representar esquemas integrados. A classe Field é a responsável para representar os atributos locais disponíveis para integração e também realizar o mapeamento dos atributos virtuais compostos do Esquema Integrado para um atributo concreto do Esquema Local.

A classe DataSource representa uma fonte de dados local. Na *Arquitetura GISE*, uma fonte de dados local terá nenhum ou vários conjuntos de dados ligados a um DataSource. A classe DataSet é responsável pelo armazenamento das informações sobre os conjuntos de dados disponíveis no DataSource.

A classe Field armazena informações sobre os atributos pertencentes às fontes de dados mapeadas. Nesta classe, incluem-se informações sobre Tipo, Formato e Unidade dos atributos locais. Desta forma, estas informações são disponibilizadas aos componentes *Fragmentados de Operações* e *Desfragmentados de Resultados* no processo de integração de fontes de dados.

Para a representação das fontes de dados integradas, o *Serviço de Metadados* utiliza inicialmente a classe IntDatabase. Esta classe representa uma fonte de dados integrada e virtual que poderá ser utilizada para o processamento de consultas de forma transparente aos usuários da *Arquitetura GISE.*

Diferentemente das fontes de dados locais, a fonte de dados integrada não possui mais que um conjunto de dados. A fonte de dados integrada é composta por um conjunto de atributos virtuais identificados pela classe IntField e estes, por sua vez, são compostos por elementos da classe IntFieldComposition. Os elementos da classe IntFieldComposition são utilizados para representar a composição dos atributos virtuais da fonte de dados integrada em atributos locais.

Para realizar a composição dos atributos virtuais da fonte de dados integrada, a classe IntFieldComposition é composta de um conjunto de elementos da classe Field (classe utilizada para a representação de atributos locais). Desta forma, o GISE-MCA disponibiliza as informações necessárias aos componentes das camadas da *Arquitetura GISE,* fornecendo suporte claro e preciso ao processo de integração de fonte de dados.

Para o armazenamento de dados referentes ao *Dicionário de Sinônimos*, a *Arquitetura GISE* faz uso do mesmo *Serviço de Metadados*. A Figura 49 ilustra a estrutura de metadados do *Dicionário de Sinônimos*.

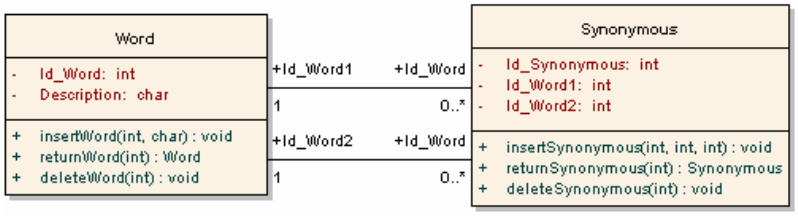


Figura 49: Estrutura de Metadados do Dicionário de Sinônimos.

Conforme ilustrado na Figura 49, para a representação do *Dicionário de Sinônimos* o GISE-MCA utiliza duas classes. A classe Word é utilizada para representar uma palavra ou conjunto de palavras e a classe Synonymous é responsável por armazenar informações sobre o relacionamento dos objetos da classe Word.

Para que seja possível prover funcionalidades de integração de fontes de dados aos clientes da arquitetura, é necessária uma camada de serviços com componentes especializados para realizarem as tarefas de integração. Esta camada é denominada na *Arquitetura GISE* de *Serviços de Integração* (Mediador). Na próxima seção serão discutidos seus aspectos, componentes e características.

4.3.3 Camada Serviços de Integração (Mediador)

A *Camada de Serviços de Integração* é o *Mediador* da *Arquitetura GISE*. O *Mediador* é responsável pela integração semântica e resolução de conflitos nas fontes de dados integradas. Os componentes do *Mediador* da *Arquitetura GISE* são descritos a seguir:

4.3.3.1 Fragmentador de Operações

O *Fragmentador de Operações* é o componente da *Camada Serviços de Integração* (Mediador) responsável por receber as requisições de operações da *Camada de Serviços Cliente*. As requisições de operações podem ser enviadas pelo *Sistema GISE* ou por qualquer outra aplicação cliente. O *Fragmentador de Operações* divide as requisições por fonte de dados com base nas informações recebidas pelo *Serviço de Metadados*.

O *Fragmentador de Operações* recebe as requisições de operações a serem executadas nos esquemas integrados (*GISE-INTEGRATED-SCHEMA*) e as decompõe. Desta forma, é gerado um conjunto de operações divididas por fonte de dados. O *Fragmentador de Operações* é executado pelo acionamento do botão *Processar Consulta* na tela de criação de consultas integradas, conforme ilustrado na Figura 43.

Inicialmente o *Fragmentador de Operações* identifica quais são as fontes de dados que serão utilizadas para processamento da requisição integrada. Após a identificação, é criada uma matriz onde o número de linhas é igual ao número de fontes de dados envolvidas na integração e o número de colunas é igual ao número de campos envolvidos por cada fonte de dados.

O próximo passo do *Fragmentador de Operações* é identificar a operação integrada que será realizada e seus parâmetros. O *Fragmentador de Operações* se responsabiliza por gerar um conjunto de operações locais, separadas por fontes de dados. As requisições locais são armazenadas em *Documentos de Atividade* (*Activity Documents*).

Os *Documentos de Atividade* armazenam um conjunto de operações decompostas para serem executadas nas fontes de dados locais no formato SQL (*Structured Query Language*). Os *Documentos de Atividade* são processados pela *Camada de Serviço de Acesso aos Dados*. A Figura 50 ilustra o funcionamento do *Fragmentador de Operações.*

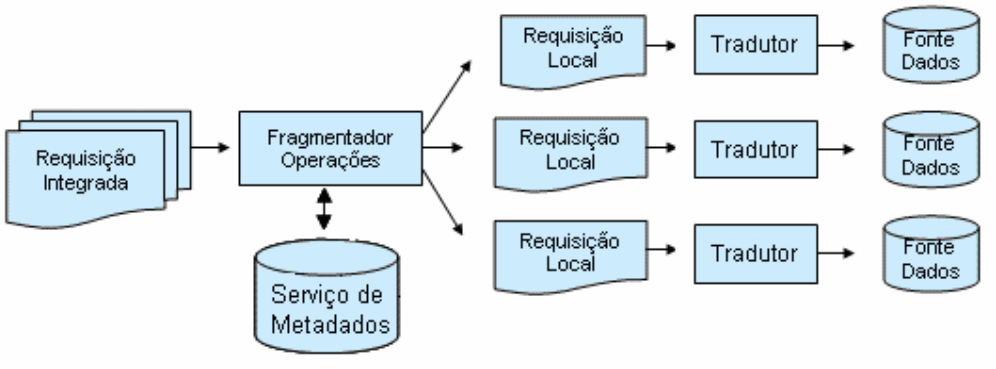


Figura 50: Funcionamento do Fragmentador de Operações.

4.3.3.2 Conversor Estrutural

O *Conversor Estrutural* é o componente da *Camada Serviços de Integração* (Mediador) que resolve os conflitos de tipo e formato que podem ocorrer no processo de integração de fontes de dados. O *Conversor Estrutural* fornece suporte adequado para que o *Desfragmentador de Resultados* retorne ao usuário da *Arquitetura GISE* ou aplicação cliente resultados válidos.

4.3.3.3 Conversor de Formato e Tipo

O *Conversor de Formato e Tipo* é o componente da *Camada Serviços de Integração* (Mediador) responsável pela conversão dos dados retornados pelo *Desfragmentador de Resultados* em padrões únicos.

A conversão é realizada consultando as informações sobre os esquemas locais e esquemas integrados no *Serviço de Metadados*. Desta forma, são resolvidos os conflitos de formato e tipo identificados.

4.3.3.4 Desfragmentador de Resultados

O *Desfragmentador de Resultados* é o componente utilizado no processo de consulta de dados distribuída. Este componente tem a função inversa do *Fragmentador de Operações*. Sua responsabilidade é de integrar, de acordo com as informações armazenadas no Esquema Integrado definido pelo usuário (*GISE-INTEGRATED-SCHEMA*), o conjunto de respostas recebidas das consultas locais executadas pelos *Tradutores* na Grade Computacional em uma única resposta. O resultado da consulta integrada pode ser disponibilizado ao usuário pelo *Sistema GISE* ou por uma aplicação cliente, uma vez que o retorno da consulta é um documento XML.

O *Desfragmentador de Resultados* faz uso dos serviços providos pelo *Conversor Estrutural* para resolver conflitos de tipo e formato decorrentes do processo de integração semântica de fonte de dados.

A estrutura do componente *Desfragmentador de Resultados* está ilustrada na Figura 51.

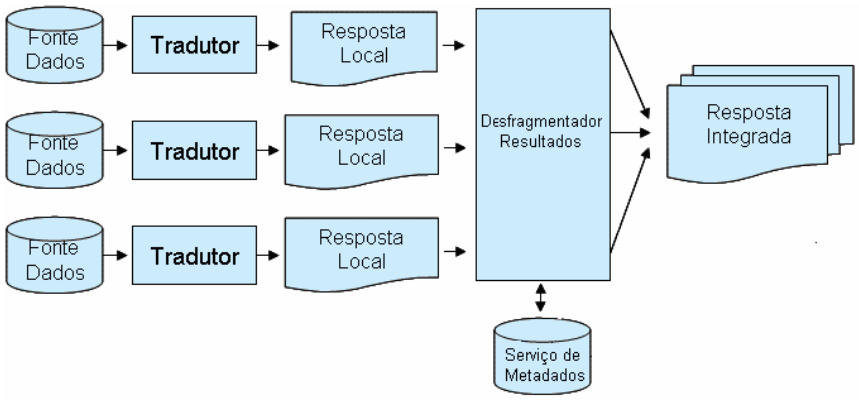


Figura 51: Funcionamento do Desfragmentador de Resultados.

4.3.4 Camada Serviços do Globus Toolkit

A *Camada de Serviços do Globus Toolkit* é responsável for oferecer os serviços necessários para utilização da Grade Computacional pela *Arquitetura GISE*. Os serviços desta camada são gerenciados pelo *Globus Toolkit* e estão descritos a seguir:

4.3.4.1 Infra-Estrutura de Segurança (GSI)

Para impossibilitar o acesso não autorizado na *Arquitetura GISE*, são utilizados como infra-estrutura de segurança os mesmos mecanismos implementados no GT4 através do GSI (*Globus Infrastructure Security*). O GSI disponibiliza mecanismos de segurança ao nível de mensagens ou ao nível de transporte.

O GSI implementa a segurança da Grade Computacional com componentes baseados em credenciais e protocolos padrões para a proteção de mensagens, autenticação, delegação e autorização. Conforme ilustrado na Figura 52, o suporte é provido para compilação WS-Security em nível de troca de mensagens com *credenciais X.509* (primeiro bloco) ou com *usernames* e *passwords* (segundo bloco). Para a segurança em nível de transporte, são utilizadas as *credenciais X.509* (terceiro bloco) (WELCH, 2005).

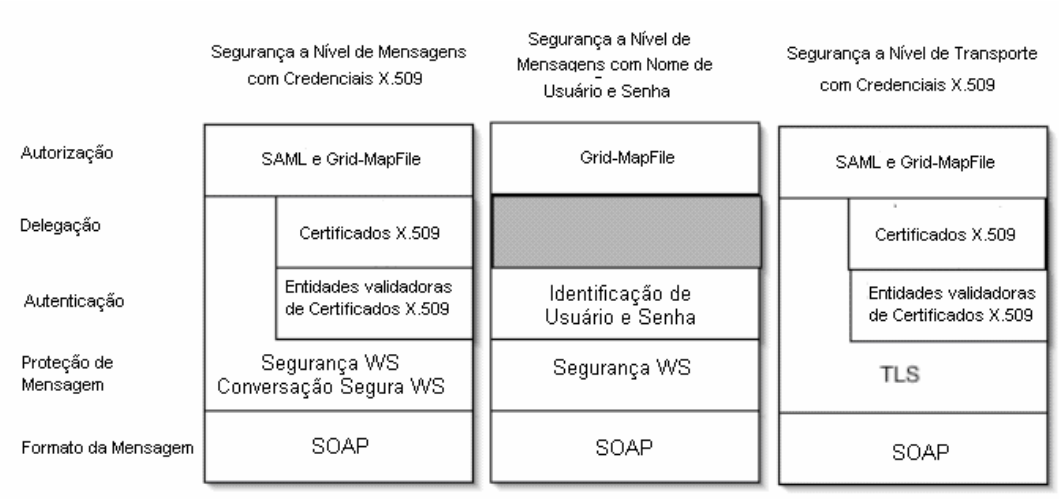


Figura 52: Infra-Estrutura de Segurança do GT4 (WELCH, 2005).

No GT4 (configuração padrão), cada usuário e recurso têm uma credencial pública X.509. Os protocolos são implementados para permitir que entidades validem outras credenciais. As credenciais são utilizadas para estabelecer um canal seguro para troca de mensagens (WELCH, 2005).

As chamadas de autorização podem estar associadas com serviços GT4 para determinar se as requisições são permitidas. Os componentes de autorização permitem encadear os módulos de autorização com as interfaces (WELCH, 2005).

Para que o usuário tenha acesso às informações contidas em uma fonte de dados, a mesma deverá contar com a identidade *Globus* do usuário cadastrada em sua lista de controle de acesso. Desta forma, é garantido que apenas pessoas com acessos claramente definidos possam fazer uso dos serviços e dados disponibilizados na Grade Computacional e consequentemente na *Arquitetura GISE*.

4.3.4.2 Serviços de Informações (MDS4)

O *Serviço de Informações* (*MDS – Monitoring and Discovery Service*) do GT4 é serviço responsável por fornecer informações para as camadas da *Arquitetura GISE* sobre os serviços (*Grid Services*) disponíveis para uso na Grade Computacional. O MDS4 disponibiliza informações como: recursos, localização, propriedade, custo, carga, entre outras. O MDS obtém as informações distribuídas disponibilizadas em diferentes recursos ou outros serviços na Grade Computacional e as disponibiliza aos usuários. O MDS, além de representar as informações sobre os recursos e serviços na Grade Computacional, fornece interfaces para acesso e disseminação destas informações (CZAJKOWSKI et al., 2001).

A *Arquitetura GISE* utiliza o *MDS-Index* como um serviço agregador que se responsabiliza por captar as informações nas fontes de dados locais distribuídas na Grade Computacional e o *WebMDS* para oferecer uma interface de consulta as informações captadas pelo *MDS-Index*.

4.3.5 Camada Serviços de Acesso aos Dados

Para que a *Arquitetura GISE* disponibilize acessos transparentes e integrados as diversas fontes de dados heterogêneas, é necessária a utilização de uma camada de serviços capaz de disponibilizar uma estrutura de acesso às múltiplas fontes de dados.

Na *Arquitetura GISE*, este trabalho é realizado pela *Camada de Serviços de Acesso aos Dados*. Nesta camada são implementados componentes baseados na especificação WSRF que disponibilizam acesso às fontes de dados na Grade Computacional.

A *Camada de Serviço de Acesso a Dados* é composta pelos serviços da arquitetura do OGSA-DAI. Os serviços do OGSA-DAI utilizados pela *Arquitetura GISE* são discutidos a seguir:

4.3.5.1 OGSA-DAI

Na *Arquitetura GISE* o OGSA-DAI é o componente responsável por prover a transparência de heterogeneidade no acesso aos dados. Ele atua como um *middleware* permitindo que as fontes de dados implementadas através de diferentes tecnologias sejam acessadas na Grade Computacional. O OGSA-DAI possui em sua arquitetura um serviço que informa quais são as fontes de dados que estão disponíveis para receber operações denominado *Data Service Resource*.

O OGSA-DAI também se responsabiliza, na *Arquitetura GISE*, pelo gerenciamento do ciclo de vida do conjunto de instâncias de *Grid Data Services* (Tradutores) necessários para o acesso às fontes de dados locais.

Por fim, o OGSA-DAI se responsabiliza por receber os resultados em XML das operações locais efetuadas pelos *Grid Data Services*. A quantidade de arquivos XML gerada pelo OGSA-DAI é dependente do número de fontes de dados acessadas no processamento da requisição integrada, ou seja, para cada fonte de dados local é criado um arquivo XML com os resultados locais.

4.3.5.2 *Data Service Resource*

O *Grid Data Service Resource* é um recurso utilizado pelo OGSA-DAI e pela *Arquitetura GISE* para disponibilizar informações sobre quais as fontes de dados que estão disponíveis para receber requisições locais e quais são os *Grid Data Service* (Tradutores) que deverão ser utilizados para realizar o acesso a uma fonte de dados específica.

O *Grid Data Service Resource* é um serviço persistente na *Arquitetura GISE* e sua distribuição é nativa no GT4.

4.3.5.3 *Grid Data Service* (Tradutores)

O *Grid Data Service* é um serviço transiente na *Arquitetura GISE* e seu tempo de vida é gerenciado pelo OGSA-DAI. O *Grid Data Service* também é conhecido como *Tradutor* ou *Wrapper*. O objetivo de um *Grid Data Service* é acessar uma fonte de dados específica (*Data Resource File*) e processar uma requisição local. Após obter a resposta da requisição local, o *Grid Data Service* a envia para o OGSA-DAI e é automaticamente destruído, liberando seus recursos computacionais alocados para a execução de novas tarefas.

Caso exista a necessidade de novamente utilizar um *Grid Data Service*, o OGSA-DAI se encarregará de criar uma nova instância para o mesmo e o GRAM será responsável por alocar os recursos computacionais necessários para sua execução.

4.3.6 Camada Fontes de Dados (Data Resource Files)

A *Camada de Fonte de Dados* (*Data Resource Files*) é a última camada da *Arquitetura GISE*. Esta é a camada onde residem as estruturas de dados locais disponíveis para integração na Grade Computacional. Nesta camada, a heterogeneidade entre as diversas tecnologias utilizadas para implementação de fontes de dados é evidente.

A *Camada de Fontes de Dados* é estática, ou seja, nela não são implementados serviços de utilização da *Arquitetura GISE*. Esta camada apenas armazena os repositórios de fontes de dados que são acessados pelos *Grid Data Services* (Tradutores).

Capítulo 5 – Questões de Implementação de Estudo de Caso

Este capítulo tem como objetivo apresentar um estudo de caso no qual um protótipo da *Arquitetura GISE* foi implementado para validar suas funcionalidades. Neste estudo de caso também é realizada uma pequena exposição sobre as questões de implementação e algoritmos utilizados pelo Mediador da *Arquitetura GISE.* Este capítulo é finalizado com a apresentação do processamento de uma consulta integrada.

5.2 Concepção GISE

A *Arquitetura GISE* foi implementada para ser uma arquitetura multi-plataforma e facilmente extensível. Desta forma, foram utilizados em sua implementação tecnologias padrões e ferramentas de software de código aberto ou de livre utilização.

A *Arquitetura GISE* foi construída em dois estágios. Inicialmente foram realizadas as tarefas de análise, projeto, implementação, testes e correções finais para o *Serviço de Metadados* e um *Cliente*.

A implementação do Mediador foi o estágio seguinte na construção da *Arquitetura GISE*. Esta ordenação na implementação das camadas foi necessária, pois não seria possível desenvolver um Mediador eficaz sem um *Serviço de Metadados* e um *Cliente* em perfeito funcionamento.

O *Serviço de Metadados* da *Arquitetura GISE* armazena através de documentos XML os esquemas locais e esquemas integrados que são utilizados pela *Camada de Serviços de Integração* (Mediador) e *Camada Cliente*. Na implementação do *Serviço de Metadados* (GISE-MCA), foi utilizado *PostgreSQL* versão 8.0. A estrutura do *Serviço de Metadados* está ilustrada na Tabela 2:

Esquemas Locais:	IdEsquemaLocal (PK_AutoIncrement)	Descrição (VarChar)	Arquivo (VarChar)
Esquemas Integrados:	IdEsquemaIntegrado (PK_AutoIncrement)	Descrição (VarChar)	Arquivo (VarChar)

Tabela 2: Estrutura de Metadados utilizada no PostgreSQL 8.0

Para o desenvolvimento das interfaces do *Cliente* foi utilizado o *Borland Delphi 7.0* e para a implementação do Mediador da arquitetura foi utilizada a linguagem *JAVA*, da Sun. A Figura 53 ilustra a *Arquitetura GISE* e as ferramentas utilizadas em seu processo de implementação:

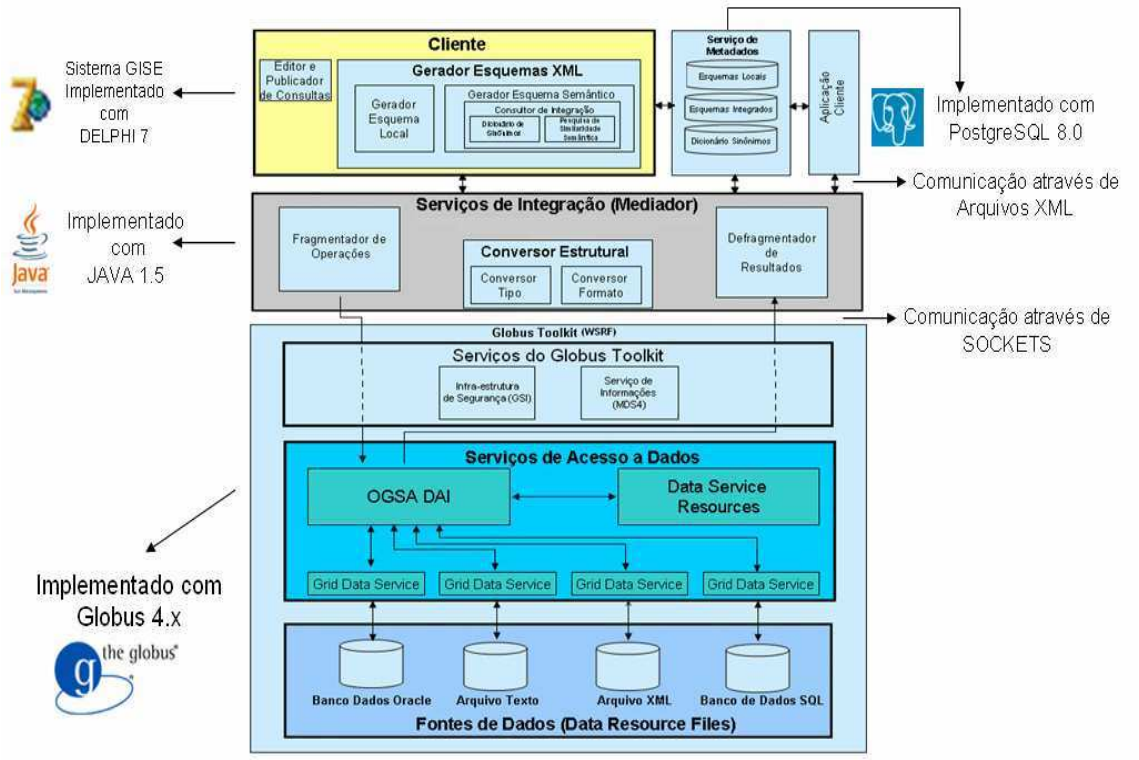


Figura 53: Ferramentas de Implementação da Arquitetura GISE.

A implementação através de camadas da *Arquitetura GISE* permite com que cada camada seja facilmente substituída ou reutilizada em outros projetos.

Para disponibilizar a comunicação entre as *Camadas Cliente* e *Camada Serviços de Integração* (Mediador) se faz necessário a utilização de mecanismos padrões de comunicação. Desta forma, foi construído através da linguagem XML um esquema padrão para os arquivos de comunicação (arquivos estes que são criados pela *Camada Cliente* e são processados pela *Camada de Serviços de Integração* - Mediador). A Figura 54 ilustra o esquema de comunicação entre a *Camada Cliente* e a *Camada de Serviços de Integração* (Mediador).

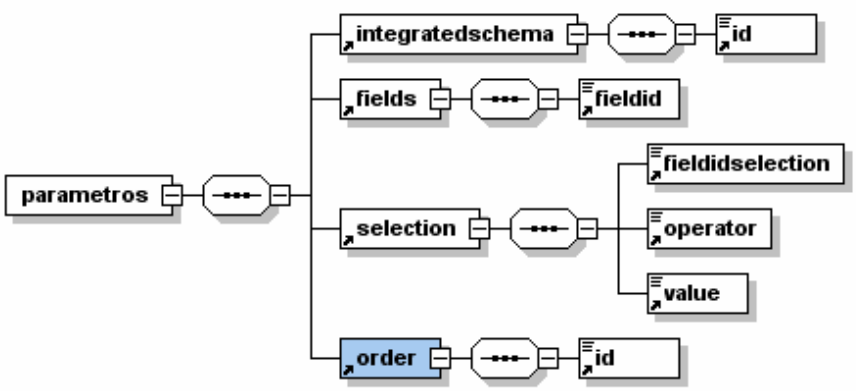


Figura 54: Esquema do Arquivo de Comunicação entre as Camadas Cliente e Serviços de Integração (Mediador).

Para a construção da *Camada de Serviços do Globus Toolkit* foi utilizado o GT4. O GT4 é até o presente momento (junho/2006) a última versão disponível do *Globus Toolkit* e já disponibiliza em seu conjunto de serviços o OGSA-DAI. O OGSA DAI foi o *middleware* utilizado para disponibilizar acesso às fontes de dados na Grade Computacional.

Para implementar a *Camada de Serviços de Integração* (Mediador) foi necessário o desenvolvimento de algoritmos para resolverem os conflitos decorrentes do processo de integração de fontes de dados na seção 3.1.1. Os algoritmos implementados serão discutidos a seguir.

5.2.1 Algoritmos Implementados

As camadas da *Arquitetura GISE* possuem componentes de uso comum que são amplamente utilizados em múltiplos domínios, como é o caso do *Globus Toolkit* e o OGSA-DAI. Porém, para a realização deste trabalho foi necessário a implementação de um Mediador que complementa as funcionalidades dos componentes já citados, como o *Fragmentador de Operações*, *Desfragmentador de Resultados* e o *Conversor de Formato*.

O *Conversor de Tipo* e o *Conversor de Unidade* não necessitam a implementação de algoritmos individuais. A conversão de tipos é resolvida com a utilização da linguagem SQL para comunicação com as fontes de dados pelo OGSA-DAI. Na linguagem SQL não existe distinção entre tipos de atributos no retorno de uma consulta.

A conversão de unidades é resolvida pela utilização do SQL em conjunto com as informações armazenadas no Esquema Integrado. A *Arquitetura GISE* realiza uma requisição de retorno de atributos com processamento de dados, como por exemplo: *SELECT NúmeroSalariosMínimos*340 FROM Pacientes.* Os passos executados pelo *Fragmentador de Operações* e *Desfragmentador de Resultados* estão ilustrados na Tabela 3.

Passo	Ação	
1	Ler o arquivo de comunicação do Cliente;	**Fragmentador de Operações**
2	Buscar o valor do elemento IntegratedSchemaId;	
3	Criar um array com os campos integrados (array 1);	
4	Caso Necessário:	
5	Criar um array com os elementos: FieldSelection, Operator e Value;	
6	Buscar o valor do elemento Order;	
7	Solicitar ao serviço de metadados o Esquema Integrado (IntegratedSchemaId);	
8	Para cada elemento do array de campos integrados:	
9	Criar um array de valores não duplicados com o elemento DataSourceId (array 2);	
10	Criar um array (id fonte / comando) de comandos para cada fonte de dados (array 3);	

11	Solicitar ao serviço de metadados os esquemas locais do array DataSourceID (array 2);	
12	Para cada elemento do array de campos integrados (array 1);	
13	Criar um array com os elementos do campo integrado (array 4);	
14	Para cada elemento do array do campo integrado (array 4):	
15	Gravar os elementos: DataSourceId, DataSetId, FieldId e Function;	
16	Buscar no Esquema Local a descrição do campo;	
17	Aplicar a função de composição;	
18	Atulizar o array de comandos da fonte de dados (array 3);	
19	Processar os comandos gravados no array 3;	**OGSA – DAI**
20	Verificar a quantidade de elementos retornados por fonte de dados;	**Desfragmentador de Resultados**
21	Processar o array de decomposição de campos (array 4) e gerar o arquivo XML Integrado por Atributo;	
22	Converter o arquivo gerado de atributo para elemento;	
23	Ordenar os resultados;	
24	Gerar os arquivos de consulta (Genérico e DELPHI);	

Tabela 3: Processamento Executado pelo Fragmentador de Operações e Desfragmentador de Resultados

Na *Arquitetura GISE* o *Fragmentador de Operações* e o *Desfragmentador de Resultados* executam seu processamento em seqüência, mas são implementados através de componentes distintos. Desta forma, podemos utilizar o OGSA-DAI para realizar o acesso às fontes de dados após o término do processamento do *Fragmentador de Operações* e iniciar o *Desfragmentador de Resultados* após o processamento das tarefas do OGSA-DAI.

A implementação através de componentes da *Camada Serviços de Integração* (Mediador) também pode oferecer maior desempenho na tarefa de integração de fonte de dados. Por exemplo: podemos iniciar o processamento de uma consulta integrada até o passo 18 (ilustrado na Tabela 3) sem que as fontes de dados locais estejam disponíveis para uso. Isso é facilmente possível, uma vez que para realizar a *Fragmentação de Operações* somente

é necessário um *Cliente*, o *Serviço de Metadados* e o próprio *Fragmentador de Operações*. O usuário então poderá aguardar e, quando as fontes de dados estiverem disponíveis para uso, somente é necessário iniciar o processamento da consulta integrada partir do passo 19 (ilustrado na Tabela 3).

O *Desfragmentador de Resultados* está implementado para oferecer dois formatos de saída dos resultados gerados, ambos em XML. A primeira é um formato genérico que pode ser lido por humanos ou por qualquer aplicação cliente, conforme ilustrado na Listagem 1 e a segunda é um formato de retorno específico para ser utilizado no componente *DataGrid* oferecido pela linguagem de programação *Borland Delphi 7.0*, conforme ilustrado na Listagem 2.

```
<?xml version='1.0'?>
        <fonteintegrada>
                <metadados>
                        <campos>
                                <identificacao>1</identificacao>
                                 <descricao>NUMERO DO ATENDIMENTO</descricao>
                                 <chaveprimaria>Sim</chaveprimaria> <formato>99999999999</formato>
                                 <unidade>NA</unidade> <tipo>Numero</tipo>
                                 <MetaData>NUMERO DO PROCEDIMENTO DE ALTA
                                 COMPLEXIDADE</MetaData>
                                 <identificacao>2</identificacao> <descricao>CARTAO NACIONAL DE
                                 SAUDE</descricao> <chaveprimaria>Nao</chaveprimaria>
                                 <formato>999999999999999</formato>
                                 <unidade>NA</unidade> <tipo>Numero</tipo> <MetaData>NUMERO DO
                                 CNS</MetaData> <identificacao>3</identificacao>
                                 <descricao>NOME</descricao> <chaveprimaria>Nao</chaveprimaria>
                                 <formato>@!</formato><unidade>NA</unidade> <tipo>Texto</tipo>
                                 <MetaData>NOME DO PACIENTE</MetaData>
                        </campos>
                </metadados>
                <valores>
                        <linha><coluna>00102316192</coluna><coluna>801434145927414</coluna><colu
                        na>ANTONIO ALMEIDA CARDOSO</coluna></linha>
                        <linha><coluna>00102316214</coluna><coluna>801434145928380</coluna><colu
                        na>FRAGMON DA SILVA</coluna></linha>
                        <linha><coluna>00102316423</coluna><coluna>801434145926566</coluna><colu
                        na>KAHUE AUGUSTO DOS SANTOS SILVA</coluna></linha>
                </valores>
        </fonteintegrada>
```

Listagem 1: Arquivo Genérico de Retorno do Desfragmentador de Resultados

```
<?xml version="1.0" standalone="yes"?>
   <DATAPACKET Version="2.0">
      <METADATA>
         <FIELDS>
            <FIELD attrname="NUMERO DO ATENDIMENTO"  fieldtype="string" WIDTH="15"/>
            <FIELD attrname="CARTAO NACIONAL DE SAUDE"  fieldtype="string" WIDTH="15"/>
            <FIELD attrname="NOME"  fieldtype="string" WIDTH="30"/>
         </FIELDS>
         <PARAMS DEFAULT_ORDER="1" PRIMARY_KEY="1"/></METADATA>
         <ROWDATA>
            <ROW NUMERO DO ATENDIMENTO="00102316192" CARTAO NACIONAL DE
              SAUDE="801434145927414" NOME="ANTONIO ALMEIDA CARDOSO" />
            <ROW NUMERO DO ATENDIMENTO="00102316203" CARTAO NACIONAL DE
              SAUDE="801434145928259" NOME="CASSIO DOS SANTOS ROSA" />
            <ROW NUMERO DO ATENDIMENTO="00102316214" CARTAO NACIONAL DE
              SAUDE="801434145928380" NOME="FRAGMON DA SILVA" />
         </ROWDATA>
   </DATAPACKET>
```

Listagem 2: Arquivo Delphi de Retorno do Desfragmentador de Resultados

O resultado do processamento da Listagem 2 com o componente *DataGrid* do *Borland Delphi 7.0* é implementado no *Sistema Cliente GISE* e está ilustrado na Figura 55.

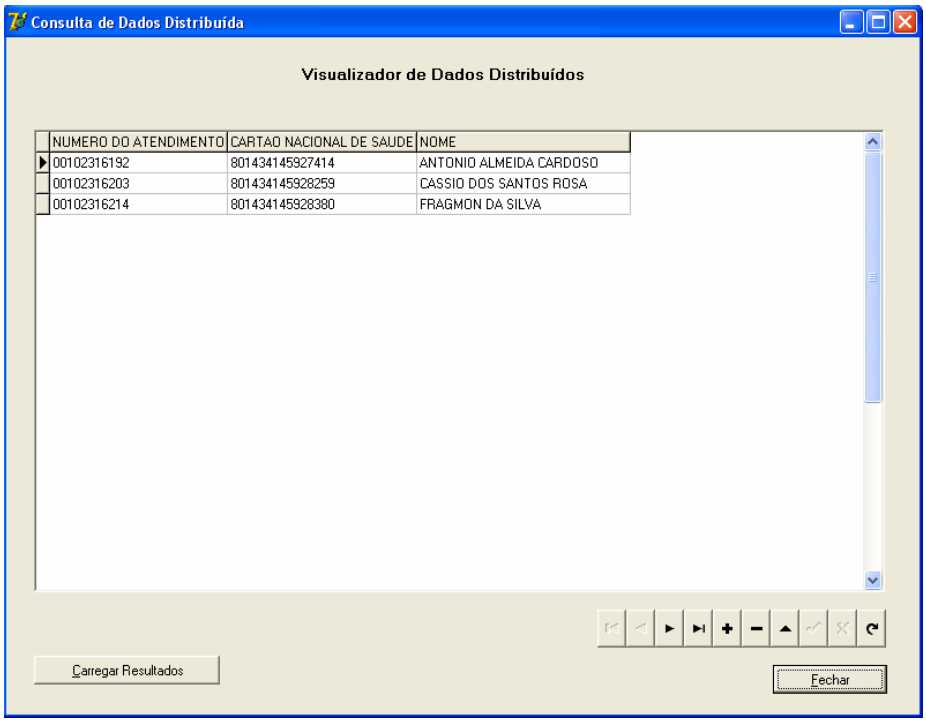


Figura 55: Resultado do Arquivo de Retorno Delphi no Sistema Cliente GISE.

Conversor de Formato

Para que a *Arquitetura GISE* possa realizar a conversão entre os diferentes formatos existentes nas fontes de dados, são armazenadas informações referentes ao formato do campo no Esquema Local.

O *Conversor de Formato* da *Arquitetura GISE* está limitado ao tratamento de campos numéricos, mas pode ser adaptado para o tratamento de múltiplos tipos de dados. Antes de iniciar a conversão de formatos, o componente faz uma análise em busca de eventuais formatações no campo desejado e caso o resultado de sua busca seja diferente de nulo, ele exclui toda a formatação antes de aplicar o novo formato.

O algoritmo do *Conversor de Formato* está ilustrado na Tabela 4.

Passo	Ação
1	Procurar por Formatação Existente;
2	Caso Encontrado:
3	Excluir Formatações;
4	Buscar a Formatação do Campo no Esquema Integrado;
5	Buscar a Quantidade de Elementos do Campo;
6	Faca para cada elemento do campo:
7	Compare elemento no Campo Integrado;
8	Se elemento no Campo Integrado for numérico;
9	Acrescente elemento do campo;
10	Senão
11	Acrescente formatação;
12	Retorne o resultado formatado;

Tabela 4: Algoritmo do Conversor de Formato

5.3 Estudo de Caso

O presente estudo de caso tem por objetivo utilizar a *Arquitetura GISE* e testar seu comportamento em confronto com uma situação no mundo real. Neste estudo de caso serão abordadas as funcionalidades de geração de esquemas locais das fontes de dados, geração de esquemas integrados para a virtualização das fontes de dados e por fim será executada uma consulta distribuída que será processada pelo Mediador da *Arquitetura GISE*.

Este estudo de caso parte da necessidade de se integrar três fontes de dados contendo informações sobre pessoas da comunidade atendidas nas Unidades Básicas de Saúde dos municípios de Mauá, Ribeirão Pires e Rio Grande da Serra. Neste Estudo de Caso, a integração tem como objetivo fornecer suporte para que o Módulo de Detecção de Epidemias do Projeto IntegraEPI possa detectar uma epidemia e simular sua propagação por estas cidades.

As Tabelas 5, 6 e 7 ilustram apenas os atributos das estruturas das fontes de dados que serão integrados. Os demais atributos destas estruturas de dados foram ocultados por não serem utilizadas por este estudo de caso. As Tabelas são respectivamente das UBS de Mauá, Ribeirão Pires e Rio Grande da Serra e seus esquemas locais em XML estão ilustrados nos Apêndices B.2.1, B.2.2 e B.2.3 respectivamente.

Fonte de Dados – Mauá		
Denominação	**Tipo**	**Formato**
ID_Atend	Numérico	99999999999
CNS	Numérico	999999999999999
Nome	Texto	Todos Caracteres
Endereço	Texto	Todos Caracteres
Cód_Munic	Numérico	9999999
CID	Texto	Todos Caracteres
Remuneração	Numérico	99999,99
Procedimen	Numérico	99999999

O número 9 representa um elemento numérico

Tabela 5: Estrutura da Fonte de Dados - UBS Mauá

Fonte de Dados – Ribeirão Pires		
Denominação	**Tipo**	**Formato**
Atendiment	Numérico	99999999999
Paciente	Texto	Todos Caracteres
NCNS	Numérico	999999999999999
Endereço	Texto	Todos Caracteres
Cidade	Numérico	9999999
Salário	Numérico	99,99
ProcSUS	Numérico	99999999
Cód_Doença	Texto	Todos Caracteres

O número 9 representa um elemento numérico

Tabela 6: Estrutura da Fonte de Dados – UBS Ribeirão Pires

Fonte de Dados – Rio Grande da Serra		
Denominação	**Tipo**	**Formato**
Usuário	Texto	Todos Caracteres
NCNS	Numérico	999999999999999
NAPAC	Numérico	99999999999
Endereço	Texto	Todos Caracteres
Cód_Cidade	Numérico	9999999
Salário	Numérico	99.999,99
Cód_Cid	Texto	Todos Caracteres
PSUS	Numérico	99999999

O número 9 representa um elemento numérico

Tabela 7: Estrutura da Fonte de Dados – Rio Grande da Serra

As três fontes de dados ilustradas nas Tabelas 5, 6 e 7 deram origem a uma fonte de dados virtualizada através do processo de integração de fontes de dados disponibilizado pela *Arquitetura GISE*. Neste estudo de caso vale observar as seguintes características nas fontes de dados locais:

- Os campos possuem denominações diferentes em cada fonte de dados; e
- Os campos referentes à remuneração estão armazenados com formatos e unidades diferentes (Ribeirão Pires armazena o número de salários e não o respectivo valor);

Após a definição das fontes de dados envolvidas no processo de integração, se faz necessário a definição da estrutura integrada que servirá como base para a fonte de dados virtualizada. A Tabela 8 ilustra a estrutura integrada utilizada neste estudo de caso que foi denominada *IntegraçãoPacientes* e a Tabela 9 ilustra a composição dos atributos do Esquema Integrado e mapeamento entre os esquemas locais. O Esquema Integrado XML de *IntegraçãoPacientes* com as referências para os esquemas locais está ilustrada no Apêndice B.3.

Fonte de Dados Integrada		
Denominação	**Tipo**	**Formato**
NumeroAtendimento	Numérico	99999999999
Cartão	Numérico	999999999999999
Nome	Texto	Todos Caracteres
Endereço	Texto	Todos Caracteres
Município	Numérico	9999999
CID	Texto	Todos Caracteres
SUS	Numérico	99999999
Remuneração	Numérico	99.999,99

Tabela 8: Estrutura da Fonte de Dados *IntegraçãoPacientes*

Composição e Mapeamento de *IntegraçãoPacientes*									
Denominação	**Composição**								
	Fonte de Dados Mauá			**Fonte de Dados Ribeirão Pires**			**Fonte de Dados Rio Grande da Serra**		
	Denomin.	**T**	**Formato**	**Denomin.**	**T**	**Formato**	**Denomin.**	**T**	**Formato**
NúmeroAtendimento	ID_Atend	N	99999999999	Atendiment	N	99999999999	NAPAC	N	99999999999
Cartão	CNS	N	999999999999999	NCNS	N	999999999999999	NCNS	N	999999999999999
Nome	Nome	T	Todos Caracteres	Paciente	T	Todos Caracteres	Usuário	T	Todos Caracteres
Endereço	Endereço	T	Todos Caracteres	Endereço	T	Todos Caracteres	Endereço	T	Todos Caracteres
Município	Cód_Munic	N	9999999	Cidade	N	9999999	Cód_Cidade	N	9999999
CID	CID	T	Todos Caracteres	Cód_Doença	T	Todos Caracteres	Cód_Cid	T	Todos Caracteres
SUS	Procedimen	N	99999999	ProcSUS	N	99999999	PSUS	N	99999999
Remuneração	Remuneração	N	99999,99	Salário	N	99,99	Saláio	N	99.999,99

Tabela 9: Composição e Mapeamento de *IntegraçãoPacientes*

A seguir, serão ilustradas as gerações dos esquemas locais e esquemas integrados necessários para a integração das fontes de dados citadas nas Tabelas 5, 6 e 7 e ilustrados através de arquivos XML nos Apêndices B.2.1, B.2.2 e B.2.3 respectivamente.

5.3.1 Geração de Esquemas Locais

Uma vez definida qual o conjunto de fontes de dados locais que serão integradas pela *Arquitetura GISE*, é necessária à confecção dos esquemas locais destas fontes de dados e em seqüência a atualização do *Serviço de Metadados* com estas novas informações. Para isso, o usuário pode utilizar a Ferramenta de *Geração de Esquemas Locais* do *Sistema Cliente*. Para utilizar esta ferramenta, o usuário deve acessá – la através do *Menu Gerador de Esquemas* e a opção *Esquemas Locais*, conforme ilustrado na Figura 30.

O processo para geração de esquemas locais é executado uma única vez para cada fonte de dados local e, através deste, o usuário disponibiliza informações sobre a modelo da fonte de dados local que será posteriormente integrada. O processo para geração de esquemas locais está descrito no Capítulo 4 e ilustrado nas Figuras 33 até 38.

Os esquemas locais gerados pela *Arquitetura GISE* são muito extensos. Desta forma, os mesmos podem ser encontrados no Apêndice B.2.1 para os pacientes da cidade de Mauá

(Arquivo Mauá.xml), no Apêndice B.2.2 para os pacientes da cidade de Ribeirão Pires *(Arquivo RibeirãoPires.xml)* e finalmente no Apêndice B.2.3 para os pacientes da cidade de Rio Grande da Serra *(Arquivo RioGrandedaSerra.xml)*.

5.3.2 Geração de Esquemas Integrados

O Esquema Integrado tem por objetivo disponibilizar uma fonte de dados virtual para representar a integração de diferentes fontes de dados locais. O Esquema Integrado utilizado neste estudo de caso é ilustrado no Apêndice B.3 através do arquivo *IntegraçãoPacientes.xml.* Neste arquivo são disponibilizadas as informações de suporte que definem o mapeamento do Esquema Integrado para os Esquemas Locais das fontes de dados envolvidas no processo de integração.

Similarmente à política adotada nos esquemas locais, este processo é executado somente uma única vez para cada fonte de dados integrada. O processo para geração de Esquemas Integrados pode ser executado pelo *Sistema Cliente*. Para isso, o usuário deverá escolher a opção *Gerador de Esquemas* e em seqüência a opção *Esquemas Integrados*, conforme ilustrado na Figura 30. O funcionamento desta ferramenta está descrito no Capítulo 4 e ilustrado nas Figuras 39 até 42.

Exatamente como acontece nos esquemas locais, o Esquema Integrado criado para este estudo de caso (*IntegraçãoPaciente.XML*) é extenso e pode ser encontrado no Apêndice B.3 deste trabalho.

5.3.3 Processamento e Resultado de Consultas Integradas

Até o presente momento, foram discutidos apenas os aspectos referentes à geração de esquemas locais e esquemas integrados para este estudo de caso. Porém, com o conjunto de esquemas que foram gerados, já podemos realizar consultas distribuídas na fonte de dados integrada através do *Editor de Consultas Integradas*. O acesso e o funcionamento desta ferramenta estão ilustrados na Figura 31 e na Figura 43 respectivamente.

Neste estudo de caso, foi processada uma consulta distribuída na fonte de dados integrada representada pelo esquema *IntegraçãoPacientes.xml.* Os parâmetros para esta consulta foram: retornar todos os campos da fonte de dados virtualizada, selecionar apenas os pacientes que apresentam doença com o CID = H903 e ordenar os resultados por número de atendimento, conforme ilustrado na Figura 56.

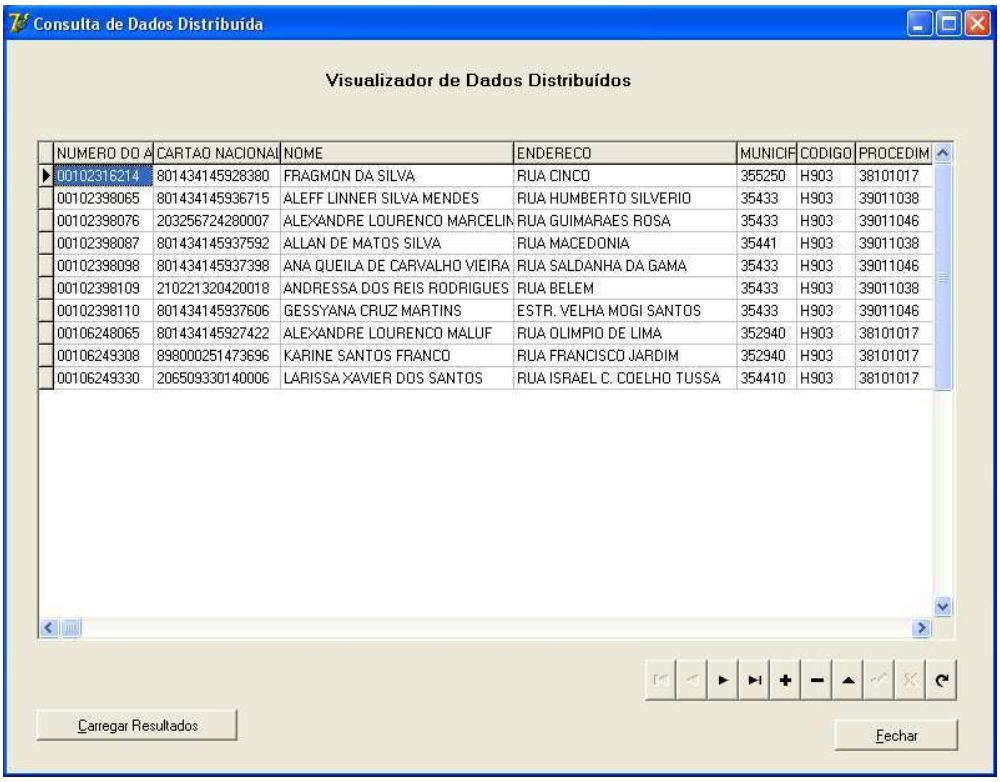


Figura 56: Consulta Distribuída Executada no Estudo de Caso.

Uma vez confeccionada a consulta distribuída pelo *Sistema Cliente*, a mesma é executada através do botão *Processar Consulta.* No momento que o usuário efetua um clique neste botão, é gerado um arquivo de comunicação no formato XML com os parâmetros da consulta distribuída para ser processado pela *Camada Serviços de Integração* (Mediador). O conteúdo do arquivo de comunicação está ilustrado na Listagem 3.

```
<?xml version="1.0" encoding="UTF-8"?>
    <parametros>
        <integratedschema>
            <id>37</id>
        </integratedschema>
        <fields>
            <fieldid>115,116,117,118,119,120,121,122</fieldid>
        </fields>
        <selection>
            <fieldidselection>120</fieldidselection>
            <operator>=</operator>
            <value>"H903"</value>
        </selection>
        <order>
            <id>115</id>
        </order>
    </parametros>
```

Listagem 3 – Conteúdo do Arquivo de Comunicação das Camadas de Serviços Cliente para Serviços de Integração (Mediador)

Conforme ilustrado na Listagem 3, a *Camada Cliente* se comunica com a *Camada de Serviços de Integração* (Mediador) para informar qual é o Esquema Integrado utilizado para a consulta distribuída, os atributos de retorno ao usuário, além dos parâmetros para seleção e ordenação.

No arquivo de comunicação, os atributos são identificados através de seu *id* no *Serviço de Metadados*. Desta forma, a arquitetura garante que não aconteçam conflitos com campos de outras fontes de dados que possuem a mesma denominação. O arquivo de comunicação para retorno do Esquema Integrado e dos esquemas locais utilizados neste estudo de caso está ilustrado nos Apêndices B.4.1 para solicitação do Esquema Integrado *IntegraçãoPacientes.xml,* B.4.2 para solicitação do Esquema Local *Mauá.xml*, B.4.3 para

solicitação do Esquema Local *RibeirãoPires.xml* e B.4.4 para solicitação do Esquema Local *RioGrandedaSerra.xml*.

Uma vez executada a solicitação do Esquema Integrado e dos esquemas locais pertinentes ao *Serviço de Metadados*, o *Fragmentador de Operações* inicia seu processamento e gera as solicitações individuais para cada fonte de dados envolvida no processo de integração. Os arquivos de configuração para execução das solicitações as fontes de dados locais geradas neste estudo de caso podem ser consultados no Apêndice B.5.1 para a fonte de dados local representada pelo arquivo *Mauá.xml,* no Apêndice B.5.2 para a fonte de dados local representada pelo arquivo *RibeirãoPires.xml* e no Apêndice B.5.3 para a fonte de dados local representada pelo arquivo *RioGrandedaSerra.xml*.

Uma vez executada a distribuição de consultas para as fontes de dados locais, o *Fragmentador de Operações* finaliza seu processamento e solicita para a *Camada de Acesso aos Dados* a execução das consultas por fonte de dados. Nos apêndices B.6.1, B.6.2 e B.6.3 estão ilustradas fragmentos dos arquivos de retorno processados nas fontes de dados locais da cidade de Mauá, Ribeirão Pires e Rio Grande da Serra respectivamente.

Uma vez finalizado o processamento do OGSA-DAI, o *Desfragmentador de Resultados* inicia suas atividades e disponibiliza um resultado único ao usuário. O *Desfragmentador de Resultados* pode retornar ao usuário um arquivo integrado de formato genérico que pode ser utilizado por qualquer aplicação cliente e um arquivo especializado para ser utilizado na linguagem *Delphi*, conforme ilustrados nos apêndices B.7 e B.8 respectivamente.

Conclusão 6 – Considerações Finais

6.1 Conclusões

Neste trabalho foi apresentada uma arquitetura para a integração de fontes de dados na Grade Computacional denominada GISE.

A Grade Computacional é um ambiente capaz de suportar uma grande variedade de aplicações e possui características não encontradas em outros ambientes distribuídos, onde a manipulação, acesso e publicação de dados são assuntos de grande importância.

O GISE proporciona uma arquitetura flexível integrada aos serviços disponíveis na Grade Computacional, disponibilizando aos seus usuários serviços eficientes para a integração de fontes de dados heterogêneas e geograficamente distribuídas. A *Arquitetura GISE* é baseada na utilização de MDC extensíveis, construídos com a utilização da linguagem XML. O MDC do GISE é dividido em duas partes. A primeira é responsável por armazenar esquemas que representam fontes de dados locais disponíveis para a integração e a segunda é responsável por armazenar a representação de uma fonte de dados virtualizada.

A *Arquitetura GISE* possui uma série de componentes que são inseridos em camadas, onde podemos citar: *Camada Cliente*, *Camada de Serviços de Metadados*, *Camada de Serviços de Integração* (Mediador), *Camadas de Serviços do Globus Toolkit*, *Camada de Serviços de Acesso a Dados* e *Camada de Fontes de Dados*. Os componentes das camadas da *Arquitetura GISE* são altamente flexíveis e adaptáveis em quaisquer outras arquiteturas para a integração de fontes de dados.

A *Arquitetura GISE* é baseada na arquitetura de Mediadores. Desta forma, podemos ter múltiplos Mediadores na *Arquitetura GISE* especializados no processamento de dados em domínios específicos, tais como: saúde, biologia, etc. Assim podemos disponibilizar, por exemplo, resultados já processados / minerados de consultadas integradas aos usuários.

O GISE é capaz de acessar múltiplos formatos de fontes de dados. Esta capacidade se justifica pela utilização do componente OGSA-DAI para acesso as fontes de dados.

O *Sistema GISE* disponibiliza aos usuários mecanismos simples para realizar as tarefas de geração de esquemas locais e geração de esquemas integrados, além da criação de consultas integradas, processamento e visualização de seus respectivos dados.

O GISE foi implementado utilizando ferramentas e padrões de código aberto. Contudo, seus componentes podem ser incorporados ou especializados de acordo com as necessidades do usuário e domínio. Uma outra vantagem do GISE é a utilização de componentes desenvolvidos, implementados e testados no GT4, disponibilizando uma arquitetura segura, escalável e autonômica para a integração de fontes de dados.

Uma das grandes vantagens da *Arquitetura GISE* é que ela também pode ser aplicada em outros ambientes distribuídos, pois seus componentes implementam algoritmos genéricos que podem ser facilmente utilizados para a integração de fontes de dados em múltiplas plataformas de computação. Para que a *Arquitetura GISE* possa ser utilizada em outras plataformas computacionais, deverá ser inicialmente substituída a *Camada de Serviços do Globus Toolkit* por uma camada de componentes capazes de gerenciar os serviços, recursos e a segurança da infra-estrutura computacional utilizada. Por fim, também será necessária a substituição da *Camada de Serviço de Acesso aos Dados* por uma camada de componentes que disponibilizam serviços para o acesso às fontes de dados individuais através de requisições no formato SQL, retornando os resultados solicitados. É desejável que esta camada forneça os dados das consultas executadas no formato XML para o *Desfragmentador de Operações.* Porém, se a camada utilizada para realizar o acesso às fontes de dados não for capaz de retornar os resultados no formato XML, deverá ser implementado um componente auxiliar nesta camada para realizar a função de conversão dos dados recebidos pelas fontes de dados em XML.

6.2 Contribuições

Neste trabalho de pesquisa que teve por objetivo disponibilizar uma arquitetura extensível para a integração de múltiplas fontes de dados heterogêneas na Grade Computacional foi possível destacar uma série de contribuições, conforme descritas a seguir:

- Uma completa arquitetura para a integração de fontes de dados que pode ser facilmente aplicada em múltiplas plataformas de computação;
- O desenvolvimento de um MDC que representa fontes de dados locais e a virtualização de fontes de dados integradas;
- O protótipo da *Arquitetura GISE* utilizado no estudo de caso deste trabalho. Os módulos implementados neste protótipo permitem aos usuários integrarem fontes de dados de forma simples e segura, resolvendo os conflitos de denominações, formatos, tipos e unidades encontradas nos processos de integrações de fontes de dados. O protótipo da *Arquitetura GISE* foi testado através de um estudo de caso real construído para exemplificar uma visão geral da arquitetura e apresentar suas diversas funcionalidades; e
- Finalmente, o Projeto Integra-EPI recebeu uma ferramenta capaz de integrar fontes de dados heterogêneas e geograficamente distribuídas, utilizando os recursos da plataforma de Grade Computacional para auxiliar no monitoramento de epidemias.

6.3 Perspectivas Futuras

O primeiro protótipo da *Arquitetura GISE* foi desenvolvido, implementado e testado como foi apresentado no Capítulo 5. Entretanto, foi observado que a mesma pode ser aprimorada com a inserção e/ou atualização de algumas funcionalidades em trabalhos futuros.

Primeiramente, podemos focar nossa atenção no *Serviço de Metadados*. Atualmente, quando uma fonte de dados local é atualizada, todos os esquemas que fazem referência a esta

fonte de dados são invalidados. Podemos disponibilizar a *Arquitetura GISE* mecanismos capazes de atualizar as informações armazenadas nos esquemas locais e esquemas integrados automaticamente, libertando o usuário de recadastrar os respectivos esquemas de fontes de dados que sofreram atualizações em suas estruturas. Estes mecanismos podem ser implementados através, de forma simples, pelo versionamento de esquemas e de suas respectivas fontes de dados ou de uma forma mais complexa através da utilização do serviço de notificação disponibilizado pela infra-estrutura computacional.

Além disso, ainda no *Serviço de Metadados*, podemos pensar em mecanismos capazes de extrair automaticamente os esquemas locais e integrados das fontes de dados. Uma forma de se facilitar à geração de esquemas integrados na *Arquitetura GISE* é através do uso intensivo de *Ontologias*. Porém, a geração de esquemas integrados por melhor das hipóteses, será semi-automática.

Ainda no *Serviço de Metadados*, podemos disponibilizar uma arquitetura de armazenamento de informações distribuídas e/ou replicadas. Desta forma, a *Arquitetura GISE* teria acesso a um número maior de instâncias do *Serviço de Metadados* para consulta de esquemas locais e esquemas integrados, oferecendo maior desempenho e tolerância à falhas aos seus usuários.

Na camada de *Serviços de Integração* (Mediador), podemos adicionar um número maior de componentes para a resolução de conflitos decorrentes do processo de integração de múltiplas fontes de dados identificados no Capítulo 3 deste trabalho. Ainda nesta camada, podemos distribuir o processamento das funcionalidades de seus componentes. Uma forma de implementar esta funcionalidade é com a utilização do OGSA-DQP. Desta forma, a *Arquitetura GISE* teria um melhor desempenho no processamento de consultas distribuídas pela utilização de múltiplos recursos computacionais.

Ainda na camada de *Serviços de Integração* (Mediador), podemos disponibilizar componentes capazes de executar o processamento posterior dos dados integrados e disponibiliza-los aos usuários. Por exemplo, podemos pensar na implementação de componentes especializados na mineração de dados *(DataMining)* na área de Saúde Coletiva para serem utilizados na *Arquitetura GISE*.

Podemos ainda, na camada de *Serviços de Integração* (Mediador), disponibilizar aos usuários um maior número de formatos de saída das consultas distribuídas. Desta forma, os dados recebidos poderiam ser trabalhados nativamente em um maior número de Sistemas Gerenciadores de Banco de Dados.

Podemos também acrescentar a *Arquitetura GISE* mecanismos capazes de integrar tipos de dados complexos, como por exemplo, dados georeferenciados.

Finalmente, podemos disponibilizar as funcionalidades de inclusão, alteração e exclusão de dados geograficamente distribuídos de forma transparente na *Arquitetura GISE*.

Referências

ABITEBOUL, S.; BUNEMAN, P.; SUCIU, D. *Data on the Web: from Relations to Semistructured Data and XML.* San Francisco, California, USA: Morgan Kaufmann Publishers, 2000.

ALLCOCK, W.; BESTER, J.; BRESNAHAN, J.; CHERVENAK, A.; LIMING, L.; MEDER, S.; TUECK, S. *GridFTP protocol specification*. [S.I.: s.n], 2002. Disponível em: http://www.globus.org/research/papers/GFD-R.0201.pdf. Acessado em: 13/12/2005.

ALPDEMIR, M. N.; MUKHERJEE, A.; PATON, N. W.; WATSON, P.; FERNANDES, A. A.; GOUNARIS, A.; SMITH, J. *Service-based distributed querying on the grid.* [S.I.]: Springer, 2003. p.467-482.

ANDREWS, T.; CURBERA, F.; DHOLAKIA, H.; SYSTEMS, S.; GOLAND, Y.; KLEIN, J.; LEYMANN, F.; LIU, K.; ROLLER, D.; SMITH, D.; THATTE, S.; TRICKOVIC, I.; WEERAWARENA, S. *Business Process Execution Language for Web Services – Version 1.1*. [S.I.: s.n], 2003. Disponível em: ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf. Acessado em: 15/12/2005.

ATKINSON, M. *Rationale for Choosing the Open Grid Services Architecture.* In: Grid Computing. Chichester - England: Wiley & Sons, 2003. p. 199-215.

ATKINSON, M.; CHERVENAK, A. L.; KUNSZT, P.; NARANG, I.; PATON, N. W.; PEARSON, D.; SHOSHANI, A.; WATSON, P. *Data Access, Integration and Management.* In: The Grid: Blueprint for a New Computing Infrastructure. San Franciso – Califórnia: Eds. Morgan Kaufmann, 2004.

ATKINSON, M.; KARASAVVAS, K.; ANTONIOLETTI, M.; BAXTER, R.; BORLEY, A.; CHUE, H. N.; HUME, A.; JACKSON, M.; KRAUSE, A.; LAWS, S.; PATON, N.; SCHOPF, J. M, TOURLAS, K.; WATSON, P. *A New Architecture for OGSA-DAI.* [S.I.: s.n], 2005. Disponível em: http://www.ogsadai.org.uk/docs/OtherDocs/432.pdf. Acessado em: 19/11/2005.

BAKER, M.; BUYA, R.; LAFORENZA, D. *Grids and Grid Technologies for Wide Area Distributed Computing, Software*. In: Software Practice and Experience. [S.I.]: Wiley & Sons, 2002. p. 1437-1466.

BANAEI-KASHANI, F.; CHEN, C.; SHANABI, C. *WSPDS: Web Services Peer-to-Peer Discovery Device*. In: Proc. Of International Sysmposium on Web Services and Applications (ISWS'04). Los Angeles – USA: [s.n], 2004. Disponível em: http://citeseer.ist.psu.edu/correct/695035. Acessado em: 16/12/2005.

BARU, C.; GUPTA, A.; LUDÄSCHER, B.; MARCIANO, R.; PAPAKONSTANTINOU, Y.; VELIKHO, V P.; YANNAKOPULOS, A. *XML-Based Information Mediation with MIX.* In: in Demo Session - ACM-SIGMOD'99. Philadelphia: [s.n], 1999. Disponível em http://citeseer.ist.psu.edu/baru99xmlbased.html. Acessado em: 05/10/2005.

BOTELHO, L. R. *Arquitetura Extensível para Publicação de Dados Georeferrenciados.* Tese; (Mestrado em Informática) - Universidade Federal do Rio de Janeiro. Rio de Janeiro: [s.n], 2004.

CHEDE, C. T. *Service Grids, OGSA e Web Services.* In: Grid Computing – Um Novo Paradigma Computacional. Rio de Janeiro – Brasil: Eds. Brasport - ISBN 85-7452-193-0, 2004. p. 89-98.

CHRISTOPHIDES, V.; CLUET, S.; SIMÉON, J. *On Wrapping Query Languages, Efficient XML Integration.* In: Proceedings of ACM SIGMOD Conference on Management of Data. Dallas – Texas – USA: [s.n], 2000. Disponível em: http://citeseer.ist.psu.edu/ christophides00wrapping.html. Acessado em: 05/10/2005.

COMITO, C.; TALIA, D.; PAOLO, T. *Grid Services: Principles, implementations and use.* In: Int. Journal of Web and Grid Services (IJWGS). Calábria – Itália: Inderscience Publishers, 2005. p. 48-68.

CORBA. Corba Home Page. Disponível em: http://www.corba.org. Acessado em: 15/11/2005.

CZAJKOWSKI, K.; FOSTER, I.; KARONIS, N.; KESSELMAN, C.; MARTIN, C.; SMITH, W.; TUECKE, S. *A Resource Management Architecture for Metacomputing Systems*. In: Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing. [S.I.: s.n.], 1998. p. 62-82.

CZAJKOWSKI, K.; FOSTER, I.; KESSELMAN, C. *Resource Co-Allocation in Computational Grids.* In: Proceedings of the Eighth IEEE International Symposium on High Performance Distributed Computing (HPDC-8). [S.I: s.n.], 1999. p. 219-228.

CZAJKOWSKI, K.; FITZGERALD, S.; FOSTER, I.; KESSELMAN, C. *Grid Information Services for Distributed Resource Sharing.* In: Proceedings of the 12º IEEE International Symposium on High-Performance Distributed Computing (HPDC-10). [S.I]: IEEE Press, 2001.

CZAJKOWSKI, K.; FERGUSON, D. F.; FOSTER, I.; FREY, J.; GRAHAM, S.; SEDUKHIN, I.; SNELLING, D.; TUECKE, S.; VAMBENEPE, W. *The Ws Resource Framework.* [S.I.: s.n], 2004. Disponível em: http://www.globus.org/wsrf/specs/ws-wsrf.pdf. Acessado em: 22/12/2005.

CZAJKOWSKI, K.; FERGUSON, D.; FOSTER, I.; FREY, J.; GRAHAM, S.; MAGUIRE, T.; SNELLING, D.; TUECKE, S. *From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & Evolution*. [S.I.: s.n], 2005. Disponível em: http://www.globus.org/wsrf/specs/ogsi_to_wsrf_1.0.pdf. Acessado em: 20/12/2005.

DATA. *Data Management: Key Concepts*. [S.I.: s.n], 2005. Disponível em: http://www-unix.globus.org/toolkit/docs/4.0/data/key/. Accessado em: 18/12/2005.

DRS. *GT 4.0 Release Notes: Data Replication Service (DRS).* [S.I.: s.n], 2005. Disponível em: http://www-unix.globus.org/toolkit/docs/4.0/techpreview/datarep/DataRep_Release_Notes.html. Acessado em: 18/12/2005.

FAFNER. *FAFNER Home Page*, 2005. Disponível em: http://www.npac.syr.edu/factoring.html. Acessado em: 15/11/2005.

FOSTER, I. *What is the Grid? A Three Point Checklist.* In: GRID Today - Daily News and Information for the Global Grid Community. [S.I.: s.n], 2002.

FOSTER, I. *Globus Toolkit Version 4: Software for Service-Oriented Systems.* In: IFIP International Conference on Network and Parallel Computing. [S.I.]: Springer-Verlag LNCS, 2005. p. 2-13.

FOSTER, I.; KESSELMAN, C. *Globus: A Metacomputing Infrastructure Toolkit* In: Intl J. Supercomputer Applications, [S.I.: s.n], 1997. p. 115-128.

FOSTER, I.; KESSELMAN, C. *The Globus Project: A Status Report*. In: Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop. [S.I.: s.n], 1998. p. 4-18.

FOSTER, I.; KESSELMAN, C. *Computational Grids"* In: The Grid: Blueprint for a New Computing Infrastructure. [S.I.]: Morgan-Kaufman, 1999.

FOSTER, I.; KESSELMAN, C. *Globus: A Toolkit-Based Grid Architecture.* In: The Grid: Blueprint for a New Computing Infrastructure. [S.I.]: Morgan Kaufmann, 1999. p. 259-278.

FOSTER, I.; GEISLER, J.; NICKLESS, W.; SMITH, W.; TUECKE, S. *Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment.* In: Proc. 5th IEEE Symposium on High Performance Distributed Computing. [S.I.: s.n], 1997. p. 562-571.

FOSTER, I.; ROY, A.; SANDER, V. *A Quality of Service Architecture that Combines Resource Reservation and Application Adaptation.* In: Proc. 8th International Workshop on Quality of Service. [S.I.: s.n], 2000.

FOSTER, I.; KESSELMAN, C.; TUECKE, C. *The Anatomy of the Grid: Enabling Scalable Virtual Organizations*. In: International J. Supercomputer Applications - 15(3). [S.I.: s.n], 2001.

FOSTER, I.; KESSELMAN, C.; NICK, J.; TUECKE, S. *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.* [S.I.: s.n], 2002. Disponível em: http://www.globus.org/research/papers/ogsa.pdf. Acessado em: 19/12/2005.

FOSTER, I.; KESSELMAN, C.; NICK, J.; TUECKE, S. *Grid Services for Distributed System Integration.* In: IEEE Computer, 35(6). [S.I.: s.n], 2002. p. 37-46.

FOSTER, I.; FREY, J.; GRAHAM, S.; TUECKE, S.; CZAJKOWSKI, K.; FERGUSON, D.; LEYMANN, F.; NALLY, M.; SEDUKHIN, I.; SNELLING, D.; STOREY, T.; VAMBENEPE, W.; WEERAWARANA, S. *Modeling Stateful Resources with Web Services.* [S.I.: s.n], 2004. Disponível em: http://www.ibm.com/developerworks /webservices/library/specification/ws-resource/ws-modelingresources.html. Acessado em: 11/12/2005.

GARDARIN, G.; SHA, F.; NGOC, T. *XML-based Components for Federating Multiple Heterogeneous Data Sources.* In: Proc. of International Conference on Conceptual Modeling. Paris – France: LNCS, 1999. p. 506-519.

GLOBUS, *Globus Home Page*, 2004. Dísponível em: http://www.globus.org. Acessado em: 15/11/2005.

GOLDMAN, R.; MCHUGH, J.; WIDOM, J. *From Semistructured Data to XML:Migrating the Lore Data Model and Query Language.* In: 2nd International Workshop on the Web and Databases (WebDB '99). Philadelphia – Pennsylvania - USA: [s.n], 2005. p. 25-30.

GRIMSHAW, A. S.; WULF, W. A. *The Legion Vision of a Worldwide Virtual Computer*" In: Communications of the ACM - vol. 40. [S.I.: s.n], 1997. p. 39–45.

HASS, H. *Web Services Activity,* [S.I.: s.n], 2002. Disponível em: http://www.w3.org/2002/ws/. Acessado em: 15/11/2005.

HURSON, A. R.; BRIGHT, W. M.; PAKZAD, S. H. *Multidatabase Systems: An Advanced Solution for Global Information Sharing*. Los Alamitos – CA: IEEE Computer Society, 1994.

KREGER, H. *Web Services Conceptual Architrecture*. [S.I.: s.n], 2004. Disponível em: http://www-3.ibm.com/software/solutions/webservices/pdf/WSCA.pdf. Acessado em: 15/11/2005.

LITZKOW, M.; LIVNY, M.; MUTKA, M. *Condor-a Hunter of Idle Workstations*. In: 8th International Conference on Distributed Computing Systems. [Washington – USA: s.n], 1988. p. 104-111.

LUDASCHER, B.; PAPAKONSTANTINOU, Y.; VELIKHOV, P. *A Brief Introduction to XMAS.* [S.I.: s.n], 1999: Disponível em: http://www.db.ucsd.edu/Projects/MIX/docs /XMAS-intro.pdf. Acessado em: 04/12/2005.

LUNIEWSKI, A. W.; NIBLACK, W.; PETKOVIC, D.; THOMAS, J.; WILLIAMS, J. H.; WIMMERS, E. L. *Towards Heterogeneous Multimedia Iinformation Systems: The Garlic Approach*. In: Research Issues in Data Engineering. Los Alamitos, California: [s.n.], 1995. p. 124-131.

MACEDO, R. C.; MESQUITA, V.; CAETANO, A.; VASCONCELOS, A.; TRIBOLET, J. *Web Services como Tecnologia de Suporte a Processos de Negócio.* In: 5ª Conferência da APSI. Lisboa – Portugal: [s.n], 2004.

MACFARLANE, A.; MCCANN, LIDDELL, H. *A Common Data Model For Meta-Data Interoperable Environments.* In: Proceedings of II International Baltic Workshop onDatabase and Information Systems. [S.I.: s.n], 1996.

MDS. *GT 3.9.4 Information Services (MDS): Key Concepts*. [S.I.: s.n], 2005. Disponível em: http://www.globus.org/toolkit/docs/development/3.9.4/info/key. Acessado em: 04/12/2005.

OGSA. *Open Grid Service Architecture*, 2003. Disponível em: http://www.globus.org/ogsa. Acessado em: 15/11/2003.

OGSA-DAI-WSRF. *Open Grid Service Architecture – Data Access Integration WSRF 2.1 User Guide*. [S.I.: s.n], 2005. Disponível em: http://www.ogsadai.org.uk/docs/WSRF2.1/doc/index.html. Acessado em: 19/11/2005.

OZSU, M. T.; VALDURIEZ, P. *Principles of Distributed Database System,* 2ª Edition. New Jersey: Eds. Prentice Hall, 1999.

PINTO, G. R. B. *Integração de Bases de Dados Ambientais*. Tese; (Mestrado em Informática) - Universidade Federal do Rio de Janeiro. Rio de Janeiro: [s.n], 2002.

POTTS, S.; KOPACK, M. *Teach Yourself Web Services in 24 hours.* [S.I.]: Elsevier, 2003. p. 104-112.

RAMAN, V.; NARANG, I.; CRONE, C.; HAAS, L.; MALAIKA, S.; MUKAI, T.; WOLFSON, D.; BARU, C. *Services for Data Access and Processing on Grids* [S.I.: s.n], 2003. Disponível em: http://www.gridforum.org/documents/GFD.14.pdf. Acessado em: 12/11/2005.

RLS-GT. *RLS: System Administrator's Guide.* [S.I.: s.n], 2005. Disponível em: http://www-unix.globus.org/toolkit/docs/4.0/data/rls. Acessado em: 06/12/2005.

RMI. *Remote Method Invocation.* [S.I.: s.n], 2005. Disponível em: http://java.sun.com/j2se/1.4.2/docs/guide/rmi/spec/rmiTOC.html. Acessado em: 15/11/2005

RODRIGEZ-MARTINEZ, M.; ROUSSOPOULOS, N. *Automatic Deployment of Application-Specific Metadata and Code in MOCHA*. InL EDBT'00: Proceeding of the 7th International Conference on Extending Database Technology. [S.I.: s.n], 2000. p. 69-85.

ROURE, D. D.; BAKER, M. A.; JENNINGS, N. R.; SHADBOLT, N. R. *The Evolution of the Grid*. In: Grid Computing – Making the Global Infrastructure a Reality. [S.I.]: Willey - ISBN: 0-470-85319-0, 2003.

SILVA, F.; SENGER, H. *The Grid: An Enabling Infrastructure for Future E-Business, E-Commerce and E-Government Applications* In: Manuel Mendes; Carlos Passos; Reima Suomi. (Org.). Digital Communities in a Networked Society: eCommerce, eGovernment and eBusiness. Dordrecht – Holanda: Kluwer Academic Publishers - ISBN: 1402077955, 2004. p. 253-266.

SOTOMAYOR, B. *Globus Toolkit 3.0 Tutorial*. [S.I.: s.n], 2005. Disponível em: http://gdp.globus.org/gt3-tutorial/singlehtml/progtutorial_0.4.3.html. Acessado em: 15/11/2005.

SZYPERSKI, C. *Independly Extensible Systems – Software Engineering Potential and Challenges* In: Proceedings of 19th Australian Computer Science Conference. Melbourne – Australia: [s.n], 1996.

TUECKE, S.; CZAJKOWSKI, K.; FOSTER, I.; FREY, J.; GRAHAM, S.; KESSELMAN, C.; MAQUIRE, T.; SANDHOLM, T.; SNELLING, D.; VANDERBILT, P. *Open Grid Services Infrastructure Specification.* [S.I.: s.n], 2003. Disponível em: http://xml.coverpages.org/OGSI-SpecificationV110.pdf. Acessado em: 20/11/2005. 2003

UDDI. *UDDI: Universal Description, Discovery and Integration of Web Services,* 2002. Disponível em: http://www.uddi.org. Acessado em: 15/11/2005.

VENTRONE, V.; HEILER, S. *Semantic Heterogeneity as a Result of Domain Evolution*. In: ACM SIGMOD Record, v. 20, n. 4 (Dec.), [S.I.: s.n], 1991. p. 16-20.

WELCH, V. *Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective.* [S.I.: s.n], 2005. Disponível em: http://www.globus.org/toolkit/docs/4.0/security/GT4-GSI-Overview.pdf. Acessado em: 18/12/2005.

WELCH, V.; SIEBENLIST, F.; FOSTER, I.; BRESNAHAN, J.; CZAJKOWSKI, K.; GAWOR, J.; KESSELMAN, C.; MEDER, S.; PEARLMAN, L.; TUECKE, S. *Security for Grid Services.* In: *12º* International Symposium on High Performance Distributed Computing (HPDC-12). [S.I.]: IEEE Press, 2003.

Apêndice A – Glossário

É apresentado neste apêndice um pequeno glossário de alguns termos utilizados ao longo desta dissertação:

A.1 Termos e Afins

BPEL4WS: *Business Process Execution Language for Web Services* - Linguagem que estende o Modelo de Interação de *Web Services* para a especificação de processos de negócio e protocolos de interação.

Certificados X.509: Certificado obtido através de uma CA *(Certification Authority)* que garante uma identidade.

CORBA: *Common Object Request Broker Access* - Especificação que permite a interoperabilidade entre objetos distribuídos.

CP: *Client Application* - Entidade na Arquitetura MOCHA responsável por enviar consultas distribuídas para processamento.

DAIS-WG: *Data Access Integration Service – Working Group* - Grupo de Pesquisa pertencente ao GGF que propõe projetos e especificações para a Integração de Fontes de Dados na Grade Computacional

DAP: *Data Access Provider* - Entidade na Arquitetura MOCHA responsável pelo acesso às fontes de dados.

DNS: *Domain Name Service* – Serviço para a tradução de nomes muito utilizado na Internet para mapear nomes de domínio em endereço IP.

DRS: *Data Replication Service* - Serviço para a replicação de dados na Grade Computacional que integra as funcionalidades do RLS e RFT.

XML Schema: Arquivo no formato XML que armazena regras para a construção e validação de arquivos XML.

FTP: *File Transfer Protocol* - Protocolo utilizado para a transferência de arquivos na Internet.

GARA: *General-Purpose Architecture for Reservation and Allocation* - Componente utilizado para a reserva e alocação de recursos na Grade Computacional.

GDQS: *Grid Distributed Query Service* - Serviço responsável por receber consultas às fontes de dados pelo OGSA-DAI e processar-las na Grade Computacional.

GGF: *Global Grid Fórum*: Grupo internacional formado por diversas empresas e pesquisadores para o desenvolvimento e geração de padrões para a Grade Computacional

GQES: *Grid Query Evaluation Service* - Serviço que recebe uma requisição através de um GDQS e se comunica diretamente com as fontes de dados.

GRAM: *Globus Resource Allocation Manager* – Serviço utilizado no GT3 e no GT4 para alocação e gerenciamento de recursos na Grade Computacional. O GRAM é uma evolução do GARA, utilizado no GT2.

GRIDFTP: *GRID File Transfer Protocol* - Protocolo que determina as regras para a transferência de arquivos na Grade Computacional. O GRIDFTP é uma adaptação através da adoção de funcionalidades ao FTP tradicional visando sua aplicação na Grade Computacional.

GSF: *Grid Service Factory* - Serviço persistente no *Globus Toolkit* responsável por criar e gerenciar o ciclo de vida de instâncias de serviços disponibilizados aos usuários.

GSH: *Grid Service Handle*: É um endereço que é gerado pelo GSF para identificar às instâncias de serviços na Grade Computacional.

GSI: *Globus Security Infrastructure* - É a infra-estrutura de segurança utilizada pelo GT2, GT3 e GT4. O GSI disponibiliza a autenticação única de usuários na Grade Computacional através de Certificados X509 e comunicação segura entre recursos através de SSL, disponibilizando mecanismos de segurança na Grade Computacional.

GSR: *Grid Service Registry* - Serviço persistente responsável por armazenar informações sobre os GSF disponíveis para uso na Grade Computacional.

GT: *Globus Toolkit* – É o *Middleware* mais utilizado no mundo para a construção de Grades Computacionais.

GT2: *Globus Toolkit 2* – São as distribuições 2.x do GT.

GT3: *Globus Toolkit 3* – São as distribuições 3.x do GT.

GT4: *Globus Toolkit 4* – São as distribuições 4.x do GT.

GTCP: *Grid TeleControl Protocol* - protocolo disponibilizado apenas no GT4 para gerenciar a utilização remota de instrumentos na Grade Computacional.

HTML: *Hiper Text Markup Language* - É a linguagem para a publicação de hipertextos na *World Wide Web*. É um formato não proprietário e pode ser criado e processado por uma grande variedade de ferramentas. HTML usa tags como "<h1>" e </h1> para estruturar o texto em cabeçalhos, parágrafos, listas, *links* de hipertexto, etc.

HTTP: *Hiper Text Transfer Protocol* - Protocolo utilizado para a transferência de documentos HTML na Internet.

IIOP: *Internet Inter-ORB Protocol* – Protocolo de transporte usado para disponibilizar comunicação entre objetos CORBA.

JDBC: *Java DataBase Connectivity* é um conjunto de classes e interfaces (API) escritas em *Java* que faz o envio de cláusulas SQL para qualquer banco de dados relacional;

LOA: *Legion Object Access* - Endereço utilizado para a localização de objetos *Legion*. O LOA é formado pelo endereço IP do local que armazena o serviço e sua porta de acesso.

LOID: *Legion Object Identifier* - Identificador único para Objetos *Legion*.

MCS: *Metadata Catalog System* - Sistema que permite o armazenamento de quaisquer metadados de recursos e/ou serviços na Grade Computacional

MDC: *Modelo de Dados Canônico* - É um modelo padrão para quais as bases de dados envolvidas no processo de integração são mapeadas a fim de minimizar suas diferenças sintáticas.

MDS: *Monitoring Discovery Service* - Serviço que armazena e disponibiliza informações específicas sobre os recursos computacionais disponíveis na Grade Computacional.

MDS2: *Monitoring Discovery Service* utilizado no GT2.

MDS3: *Monitoring Discovery Service* utilizado no GT3.

MDS4: *Monitoring Discovery Service* utilizado no GT4.

ODBC: Open *Database Communication* - É uma API *(Application Program Interface)* padrão, utilizada para acessar sistemas de gerenciamento de banco de dados, desenvolvida pela Microsoft.

OGSA: *Open Grid Service Architecture* – Arquitetura que define os padrões e componentes necessários para a construção de Grades Computacionais orientadas a serviços

OGSA-DAI: *Open Grid Service Architecture – Data Access Integration* - Arquitetura baseada em OGSA para o acesso a fonte de dados na Grade Computacional.

OGSA-DQP: *Open Grid Service Architecture - Distributed Query Processor* - Arquitetura baseada em OGSA que permite o processamento distribuído de consultas às fontes de dados na Grade Computacional

OGSI: *Open Grid Service Infrastructure* - Especificação formal que implementa os componentes do OGSA e especifica regras para a construção de *Grid Services*. O OGSI é utilizado até a versão 3.2 do *Globus Toolkit*.

ORB: *Object Request Broker* - Componente do CORBA que disponibiliza um Proxy a um objeto distribuído, podendo desta forma, invocar-lo como se o objeto estivesse armazenado localmente.

PPL: *Peer Programming Language* - Linguagem para a programação dos pontos na rede GDIS

QPC: *Query Processor Coordinator* - Componente responsável pela análise, validação, otimização, planejamento de decomposição, execução de consultas e gerenciamento de erros na arquitetura MOCHA.

RFT: *Reliable File Transfer* – Componente do *Globus Toolkit* que permite a transferência confiável de arquivos na Grade Computacional.

RLS: *Replica Location Service* – Componente do *Globus Toolkit* para a localização de réplicas na Grade Computacional.

RMI: *Remote Method Invocation* – Método utilizado para a invocação remota de objetos na linguagem *JAVA*. Com o RMI podemos utilizar dos métodos de um objeto distribuído na rede como se ele estivesse armazenado localmente. O RMI é projetado apenas para se trabalhar com objetos *JAVA*.

RSA: O RSA é um algoritmo de criptografia de dados, que deve o seu nome a três professores do Instituto MIT (fundadores da atual empresa RSA Data Security, Inc.), Ron Rivest, Adi Shamir e Len Adleman, que inventaram este algoritmo. É a mais bem sucedida implementação de sistemas de chaves assimétricas, e fundamenta-se em Teorias Clássicas dos Números. É considerado dos mais seguros algoritmos para criptografia de dados.

RSL: *Resource Specification Language* - Linguagem utilizada pelo GRAM para a especificação de recursos na Grade Computacional

SOAP: *Simple Object Access Protocol* - Protocolo que define uma gramática para a troca de mensagens que não é vinculada a nenhuma arquitetura de hardware ou software.

SQL: *Structured Query Language*, ou Linguagem de Consulta Estruturada. O SQL é uma linguagem de pesquisa declarativa para banco de dados relacional. Muitas das características originais do SQL foram inspiradas na álgebra relacional.

SSL: *Security Socket Layer* – Sistema que permite a comunicação segura através de chaves de criptografia entre duas entidades.

TCP/IP: *Transfer Control Protocol / Internet Protocol* – Protocolo utilizado na Internet para o controle da transferência de dados entre duas entidades.

UDDI: *Universal Description, Discovery and Integration* – É um Web Service implementado para disponibilizar um repositório de informações para a localização de outros *Web Services*.

WMS: *Workspace Management Service* – Serviço disponível no GT4 para a alocação dinâmica de recursos para contas Unix na Grade Computacional.

WSDL: *Web Service Description Language* - Documento que descreve a forma de acesso a um Web Service, tais como, interfaces, parâmetros, etc.

WSPDS: *Web Services Peer-to-Peer Discovery Device* - Registro para o armazenamento de informações para a localização de *Web Services* através da Arquitetura ponto – a – ponto.

WSRF: *Web Service Resource Framework* - Especificação formal que substitui o OGSI e especifica regras para a construção de *Grid Services*. O WSRF é utilizado a partir da versão 4.x do *Globus Toolkit*.

XML: *Extensible Markup Language* - É uma linguagem para descrever outras linguagens, onde podemos criar nossa própria linguagem de marcação. Ao contrário da HTML que tem um conjunto fixo de marcações que são entendidas por todos os navegadores, mas que são limitados à exibição de dados, a XML é o resultado de uma evolução no sentido de definir um formato padrão para documentos que seja portátil e estruturado e que forneça informações tanto sobre o conteúdo quanto contexto.

XSD: *XML Schema Document* - Extensão utilizada em arquivos que armazenam esquemas para a construção de documentos XML.

Apêndice B – Esquemas e Arquivos XML Gerados pelo Estudo de Caso do GISE

B.1 Esquemas XSD

B.1.1 - Esquema de Validação de Esquemas Locais (*GISE-LOCAL-SCHEMA*)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
	<xs:element name="DataSet">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="DataSetId"/>
				<xs:element ref="DatasetDescription"/>
				<xs:element ref="MetaData"/>
				<xs:element ref="Field" maxOccurs="unbounded"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<xs:element name="DataSetId">
		<xs:simpleType>
			<xs:restriction base="xs:byte">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="DataSource">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="DataSourceInformation"/>
				<xs:element ref="DataSet"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<xs:element name="DataSourceAlias">
		<xs:simpleType>
			<xs:restriction base="xs:string">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="DataSourceDescription">
		<xs:simpleType>
			<xs:restriction base="xs:string">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="DataSourceID">
		<xs:simpleType>
			<xs:restriction base="xs:byte">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="DataSourceInformation">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="DataSourceAlias"/>
				<xs:element ref="DataSourceID"/>
				<xs:element ref="DataSourceDescription"/>
```

```
                        <xs:element ref="DataSourceType"/>
                        <xs:element ref="DataSourceOnwer"/>
                        <xs:element ref="Validate"/>
                        <xs:element ref="ResourceName"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="DataSourceOnwer">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="DataSourceType">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="DatasetDescription">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="Field">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="FieldId"/>
                        <xs:element ref="FieldDescription"/>
                        <xs:element ref="FieldIdType"/>
                        <xs:element ref="Is_PK"/>
                        <xs:element ref="Format"/>
                        <xs:element ref="StandardValue"/>
                        <xs:element ref="Is_Required"/>
                        <xs:element ref="Size"/>
                        <xs:element ref="Unit"/>
                        <xs:element ref="MetaData"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="FieldDescription">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="FieldId">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="FieldIdType">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="Format">
```

```
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="Is_PK">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="Is_Required">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="MetaData">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="MetaData">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="ResourceName">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="Size">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="StandardValue" type="xs:string"/>
        <xs:element name="Unit" type="xs:string"/>
        <xs:element name="Validate">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
</xs:schema>
```

B.1.2 – Esquema de Validação de Esquemas Integrados (*GISE-INTEGRATED-SCHEMA*)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
	<xs:element name="IntDataBaseOnwer">
		<xs:simpleType>
			<xs:restriction base="xs:string">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="IntDatabase">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="IntDatabaseID"/>
				<xs:element ref="IntDatabaseAlias"/>
				<xs:element ref="IntDatabaseDescription"/>
				<xs:element ref="IntDatabaseType"/>
				<xs:element ref="IntDataBaseOnwer"/>
				<xs:element name="NFields"/>
				<xs:element ref="Validate"/>
				<xs:element name="MetaData"/>
				<xs:element ref="IntField" maxOccurs="unbounded"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>
	<xs:element name="IntDatabaseAlias">
		<xs:simpleType>
			<xs:restriction base="xs:string">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="IntDatabaseDescription">
		<xs:simpleType>
			<xs:restriction base="xs:string">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="IntDatabaseID">
		<xs:simpleType>
			<xs:restriction base="xs:byte">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="IntDatabaseType">
		<xs:simpleType>
			<xs:restriction base="xs:string">
			</xs:restriction>
		</xs:simpleType>
	</xs:element>
	<xs:element name="IntField">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="IntFieldId"/>
				<xs:element ref="IntIDFieldDatabase"/>
				<xs:element ref="IntFieldDescription"/>
				<xs:element ref="IntIs_PK"/>
				<xs:element ref="IntIs_Required"/>
```

```
                        <xs:element ref="IntFormat"/>
                        <xs:element ref="IntStandardValue"/>
                        <xs:element ref="IntSize"/>
                        <xs:element ref="IntUnit"/>
                        <xs:element ref="IntFieldIdType"/>
                        <xs:element name="MetaData"/>
                        <xs:element ref="IntFieldComposition" maxOccurs="unbounded"/>
                </xs:sequence>
        </xs:complexType>
</xs:element>
<xs:element name="IntFieldComposition">
        <xs:complexType>
                <xs:sequence>
                        <xs:element ref="IntFieldCompositionId"/>
                        <xs:element ref="IntFieldId"/>
                        <xs:element ref="IntFieldCompositionDatasourceId"/>
                        <xs:element ref="IntFieldCompositionDatasetId"/>
                        <xs:element ref="IntFieldCompositionFieldId"/>
                        <xs:element ref="IntFieldCompositionFunction"/>
                </xs:sequence>
        </xs:complexType>
</xs:element>
<xs:element name="IntFieldCompositionDatasetId">
        <xs:simpleType>
                <xs:restriction base="xs:byte">
                </xs:restriction>
        </xs:simpleType>
</xs:element>
<xs:element name="IntFieldCompositionDatasourceId">
        <xs:simpleType>
                <xs:restriction base="xs:byte">
                </xs:restriction>
        </xs:simpleType>
</xs:element>
<xs:element name="IntFieldCompositionFieldId">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                </xs:restriction>
        </xs:simpleType>
</xs:element>
<xs:element name="IntFieldCompositionFunction">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                </xs:restriction>
        </xs:simpleType>
</xs:element>
<xs:element name="IntFieldCompositionId">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                </xs:restriction>
        </xs:simpleType>
</xs:element>
<xs:element name="IntFieldDescription">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                </xs:restriction>
        </xs:simpleType>
</xs:element>
<xs:element name="IntFieldId">
        <xs:simpleType>
```

```
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntFieldIdType">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntFormat">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntIDFieldDatabase">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntIs_PK">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntIs_Required">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntSize">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="IntStandardValue" type="xs:string"/>
        <xs:element name="IntUnit">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="MetaData" type="xs:string"/>
        <xs:element name="Validate">
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
</xs:schema>
```

B.2 Esquemas Locais

B.2.1 – Pacientes de Mauá (*Mauá.xml)*

```
<?xml version="1.0" encoding="UTF-8"?>
        <DataSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\GISE\Schemas\BaseSchema.xsd">
                <DataSourceInformation>
                        <DataSourceAlias>MAUA</DataSourceAlias>
                        <DataSourceID>92</DataSourceID>
                        <DataSourceDescription>MAUA</DataSourceDescription>
                        <DataSourceType>ACCESS</DataSourceType>
                        <DataSourceOnwer>DBA</DataSourceOnwer>
                        <Validate>Sim</Validate>
                        <ResourceName>FONTE DE DADOS DA UBS DE MAUA</ResourceName>
                </DataSourceInformation>
                        <DataSet>
                                <DataSetId>80</DataSetId>
                                <DatasetDescription>MAUA</DatasetDescription>
                                <MetaData>PACIENTES ATENDIDOS EM MAUA</MetaData>
                                <Field>
                                        <FieldId>129</FieldId>
                                        <FieldDescription>ID_ATEND</FieldDescription>
                                        <FieldIdType>Numero</FieldIdType>
                                        <Is_PK>Sim</Is_PK>
                                        <Format>99999999999</Format>
                                        <StandardValue></StandardValue>
                                        <Is_Required>Sim</Is_Required>
                                        <Size>0</Size>
                                        <Unit></Unit>
                                        <MetaData>IDENTIFICACAO PROCEDIMENTO ALTA
                                COMPLEXIDADE</MetaData>
                                </Field>
                                <Field>
                                        <FieldId>130</FieldId>
                                        <FieldDescription>CNS</FieldDescription>
                                        <FieldIdType>Numero</FieldIdType>
                                        <Is_PK>Nao</Is_PK>
                                        <Format>999999999999999</Format>
                                        <StandardValue></StandardValue>
                                        <Is_Required>Sim</Is_Required>
                                        <Size>0</Size>
                                        <Unit></Unit>
                                        <MetaData>CARTAO NACIONAL DE SAUDE</MetaData>
                                </Field>
                                <Field>
                                        <FieldId>131</FieldId>
                                        <FieldDescription>NOME</FieldDescription>
                                        <FieldIdType>Texto</FieldIdType>
                                        <Is_PK>Nao</Is_PK>
                                        <Format>@!</Format>
                                        <StandardValue></StandardValue>
                                        <Is_Required>Sim</Is_Required>
                                        <Size>0</Size>
                                        <Unit></Unit>
                                        <MetaData>NOME DO PACIENTE</MetaData>
                                </Field>
                                <Field>
                                        <FieldId>132</FieldId>
                                        <FieldDescription>ENDERECO</FieldDescription>
                                        <FieldIdType>Texto</FieldIdType>
```

```
                <Is_PK>Nao</Is_PK>
                <Format>@!</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>ENDERECO DO PACIENTE</MetaData>
            </Field>
            <Field>
                <FieldId>133</FieldId>
                <FieldDescription>COD_MUNIC</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>9999999</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>MUNICIPIO DE RESIDENCIA DO
            ATENDIDO</MetaData>
            </Field>
            <Field>
                <FieldId>134</FieldId>
                <FieldDescription>CID</FieldDescription>
                <FieldIdType>Texto</FieldIdType>
                <Is_PK></Is_PK>
                <Format>@!</Format>
                <StandardValue></StandardValue>
                <Is_Required></Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO INTERNACIONAL DE
            DOENCA</MetaData>
            </Field>
            <Field>
                <FieldId>135</FieldId>
                <FieldDescription>REMUNERAC</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK></Is_PK>
                <Format>99.999,99</Format>
                <StandardValue></StandardValue>
                <Is_Required>Nao</Is_Required>
                <Size>0</Size>
                <Unit>REAL</Unit>
                <MetaData>REMUNERACAO DO ATENDIDO</MetaData>
            </Field>
            <Field>
                <FieldId>136</FieldId>
                <FieldDescription>PROCEDIMEN</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>99999999</Format>
                <StandardValue></StandardValue>
                <Is_Required>Nao</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO NA TABELA SIA SUS</MetaData>
            </Field>
        </DataSet>
</DataSource>
```

B.2.2 – Pacientes de Ribeirão Pires *(RibeirãoPires.xml)*

```
<?xml version="1.0" encoding="UTF-8"?>
        <DataSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\GISE\Schemas\BaseSchema.xsd">
                <DataSourceInformation>
                        <DataSourceAlias>RPIRES</DataSourceAlias
                        <DataSourceID>93</DataSourceID>
                        <DataSourceDescription>RPIRES</DataSourceDescription>
                        <DataSourceType>INTERBASE</DataSourceType>
                        <DataSourceOnwer>EDUARDO</DataSourceOnwer>
                        <Validate>Sim</Validate>
                        <ResourceName>FONTES DE DADOS DOS USUARIOS DE RIB.
                PIRES</ResourceName>
                </DataSourceInformation>
                        <DataSet>
                                <DataSetId>81</DataSetId>
                                <DatasetDescription>RPIRES</DatasetDescription>
                                <MetaData>USUARIOS DE RIBEIRAO PIRES</MetaData>
                                <Field>
                                        <FieldId>137</FieldId>
                                        <FieldDescription>ATENDIMENT</FieldDescription>
                                        <FieldIdType>Numero</FieldIdType>
                                        <Is_PK>Sim</Is_PK>
                                        <Format>99999999999</Format>
                                        <StandardValue></StandardValue>
                                        <Is_Required>Sim</Is_Required>
                                        <Size>0</Size>
                                        <Unit></Unit>
                                        <MetaData>IDENTIFICACAO DO ATENDIMENTO</MetaData>
                                </Field>
                                <Field>
                                        <FieldId>138</FieldId>
                                        <FieldDescription>PACIENTE</FieldDescription>
                                        <FieldIdType>Texto</FieldIdType>
                                        <Is_PK>Nao</Is_PK>
                                        <Format>@!</Format>
                                        <StandardValue></StandardValue>
                                        <Is_Required>Nao</Is_Required>
                                        <Size>0</Size>
                                        <Unit></Unit>
                                        <MetaData>NOME DO PACIENTE</MetaData>
                                </Field>
                                <Field>
                                        <FieldId>139</FieldId>
                                        <FieldDescription>NCNS</FieldDescription>
                                        <FieldIdType>Numero</FieldIdType>
                                        <Is_PK></Is_PK>
                                        <Format>999999999999999</Format>
                                        <StandardValue></StandardValue>
                                        <Is_Required>Nao</Is_Required>
                                        <Size>0</Size>
                                        <Unit></Unit>
                                        <MetaData>NUMERO DO CNS DO PACIENTE</MetaData>
                                </Field>
                                <Field>
                                        <FieldId>140</FieldId>
                                        <FieldDescription>ENDERECO</FieldDescription>
                                        <FieldIdType>Texto</FieldIdType>
                                        <Is_PK>Nao</Is_PK>
```

```
                <Format>@!</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>ENDERECO DE RESIDENCIA DO
            PACIENTE</MetaData>
            </Field>
            <Field>
                <FieldId>141</FieldId>
                <FieldDescription>CIDADE</FieldDescription>
                <FieldIdType>Texto</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>@!</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CIDADE DE RESIDENCIA DO
            PACIENTE</MetaData>
            </Field>
            <Field>
                <FieldId>142</FieldId>
                <FieldDescription>SALARIO</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>99,99</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit>SALARIO</Unit>
                <MetaData>NUMERO DE SALARIOS RECEBIDOS PELO
            PACIENTE</MetaData>
            </Field>
            <Field>
                <FieldId>143</FieldId>
                <FieldDescription>PROCSUS</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>99999999</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO DO PROCEDIMENTO SUS</MetaData>
            </Field>
            <Field>
                <FieldId>144</FieldId>
                <FieldDescription>COD_DOENCA</FieldDescription>
                <FieldIdType>Texto</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>@!</Format>
                <StandardValue></StandardValue>
                <Is_Required>Nao</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO DE DOENCA CID</MetaData>
            </Field>
        </DataSet>
</DataSource>
```

B.2.3 – Pacientes de Rio Grande da Serra *(RioGrandedaSerra.xml)*

```
<?xml version="1.0" encoding="UTF-8"?><DataSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="C:\GISE\Schemas\BaseSchema.xsd">
	<DataSourceInformation>
		<DataSourceAlias>RGS</DataSourceAlias>
		<DataSourceID>94</DataSourceID>
		<DataSourceDescription>RGS</DataSourceDescription>
		<DataSourceType>INFORMIX</DataSourceType>
		<DataSourceOnwer>DBA</DataSourceOnwer>
		<Validate>Sim</Validate>
		<ResourceName>RGS</ResourceName>
		</DataSourceInformation>
		<DataSet>
			<DataSetId>82</DataSetId>
			<DatasetDescription>RGS</DatasetDescription>
			<MetaData>PACIENTE ATENDIDOS EM RGS</MetaData>
			<Field>
				<FieldId>145</FieldId>
				<FieldDescription>USUARIO</FieldDescription>
				<FieldIdType>Texto</FieldIdType>
				<Is_PK>Nao</Is_PK>
				<Format>@!</Format>
				<StandardValue></StandardValue>
				<Is_Required>Sim</Is_Required>
				<Size>0</Size>
				<Unit></Unit>
				<MetaData>NOME DO USUARIO</MetaData>
			</Field>
			<Field>
				<FieldId>146</FieldId>
				<FieldDescription>NCNS</FieldDescription>
				<FieldIdType>Numero</FieldIdType>
				<Is_PK>Nao</Is_PK>
				<Format>999999999999999</Format>
				<StandardValue></StandardValue>
				<Is_Required>Sim</Is_Required>
				<Size>0</Size>
				<Unit></Unit>
				<MetaData>NUMERO DO CARTAO CNS</MetaData>
			</Field>
			<Field>
				<FieldId>147</FieldId>
				<FieldDescription>NAPAC</FieldDescription>
				<FieldIdType>Numero</FieldIdType>
				<Is_PK>Sim</Is_PK>
				<Format>99999999999</Format>
				<StandardValue></StandardValue>
				<Is_Required>Sim</Is_Required>
				<Size>0</Size>
				<Unit></Unit>
				<MetaData>NUMERO DE ATENDIMENTO DO PACIENTE</MetaData>
			</Field>
			<Field>
				<FieldId>148</FieldId>
				<FieldDescription>ENDERECO</FieldDescription>
				<FieldIdType>Texto</FieldIdType>
				<Is_PK>Nao</Is_PK>
				<Format>@!</Format>
				<StandardValue></StandardValue>
```

```
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>ENDERECO DE MORADIA DO PACIENTE</MetaData>
            </Field>
            <Field>
                <FieldId>149</FieldId>
                <FieldDescription>COD_CIDADE</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>9999999</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO BRASILEIRO DE MUNICIPIO</MetaData>
            </Field>
            <Field>
                <FieldId>150</FieldId>
                <FieldDescription>SALARIO</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>99999,99</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit>REAL</Unit>
                <MetaData>SALARIO RECEBIDO PELO PACIENTE</MetaData>
            </Field>
            <Field>
                <FieldId>151</FieldId>
                <FieldDescription>COD_CID</FieldDescription>
                <FieldIdType>Texto</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>@!</Format>
                <StandardValue>
                </StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO INTERNACIONAL DO CID10</MetaData>
            </Field>
            <Field>
                <FieldId>152</FieldId>
                <FieldDescription>PSUS</FieldDescription>
                <FieldIdType>Numero</FieldIdType>
                <Is_PK>Nao</Is_PK>
                <Format>99999999</Format>
                <StandardValue></StandardValue>
                <Is_Required>Sim</Is_Required>
                <Size>0</Size>
                <Unit></Unit>
                <MetaData>CODIGO DO PROCEDIMENTO SIA SUS</MetaData>
            </Field>
        </DataSet>
</DataSource>
```

B.3 Esquema Integrado IntegraçãoPacientes *(IntegraçãoPacientes.xml)*

```
<?xml version="1.0" encoding="UTF-8"?><IntDatabase
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\GISE\Schemas\IntegratedSchema.xsd">
        <IntDatabaseID>37</IntDatabaseID>
        <IntDatabaseAlias>FD VIRTUALIZADA MAUA, RIB.PIRES E
RGS</IntDatabaseAlias>
        <IntDatabaseDescription>DADOS INTEGRADOS 3
CIDADES</IntDatabaseDescription>
        <IntDatabaseType>GISE</IntDatabaseType>
        <IntDataBaseOnwer>EDUARDO</IntDataBaseOnwer>
        <Validate>Sim</Validate>
        <MetaData></MetaData>
        <IntField>
                <IntFieldId>115</IntFieldId>
                <IntIDFieldDatabase>1</IntIDFieldDatabase>
                <IntFieldDescription>NUMERO DO ATENDIMENTO</IntFieldDescription>
                <IntIs_PK>Sim</IntIs_PK>
                <IntIs_Required>Sim</IntIs_Required>
                <IntFormat>99999999999</IntFormat>
                <IntStandardValue></IntStandardValue>
                <IntSize>15</IntSize>
                <IntUnit>NA</IntUnit>
                <IntFieldIdType>Numero</IntFieldIdType>
                <MetaData>NUMERO DO PROCEDIMENTO DE ALTA
        COMPLEXIDADE</MetaData>
                        <IntFieldComposition>
                                <IntFieldCompositionId>144</IntFieldCompositionId>
                                <IntFieldId>115</IntFieldId>
                                <IntFieldCompositionDatasourceId>92</IntFieldComposition
                        DatasourceId>
                                <IntFieldCompositionDatasetId>80</IntFieldCompositionData
                        setId>
                                <IntFieldCompositionFieldId>129</IntFieldCompositionFieldI
                        d>
                                <IntFieldCompositionFunction>
                        </IntFieldCompositionFunction>
                        </IntFieldComposition>
                        <IntFieldComposition>
                                <IntFieldCompositionId>145</IntFieldCompositionId>
                                <IntFieldId>115</IntFieldId>
                                <IntFieldCompositionDatasourceId>94</IntFieldComposition
                        DatasourceId>
                                <IntFieldCompositionDatasetId>82</IntFieldCompositionData
                        setId>
                                <IntFieldCompositionFieldId>147</IntFieldCompositionFieldI
                        d>
                                <IntFieldCompositionFunction>
                        </IntFieldCompositionFunction>
                        </IntFieldComposition>
                        <IntFieldComposition>
                                <IntFieldCompositionId>146</IntFieldCompositionId>
                                <IntFieldId>115</IntFieldId>
                                <IntFieldCompositionDatasourceId>93</IntFieldComposition
                        DatasourceId>
                                <IntFieldCompositionDatasetId>81</IntFieldCompositionData
                        setId>
                                <IntFieldCompositionFieldId>137</IntFieldCompositionFieldI
                        d>
```

```
                    <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
</IntField>
        <IntField>
                <IntFieldId>116</IntFieldId>
                <IntIDFieldDatabase>2</IntIDFieldDatabase>
                <IntFieldDescription>CARTAO NACIONAL DE
        SAUDE</IntFieldDescription>
                <IntIs_PK>Nao</IntIs_PK>
                <IntIs_Required>Sim</IntIs_Required>
                <IntFormat>999999999999999</IntFormat>
                <IntStandardValue></IntStandardValue>
                <IntSize>15</IntSize>
                <IntUnit>NA</IntUnit>
                <IntFieldIdType>Numero</IntFieldIdType>
                <MetaData>NUMERO DO CNS</MetaData>
                <IntFieldComposition>
                        <IntFieldCompositionId>147</IntFieldCompositionId>
                        <IntFieldId>116</IntFieldId>
                        <IntFieldCompositionDatasourceId>92</IntFieldComposition
                DatasourceId>
                        <IntFieldCompositionDatasetId>80</IntFieldCompositionData
                setId>
                        <IntFieldCompositionFieldId>130</IntFieldCompositionFieldI
                d>
                        <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
                        <IntFieldComposition>
                                <IntFieldCompositionId>148</IntFieldCompositionId
                                >
                                <IntFieldId>116</IntFieldId>
                                <IntFieldCompositionDatasourceId>94</IntFieldCom
                        positionDatasourceId>
                                <IntFieldCompositionDatasetId>82</IntFieldComposi
                        tionDatasetId>
                                <IntFieldCompositionFieldId>146</IntFieldComposit
                        ionFieldId>
                                <IntFieldCompositionFunction>
                        </IntFieldCompositionFunction>
                        </IntFieldComposition>
                        <IntFieldComposition>
                                <IntFieldCompositionId>149</IntFieldCompositionId
                                >
                                <IntFieldId>116</IntFieldId>
                                <IntFieldCompositionDatasourceId>93</IntFieldCom
                        positionDatasourceId>
                                <IntFieldCompositionDatasetId>81</IntFieldComposi
                        tionDatasetId>
                                <IntFieldCompositionFieldId>139</IntFieldComposit
                        ionFieldId>
                                <IntFieldCompositionFunction>
                        </IntFieldCompositionFunction>
                </IntFieldComposition>
</IntField>
        <IntField>
        <IntFieldId>117</IntFieldId>
        <IntIDFieldDatabase>3</IntIDFieldDatabase>
        <IntFieldDescription>NOME</IntFieldDescription>
```

```
        <IntIs_PK>Nao</IntIs_PK>
        <IntIs_Required>Sim</IntIs_Required>
        <IntFormat>!</IntFormat>
        <IntStandardValue></IntStandardValue>
        <IntSize>30</IntSize>
        <IntUnit>NA</IntUnit>
        <IntFieldIdType>Texto</IntFieldIdType>
        <MetaData>NOME DO PACIENTE</MetaData>
        <IntFieldComposition>
                <IntFieldCompositionId>150</IntFieldCompositionId>
                <IntFieldId>117</IntFieldId>
                <IntFieldCompositionDatasourceId>92</IntFieldCompositionDatasourc
        eId>
                <IntFieldCompositionDatasetId>80</IntFieldCompositionDatasetId>
                <IntFieldCompositionFieldId>131</IntFieldCompositionFieldId>
                <IntFieldCompositionFunction> </IntFieldCompositionFunction>
        </IntFieldComposition>
                <IntFieldComposition>
                        <IntFieldCompositionId>151</IntFieldCompositionId>
                        <IntFieldId>117</IntFieldId>
                        <IntFieldCompositionDatasourceId>94</IntFieldComposition
                DatasourceId>
                        <IntFieldCompositionDatasetId>82</IntFieldCompositionData
                setId>
                        <IntFieldCompositionFieldId>145</IntFieldCompositionFieldI
                d>
                        <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
                <IntFieldComposition>
                        <IntFieldCompositionId>152</IntFieldCompositionId>
                        <IntFieldId>117</IntFieldId>
                        <IntFieldCompositionDatasourceId>93</IntFieldComposition
                DatasourceId>
                        <IntFieldCompositionDatasetId>81</IntFieldCompositionData
                setId>
                        <IntFieldCompositionFieldId>138</IntFieldCompositionFieldI
                d>
                        <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
</IntField>
        <IntField>
                <IntFieldId>118</IntFieldId>
                <IntIDFieldDatabase>4</IntIDFieldDatabase>
                <IntFieldDescription>ENDERECO</IntFieldDescription>
                <IntIs_PK>Nao</IntIs_PK>
                <IntIs_Required>Sim</IntIs_Required>
                <IntFormat>!</IntFormat>
                <IntStandardValue>
                </IntStandardValue>
                <IntSize>30</IntSize>
                <IntUnit>NA</IntUnit>
                <IntFieldIdType>Texto</IntFieldIdType>
                <MetaData>LOCALIZACAO RESIDENCIA PACIENTE</MetaData>
                <IntFieldComposition>
                        <IntFieldCompositionId>153</IntFieldCompositionId>
                        <IntFieldId>118</IntFieldId>
                        <IntFieldCompositionDatasourceId>92</IntFieldComposition
                DatasourceId>
```

```
                    <IntFieldCompositionDatasetId>80</IntFieldCompositionData
                setId>
                    <IntFieldCompositionFieldId>132</IntFieldCompositionFieldI
                d>
                    <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
                <IntFieldComposition>
                    <IntFieldCompositionId>154</IntFieldCompositionId>
                    <IntFieldId>118</IntFieldId>
                    <IntFieldCompositionDatasourceId>94</IntFieldComposition
                DatasourceId>
                    <IntFieldCompositionDatasetId>82</IntFieldCompositionData
                setId>
                    <IntFieldCompositionFieldId>148</IntFieldCompositionFieldI
                d>
                    <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
                <IntFieldComposition>
                    <IntFieldCompositionId>155</IntFieldCompositionId>
                    <IntFieldId>118</IntFieldId>
                    <IntFieldCompositionDatasourceId>93</IntFieldComposition
                DatasourceId>
                    <IntFieldCompositionDatasetId>81</IntFieldCompositionData
                setId>
                    <IntFieldCompositionFieldId>140</IntFieldCompositionFieldI
                d>
                    <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
</IntField>
        <IntField>
                <IntFieldId>119</IntFieldId>
                <IntIDFieldDatabase>5</IntIDFieldDatabase>
                <IntFieldDescription>MUNICIPIO</IntFieldDescription>
                <IntIs_PK>Nao</IntIs_PK>
                <IntIs_Required>Sim</IntIs_Required>
                <IntFormat>9999999</IntFormat>
                <IntStandardValue></IntStandardValue>
                <IntSize>10</IntSize>
                <IntUnit>NA</IntUnit>
                <IntFieldIdType>Numero</IntFieldIdType>
                <MetaData>CODIGO BRASILEIRO DE MUNICIPIOS</MetaData>
                <IntFieldComposition>
                    <IntFieldCompositionId>156</IntFieldCompositionId>
                    <IntFieldId>119</IntFieldId>
                    <IntFieldCompositionDatasourceId>92</IntFieldComposition
                DatasourceId>
                    <IntFieldCompositionDatasetId>80</IntFieldCompositionData
                setId>
                    <IntFieldCompositionFieldId>133</IntFieldCompositionFieldI
                d>
                    <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
                    <IntFieldComposition>
                        <IntFieldCompositionId>157</IntFieldCompositionId
                        >
                        <IntFieldId>119</IntFieldId><
```

```
                        IntFieldCompositionDatasourceId>94</IntFieldComp
                    ositionDatasourceId>
                        <IntFieldCompositionDatasetId>82</IntFieldComposi
                    tionDatasetId>
                        <IntFieldCompositionFieldId>149</IntFieldComposit
                    ionFieldId>
                        <IntFieldCompositionFunction>
                    </IntFieldCompositionFunction>
                    </IntFieldComposition>
                    <IntFieldComposition>
                        <IntFieldCompositionId>158</IntFieldCompositionId
                    >
                        <IntFieldId>119</IntFieldId>
                        <IntFieldCompositionDatasourceId>93</IntFieldCom
                    positionDatasourceId>
                        <IntFieldCompositionDatasetId>81</IntFieldComposi
                    tionDatasetId>
                        <IntFieldCompositionFieldId>141</IntFieldComposit
                    ionFieldId>
                        <IntFieldCompositionFunction>
                    </IntFieldCompositionFunction>
                    </IntFieldComposition>
</IntField>
<IntField>
        <IntFieldId>120</IntFieldId>
        <IntIDFieldDatabase>6</IntIDFieldDatabase>
        <IntFieldDescription>CODIGO INTERNACIONAL DE
DOENCA</IntFieldDescription>
        <IntIs_PK>Nao</IntIs_PK>
        <IntIs_Required>Sim</IntIs_Required>
        <IntFormat>!</IntFormat>
        <IntStandardValue></IntStandardValue
        ><IntSize>10</IntSize>
        <IntUnit>NA</IntUnit>
        <IntFieldIdType>Texto</IntFieldIdType>
        <MetaData>CODIGO EXISTENTE NA TABELA CID10</MetaData>
        <IntFieldComposition>
                <IntFieldCompositionId>159</IntFieldCompositionId>
                <IntFieldId>120</IntFieldId>
                <IntFieldCompositionDatasourceId>92</IntFieldCompositionDatasourc
        eId>
                <IntFieldCompositionDatasetId>80</IntFieldCompositionDatasetId>
                <IntFieldCompositionFieldId>134</IntFieldCompositionFieldId>
                <IntFieldCompositionFunction> </IntFieldCompositionFunction>
        </IntFieldComposition>
        <IntFieldComposition>
                <IntFieldCompositionId>160</IntFieldCompositionId>
                <IntFieldId>120</IntFieldId>
                <IntFieldCompositionDatasourceId>94</IntFieldCompositionDatasourc
                eId>
                <IntFieldCompositionDatasetId>82</IntFieldCompositionDatasetId>
                <IntFieldCompositionFieldId>151</IntFieldCompositionFieldId>
                <IntFieldCompositionFunction> </IntFieldCompositionFunction>
        </IntFieldComposition>
        <IntFieldComposition>
                <IntFieldCompositionId>161</IntFieldCompositionId>
                <IntFieldId>120</IntFieldId>
                <IntFieldCompositionDatasourceId>93</IntFieldCompositionDatasourc
        eId>
                <IntFieldCompositionDatasetId>81</IntFieldCompositionDatasetId>
```

```
                <IntFieldCompositionFieldId>144</IntFieldCompositionFieldId>
                <IntFieldCompositionFunction> </IntFieldCompositionFunction>
        </IntFieldComposition>
</IntField>
        <IntField>
                <IntFieldId>121</IntFieldId>
                <IntIDFieldDatabase>7</IntIDFieldDatabase>
                <IntFieldDescription>PROCEDIMENTO SUS</IntFieldDescription>
                <IntIs_PK>Nao</IntIs_PK>
                <IntIs_Required>Sim</IntIs_Required>
                <IntFormat>99999999</IntFormat>
                <IntStandardValue></IntStandardValue>
                <IntSize>10</IntSize>
                <IntUnit>NA</IntUnit>
                <IntFieldIdType>Numero</IntFieldIdType>
                <MetaData>CODIGO DO PROCEDIMENTO EXECUTADO SIA
        SUS</MetaData>
                <IntFieldComposition>
                        <IntFieldCompositionId>162</IntFieldCompositionId>
                        <IntFieldId>121</IntFieldId>
                        <IntFieldCompositionDatasourceId>92</IntFieldComposition
                DatasourceId>
                        <IntFieldCompositionDatasetId>80</IntFieldCompositionData
                setId>
                        <IntFieldCompositionFieldId>136</IntFieldCompositionFieldI
                d>
                        <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
                <IntFieldComposition>
                        <IntFieldCompositionId>163</IntFieldCompositionId>
                        <IntFieldId>121</IntFieldId>
                        <IntFieldCompositionDatasourceId>94</IntFieldComposition
                DatasourceId>
                        <IntFieldCompositionDatasetId>82</IntFieldCompositionData
                setId>
                        <IntFieldCompositionFieldId>152</IntFieldCompositionFieldI
                d>
                        <IntFieldCompositionFunction>
                </IntFieldCompositionFunction
                </IntFieldComposition>
                <IntFieldComposition>
                        <IntFieldCompositionId>164</IntFieldCompositionId>
                        <IntFieldId>121</IntFieldId>
                        <IntFieldCompositionDatasourceId>93</IntFieldComposition
                DatasourceId>
                        <IntFieldCompositionDatasetId>81</IntFieldCompositionData
                setId>
                        <IntFieldCompositionFieldId>143</IntFieldCompositionFieldI
                d>
                        <IntFieldCompositionFunction>
                </IntFieldCompositionFunction>
                </IntFieldComposition>
</IntField>
<IntField>
        <IntFieldId>122</IntFieldId>
        <IntIDFieldDatabase>8</IntIDFieldDatabase>
        <IntFieldDescription>REMUNERACAO</IntFieldDescription>
        <IntIs_PK>Nao</IntIs_PK>
        <IntIs_Required>Sim</IntIs_Required>
```

```
                <IntFormat>99.999,99</IntFormat>
                <IntStandardValue>
                </IntStandardValue>
                <IntSize>10</IntSize>
                <IntUnit>REAL</IntUnit>
                <IntFieldIdType>Numero</IntFieldIdType>
                <MetaData>REMUNERACAO EM REAIS RECEBIDA PELO
        PACIENTE</MetaData
                ><IntFieldComposition>
                        <IntFieldCompositionId>165</IntFieldCompositionId>
                        <IntFieldId>122</IntFieldId>
                        <IntFieldCompositionDatasourceId>92</IntFieldCompositionDatasourc
                eId>
                        <IntFieldCompositionDatasetId>80</IntFieldCompositionDatasetId>
                        <IntFieldCompositionFieldId>135</IntFieldCompositionFieldId>
                        <IntFieldCompositionFunction> </IntFieldCompositionFunction>
                </IntFieldComposition>
                <IntFieldComposition>
                        <IntFieldCompositionId>166</IntFieldCompositionId>
                        <IntFieldId>122</IntFieldId>
                        <IntFieldCompositionDatasourceId>94</IntFieldCompositionDatasourc
                eId>
                        <IntFieldCompositionDatasetId>82</IntFieldCompositionDatasetId>
                        <IntFieldCompositionFieldId>150</IntFieldCompositionFieldId>
                        <IntFieldCompositionFunction> </IntFieldCompositionFunction>
                </IntFieldComposition>
                <IntFieldComposition>
                        <IntFieldCompositionId>168</IntFieldCompositionId>
                        <IntFieldId>122</IntFieldId>
                        <IntFieldCompositionDatasourceId>93</IntFieldCompositionDatasourc
                eId>
                        <IntFieldCompositionDatasetId>81</IntFieldCompositionDatasetId>
                        <IntFieldCompositionFieldId>142</IntFieldCompositionFieldId>
                        <IntFieldCompositionFunction> </IntFieldCompositionFunction>
                </IntFieldComposition>
        </IntField>
</IntDatabase>
```

B.4 Arquivos de Comunicação do Mediador com Serviço de Metadados

B.4.1 – Solicitação do Esquema *IntegraçãoPacientes.xml*

```
<?xml version='1.0' encoding='UTF-8'?>
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'>
                <sqlQueryStatement name='statement'>
                        <expression>Select arquivo from esquemaintegrado where idesquemaintegrado='37'
                        </expression>
                        <webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.4.2 – Solicitação do Esquema *Mauá.xml*

```
<?xml version='1.0' encoding='UTF-8'?>
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'>
                <sqlQueryStatement name='statement'>
                        <expression>Select arquivo from esquemabase where
                idesquemabase='92'</expression><webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.4.3 – Solicitação do Esquema *RibeirãoPires.xml*

```
<?xml version='1.0' encoding='UTF-8'>
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'>
                <sqlQueryStatement name='statement'><expression>Select arquivo from esquemabase where
        idesquemabase='94'</expression><webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.4.4 – Solicitação do Esquema *RioGrandedaSerra.xml*

```
<?xml version='1.0' encoding='UTF-8'>
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'
                <sqlQueryStatement name='statement'>
                expression>Select arquivo from esquemabase where
        idesquemabase='93'</expression><webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.5 Solicitações às Fontes de Dados Locais

B.5.1 – Solicitação a Fonte de Dados Mauá

```
<?xml version='1.0' encoding='UTF-8'?>
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'>
                <sqlQueryStatement name='statement'>
                <expression>SELECT ID_ATEND,CNS,NOME,ENDERECO,COD_MUNIC,CID FROM
        MAUA where CID="H903"</expression><webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.5.2 – Solicitação a Fonte de Dados Ribeirão Pires

```
<?xml version='1.0' encoding='UTF-8'
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'>
                <sqlQueryStatement name='statement'><expression>SELECT
        NAPAC,NCNS,USUARIO,ENDERECO,COD_CIDADE,COD_CID FROM RGS where
        COD_CID="H903"</expression><webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.5.3 – Solicitação a Fonte de Dados Rio Grande da Serra

```
<?xml version='1.0' encoding='UTF-8'
        <perform xmlns='http://ogsadai.org.uk/namespaces/2004/09/service/types'>
                <sqlQueryStatement name='statement'><expression>SELECT
        ATENDIMENT,NCNS,PACIENTE,ENDERECO,CIDADE,COD_DOENCA FROM RPIRES where
        COD_DOENCA="H903"</expression><webRowSetStream name='statementOutput'/>
                </sqlQueryStatement>
        </perform>
```

B.6 Fragmentos dos Arquivos de Respostas

B.6.1 – Fragmento da Fonte de Dados Mauá


```
<?xml version="1.0" encoding="UTF-8"?>
	<OGSADAI>
		<line><column>00102316214</column><column>801434145928380</column><column>FR
	AGMON DA SILVA</column><column>RUA
	CINCO</column><column>355250</column><column>H903</column></line>
		<line><column>00106248065</column><column>801434145927422</column><column>AL
	EXANDRE LOURENCO MALUF</column><column>RUA OLIMPIO DE
	LIMA</column><column>352940</column><column>H903</column></line>
	</OGSADAI>
```


B.6.2 – Fragmento da Fonte de Dados Ribeirão Pires


```
<?xml version="1.0" encoding="UTF-8"?>
	<OGSADAI>
		<line><column>00102316423</column><column>801434145926566</column><column>KA
	HUE AUGUSTO DOS SANTOS SILVA</column><column>RUA LUIZ
	BOTTACIN</column><column>354330</column><column>F848</column><column>38101017</col
	umn><column>200</column></line>
		<line><column>00106249297</column><column>200560832240018</column><column>KA
	IQUE MENDES PEREIRA</column><column>RUA PRIMEIRO DE
	JUNHO</column><column>354330</column><column>F848</column><column>38101017</column
	><column>200</column></line>
		<line><column>00106249308</column><column>898000251473696</column><column>KA
	RINE SANTOS FRANCO</column><column>RUA FRANCISCO
	JARDIM</column><column>352940</column><column>H903</column><column>38101017</colu
	mn><column>200</column></line>
		<line><column>00106249320</column><column>801434145926043</column><column>LA
	RISSA QUERUBINO BRANQUINHO</column><column>RUA SAO CAETANO DO
	SUL</column><column>354330</column><column>F848</column><column>38101017</column><
	column>200</column></line>
		<line>
		<column>00106249330</column><column>206509330140006</column><column>LARISS
	A XAVIER DOS SANTOS</column><column>RUA ISRAEL C. COELHO
	TUSSA</column><column>354410</column><column>H903</column><column>38101017</colum
	n><column>200</column></line>
		<line><column>00106249341</column><column>801434145926604</column><column>LE
	ANDRO GONCALVES CORREIA</column><column>RUA
	RECREIO</column><column>354330</column><column>F848</column><column>38101017</colu
	mn><column>200</column></line>
</OGSADAI>
```


B.6.3 – Fragmento da Fonte de Dados Rio Grande da Serra


```
<?xml version="1.0" encoding="UTF-8"?>
	<OGSADAI>
		<line><column>00102398065</column><column>801434145936715</column><column>AL
	EFF LINNER SILVA MENDES</column><column>RUA HUMBERTO
	SILVERIO</column><column>35433</column><column>H903</column><column>39011038</colu
	mn><column>300</column></line>
		<line><column>00102398076</column><column>203256724280007</column><column>AL
	EXANDRE LOURENCO MARCELINO</column><column>RUA GUIMARAES
	ROSA</column><column>35433</column><column>H903</column><column>39011046</column>
	<column>300</column></line>
		<line><column>00102398087</column><column>801434145937592</column><column>AL
	LAN DE MATOS SILVA</column><column>RUA
```

MACEDONIA</column><column>35441</column><column>H903</column><column>39011038</column><column>300</column></line>
<line><column>00102398098</column><column>801434145937398</column><column>ANA QUEILA DE CARVALHO VIEIRA</column><column>RUA SALDANHA DA GAMA</column><column>35433</column><column>H903</column><column>39011046</column><column>300</column></line>
<line><column>00102398109</column><column>210221320420018</column><column>ANDRESSA DOS REIS RODRIGUES</column><column>RUA BELEM</column><column>35433</column><column>H903</column><column>39011038</column><column>300</column></line>
<line><column>00102398110</column><column>801434145937606</column><column>GESSYANA CRUZ MARTINS</column><column>ESTR. VELHA MOGI SANTOS</column><column>35433</column><column>H903</column><column>39011046</column><column>300</column></line>
</OGSADAI>

B.7 Arquivo de Resposta Integrada (Genérico)

```
<?xml version='1.0' encoding="UTF-8"?>
	<fonteintegrada>
		<metadados>
			<campos>
				<identificacao>1</identificacao>
				<descricao>NUMERO DO ATENDIMENTO</descricao>
				<chaveprimaria>Sim</chaveprimaria>
				<formato>99999999999</formato>
				<unidade>NA</unidade>
				<tipo>Numero</tipo>
				<MetaData>NUMERO DO PROCEDIMENTO DE ALTA
			COMPLEXIDADE</MetaData>
				<identificacao>2</identificacao>
				<descricao>CARTAO NACIONAL DE SAUDE</descricao>
				<chaveprimaria>Nao</chaveprimaria>
				<formato>999999999999999</formato>
				<unidade>NA</unidade>
				<tipo>Numero</tipo>
				<MetaData>NUMERO DO CNS</MetaData>
				<identificacao>3</identificacao>
				<descricao>NOME</descricao>
				<chaveprimaria>Nao</chaveprimaria>
				<formato>!</formato>
				<unidade>NA</unidade>
				<tipo>Texto</tipo>
				<MetaData>NOME DO PACIENTE</MetaData>
				<identificacao>4</identificacao>
				<descricao>ENDERECO</descricao>
				<chaveprimaria>Nao</chaveprimaria>
				<formato>!</formato>
				<unidade>NA</unidade>
				<tipo>Texto</tipo>
				<MetaData>LOCALIZACAO RESIDENCIA PACIENTE</MetaData>
				<identificacao>5</identificacao>
				<descricao>MUNICIPIO</descricao>
				<chaveprimaria>Nao</chaveprimaria>
				<formato>9999999</formato>
				<unidade>NA</unidade>
				<tipo>Numero</tipo>
				<MetaData>CODIGO BRASILEIRO DE MUNICIPIOS</MetaData>
				<identificacao>6</identificacao>
				<descricao>CODIGO INTERNACIONAL DE DOENCA</descricao>
				<chaveprimaria>Nao</chaveprimaria>
				<formato>!</formato>
				<unidade>NA</unidade>
				<tipo>Texto</tipo>
				<MetaData>CODIGO EXISTENTE NA TABELA CID10</MetaData>
				<identificacao>7</identificacao>
				<descricao>PROCEDIMENTO SUS</descricao>
				<chaveprimaria>Nao</chaveprimaria>
				<formato>99999999</formato>
				<unidade>NA</unidade>
				<tipo>Numero</tipo>
				<MetaData>CODIGO DO PROCEDIMENTO EXECUTADO SIA
			SUS</MetaData>
				<identificacao>8</identificacao>
				<descricao>REMUNERACAO</descricao>
				<chaveprimaria>Nao</chaveprimaria>
```

```
                <formato>99.999,99</formato>
                <unidade>REAL</unidade>
                <tipo>Numero</tipo>
                <MetaData>REMUNERACAO EM REAIS RECEBIDA PELO
        PACIENTE</MetaData>
        </campos>
</metadados>
<valores>
        <linha><coluna>00102316192</coluna><coluna>801434145927414</coluna><colun
a>ANTONIO ALMEIDA CARDOSO</coluna><coluna>RUA PAULO
LORO</coluna><coluna>352940</coluna><coluna>F841</coluna><coluna>38101017</colu
na><coluna>1,00</coluna></linha>
        <linha><coluna>00102316203</coluna><coluna>801434145928259</coluna><colun
a>CASSIO DOS SANTOS ROSA</coluna><coluna>RUA
ALASCA</coluna><coluna>352940</coluna><coluna>F841</coluna><coluna>38101017</c
oluna><coluna>1,00</coluna></linha>
        <linha><coluna>00102316214</coluna><coluna>801434145928380</coluna><colun
a>FRAGMON DA SILVA</coluna><coluna>RUA
CINCO</coluna><coluna>355250</coluna><coluna>H903</coluna><coluna>38101017</col
una><coluna>1,00</coluna></linha>
        <linha><coluna>00102316423</coluna><coluna>801434145926566</coluna><colun
a>KAHUE AUGUSTO DOS SANTOS SILVA</coluna><coluna>RUA LUIZ
BOTTACIN</coluna><coluna>354330</coluna><coluna>F848</coluna><coluna>38101017
</coluna><coluna>2,00</coluna></linha>
        <linha><coluna>00102398065</coluna><coluna>801434145936715</coluna><colun
a>ALEFF LINNER SILVA MENDES</coluna><coluna>RUA HUMBERTO
SILVERIO</coluna><coluna>35433</coluna><coluna>H903</coluna><coluna>39011038</c
oluna><coluna>3,00</coluna></linha>
        <linha><coluna>00102398076</coluna><coluna>203256724280007</coluna><colun
a>ALEXANDRE LOURENCO MARCELINO</coluna><coluna>RUA GUIMARAES
ROSA</coluna><coluna>35433</coluna><coluna>H903</coluna><coluna>39011046</colun
a><coluna>3,00</coluna></linha>
        <linha><coluna>00102398087</coluna><coluna>801434145937592</coluna><colun
a>ALLAN DE MATOS SILVA</coluna><coluna>RUA
MACEDONIA</coluna><coluna>35441</coluna><coluna>H903</coluna><coluna>3901103
8</coluna><coluna>3,00</coluna></linha>
        <linha><coluna>00102398098</coluna><coluna>801434145937398</coluna><colun
a>ANA QUEILA DE CARVALHO VIEIRA</coluna><coluna>RUA SALDANHA DA
GAMA</coluna><coluna>35433</coluna><coluna>H903</coluna><coluna>39011046</colu
na><coluna>3,00</coluna></linha>
        <linha><coluna>00102398109</coluna><coluna>210221320420018</coluna><colun
a>ANDRESSA DOS REIS RODRIGUES</coluna><coluna>RUA
BELEM</coluna><coluna>35433</coluna><coluna>H903</coluna><coluna>39011038</col
una><coluna>3,00</coluna></linha>
        <linha><coluna>00102398110</coluna><coluna>801434145937606</coluna><colun
a>GESSYANA CRUZ MARTINS</coluna><coluna>ESTR. VELHA MOGI
SANTOS</coluna><coluna>35433</coluna><coluna>H903</coluna><coluna>39011046</co
luna><coluna>3,00</coluna></linha>
        <linha><coluna>00106248043</coluna><coluna>206107629010002</coluna><colun
a>ALEX FAGUNDES FARIA</coluna><coluna>RUA ROMULO
LORENZONE</coluna><coluna>354330</coluna><coluna>F841</coluna><coluna>3810101
7</coluna><coluna>1,00</coluna></linha>
        <linha><coluna>00106248065</coluna><coluna>801434145927422</coluna><colun
a>ALEXANDRE LOURENCO MALUF</coluna><coluna>RUA OLIMPIO DE
LIMA</coluna><coluna>352940</coluna><coluna>H903</coluna><coluna>38101017</colu
na><coluna>1,00</coluna></linha>
        <linha><coluna>00106248076</coluna><coluna>801434145927406</coluna><colun
a>ALINE MOSCATELLI CARCILLO</coluna><coluna>RUA ARGELINO F.
GIANASI</coluna><coluna>354330</coluna><coluna>F841</coluna><coluna>38101017</c
oluna><coluna>1,00</coluna></linha>
```

```
        <linha><coluna>00106249297</coluna><coluna>200560832240018</coluna><colun
    a>KAIQUE MENDES PEREIRA</coluna><coluna>RUA PRIMEIRO DE
    JUNHO</coluna><coluna>354330</coluna><coluna>F848</coluna><coluna>38101017</col
    una><coluna>2,00</coluna></linha>
        <linha><coluna>00106249308</coluna><coluna>898000251473696</coluna><colun
    a>KARINE SANTOS FRANCO</coluna><coluna>RUA FRANCISCO
    JARDIM</coluna><coluna>352940</coluna><coluna>H903</coluna><coluna>38101017</c
    oluna><coluna>2,00</coluna></linha>
        <linha><coluna>00106249320</coluna><coluna>801434145926043</coluna><colun
    a>LARISSA QUERUBINO BRANQUINHO</coluna><coluna>RUA SAO CAETANO DO
    SUL</coluna><coluna>354330</coluna><coluna>F848</coluna><coluna>38101017</coluna
    ><coluna>2,00</coluna></linha>
        <linha><coluna>00106249330</coluna><coluna>206509330140006</coluna><colun
    a>LARISSA XAVIER DOS SANTOS</coluna><coluna>RUA ISRAEL C. COELHO
    TUSSA</coluna><coluna>354410</coluna><coluna>H903</coluna><coluna>38101017</col
    una><coluna>2,00</coluna></linha>
        <linha><coluna>00106249341</coluna><coluna>801434145926604</coluna><colun
    a>LEANDRO GONCALVES CORREIA</coluna><coluna>RUA
    RECREIO</coluna><coluna>354330</coluna><coluna>F848</coluna><coluna>38101017</c
    oluna><coluna>2,00</coluna></linha>
    </valores>
</fonteintegrada>
```

B.8 Arquivo de Resposta Integrada (Borland Delphi)

```
<?xml version="1.0" standalone="yes"?>
	<DATAPACKET Version="2.0">
		<METADATA>
			<FIELDS>
				<FIELD attrname="NUMERO DO ATENDIMENTO"  fieldtype="string"
			WIDTH="15"/>
				<FIELD attrname="CARTAO NACIONAL DE SAUDE"  fieldtype="string"
			WIDTH="15"/>
				<FIELD attrname="NOME"  fieldtype="string" WIDTH="30"/>
				<FIELD attrname="ENDERECO"  fieldtype="string" WIDTH="30"/>
				<FIELD attrname="MUNICIPIO"  fieldtype="string" WIDTH="10"/>
				<FIELD attrname="CODIGO INTERNACIONAL DE DOENCA"
			fieldtype="string" WIDTH="10"/>
				<FIELD attrname="PROCEDIMENTO SUS"  fieldtype="string"
			WIDTH="10"/>
				<FIELD attrname="REMUNERACAO"  fieldtype="string" WIDTH="10"/>
			</FIELDS>
			<PARAMS DEFAULT_ORDER="1" PRIMARY_KEY="1"/>
		</METADATA>
		<ROWDATA>
			<ROW NUMERO DO ATENDIMENTO="00102316192" CARTAO NACIONAL
		DE SAUDE="801434145927414" NOME="ANTONIO ALMEIDA CARDOSO"
		ENDERECO="RUA PAULO LORO" MUNICIPIO="352940" CODIGO INTERNACIONAL
		DE DOENCA="F841" PROCEDIMENTO SUS="38101017" REMUNERACAO="1,00" />
			<ROW NUMERO DO ATENDIMENTO="00102316203" CARTAO NACIONAL
		DE SAUDE="801434145928259" NOME="CASSIO DOS SANTOS ROSA"
		ENDERECO="RUA ALASCA" MUNICIPIO="352940" CODIGO INTERNACIONAL DE
		DOENCA="F841" PROCEDIMENTO SUS="38101017" REMUNERACAO="1,00" />
			<ROW NUMERO DO ATENDIMENTO="00102316214" CARTAO NACIONAL
		DE SAUDE="801434145928380" NOME="FRAGMON DA SILVA" ENDERECO="RUA
		CINCO" MUNICIPIO="355250" CODIGO INTERNACIONAL DE DOENCA="H903"
		PROCEDIMENTO SUS="38101017" REMUNERACAO="1,00" />
			<ROW NUMERO DO ATENDIMENTO="00102316423" CARTAO NACIONAL
		DE SAUDE="801434145926566" NOME="KAHUE AUGUSTO DOS SANTOS SILVA"
		ENDERECO="RUA LUIZ BOTTACIN" MUNICIPIO="354330" CODIGO
		INTERNACIONAL DE DOENCA="F848" PROCEDIMENTO SUS="38101017"
		REMUNERACAO="2,00" />
			<ROW NUMERO DO ATENDIMENTO="00102398065" CARTAO NACIONAL
		DE SAUDE="801434145936715" NOME="ALEFF LINNER SILVA MENDES"
		ENDERECO="RUA HUMBERTO SILVERIO" MUNICIPIO="35433" CODIGO
		INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="39011038"
		REMUNERACAO="3,00" />
			<ROW NUMERO DO ATENDIMENTO="00102398076" CARTAO NACIONAL
		DE SAUDE="203256724280007" NOME="ALEXANDRE LOURENCO MARCELINO"
		ENDERECO="RUA GUIMARAES ROSA" MUNICIPIO="35433" CODIGO
		INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="39011046"
		REMUNERACAO="3,00" />
			<ROW NUMERO DO ATENDIMENTO="00102398087" CARTAO NACIONAL
		DE SAUDE="801434145937592" NOME="ALLAN DE MATOS SILVA"
		ENDERECO="RUA MACEDONIA" MUNICIPIO="35441" CODIGO INTERNACIONAL
		DE DOENCA="H903" PROCEDIMENTO SUS="39011038" REMUNERACAO="3,00" />
			<ROW NUMERO DO ATENDIMENTO="00102398098" CARTAO NACIONAL
		DE SAUDE="801434145937398" NOME="ANA QUEILA DE CARVALHO VIEIRA"
		ENDERECO="RUA SALDANHA DA GAMA" MUNICIPIO="35433" CODIGO
		INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="39011046"
		REMUNERACAO="3,00" />
			<ROW NUMERO DO ATENDIMENTO="00102398109" CARTAO NACIONAL
		DE SAUDE="210221320420018" NOME="ANDRESSA DOS REIS RODRIGUES"
```

ENDERECO="RUA BELEM" MUNICIPIO="35433" CODIGO INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="39011038" REMUNERACAO="3,00" />
<ROW NUMERO DO ATENDIMENTO="00102398110" CARTAO NACIONAL DE SAUDE="801434145937606" NOME="GESSYANA CRUZ MARTINS" ENDERECO="ESTR. VELHA MOGI SANTOS" MUNICIPIO="35433" CODIGO INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="39011046" REMUNERACAO="3,00" />
<ROW NUMERO DO ATENDIMENTO="00106248043" CARTAO NACIONAL DE SAUDE="206107629010002" NOME="ALEX FAGUNDES FARIA" ENDERECO="RUA ROMULO LORENZONE" MUNICIPIO="354330" CODIGO INTERNACIONAL DE DOENCA="F841" PROCEDIMENTO SUS="38101017" REMUNERACAO="1,00" />
<ROW NUMERO DO ATENDIMENTO="00106248065" CARTAO NACIONAL DE SAUDE="801434145927422" NOME="ALEXANDRE LOURENCO MALUF" ENDERECO="RUA OLIMPIO DE LIMA" MUNICIPIO="352940" CODIGO INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="38101017" REMUNERACAO="1,00" />
<ROW NUMERO DO ATENDIMENTO="00106248076" CARTAO NACIONAL DE SAUDE="801434145927406" NOME="ALINE MOSCATELLI CARCILLO" ENDERECO="RUA ARGELINO F. GIANASI" MUNICIPIO="354330" CODIGO INTERNACIONAL DE DOENCA="F841" PROCEDIMENTO SUS="38101017" REMUNERACAO="1,00" />
<ROW NUMERO DO ATENDIMENTO="00106249297" CARTAO NACIONAL DE SAUDE="200560832240018" NOME="KAIQUE MENDES PEREIRA" ENDERECO="RUA PRIMEIRO DE JUNHO" MUNICIPIO="354330" CODIGO INTERNACIONAL DE DOENCA="F848" PROCEDIMENTO SUS="38101017" REMUNERACAO="2,00" />
<ROW NUMERO DO ATENDIMENTO="00106249308" CARTAO NACIONAL DE SAUDE="898000251473696" NOME="KARINE SANTOS FRANCO" ENDERECO="RUA FRANCISCO JARDIM" MUNICIPIO="352940" CODIGO INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="38101017" REMUNERACAO="2,00" />
<ROW NUMERO DO ATENDIMENTO="00106249320" CARTAO NACIONAL DE SAUDE="801434145926043" NOME="LARISSA QUERUBINO BRANQUINHO" ENDERECO="RUA SAO CAETANO DO SUL" MUNICIPIO="354330" CODIGO INTERNACIONAL DE DOENCA="F848" PROCEDIMENTO SUS="38101017" REMUNERACAO="2,00" />
<ROW NUMERO DO ATENDIMENTO="00106249330" CARTAO NACIONAL DE SAUDE="206509330140006" NOME="LARISSA XAVIER DOS SANTOS" ENDERECO="RUA ISRAEL C. COELHO TUSSA" MUNICIPIO="354410" CODIGO INTERNACIONAL DE DOENCA="H903" PROCEDIMENTO SUS="38101017" REMUNERACAO="2,00" />
<ROW NUMERO DO ATENDIMENTO="00106249341" CARTAO NACIONAL DE SAUDE="801434145926604" NOME="LEANDRO GONCALVES CORREIA" ENDERECO="RUA RECREIO" MUNICIPIO="354330" CODIGO INTERNACIONAL DE DOENCA="F848" PROCEDIMENTO SUS="38101017" REMUNERACAO="2,00" />
</ROWDATA>
</DATAPACKET>