O Uso do Framework NFR no Projeto de Banco de Dados Distribuído

Márcia Santos, Maria Lencastre, Jaelson Brelaz de Castro[1], Décio Fonseca

Centro de Informática - Universidade Federal de Pernambuco
Av. Professor Luís Freire s/n Cidade Universitária
E-mail{mrfps, mlpm, jbc, decio}@cin.ufpe.br

Resumo. O projeto de banco de dados distribuído é um processo bastante complexo que envolve aspectos distintos para a realização de uma adequada distribuição dos dados (Buretta 1997). Muitos desses aspectos correspondem a requisitos não funcionais (propriedades de qualidade ou restrições dos sistemas), como por exemplo, disponibilidade, custos e desempenho. Esses requisitos normalmente são pouco explorados nas etapas do projeto, deixando assim de auxiliar no processo de distribuição. Este artigo tem como objetivo representar os requisitos não funcionais nas fases iniciais do projeto de banco de dados distribuído, através da integração de estratégias propostas pela área de Engenharia de Requisitos. O trabalho foca em especial o uso do *Framework NFR* (Chung et al. 2000) e faz uma extensão de seus catálogos de requisitos não funcionais, a fim de integrar os principais aspectos relacionados com a distribuição dos dados.

Abstract. Distributed database design is a complex process that involves a set of distinct aspects for the accomplishment of an adequate data distribution (Buretta 1997). Many of these aspects correspond to non functional requirements (quality properties or systems restrictions), for example, availability, costs and performance, that are abstract questions. Non functional requirements are normally not explored in distributed databases design phases, giving a little contribution to the distribution process. This article aims to represent non functional requirements in the initial phases of distributed database design through the integration of strategies proposed by Requirements Engineering area. This work emphasizes the use of *Framework NFR* (Chung et al. 2000) and makes an extension of its non functional requirements catalogues in order to integrate the major aspects related with data distribution.

Keywords: Engenharia de Requisitos, Framework NFR, projeto de banco de dados distribuídos

[1] Este trabalho foi realizado durante a vista do autor ao Department of Computer Science, University of Toronto. O trabalho foi parcialmente apoiado pelo CNPq Proc. 203262/86-7 (NV)

1 Introdução

O projeto de banco de dados distribuído é uma importante área de pesquisa que tem como objetivo garantir uma adequada distribuição dos dados, visando atingir vantagens como autonomia, melhoria de desempenho, disponibilidade e confiabilidade para os sistemas de banco de dados distribuído. Esse objetivo é basicamente alcançado através da eliminação de dados irrelevantes acessados pelas aplicações e pela redução da transferência de dados entre os *sites* da rede de computadores (Lencastre et al. 1999). Geralmente, as principais questões envolvidas com essa área estão relacionadas com as técnicas de alocação, fragmentação e replicação dos dados. A fragmentação faz o particionamento dos dados, formando fragmentos que são unidades lógicas de alocação, ou seja, pontos de partida apropriados para o problema posterior de alocação dos dados. A alocação armazena os conjuntos de dados, particionados ou não, nos *sites* onde as aplicações, que os acessam, são ativadas com maior freqüência. Já a replicação consiste na alocação de um fragmento em mais de um *site*.

O projeto de banco de dados em um ambiente distribuído é significativamente mais complexo do que em um ambiente centralizado, uma vez que envolve questões sobre recursos de rede, esquemas de particionamento de dados e alternativas de armazenamento de dados redundante. Além dessas questões, este tipo de projeto é fortemente influenciado por fatores como disponibilidade, segurança e custos, denominados pela área de Engenharia de Requisitos de requisitos não funcionais. Estes requisitos influenciam o projeto de distribuição dos dados, porém são pouco tratados/representados durante as diversas fases do projeto. Na maioria das abordagens para projeto de sistemas, os desenvolvedores focam seus esforços inicialmente para alcançar a funcionalidade do sistema. Assim, os requisitos não funcionais são vistos como "conseqüências" das decisões, e não como algo que o desenvolvedor possa empenhar-se de uma maneira coerente.

A não consideração dos requisitos não funcionais na definição requisitos constitui uma das principais razões de insatisfação do usuário com relação ao produto final, principalmente em ambientes distribuídos, onde este tipo de requisitos influencia a especificação final do produto. De acordo com Cysneiros (1997), esses requisitos são, quando muito, simplesmente listados e vagamente observados, sendo, em geral, esquecidos durante a fase de especificação de sistemas. Segundo Chung et al. (2000) e Mylopoulos et al. (1999), os requisitos não funcionais (RNFs), por sua natureza, são difíceis de serem tratados durante o projeto e a implementação, são difíceis de validar, freqüentemente são descritos de forma breve e ambígua, interagem com outros requisitos não funcionais e possuem um impacto global no sistema. Dessa forma, o objetivo deste artigo consiste em representar e tratar os requisitos não funcionais, que influenciam o projeto de banco de dados distribuído, através de modelos já existentes, facilitando o processo como um todo.

A seção 2, apresenta alguns trabalhos sobre os requisitos não funcionais. Na seção 3 é apresentada uma metodologia de suporte ao projeto distribuído, (Santos

2000), onde é focalizada a extensão aos catálogos do *Framework NFR*, que corresponde a um dos modelos escolhidos, pelo presente trabalho, para modelagem de aspectos não funcionais que influenciam a distribuição dos dados. A seção 4 apresenta um exemplo da utilização dessa abordagem, e por fim, na seção 5, são feitas algumas considerações relacionadas com o estudo desenvolvido.

2 Trabalhos que Tratam Requisitos Não Funcionais

Alguns trabalhos (Buretta 1997, Dobson e McDermid 1991, Cysneiros e Leite 1997, Chung et al. 2000) já consideram os requisito não funcionais como relevantes em suas abordagens, entre eles temos:

a) (Buretta 1997), por exemplo, apresenta uma metodologia que define diretrizes gerais para tornar a tarefa de distribuição dos dados mais fácil e com maior garantia de qualidade. Ela relaciona algumas informações relativas ao contexto distribuído, dentre as quais destaca-se a disponibilidade, segurança e volume de dados, todos requisitos não funcionais. Apesar de se preocupar com esse tipo de informação, essa metodologia não apresenta modelos integrados com a Engenharia de Requisitos para auxiliar na sua representação, sendo na sua maioria descrições textuais na forma de tabelas.

b) O trabalho apresentado em (Dobson e McDermid 1991), por sua vez, segue a linha do RM-ODP (Draft Recomendation 1995), que tem como objetivo projetar o sistema distribuído a partir de cinco pontos de vista (empresa, informação, computação, engenharia e tecnologia). A abordagem (Dobson e McDermid 1991) identifica requisitos não funcionais nas diversas projeções do padrão RM-ODP, em especial nas projeções de empresa e de computação, correspondendo, assim, a uma tentativa de incorporar esse tipo de requisito à modelagem de ambientes distribuídos. Uma deficiência desse trabalho é o fato dele não apresentar modelos para essa integração.

c) Uma outra abordagem, preocupada em retratar os requisitos não funcionais, é apresentada em (Cysneiros 1997, Cysneiros e Leite 1997). Esta abordagem engloba a construção de um léxico, definição do modelo de dados e a construção dos grafos de RNFs. A proposta integra os requisitos não funcionais com o modelo Entidade-Relacionamento, utilizando uma adaptação do grafo de requisitos não funcionais, proposto por Chung. Uma continuação desse trabalho, descrito em Cysneiros et al. (1999), faz a integração dos RNFs com o modelo orientado a objetos.

d) Uma forma muito utilizada, na representação dos requisitos não funcionais, é a proposta de Chung et al. (2000). O trabalho define um framework - chamado Framework NFR - para representar requisitos não funcionais, onde estes são vistos como softgoals (objetivos que não possuem uma definição clara para garantir que foram satisfeitos ou não). Em contraste com as abordagens voltadas à funcionalidade, o Framework NFR usa os requisitos não funcionais para guiar todo o processo de projeto e oferece uma estrutura - Gráficos de Interdependência de Softgoals (SIG) - para representar e armazenar os passos e raciocínios do projeto. Um

gráfico SIG não representa apenas um meio de tratar requisitos não funcionais, mas pode ser visto como um registro do histórico do desenvolvimento que pode ser usado para posterior revisão, justificativa e mudanças. Além do gráfico, o Framework NFR oferece também catálogos (de tipos, métodos e interdependências) que são usados para expressar, previamente, o conhecimento sobre os RNFs e assim servirem como apoio para a posterior definição de SIGs.

3 Modelos da Engenharia de Requisitos como Apoio ao Projeto de BDD

Em (Santos 2000) foi proposta uma metodologia, fundamentada em algumas das técnicas usadas na Engenharia de Requisitos, para apoiar a fase de definição de requisitos do projeto de banco de dados distribuído. O trabalho de Santos (2000) está relacionado com um projeto maior, que engloba os trabalhos de Barroso (1998) e Lencastre (2000), e visa a realização de um projeto de banco de dados distribuído mais bem fundamentado e completo. A extensão dos catálogos, que será apresentada neste artigo, serve como suporte a essa metodologia.

A metodologia descrita em (Santos 2000) busca garantir, através da integração dos modelos mais difundidos e de ferramentas disponíveis, que a informação essencial a todo o projeto vai ser identificada no mais alto nível e projetada, posteriormente, entre as diversas fases do projeto de banco de dados distribuído. Na Fig.1, cada plano corresponde a uma fase do projeto. Nela são sugeridos os modelos (representados por elipses) para as fases de definição de requisitos e projeto conceitual, uma vez que estes constituem o foco deste trabalho.

Os requisitos podem ser classificados em tipos distintos, entretanto essa classificação pode variar de autor para autor (Hofmann 1993, Loucopoulos e Karakostas 1995), não existindo um consenso a ser seguido. No trabalho foi considerada a abordagem de Loucopoulos e Karakostas (1995), onde os requisitos são subdivididos em requisitos organizacionais, funcionais e não funcionais. Cada um desses requisitos preocupa-se com um aspecto distinto do problema a ser analisado e têm uma importância bem definida na análise e projeto de distribuição dos dados (Santos 2000).

Na metodologia o primeiro nível - Definição de Requisitos – é responsável: pelos requisitos da organização e do software. Dessa forma engloba: a definição do ambiente da organização, realizada através do Modelo Organizacional (Bubenko 1994a e Bubenko 1994b); pela obtenção dos requisitos não funcionais através do Framework NFR (Chung et al. 2000); e pela representação dos requisitos funcionais com a utilização do diagrama de Use Cases da UML (Booch et al. 1999).

Conforme descrito em (Santos 2000) as informações obtidas na definição de requisitos são essenciais para um projeto distribuído porém, na maioria das vezes, elas são pouco exploradas. Após serem coletadas, essas informações servem como entrada para os níveis seguintes – Projeto Conceitual e Projeto de Distribuição.

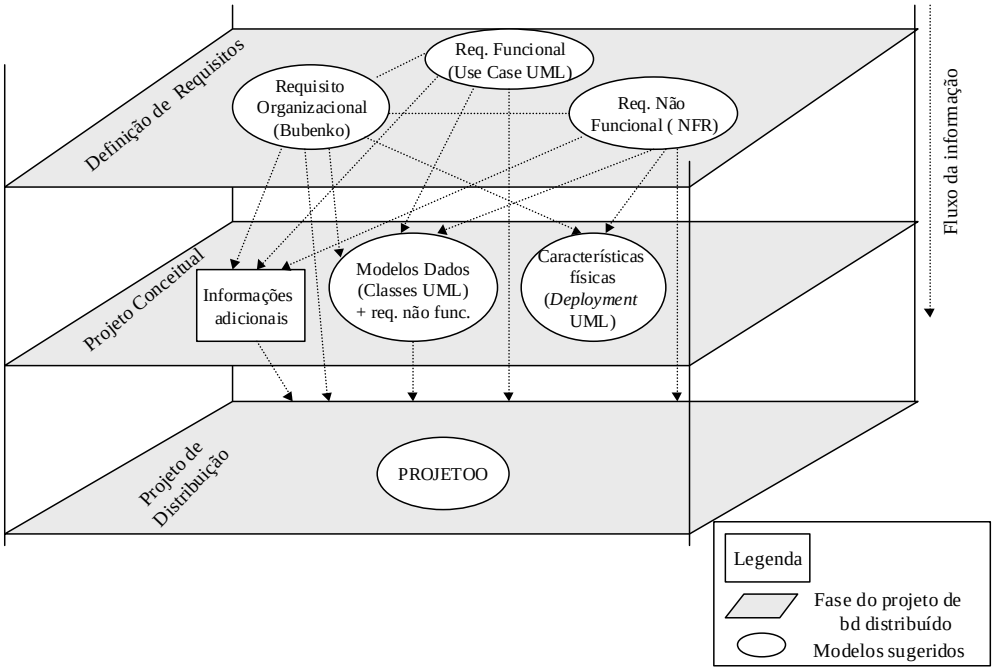

Fig.1. Fases da metodologia proposta para apoio ao projeto de banco de dados distribuído

No nível do Projeto Conceitual encontra-se o diagrama de classes da UML (Booch et al. 1999) que corresponde aos objetos a serem fragmentados, alocados ou replicados. A este diagrama foram incorporadas informações relativas aos requisitos não funcionais, seguindo a proposta de (Cysneiros et al. 1999). Um outro modelo corresponde ao diagrama *deployment* (Booch et al. 1999), o qual permite capturar informações relacionadas com o ambiente físico utilizado, tais como velocidade de comunicação e capacidade de armazenamento das máquinas. Essas informações são importantes pois auxiliam na determinação dos melhores locais onde os dados podem ser armazenados. Na fase de Projeto Conceitual, encontram-se também informações relacionadas às estatísticas e uso das aplicações (representadas por um retângulo), extremamente úteis para a distribuição dos dados, porém não foram encontrados, na Engenharia de Requisitos, modelos que as representassem.

O nível de Projeto de Distribuição, por sua vez, recebe as informações oriundas das etapas anteriores. Neste nível encontra-se o trabalho de Barroso (1998) o qual apresenta uma metodologia - PROJETOO - destinada à fase de fragmentação e alocação dos dados. O PROJETOO considera informações relativas a: *sites*, aplicações disparadas em cada *site*, freqüência de acesso; e os requisitos não funcionais, referentes a confiabilidade, disponibilidade e custos. Esta fase do projeto não é tratada neste trabalho, uma vez que é o foco do trabalho de Barroso (1998) e Lencastre (2000).

A seguir vamos apresentar o funcionamento do *Framework NFR* e a extensão realizada nos seus catálogos de Requisitos Não Funcionais. A definição prévia desses catálogos serve como apoio para a identificação das melhores estratégias a serem utilizadas na definição dos SIGs, no caso especial deste trabalho, durante o projeto de sistemas de banco de dados distribuído.

3.1 O Framework NFR

A utilização do *Framework NFR* (Chung et al. 2000) consiste em uma série de passos, através dos quais os requisitos não funcionais são identificados, refinados, correlacionados com outros requisitos e operacionalizados. Esses passos ajudam a decompor os requisitos não funcionais, além de tratar ambigüidades e outros conflitos de especificação. De um modo geral, pode-se considerar que o *framework* possui três fases:

1. A **criação prévia de catálogos** relacionados com requisitos não funcionais, que servem para expressar o conhecimento sobre os RNFs;

- Catálogos de tipos de RNFs (Requisitos Não Funcionais): são usados para fornecer uma terminologia e classificação de conceitos dos RNFs;
- Catálogos de métodos: possuem informações que ajudam a refinar os gráficos através da decomposição de *softgoals* e operacionalizações (estratégias de implementação);
- Catálogos de correlação: possuem conhecimento que ajudam a detectar interdependências implícitas entre os RNFs.

2. A **definição dos gráficos relacionados com o problema** em questão e,
3. A **seleção de alternativas e avaliação do impacto das decisões no problema sendo tratado**.

O presente trabalho optou por representar os requisitos não funcionais através do *Framework NFR*, por ser uma técnica abrangente e bem definida, que auxilia na captura e representação dos requisitos não funcionais, além de facilitar a decisão entre as diversas alternativas integradas a cada requisito em particular.

3.2 Extensão dos Catálogos do *Framework* NFR

(Chung et al. 2000) trata especificamente os requisitos não funcionais *precisão*, *segurança* e *desempenho*, em seus catálogos pré-definidos. Em virtude do *framework* permitir a extensão de seus catálogos, neste artigo, foram identificadas e acrescentadas informações relacionadas ao projeto distribuído, tomando-se como base diversos trabalhos da área de banco de dados (Barroso 1998, Buretta 1997, Korth 1995, Pires 1997, Özsu 1991) e de outros trabalhos que tratam dos sistemas distribuídos (Coulouris et al. 1996, Tanenbaum 1995). Dessa forma, foram incluídos

nos catálogos os requisitos não funcionais: **desempenho**, **disponibilidade** e **confiabilidade**, por corresponderem aos principais objetivos do projeto de distribuição (Barroso 1998). Além disso foram incluídos, também, os requisitos não funcionais **sobrecarga de controle de distribuição** e os **custos**, por serem requisitos que influenciam, de forma significativa, o processo de distribuição dos dados.

Os catálogos têm como objetivo representar o conhecimento sobre os requisitos não funcionais e sobre o projeto (incluindo técnicas de desenvolvimento), acumulado em experiências anteriores, levando o projetista a verificar como os requisitos não funcionais podem se relacionar com requisitos funcionais, além de mostrar como eles podem ser influenciados por outros requisitos não funcionais. Assim, a introdução de novas informações, relacionadas com os aspectos de distribuição, é bastante relevante, uma vez que os requisitos que influenciam a fase do projeto de distribuição de banco de dados podem ser tratados logo no início do projeto

A seguir, serão descritos os passos, estabelecidos pelo *Framework NFR*, relacionados com a integração dessas novas informações aos catálogos.

Passo 1: Acrescentar requisitos não funcionais aos catálogos

Passo 2: Definição dos catálogos de métodos;

Passo 3: Definição dos catálogos de interdependências.

Inicialmente, será descrito um catálogo, contendo as informações que foram acrescentadas, bem como algumas que já faziam parte do *Framework NFR* e que foram consideradas relevantes ao projeto de banco de dados distribuído. Em seguida serão representados os métodos de operacionalização de alguns dos requisitos especificados e serão estabelecidos os inter-relacionamentos entre alguns desses requisitos. O uso das informações representadas nesses catálogos será apresentado, através de um exemplo, na seção 4 deste trabalho.

3.2.1 Acrescentando informações aos catálogos

Tomando-se como base o *Framework NFR* (Chung et al. 2000), tem-se que, inicialmente, os requisitos não funcionais mais gerais são decompostos em vários outros requisitos, onde cada um deles também pode ter suas subdivisões. O catálogo da Fig.2 apresenta a hierarquia de alguns requisitos não funcionais, onde os requisitos mais gerais aparecem acima dos mais específicos.

De acordo com essa Fig. 2, existe um conjunto de requisitos não funcionais que estão diretamente relacionados com o projeto de banco de dados distribuído, tais como **desempenho**, **disponibilidade**, **confiabilidade**, **sobrecarga de controle de distribuição**, **custos**, **volume de dados**. Alguns desses requisitos já foram definidos em (Chung et al. 2000). Assim, para permitir uma maior diferenciação entre eles, os requisitos que foram introduzidos neste catálogo, pelo presente trabalho, e que não foram tratados em (Chung et al. 2000), são representados em negrito.

A definição de cada um dos requisitos não funcionais introduzidos é apresentada a seguir para, posteriormente no passo 2 desta etapa (seção 3.2.2), servirem como base para a definição dos respectivos catálogos de métodos.

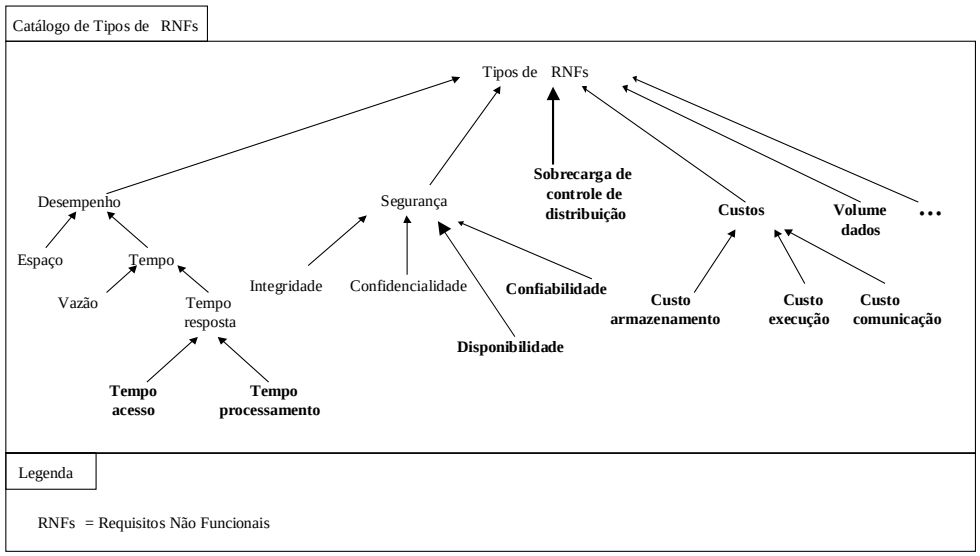

Fig. 2. Catálogo de Tipos incluindo RNFs propostos

a) Tempos de Acesso e Processamento

O desempenho é um fator de qualidade vital para qualquer sistema (Chung et al. 2000). Este requisito possui ainda uma maior ênfase no projeto de banco de dados distribuído, uma vez que a maioria dos sistemas distribuídos são críticos e complexos, sendo o desempenho dificultado pela existência de diferentes localidades, conexões, etc. Para que o desempenho seja alcançado, é necessário considerar outros requisitos como por exemplo, espaço e tempo, ou seja, deve-se diminuir/reduzir o tempo e o espaço de armazenamento a fim de se obter um melhor desempenho. O tempo por sua vez pode ser decomposto em vazão e tempo de resposta, no caso relativo às aplicações. Dessa forma, para melhorar o tempo, é necessário diminuir o tempo de resposta e aumentar a vazão (que corresponde a capacidade de transmitir mais informações por unidade de tempo). O tempo de resposta - intervalo de tempo necessário para que os resultados de uma aplicação estejam disponíveis - é decomposto em **tempo de acesso** – intervalo de tempo necessário para que o SGBDD localize e recupere os dados requisitados por uma determinada aplicação e **tempo de processamento** – intervalo de tempo para a realização do processamento.

b) Confiabilidade e Disponibilidade

Quanto à segurança, ela é tratada como um problema de difícil solução em um ambiente distribuído (Pires 1997). Alguns dos problemas incluem canais de comunicação inseguros, diferentes níveis de segurança para diferentes nós e o grande

número de usuários do sistema global. Este requisito é uma importante função do sistema de banco de dados e tem como objetivo proteger os dados contra acessos não autorizados. (Chung et al. 2000) apresenta um catálogo detalhado sobre a segurança, a qual envolve diferentes aspectos, como integridade - proteger contra uso ou atualização não autorizada - e confidencialidade - proteger contra revelação não autorizada. Além desses dois aspectos, este trabalho introduz outros dois – **confiabilidade** e **disponibilidade**.

A **disponibilidade**, no contexto de banco de dados distribuído, corresponde à garantia de que se pode fazer acesso aos dados apesar da falha em alguns nós (Korth 1995). Segundo Tanenbaum (1995), a disponibilidade refere-se à fração do tempo em que o sistema pode ser utilizado plenamente. Existem vários fatores associados à disponibilidade, tais como: a topologia, a qualidade e velocidade da rede, o local do processamento (estação cliente/servidor), o desempenho do SGBDD e a aplicação propriamente dita.

A **Confiabilidade**, por sua vez, é a capacidade do sistema se comportar de acordo com sua especificação, durante um certo período de tempo, de forma aceitável e num ambiente apropriado. A **confiabilidade**, segundo Korth (1995), refere-se à garantia de que mesmo que ocorra falha em algum nó ou linha de comunicação, os dados estarão corretos. De acordo com Özsu (1991), quando os componentes de um ambiente distribuído falham, um SGBDD confiável deve ser capaz de continuar a executar as consultas dos usuários sem violar a consistência do banco de dados.

c) Sobrecarga de Controle

A questão da distribuição dos dados é responsável por grande parte das sobrecargas acarretadas no SGBDD, que influenciam no sistema como um todo. Assim, a **sobrecarga de controle de distribuição** foi acrescentada como um requisito não funcional de destaque em ambientes distribuídos. Nesse tipo de ambiente, a troca de mensagens e a computação adicional requeridas para realizar a coordenação entre os nós, e a informação distribuída, são uma forma de sobrecarga que não surgem em sistemas centralizados. É essencial manter essa sobrecarga em um nível razoável.

d) Custos

Outro requisito não funcional, apresentado na Fig. 2, corresponde aos **custos**. Segundo Özsu (1991), a função custo total consiste no **custo de armazenar** os fragmentos em seus respectivos *sites*, o **custo de execução** (custo para consultar determinado fragmento em um *site* e atualizar os *sites* onde os dados são armazenados) e o **custo de comunicação** de dados (tempo gasto e valor financeiro).

Com relação aos custos de comunicação, Barroso (1998) afirma que quanto maior o volume de processamento remoto, maiores serão os custos de comunicação. Consequentemente, maior será o tempo de resposta de uma determinada aplicação, uma vez que os custos de comunicação encontram-se embutidos no tempo de resposta, juntamente com o tempo de acesso e o tempo de processamento. Portanto, na análise do benefício de um determinado esquema de alocação, deve-se considerar os custos de comunicação existentes.

Essas questões podem ser melhor analisadas quando são fornecidas técnicas (operacionalizações) que possam implementá-las, ou seja, que as tornem mais concretas a nível de projeto. Por esse motivo, é importante a representação através dos catálogos de métodos, os quais serão introduzidos a seguir.

3.2.2 Definição dos Catálogos de Métodos

Após os requisitos não funcionais terem sidos suficientemente refinados, na seção 3.2.1, o presente passo consiste em identificar as possíveis técnicas existentes para implementá-los. Essas técnicas são denominadas **operacionalizações** e são definidas para os requisitos não funcionais de mais baixo nível, no caso deste trabalho tempo de acesso, tempo de processamento, disponibilidade, confiabilidade, sobrecarga de controle de distribuição e custo de comunicação. Os catálogos de métodos de operacionalizações, para cada um desses requisitos, são descritos a seguir.

É importante ressaltar que, nesta fase da metodologia, não foi objetivo a procura exaustiva de todas as possíveis estratégias de implementação, e sim indicar algumas operacionalizações bastante mencionadas na literatura. Algumas das operacionalizações apresentadas representam estratégias a serem selecionadas como alternativas pelo projetista, outras estão associadas com as funcionalidades geralmente disponíveis nos SGBDs. No último caso, elas são importantes para que o projetista verifique se no SGBD utilizado elas são implementadas ou não.

a) Operacionalizações dos Tempos de Acesso e Processamento

A Fig. 3 apresenta o catálogo de métodos para operacionalização[2] do Tempo de Acesso.

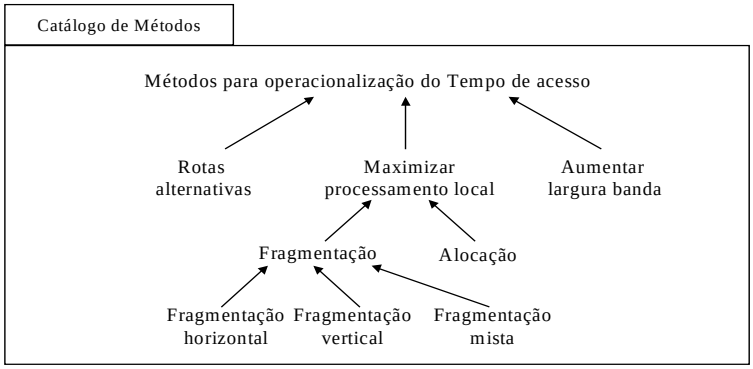

Fig. 3. Catálogo de Métodos para operacionalização do Tempo de Acesso

[2] As operacionalizações também podem ser refinadas, conforme pode ser observado na Fig. 3 e demais figuras desta subseção.

De acordo com a Fig.3, a **Maximização do processamento local** é uma estratégia para se obter um melhor tempo de acesso. Esta estratégia consiste no aumento da carga de processamento nos locais onde a informação se encontra, evitando acessos a dados localizados em outros *sites*. A maximização, por sua vez, pode ser obtida através da utilização de técnicas de alocação e fragmentação (que pode ser de três tipos: **horizontal**, **vertical** ou **mista**, fornecendo soluções para diferentes necessidades de acesso). Outras alternativas para se obter um melhor tempo de acesso são **Rotas alternativas** e **Aumento da largura de banda**, que por se tratarem de questões relacionadas com a rede, não serão detalhadas neste trabalho

O Tempo de Processamento, por sua vez, conforme pode ser visto na Fig. 4, pode ser minimizado através da **Utilização de paralelismo na execução das aplicações** e da **Melhor distribuição da carga de trabalho**. O paralelismo é alcançado identificando-se quais operações podem ser executadas de forma independente, por exemplo, operações que não acessam dados em comum. Segundo (Barroso 1998), a distribuição da carga de trabalho é empregada com o objetivo de aproveitar as diferentes capacidades e ocupações dos computadores de cada *site*. A distribuição da carga de trabalho também é possível através das operações de fragmentação, com suas subdivisões, e alocação, baseadas em aspectos relacionados com as capacidades das máquinas e com o equilíbrio da carga de trabalho.

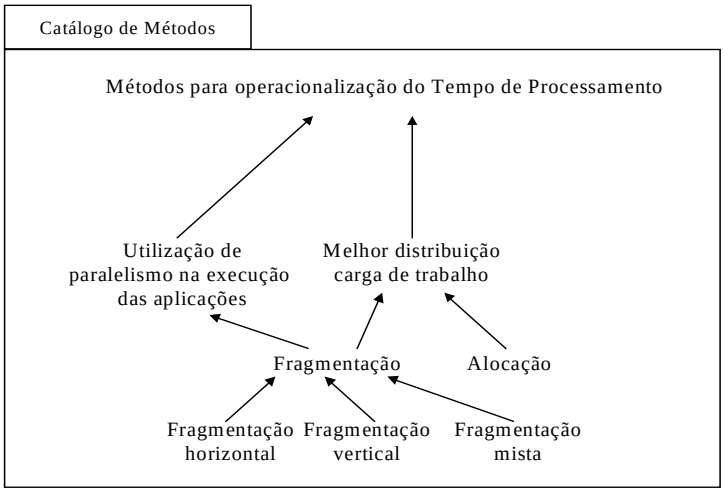

Fig. 4. Catálogo de Métodos para operacionalização do Tempo de Processamento

b) Operacionalização da Disponibilidade

De acordo com a Fig. 5, pode-se perceber que para se alcançar a disponibilidade é necessário considerar as soluções de distribuição dos dados e de

rede[3]. Com relação à distribuição dos dados, esse processo consiste de três aspectos essenciais: fragmentação, alocação e replicação.

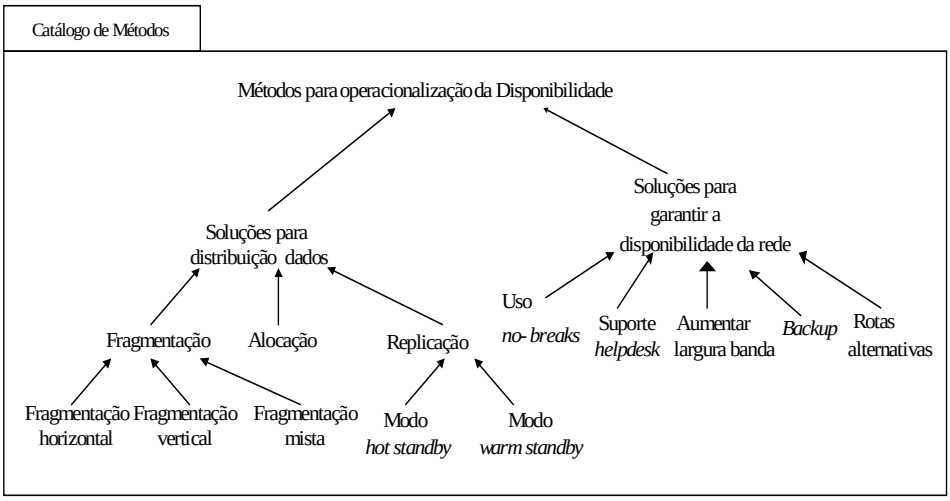

Fig. 5. Catálogo de Métodos para operacionalização da Disponibilidade

Segundo Tanenbaum (1995), a redundância pode melhorar a disponibilidade uma vez que peças-chave de hardware e software podem ser replicadas, de maneira que se uma delas vier a falhar, as outras cobrirão a lacuna. A mesma coisa acontece com os dados, ou seja, em caso de falha em um *site* do sistema, dados replicados em outros *sites* podem ser acessados. Assim, a **replicação** dos dados é uma das estratégias usadas para se obter a **disponibilidade**.

De acordo com Buretta (1997), existem duas soluções para realizar a replicação dos dados: ***hot standby*** e ***warm standby***, conforme pode ser observado na Fig. 5. Para implementar a solução *hot standby* o dado redundante é mantido consistente em tempo real usando replicação síncrona de dados. E, para implementar a solução *warm standby* o dado redundante é mantido consistente em tempo quase real usando replicação assíncrona de dados.

As outras estratégias, apresentadas na Fig. 5, para se garantir a disponibilidade correspondem às técnicas de **fragmentação** (fragmentação horizontal, vertical e mista) e **alocação**.

c) Operacionalização da Confiabilidade

A operacionalização do requisito confiabilidade é apresentada na Fig. 6. Pode-se observar que uma das estratégias para garantir a confiabilidade é a **replicação** dos dados, significando que dados corrompidos, devido à ocorrência de falhas, podem ser substituídos por dados replicados, garantindo assim, a

[3] Como este trabalho está voltado para a questão dos dados, as soluções que envolvem a rede não serão descritas aqui.

confiabilidade dos dados antes da falha. Mesmo que não ocorram falhas, o fato dos dados serem replicados em mais de um local também permite que se garanta a confiabilidade.

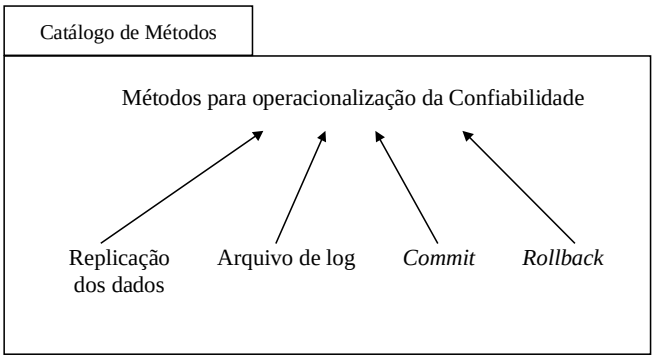

Fig. 6. Catálogo de Métodos para operacionalização da Confiabilidade

Outras técnicas para se conseguir a confiabilidade estão associadas com o suporte às transações, como é o caso do ***commit***, do ***rollback*** e do **arquivo log**. *Commit* é uma operação que sinaliza que uma transação foi realizada com sucesso, assim as atualizações executadas pela transação podem ser armazenadas permanentemente no banco de dados. *Rollback* corresponde à operação que sinaliza que a transação não foi finalizada com sucesso, assim, quaisquer mudanças aplicadas ao banco de dados devem ser desfeitas. O arquivo de *log*, por sua vez, é composto por uma série de registros que armazenam as operações, realizadas pelas transações, que afetam os valores dos itens do banco de dados. Este arquivo, portanto, é recuperado em caso de falhas nas transações (Navathe e Elmasri 2000).

d) Operacionalização da Sobrecarga de controle de distribuição

O refinamento do requisito não funcional **sobrecarga de controle de distribuição** é apresentado na Fig. 7. Uma das funcionalidades que influenciam a sobrecarga é a **atualização de réplicas** de uma dada informação. Ou seja, segundo Korth (1995), o SGBDD precisa assegurar que todas as cópias da mesma informação sejam consistentes a fim de não resultar em computações errôneas. Dessa forma, sempre que a informação for atualizada, essa atualização precisa ser propagada para todos os nós que contenham réplicas dessa informação.

O **processamento de consultas** é um outro item que influencia a sobrecarga de controle. As consultas podem envolver dados que podem estar fragmentados e/ou replicados. Assim, a consulta tem de ser **decomposta** e **otimizada,** e os dados, necessários para a realização da consulta, **localizados**. Se os dados estiverem em outros locais, possivelmente será necessário **movimentá-los** para o local apropriado. O resultado de tudo isso é o aumento da sobrecarga. Além disso, os **mecanismos de controle de concorrência**, necessários para garantir o acesso simultâneo aos dados, também colaboram para a sobrecarga de controle.

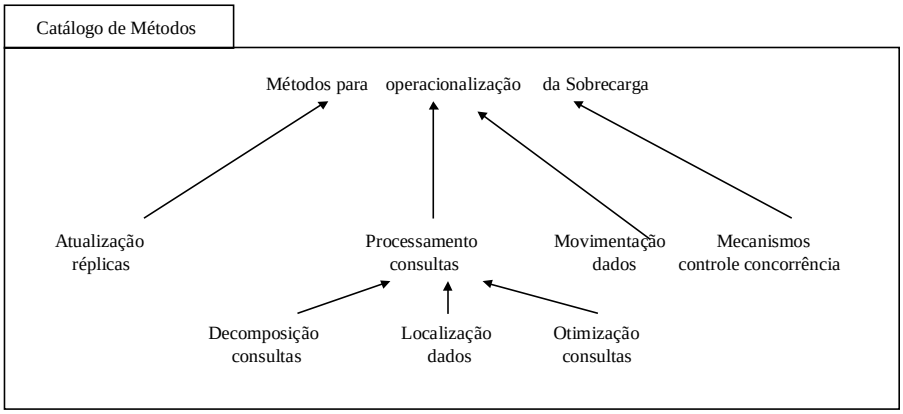

Fig. 7. Catálogo de Métodos para operacionalização da Sobrecarga de controle de distribuição

Segundo Özsu (1991), o processamento de consulta é uma questão crítica de desempenho. No contexto distribuído, o problema do processamento de consultas torna-se mais difícil do que em ambientes centralizados devido à existência de um grande número de parâmetros que afetam o desempenho das consultas distribuídas. Em especial, as relações envolvidas em uma consulta distribuída podem ser fragmentadas e/ou replicadas, induzindo um *overhead* dos custos de comunicação.

e) Operacionalização dos Custos de Comunicação

A Fig. 8 apresenta o catálogo de métodos relacionados ao custo de comunicação. O **custo de comunicação**, segundo Korth (1995), corresponde ao custo em tempo e dinheiro para enviar dados de um nó para outro em um sistema de banco de dados distribuído. Por questões de espaço, neste artigo é detalhada apenas a operacionalização do custo de comunicação com relação ao tempo gasto, não sendo abordada a parte financeira.

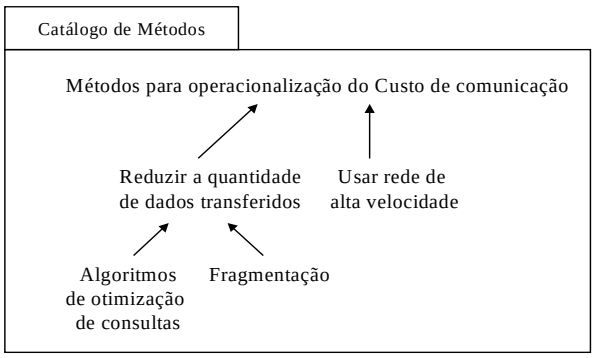

Fig. 8. Catálogo de Métodos para operacionalização do Custo de comunicação

Dentre as estratégias para se reduzir os custos de comunicação, encontram-se a **redução da quantidade de dados transferidos** (que é realizado através de **algoritmos de otimização de consultas** e de técnicas como a **fragmentação**) e o **uso de redes de alta performance** (os custos de comunicação são menores, em termos de tempo, se os *sites* envolvidos são conectados através desse tipo de rede).

3.2.3 Definição do Catálogo de Interdependências

A Fig. 9 apresenta um catálogo de interdependências, onde são descritos como determinadas operacionalizações (técnicas), definidas anteriormente, podem influenciar os demais requisitos, onde essa influência pode ser positiva (+), significando que a técnica contribui para melhorar determinado requisito não funcional; ou negativa (-), caso contrário. Apenas a título de exemplificação do uso desse tipo de catálogo, foram escolhidas as técnicas **replicação**, **fragmentação** e **alocação**, as quais são bastante referenciadas nos catálogos de métodos anteriormente definidos.

Catálogo de correlações								
Contribuições das operacionali-zações	**Aos RNFs**							
	Temp. acesso	Temp. proc.	Conf-bilidade	Vazão	Dispo-nibil.	Sobrec Contr.	Custo Armaz.	Custo Com.
Replicação			+		+	_	_	+
Fragmentação	+	+		+	+	_		+
Alocação	+	+		+	+			

Fig. 9. Catálogo de Interdependências

A **replicação** dos dados, independentemente de qual modo (*hot standby*, *warm standby*) seja escolhido, contribui positivamente para garantir a **disponibilidade** (Fig.5) e a **confiabilidade** (Fig.6), uma vez que garante a possibilidade de recuperação do sistema no caso de destruição da cópia ou queda do sistema. Entretanto, quanto mais cópias existirem, maiores são as chances de ocorrer inconsistência, em especial no caso de atualizações freqüentes, ocasionando uma maior **sobrecarga do controle de distribuição**. Barroso (1998) afirma que o benefício da replicação diminui com o aumento dos **custos de armazenamento**, já que cópias replicadas necessitam de mais espaço.

A decomposição de uma relação em fragmentos permite que uma série de transações sejam executadas de forma concorrente. Dessa forma, a **fragmentação** contribui para o aumento do nível de concorrência, o que consequentemente aumenta

a **sobrecarga de controle de distribuição**, e da **vazão** do sistema (Barroso 1998). Além disso, a fragmentação é uma estratégia usada para melhorar o **tempo de acesso**, **tempo de processamento**, **disponibilidade** e **custo de comunicação**, conforme foi apresentado, respectivamente (Fig. 3, Fig. 4, Fig. 5 e Fig. 8).

A **alocação** é uma estratégia que aumenta o **desempenho**, isto é, ela minimiza o **tempo de resposta** através da redução do tempo de acesso (Fig. 3) e tempo de processamento (Fig. 4) e maximiza a **disponibilidade** (Fig. 5) e a **vazão** do sistema em cada *site* (Özsu 1991).

Pelo que foi apresentado, a Fig. 9 mostra a influência das técnicas de implementação (eixo vertical) sobre os requisitos não funcionais (eixo horizontal). Essas informações são úteis ao projetista de banco de dados, uma vez que eles podem verificar logo no início do processo o impacto que seu projeto vai ter ao serem selecionadas determinadas técnicas.

4 Exemplo da Aplicação do Framework NFR

Os requisitos não funcionais podem ser detalhados com a utilização do *Framework* NFR, a partir do qual podem ser definidos os SIGs (gráficos de interdependência dos *softgoals*), que visam auxiliar o projetista a determinar e avaliar a escolha de uma determinada operacionalização em busca de se alcançar o requisito inicial.

O presente passo consiste em mapear as informações, definidas nos "catálogos de métodos" (seção 3.2.2) e nos "catálogos de interdependências" (seção 3.2.3), para os SIGs que representam o problema real. O exemplo, usado neste artigo, tem como cenário um hospital fictício composto pelos setores: Recepção, Administração, Laboratório, Clínicas (Consultórios), UTI. Estas unidades, apesar de funcionarem de forma autônoma, devem estar interligadas, permitindo, a circulação de informação entre elas, de forma a atender as diferentes necessidades.

Esta seção irá descrever um gráfico distinto para cada requisito desejado. Foi escolhido o requisito não funcional disponibilidade, que será tratado em diferentes contextos. É importante observar que o mesmo requisito – disponibilidade – é representado nas Fig.s 10, 11 e 12, porém elas retratam realidades diferentes, ou seja, o requisito disponibilidade está sendo aplicado em situações diferenciadas para pacientes, consultas e exames. O que se está querendo mostrar com isso é que a escolha de soluções, a partir dos catálogos de métodos, vai depender do problema em questão.

Seguindo o *Framework NFR*, para alcançar um determinado requisito é necessário selecionar entre as possíveis técnicas, especificadas nos catálogos de operacionalizações, aquelas que forem específicas para o sistema real. A Fig. 10 apresenta um SIG para o requisito disponibilidade dos dados do paciente. Para garantir a disponibilidade da classe paciente, é necessário disponibilizar os atributos do paciente. Em seguida são definidas as operacionalizações (soluções para distribuição dos dados), com suas respectivas subdivisões (fragmentação, replicação,

alocação), seguindo o catálogo de métodos do requisito disponibilidade apresentado na seção 3. O projetista pode, então, rejeitar (indicado por "✗") ou aceitar (indicado por "✓") as operacionalizações propostas. No caso da Fig. 10, a operacionalização escolhida foi a fragmentação, mais especificamente a fragmentação vertical, uma vez que tem-se a informação de que diferentes dados (atributos) são acessados por diferentes lugares. Os atributos do paciente serão, então, fragmentados verticalmente em atributos de saúde e atributos administrativos, uma vez que os dados de saúde são freqüentemente consultados por consultórios, UTI e médicos; e que os dados administrativos são acessados pelo departamento administrativo e pela recepção. Depois de fragmentados, os atributos de saúde serão alocados nos consultórios e os atributos administrativos no departamento administrativo.

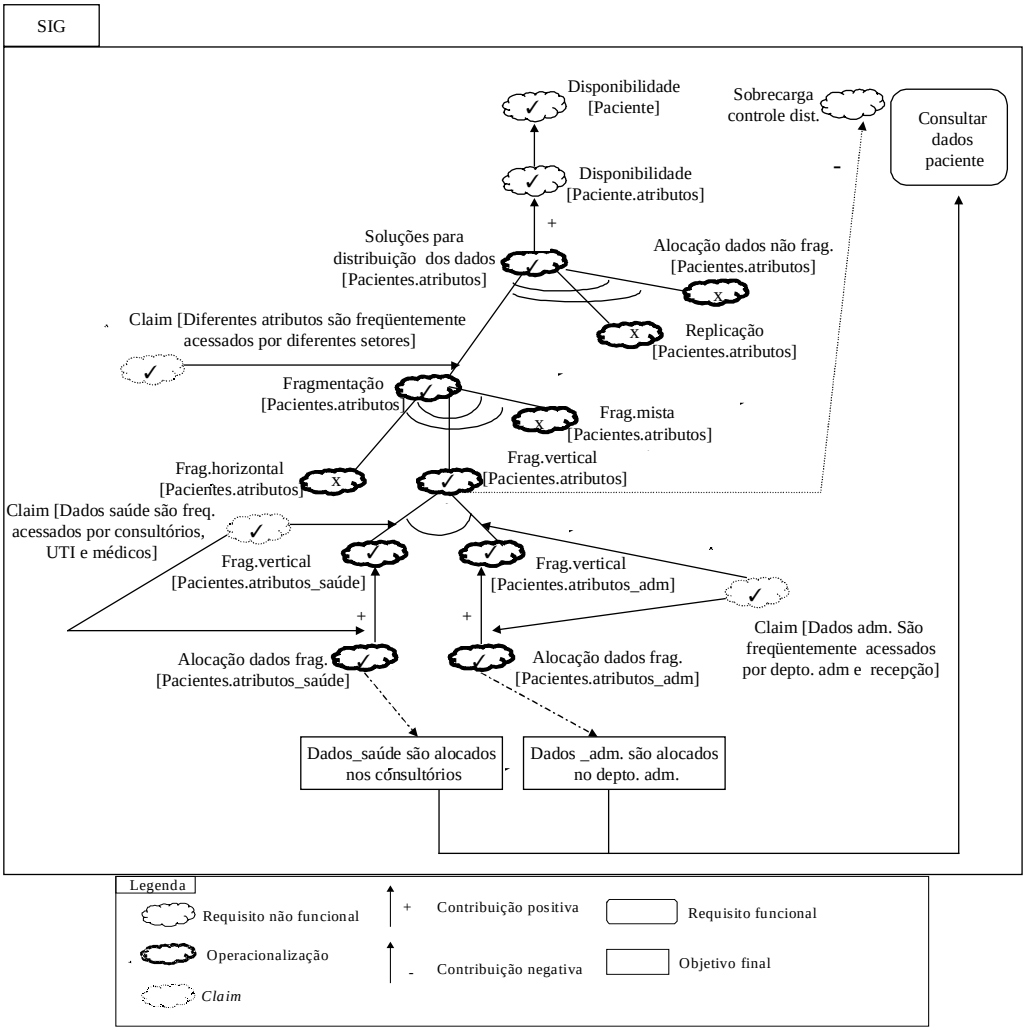

Fig. 10. SIG para disponibilidade dos dados do paciente

Em seguida, o impacto da escolha dessas operacionalizações é refletido nos requisitos mais acima, ou seja, a propagação das decisões segue uma abordagem *bottom-up*, até o requisito raiz, ou seja, até que o requisito disponibilidade seja satisfeito. A parte de baixo da Fig. 10 mostra o relacionamento das operacionalizações selecionadas com o objetivo final (dados administrativos são alocados no departamento administrativo e dados de saúde são alocados nos consultórios). O lado direito da Fig. 10 mostra a ligação desse objetivo com o requisito funcional (consultar dados do paciente).

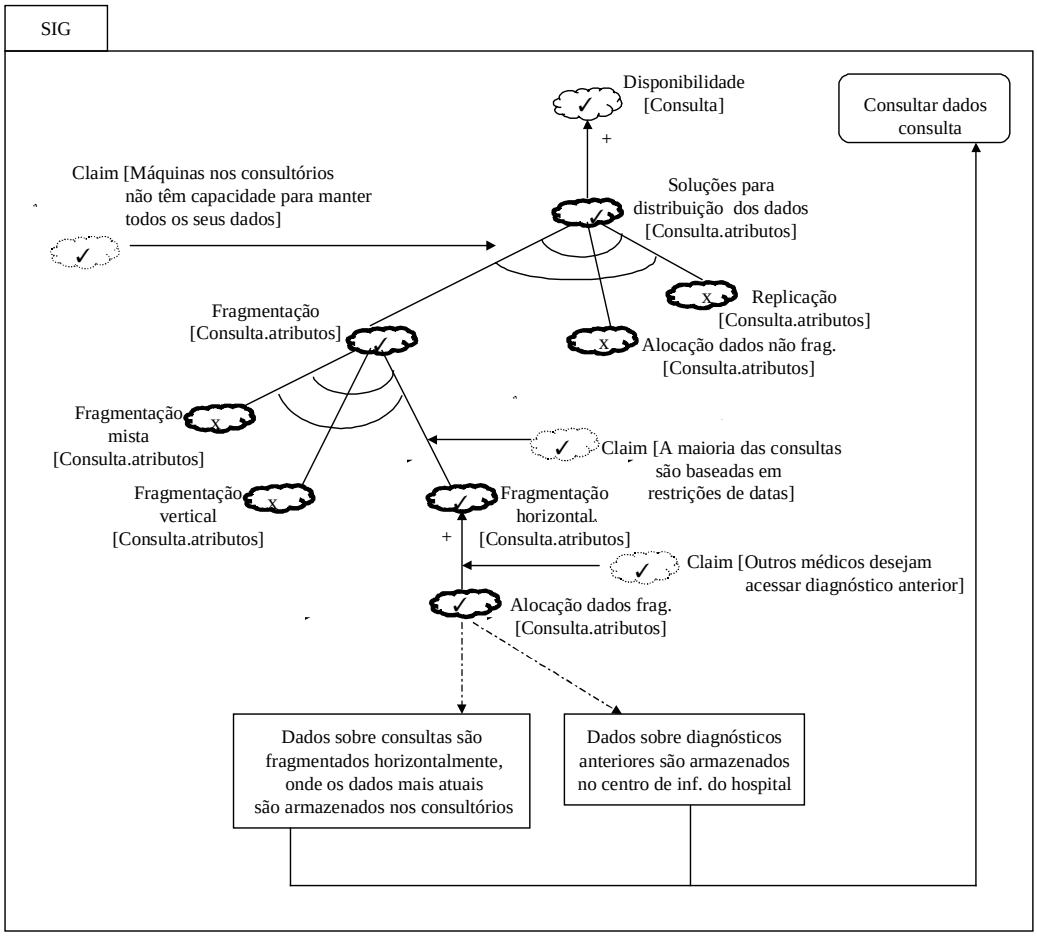

Fig. 11. SIG para disponibilidade dos dados sobre consultas

A Fig. 11 apresenta o SIG que trata do requisito disponibilidade dos dados sobre as consultas. Ela segue a mesma filosofia empregada para o SIG da Fig. 10. Assim, para se alcançar a disponibilidade são apresentadas as técnicas para distribuição dos dados, onde é selecionada a estratégia de fragmentação. A necessidade da fragmentação vem do fato das máquinas nas clínicas não terem

capacidade para manter todos os seus dados. Em seguida, é selecionada a fragmentação horizontal em virtude da maioria das consultas desejadas serem baseadas em restrições nas datas (por exemplo, data_atual – data_consulta < 12 meses). O próximo passo consiste em alocar os fragmentos. Assim, como as consultas basicamente são acessadas por médicos e devido às limitações das máquinas, apenas os dados mais atuais são armazenados nos consultórios. Além disso, outros médicos podem desejar ter acesso a um diagnóstico anterior, por exemplo. Desse modo, os dados devem ser alocados em outro local (no centro de informática do hospital, por exemplo), fazendo com que diminua a concorrência aos dados dos consultórios.

A Fig. 12 apresenta o SIG para a disponibilidade dos dados sobre exames. Aqui são selecionados duas técnicas para distribuição dos dados: alocação e replicação. A alocação foi selecionada devido à necessidade de autonomia de funcionamento do laboratório, o que resulta no armazenamento e controle dos dados pelo laboratório. A escolha da técnica de replicação baseia-se na necessidade de disponibilizar, de forma mais rápida, informações relativas a um paciente em estado de urgência. Assim, dados sobre exames devem ser replicados na UTI ou nos consultórios.

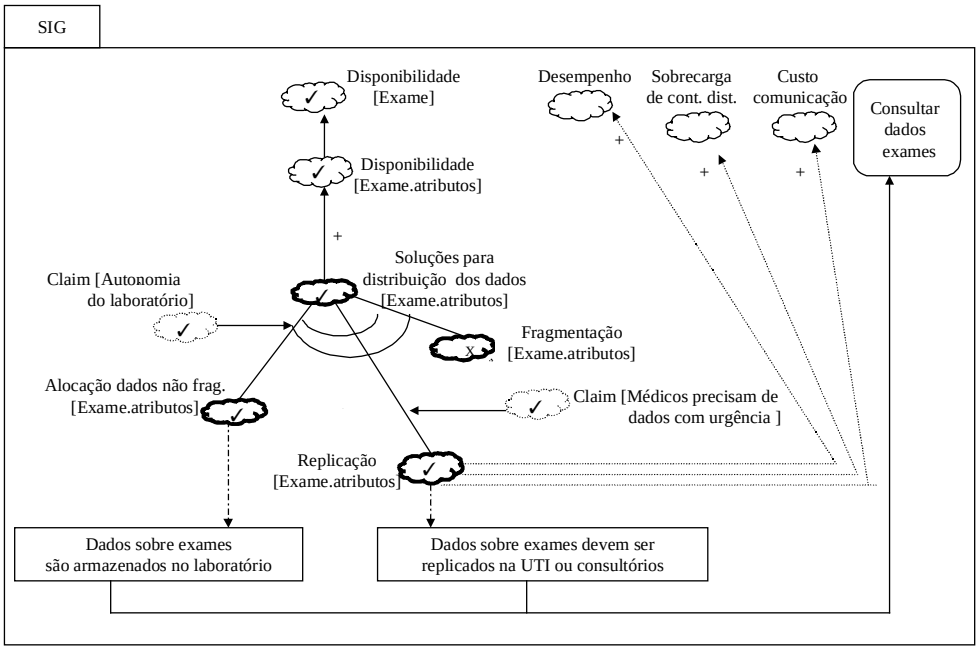

Fig. 12. SIG para disponibilidade dos dados sobre exames

A Fig. 12 apresenta também a influência de algumas operacionalizações em determinados requisitos não funcionais. No caso, a replicação dos dados contribui positivamente para alcançar o desempenho, uma vez que quanto mais dados replicados, menor será o tempo de resposta e, consequentemente, maior o desempenho. Além disso, como os dados que foram replicados não precisam ser atualizados uma vez que as operações realizadas sobre eles são apenas de consultas,

não vai haver uma sobrecarga de controle de distribuição. A replicação também causa influência positiva no custo de comunicação (evita deslocamento de dados).

5 Conclusão

Este trabalho faz parte do contexto de definição de uma metodologia que busca apoiar a fase de Definição de Requisitos do projeto de banco de dados distribuído, visando projetos mais bem fundamentados e completos, integrando informações de negócio, de requisitos não funcionais e funcionais (Santos 2000). Entre as metodologias propostas, encontra-se a utilização do *Framework NFR* que permite tratar, logo no início do projeto de banco de dados distribuído, os requisitos não funcionais que o influenciam, e que não eram explorados, pelas outras metodologias, durante as diversas fases do projeto. Com o emprego deste *framework* é que os projetistas podem verificar, através dos SIGs, como os requisitos não funcionais se relacionam com os requisitos funcionais, além de apresentar como eles são influenciados por outros requisitos não funcionais. Através dos catálogos pré-definidos, os projetistas podem selecionar, entre as possíveis alternativas de implementação, as técnicas que forem mais adequadas para modelar o problema real.

Este artigo mostrou dentro do contexto de distribuição, como o *Framework NFR* pode ser utilizado para retratar informações especificamente relacionadas com o projeto de banco de dados distribuído. Através do exemplo apresentado, mostrou-se como a ferramenta *Framework NFR* é relevante para melhorar a compreensão dos processos que envolvem o projeto de dados distribuído. Isso representou um passo significativo, uma vez que essa é uma técnica recente e que ainda está sendo estudada na área de Engenharia de Requisitos.

Referências

Barroso, M. C.: *PROJETOO – Uma metodologia para Projeto de Banco de Dados Orientado a Objetos*. Dissertação de Mestrado, Centro de Informática, UFPE, Pernambuco, Brasil 1998.

Booch, G., Rumbauch, J., Jacobson, I.: *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.

Bubenko, J.A., Kirikova, M.. *Software Requirements Acquisition through Enterprise Modelling*. Royal Institute of Technology and Stockholm University, Sweden, 1994.

Bubenko, J.A. Kirikova, M. . *Enterprise Modelling: Improving the Quality of Requirements Specifications. IRIS-17 Information Systems Research Seminar in Scandinavia. Oulu, Finland, 1994.*

Buretta, M.: *Data Replication – Tools and Techniques for Managing Distributed Information*. Jonh Wiley & Sons, Inc. 1997.

Coulouris, G., Dollimore, J. Kindberg, T.: *Distributed Systems – Concepts and Design*. Addison-Wesley, Second Edition, 1996.

Cysneiros, L. M.: *Integrando Requisitos Não Funcionais ao Processo de Desenvolvimento de Software*. Dissertação de Mestrado, PUC/RJ, Brasil, 1997.

Cysneiros, L. M., Leite, J. C. S. P.: *Definindo Requisitos Não Funcionais*. XI SBES – Simpósio Brasileiro de Engenharia de Software, págs. 49-64, Fortaleza, Ceará, Brasil 1997.

Cysneiros, L. M., Neto, J. M. S., Leite, J. C. S. P.: *Integrando Requisitos Não Funcionais na Modelagem Orientada a Objetos*. WER´99 – Workshop de Engenharia de Requisitos, págs. 117-128, Buenos Aires, Argentina, Set, 1999.

Chung, L., Nixon, B. A., Yu, E., Mylopoulos, J.: *Non-Funcional Requirements in Software Engineering*, Kluwer Academic Publishers, 2000, ISBN 0-7923-8666-3.

Dobson, J. E., McDermid, J. A: *On "Non-Functional" Requirements*. University of Newcastle, University of York, Technical report, PDCS Nº 65, 1991.

Draft Recomendation ITU-T X.901, 902, 903|ISO/IEC 10746-1, ODP Reference Model – Part 1, Part 2 and Part3 1995;

Hofmann, H. F.: *Requirements Engineering – A Survey of Methods and Tools. Institute for Informatics*, University of Zurich, 1993.

Korth, H. F., Silberschatz, A.: *Sistemas de Bancos de Dados*. Editora McGraw-Hill Ltda. 2ª edição revisada, 1995.

Lencastre, M., Paiva, M., Fonseca, D.: *Integrando Aspectos de Distribuição ao Modelo de Negócio*. Relatório Técnico, Centro de Informática, UFPE, Brasil, 1999.

Lencastre M.: *Metodologias para Apoio ao Projeto de Banco de Dados Distribuído*, Exame Qualify, Centro de Informática, UFPE, Pernambuco, Brasil, 2000.

Loucopoulos, P., Karakostas, V.: *Systems Requirements Engineering*. MacGRAW-HILL, 1995.

Mylopoulos, J., Chung, L., Yu, E.: *From Object-Oriented to Goal-Oriented – Requirements Analysis*. Communications of the ACM, Janeiro, Vol.42, Nº 1, 1999.

Navathe, S. B., Elmasri, R.: *Fundamentals of Database Systems*. Third Edition, Addison-Wesley, 2000.

Özsu, M. T., Valduriez, P.: *Principles of Distributed Dabase Systems*, chapter 1, pages 1-16. Prentice-Hall, 1991.

Pires, P. F.: *Himpar - Uma Arquitetura para Interoperabilidade de Objetos Distribuídos*. Dissertação de Mestrado, COPPE, Universidade Federal do Rio de Janeiro, Brasil,1997.

Santos, M. P.: *O Uso da Análise de Requisitos como Estratégia para apoio ao Projeto BDD* Tese de Mestrado, Centro de Informática, UFPE- Brasil 2000

Tanenbaum, A. S:. *Sistemas Operacionais Modernos*. Prentice Hall do Brasil, 1995.