4
Uma implementação da arquitetura MVC baseada em modelos

4.1.
Análise do banco de dados da aplicação

4.1.1.
Análise das características do sistema gerenciador de banco de dados

As características fundamentais que devem ser levadas em consideração na fase de escolha do SGBD, de forma a ser aplicado à arquitetura de solução proposta são:

- Estar baseado na tecnologia objeto-relacional (SGBD-OR),
- Tratar tipos de dados geográficos, fornecendo funcionalidades e procedimentos que permitam armazenar, acessar e analisar dados geográficos de formato vetorial,
- Possibilitar a estimativa de custo das consultas, principalmente as espaciais, para que previamente se identifique a possibilidade de adaptação do sistema, de forma que mesmo que haja uma forte relação semântica entre as classes de informações (CI's), o custo da consulta irá determinar se a CI relacionada será exibida em um primeiro momento ou se o sistema indicará ao usuário a existência desta informação para uma consulta posterior.

4.1.2.
Modelagem de dados

As características fundamentais que devem ser levadas em consideração na fase de projeto do banco de dados, de forma a ser aplicado à arquitetura de solução proposta são:

- Identificar se a CI possui informação geográfica ou não, ou seja, se a CI modelada utiliza tipos geométricos para representação dos dados vetoriais: ponto, linha e polígono, de forma que se possam realizar

consultas espaciais (operações topológicas, vide seção 1.1) sobre a CI. Esta informação será importante para a Visão na hora desta apresentar as informações que foram obtidas do banco de dados,

- Criar uma tabela específica para conter os dados geográficos da CI e associá-la à tabela que contém os dados relacionais desta mesma CI, de forma a possibilitar o uso dos atributos relacionais da CI em aplicações que não necessitem das suas informações geográficas para a realização de suas tarefas. Esta prática também é útil por duas razões:
    - Atualmente, quando utilizamos um SGBD_OR, como por exemplo o Oracle®, as tabelas que contêm atributos geográficos não são passíveis de replicação automática. Logo, toda uma logística de exportação destas tabelas e recriação dos índices espaciais será necessária.
    - Quando utilizarmos um SGBD relacional que não possua o recurso que trate dos tipos de dados geográficos, o dicionário de dados não reconhecerá as tabelas com atributos geográficos, de forma que, para criar uma nova instância ou esquema de dados deste banco, estas tabelas deverão ser excluídas. Desde que elas sejam criadas em separado, esta tarefa será facilitada.
- Identificar as tabelas de dicionário, cujas instâncias de alguns de seus atributos deveriam ficar no modelo de domínio, mas ficam no banco de dados por questões de desempenho e compatibilidade com os sistemas legados. Conseqüentemente as consultas SQL para obtenção destas instâncias no banco de dados, podem ser geradas previamente e armazenadas no modelo de domínio, associadas a esses atributos. Esta informação será importante para o Controle, pois a partir da execução destas consultas é que ele poderá preencher as instâncias deles. Estes atributos na realidade são os parâmetros das interfaces de consultas de filtro das CI's.

4.1.3.
Exemplo da estrutura lógica do banco de dados da aplicação

O fragmento de modelo entidade-relacionamento apresentado na Figura 10 serve para exemplificar algumas entidades utilizadas na estrutura lógica geral do banco de dados do domínio da aplicação.

Neste diagrama é apresentado a entidade Bacia e o atributo Geometria como exemplo de uma entidade que possui informações geográficas, como apresenta também os atributos Classificação_Poço e Tipo_Reclassificação como exemplo de atributos multivalorados da entidade Poço. Além disso, o fragmento mostra um relacionamento entre as entidades Poço e Bacia, indicando que consultas não-espaciais também podem ser feitas entre estas duas entidades.

As entidades Bacia e Poço serão utilizadas com freqüência nos exemplos que serão apresentados no decorrer do texto.

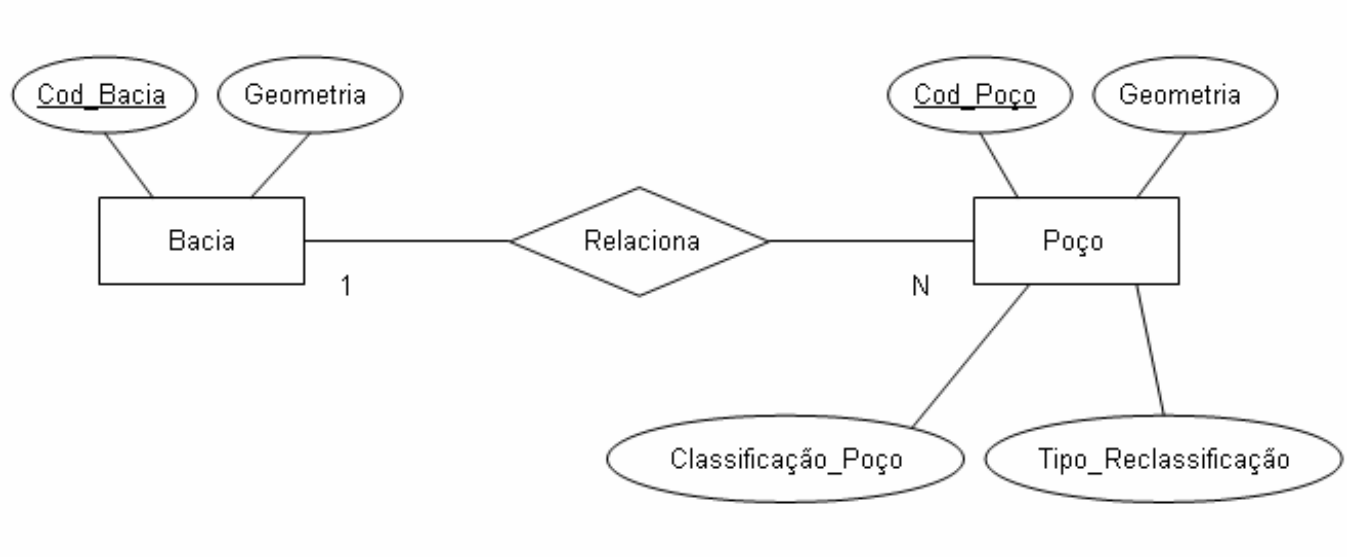

Figura 10: Fragmento do modelo entidade-relacionamento do banco de dados

4.2. Implementação dos modelos

4.2.1. Implementação dos modelos no banco de dados

Os modelos deste trabalho foram implementados em tabelas do SGBD Oracle® 9i. A opção por este tipo de implementação foi tomada porque não há nenhum tipo de inferência sobre o domínio utilizado e as operações executadas sobre os modelos são plenamente atendidas pelo uso de um SGBD e linguagem SQL.

Na Figura 11 é apresentado o diagrama de entidade-relacionamento dos modelos utilizados neste trabalho.

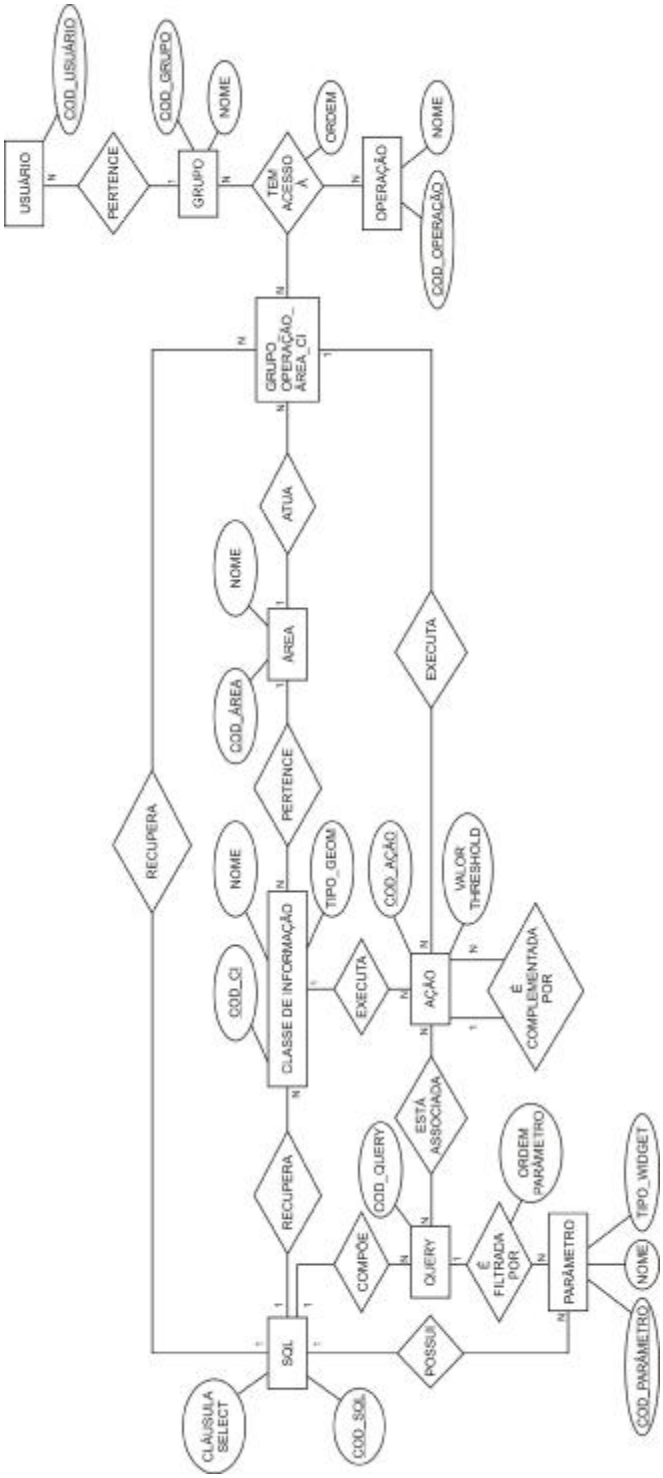

Figura 11: Diagrama do modelo entidade-relacionamento dos modelos de usuário, interface e domínio

4.2.2.
Implementação do modelo de domínio

O modelo de domínio que foi desenvolvido, no contexto deste trabalho, formaliza os conceitos do domínio da área de petróleo, especificamente na área de Exploração e Produção (E&P), que se encontravam embutidos no código do sistema de informações de E&P - VGE (Visualizador de Gestão da Exploração), atualmente sendo utilizado na Petrobras. Este sistema servirá como exemplo de uma aplicação que utiliza a arquitetura proposta no capítulo 3 e que será detalhado no capítulo 5.

4.2.2.1.
Características gerais do modelo de domínio

De forma a poder aplicar o modelo de domínio à arquitetura de solução proposta, o projeto do modelo de domínio tem que ter certas características fundamentais:

- Identificar as CI's que serão modeladas;
- Identificar a existência e o grau de relacionamento semântico entre as CI's, se forte ou fraco, de forma a possibilitar a ação de adaptação da aplicação;
- Informar o tipo da representação geométrica das CI's, quando existir;
- Caso as instâncias da CI possuam informações geográficas:
    - Identificar o seu tipo de representação vetorial: ponto, linha ou polígono,
    - Associar aos *templates* das consultas espaciais com as demais CI's,
    - Associar ao *template* da consulta espacial para obtenção de suas coordenadas,
    - Associar os tipos de relacionamentos topográficos para cada relacionamento entre as CI's,
- Relacionar as ações que podem ser realizadas sobre a CI,

- Associar as CI's relacionadas a cada uma das ações do item anterior,
- Associar cada CI às suas consultas de filtro,
- Associar os atributos em cada uma das interfaces de consulta de filtro da CI,
- Relacionar os atributos considerados obrigatórios em uma consulta de filtro de cada CI, juntamente com seu valor padrão,
- Associar cada atributo da consulta de filtro:
  - À sua ordem de aparição na interface,
  - À sua *widget* de apresentação: lista, texto, data, etc,
  - À consulta SQL para o seu preenchimento, caso seja um atributo multivalorado,
  - A um valor padrão, quando existir,
  - A um rótulo de apresentação, para internacionalização da interface.

4.2.3. Implementação do modelo de interface

O modelo de interface que foi desenvolvido, no contexto deste trabalho, formaliza a parte da apresentação e comportamento de uma aplicação exemplo que será descrita no capítulo 5, onde suas características de interface principais são a geração dinâmica e personalizada de menus *drop-downs*, apresentados a partir da seleção de uma ou mais instâncias de uma CI. Interfaces de filtro, responsáveis pelo refinamento da pesquisa na base de dados. Um *canvas* 2D, onde são apresentadas as informações geográficas das CI's utilizando-se técnicas de multi-resolução e a representação destas informações associadas à escala de representação do mapa.

O *canvas*, os menus *drop-downs* e as interfaces de filtro podem ser considerados elementos básicos na arquitetura deste tipo de aplicação, isto é, são seus *frozen spots*. Por sua vez, as CI's que são tratadas, as ações disponíveis e os atributos das interfaces de filtro de uma CI são os *hot spots.* São os elementos que atendem as necessidades e requerimentos específicos da aplicação que será instanciada.

4.2.3.1.
Características gerais do modelo de interface

De forma a poder aplicar o modelo de interface à arquitetura de solução proposta, o projeto do modelo de interface tem que ter certas características fundamentais:

- Relacionar as interfaces apropriadas para:
  - Seleção das ações que podem ser executadas em cada CI,
  - Seleção das CI's relacionadas com a CI selecionada,
  - Seleção das consultas de filtro de cada CI,
  - Representação das informações obtidas da base de dados dependentes da escala do mapa.

4.2.4.
Implementação do modelo de usuário

Conforme dito anteriormente, o modelo do usuário pode ser interpretado como sendo o modelo do domínio visto por um determinado usuário, como se fosse uma visão do banco de dados.

Ele representa o relacionamento de grupos de usuários, sem uma hierarquia definida entre eles, e o domínio, onde este relacionamento determina vários fatores, desde a relação das CI's que poderão ser pesquisadas pela aplicação, até a ordem de aparecimento de um atributo em uma interface de filtro.

A aquisição das informações para a criação do modelo de usuário bem como seus ajustes, foram feitas de forma manual, onde estas informações não são fixas, isto é, podem ser alteradas conforme o nível de adaptação que se queira alcançar.

Através da associação dos modelos de usuário, domínio e de interface é que será possível a projeção de um sistema mais customizado para cada usuário.

4.2.4.1.
Características gerais do modelo de usuário

De forma a poder aplicar o modelo de usuário à arquitetura de solução proposta, o projeto do modelo de usuário tem que ter certas características fundamentais:

- Identificar os usuários da aplicação;

- Identificar os papéis destes usuários, de forma a agrupá-los;
- Associar cada um destes grupos às características relacionadas nas seções 4.2.2.1 e 4.2.3.1;

4.3. Implementação da arquitetura MVC

4.3.1. Modelo

Na arquitetura proposta, as funcionalidades básicas do Modelo não foram alteradas pelo uso dos modelos, cabendo ao Modelo a execução das consultas instanciadas pelo Controle.

Os diferenciais propostos para este componente são o uso de *caches* dinâmicos para armazenamento dos identificadores das instâncias das CI's recuperadas da base de dados e o emprego da técnica de multi-resolução de linhas poligonais. A primeira possibilita o Modelo verificar quais instâncias deverão realmente ter seus dados preservados, após a recuperação, e tratados pela Visão. A outra somente será empregada na exibição das instâncias das CI's que possuem um número muito grande de coordenadas para a sua representação. No exemplo do sistema que será tratado no capítulo 5, elas são representadas pelas CI's País e Bacia.

Com o uso da *cache,* uma série de ações que devem ser executadas, desde a recuperação até a exibição da informação, deixa de ser necessária. Pois após uma seqüência de iterações de seleção e recuperação do usuário, a probabilidade de que vários identificadores das instâncias solicitadas já terem sido recuperados é grande.

Com o uso da técnica de multi-resolução pretende-se minimizar o volume de informações a serem tratadas pela Visão, dado que uma mesma instância da CI pode ser representada sob formas geométricas diferentes em várias escalas. Desta forma a representação pode ser feita com maior ou menor riqueza de detalhes.

A mesma técnica deverá ser empregada nas operações de aproximação e afastamento sucessivos, onde o Modelo deverá ou não obter mais informações da base de dados, para poder representar as instâncias da CI que a Visão deverá apresentar na GUI.

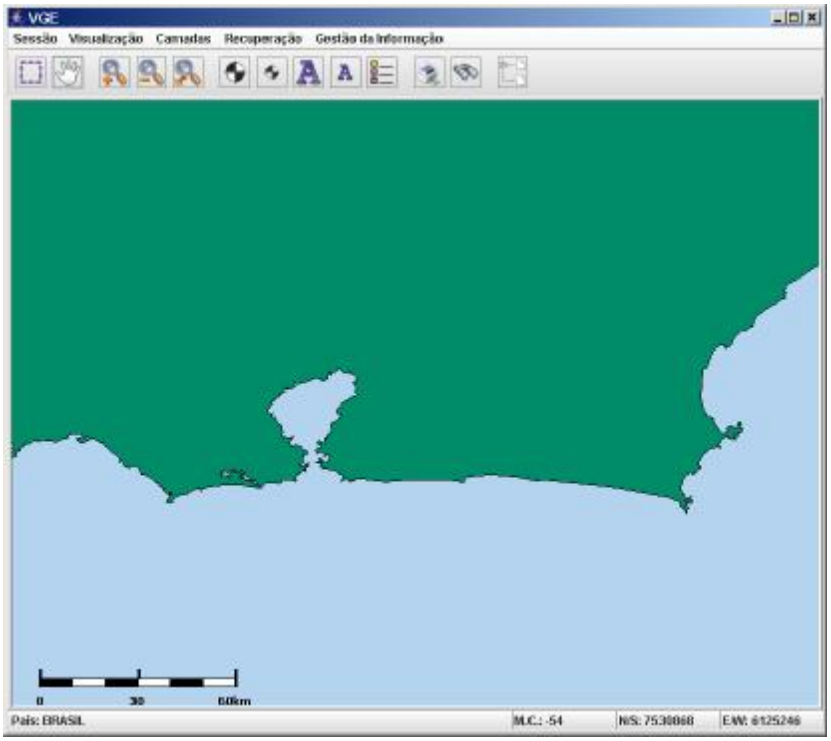

Figura 12: Litoral do estado do Rio de Janeiro utilizando multi-resolução

4.3.2. Visão

Como visto anteriormente, na arquitetura proposta, a Visão é responsável pela funcionalidade de apresentação na GUI das interfaces de seleção da aplicação, das interfaces de filtro das CI's e dos desenhos das instâncias da CI recuperadas pelo Modelo.

Para realizar esta última função, este componente necessitará antes obter, do modelo de interface, o valor do *threshold*. Este valor representa a semântica que a Visão utiliza para determinar o tipo de ação mais apropriada para representar as informações recuperadas, em conjunto com a escala atual do mapa e o perfil do usuário. A Visão pode exibir o resultado da recuperação de duas formas: ou exibir todas as coordenadas das instâncias da CI ou exibir um símbolo representando estas coordenadas. A primeira forma será escolhida se o valor da escala for menor que a do *threshold*, caso contrário, se o valor da escala for maior que a do *threshold*, a segunda forma será escolhida.

Se a segunda forma for escolhida, a Visão utiliza as coordenadas da primeira instância recuperada para exibir o símbolo no *canvas*, sendo dado o

nome “detalhar” para esta ação. Na barra de menu principal da aplicação, projetada sobre a arquitetura proposta neste trabalho, é possível o usuário “ligar” ou “desligar” esta ação. Desligar é possível mesmo que no modelo de domínio, associada ao perfil do usuário, esteja definido um valor de *threshold* para a ação de “exibir elementos da classe” para uma determinada CI. Vale ressaltar que as opções escolhidas só são válidas para a seção corrente.

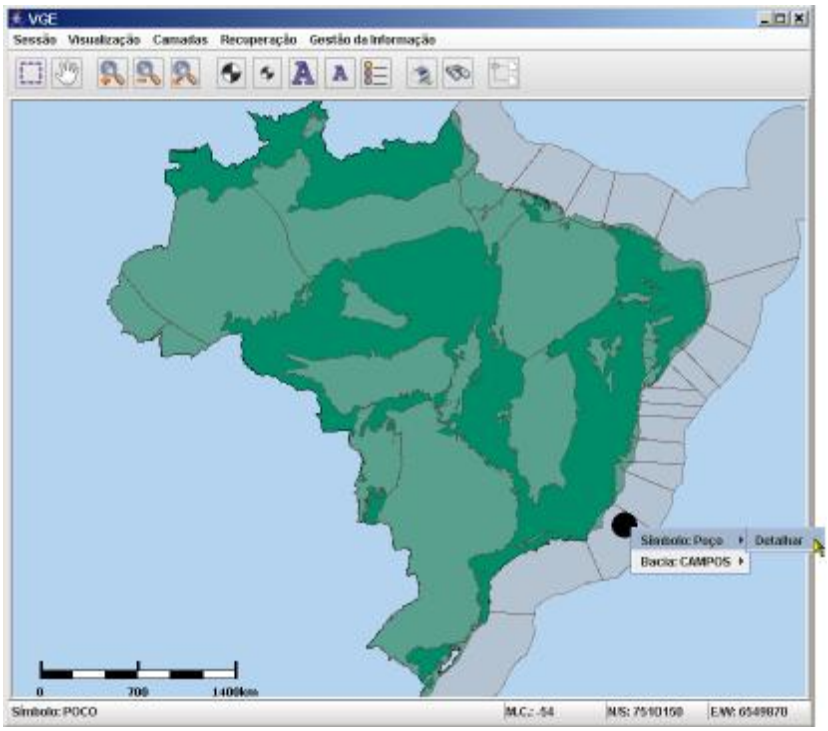

Figura 13: Exemplo da ação “detalhar”

4.3.3. Controle

Segundo a arquitetura proposta, o Controle para executar suas funções necessita obter informações tanto dos modelos que dão suporte ao sistema, quanto diretamente das bases de dados.

Seu papel é fundamental para que funcionalidades como a geração dinâmica das interfaces e o poder de adaptação da aplicação possam ser realizados baseados em um modelo de usuário.

Com o uso combinado dos repositórios de modelos (domínio, interface e usuário) e de dados (bases de dados legadas), os sistemas de informações

projetados segundo a abordagem proposta neste trabalho passam a dispor de um conjunto de informações mais rico. Por exemplo, as relações entre as CI's do domínio sobre o qual o sistema está atuando. Estas podem ser tanto relações topológicas (vide seção 1.1) quanto convencionais entre as CI's. A princípio estas relações podem não estar modeladas no banco de dados, ou então estarem dispersas dentro dos códigos da aplicação. Com esta nova abordagem, elas se tornam explícitas através da modelagem dos meta-relacionamentos no modelo de domínio.

Um diferencial, quanto a arquitetura MVC tradicional, é o acesso direto ao SGBDOR pelo Controle. Esse acesso se faz necessário porque as interfaces de filtro associadas às consultas das CI's, normalmente, possuem atributos com valores multivalorados. As instâncias destes atributos, por uma questão de desempenho e compatibilidade com os sistemas legados, permanecerão fora dos modelos.

No exemplo de uso da arquitetura, que veremos no próximo capítulo, temos o atributo ***classificação***, normalmente utilizado para filtrar a CI Poço em vários perfis de usuário, possui atualmente cinco instâncias: ***especial***, ***estratigráfico***, ***extensão***, ***injeção*** e ***jazida mais profunda***. Esta informação, que se encontra na base de dados, pode ser utilizada tanto por aplicações legadas, quanto pelas que utilizam modelos como suporte às suas funcionalidades.

A necessidade do uso de uma interface de consulta de filtro da CI se justifica quando lidamos com CI's com uma quantidade de instâncias muito grande, quando queremos definir melhor o tipo de informação da CI que se quer recuperar ou ambas.

No domínio de E&P do sistema que será visto no próximo capítulo, temos como exemplo a CI Bloco. Um bloco ao longo de sua existência passa por uma série de transformações geométricas desde que é adquirido na licitação até a sua devolução total à Agência Nacional de Petróleo (ANP), de forma que cada transformação gera uma nova versão geométrica deste bloco na base de dados. Neste caso se não existir nenhum filtro prévio, todas as versões de contornos geométricos do bloco serão recuperadas pelo usuário em uma consulta.

Vale ressaltar que para uma dada CI pode existir mais de uma consulta de filtro, a ela associada, pois nem sempre um único tipo de consulta poderá atender a todos os grupos de usuários da aplicação. Os grupos podem ter focos diferentes

sobre a mesma CI. Alguns enfatizando, por exemplo, o aspecto espacial e outro o aspecto financeiro da CI. Por isso cada consulta de filtro poderá ter atributos completamente diferentes das outras.

Quando for identificado que a CI possui informação geográfica, existirão *templates* das consultas espaciais entre ela e as demais CI's, associados ao meta-relacionamento semântico existente entre elas (vide seção 2.3).

Estes *templates* serão instanciados pelo Controle segundo o mecanismo de adaptação do sistema descrito no modelo do usuário. Isto acontecerá após a seleção da ação de exibir instâncias de uma CI a partir de outra CI. O sistema obterá então, do modelo de domínio, o conjunto de CI's que possuem relacionamento semântico forte com a CI solicitada. Com esses dados o Controle instanciará os *templates* de forma que, além da consulta original outras consultas relevantes sejam realizadas, enriquecendo o conjunto de informações que seria obtido inicialmente.

Os *templates* das consultas espaciais instanciados, juntamente com o código SQL dos atributos selecionados nas interfaces de filtro da CI, formam o conjunto de consultas que o Controle enviará ao Modelo para a obtenção das coordenadas de mundo das CI's a serem apresentadas na GUI pela Visão.

Um exemplo do uso de *templates*, que será visto na aplicação descrita no capítulo 5, é quando o usuário solicita instâncias da CI Poço a partir de uma instância da CI Bacia sendo que a CI poço possui forte relacionamento com a CI Bloco, segundo o perfil deste usuário. Neste caso, o sistema instanciará o *template* da consulta espacial entre as CI's Bacia e Poço, bem como o *template* da consulta espacial entre poço e bloco, para que além das instâncias da CI Poço que atendam a consulta original, as instâncias da CI Bloco que possuem algum relacionamento topológico (vide seção 1.1) com os poços recuperados também sejam exibidas.

Na barra de menu principal da aplicação, projetada sobre a arquitetura proposta neste trabalho, é possível o usuário "ligar" ou "desligar" a opção de adaptação. Desligar significa que a adaptação não será realizada, mesmo que no modelo de domínio associada ao perfil do usuário seja sugerida alguma consulta adicional para as consultas que serão feitas. Vale ressaltar que as opções escolhidas só são válidas para a seção corrente.

Para exemplificar algumas das funcionalidades do Controle, as Figuras 14 e 15 apresentam dois tipos de interfaces de seleção geradas dinamicamente pela aplicação que será descrita no capítulo 5. Onde podem ser vistas algumas das CI's que fazem parte do modelo de domínio. Por exemplo: Bacia, Bloco, Sísmica 2D Aquisição e Poço. E as tarefas que a CI Bacia pode executar.

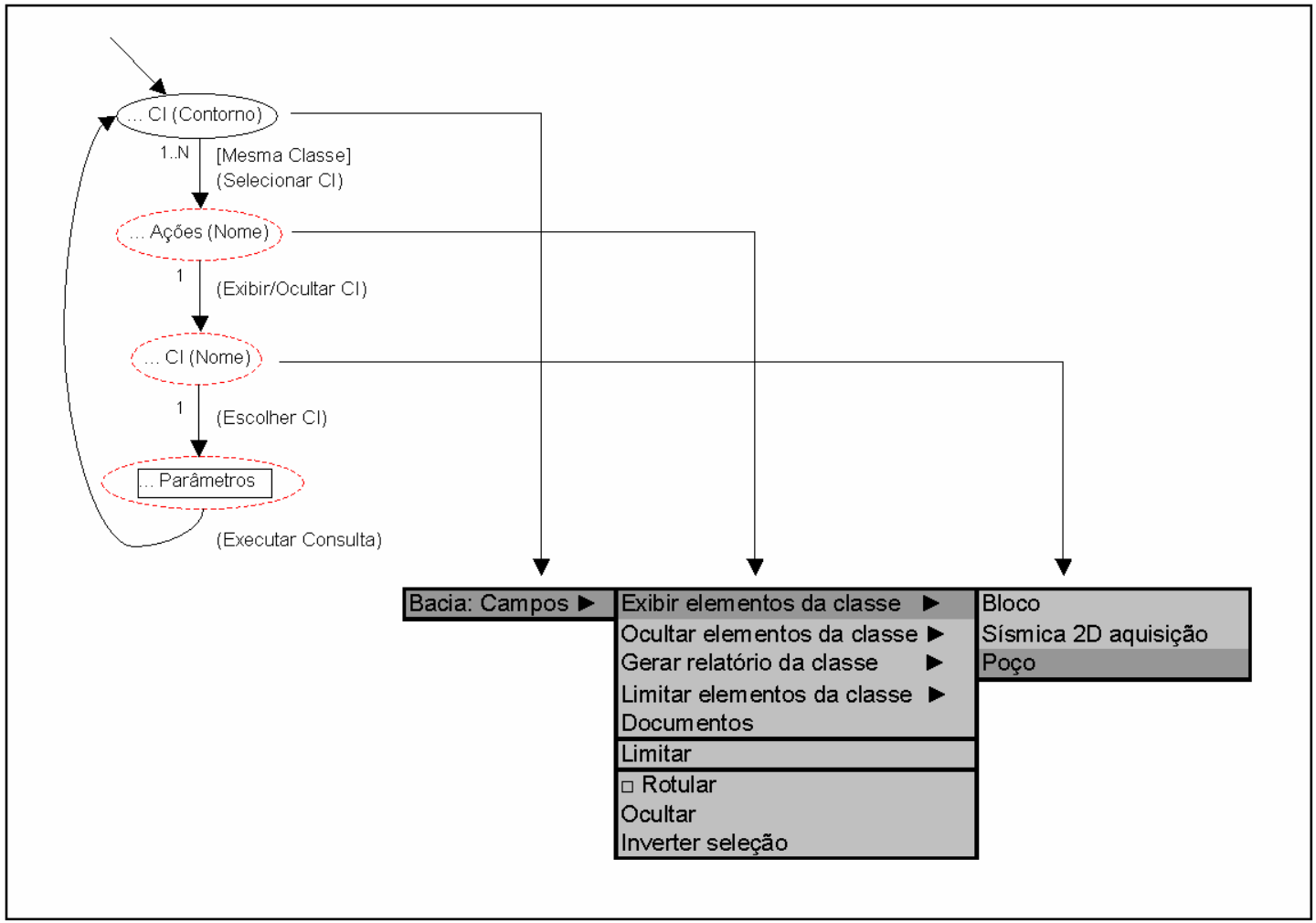

Figura 14: Exemplo de menu de seleção gerado dinamicamente (I)

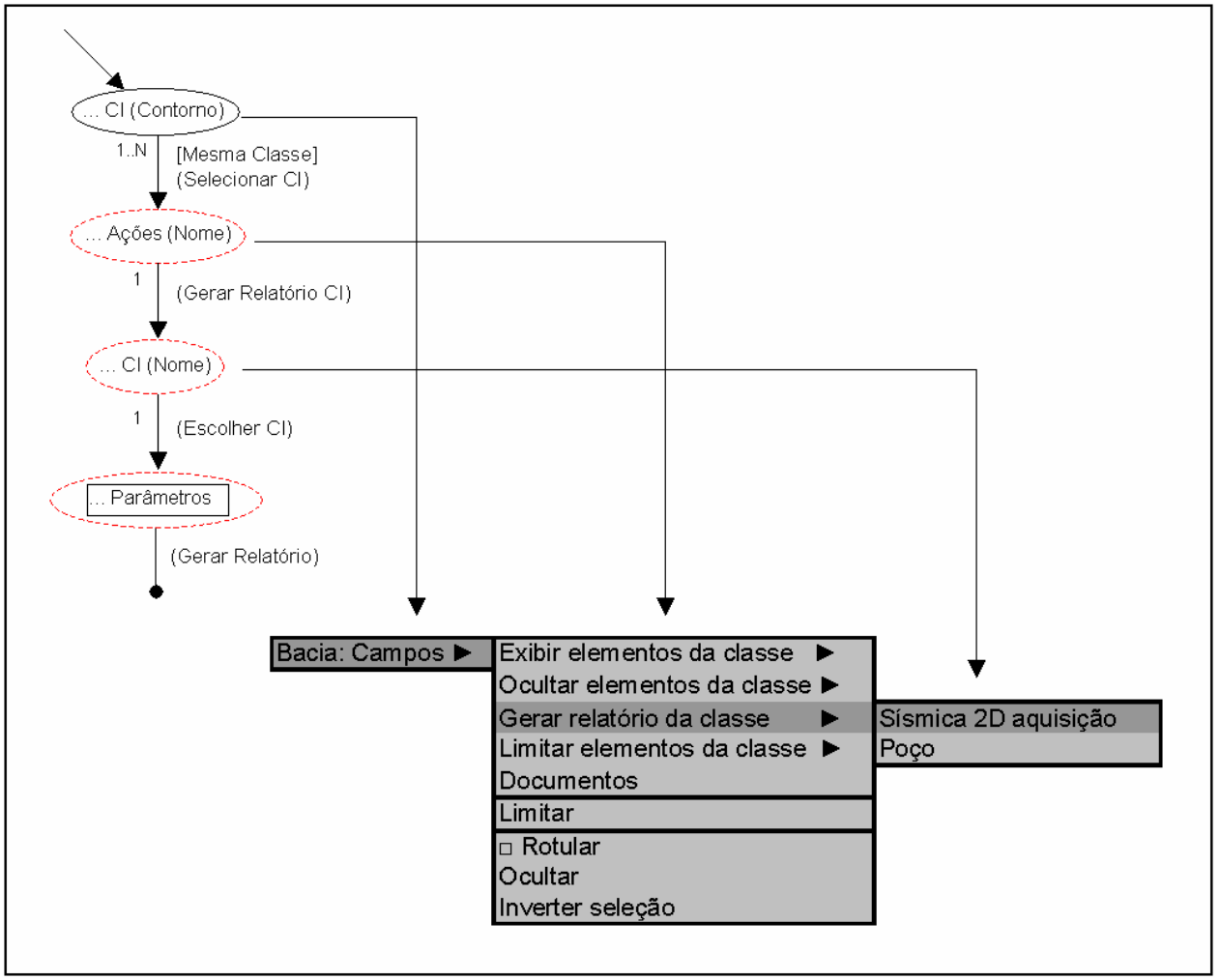

Figura 15: Exemplo de menu de seleção gerado dinamicamente (II)