UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
DEPARTAMENTO DE INFORMÁTICA APLICADA
INF01120 - TÉCNICAS DE CONSTRUÇÃO DE PROGRAMAS - 2009/2

Trabalho Prático - Editor de Diagramas QOC

Tomas Mattia (180687)

O trabalho prático determina o desenvolvimento de um software para edição de Diagramas QOC (*Questions*, *Options* e *Criteria*), um modelo de *design rationale* que serve para estruturar e documentar as decisões (e suas justificativas) dos membros da equipe de desenvolvimento de software. Para tal, foram determinadas etapas para o desenvolvimento, detalhadas ao longo deste trabalho: especificação informal de requisitos, decomposição OO e projeto de classes, projeto da interface do usuário, implementação e testes.

Porto Alegre, 26 de novembro de 2009.

1 Especificação Informal de Requisitos

Funcionais

- Usuário pode criar elementos e associar uma descrição a eles:
    - questões
    - opções
    - critérios
- Usuário pode criar/editar/excluir relacionamentos:
    - Questões podem ter zero ou mais opções
    - Questões podem ter zero ou mais critérios
    - Opções podem ter zero ou mais questões
    - Opções podem ter zero ou mais critérios
- Os relacionamentos são direcionais
- Há dois tipos de relacionamento:
    - pró: significa que o primeiro elemento favorece o segundo
    - contra: significa que o primeiro elemento não favorece o segundo
- Interface deve fornecer uma janela, com menus para realização de operações:
    - Básicas de arquivos
    - De inserção de elementos
    - Básicas de ajuda
- Interface deve representar elementos através de quadrados com texto dentro
- Interface deve representar elementos diferentes com cores de fundo diferentes
- Interface deve permitir seleção de elementos através de clique do mouse
- Interface deve permitir movimentação de elementos na tela através de comandos do teclado
- Interface deve permitir exclusão de elementos a partir de seleção
- Interface deve representar relacionamentos como linhas entre elementos

Não funcionais

- O sistema deve ser entregue até o dia 26/11/2009
- O sistema deve ter um tempo de resposta rápido
- O sistema deve possuir uma interface limpa e agradável tanto visualmente como funcionalmente

2 Decomposição OO e Projeto de Classes

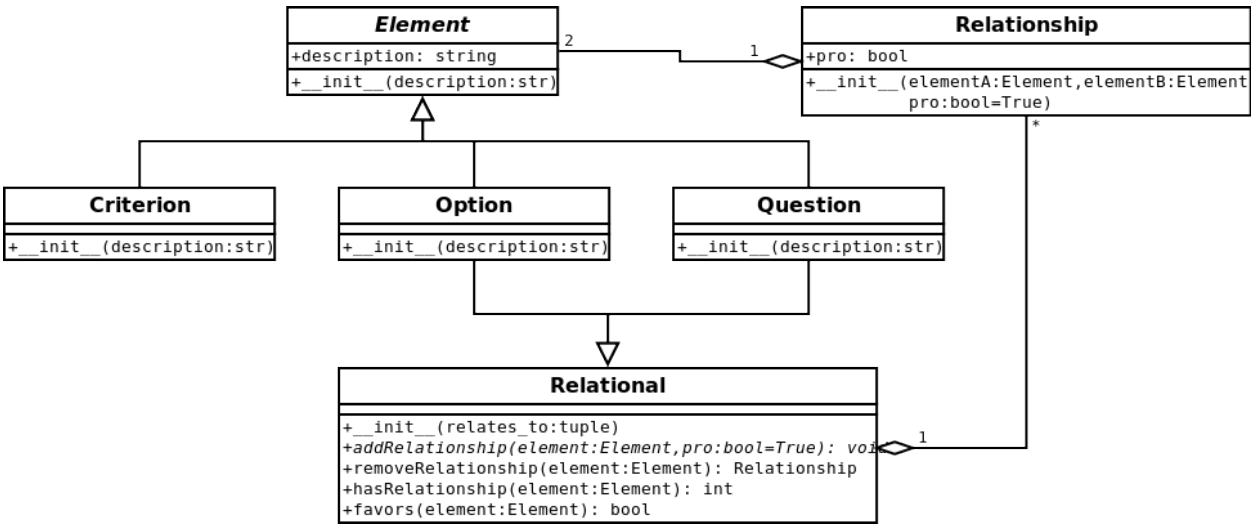

Figura 1 - Diagrama de classes da lógica de negócio

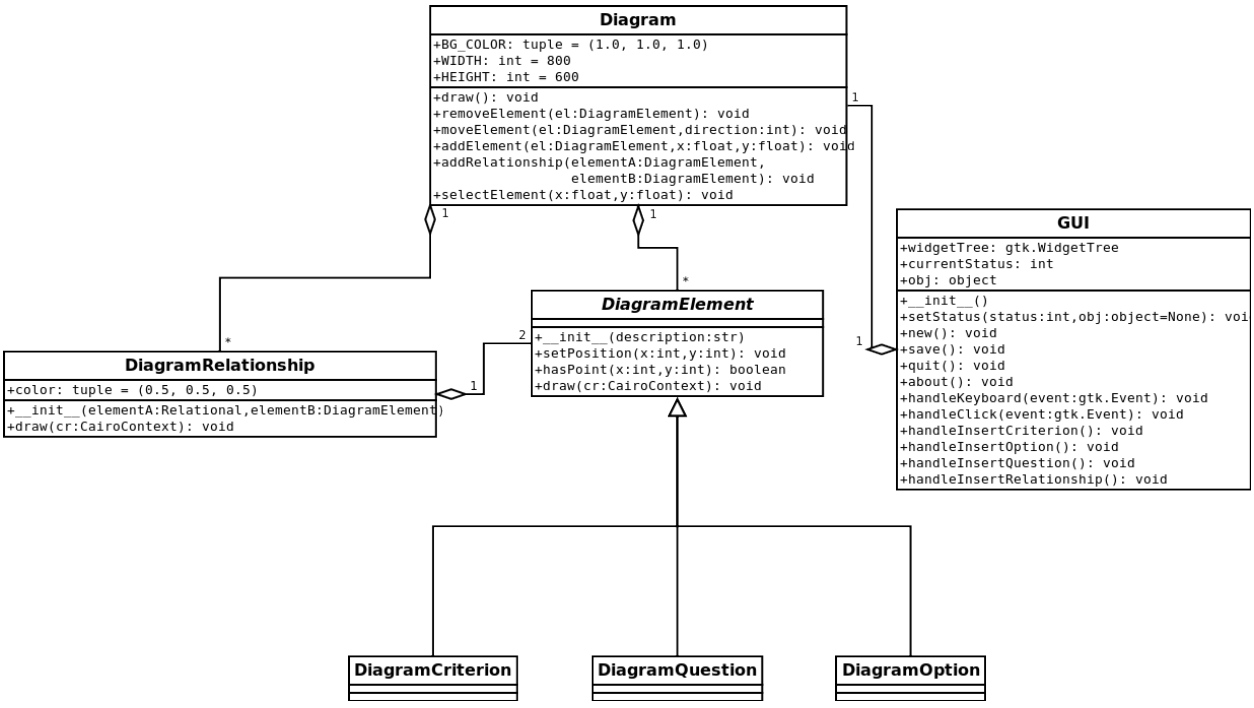

Figura 2 - Diagrama de classes da parte de interface gráfica

3 Projeto da Interface do Usuário

A Figura 3 mostra a janela principal da aplicação, com área em branco para desenho e menus de ação. O menu "File" contém operações básicas de arquivos, como "Novo arquivo", "Abrir", "Salvar" e "Sair". O menu "Insert" contém operações de inserção de elementos e relacionamentos, e o menu "Help" contém informações sobre o software.

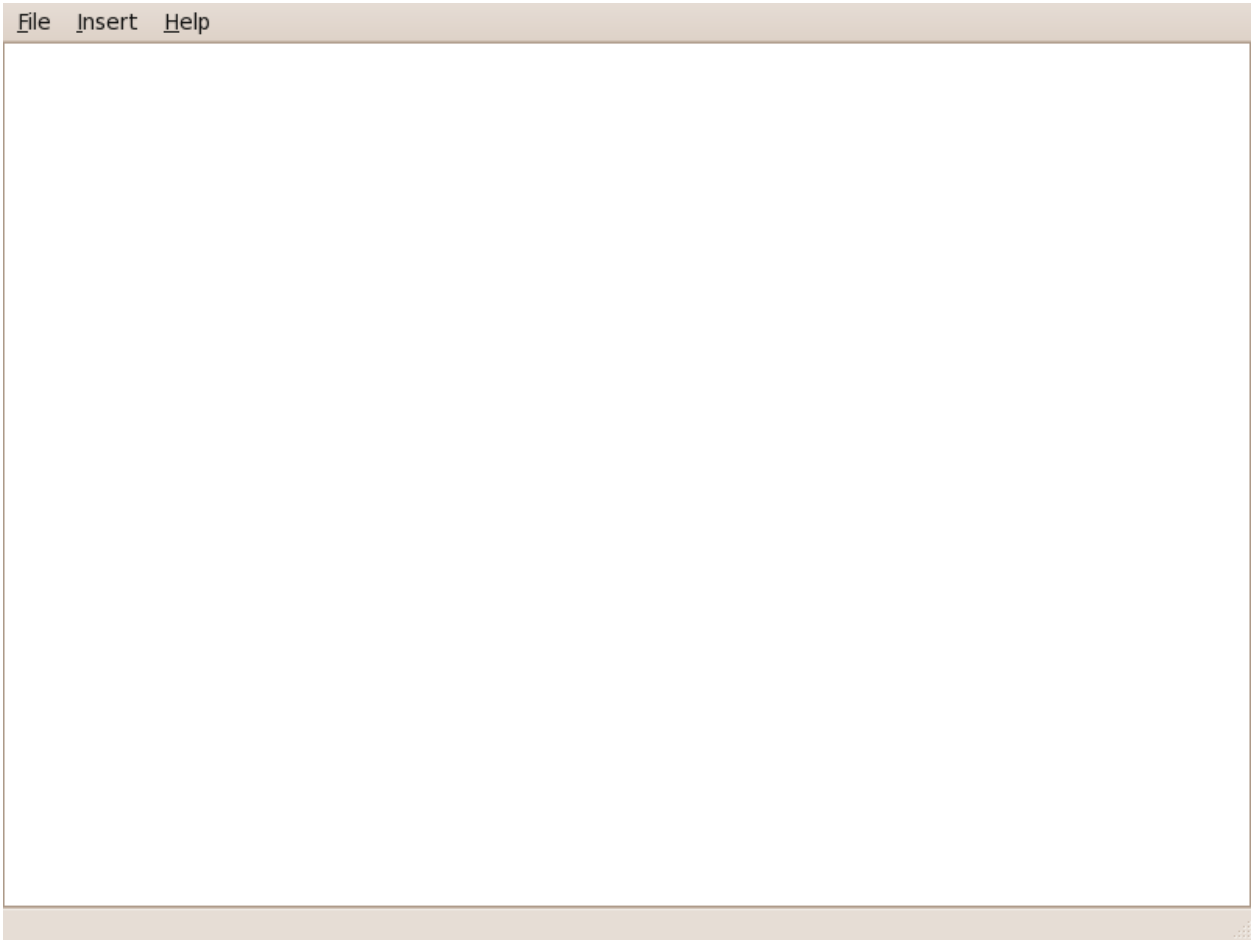

Figura 3 - Janela principal da aplicação

A Figura 4 mostra uma janela modal com campo para entrada de texto. Utilizada para recolher a descrição dos elementos no momento da inserção dos mesmos.

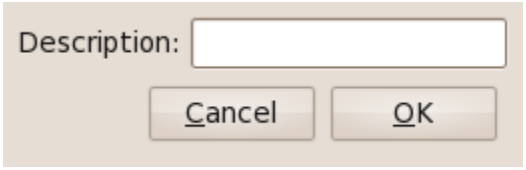

Figura 4 - Janela modal

A Figura 5 mostra um diagrama com alguns elementos desenhados. Quadrados em vermelho são critérios, quadrados em azul são questões, e quadrados em verde são opções. Relacionamentos são representados por linhas cinzas, no caso de ser uma relação a favor, e linhas pretas, no caso de ser uma relação contrária.

A barra abaixo da área de desenho é a barra de status, ela indica qual o estado atual do editor. No caso da Figura 5, a opção de inserção de relacionamento acabou de ser selecionada.

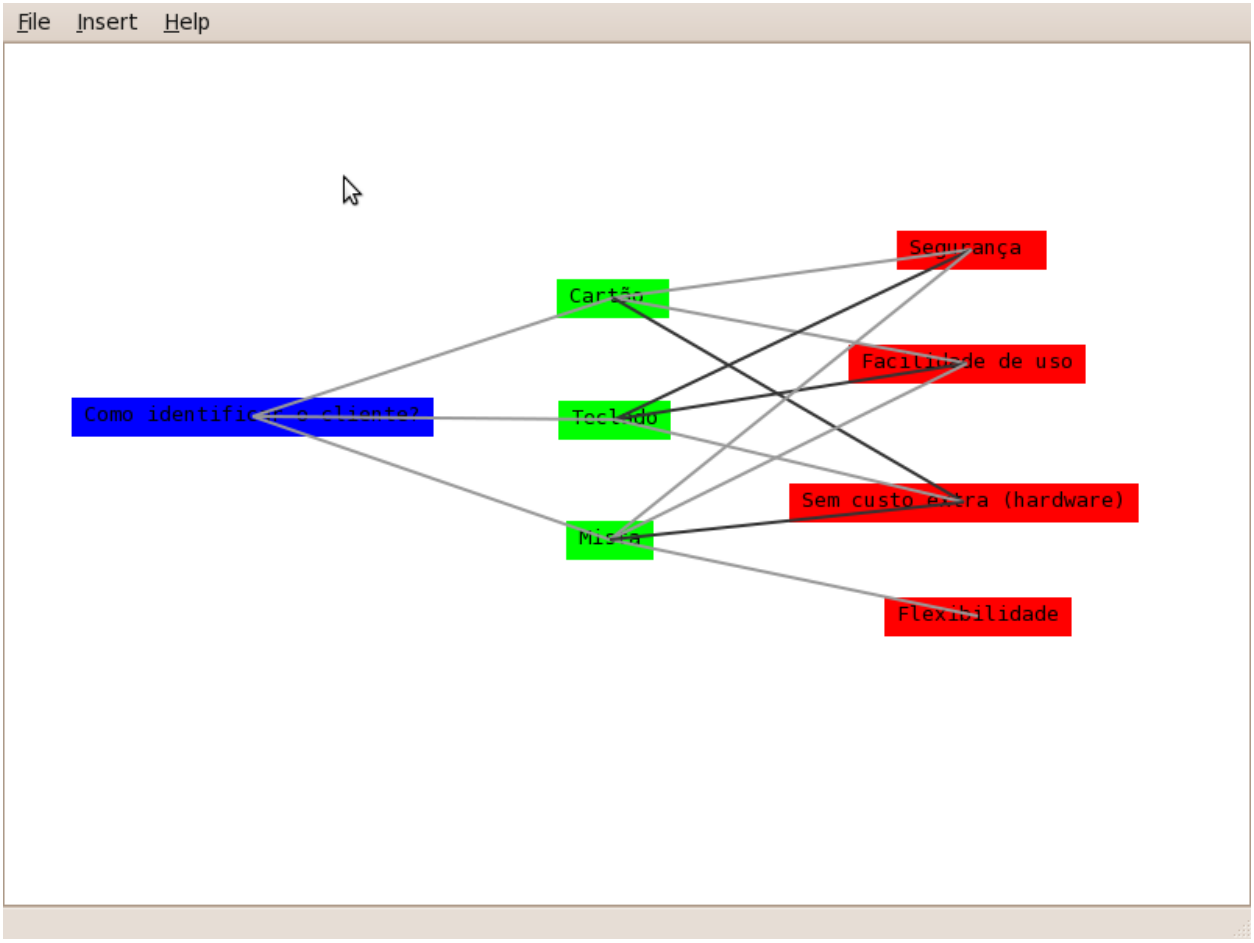

Figura 5 - Exemplo de diagrama QOC

Ações

Inserção de Elementos

Para inserir um elemento, seja ele Critério, Questão ou Opção, deve-se selecionar o menu "Insert" e em seguida o elemento desejado. Uma janela modal, conforme a Figura 4 vista anteriormente, aparecerá com um campo para entrada de texto. Neste campo deve ser informada a descrição do elemento. Os botões de "OK" e "Cancel" confirmam ou cancelam a operação, respectivamente. Se confirmada a operação, a barra de status será atualizada com mensagem referente a ação escolhida. Para desenhar o elemento na tela, basta clicar em uma parte de área de desenho.

Inserção de Relacionamentos

Para inserir um relacionamento, deve-se selecionar o menu "Insert" e em seguida a opção "Relationship". A barra de status será atualizada com mensagem referente a ação escolhida. Como os relacionamentos são direcionais, o primeiro elemento será a fonte do relacionamento. Para determiná-lo, basta clicar sobre a sua área. Da mesma forma, para determinar o segundo elemento deve-se clicar sobre a área do mesmo. Se o clique sobre o segundo elemento é feito com o botão esquerdo do mouse, o relacionamento é considerado do tipo "pró". Caso contrário, o relacionamento é do tipo "contra".

Seleção de Elementos

Para selecionar um elemento, basta clicar com o mouse sobre a sua área. Quando um elemento é selecionado, a barra de status é atualizada. Para cancelar a seleção, utiliza-se a tecla "ESC".

Movimentação de Elementos

Para movimentar um elemento, basta selecioná-lo e utilizar as setas do teclado como orientação.

Exclusão de Elementos

Para excluir um elemento, basta selecioná-lo e utilizar a tecla "DELETE". Quando um elemento é excluído, todas os seus relacionamentos também são.

4 Implementação

Para implementação do editor, partiu-se de uma definição básica de requisitos e implementação da lógica do negócio por trás dos mesmos. A partir daí, foi-se incrementando os requisitos e aperfeiçoando o modelo, até atingir a necessidade de uma interface gráfica. A partir daí, repetiu-se o processo, mas dessa vez levando em consideração a evolução da interação do usuário com o sistema, não apenas as suas funcionalidades.

O programa foi divido em três módulos principais:

- "qoc.py": implementação da lógica de negócio
- "gui.py": mediação da interface gráfica com a representação dos elementos
- "diagram.py": representação do diagrama e seus elementos

Essa foi uma simulação do padrão MVC[1], onde "qoc.py" representa o Model, "gui.py" e "diagram.py" o Controller e parte da View. O arquivo "gui.glade" representa exclusivamente a View.

Para a implementação do editor de diagramas, foram utilizadas as seguintes ferramentas, detalhadas em seguida:

- Python
- GTK e PyGTK
- Git
- Dia
- Glade

Python

Python[2] foi utilizada como linguagem de programação por cumprir um dos requisuportar orientação a objetos. Algumas particularidades a diferem da linguagem estudada em aula (Java[3]), dentre as quais pode-se perceber na implementação:

- Inexistência de atributos e métodos privados. Na verdade, eles existem, mas de maneira superficial, então a convenção adotada entre programadores da linguagem é nomear os atributos e métodos prefixados por um caractere "underscore" como indicação de privado;
- Inexistência de interfaces e classes abstratas;
- Existência de herança múltipla;
- Inexistência de constantes. A convenção adotada foi de nomear os atributos constantes com letras em caixa alta. Entenda-se por constante atributos que são definidos uma vez e não são alterados ao longo da execução do programa;
- Tipagem dinâmica, o que não permite sobrecarga de métodos. Para tal, outras técnicas são utilizadas, como a verficação de tipos através do método "isinstance";
- Polimorfismo sem herança: como a tipagem é dinâmica, qualquer variável pode receber qualquer valor. Então simplesmente ignora-se o tipo do objeto e presume-se que ele possui um determinado método, como é o caso do método "draw": ele está presente em ambas as classes "DiagramRelationship" e "DiagramElement". Apesar de não haver relação entre as duas, o mesmo método é chamado sem

---

1. http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html
2. http://www.python.org
3. http://www.java.com

questionar o tipo do objeto, como pode ser observado no método "draw" da classe "Diagram".

GTK e PyGTK

GTK[4] foi utilizada como biblioteca gráfica pela facilidade de uso, portabilidade e a facildade de integração com Python, através da extensão PyGTK[5].

Git

Git[6] foi utilizado como software de controle de versão, pela praticidade e velocidade de uso. O objetivo de utilizar um software de controle de versão foi, além de outros motivos, monitorar a evolução do software ao longo do tempo, a fim de demonstrar como o desenvolvimento incremental favorece a refatoração e, consequentemente, aumenta a qualidade geral do software.

Dia

Dia[7] foi utilizado como o software de modelagem de classes por ser prático e atender às necessidades básicas. Caso diagramas mais complexos fossem necessários, provavelmente outro software seria utilizado.

Glade

Glade[8] é um editor de interfaces gráficas para GTK. Foi utilizado para auxiliar no desenvolvimento e manutenção da interface gráfica, já que a maneira tradicional envolve codificação direta dos elementos gráficos. A vantagem do Glade é a edição visual, e não por código, e a facilidade de integração com Python: basta importar um arquivo XML gerado pelo próprio editor para se obter uma estrutura de árvore com os elementos gráficos.

---

4. http://www.gtk.org
5. http://www.pygtk.org
6. http://www.git-scm.com
7. http://www.gnome.org/projects/dia
8. http://glade.gnome.org

5 Testes

Para garantir que as mudanças feitas ao longo do desenvolvimento eram corretas, testes unitários e de integração foram implementados através do módulo de testes "doctest"[9] do Python. "doctests" são sessões interativas de execução de código, simulando o console interativo do Python. De modo geral, seu funcionamento é bastante simples: digita-se um comando e espera-se uma resposta. Essa sessão interativa é então gravada em formato de texto (no caso deste trabalho, no arquivo "tests-qoc.txt"). Para executar os testes, basta carregar essa sessão e executá-la novamente. A execução sem erros se dá sem efeitos colaterais, ou seja, nada acontece. Quando há um erro, uma saída é gerada indicando onde o erro aconteceu, o que esperava-se e o que foi obtido. Exemplos de testes executados são:

- Criação de elementos;
- Criação de relacionamentos;
- Comparação de relacionamentos;
- Verificação e atualização de relacionamentos.

Para a interface gráfica, as ferramentas de teste são relativamente complexas e, por isso, os testes não foram implementados de maneira automatizada.

---

9. http://docs.python.org/library/doctest.html