UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JULIO ARTUR DEBASTIANI

APLICATIVO DESKTOP PARA GERENCIAMENTO DE ATIVIDADES E PLANEJAMENTO DOMÉSTICOS

TRABALHO DE CONCLUSÃO DE CURSO 2

PATO BRANCO
2015

JULIO ARTUR DEBASTIANI

APLICATIVO DESKTOP PARA GERENCIAMENTO DE ATIVIDADES E PLANEJAMENTO DOMÉSTICOS

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Profa. Beatriz Terezinha Borsoi

PATO BRANCO
2015

ATA Nº: **003**

DEFESA PÚBLICA DO TRABALHO DE CONCLUSÃO DE CURSO DO ALUNO **JULIO ARTUR DEBASTIANI**.

Às 16:50 hrs do dia 24 de novembro de 2015, Bloco V da UTFPR, Câmpus Pato Branco, reuniu-se a banca avaliadora composta pelos professores Beatriz Terezinha Borsoi (Orientadora), Soelaine Rodrigues Ascari (Convidada) e Rúbia Eliza de Oliveira Schultz Ascari (Convidada), para avaliar o Trabalho de Conclusão de Curso do aluno Julio Artur Debastiani, matrícula 540978, sob o título **Aplicativo desktop para gerenciamento de atividades e planejamento domésticos**; como requisito final para a conclusão da disciplina Trabalho de Conclusão de Curso 2 do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, COADS. Após a apresentação o candidato foi entrevistado pela banca examinadora, e a palavra foi aberta ao público. Em seguida, a banca reuniu-se para deliberar considerando o trabalho **APROVADO**. Às 17:30 hrs foi encerrada a sessão.

Profa. Beatriz Terezinha Borsoi, Dr.
Orientadora

Profa. Soelaine Rodrigues Ascari, M.Sc.
Convidada

Profa. Rúbia Eliza de Oliveira Schultz Ascari, M.Sc.
Convidada

Profa. Soelaine Rodrigues Ascari, M.Sc.
Coordenador do Trabalho de Conclusão de Curso

Prof. Edilson Pontarolo, Dr.
Coordenador do Curso

RESUMO

DEBASTIANI, Julio Artur. Aplicativo desktop para gerenciamento de atividades e planejamento domésticos. 2015. 40f. Monografia de Trabalho de Conclusão de Curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

O controle das receitas e despesas domésticas é importante para que a família ou grupo de pessoas que compartilham um mesmo ambiente ou não e dividem despesas possam ter um planejamento mais efetivo das suas receitas e gastos. O planejamento financeiro também pode auxiliar na criação de hábitos sustentáveis, em termos de gastos em decorrência de receitas (do que a família ou grupo de pessoas recebe financeiramente), e que possam contibuir para a geração de poupança e investimentos. Além do controle das receitas e despesas, para um grupo de pessoas que convivem em um mesmo ambiente, é importante gerenciar a realização das atividades envolvidas nesse ambiente, incluindo as domésticas. Considerando a importância de controle financeiro e do gerenciamento de atividades que devem ser realizadas verificou-se a possibilidade de implementar um aplicativo computacional para auxiliar no gerenciamento dessas atividades (atribuição de tarefas e acompanhamento) e de receitas e despesas. Dentre as funcionalidades do aplicativo destacam-se a de permitir atribuir tarefas para as pessoas e de planejar o pagamento de despesas e de investimentos.

Palavras-chave: Gerenciamento de atividades domésticas. Aplicações desktop. Controle financeiro familiar.

ABSTRACT

DEBASTIANI, Julio Artur. Desktop application for activity management and domestic planning. 2015. 40f. Monografia de Trabalho de Conclusão de Curso - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

Control of income and household expenses is important for the family or group of people who share the same environment or not and divide expenses can have a more effective planning of their revenue and spending. Financial planning can also assist in creating sustainable habits, in terms of spending due to revenue (than family or group of people get financially), and can contribute to generate savings and investments. Beyond the control of income and expenses, to a group of people who live in the same environment, it is important to realize the management carrying out the activities involved in this environment, including domestic. Considering the importance of financial control and management activities that must be performed there was the possibility of implementing a computer application to help in managing these activities (allocation of tasks and follow-up) and revenues and expenses. Among the application's features stand out to allow assign tasks to people and to plan for the payment of expenses and investments.

Palavras-chave: Domestic activities management. Desktop application. Family financial control.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTAGENS DE CÓDIGO

LISTA DE SIGLAS

CRUD	*Create*, *Retrieve, Update and Delete*
IPTU	Imposto Predial e Territorial Urbano
IPVA	Imposto sobre a Propriedade de Veículos Automotores
RF	Requisitos Funcionais
RTTI	*Run-Time Type Information* ou *Run-Time Type Identification*

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta as considerações iniciais, os objetivos e a justificativa da realização deste trabalho. No final do capítulo é apresentada a organização do texto por meio de uma breve descrição dos capítulos subsequentes.

1.1 CONSIDERAÇÕES INICIAIS

O gerenciamento das atividades domésticas e das receitas e despesas é uma forma de auxiliar no compartilhamento da realização de tarefas e responsabilidades e para o planejamento financeiro em um ambiente familiar ou de convivência de um grupo de pessoas.

O orçamento doméstico é uma forma de controlar e acompanhar as despesas e o dinheiro da família (BOAVIASTA, 2015) ou de um grupo de pessoas que compartilham uma residência, como ocorre com as repúblicas estudantis, por exemplo. É baseado em medidas simples que visam fazer as contas de quanto dinheiro entra e quanto dinheiro sai no mês. E, assim, facilitar a identificação de (BOAVIASTA, 2015): quanto está sendo gasto (a soma das saídas financeiras), quanto dinheiro é obtido por mês (a soma de todas as entradas), se é possível gastar mais ou se é necessário economizar, programar poupança e a compra de bens, dentre outros.

A programação dos gastos, com o controle das despesas a partir das receitas pode impedir que as pessoas se endividem e que compras de bens sejam planejadas, no sentido de primeiramente guardar o dinheiro e depois adquirir o bem. Sendo, assim, possível economizar com a realização de compras a vista ao invés de a prazo, por exemplo, o que pode envolver juros nas prestações.

Além da gestão financeira, o gerenciamento das atividades a serem realizadas em um ambiente doméstico (residência, república ou outro) auxilia a definir as responsabilidades pelas tarefas e no compartilhamento do desenvolvimento dessas atividades. Cada um assumindo responsabilidades na realização das tarefas facilita o trabalho de todos e a convivência das pessoas.

Verificou-se assim, a possibilidade de desenvolver um aplicativo para ser utilizado no gerenciamento da gestão financeira e das atividades domésticas. Esse aplicativo será para

ambiente *desktop* e permitirá a atribuição de atividades aos seus responsáveis, o controle de receitas e despesas e o planejamento do pagamento de contas, por exemplo.

1.2 OBJETIVOS

A seguir são apresentados os objetivos deste trabalho, organizados em objetivo geral e objetivos específicos.

1.2.1 Objetivo Geral

Desenvolver um aplicativo *desktop* para gerenciamento da gestão financeira e das atividades domésticas.

1.2.2 Objetivos Específicos

- Oferecer recursos para o planejamento financeiro doméstico.
- Fornecer uma forma de gerenciamento de atividades e responsabilidades domésticas.

1.3 JUSTIFICATIVA

A gestão e o controle das finanças domésticas são aspectos importantes no sentido de auxiliar que as despesas não superem as receitas, colocando as pessoas em situação de dívida ou de impossibilidade de cumprir com os seus compromissos financeiros. O planejamento também é importante para a educação financeira. Por meio do planejamento é possível saber o que se pode gastar e planejar a economia necessária para compras futuras.

Além do planejamento financeiro, a atribuição de responsabilidades na realização de tarefas domésticas pode ser um fator relevante para a convivência e a organização em um ambiente compartilhado por diversas pessoas. Definir as responsabilidades pela realização das

atividades contribui para organizar o que precisa ser feito e para que as pessoas possam colaborar na realização dessas atividades.

Um aplicativo de gestão financeira e das atividades domésticas pode realizar o papel de organizador dessas atividades e auxílio no planejamento financeiro dos que convivem em uma mesma residência.

1.4 ESTRUTURA DO TRABALHO

Este texto está organizado em capítulos. O Capítulo 2 apresenta o referencial teórico. No Capítulo 3 são apresentados os materiais e o método utilizados para o desenvolvimento do trabalho. No Capítulo 4 está o resultado da realização do trabalho e no Capítulo 5 a conclusão.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta o referencial teórico do trabalho e está relacionado ao planejamento financeiro e orçamento familiar. O sistema desenvolvido como resultado deste trabalho é um aplicativo para auxiliar no gerenciamento dessas atividades.

2.1 PLANEJAMENTO FINANCEIRO E ORÇAMENTO FAMILIAR

O planejamento financeiro vai além de procedimentos destinados a disciplinar gastos e realizar poupança. O planejamento está relacionado a planejar as despesas em decorrência das receitas, planejar compras e investimentos, definir o papel e a colaboração de cada membro da residência nos gastos e na economia pretendida para um determinado período. Planejar as finanças significa entender o valor máximo que pode ser gasto sem comprometer o padrão de vida atual ou pretendido e sem comprometer planejamentos de investimentos e gastos futuros. Frankenberg (2002) define planejamento familiar como estabelecer e seguir uma estratégia que permita acumular bens para compor o patrimônio de uma pessoa ou família.

Cerbasi (2014) ressalta que planejamento financeiro não é simplesmente sinônimo de cortar gastos e fazer poupança. Planejar as finanças envolve gastar de maneira recompensadora (fazendo uma boa relação entre o custo, o valor do que está sendo comprado, e a sua utilidade ou benefício) e poupar de forma que o padrão de consumo atual possa ser mantido no futuro (CERBASI, 2014) e também para que seja possível atender imprevistos como de saúde e investimentos de oportunidade, como realizar uma compra muito vantajosa de um bem.

O planejamento financeiro permite estabelecer metas de consumo mais realistas e de acordo com as necessidades e possibilidades de receitas e despesas das pessoas. Essas metas auxiliam no planejamento de aquisições de bens e serviços em médio e longo prazo, como troca ou compra de veículo, compra ou melhoria de imóveis e investimento em educação ou mesmo viagens. Alves (2010) ressalta que com um adequado controle financeiro é possível ter qualidade de vida, mesmo com uma renda não muito alta.

A Tabela 1 apresenta o grau de dificuldade para chegar ao fim do mês das famílias brasileiras com o rendimento monetário que possuem. Esses dados são apresentados segundo a situação do domicílio e as classes de rendimento total e variação patrimonial mensal familiar no período de 2000 a 2009.

Tabela 1 - Distribuição das famílias, por avaliação do grau de dificuldade para chegar ao fim do mês

Situação do domicílio e Classes de rendimento total e variação patrimonial familiar (R$)	Distribuição das famílias (%)						
	Total (1)	Avaliação do grau de dificuldade para chegar ao fim do mês com o rendimento monetário familiar					
		Muita dificuldade	Dificuldade	Alguma dificuldade	Alguma facilidade	Facilidade	Muita facilidade
Total	100,0	17,9	21,4	35,9	14,3	9,5	1,0
Situação do domicílio							
Urbana	100,0	17,1	20.7	36.3	14.7	10.1	1.1
Rural	100.0	22.1	24.8	33.8	12.2	6.5	0.6
Classe de rendimento e variação patromonial							
Até 830 (2)	100.0	31.1	27.6	29.4	7.5	3.9	0.6
Mais de 830 a 1245	100.0	17.8	24.5	39.3	11.7	6.3	0.5
Mais de 1245 a 2490	100.0	11.6	20.1	42.9	16.0	8.9	0.5
Mais de 2490 a 4150	100.0	4.5	11.2	35.0	24.6	22.6	2.2
Mais de 4150 a 6225	100.0	4.5	11.2	35.0	24.6	22.6	2.2
Mais de 6225 a 10375	100,0	3.4	5.2	30.3	29.0	28.9	3.2
Mais de 10375	100.0	2.6	4.6	20.7	24.4	36.6	11.1

Fonte: Instituto… (2010).
Notas: 1. O termo família está sendo utilizado para indicar a unidade de investigação da pesquisa, unidade de consumo.
Notas: 2. As informações foram prestadas por um único membro indicado pela família.
(1) Não incluídas as famílias sem declaração do grau de dificuldade para chegar ao fim do mês.
(2) Inclui famílias sem rendimento.

Para o Instituto de Estudos Financeiros (INSTITUTO…, 2010) um bom planejamento financeiro inicia com a criação de um orçamento confiável. A confiabilidade está relacionada às previsões com grau aceitável de precisão. Esse Instituto ressalta que a falta de disciplina na execução do orçamento ocorre, principalmente, com as compras por impulso. É necessário que o planejamento financeiro inclua metas e limites de gastos e que as pessoas envolvidas se comprometam a efetivamente atender ao que foi planejado.

A execução e a manutenção de um orçamento familiar levam as pessoas a se depararem com uma série de situações das quais até então não tinham se dado conta (STRATE, 2010). Isso ocorre pela falta de formação sobre planejamento e economia e pela falta de prática na realização de planejamento e orçamento financeiros.

3 MATERIAIS E MÉTODO

Este capítulo apresenta os materiais e o método utilizados para a realização deste trabalho. Os materiais estão relacionados às tecnologias e ferramentas utilizadas e o método apresenta a sequência das principais atividades realizadas.

3.1 MATERIAIS

O Quadro 1 apresenta as ferramentas e as tecnologias utilizadas na modelagem e na implementação do sistema.

Ferramenta / Tecnologia	**Versão**	**Finalidade**
Astah Community	6.9.0	Modelagem do sistema (casos de uso e diagrama de classes)
Case Studio	2.25	Modelagem do banco de dados
Delphi	XE6	Linguagem de Programação/Ambiente de Desenvolvimento
SQLite	3.7.4	Banco de dados.
SQLite Studio	2.1.5	Administrador do banco de dados.
nORMot	1.18	*Framework* para acesso ao banco de dados.
FortesReport	4.0	Geração dos relatórios.

Quadro 1 – Tecnologias e ferramentas utilizadas

3.2 MÉTODO

A modelagem e o desenvolvimento do aplicativo foram realizados seguindo o modelo sequencial linear definido por Pressman (2005). A seguir estão as principais atividades realizadas em cada uma dessas etapas.

Requisitos

Uma busca na Internet por programas que realizassem as funcionalidades pretendidas de gerenciamento de orçamentos e tarefas domésticas foi realizada. Para o gerenciamento financeiro foram encontradas várias soluções. Dentre as quais podem ser citadas: ProFamilia (LOGSOFT, 2015) e as apresentadas em TecMundo (2015): FinanceDesktop, Hábil,

FinançasBR-Cofrinho, MoneyWise, iContas e FinanceMobile. Os requisitos foram definidos a partir das características gerais desses aplicativos, não foi utilizado um em particular.

Esses aplicativos, juntamente com as necessidades e interesses identificados em alguns grupos familiares analisados, auxiliaram a definir os requisitos do sistema em relação à gestão financeira. Os requisitos relacionados à gestão das tarefas domésticas foram definidos com base na experiência pessoal e análise de famílias e grupos (repúblicas, por exemplo) observados.

Modelagem

A análise e o projeto foram realizados por meio da definição dos requisitos funcionais e não funcionais que deram origem aos casos de uso. Em seguida foram definidas as entidades e relacionamentos do banco de dados.

Implementação

No desenvolvimento, inicialmente foram implementados os métodos padrões para as operações de cadastros. Em seguida esss métodos foram herdados nos formulários implementados, facilitando a codificação e a consistência da implementação.

Testes

Os testes foram unitários e visaram identificar erros de código. Os testes foram realizados pelo autor deste trabalho.

4 RESULTADO

Este capítulo apresenta o resultado obtido com a realização do trabalho. Na Seção 4.1 é apresentado o escopo do sistema modelado e desenvolvido. A Seção 4.2 apresenta a modelagem realizada. Na Seção 4.3 está a apresentação do sistema por meio das suas telas e na Seção 4.4 estão exemplos da codificação realizada.

4.1 ESCOPO DO SISTEMA

O aplicativo destina-se a prover uma forma de gerenciamento financeiro e de atividades domésticas. O gerenciamento financeiro está associado a um controle de receitas e despesas e planejamento para poupança, pagamento de despesas, investimentos, aquisição de bens e outros. O gerenciamento das atividades tem como finalidade básica organizar o que precisa ser realizado no ambiente doméstico e quem é responsável pela realização das tarefas. Os dois módulos do sistema são: gerenciamento de atividades e gerenciamento de finanças.

a) Gerenciamento de atividades

A finalidade essencial desse módulo é o gerenciamento de atividades que são todas as tarefas que precisam ser realizadas e que são atribuídas sob responsabilidade de alguém para que sejam feitas. Essas tarefas podem ser periódicas ou esporádicas e possuem um controle para a realização das mesmas. As tarefas têm um período de início e fim ou podem ser atribuídas sem data final pré-definida. Exemplos de tarefas: cuidar do cachorro, secar os pratos, levar o lixo para fora, recolher a roupa do varal e recolher os brinquedos.

O aplicativo permitirá atribuir mais de uma atividade por pessoa e mais de uma pessoa responsável por uma atividade. E gerará um “quadro de avisos” no sistema de quem é responsável por qual atividade. A pessoa responsável registra se não será possível realizar (ou se não a realizou) para que a tarefa seja atribuída para outra pessoa.

b) Gerenciamento de finanças

Registrar as receitas e despesas e fazer o planejamento financeiro, incluindo o registro de cotações de preços e a definição de metas para aquisição de bens, pagamento de dívidas e outros. Registrar a responsabilidade pelo pagamento de contas, como em repúblicas, por exemplo.

Planejar o que será comprado e o prazo para isso. Definir a economia que deverá ser realizada (por exemplo, 500 reais por mês) para adquirir o produto. Registrar se isso está sendo cumprido.

Exemplo para família:

Meta: trocar geladeira em dezembro 2015.

Ação: economizar 150,00 reais por mês. Mês a mês anotar se a economia está sendo feita.

Exemplo para crianças:

Meta: comprar determinado brinquedo para o dia das crianças

Ação: economizar 5 reais por mês do valor da mesada. Registrar mês a mês (ou outro período) o que está sendo economizando.

4.2 MODELAGEM DO SISTEMA

O Quadro 2 apresenta a listagem dos Requisitos Funcionais (RF) identificados para o sistema

Identificação	**Nome**	**Descrição**
RF 01	Manter usuários	Inclusão, exclusão, consulta e alteração de usuários e atribuição de permissões.
Caso de uso: gerenciar despesas		
RF 02	Manter despesas	Incluir, excluir, alterar e consultar despesas. As despesas são categorizadas por tipos e também é registrada a responsabilidade pela mesma, caso esteja ativa. A consulta é realizada por categoria, despesas ativas e por período. Atributos identificados: identificador, nome, descrição, estado.
RF 03	Manter tipos de despesas	As despesas podem ser categorizadas em agrupamentos como, por exemplo, fixas, esporádicas e temporárias. Atributos identificados: identificador único, nome e descrição.
Caso de uso: gerenciar receitas		
RF 04	Manter receitas	Incluir, excluir, alterar e consultar receitas. As receitas são categorizadas como, por exemplo, periódicas e esporádicas.
RF 05	Manter tipo de receitas	Tipos de receitas permitem categorizá-las.
Caso de uso: gerenciar orçamento		
RF 06	Gerenciar estado das despesas	Identificar as despesas correntes ou ativas. O estado da despesa identifica se ela está ativa ou não.

RF 07	Registrar receitas	Registrar o valor e a data de disponibilidade da receita.
RF 08	Registrar despesas	Registrar o limite para pagamento e/ou sua periodicidade das despesas, a data e a pessoa que realizou o pagamento.
RF 09	Manter contabilidade	Apresentar um balanço de receitas e despesas vigentes e correntes.
Caso de uso: gerenciar atividades		
RF 10	Manter atividades	Incluir, excluir, consultar e alterar atividades cadastradas. As atividades definem tarefas, responsabilidades e outros atribuídas aos membros da residência.
RF 11	Gerenciar estado das atividades	As atividades vigentes são tornadas ativas e é indicada a periodicidade de realização ou o prazo final de sua vigência.
Caso de uso: alterar estado da atividade		
RF 12	Controlar a realização da atividade	O responsável pela atividade ou o gerente atribui um estado atual para a atividade.
RF 13	Gerenciar estados	Estados (*status*) para a atividade são cadastrados. Exemplos: aberta, atrasada, realizada. É possível associar um ícone a cada estado, facilitando a visualização.
Caso de uso: realizar planejamento		
RF 14	Definir prioridades de despesas	Ordenar as despesas por prioridade, permitindo identificar o que pode ser eliminado.
RF 15	Estabelecer limite de gastos	Para cada despesa (exemplo: supermercado, luz, água, telefone, internet, créditos de celular, roupas, calçados, entretenimento e outros).
RF 16	Planejar compras e pagamento de despesas	Definir o que se pretende comprar, registrar o valor e estabelecer metas de economia para a aquisição do bem no prazo estabelecido. Permitir o registro para acompanhamento se as metas estão sendo cumpridas. Esse planejamento pode ser realizado para reformas, despesas como impostos (Imposto Predial e Territorial Urbano (IPTU) e Imposto sobre a Propriedade de Veículos Automotores (IPVA), aquisição de móveis e eletrodomésticos, brinquedo de crianças e etc.
RF 17	Estabelecer metas de poupança	Definir valores e periodicidade de depósito. Permitir o registro de que as metas estão sendo cumpridas.

Quadro 2 – Requisitos identificados para o aplicativo

A Figura 2 apresenta os casos de uso definidos para o aplicativo. São três atores: o administrador com acesso a todas as funcionalidades do sistema. O gerente que é o responsável pelo registro de receitas e despesas, do planejamento financeiro, registro das tarefas e atribuição de tarefas a pessoas. E o executor que registra o andamento das tarefas que ele está realizando.

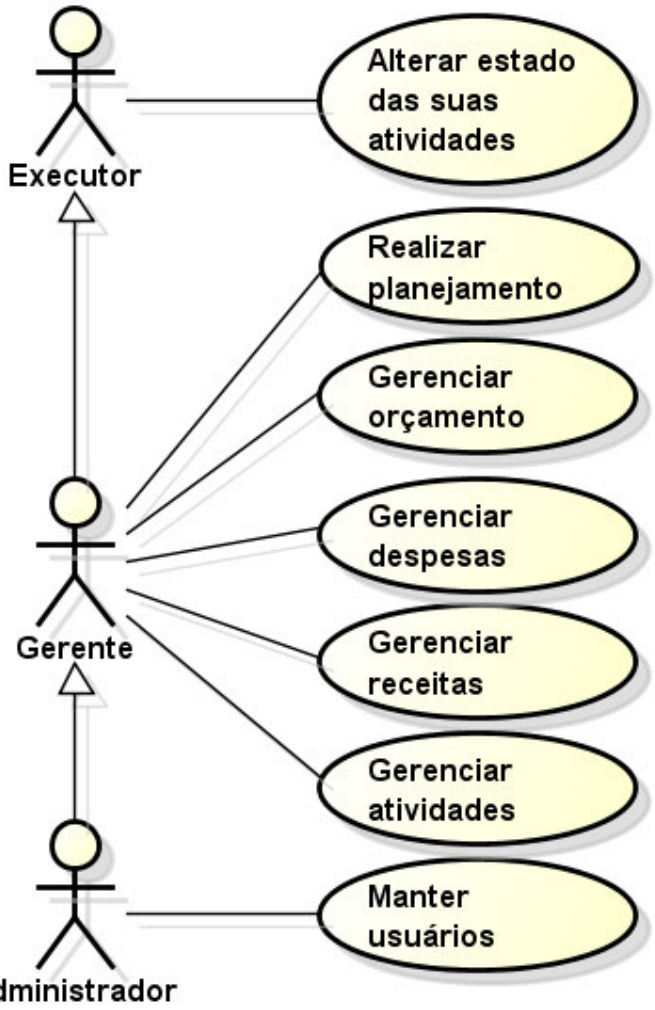

Figura 1 – Diagrama de casos de uso

No Quadro 3 está a expansão do caso de uso realizado pelo ator executor. Esse caso de uso refere-se ao acompanhamento da realização das atividades atribuída como de responsabilidade do executor.

1. Identificador do requisito: Alterar estado das suas atividades. **Descrição**: Registrar o estado de realização das suas atividades, como: em realização, finalizada, suspensa, transferida. **Evento Iniciador**: A necessidade de aterar o estado da atividade em decorrência de eventos na sua realização, como, por exemplo, a conclusão da mesma. **Atores**: Executor. **Pré-condição**: Evento que justifique mudar o estado de uma tarefa atribuída a um executor. **Sequência de Eventos**: 1- O ator executor seleciona a tarefa que pretende fazer alteração de estado. 2- Os dados da tarefa são apresentados. 3- O ator executor faz a alteração de estado. 4 – O sistema informa que o estado da tarefa foi alterado. **Pós-Condição**: As informações são salvas no banco de dados. **Requisitos não funcionais:**

Identificador	Nome	Descrição
RNF1.1	Controle de acesso	O ator executor só pode alterar o estado de tarefas atribuídas para ele.

Quadro 3 – Expansão do caso de uso alterar estado das suas atividades

Os casos de uso gerenciar atividades, receitas, despesas e orçamentos e *Create*, *Retrieve, Update and Delete* (CRUD) de usuários envolvem operações de inclusão, exclusão, consulta e alteração. As expansões dos Quadros 4 a 7 se referem a essas operações realizadas nesses quatro casos de uso. Essas operações são realizadas pelo ator gerente e pelo ator administrador, este por herança daquele.

A expansão da operação incluir está apresentada no Quadro 4.

2. Identificador do requisito: Incluir registro.
Descrição: Incluir registros no banco de dados por meio dos formulários de cadastro.
Evento Iniciador: Acesso ao formulário de receitas, despesas, orçamentos e atividades.
Atores: Gerente e administrador.
Pré-condição: Permissões para efetuar a inserção de dados.
Sequência de Eventos:
1- O ator insere os dados no formulário.
2- Os dados a serem inseridos são validados.
3- O ator recebe uma mensagem do sistema informando que a inserção foi realizada com sucesso.
Pós-Condição: Registro incluso no banco de dados.
Extensões: Em caso de erro nos dados inseridos ou dados obrigatórios faltantes, conforme o formulário de cadastro sendo utilizado, o sistema deve solicitar a inclusão do dado faltante ou correto para prosseguir.

Nome do fluxo alternativo (extensão)	Descrição
Dados faltantes	Se faltarem dados de campos obrigatórios ou campos com dados inválidos, o sistema deve retornar uma mensagem solicitando o preenchimento e informando o(s) campo(s) dos dados faltantes ou incorretos.

Inclusões: Dados faltantes.
Requisitos não funcionais:

Identificador	Nome	Descrição
RNF1.1	Controle de acesso	O ator só pode inserir dados no sistema caso possua permissão para acessar o referido cadastro.
RNF1.2	Dados incompatíveis	O sistema só pode aceitar uma inserção caso os dados sejam compatíveis com o formato do banco de dados e válidos para a regra de negócio aplicada ao campo.

Quadro 4 – Expansão da funcionalidade incluir registro

A expansão da operação excluir dos casos de uso “gerenciar” é apresentada no Quadro 5.

3. Identificador do requisito: Excluir registro.
Descrição: Excluir registros armazenados no banco de dados relacionados aos cadastros do sistema.
Evento Iniciador: Registro para ser excluído selecionado pelo ator.
Atores: Gerente e administrador.
Pré-condição: Registro não possuir vínculo com outros registros que impeçam a sua exclusão.
Sequência de Eventos:
1 – O ator exclui o registro.
2 – O registro a ser excluído é validado para evitar a geração de inconsistências no banco de dados pela exclusão de registro com vínculos.
3 – Mensagem informando que o registro foi excluído.
Pós-Condição: Registro excluído do banco de dados.
Extensões: Não deve ser possível excluir registros que estejam vinculados a outros registros no sistema. Caso não seja possível excluir o registro, o sistema deve informar ao ator o motivo da falta de exclusão.

Nome do fluxo alternativo (extensão)	Descrição
Exclusão impossibilitada	Deve ser apresentada ao usuário uma mensagem informando a impossibilidade da exclusão do registro.

Inclusões: Validar dados.
Requisitos não funcionais:

Identificador	**Nome**	**Descrição**
RNF 2.2	Validar exclusão	Os registros só podem ser excluídos caso não estejam sendo utilizados (vinculados) com outros registros do sistema.

Quadro 5 – Expansão da funcionalidade excluir registro

No Quadro 6 está a expansão da funcionalidade alteração de registros no banco de dados.

4. Identificador do requisito: Alterar registro.
Descrição: O ator pode editar e realizar alterações em dados de registros armazenados no banco de dados.
Evento Iniciador: Registro a ser alterado em edição.
Atores: Gerente e administrador.
Pré-condição: Registro ser editável.
Seqüência de Eventos:
1 – O ator altera os dados do registro em edição.
2 – O ator socilita o salvamento do registro.
3 – As alterações realizadas pelo ator são validadas para campos que assim o exigirem.
3 – O sistema emite mensagem informando a inclusão das alterações.
Pós-Condição: Os dados são alterados no registro.
Extensões: Se não é possível realizar a alteração do registro, o sistema deve informar o usuário por meio de mensagem.

Nome do fluxo alternativo (extensão)	Descrição
Alteração impossibilitada	Deve ser apresentada ao usuário uma mensagem informando o motivo da impossibilidade de inclusão das alterações no registro.

Quadro 6 – Expansão da funcionalidade alterar registro

O Quadro 7 descreve a expansão da consulta de registros.

5. Identificador do requisito: Consultar registro. **Descrição**: Consultar o conteúdo de registros armazenados no banco de dados. **Evento Iniciador**: Acesso aos cadastros para realizar consultas. **Atores**: Gerente e administrador. **Pré-condição**: Dados cadastrados. **Sequência de Eventos**: 1 – O ator informa os filtros para consulta. 2 – O sistema retorna os registros de acordo com os critérios de consulta. **Pós-Condição**: Dados como resultado da consulta apresentados para o usuário. **Extensões**:	
Nome do fluxo alternativo (extensão)	Descrição
Consulta vazia	O sistema deve informar ao usuário caso a consulta não retorne nenhum dado.

Quadro 7 – Expansão da funcionalidade consultar registros

A expansão do caso de uso realizar planejamento é apresentada no Quadro 8.

6. Identificador do requisito: Realizar planejamento. **Descrição**: Realizar planejamento financeiro. **Evento Iniciador**: Planejamento financeiro para ser realizado. **Atores**: Gerente e administrador. **Pré-condição**: Planejamento financeiro realizado. **Sequência de Eventos**: 1 – O ator informa os dados necessários para o registro de planejamento financeiro. 2 – O sistema retorna os registros de acordo com os critérios de consulta. **Pós-Condição**: Dados do planejamento realizado incluídos no sistema.

Quadro 8 – Expansão do caso de uso realizar planejamento

A Figura 2 apresenta o diagrama de entidades e relacionamentos que representam o banco de dados parcial da aplicação. Essas entidades do banco referem-se aos usuários do sistema e às funcionalidades relacionadas ao planejamento financeiro. Esse diagrama será, posteriormente complementado, com dados sobre o registro e acompanhamento de atividades a serem realizadas. A tabela Pessoas também será utilizada para atribuir executores às atividades. A tabela TamanhoTelas será utilizada para armazenar definições em termos de tamanho e localização das telas do sistema. A tabela LctosFinanceiros guardará os lançamentos financeiros feitos pelo usuário. A tabela Tarefas armazenará todas as tarefas pendentes e realizadas, bem como as demais informações referentes a essa rotina. Na tabela Usuários ficarão as informações das pessoas que poderão ter acesso ao sistema.

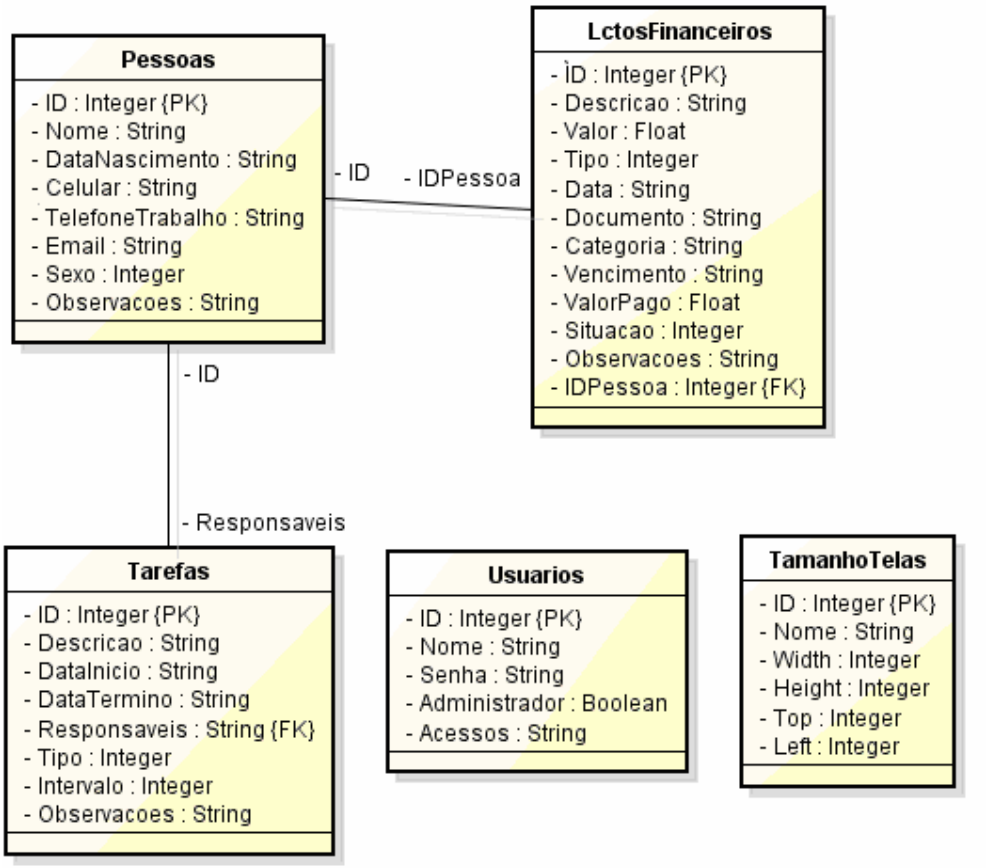

Figura 2 – Diagrama de entidades e relacionamentos do banco de dados

4.3 APRESENTAÇÃO DO SISTEMA

A seguir são apresentadas as principais funcionalidades do sistema por meio das telas desenvolvidas. A tela principal do sistema, apresentada na Figura 3, um menu superior contendo o acesso às funcionalidades relacionadas aos cadastros de usuários, pessoas, tarefas e lançamentos financeiros. Na parte central da tela é apresentada a listagem das tarefas que "ativas", ou seja, estão sendo realizadas (periódicas) ou ainda não foram realizadas. O gráfico na parte inferior apresenta a quantidade de tarefas realizadas no período.

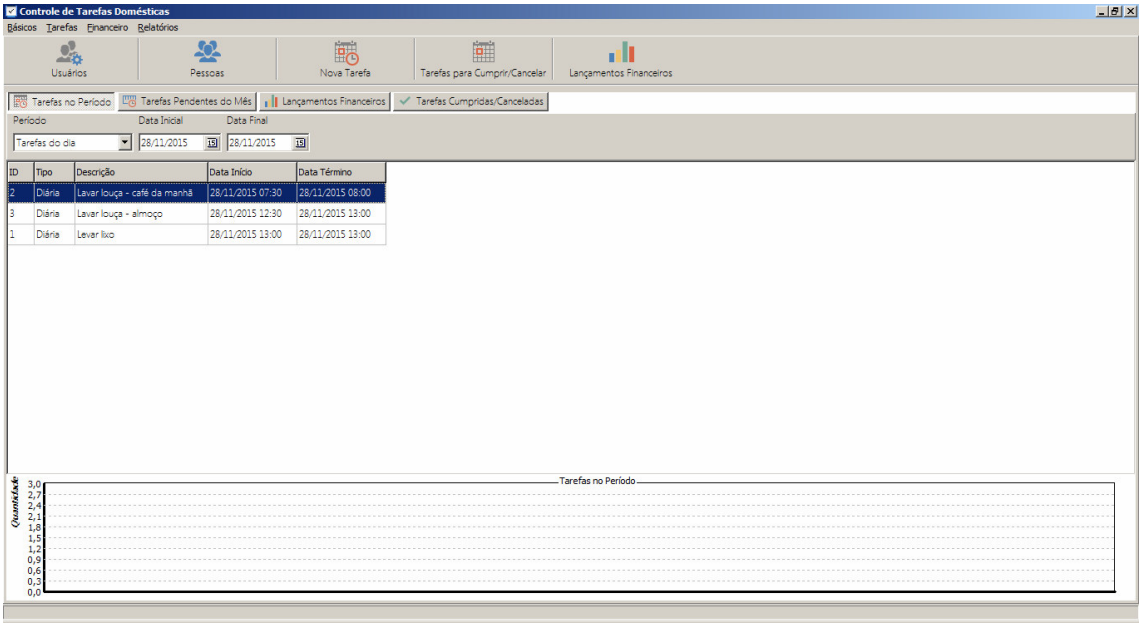

Figura 3 – Tela de resumo de registros de tarefas

A Figura 4 apresenta o cadastro de tarefas, no qual o usuário pode incluir as tarefas que deseja controlar. Os campos apresentados nessa tela podem variar de acordo com o valor escolhido para o campo duração estimada para a realização da tarefa. No exemplo apresentado na Figura 6 é como a tela será montada caso o usuário selecione Nenhum, 30 minutos, 1 hora, 90 minutos ou 2 horas.

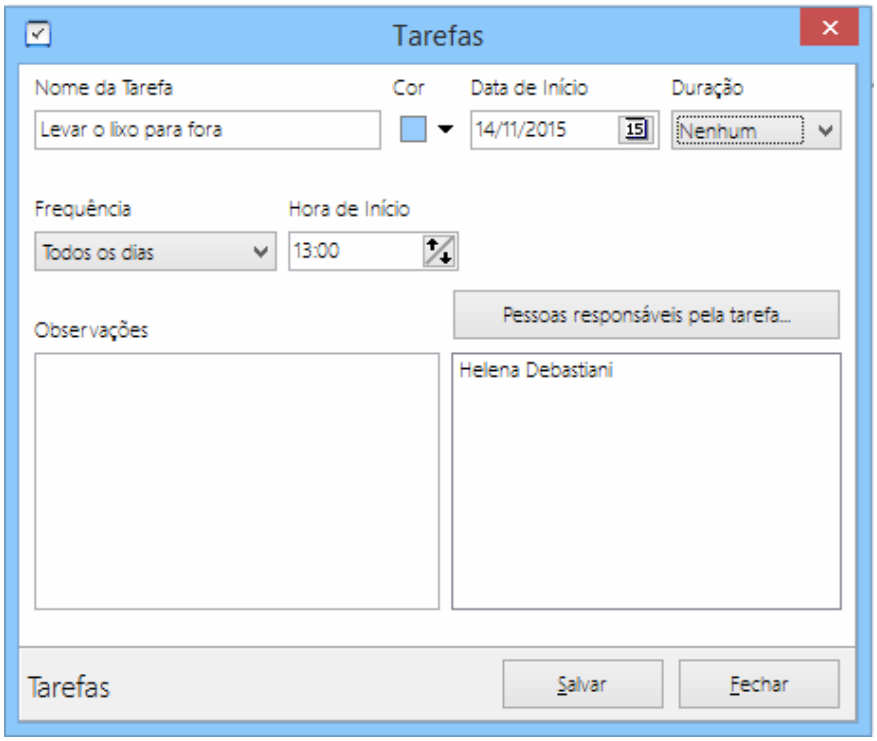

Figura 4 – Cadastro de tarefas

Caso a opção Dia inteiro seja selecionada o campo Hora de Início não aparecerá, visto que para esse caso a tarefa inicia-se às 0h00 indo até 23h59 do dia selecionado em Data de Início. Para o valor Personalizado a tela aparecerá como apresentado na Figura 5.

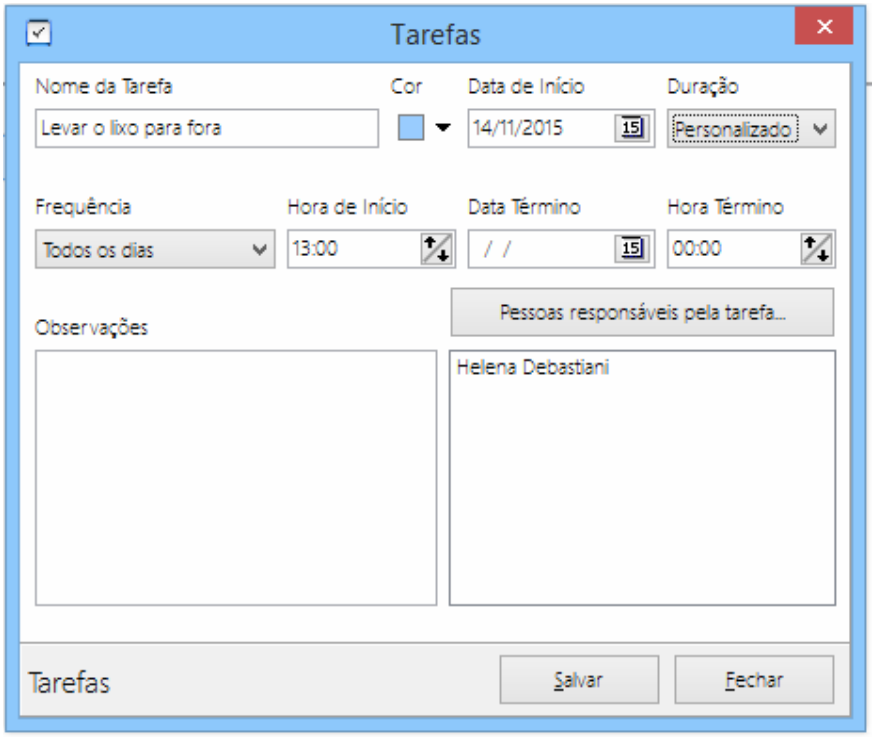

Figura 5 – Cadastro de tarefas com intervalo personalizado

Se a tarefa durar mais do que um dia ou durar um intervalo de tempo diferente dos disponíveis, o usuário poderá selecionar essa opção. Dessa maneira será possível selecionar a data e hora de início juntamente com a data e hora de término da tarefa.

Também é possível selecionar a frequência da tarefa por meio do campo Frequência, que é preenchido de acordo com a data selecionada na data inicial. Supondo que a data selecionada tenha sido quatro de novembro de 2015, as opções disponíveis são: Tarefa única, Todos os dias, Toda Quarta-feira, Todo dia 04 de cada mês e Todo 04 de novembro, conforme apresentado na Figura 6.

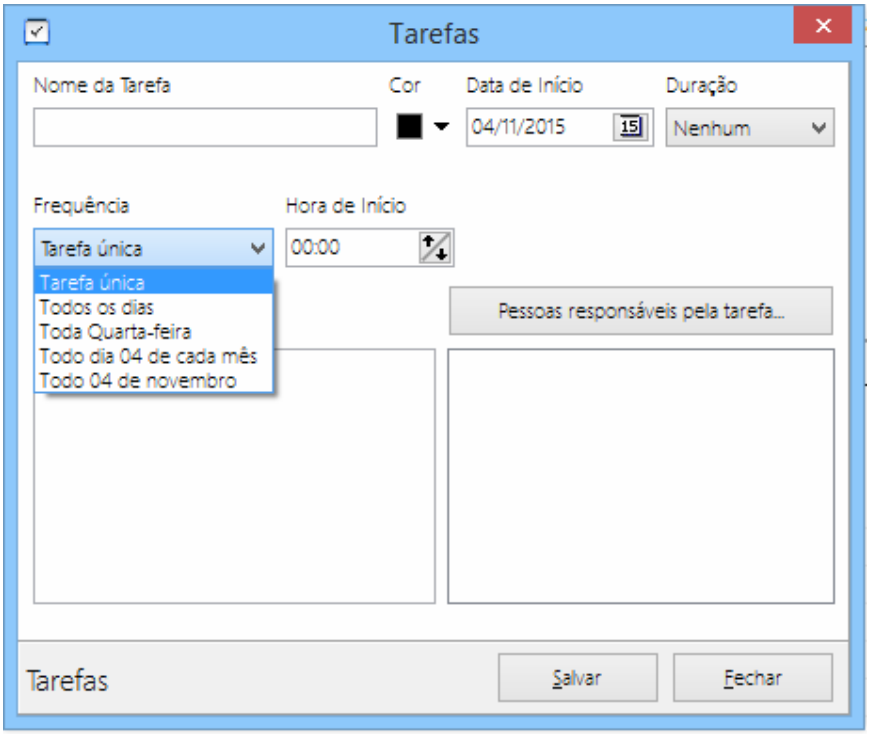

Figura 6 – Cadastro de tarefas: modelo de frequência

Nesta tela (Figura 6) também é possível selecionar os responsáveis pela tarefa cadastrada. É possível não selecionar nenhum usuário (somente cadastrar a tarefa), selecionar somente um usuário ou vários ao mesmo tempo. Isso poderá ser usado para filtrar as tarefas visualizadas na tela inicial do programa.

Na tela inicial existe uma área denominada Tarefas Pendentes no Mês (Figura 7), nessa área o usuário poderá visualizar as tarefas de um determinado mês e ainda filtrar por pessoas responsáveis.

Figura 7 – Tarefas pendentes no mês

Cada retângulo apresentado (área central da Figura 7) representa uma tarefa que precisa ser cumprida no dia. Clicando com o botão direito do mouse sobre os retângulos aparecerá a opção para cumprir a tarefa selecionada, significando que a tarefa já foi realizada. Fazendo isso ela deixará de ser apresentada como uma tarefa que deve ser cumprida. Clicando sobre o campo Filtrar por responsáveis é possível selecionar uma ou mais pessoas, limitando a visualização das tarefas somente às pessoas selecionadas. A cor do retângulo é definida no momento do cadastro da tarefa. E visa facilitar a visualização de tarefas por agrupamentos ou responsáveis, por exemplo.

Além de cumprir tarefas dessa forma, também é possível fazê-lo acessando o menu Tarefas -> Tarefas para Cumprir/Cancelar (Figura 8). Ao fazer isso, é apresentada uma tela na qual o usuário poderá cancelar ou cumprir a tarefa selecionada. É possível escolher o intervalo de dias e as pessoas responsáveis. Para cumprir a tarefa basta selecioná-la no *grid*, marcar a opção [F2] Cumprir Tarefa e clicar em Cumprir [F4]. Para cancelar é necessário selecioná-la no grid, marcar a opção [F3] Cancelar Tarefa, escrever uma justificativa do porque a tarefa está sendo cancelada e depois clicar em Cancelar [F4].

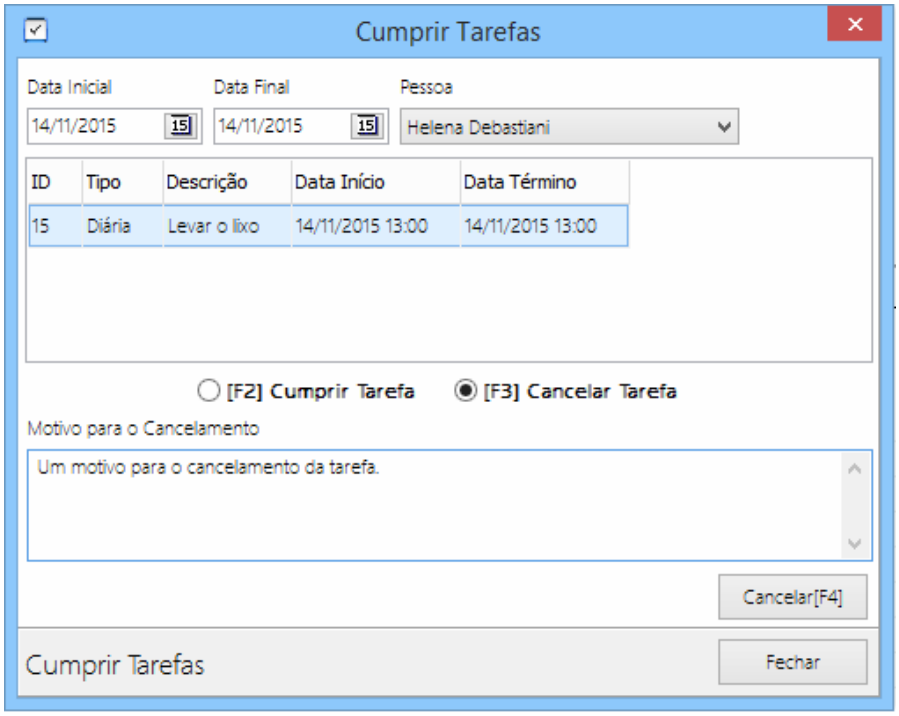

Figura 8 – Cumprir/Cancelar tarefas pendentes

Além do lançamento e controle de atividades, o sistema permite um controle financeiro simplificado que é composto basicamente pela registro de receitas e despesas. A tela que é apresentada quando essa funcionalidade é acessada é apresentada na Figura 9. Os lançamentos de receitas e despesas são apresentados em uma listagem e na parte inferior da tela é apresentado um gráfica que ilustra a relação entre receitas e despesas.

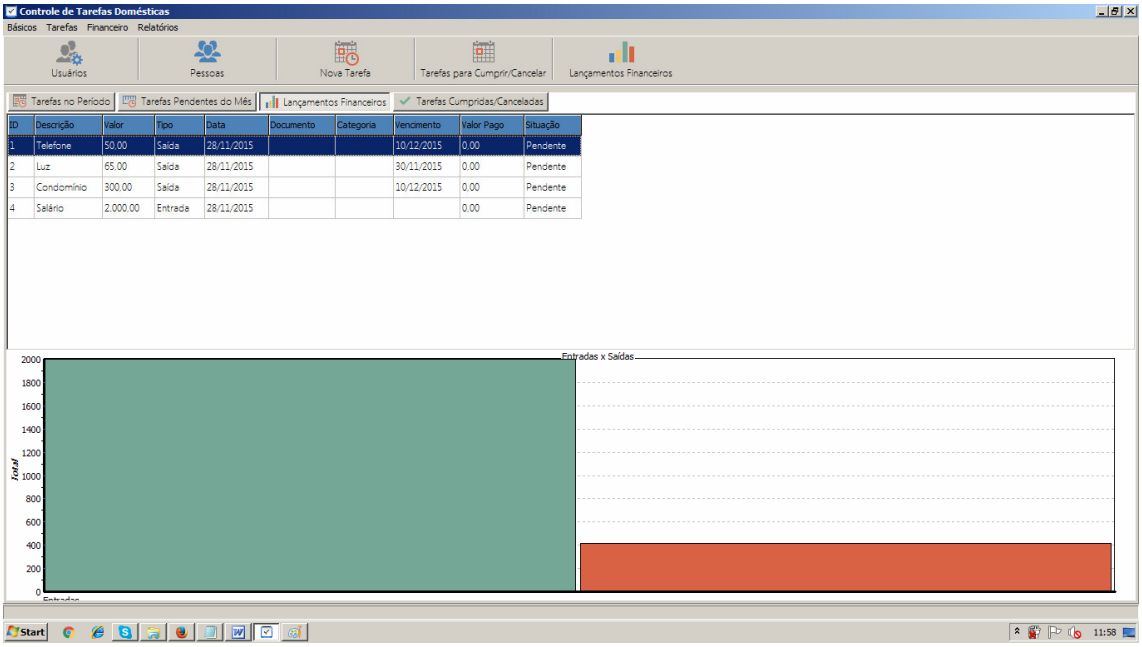

Figura 9 – Tela de resumo de lançamentos financeiros

A tela da Figura 10 apresenta o formulário para lançamentos financeiros. Descrição se refere a uma identificação textual do lançamento realizado e a situação permite identificar se já foi pago, no caso de despesa e o tipo se é entrada (receita) ou saída (despesa). A categoria permite classificar os recebimentos e pagamentos para que possa ser realizado um controle mais efetivo dos gastos e das proporções entre eles e as receitas, por exemplo.

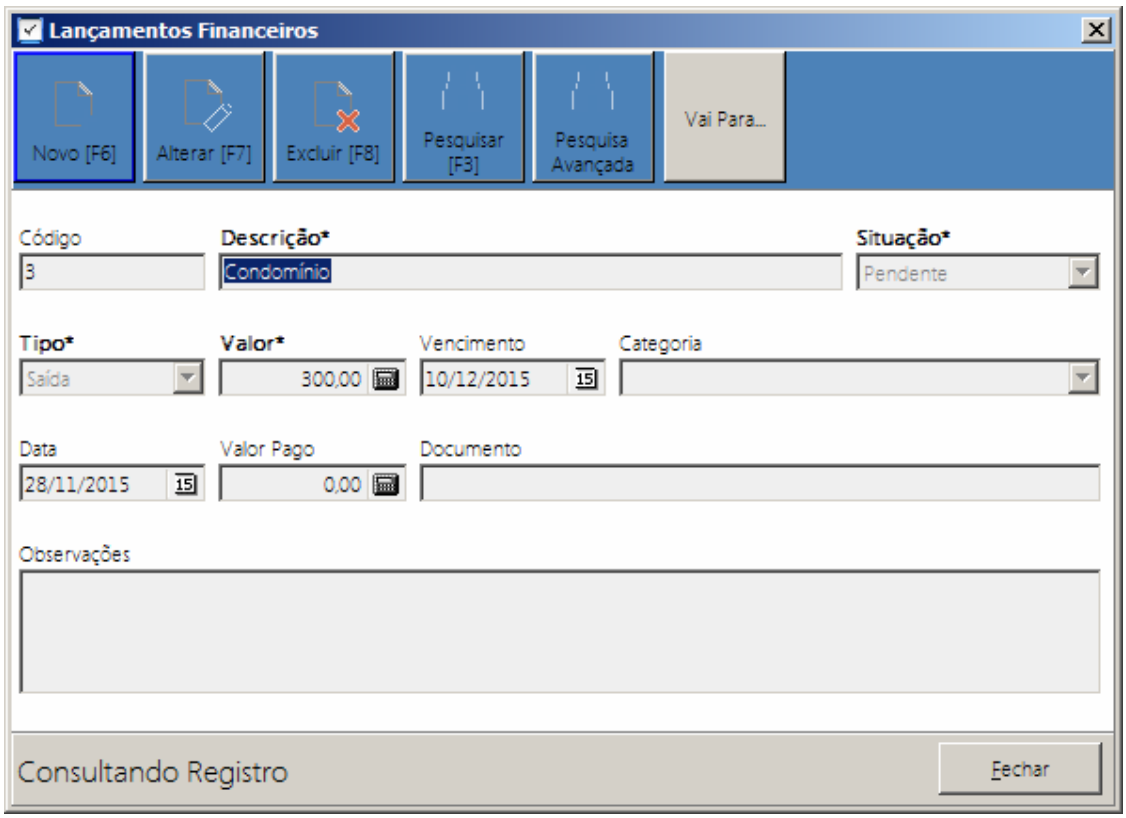

Figura 10 – Tela padrão para cadastros

Em modo de consulta todos os campos ficam desabilitados para edição. A partir do momento que o usuário inicia uma inserção ou edição do registro visualizado, os campos são liberados para alteração.

Clicando no botão Pesquisar [F3] é apresentada uma tela que permite pesquisar os registros usando o campo principal (descrição ou nome, dependendo da tabela). A Figura 11 apresenta essa tela.

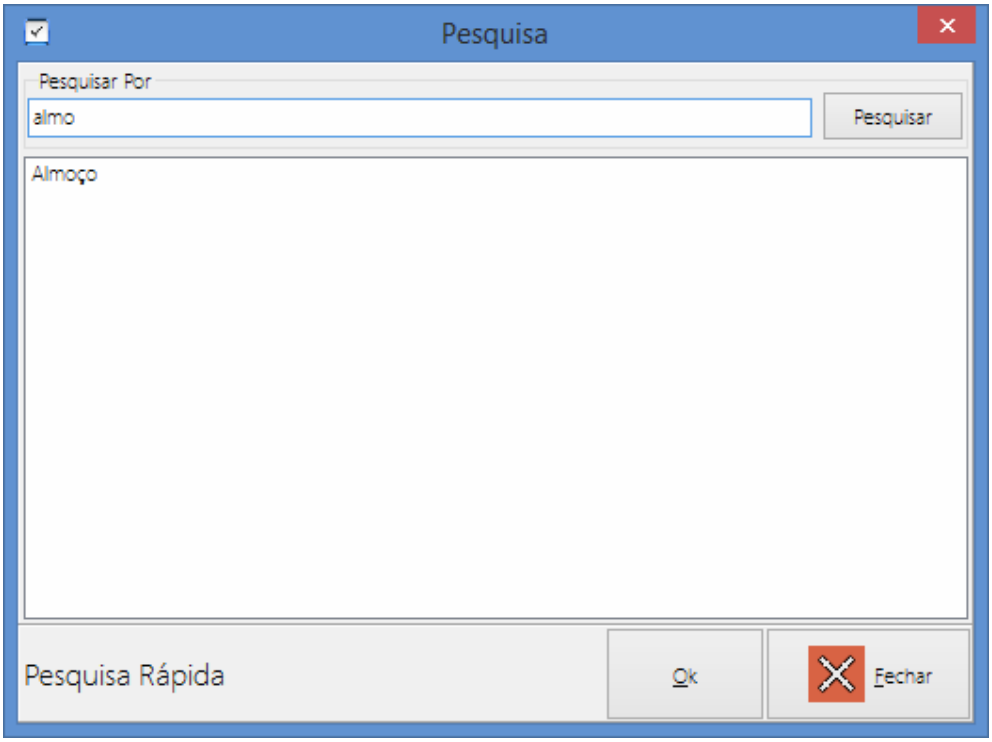

Figura 11 – Pesquisa padrão usando apenas um campo

Caso o usuário necessite de uma pesquisa mais completa, poderá utilizar a opção de Pesquisa Avançada. Ao clicar no botão aparecerá a tela conforme Figura 12.

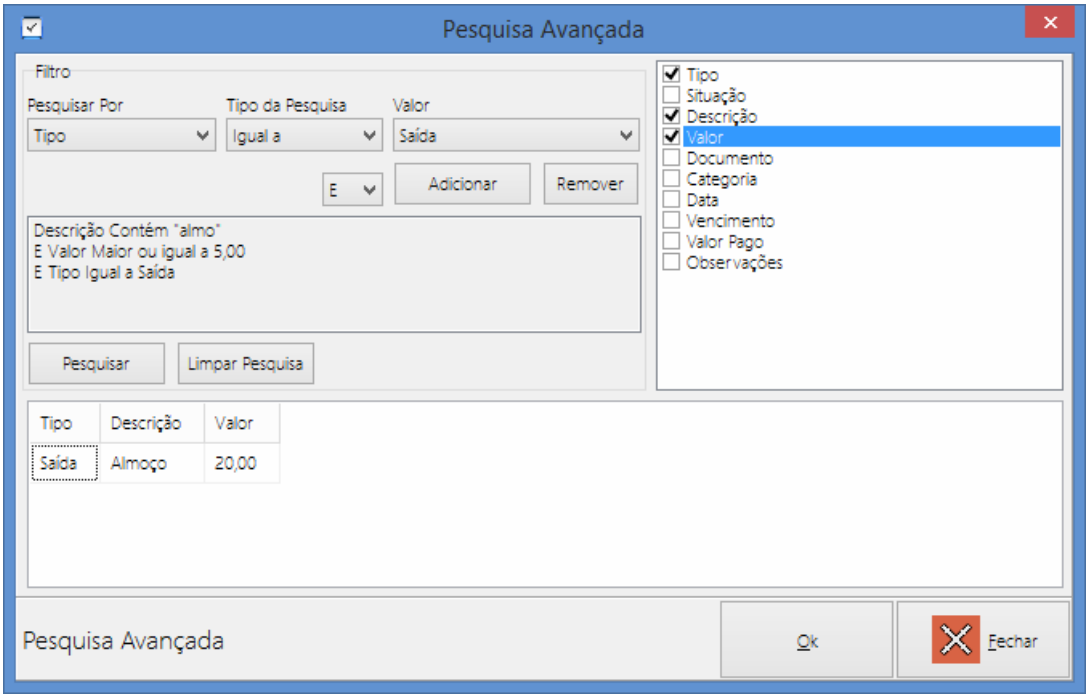

Figura 12 – Pesquisa avançada

Essa tela (Figura 12) permite ao usuário selecionar os campos que deseja usar na pesquisa, a condição usada para o campo, o cognitivo E ou OU e quais campos deseja visualizar no *grid* com o resultado. Esta tela é preenchida de forma genérica, de acordo com a tabela selecionada.

4.4 IMPLEMENTAÇÃO DO SISTEMA

Nesta seção são apresentados exemplos de códigos implementados no desenvolvimento do sistema. Para conectar com o banco de dados usando o mORMot é muito simples como apresentado na Listagem 1.

```
procedure TConexaoBancoDados.CriaBancoDados;
begin
  try // cria banco de dados;
    FModelo := CriaModelo;
    FDados := TSQLRestServerDB.Create( FModelo, FPathBanco );
    FDados.CreateMissingTables( 0 );
    if FDados.DB.User_Version < 1 then
      FDados.DB.User_Version := 1;
  except
    on e: exception do
      raise Exception.Create( '($CBD48)' + e.Message );
  end;
end;
```

Listagem 1 – mORMot para conexão com o banco de dados

A conexão é feita por meio da classe TSQLRestServerDB. Ao criar um objeto dessa classe é necessário informar o modelo do banco de dados e o nome do arquivo. O modelo é um *array* de TSQLRecordBase. Cada item do *array* é uma tabela que será criada no banco de dados. Se mais tabelas forem criadas, a função *CreateMissingTables* se encarregará de criar as que não existirem.

As classes que conectam no banco de dados herdam da classe *TSQLRecordBase* e usam uma classe chamada *TCustomAttribute*. Esse recurso permite criar propriedades personalizadas que podem ser acessadas posteriormente via *Run-Time Type Information* ou *Run-Time Type Identification* (RTTI). Esse recurso foi usado no programa para definir o nome do campo que aparecerá para o usuário na pesquisa. Também é usado na validação dos campos. Na Listagem 2 está um exemplo de como isso é feito.

```
TSQLLctosFinanceiros = class( TSQLRecordBase )
private
  FDescricao: RawUTF8;
  FValor: real;
  FTipo: TLctosFinanceirosTipo;
  FData: TDateTime;
  FDocumento: RawUTF8;
  FCategoria: RawUTF8;
  FVencimento: TDateTime;
  FValorPago: real;
  FSituacao: TLctosFinanceirosSituacao;
  FObservacoes: RawUTF8;
published
  [ TValidacao( 'Tipo', tcEnum ) ]
  [ TInformacoesCampo( TSQLLctosFInanceiros, 'Tipo', '', tcEnum,
'Entrada,Saída' ) ]
  property Tipo : TLctosFinanceirosTipo read FTipo write FTipo;
  [ TValidacao( 'Situação', tcEnum ) ]
  [ TInformacoesCampo( TSQLLctosFinanceiros, 'Situacao', 'Situação', tcEnum,
'Pendente,Baixado' ) ]
  property Situacao : TLctosFinanceirosSituacao read FSituacao write
FSituacao;
```

Listagem 2 – Criação de propriedades para uso via RTTI

A *TCustomAttribute* TValidacao é responsável pela validação do campo, ou seja, antes de salvar o CRUD é verificado se o campo é obrigatório por meio desse atributo. Os parâmetros que ele exige são: o nome do campo e o tipo (para saber qual forma de validação usar).

A *TCustomAttribute* TInformacoesCampo é usada para preencher as informações que serão mostradas ao usuário nas pesquisas, mensagens, entre outras. No exemplo do código na Listagem 2, o atributo Situacao não é acentuado, então na hora de apresentar para o usuário é necessário mostrar com a acentuação correta. Usando as propriedades *RTTI* da classe, é acessada essa propriedade e capturado o nome amigável desse atributo. Os parâmetros esperados são: a classe, o nome do campo, o nome amigável, o tipo e, caso ele seja um atributo que tenha opções (tipo *Enum* ou um *combo* com opções fixas), os valores que o campo aceita. Um exemplo de como utilizar esses recursos está na Listagem 3.

```
procedure TDicionarioDados.PreencheDicionario(AClasse:
  TSQLRecordClass);
var
  ctx : TRttiContext;
  t   : TRttiType;
  p   : TRttiProperty;
  a   : TCustomAttribute;
  dc  : TInformacoesCampo;
begin
  ctx := TRttiContext.Create;
  try
```

```
    t := ctx.GetType( AClasse );
    for p in t.GetProperties do
      for a in p.GetAttributes do
        if a is TInformacoesCampo then begin
          dc := TInformacoesCampo( a );
          dc.LancaDadosNoDicionario;
        end;
  finally
    ctx.Free;
  end;
end;
```

Listagem 3 – Classe TCustomAttribute

Usando as classes TRttiContext, TRttiType, TRttiProperty e TCustomAttributes é possível acessar as informações que foram definidas na classe e apresentadas na Listagem 3. Para acessar as informações da classe TCustomAttribute é necessário fazer laços de repetição até chegar nas propriedades desejadas.

Para o controle das tarefas com frequência diferente de *Única* foi necessário implementar a criação de tarefas em memória. Somente a tarefa principal fica armazenada no banco de dados. As cópias são criadas no momento em que o usuário efetua alguma consulta. As cópias das tarefas só são salvas no banco de dados no momento em que são cumpridas ou canceladas. Antes disso, elas existem somente em memória.

Para fazer isso foi criada uma classe chamada *TTarefa*, similar a *TSQLTarefas* (Listagem 4), mas que armazena as tarefas em memória. Essas tarefas são armazenadas em uma lista implementada na classe *TListaDeTarefas*, que herda de um objeto *TList<TTarefa>*. Por meio do método *PegaTarefasNoPeriodo* as tarefas são selecionadas no banco de dados e, caso possuam algum tipo de frequência, são criadas em memória nesse método.

```
  ATarefas := TSQLTarefas.CreateAndOpen( dm.Conexao, ' status=' +
    Integer( tsAberta ).ToString + CondicaoParaPessoas( APessoas ) +
    ' and (( tipo=' + Integer( ttUnica ).ToString +
    ' and datainicio between ' + fn.DataHoraToSQL( ADataInicial ) +
    ' and ' + fn.DataHoraToSQL( ADataFinal ) + ' ) ' +
    ' or (tipo<>' + Integer( ttUnica ).ToString + '))' );
  try
    ATarefas.Ordena( 'DataInicio', true );
    while ATarefas.FillOne do begin

      if ATarefas.Tipo = ttUnica then
        Self.Add( CriaTarefa( ATarefas ) )
      else begin

        dAux := ADataInicial;
        while dAux <= ADataFinal do begin

          case ATarefas.Tipo of
            ttTodosOsDias: begin
              if not ATarefas.TemNaData( dAux +
                fn.ExtraiTime( ATarefas.DataInicio ) ) then
```

```
                Self.Add( CriaTarefa( ATarefas, dAux +
                  fn.ExtraiTime( ATarefas.DataInicio ) ) );
              end;
              ttTodaSemana: begin
                if fn.DiaDaSemana( dAux )=fn.DiaDaSemana(ATarefas.DataInicio) then
                  if not ATarefas.TemNaData( dAux +
                    fn.ExtraiTime( ATarefas.DataInicio ) ) then
                    Self.Add( CriaTarefa( ATarefas, dAux +
                      fn.ExtraiTime( ATarefas.DataInicio ) ) );
              end;
              ttTodoMes: begin
                if fn.FormatData(dAux, 'dd')=fn.FormatData(ATarefas.DataInicio,
                  'dd' ) then // se for no mesmo dia;
                  if not ATarefas.TemNaData( dAux +
                    fn.ExtraiTime( ATarefas.DataInicio ) ) then
                    Self.Add( CriaTarefa( ATarefas, dAux +
                      fn.ExtraiTime( ATarefas.DataInicio ) ) );
              end;
              ttTodoAno: begin
                if fn.FormatData(dAux, 'ddmm')=fn.FormatData(ATarefas.DataInicio,
                  'ddmm' ) then // se for no mesmo dia e mês
                  if not ATarefas.TemNaData( dAux +
                    fn.ExtraiTime( ATarefas.DataInicio ) ) then
                    Self.Add( CriaTarefa( ATarefas, dAux +
                      fn.ExtraiTime( ATarefas.DataInicio ) ) );
              end;
            end;

            dAux := dAux + 1;
          end;
        end;
      end;
    finally
      ATarefas.Free;
    end;
```

Listagem 4 – Classe TSQL Tarefas

É a partir dessa classe que todas as consultas de tarefas são criadas. Sempre que alguma rotina do programa precisa mostrar tarefas é a classe TListaDeTarefas que é chamada, porque ela poderá criar as tarefas que não existem no banco de dados.

Para o desenvolvimento de relatórios foram utilizados componentes do FortesReport. Esses componentes só conectam no banco de dados utilizando o componente *TDataSet.* Como no projeto desenvolvido a conexão com o banco de dados é feita por meio de *framework* não havia uma forma de acesso nativa.

Para o desenvolvimento de relatórios havia uma questão importante, os campos do tipo *Enum* aparecem de uma forma para o usuário, mas são representados de maneira diferente no banco de dados. Assim, ao gerar os relatórios algumas informações poderiam aparecer confusas. O Quadro 9 exemplifica para a tabela tarefas os valores armazenados no banco de dados e os valores que devem ser apresentados ao usuário.

Valor para o usuário	Valor no banco de dados
Única	0
Todos os dias	1
Toda demana	2
Todo mês	3
Todo ano	4

Quadro 9 – Valor apresentados para o usuário e armazenado no banco de dados

Todas essas informações podem ser obtidas nas classes *TCustomAttributes* que foram implementadas juntamente com as classes de conexão com o banco de dados. Para obter a informação já filtrada foi necessário criar um objeto do tipo *TSQLRecordBase* contendo as informações do banco de dados. Esse objeto é lido e passa todas as suas informações para um *TClientDataSet* usando os recursos para "traduzir" as informações de maneira que fique claro para quem for ler o relatório. Esse *TClientDataSet* está relacionado a um *TDataSet* que faz a comunicação com os componentes do FortesReport, usados para confecção dos relatórios.

Para exemplificar como isso foi feito, na Listagem 5 está o código que traduz as informações da classe *TSQLRecordBase* para a classe *TClientDataSet*.

```
procedure TfrmBaseReport.CriaDados(ARegistros: TSQLRecordBase);
var
  ADados: TSQLDadosDicionario;
  f: TField;
begin
  CriaCamposNoDataSet( ARegistros.RecordClass );

  cdsReport.DisableControls;

  ADados := dm.Dicionario.PegaDados( ARegistros.RecordClass );
  try
    while ARegistros.FillOne do begin
      cdsReport.Append;

      ADados.FillRewind;
      while ADados.FillOne do begin
        f := cdsReport.FieldByName( ADados.Campo );
        if ADados.Tipo = tcEnum then
          f.Value := ADados.ValoresFormatados.Strings[
            StrToInt( ARegistros.GetFieldValue( f.FieldName ) ) ]
        else
          f.Value := ARegistros.GetFieldVariant( f.FieldName );
      end;

      cdsReport.Post;
    end;
  finally
    ADados.Free;
```

```
  end;

  cdsReport.First;
  cdsReport.EnableControls;
end;
```

Listagem 5 – Classe que traduz de TSQLRecordBase para a classe TClientDataSet

Para criar os campos no *TClientDataSet* foi utilizada a função apresentada na Listagem 6, que captura as informações do dicionário de dados e cria os campos, cada um no seu tipo correspondente.

```
Procedure TfrmBaseReport.CriaCamposNoDataSet(const Aclasse: TSQLRecordClass);
var
  AFieldType: TFieldType;
  ADados: TSQLDadosDicionario;
  iTamanho: integer;
begin
  ADados := dm.Dicionario.PegaDados( AClasse );
  try
    with cdsReport, cdsReport.FieldDefs do begin
      Clear;
      Close;
      while ADados.FillOne do begin
        case ADados.Tipo of
          tcEnum,
          tcTexto   : AFieldType := ftString;
          tcDataHora,
          tcData    : AFieldType := ftDateTime;
          tcValor   : AFieldType := ftFloat;
          tcMemo    : AFieldType := ftMemo;
          tcBoolean : AFieldType := ftBoolean;
          tcInteiro : AFieldType := ftInteger;
        else
          raise Exception.Create('$BaseReport Tipo nao implementado!');
        end;
        iTamanho := 0;
        if AFieldType = ftString then
          iTamanho := 100;

        Add( ADados.Campo, AFieldType, iTamanho );
      end;
      CreateDataSet;
    end;
  finally
    ADados.Free;
  end;
end;
```

Listagem 6 – Captura as informações do dicionário de dados e cria os respectivos campos

5 CONCLUSÃO

O objetivo deste trabalho foi o desenvolvimento de um sistema para a gestão financeira e de atividades domésticas. Essa gestão é no sentido de planejamento de investimentos e de contas a pagar e rendimentos de pessoas, seja uma família ou um grupo de pessoas que compartilham uma residência. Um sistema de controle simples foi implementado, mas atende aos requisitos propostos.

Em termos das tecnologias utilizadas, é possível dizer que existem muito mais vantagens do que desvantagens ao desenvolver sistemas utilizando o *framework* mORMot para acesso ao banco de dados. As conclusões se referem a esse *framework* porque foi o mais extensivamente utilizado no desenvolvimento do sistema.

Uma das vantagens é que o código-fonte fica mais legível e de fácil entendimento. Isso porque a forma de desenvolvimento aproxima-se muito mais da linguagem natural. Para acessar um campo usando um componente nativo seria necessário escrever quClientes.FieldByName('nome').AsString. Já usando uma classe desse *framework*, o acesso ao campo é feito escrevendo Cliente.Nome. A quantidade de caracteres e símbolos também é menor, o que aumenta a rapidez da codificação.

Outra vantagem é velocidade do acesso aos dados em comparação com os componentes nativos do Delphi. O mORMot foi criado para ser rápido, então ao selecionar uma quantidade grande de registros a velocidade de retorno é muito maior.

Uma das desvantagens é a complexidade no uso de duas ou mais tabelas em um mesmo select. Apesar de o componente oferecer uma maneira de fazer *inner* e *left join* (para no máximo duas tabelas), não é algo que funcione de uma maneira prática. Então, a alternativa foi usar um componente chamado TSQLTable, que se assemelha mais aos componentes nativos do Delphi para acesso ao banco de dados. Dependendo da quantidade de informações envolvidas o desempenho pode ser afetado.

O uso da classe TCustomAttributes também foi muito interessante nesse projeto, porque permitiu que várias das rotinas desenvolvidas fossem feitas de maneira genérica, facilitando a manutenção e aumentando a rapidez para implementação.

Uma das desvantagens de se desenvolver com o *framework*, é que os componentes de relatório utilizados não trabalham diretamente com a conexão de banco de dados criada pelo *framework*, e as informações recuperadas no banco de dados para geração do relatório de campos do tipo Enumerate não ficariam claras para o usuário final. Isso também foi contornado usando a classe TCustomAttribute. As informações foram resgatadas usando as

classes padrão do mORMot e transformadas em um objeto TClientDataSet, que combinado com um objeto da classe TDataSet forneceu o suporte necessário para o desenvolvimento dos relatórios.

Uma das desvantagens é que esse processo pode apresentar lentidão para casos de relatórios com grande volume de informações, porque para gerar corretamente o programa precisa resgatar as informações do banco de dados, lê-las até o final e depois mostrar as informações na tela lendo novamente do início ao fim. Se o acesso fosse nativo as informações seriam lidas somente uma vez.

Como funcionalidades de implementação futura para o aplicativo podem ser definidos novos relatórios e metas financeiras individuais ou coletivas. Por exemplo: economizar um determinado valor e registrar os valores que estão sendo economizados para alcançar essa meta.

REFERÊNCIAS

ALVES, Alessandra Batista de Melo Nóbrega. **Planejamento financeiro familiar e orçamento doméstico**: prática e importância em um grupo no município de Cataguases – MG. 2010. Trabalho de Conclusão de Curso. Faculdades Sudamérica.

BOAVISTA SERVIÇOS. **Cartilha do orçamento doméstico**. Disponível em: <http://www.boavistaservicos.com.br/wp-content/uploads/2013/12/Cartilha_Boa_Vista.pdf>. Acesso em: 30 set. 2015.

CERBASI, Gustavo. **Como ser sustentável com suas finanças**, 2014. Disponível em: <http://www.maisdinheiro.com.br/artigos/6/118/como-ser-sustentavel-com-suas-financas> Acesso em: 24 mar. 2015.

FRANKENBERG, Louiz. **Guia prático para cuidar do seu orçamento: viva melhor sem dívidas**. Rio de Janeiro: Elsevier, 2002.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. **Pesquisa de orçamentos familiares 2008-2009**. 2010. Diretoria de Pesquisas, Coordenação de Trabalho e Rendimento. Disponivel em: <http://www.ibge.gov.br/home/estatistica/populacao/condicaodevida/pof/2008_2009/tabelas_pdf/tabela3_1_1.pdf>. Acesso em: 2 abr. 2015.

LOGSOFT SISTEMAS - **Programas em Excel VBA. ProFamilia 4.6.3 – Freeware.** Disponível em: < http://www.logsoft.com.br/profamilia.asp>. Acesso em: 10 abr. 2015.

PRESSMAN, Roger. **Engenharia de software**. Rio de Janeiro: McGraw-Hill, 2005.

STRATE. Anete Berenice Schaeffer. **Implicações provenientes da elaboração de um orçamento familiar**. 2010. Mestrado Programa de Pós-Graduação Mestrado Profissional em Ensino de Ciências Exatas. Centro Universitário Univates. Disponível em: <http://www.univates.br/bdu/handle/10737/107>. Acesso em: 30 abr. 2015.

TECMUNDO. **10 programas e apps para organizar suas finanças pessoais**. Disponível em: < http://www.tecmundo.com.br/tutorial/27033-10-programas-e-apps-para-organizar-suas-financas-pessoais.htm>. Acesso em: 5 abr. 2015.