CHARLES DANILO DE SOUZA

DESENVOLVIMENTO DE SOFTWARE PARA CÁLCULO DE PROPRIEDADES DE PRODUTOS ARMAZENADOS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

LAVRAS
MINAS GERAIS – BRASIL
2005

CHARLES DANILO DE SOUZA

DESENVOLVIMENTO DE SOFTWARE PARA CÁLCULO DE PROPRIEDADES DE PRODUTOS ARMAZENADOS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Área de Concentração:
Desenvolvimento de software

Orientador:
Professor Francisco Carlos Gomes

LAVRAS
MINAS GERAIS – BRASIL
2005

Ficha Catalográfica preparada pela Divisão de Processos Técnicos da Biblioteca Central da UFLA

Souza, Charles Danilo de

Desenvolvimento de Software para Cálculo de Propriedades de Produtos Armazenados / Charles Danilo de Souza. Lavras – Minas Gerais, 2005. 57 páginas.

Monografia de Graduação – Universidade Federal de Lavras. Departamento de Ciência da Computação.

1. Propriedades físicas. 2. Produtos agrícolas. 3. Modelagem. 4. Software. I. SOUZA, C. D. II. Universidade Federal de Lavras. III. Título.

CDD

CHARLES DANILO DE SOUZA

DESENVOLVIMENTO DE SOFTWARE PARA CÁLCULO DE PROPRIEDADES DE PRODUTOS ARMAZENADOS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em 08 de julho de 2005.

_____
Professor Dr. Heitor Augustus Xavier Costa

_____
Professor Dr. André Luiz Zambalde

_____
Professor Dr. Francisco Carlos Gomes (Orientador)

LAVRAS
MINAS GERAIS – BRASIL
2005

AGRADECIMENTOS

Agradeço a minha família por ter me apoiado integralmente durante esta fase. Agradeço também meu orientador Francisco pelo apoio dado durante este trabalho e a todos que cooperaram para a conclusão do curso.

RESUMO

DESENVOLVIMENTO DE SOFTWARE PARA CÁLCULO DE PROPRIEDADES DE PRODUTOS ARMAZENADOS

A demanda de sistemas construtivos e de estruturas para armazenamento de produtos agrícolas tais como silos têm crescido constantemente. O projeto de construção destas estruturas requer o conhecimento das ações que nelas atuam. Uma das ações mais importantes é devida aos produtos armazenados, calculadas a partir das propriedades físicas destes produtos. A determinação das propriedades se dá por uma rotina de cálculo que nos conduz a erros. Dessa forma, existia a necessidade do desenvolvimento de um software para cálculo destas propriedades, visando fornecer os resultados mais exatos possíveis para que fossem mantidas a integridade da estrutura de armazenamento e a qualidade do produto armazenado. O presente trabalho expõe os conceitos importantes utilizados e o processo de desenvolvimento e modelagem do software. Além disso, são apresentados os resultados do desenvolvimento, a aplicabilidade do software e as melhorias obtidas com o novo modo de processamento dos dados conseguidos nas pesquisas.

Palavras-chaves: propriedades físicas, produtos agrícolas, modelagem, software.

ABSTRACT

DEVELOPMENT OF SOFTWARE FOR CALCULATION OF PROPERTIES OF STORED PRODUCTS

The demand of constructive systems and of structures for storage of such agricultural products as silos has been growing constantly. The project of construction of these structures requests the knowledge of the actions that act in them. One of the most important actions is owed to the stored products, calculated starting from the physical properties of these products. The determination of the properties is made by a calculation routine that leads us to mistakes. In that way, the need of the development of software for calculation of these properties existed, seeking to supply the possible most exact results so that the integrity of the storage structure and the quality of the stored product were maintained. The present work exposes the important concepts used, the development process and the modeling of the software. Besides, the results of the development, the applicability of the software and the improvements obtained with the new way of processing of the data gotten in the researches are presented.

Keywords: physical properties, agricultural products, modeling, software.

RESUMO ESTENDIDO

DESENVOLVIMENTO DE SOFTWARE PARA CÁLCULO DE PROPRIEDADES DE PRODUTOS ARMAZENADOS

A demanda de estruturas para armazenamento de produtos agrícolas tais como silos têm crescido constantemente. O projeto de construção destas estruturas requer o conhecimento das ações que nelas atuam. Uma das ações mais importantes é devida aos produtos armazenados, calculadas a partir das propriedades físicas destes produtos. Para o cálculo dessas propriedades, são usadas máquinas construídas para simular a condição de armazenamento do produto em silos, onde se realizam testes de cisalhamento sobre amostras de grãos. Desses testes, são obtidas as tensões a serem usadas nas fórmulas que calculam essas propriedades. Como as rotinas de cálculo existentes conduzem a erros e as fórmulas que fazem o tratamento matemático dos dados não são triviais, então é necessário o uso de sistemas computacionais aplicando as mesmas equações para que se consiga resultados exatos. Dessa forma, existia a necessidade do desenvolvimento de um software para cálculo destas propriedades, visando fornecer os resultados mais exatos possíveis para que fossem mantidas a integridade da estrutura de armazenamento e a qualidade do produto armazenado. O presente trabalho expõe os conceitos importantes utilizados e o processo de desenvolvimento e modelagem do software. Além disso, são apresentados os resultados do desenvolvimento, a importância e aplicabilidade do software e as melhorias obtidas com o novo modo de processamento dos dados conseguidos nas pesquisas.

Palavras-chaves: propriedades físicas, produtos agrícolas, modelagem, software.

SUMÁRIO

ÍNDICE DE FIGURAS

1. INTRODUÇÃO

1.1. Considerações Iniciais

A cada ano, a produção de grãos no Brasil tem aumentado gradativamente, embora a produtividade seja alta, ainda perde-se muito na pós-colheita, com o transporte e com os sistemas de armazenamento precários. Existe uma demanda para que novas estruturas sejam construídas para o armazenamento dos grãos produzidos. Entre essas estruturas as principais são os silos.

Os projetos de silos para armazenamento de produtos agrícolas dependem diretamente do cálculo das propriedades físicas desses produtos, visando o seu dimensionamento de maneira segura e econômica, com o tratamento matemático dos dados.

Para o cálculo dessas propriedades, são usadas máquinas construídas para simular a condição de armazenamento do produto em silos, onde se realizam testes de cisalhamento[1] sobre amostras de grãos. Desses testes, são obtidas as tensões a serem usadas nas fórmulas que calculam essas propriedades. Como são obtidos muitos dados e tais fórmulas não são triviais, então é necessário o uso de sistemas computacionais aplicando as mesmas equações para que se consiga resultados exatos.

1.2. Objetivos e Justificativas

O objetivo geral foi desenvolver um aplicativo para facilitar o tratamento dos dados obtidos, na determinação das propriedades de produtos armazenados. A finalidade do desenvolvimento de um novo programa é aprimorar os produtos de software existentes, que utilizam o ambiente MS-DOS, ou mesmo planilhas eletrônicas. O requisito para o desenvolvimento desse aplicativo, é que utilize o ambiente *Windows*, com uma interface intuitiva ao usuário.

Propôs-se a fazer os cálculos das propriedades e criar arquivos para armazenar os dados de entrada, e arquivos com o resultado do processamento.

---

[1] Cisalhamento é o rompimento abrupto do elemento do produto pela atuação de forças de atrito.

Os objetivos específicos desta pesquisa podem ser descritos nos itens a seguir:

- Fornecer telas para a entrada de dados a partir do teclado e para abrir arquivos com os dados salvos;
- Apresentar os ângulos de atrito obtidos do processamento dos dados;
- Ilustrar o gráfico com as retas representando o lugar geométrico e os círculos de tensões nos produtos;
- Possibilitar da introdução de tipos de células de características diferentes das células padrão.

A eficácia é uma característica importante, considerando que o objetivo principal é calcular as propriedades dos produtos armazenados, os quais serão usados em projetos de silos, então os resultados devem ser consistentes e exatos. A finalidade deste software também é atender as necessidades do usuário, pois em caso contrário, o usuário poderá voltar a utilizar o método anterior; ao mesmo tempo deve ser atrativo, pois em caso contrário ele terá seu uso desestimulado. A tendência a levar a erros do usuário ou do programa deve ser evitada, mas, se ocorrer, o usuário deverá ser avisado e instruído a corrigir o problema.

O desenvolvimento de um software que atenda a tais solicitações deve ter uma modelagem que atenda aos métodos padrões da engenharia de software. Pois tais métodos definem todas as tarefas para a construção de programas, desde a definição dos requisitos até os testes finais e reparo de erros notificados na distribuição ao usuário.

O próximo capítulo, o referencial teórico descreve o que são e como são determinadas as propriedades dos produtos agrícolas, descreve a importância da interface da qualidade da interface de software e os principais recursos utilizados para o desenvolvimento da interface, descreve ainda como é realizada a modelagem de software, os principais diagramas utilizados. No capítulo metodologia são descritos os procedimentos e as ferramentas utilizadas para o desenvolvimento do software. Em resultados e discussões são descritas a modelagem e a implementação obtidas, o funcionamento do software com telas mostrando a execução. Em conclusões são apresentadas as conclusões obtidas e a importância do desenvolvimento do trabalho.

2. REFERENCIAL TEÓRICO

Este capítulo apresenta referências bibliográficas consideradas importantes para o desenvolvimento deste trabalho, tais como as propriedades dos produtos armazenados, as pesquisas para a obtenção dessas propriedades, os conceitos importantes no desenvolvimento de interfaces de software, uma introdução à engenharia de software e a modelagem de software em *Unified Modeling Language* (UML – Linguagem de Modelagem Unificada).

2.1. Propriedades dos Produtos Armazenados

As estruturas de armazenamento são de grande importância para a manutenção da qualidade final dos produtos armazenados. Atualmente, existe a necessidade de desenvolvimento de novas tecnologias e soluções dos problemas existentes no que diz respeito aos critérios de dimensionamento destas unidades. É de fundamental importância um estudo aprofundado das alternativas cabíveis para o armazenamento seguro dos produtos, dentre elas, destaca-se o conhecimento do comportamento dos produtos armazenados a granel.

Sendo as propriedades físicas dos produtos armazenados diferentes das propriedades dos líquidos, o projeto de silos se torna difícil, principalmente no que diz respeito a fluxo contínuo e o comprometimento da segurança e da viabilidade econômica destas estruturas. Para atender a esta necessidade, é importante que as cargas não sejam subestimadas e nem superestimadas. Assim, no caso de armazenamento a granel, a principal meta em um projeto de silos por descarga por gravidade consiste em garantir o fluxo ininterrupto do produto armazenado, mantendo-se também a integridade da estrutura do silo, sendo para isto, necessário conhecer as propriedades físicas dos produtos armazenados (MILANI, 1993; CALIL JR., 1990).

Dentre as propriedades mais importantes, destacam-se: peso específico, ângulo de atrito interno, efetivo ângulo de atrito interno, ângulo de atrito do produto com a parede e o coeficiente de atrito entre o produto e a parede da célula (GAYLORD e GAYLORD, 1984; MOSHENIN, 1986).

2.1.1. Peso Específico

LATINCSICS (1985) apresentou um estudo sobre a influência da compressibilidade dos sólidos armazenados em silos. Observou que, embora os métodos de cálculo utilizados, tais com os de JANSSEN (1895) e outros similares, não levem em consideração essa influência, a maioria dos sólidos são relativamente compressíveis, afetando significativamente parâmetros do projeto. Constatou que os métodos de projeto existentes, tais como a norma americana ACI (1983) ou como a similar alemã DIN-1052 (1969), omitem análises resultantes da compressibilidade do produto armazenado, o que levou a concluir que o uso desses métodos, especialmente nos casos de materiais fofos com alta compressibilidade ou alto grau de compressão, resulta em valores incorretos de tensões, forças e pesos. Enfatizou que a compressibilidade deve ser considerada no projeto de um silo, tendo em vista os efeitos que pode causar no comportamento da estrutura.

BRITTON e MOYSEY (1986) observaram que o peso específico do produto armazenado em um silo é função de sua umidade, das sobre-pressões existentes no silo, do tempo de armazenamento, da taxa de carregamento, do modo de carregamento e da altura de queda do produto. Constataram que os valores reais desse parâmetro em geral divergem dos estabelecidos pela comissão de grão do Canadá (*Canada Grain Commission*) ou pelo departamento de agricultura dos Estados Unidos. Recomendaram que, para projeto, um aumento médio de 6% sobre os tais valores deve ser considerado.

2.1.2. Coeficiente de Atrito com a Parede

SCHWEDES (1983), comparando os interesses da engenharia de processos com os da engenharia civil no que diz respeito à determinação do ângulo de atrito com a parede, observou que, para combinações idênticas entre o produto armazenado e parede, as medidas desse parâmetro podem variar em até mais de 10º. Verificou que tais medidas, obtidas por um teste de cisalhamento, nem sempre resultam em uma linha reta. Creditou essa variação ao comportamento de meio não continuo do produto armazenado. Sugeriu que para o coeficiente de atrito com a parede seja considerado um intervalo de variação.

CALIL JR. (1984, 1985) determinou, a partir de dados experimentais de pressão obtidos em um modelo de silo, os ângulos de atrito interno e os ângulos de atrito do produto armazenado com a parede e analisou sua variação em função da relação entre a altura do produto armazenado e o lado da seção transversal do modelo. Dessa análise, concluiu-se que, para as relações altura/lado, 3,0 e 1,5, há discrepâncias na determinação das cargas em silos. Neste intervalo, segundo o autor, os ângulos passam de um valor constante para uma variação linear e, à medida que diminui o valor do ângulo de atrito interno, aumenta o valor do ângulo de atrito com a parede. Observou que a anomalia notada nos ensaios mostra o erro de utilização da fórmula de JANSSEN (1895) no cálculo de silos com relação altura/lado pequena.

SCHWEDES (1985) observou que valores, do coeficiente de atrito com a parede, obtidos por teste de cisalhamento apresentam, em geral, grande dispersão, devendo, portanto, este parâmetro ser estabelecido por meio de um intervalo. Mencionou a possível influência que outros parâmetros exercem sobre ele, tais como: velocidade de cisalhamento, temperatura, umidade, tempo, diferença entre atrito estático e dinâmico, vibração, etc. Observou também que, para o projeto estrutural, a influência da velocidade deve ser irrelevante, que produtos armazenados sobre pressão durante um dado período de tempo podem levar a um aumento substancial do coeficiente de atrito e que a presença de vibrações pode reduzir a resistência ao cisalhamento em até 30%. Sugeriu que os valores do coeficiente de atrito com a parede sejam estabelecidos para cada problema individualmente.

BRITTON e MOYSEY (1986) observaram que os valores de coeficiente de atrito com a parede, obtidos para um dado produto, em geral apresentam diferenças significativas. Segundo os autores, essa variação seria causada pela diversidade de processos utilizados na obtenção desse parâmetro. Recomendaram que o coeficiente de atrito seja tratado como constante em silos de paredes lisas e que, para paredes de aço corrugado, seja assumido como constante e estimado como a tangente do ângulo de atrito interno.

HAAKER (1988) propôs um novo tipo de equipamento para a determinação do coeficiente de atrito com a parede onde fatores tais como a velocidade de deslizamento, a

pressão normal, a temperatura, a presença de vibrações e o comportamento do atrito são considerados. Dos valores medidos com esse equipamento, pôde concluir que o coeficiente de atrito com a parede para uma certa combinação produto/parede não é um valor único e pode depender fortemente da velocidade de deslizamento e menor proporção da pressão normal; pode mudar significativamente com o tempo devido à variação das características das paredes, causada pela ação do deslizamento do produto.

ZHANG et al (1988) realizaram testes de cisalhamento direto para estudar o comportamento do atrito entre o trigo e alguns materiais construtivos (alumínio, aço galvanizado, madeira compensada e concreto) e propuseram uma função exponencial para modelar o comportamento desse atrito. Verificaram que simulações do modelo se comparam favoravelmente aos dados obtidos nos testes. O modelo proposto, segundo os autores, pode ser prontamente adaptado ao modelo de elementos finitos para caracterizar adequadamente a interação entre produto armazenado e estrutura.

BUKLING et al (1989) realizaram testes sobre amostras de aços não galvanizados e alumínio para estudar fontes de variação no coeficiente de atrito do trigo sobre o metal. Os fatores analisados foram tipo de metal, a pressão exercida pelos grãos e o número de ensaios. Segundo os autores, o coeficiente de atrito variou com a pressão e com o tipo de metal e diminuiu à medida que o deslizamento dos grãos sobre a parede foi se repetindo.

CHUNG e VERMA (1989), em um estudo experimental sobre coeficientes de atrito estático e dinâmico, verificaram que os efeitos de umidade e do tipo de superfície foram significativos. Segundo o estudo, a umidade influenciou mais o atrito estático e a superfície mais o atrito dinâmico. O trabalho relaciona ainda alguns autores que buscaram estabelecer formas de medidas para este parâmetro.

CHUNG e VERMA (1989), investigaram o comportamento do coeficiente de atrito com a parede, levando em consideração o efeito de diferentes rugosidades e da presença de descontinuidades nas paredes causadas por dobras, soldas, sobreposições, etc. Os testes realizados mostraram que esse parâmetro é sensível às descontinuidades em uma superfície, sejam elas paralelas ou normais à direção do movimento. O autor sugeriu uma relação semi-empírica para explicar o comportamento do coeficiente de atrito com a parede em função dos efeitos observados.

2.1.3. Produto Armazenado e Lugar Geométrico

Conforme CALIL JR. (1990), um produto armazenado em um recipiente é solicitado por pressões que causam consolidação e fornecem resistência. As pressões mais importantes ocorrem durante o fluxo, isto é, durante a deformação contínua do produto acima do seu limite.

Consideremos o elemento do produto armazenado em um silo, mostrado na Figura 2.1, $\sigma 1$ e $\sigma 2$ são respectivamente a maior e a menor tensão principal e são indicadas pelo semicírculo de *MOHR*. Se o elemento para esta condição de consolidação é cisalhado sob várias cargas normais, então é obtido o lugar geométrico de deslizamento. O semicírculo de *MOHR*, através da origem, define a tensão de deslizamento inconfinada, que representa a resistência do material em uma superfície livre. Estendendo o lugar geométrico para interceptar o eixo τ, é definida a coesão aparente C. O angulo ϕ é o angulo de atrito interno. A coesão C é a tensão de cisalhamento sob tensão normal nula, e mesmo sendo uma propriedade do produto armazenado, ela não é usada na teoria de fluxo em silos.

Portanto, a determinação das propriedades dos materiais armazenados depende do conhecimento dos lugares geométricos de deslizamento determinados pela relação entre a tensão de cisalhamento e a tensão normal para o material armazenado, avaliando como ele desliza relativamente a si próprio e relativamente ao material da parede de construção do silo. Esta informação é obtida de testes a partir da célula de cisalhamento, obtendo-se assim a tensão sob armazenamento e as condições de fluxo que podem ocorrer em unidades de armazenamento.

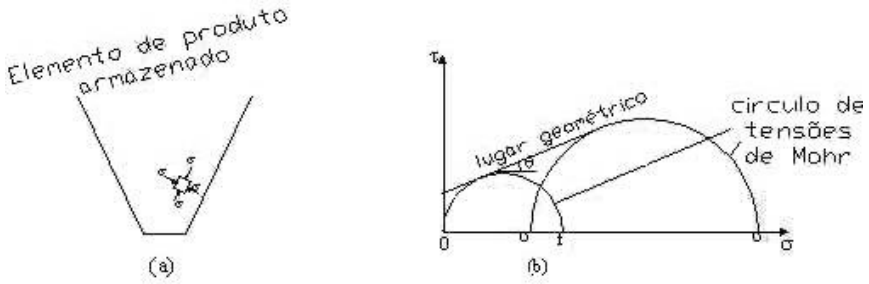


Figura 2.1 – Lugar geométrico de deslizamento do produto armazenado

Segundo JENIKE (1964), os produtos armazenados apresentam propriedades elasto-plásticas e viscosas. Algumas informações podem ser obtidas de uma análise

baseada na suposição de que produtos armazenados se comportam com um modelo rígido – plástico do tipo de Coulomb.

Chamando a tensão de cisalhamento de τ, e a tensão normal de σ e usando um sistema de coordenadas (τ x σ), o limite da função de tensões é representado por um lugar geométrico YL. Para o sólido de Coulomb, rígido – plástico, o lugar geométrico é uma linha inclinada de um ângulo de atrito ϕ, com eixo σ, e interceptando o eixo τ, no valor igual a C, o qual é chamado de coesão do sólido.

Em testes instantâneos, o sólido é conduzido a uma condição de deformação estável, sobre uma pressão normal pré-definida em um plano de cisalhamento. A determinação do lugar geométrico de deslizamento (YL) e do lugar geométrico de deslizamento da parede (WYL) representa também o caminho para predizer a forma da tremonha e as dimensões da boca de descarga, que irão fornecer, um determinado fluxo do material armazenado e um pré-determinado tipo de fluxo.

2.2. Pesquisas sobre as Propriedades Físicas dos Produtos Agrícolas

Um dos estudos considerados de maior importância na determinação das propriedades físicas dos produtos armazenados em silos foi apresentado por JENIKE (1964). Em busca de uma forma adequada para medir tais propriedades, foi analisada a aplicabilidade de equipamentos de teste utilizados em solos. Diante dos resultados considerados não satisfatórios, decidiu-se desenvolver um aparelho de cisalhamento direto apropriado para produtos a serem armazenados. O aparelho denominado "*Jenike Shear Cell*" tem sido usado, desde então, por diversos pesquisadores europeus.

Com o objetivo de simplificar os ensaios de propriedades, vários equipamentos estão sendo testados, buscando a diminuição dos custos com aplicação de equipamentos como os de cisalhamento direto o que fatalmente permitiria um maior domínio das teorias envolvidas para o processamento de produtos.

Para Milani (1993), as pesquisas para a obtenção das propriedades são realizadas na máquina de Jenike, com procedimentos padronizados de aplicação de quantidades

determinadas de repetições, diferentes pressões sobre a célula onde se encontra o produto, pela máquina. A máquina oferece como resultado de cada repetição a força de tensão normal e a força de atrito, o qual é usado como entrada de fórmulas para os cálculos dos ângulos de atrito, ou seja, como entrada para o aplicativo.

A partir das forças obtidas, das características da célula usada, das forças aplicadas sobre a célula e da rugosidade da superfície que será usada nas paredes do silo; poderá se determinar através de cálculos tais ângulos de atrito dos produtos armazenados. Em cada teste, pode-se usar uma célula com característica diferente e uma pressão diferente sobre o produto e, conforme suas características, o resultado é uma força diferente.

O uso das propriedades dos produtos, vindas das tabelas das normas internacionais, segundo Silva (2003), conduz a erros expressivos nos projetos de fluxos e pressões dos silos. Portanto, é essencial que sejam feitos testes e usados produtos de software, seguindo sempre as normas internacionais, para o cálculo de tais propriedades.

2.3. Conceitos de Desenvolvimento de Software

2.3.1. Importância de uma Interface de Software com Qualidade

A interação se torna mais fácil quando os usuários conseguem assimilar um modelo mental consistente e íntegro do software, expõe Paula Filho (2003). Todas as interações devem diminuir a necessidade de memorização e a possibilidade de erros. O modelo mental expressa como o usuário entende o funcionamento do sistema, se houver exceções e inconsistências, o usuário sente dificuldades.

O usuário espera sempre usar um software que atenda às suas necessidades, mas que lhe requeira o mínimo esforço. Segundo Bonsiepe (1997), citado em Cunha (2004), a interface gráfica computacional é definida com uma especificação dos objetos que o usuário vê no monitor e as regras para sua interação é uma ferramenta de interação entre o homem e a máquina.

Cunha (2004) cita que a interface é tão importante que influencia até na produtividade do usuário, pois ele nem sempre quer um sistema com mais recursos ou eficiente do ponto de vista computacional, ele quer um sistema que tenha utilidade no seu trabalho. A interface determina, em grande parte, o sucesso ou o fracasso de um produto, dada a sua importância.

2.3.2. Principais Conceitos para o Desenvolvimento de Interfaces

Um sistema não deve decepcionar seus usuários, segundo Heckel (1993), seja por falhas de hardware, software ou por erro do usuário, os sistemas devem impedir ou ao menos amenizar os danos. Existem muitas técnicas que podem dar confiabilidade a um sistema, como testes, técnicas de garantia de qualidade, provas de exatidão e metodologias, os quais devem ser usados em qualquer software para garantir o crédito de um sistema.

Conforme Manzano, A. e Manzano, M. (1998), o MS-DOS tem como característica uma interface de linha de comando, onde o usuário tem que decorar os comandos de sua tarefa. A partir do *Microsoft Windows*, o usuário dispõe de uma interface gráfica amigável, com ícones de desenho intuitivo e mouse para realizar suas tarefas.

Isso facilitou muito para o usuário leigo, pois agora ele se preocupa mais com a sua tarefa e menos com interface do software que usa. Foi tanta a revolução que a interface gráfica provocou, que pouco tempo depois, não se usava sistema operacional sem interface gráfica e a *Microsoft* ainda domina o mercado mundial de sistemas operacionais. A partir desses fatores, pode-se ver a importância de que seja desenvolvido um software, em substituição aos usados atualmente de ambiente MS-DOS.

A interface de um software ao nível de projeto, qualquer que ele seja, deve ser tão importante quanto a sua eficiência, pela importância que o usuário considera. Segundo Heckel (1993), o projetista deve conhecer bem o software que faz, que finalidade deve fornecer ao usuário, qual a importância no seu trabalho; deve também conhecer quem é o usuário, qual o seu nível de escolaridade, qual deve ser a sua assiduidade ao uso.

2.3.3. Recursos Utilizados no Projeto de Interfaces

O interesse do usuário ao usar o software deve ser mantido, mostrando os resultados de cálculos logo após a sua digitação, uma vez que o usuário não deva esperar pelo processamento e não se distraia da sua tarefa.

Para que esse interesse seja mantido, é necessária uma comunicação visual entre os dois componentes, a fim de que o usuário seja informado em que etapa se encontra a tarefa. Heckel (1993) cita que imagens podem ser incorporadas às mensagens para que as informações passadas de forma mais atrativa ao usuário.

Ao se conhecer o usuário, o projetista de software pode usar o vocabulário comum do cliente, nas mensagens para se comunicar com ele, o que torna mais fácil o uso do software, pois segundo cita Heckel (1993), "não podemos projetar uma boa aplicação se não entendermos o dialeto dos usuários, e mais importante, o significado e conhecimento que fundamentam esse dialeto. O uso de uma expressão que faça parte do dialeto do programador e que não seja familiar ao usuário deve ser evitado".

O recurso do uso de metáforas também deve ser aproveitado, pois mantém a atenção do usuário, segundo Heckel (1993), mantém o interesse e se dirige ao público usando vários níveis de sofisticação. As metáforas tornam mais fácil a comunicação entre o usuário e o seu programa. Muitas das melhores interfaces com o usuário são aquelas que escolhem metáforas familiares para se comunicar.

O software a ser projetado deve dar ao usuário toda a funcionalidade necessária para calcular os ângulos de atrito procurados. Segundo Heckel (1993), o projetista deve comunicar ao usuário o que o sistema faz, se isso acontecer o programa parecerá simples, os detalhes da interface com o usuário podem ser elaborados de modo a reforçar o modelo que está sendo comunicado, de modo a criar uma ilusão visual na mente do usuário. É necessário que o usuário possa escolher a quantidade de vezes que a pressão foi aplicada ao produto. Também é obrigado a fornecer opções para a escolha da célula que foi usada no teste.

A interface do software deverá ser elaborada com cuidado, pois sendo uma ferramenta essencial para projetos de estruturas de armazenamento, deve possuir fatores

como boa facilidade de aprendizado, rápida velocidade de resposta ao usuário, baixa taxa de erros, retenção com o tempo e satisfação subjetiva. Como citado por Zambalde e Alves (2004), "é de fundamental importância entender que qualquer interface quando bem projetada deve desaparecer, ou seja, não representar problemas para os usuários, fazendo com que os mesmos se concentrem exclusivamente em suas atividades".

2.3.4. Conceitos Iniciais de Orientação a Objetos

A linguagem *Java* é baseada no paradigma orientado por objetos. Camarão e Figueiredo (2003) explicam que, em linguagens orientadas a objetos, cada entidade que compõe um sistema é representada por um objeto, com atributos e comportamento definidos. O uso de uma linguagem orientada a objeto facilita a análise, modelagem e implementação de um software, ou seja, a engenharia do software, pois os componentes do sistema podem ser modelados em objetos.

Os principais conceitos do paradigma orientado a objetos, segundo Rezende (2002) são definidos a seguir: o objeto é uma coisa real ou abstrata, sobre a qual se armazenam dados e operações que manipulam os dados. As operações são usadas para ler ou manipular os dados de um objeto. Os métodos são a maneira como as operações são codificadas no software, a classe é um tipo de objeto e, possui estrutura e métodos, que especificam as operações que podem ser feitas com aquela estrutura.

Os princípios básicos da orientação a objetos são os seguintes: a classificação, a encapsulação, a herança e o polimorfismo. Segundo Rezende (2002), a classificação é o agrupamento de objetos com os mesmos atributos e as mesmas operações em uma classe; a encapsulação oculta dos usuários detalhes de funcionamento interno da classe, ou seja, os usuários sabem o que a classe pode fazer, mas não sabem como ela faz; a herança é o processo de criar novos tipos de dados baseados nos existentes, ou seja, com a herança pode especificar novos tipos de dados, com os atributos e métodos herdados da classe-pai e com outros especificados na própria classe; o polimorfismo é o uso do mesmo nome para métodos de classes diferentes, sobrepondo implementações destes em classes generalizadas.

2.3.5. O Uso de Linguagem Orientada a Objetos para Desenvolvimento de Software

Para Rezende (2002), o uso de linguagem que implementa o paradigma orientado a objetos torna os programas mais adaptáveis a mudanças. Segundo Rumbaugh (1997) citado em Rezende (2002), as classes podem ser testadas antes de serem usadas, tornando o sistema mais confiável; as classes podem ser reutilizadas em outros sistemas.

Para Silveira e Schmitz (1999), a orientação a objetos, a partir da década de 90 se afirmou como o estilo de projeto mais utilizado no desenvolvimento de software, pois representam melhor o mundo real devido ao mapeamento que existe entre os objetos do mundo real e os objetos do mundo computacional.

2.3.6. Fundamentos de Desenvolvimento Baseado em Engenharia de Software

Segundo Pressman (1995), a definição de engenharia de software, proposta por Fritz Bauer, é a seguinte: "é o estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione em máquinas reais". Segundo Martin e Mcclure (1991), citado em Rezende (2002), engenharia de software é o estudo dos princípios e sua aplicação no desenvolvimento e manutenção de sistemas de software.

Um software com qualidade é aquele que é capaz de atender as necessidades dos usuários, segundo Booch et al. (2000). Se ele for desenvolvido de maneira previsível, em um período determinado, com a utilização eficiente e eficaz dos recursos, ele será um bom negócio ao desenvolvedor. Assim a modelagem está entre as atividades que levam à implantação de um bom software.

2.3.7. A Importância da Modelagem de Software

Os métodos para construir um software, segundo Rezende (2002), envolvem tarefas como planejamento e estimativa de projeto, análise de requisitos, projeto de estruturas de

dados, arquitetura de programa e de algoritmo de processamento, codificação, teste e manutenção. Para Althoff et al. (2002), ao fim da modelagem do sistema, ele está formalmente especificado e, assim, pode-se usar ferramentas para realizar a validação e a verificação reduzindo o tempo gasto na etapa de teste do sistema.

Para o desenvolvimento de um software, sempre é necessário que se tenha um projeto de desenvolvimento claramente modelado. Como "a UML é uma linguagem-padrão para a elaboração da estrutura de projetos de software" de Booch et al. (2000). A UML pode ser usada para se compreender como o software deve ser construído e utilizado pelos usuários, para descrever abstratamente seu desenvolvimento e a seqüência de ações a serem executadas. Para Rezende (2002), "todo e qualquer projeto deve ser planejado por meio de metodologias estruturadas, modernas e que forneçam documentação completa".

Para todo projeto de software, a modelagem pode ter diferentes visões da arquitetura do software, pois em cada foco uma característica diferente pode ser visualizada. Se todas as visões possíveis forem modeladas, o software terá seu comportamento descrito completamente. Para Booch et al. (2000), os diagramas desenvolvidos em UML podem ser conectados às linguagens de programação para a geração de código.

2.3.8. Modelagem de Software usando *Unified Modeling Language* (UML – Linguagem de Modelagem Unificada)

Segundo Vasconcelos (2002), citado por Cunha (2004), o desenvolvimento de software deixou de ser artesanal. Pois, há poucos anos, a produção do software era feita sem padronização das ações, sem a preparação para um posterior aumento na funcionalidade, sem a realização de testes de detecção de erros. Hoje em dia, há um enfoque estruturado e metódico que prevê a realização de todas as tarefas de desenvolvimento necessárias.

Para a especificação e realização dessas tarefas foi criada a linguagem UML em 1995, definindo padrões de engenharia de software que devem ser usados para a modelagem.

2.3.9. Os Principais Componentes da UML

Os objetivos principais da modelagem são, segundo Booch et al. (2000), os seguintes: os modelos ajudam a visualizar como é o sistema ou como se deseja que seja, permitem especificar a estrutura ou o comportamento do sistema, proporcionam um guia para a construção do sistema e documentam as decisões tomadas.

A UML, para Booch et al. (2000), abrange três blocos de construção para a modelagem de software, eles são: os itens, os diagramas e os relacionamentos. Os itens são divididos em estruturais, comportamentais, de agrupamentos e anotacionais. Serão descritos a seguir os itens que deverão ser usados na modelagem do software.

2.3.9.1. Os Itens da Modelagem

Os principais itens estruturais são as classes e os casos de uso. As classes descrevem objetos e possuem atributos, operações, relacionamentos e semântica e podem implementar interfaces.

Os casos de uso são descrições de conjuntos de seqüências de ações realizadas pelo sistema, proporcionando resultados observáveis, segundo Paula (2003), um caso de uso é a especificação de uma seqüência de ações, que um sistema ou produto pode executar quando interage com os seus usuários ou com sistemas externos.

Os itens comportamentais, segundo Booch et al. (2000), são estruturados pelos casos de uso, representam procedimentos no espaço / tempo e o principal deles é interação. Esse item comportamental abrange trocas de mensagens, seqüências de ações e ligações entre objetos para a realização de propósitos específicos.

Os itens de agrupamento organizam os modelos e o principal deles é o pacote. Segundo Booch et al. (2000), os pacotes podem agrupar quaisquer itens, mas pacotes são

apenas conceituais, pois sua única função é decompor o modelo para facilitar a visualização.

Um item anotacional é apenas uma parte explicativa do sistema, é incluído apenas para explicar um elemento do modelo. Existe apenas um item anotacional, a nota, ela aprimora o diagrama com comentários que são mais bem explicados em texto do que em elementos da UML.

2.3.9.2. Os Relacionamentos da Modelagem

Os relacionamentos são divididos em: dependência, associação, generalização e realização.

A dependência, segundo Booch et al. (2000), é uma relação entre dois elementos de modelagem em que a alteração do elemento de modelagem independente pode afetar a semântica do elemento de modelagem dependente. A associação é um relacionamento estrutural que descreve conexões entre elemento de modelagem. A generalização é uma herança, sendo que o elemento de modelagem filho compartilha a estrutura e o comportamento do elemento de modelagem pai. A realização garante que um elemento de modelagem vai executar o método conforme especificado por outro elemento de modelagem.

2.3.9.3. Os Diagramas

Os diagramas permitem a visualização do sistema sob diferentes perspectivas. Os diagrama de classes, de casos de uso e de seqüência serão explicados a seguir.

Segundo Booch et al. (2000): o diagrama de classes exibe um conjunto de classes, interfaces e seus relacionamentos, por isso é encontrado freqüentemente na modelagem de sistemas orientados a objetos. Segundo Silveira e Schmitz (1999), o diagrama de classes é estático, pois a estrutura descrita por ele é sempre válida para qualquer instante do desenvolvimento do projeto. Para Rumbaugh et al. (1999), o diagrama de classes é um conceito importante da aplicação, pois esse diagrama ilustra todos os atributos e comportamentos do sistema através de objetos.

Objetos devem ser distribuídos em pacotes, a fim de serem organizados para que o diagrama seja mais bem visualizado. As classes, principais elemento de modelagem mostrados no diagrama de classe, têm ligações entre si através dos relacionamentos, ou seja, as classes possuem dependências, associações, realizações ou generalizações, para definir o tipo de acesso a outras classes.

O diagrama de casos de uso exibe um conjunto de casos de uso e atores, e seus relacionamentos. Segundo Althoff et al. (2002), o diagrama de casos de uso deve ser capaz de mostrar o que o sistema vai fazer e não como fazer. Esse tipo de diagrama é muito utilizado quando o sistema está sendo especificado e a iteração com o cliente, neste ponto, deve ser intensa.

O diagrama de seqüência segundo Booch et al. (2000), modela aspectos dinâmicos do sistema, dando ênfase à ordenação temporal de mensagens trocadas entre objetos que compõem o sistema.

Segundo Rumbaugh et al. (1999) o diagrama de seqüência exibe uma interação em um quadro bidimensional. A dimensão vertical é o eixo do tempo, é ordenado de cima para baixo. A dimensão horizontal mostra os papéis do classificador que representam objetos individuais na colaboração. Cada papel do classificador é representado por uma coluna vertical chamada de linha de vida. Durante o tempo que o objeto existe, o papel é mostrado por uma linha seccionada. Durante o tempo que um procedimento no objeto está ativo a linha de vida é mostrada contínua e dupla. Uma mensagem é mostrada como uma seta da linha de vida de um objeto para a linha de vida de outro objeto. As setas são organizadas sucessivamente no tempo diagrama abaixo.

3.METODOLOGIA

Neste capítulo serão descritos os procedimentos utilizados no desenvolvimento do software, incluindo uma análise dos sistemas existentes, a construção dos diagramas e as ferramentas usadas durante todo o desenvolvimento.

3.1. Levantamento dos requisitos do sistema

A pesquisa iniciou-se com a análise do uso do aplicativo do ambiente MS-DOS *YLocus* pelos usuários, o qual é método atual de cálculo das propriedades dos produtos armazenados. A partir desse momento, foram levantadas questões quanto aos problemas e o que deveria ser melhorado no aplicativo desenvolvido. Foram obtidas informações sobre novas funções que deveriam estar disponíveis no novo software.

O processo de análise começou pela tela principal, onde foram verificados os tipos de entradas de dados, de gráficos e outras opções. Seguiu-se então, analisando quais dados são pedidos e como são pedidos ao usuário, de todas os tipos de entrada. Também foram verificadas as seguintes funções no programa:

- A impressão dos gráficos na tela;
- O cálculo dos resultados pelo programa;
- A impressão dos resultados em papel;
- A gravação e a abertura de arquivos pelo usuário;
- A verificação da validade dos dados digitados pelo usuário.

Essa análise detalhada é anotada para consulta, durante o planejamento e desenvolvimento do software.

3.2. Procedimentos Metodológicos

Para o desenvolvimento foi empregado o modelo de ciclo de vida de processo de software chamado desenvolvimento evolucionário. Neste modelo os resultados parciais são mostrados rapidamente ao usuário, o qual faz comentários sobre os requisitos a serem atendidos nas próximas versões e as falhas a serem corrigidas. Este modelo foi o mais

apropriado ao método de trabalho empregado por se esperar funcionalidades prontas mais rapidamente o possível.

A partir da análise dos requisitos do software, foi iniciada a modelagem do sistema, utilizando UML, os diagramas construídos foram o diagrama de casos de uso, o diagrama de classes e diagrama de sequência.

A modelagem é apenas a primeira parte do desenvolvimento de software, a implementação é segunda parte e a terceira é a fase de testes. Baseando-se nas classes obtidas do diagrama de classes iniciou-se a implementação do sistema em linguagem *Java*, usando fórmulas vindas da bibliografia sobre as propriedades físicas dos produtos, código-fonte do aplicativo existente foram codificadas as equações de modo a executarem os cálculos necessários. A terceira fase iniciou-se com testes de falhas, verificação da eficácia da aplicação e correção de fórmulas para se alcançar os resultados esperados. Os testes foram realizados por meio de comparação dos resultados obtidos com os resultados encontrados nos métodos existentes e assim as equações foram corrigidas até alcançarem os valores mais precisos que se pode obter.

3.3. Ferramentas Utilizadas

A modelagem do sistema foi realizada com a ferramenta *ArgoUML*[2], versão 0.16.1. Foi escolhida por ser uma ferramenta que pode ser obtida gratuitamente na Internet, ser totalmente compatível com a sintaxe da UML. A partir da modelagem realizada com a ferramenta, foram geradas as classes principais do sistema.

Foi utilizado o ambiente de desenvolvimento *Eclipse*[3], versão 2.1.3. Nessa ferramenta, foi realizada a edição do código do sistema na linguagem Java. A máquina virtual Java usada para a criação dos arquivos compilados foi a JDK versão 1.4.1_01, pois era a versão mais recente durante a implementação. Para o desenho dos gráficos na tela foi utilizada a biblioteca *JFreeChart*[4], versão 0.9.19, que se mostrou simples de usar e bastante versátil para o desenvolvimento de aplicações.

2 Veja: http://argouml.tigris.org
3 (c) Copyright IBM Corp. and others 2000, 2003. Veja: http://www.eclipse.org
4(c) Copyright 2000 – 2004 Object Refinery Limited and Contributors Veja: http://www.jfree.org

4. RESULTADOS E DISCUSSÕES

Neste capítulo, serão apresentados os resultados provenientes da execução do presente projeto, tal como a modelagem do sistema, a explicação do funcionamento do sistema e serão mostradas as telas do "CPFP (Cálculo de Propriedades Físicas de Produtos)", o software desenvolvido, com exemplos reais do programa sendo executado.

4.1. Especificação do Software

A necessidade de um método eficiente de cálculo matemático das propriedades físicas dos produtos agrícolas levou ao desenvolvimento de um software. As necessidades do com os requisitos listados a seguir:

- Fornecer um formulário para de entrada de yield locus de modo que possa preencher os dados com até seis yield loci, com cinco pré-shear cada, onde é digitada a célula usada no ensaio, o peso aplicado e a força obtida;
- Fornecer um formulário para entrada de wall yield locus, onde são digitados o tipo de parede e a célula usada no ensaio e as leituras obtidas na máquina de Jenike;
- Fornecer uma planilha para entrada de cisalhamento direto, onde são digitadas as tensões vertical, horizontal e da mola.
- Fornecer os resultados do processamento das entradas de dados correspondentes através de gráficos e de relatórios mostrando os valores das variáveis calculadas.

4.2. Análise dos Problemas do Software Antigo e as Soluções Implementadas

Os métodos existentes de cálculo das propriedades baseiam-se no uso de sistemas de modo MS-DOS. Os quais possuem muitos problemas, entre estes, problemas de interface e interação com o usuário e resultados obtidos não tão precisos quanto deveriam.

A análise do software existente foi o primeiro passo para a modelagem. Na análise foram encontrados problemas, principalmente relacionados ao ambiente de execução, os problemas são apresentados a seguir:

- Dificuldade de navegação pelos menus;
- Não se pode ver ao mesmo tempo mais de uma tela de entrada de dados ou de gráfico;
- Os erros causados por digitação incorreta de dados não são informados corretamente e se percebidos pelo usuário a correção é difícil de ser realizada;
- A interação entre o software com o usuário não satisfaz suas necessidades, ele não é informado sobre erros ocorridos, sobre quais os arquivos carregados na memória.

Para a resolução dos dois primeiros problemas, optou-se pela criação de um sistema com uma janela principal, no qual contém barras de menus e botões, onde são acessados todos os comandos, e uma área de trabalho dentro da janela, onde podem ser colocadas múltiplas janelas internas. Nesta área de trabalho, as janelas podem ser arrastadas, redimensionadas e sobrepostas conforme a pretensão do usuário.

Para a resolução dos dois últimos problemas, optou-se por uma interação onde o usuário é informado, no momento da digitação, da entrada de um caractere inválido, onde ele pode digitar o correto; toda janela interna aberta tem no título o tipo de janela a que pertence e o nome do arquivo de onde estão os dados mostrados. Deste modo, o sistema sempre está informando ao usuário sobre que parte ele se encontra na entrada dos dados. Conforme citado na referência bibliográfica, esta é uma parte importante para qualquer interface de software.

4.3. Modelagem do Diagrama de Casos de Uso

A partir da mesma análise, foi construído o diagrama de casos de uso, o qual é mostrado na Figura A.1 no Apêndice com as descrições dos casos de uso no Apêndice B, que tem a função de indicar o comportamento do sistema de uma forma abstrata, pois

apenas exibe o que o software deve oferecer de funcionalidade e não de que modo vai fazê-la. O único ator do sistema é o usuário, o qual executa os principais casos de uso, como criar as entradas de dados e desenhar gráfico.

Os casos de uso são representados pelas elipses com o nome dentro dela. As associações são as linhas simples que ligam o usuário aos casos de uso. As inclusões são as linhas tracejadas, com a palavra "*include*" entre ângulos duplos. As linhas simples com uma seta na ponta são as generalizações.

Os principais casos de uso são abrir arquivo e criar entrada de dados, os quais se especializam nos casos de uso criar *yield locus*, criar *wall yield locus* e criar cisalhamento direto. O caso de uso criar entrada de dados inclui outros casos de uso, como salvar os dados em arquivo, calcular as variáveis usadas no gráfico, mostrar o arquivo de tais variáveis, desenhar o gráfico na tela. Também são incluídos no diagrama, os casos de uso para editar os tipos de células e tipos de paredes disponíveis e de mostrar a ajuda do sistema.

Estes são os casos de uso de maior importância para esta modelagem, pois descrevem completamente o que o software deve oferecer ao usuário. Este diagrama é útil a todas as pessoas relacionadas ao sistema, pois a partir deste, os usuários podem entender melhor o que está sendo feito; o projetista também comunica ao usuário de uma maneira mais formal o que entendeu. Dessa maneira, eles podem discutir melhorias nas especificações e chegar a um consenso sobre a funcionalidade oferecida.

Usando generalizações, foi possível o uso de um menor número de associações e inclusões simplificando o diagrama. Também optou por fazer o mínimo possível de sobreposição de linhas e casos de uso, proporcionando maior legibilidade dos diagramas.

4.4. Modelagem do Diagrama de Classes

A partir do diagrama de casos de uso, dos requisitos especificados para o software, pôde-se então modelar o diagrama de classes, o qual descreve a estrutura do sistema, mostra os objetos a serem criados e seus relacionamentos.

O diagrama de classes está ilustrado na Figura A.2 no Apêndice. Nele, estão expostos apenas classes, relacionamentos, atributos e métodos essenciais ao entendimento da modelagem do sistema. Os atributos possuem o nome seguido pelo seu tipo; os métodos possuem o nome seguido por parênteses, que podem ter os parâmetros entre eles, e ao fim o tipo de retorno do método. Todos os métodos e atributos de classes com o sinal "+", são acessíveis a todas as classes, sempre através de objetos instâncias das classes, pois não há métodos ou atributos estáticos, ou seja, possíveis de serem acessados sem a criação de instâncias da classe. Os métodos e atributos com o sinal "-" são privados, acessíveis somente dentro da própria classe.

Os relacionamentos mostrados neste diagrama são: as associações unidirecionais, representadas pelas setas com ponta aberta; as generalizações, representadas pelas setas com ponta fechada; e as agregações por composição, representadas pelas linhas com diamante em uma ponta.

O ponto central deste diagrama é a classe abstrata *NovaEntradaDeDados*, a qual possui somente métodos abstratos que são ser implementados por todas as suas classes especializadas. Esta classe possui associações unidirecionais para as classes *CalculosYlocus* e *GraficosYlocus*, as quais são acessíveis às classes que a generalizam.

A classe *GraficoYlocus* possui todos os métodos necessários para a realização de todos os casos de uso relacionados aos gráficos. A classe *CalculosYlocus* possui os métodos necessários para os cálculos dos ângulos de atrito a partir dos dados fornecidos de acordo com o tipo de entrada.

As classes *NovoYieldLocus*, *NovoWallYieldLocus* e *NovoCisalhamentoDireto* especializam a classe *NovaEntradaDeDados*, possuem seu próprio construtor e implementam todos os seus métodos abstratos, cada uma de seu próprio modo.

A classe *NovoYieldLocus* possui as classes *YieldLoci* e *PreShear* através da agregação por composição, com vetores de objetos de tais classes. A classe *NovoYieldLocus* possui um vetor com, no máximo, seis objetos da classe *YieldLoci*, cada qual possui um vetor de, no máximo, cinco objetos da classe *PreShear*. O tamanho de cada vetor é definido pelo usuário durante a entrada de dados nos campos que possuem os mesmos nomes dessas classes. A armazenagem em tempo de execução dos dados

digitados pelo usuário nestas classes é feita do mesmo modo na classe com nome correspondente e com o prefixo "Dados", pois, deste modo, facilita o acesso aos dados quando for necessário para calcular as variáveis do gráfico.

Para as classes *NovoWallYieldLocus* e *NovoCisalhamentoDireto*, o armazenamento dos dados digitados pelo usuário é feito somente através de vetores. Porém, na primeira classe através de um vetor unidimensional e na segunda classe através de um vetor bidimensional.

Neste diagrama, também aparecem os estereótipos *create* e *type*, mostrados como palavras entre ângulos duplos. O primeiro estereótipo informa que o método é um construtor para a classe que o contém, o segundo informa que a classe especificada é como um tipo de dado mais complexo, neste diagrama a classe armazena valores em atributos.

4.5. Modelagem do diagrama de seqüência

O diagrama de seqüência foi modelado de acordo com as classes existentes no diagrama de classes. Com os objetos instâncias destas classes trocando mensagens entre si foi modelada a seqüência de interações que descrevem o comportamento de objetos. A modelagem foi dividida em três diagramas correspondentes a cada entrada de dados.

Para os objetos relacionados à entrada de dados yield locus o diagrama construído está ilustrado na Figura A.3 no Apêndice. A Figura A.4 do Apêndice ilustra o mesmo diagrama para os objetos relacionados à entrada de dados wall yield locus e o diagrama da Figura A.5 para os objetos relacionados a entrada de dados cisalhamento direto.

4.6. A Implementação do Software

A implementação do software foi onde se gastou mais tempo durante o desenvolvimento, pois foram encontradas as maiores dificuldades, tais como, a elaboração de uma interface de software simples e funcional; a estrutura de armazenamento em memória dos dados digitados para acesso quando forem calculadas as variáveis usadas no gráfico; a obtenção das equações certas para que se encontrem os resultados corretos dos

ângulos de atrito e dos desenhos nos gráficos; a maneira certa de desenhar gráficos na tela de acordo com as referências bibliográficas. Tais problemas foram resolvidos com consultas às bibliografias e com pesquisas em fóruns sobre programação na Internet.

4.7. Descrição do Funcionamento do Sistema

A tela principal do CPFP está ilustrada na Figura 4.1, onde se pode ver uma simples janela com uma barra de menus e uma barra de botões. Na área de trabalho desta janela, podem estar abertas, ao mesmo tempo, qualquer quantidade de janelas internas e de quaisquer tipos. Ao se acionar um comando qualquer na barra de menus ou botões, como, por exemplo, desenhar gráfico, salvar em arquivo ou abrir arquivo com as variáveis calculadas, é importante que a janela interna, a qual contém os dados em que se deseja executar a ação, esteja selecionada.

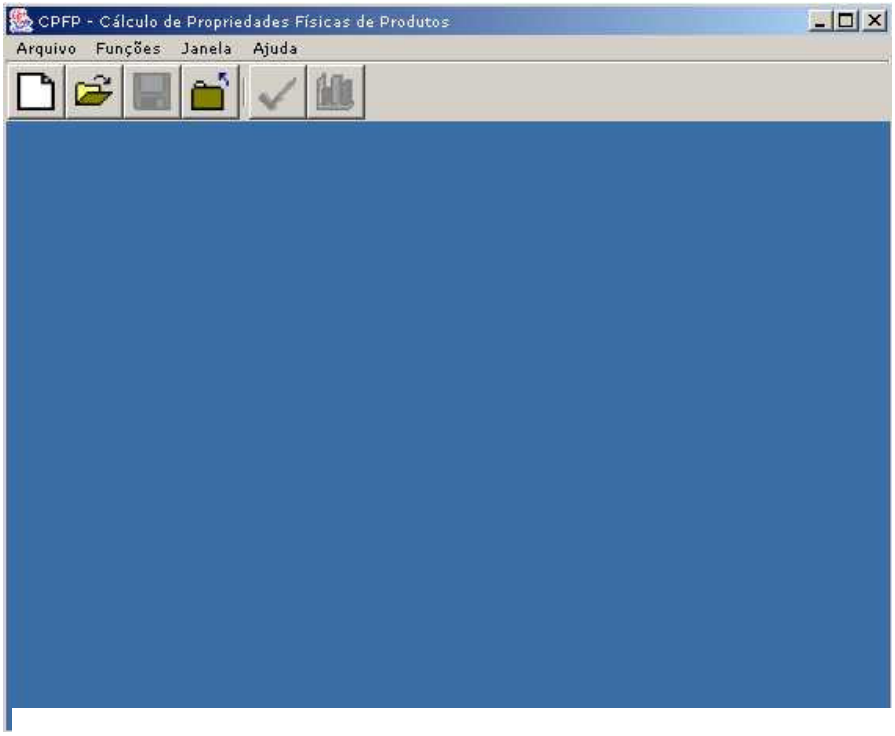


Figura 4.1 – A janela principal do software CPFP – Cálculo de Propriedades Físicas de Produtos

O primeiro passo para o uso do software é acionar o comando para abrir um arquivo do disco rígido ou para digitar uma nova entrada de dados de escolha do usuário, a seguir deve-se escolher o tipo de entrada de dados os quais podem ser de *Yield Locus*, de *Wall Yield Locus*, ou de Cisalhamento Direto. Quando é escolhido abrir arquivo, aparece a tela padrão do sistema operacional para salvar ou abrir arquivo, a qual é ilustrada na Figura 4.2. Depois de escolhido o tipo e o arquivo, os dados aparecem na janela como se tivessem sido digitados.

Em todos os gráficos que são ilustrados a seguir, podem-se aplicar ferramentas para aproximar ou distanciar o gráfico conforme conveniência para melhor visualização, ou ainda para focalizar determinada área de acordo com as coordenadas ilustradas nos eixos à esquerda e abaixo.

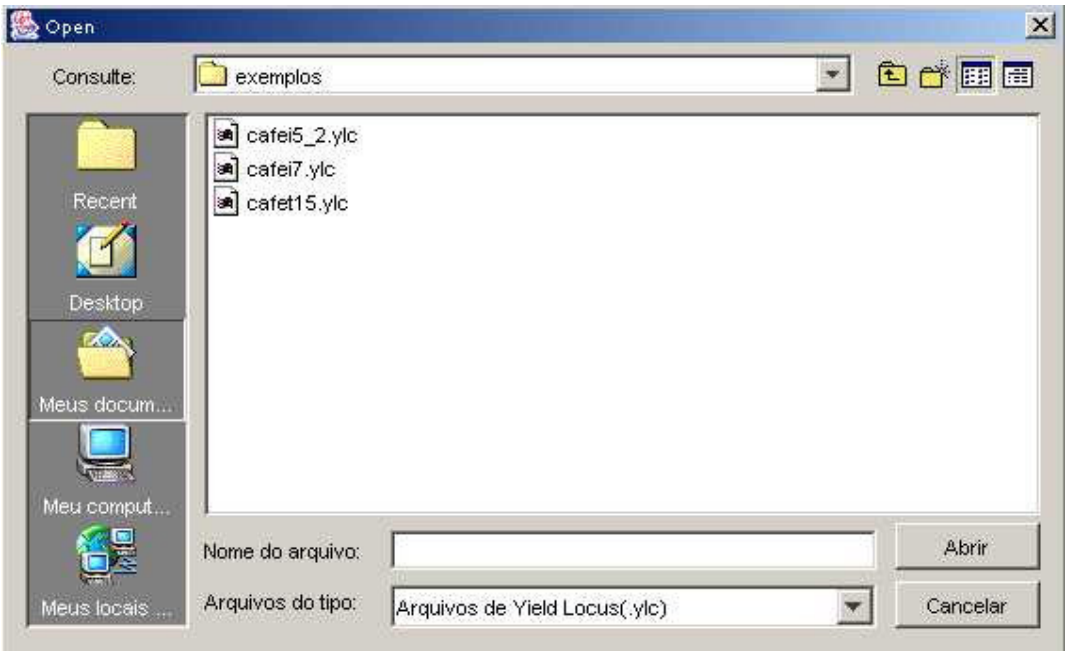


Figura 4.2 – A janela de padrão do sistema operacional para abertura de arquivos usada no software

4.8. A entrada de Dados de *Yield Locus*

A digitação de uma entrada de dados de *Yield Locus* é a de maior esforço, pois dependendo do ensaio realizado com o produto na máquina, é necessária a entrada de até seis *Yield Loci*, cada um com até cinco *Pré-Shear*, o qual deve-se fornecer o peso e a força total, o peso e a força de shear e a célula da máquina no ensaio.

Por exemplo, para um ensaio realizado com três *Yield Loci* e cada um com três *Pré-Shear*, foi obtida a tela da Figura 4.3, com todos os dados digitados. É importante considerar que, quando o ensaio possui mais que *Yield Loci*, o sistema mostra um a cada vez enquanto o usuário preenche os dados. Pois, se fosse o contrário causaria confusão ao usuário e até poderia não ter espaço na tela para se expor todos de uma vez.

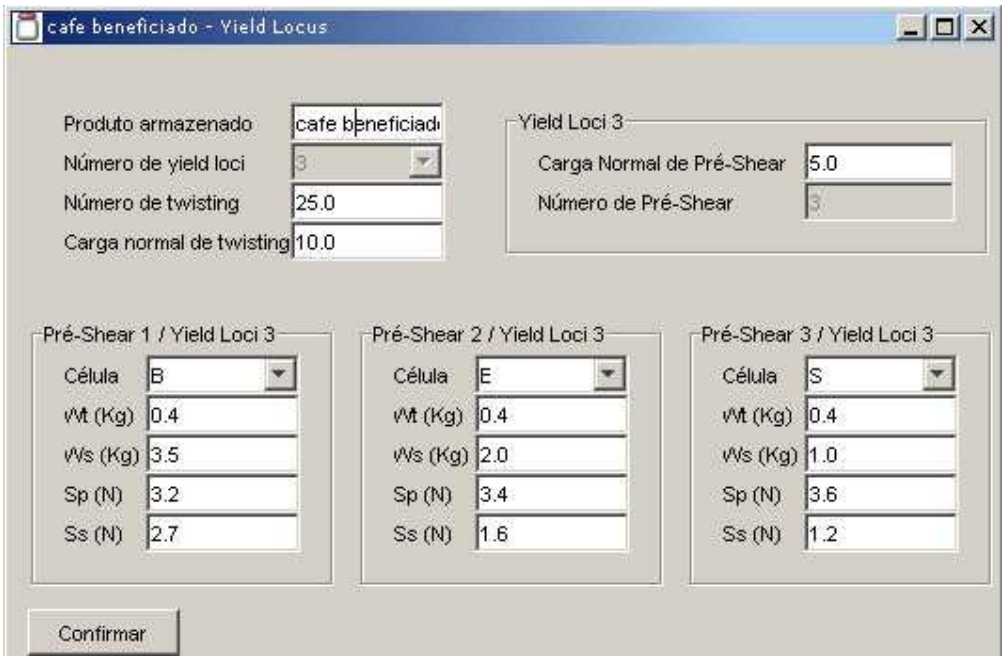


Figura 4.3 – A entrada de dados de *Yield Locus* com um exemplo de entrada de dados obtidos dos testes na máquina de Jenike.

Como este é um tipo de entrada com quantidade de dados grande e muito variável, podem-se fazer quatro tipos de gráficos diferentes, os quais são ilustrados a seguir.

O primeiro gráfico é ilustrado na Figura 4.4, exibindo o ângulo interno e externo, do lado esquerdo. No gráfico, exibem-se as retas obtidas de tais ângulos, o primeiro e o segundo semicírculos de *Mohr* – o menor e o maior, respectivamente, ilustrados no

gráfico, os três pontos da amostra – pequenos pontos circulares, e o ponto de consolidação – pontos retangulares. Todos estes na mesma cor, mas para cada *Yield Loci* de uma cor diferente.

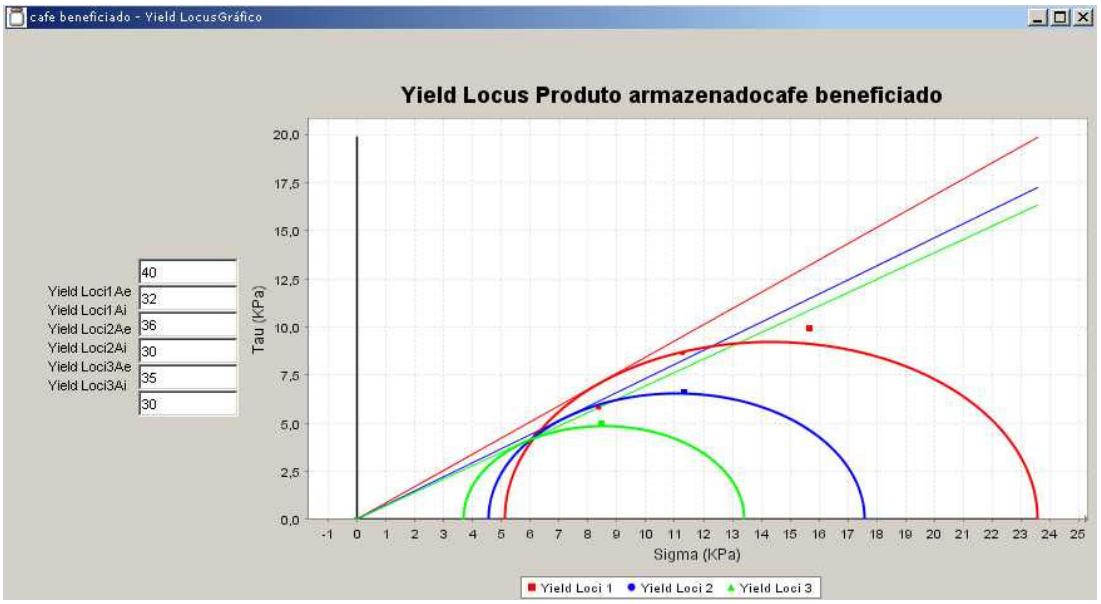


Figura 4.4 – O gráfico principal produzido pela entrada dos dados mostrada na Figura 4.4.

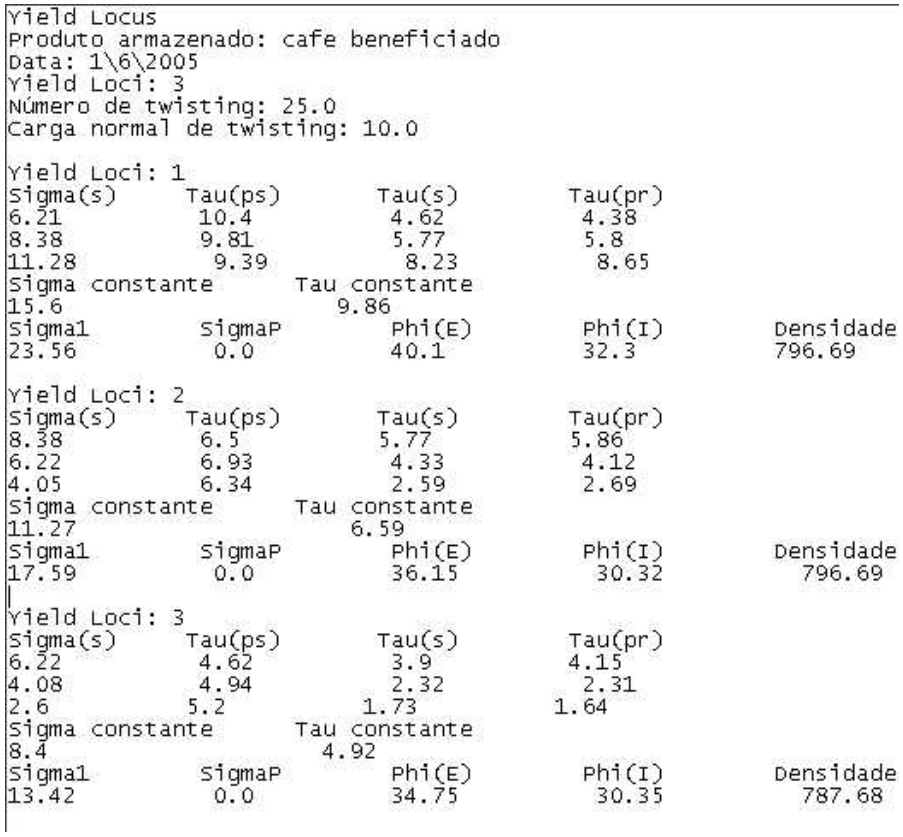
O gráfico de densidade obtido é ilustrado na Figura A.6 no Apêndice, a densidade da amostra é ilustrada à esquerda do gráfico, os círculos são os pontos de densidade e a reta liga os centros destes círculos.

A Figura A.7 no Apêndice ilustra o gráfico Delta 1 por VM e Delta 2 por VM, os deltas obtidos são ilustrados à esquerda do gráfico, os pontos de Delta 1 por VM são aqueles por onde se passa a reta vermelha pelo seu centro e os pontos de Delta 2 por VM são aqueles por onde se passa a reta azul em seu centro.

A Figura A.8 no Apêndice ilustra o último gráfico de *Yield Locus*, o gráfico de FC por VM. Ele exibe à direita, na expressão (FC = ca * VM + (cl)), os valores calculados das variáveis "ca" (coeficiente angular) e "cl" (coeficiente linear) que formam a reta mostrada.

Para a entrada de dados da Figura 4.3, ainda foi obtido o arquivo com as variáveis calculadas ilustrado na Figura 4.5. Esta figura mostra nas seis primeiras linhas, os dados principais sobre a entrada de dados. Nas linhas seguintes, mostra para cada *Yield Loci* os

valores encontrados das variáveis usadas para traçar todos os gráficos citados anteriormente.



```
Yield Locus
Produto armazenado: cafe beneficiado
Data: 1\6\2005
Yield Loci: 3
Número de twisting: 25.0
Carga normal de twisting: 10.0

Yield Loci: 1
Sigma(s)     Tau(ps)       Tau(s)        Tau(pr)
6.21          10.4          4.62          4.38
8.38          9.81          5.77          5.8
11.28          9.39          8.23          8.65
Sigma constante      Tau constante
15.6                    9.86
Sigma1        SigmaP         Phi(E)        Phi(I)        Densidade
23.56          0.0           40.1          32.3          796.69

Yield Loci: 2
Sigma(s)     Tau(ps)       Tau(s)        Tau(pr)
8.38          6.5           5.77          5.86
6.22          6.93          4.33          4.12
4.05          6.34          2.59          2.69
Sigma constante      Tau constante
11.27                   6.59
Sigma1        SigmaP         Phi(E)        Phi(I)        Densidade
17.59          0.0           36.15          30.32          796.69

Yield Loci: 3
Sigma(s)     Tau(ps)       Tau(s)        Tau(pr)
6.22          4.62          3.9           4.15
4.08          4.94          2.32          2.31
2.6           5.2          1.73          1.64
Sigma constante      Tau constante
8.4                     4.92
Sigma1        SigmaP         Phi(E)        Phi(I)        Densidade
13.42          0.0           34.75          30.35          787.68
```

Figura 4.5 – As variáveis calculadas para o *Yield Locus* da Figura 4.4.

4.9. A Entrada de Dados de *Wall Yield Locus*

A entrada de dados de *Wall Yield Locus* é a mais fácil, pois necessita de apenas seis leituras da máquina de cisalhamento para cada tipo de parede em que se realizou o ensaio. A conseqüência clara de ser necessária a digitação de poucos dados neste tipo de entrada é a de que é gerado apenas um tipo de gráfico.

A Figura 4.6 ilustra a entrada para uma parede de silo de aço rugoso, a célula usada no ensaio e os valores das leituras obtidas na máquina. A Figura A.9 do anexo ilustra o

gráfico obtido da entrada de tais dados, com o ângulo de atrito mostrado à esquerda e a reta com os pontos da leitura à direita.

Figura 4.6 – Um exemplo de entrada do tipo *Wall Yield Locus*.

Nesta janela podem-se fazer quantas entradas de dados for necessário, sendo ou não de tipos de paredes diferentes. Para cada janela de entrada é mostrado o gráfico em uma nova janela, tais janelas podem-se organizadas na área de trabalho da janela principal para efeito de comparação entre dados e gráficos.

A partir da entrada de dados ilustrada na Figura 4.6, obteve-se o arquivo com as variáveis calculadas ilustrado na Figura A.10 do Apêndice. Este arquivo mostra os dados principais sobre a entrada nas três primeiras linhas e nas linhas seguintes os valores dos pesos usados nas leituras, as forças obtidas nas leituras e as variáveis que usadas para o desenho do gráfico da Figura A.9.

4.10. A Entrada de Dados de Cisalhamento Direto

A entrada de dados de cisalhamento direto é uma simples tabela, onde o usuário digita a leitura obtida na máquina dos extensômetros vertical, horizontal e da mola, em cada linha e no máximo em 24 linhas. Essa tabela pode ser copiada para um arquivo para impressão. Quando são digitados dados em uma linha, o sistema mostra as variáveis ε(%), $\Delta V(cm^3)$, Fh (kgf) e τ (kgf/cm^2) calculadas instantaneamente. A Figura 4.7 ilustra um exemplo dessa tabela, com os dados preenchidos de um ensaio realizado na máquina de cisalhamento.

A Figura A.11 do Apêndice ilustra o gráfico obtido do ensaio acima, onde o eixo horizontal dá o valor da deformação do produto e o eixo vertical dá o valor da tensão cisalhante.

Vertical (lv)	Horizontal (lh)	Mola (lm)	(%)	(cm^3)	(kgf)	(kgf / cm^2)
0.9	0.21	1.01	0.42	0.0	0.86200000000000...	0.03448000000000...
0.9	0.22	1.01	0.44	0.0	0.86199915	0.03447997
0.9	0.71	1.01	1.42	0.0	0.86199915	0.03447997
0.9	1.34	1.02	2.68	0.0	1.7239983	0.06895994
0.9	2.11	1.03	4.22	0.0	2.5859976	0.103439905
0.9	2.66	1.04	5.3199997	0.0	3.4479966	0.13791987
0.9	3.48	1.04	6.96	0.0	3.4479966	0.13791987
0.9	4.19	1.03	8.38	0.0	2.5859976	0.103439905
0.9	4.31	1.04	8.62	0.0	3.4479966	0.13791987
0.9	5.23	1.04	10.46	0.0	3.4479966	0.13791987
0.9	6.07	1.05	12.139999	0.0	4.3099957	0.17239983
0.9	6.74	1.05	13.479999	0.0	4.3099957	0.17239983
0.9	7.29	1.04	14.58	0.0	3.4479966	0.13791987
0.9	7.94	1.05	15.880001	0.0	4.3099957	0.17239983
0.9	8.68	1.04	17.36	0.0	3.4479966	0.13791987
0.9	9.14	1.05	18.28	0.0	4.3099957	0.17239983
0.9	9.64	1.06	19.28	0.0	5.171995	0.20687981
0.9	10.09	1.06	20.180002	0.0	5.171995	0.20687981
0.9	10.64	1.07	21.279999	0.0	6.0340047	0.24136019
0.9	11.24	1.07	22.48	0.0	6.0340047	0.24136019
0.9	11.81	1.06	23.62	0.0	5.171995	0.20687981
0.9			24.06	0.0	6.0340047	0.24136019

Tabela de dados1

Figura 4.7 – Exemplo de entrada de dados de Cisalhamento Direto

4.11. A Adição de Novos Tipos de Células e Paredes às Equações do Software

A adição de novos tipos de células de cisalhamento e novos tipos de paredes de silos é uma função inovadora neste tipo de software. Pois, antes do desenvolvimento deste trabalho não havia como adicionar novos coeficientes às equações de cálculo das propriedades.

Com isto, o software se tornou mais flexível aos diferentes tipos de pesquisas em que ele é aplicado. Por exemplo, se uma nova célula com características diferentes for inventada para um trabalho específico, suas características poderão ser adicionadas como opção de escolha para o cálculo das propriedades. O mesmo pode acontecer com o desenvolvimento de um novo tipo de parede.

Tal adição de coeficientes é feita de modo simples, quando se aciona tal opção na interface do software, é aberto arquivo que contém todos os coeficientes usados atualmente.

A Figura 4.8 ilustra o arquivo de tipos de células disponíveis somente com as células comuns. Para adicionar novas células, é necessário colocar uma nova linha no fim do arquivo escrever o nome e as características da célula seguindo o formato padrão do arquivo. Depois de adicionada a célula, é indispensável reiniciar o software para que esta célula esteja pronta para seleção.

#celula	wb	Vb	Wr	A	Vr	Wl
S	0.1242	0.0000862	0.0384	0.00679	0.0001093	0.0895
A	0.1175	0.0000815	0.0396	0.00679	0.0001086	0.0898
B	0.1154	0.0000818	0.0399	0.00679	0.0001087	0.0904
C	0.1153	0.0000811	0.0403	0.00680	0.0001088	0.0880
D	0.1153	0.0000841	0.0400	0.00679	0.0001086	0.0895
E	0.1181	0.0000833	0.0403	0.00674	0.0001079	0.0893
F	0.1167	0.0000815	0.0396	0.00679	0.0001086	0.0902
S140	0.2363	0.0002365	0.0810	0.01355	0.0003076	0.1313

Figura 4.8 – Exemplo da visualização dos tipos de células disponíveis para as entradas de dados de *Yield Locus* e *Wall Yield Locus*

A Figura 4.9 ilustra o arquivo de tipos de paredes disponíveis somente com paredes comuns. A adição de um novo tipo de parede é feita de modo análogo à adição de tipo de célula, é necessário colocar uma nova linha no fim do arquivo, escrever o nome da parede do silo e o seu coeficiente.

```
#tipo de parede  coeficiente
Aço rugoso        0.8223
Aço liso          0.818
Madeira           0.0605
Plástico          0.11493
Concreto          0.201
Alumínio          0.2831
```

Figura 4.9 – Exemplo de visualização dos tipos de paredes disponíveis para a entrada de dados *Wall Yield Locus*

5. CONCLUSÕES

Os projetos de silos para o armazenamento de produtos agrícolas necessitam do conhecimento de propriedades físicas dos produtos que se desejam armazenar.

A obtenção destas propriedades atualmente é feita de diversos modos, entre estes os principais são os ensaios nas máquinas de cisalhamento e de *Jenike*. Destes ensaios, são obtidas as forças aplicadas aos grãos, estas forças são as entradas para as fórmulas que calculam as propriedades dos produtos.

Neste trabalho objetivou-se desenvolver um aplicativo que resolvesse diversos problemas dos usuários na resolução das equações para encontrar tais propriedades e ainda atendeu às necessidades de pesquisadores e projetistas de silos na questão da melhoria do tratamento dos dados obtidos.

O novo sistema substitui completamente os existentes com adição de novas funções, atinge os objetivos propostos de melhorias nas interações com o usuário através do uso de uma interface amigável e principalmente de componentes gráficos que diminuem o esforço do usuário na digitação dos dados.

Os objetivos específicos como fornecer telas de entradas de dados, apresentar todos os resultados numéricos e gráficos foram alcançados com resultados satisfatórios e esperados. Assim sendo, este é um sistema eficaz de resolução das equações de obtenção das propriedades.

O software desenvolvido auxilia na determinação de propriedades físicas de produtos armazenados, propriedades estas indispensáveis para o cálculo de pressões e para a determinação do fluxo do produto dentro do silo. A determinação destas propriedades é realizada em projetos de graduação e pós-graduação, que contribuirão para a elaboração da norma brasileira de silos, em laboratórios de engenharia agrícola de diversas universidades e em empresas privadas que projetam estruturas de armazenamento.

O software poderá ser incrementado com novas funções em trabalhos futuros. Funções como o cálculo de propriedades que não estão disponíveis, como a implementação de entradas de dados para outras máquinas de cisalhamento. Poderão ser corrigidas falhas ainda não encontradas na interface e nas equações de cálculo de propriedades.

6. REFERÊNCIAS BIBLIOGRÁFICAS

ALTHOFF, G. F.; RAPOSO, E. P. STEMMER, M. R., Modelagem de sistemas através da UML – Exemplo prático. **CTAI - Revista de Automação e Tecnologia da Informação**, vol. 01, nro. 02, julho / 2002.

AMERICAN CONCRETE INSTITUTE (ACI), **Recommended Practice for Design and Construction of Concrete Bins, Silos and Bunkers for Storing Granular Materials.** P. 313-377. EUA, 1983.

BONSIEPE, G., **Design: do material ao digital.** Florianópolis: SEBRAE/SC, tradução de Cláudio Dutra, 1997.

BOOCH, G.; RUMBAUGH, J.; JACOBSON I., **UML, guia do usuário.** Rio de Janeiro: Campus, 2000.

BOUMANS, G., **Grain handling and storage.** Elsevier Science Publishers B.V., 1985.

BRITTON, M. G.; MOYSEY, E. B., **Grain properties in the proposed new Engineering practice on bin loads.** St. Joseph. American Society of Agricultural Engineers. Paper ASAE, n.86 – 4502, 1986.

BUCKLIN, R. A. et al., **Apparent coefficient of friction of wheat on bin wall material.** Trans. ASAE, v.32, n.5, p. 1769-73, 1989.

CALIL Jr. C., **Sobrepressiones en las paredes de los silos para almacenamiento de produtos pulverulentos cohesivos.** São Carlos. Escola de Engenharia de São Carlos USP, 184 p, 1984.

CALIL Jr., C., **Recomendações de Fluxo e de Cargas para o Projeto de Silos Verticais.** EESC - USP. 198 p, 1990.

CAMARÃO, C.; FIGUEIREDO L., **Programação de computadores em Java.** Rio de Janeiro: LTC Editora, 2003.

CHUNG, J. H.; VERMA, L. R., **Determination of friction coefficients of beans and peanuts.** Trans. ASAE, v.32, n.2, p. 745-50, 1989.

CUNHA, L. H. R. T., **Modelagem e desenvolvimento de um software para a diretoria de recursos humanos (DRH) da Universidade Federal de Lavras/MG – UFLA.** 2004.

Monografia (Bacharelado em Ciência da Computação) – Universidade Federal de Lavras, Lavras-MG.

DEUTSCHE INDUSTRIE NORMEM (DIN-1052), **Holzbauwerke Berechnung und Ausfuhrung.** Deustscheland**,** 1969.

FERREIRA, A. B. H., **O novo dicionário Aurélio - Século XX**, 1999. Cd-rom.

GAYLORD, E. D.; GAYLORD, C. N., **Design of steel bins for storage of bulk solids.** New Jersey: Prentice-Hall, 1984.

HAAKER, G., **Measurement of wall friction and wear in Bulk solids handling**. In: Silos – Forschung und Praxis, Karlsruhe, TAGUNG'88 in karlsruhe. Kalrlsruhe, Universitat Karlsruhe. P. 389-403, 1988.

HECKEL, P., **Software Amigável, técnicas de projeto de software para uma melhor interface com o usuário**. Rio de Janeiro: Campus 1993.

JANSSEN, H. A., **Experiments on grain pressure in silos.** Verein Dentscher, p 1045-1049, 1895.

JENIKE, A. W., **Storage and Flow of Solids.** Salt Lake City. University of Utah. (Bulletin. Utah Engineering Experiment Station, n.123), 1964.

LATINCSICS, N. K., **Silos and Storage Vessels: influence of compressibility of the stored solids on the design methods, the process and conflicts in existing codes.** Bulk solids Handling, v5, n 1, p. 219-24, 1985.

MANZANO, A. L. N. G.; MANZANO, M. I. N. G., **Estudo dirigido de informática básica.** São Paulo: Érica, 1998.

MARTIN, J. McCLURE, C., **Técnicas estruturadas e Case.** São Paulo: Makron Books , 1991.

MILANI, A. P., **Determinação das propriedades de produtos armazenados para o projeto de pressões e fluxo em silos.** 1993. Dissertação (Doutorado em Engenharia de Estruturas) – Universidade de São Paulo, São Carlos.

MOHSENIN, N. N., **Phisical properties of plant and animal materials.** New York: Gordon and Breach Science, 1986.

PAULA FILHO, W. de P., **Engenharia de Software: Fundamentos, métodos e padrões.** Rio de Janeiro: LTC Editora, 2003.

PRESSMAN, R. S., **Engenharia de software.** São Paulo: Makron Books, 1995.

REZENDE, D. A., **Engenharia de software e sistemas de informação.** Segunda edição. Rio de Janeiro: Brasport, 2002.

RUMBAUGH, J.; JACOBSON I.; BOOCH, G., **The Unified Modeling Language – Reference Manual**. Reading, Massachusetts: Addison Wesley Longman, Inc., 1999.

SCHWEDES, J., **Measurement of Flow properties of Bulk Solids.** In: Proceedings International Symposium of Powder Technology 81 p.89 – 98, 1986.

SCHWEDES, J., **Influence of wall friction on silo design in process and structural engineering.** Ger. Chem. Emg. (8) p. 131, 1985.

SILVA, F. S., **Propriedades físicas dos grãos de café para o projeto tecnológico de equipamentos de silos.** 2003. Tese (Doutorado em Engenharia Agrícola) – Universidade Federal de Viçosa, Viçosa.

SILVEIRA, D. S., SCHMITZ, E. A., **Fast Case – Uma ferramenta Case para o desenvolvimento visual de sistemas orientados a objetos.** Florianópolis, XIII Simpósio Brasileiro de Engenharia de Software, 1999.

VASCONCELOS, A. M. L.; MACIEL T. M. M. **Introdução à engenharia de software e os princípios de qualidade.** Lavras: FAEPE, 2002.

ZAMBALDE, A. L.; ALVES, R. M., **Interface homem-máquina e ergonomia.** Lavras: UFLA/FAEPE, 2004.

ZHANG, Q.; PURI, V. M.; MAMBECK, H. B., **Model for frictional behavior of wheat on structural materials.** Trans.of the ASAE, v.31, n.3, p.898-903, 1988.

APÊNDICE A

Neste apêndice são ilustrados os diagramas de casos de uso, de classes, e de seqüência construídos para o software, e algumas telas do software em execução e gráficos gerados.

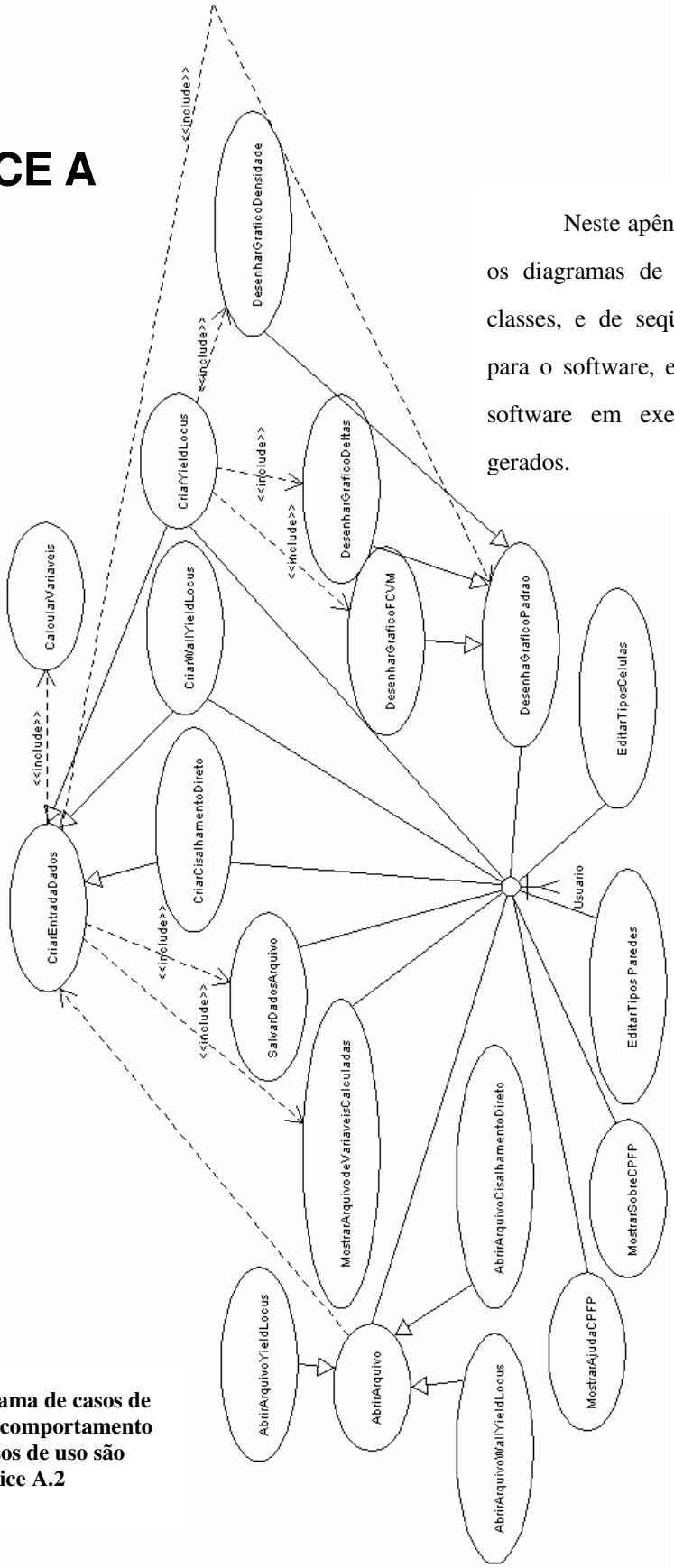


Figura A.1 – Diagrama de casos de uso que descreve o comportamento do software. Os casos de uso são descritos no Apêndice A.2

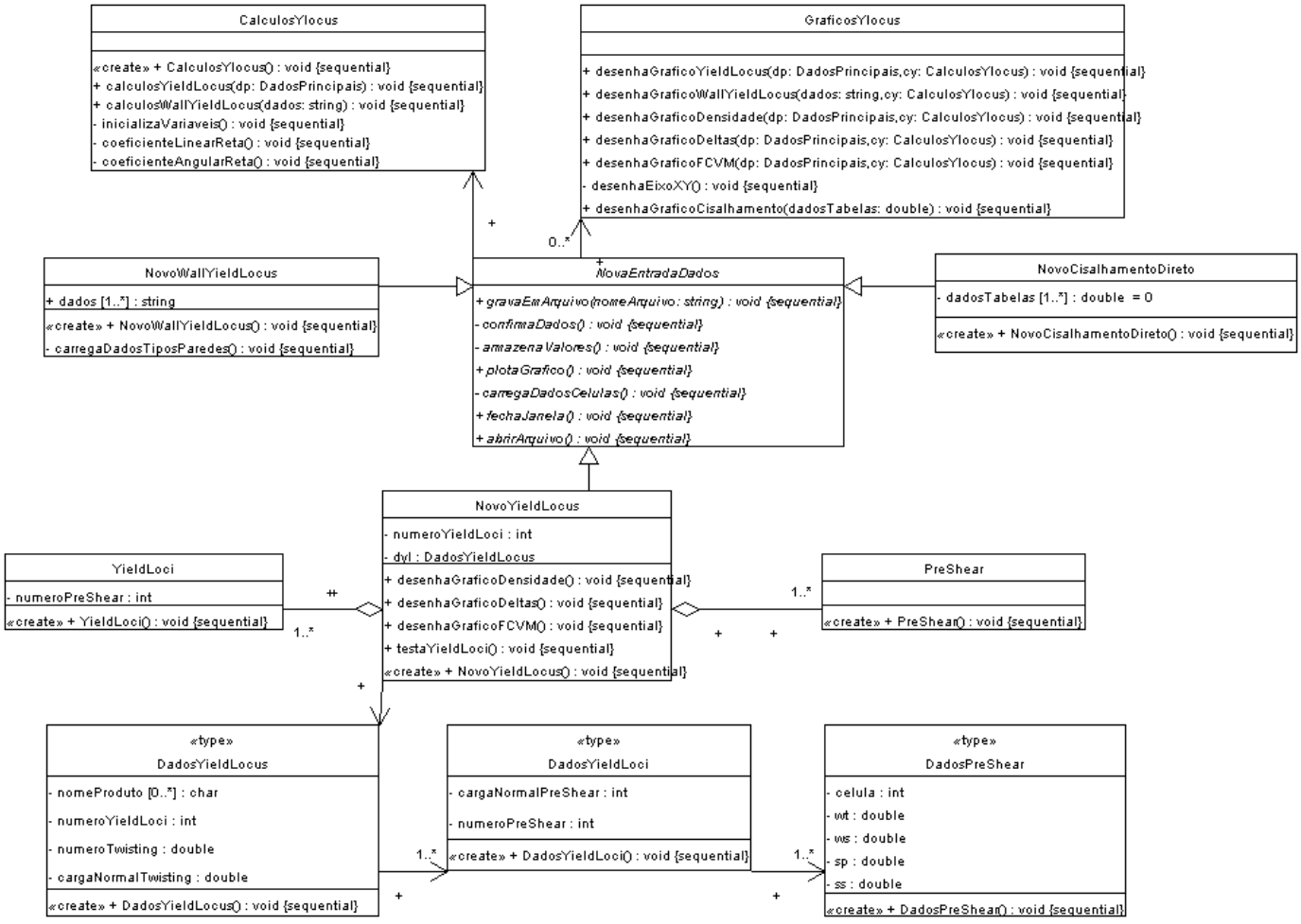


Figura A.2 – Diagrama de classes modelado para o software

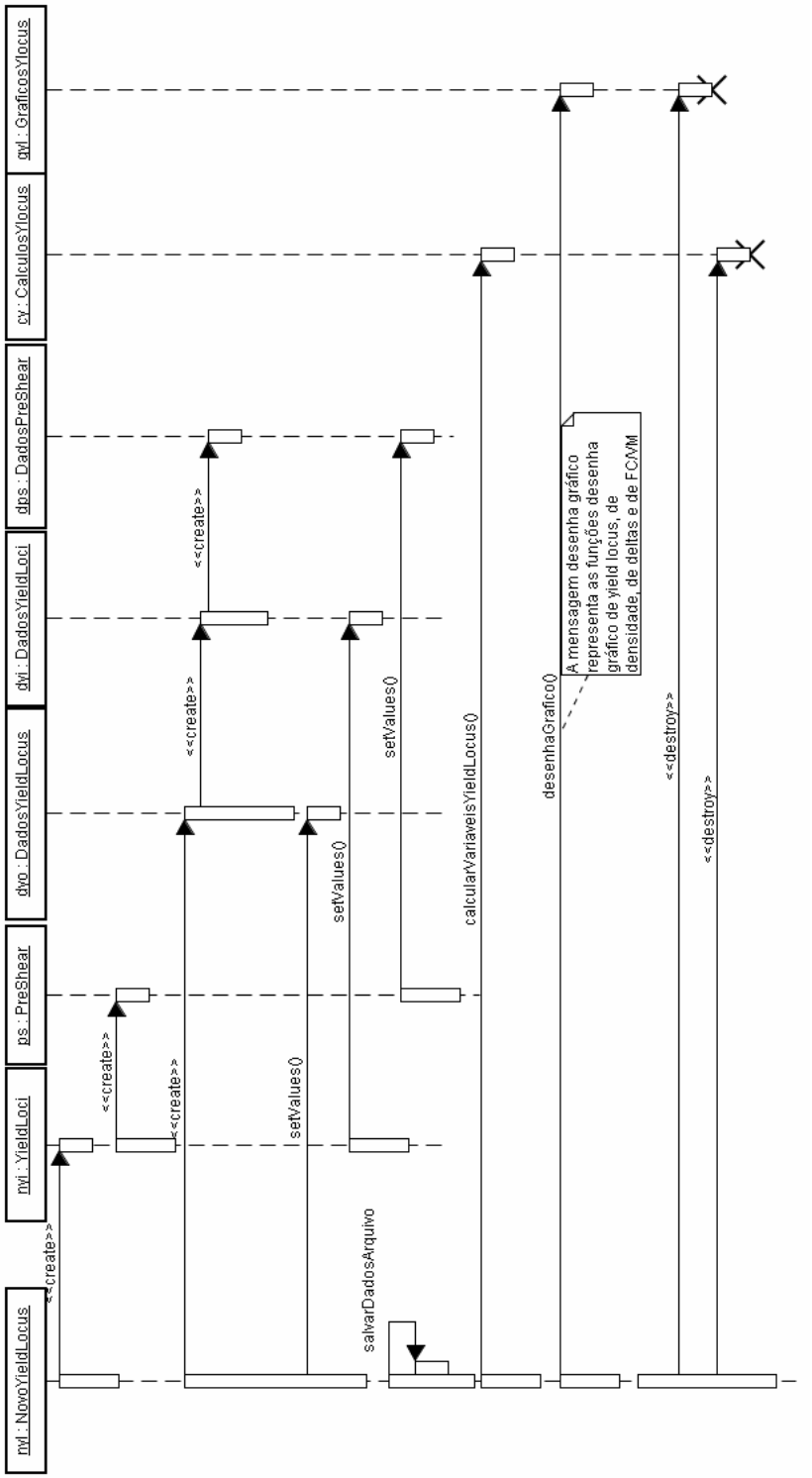


Figura A.3 – Diagrama de seqüência ilustrando a troca de mensagens entre as classes relacionadas à entrada de dados *yield locus*

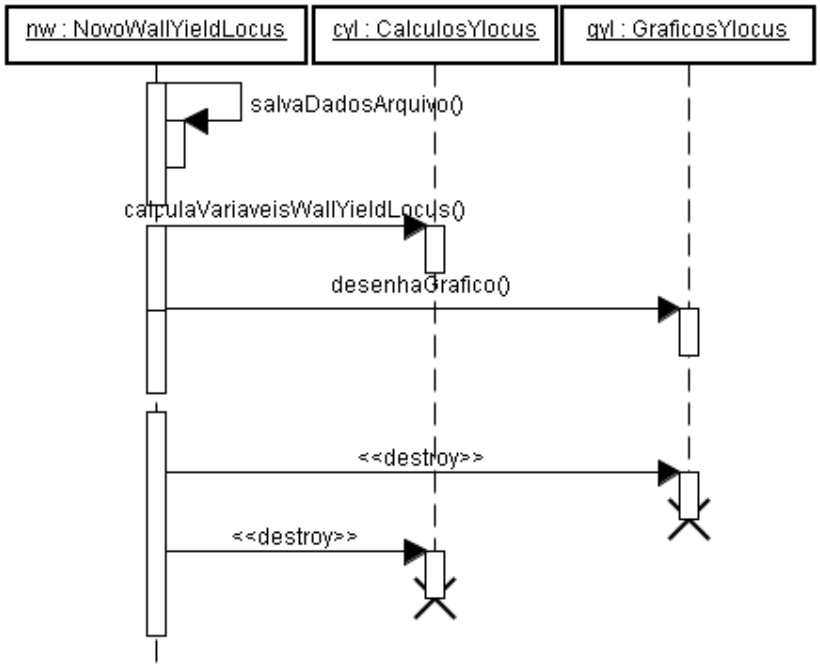


Figura A.4 - Diagrama de seqüência ilustrando a troca de mensagens entre classes relacionadas à entrada de dados *wall yield locus*

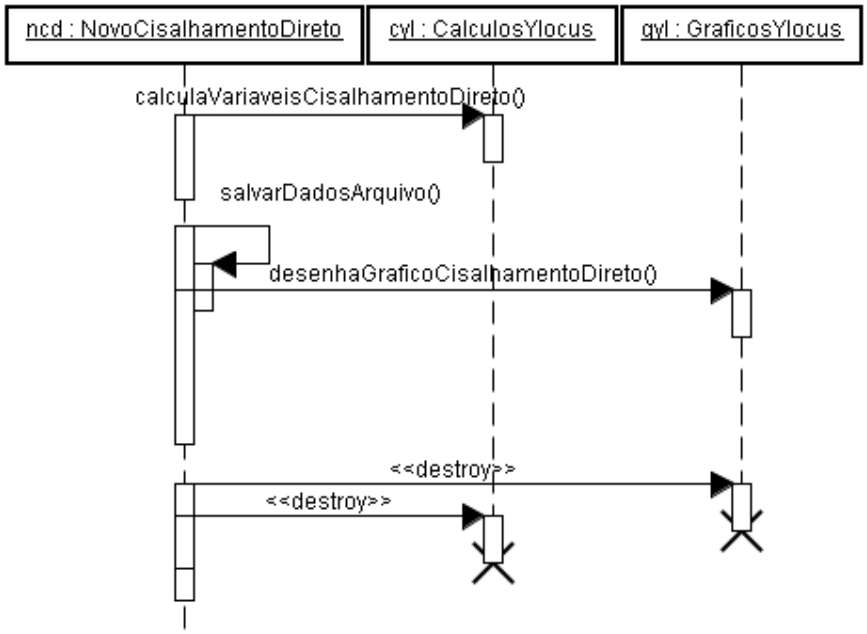


Figura A.5 – Diagrama de seqüência ilustrando a troca de mensagens entre as classes relacionadas à entrada de dados cisalhamento direto

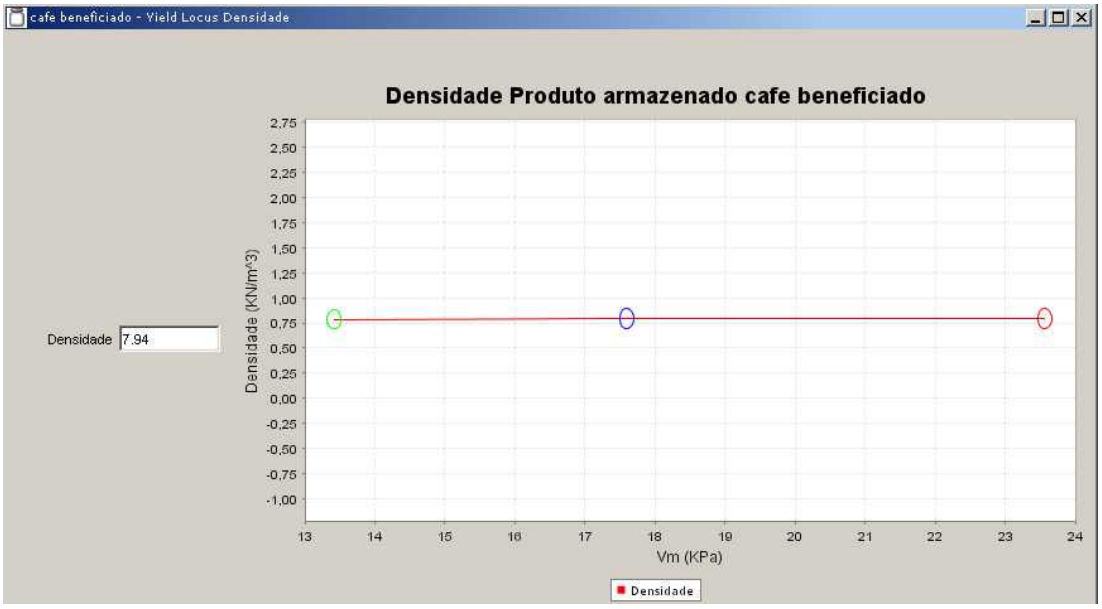


Figura A.6 – Gráfico de densidade obtido para a entrada de dados ilustrada na Figura 4.3.

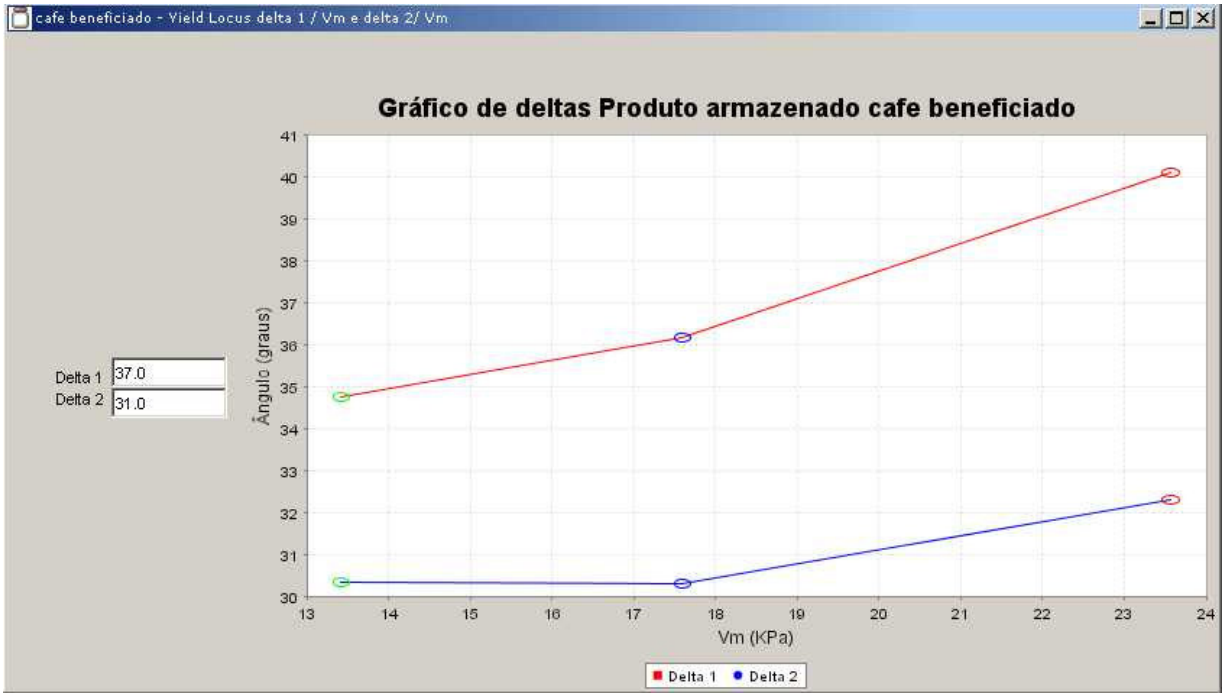


Figura A.7 – Gráfico de Delta 1 por VM e Delta 2 por VM para a entrada de dados da Figura 4.3.

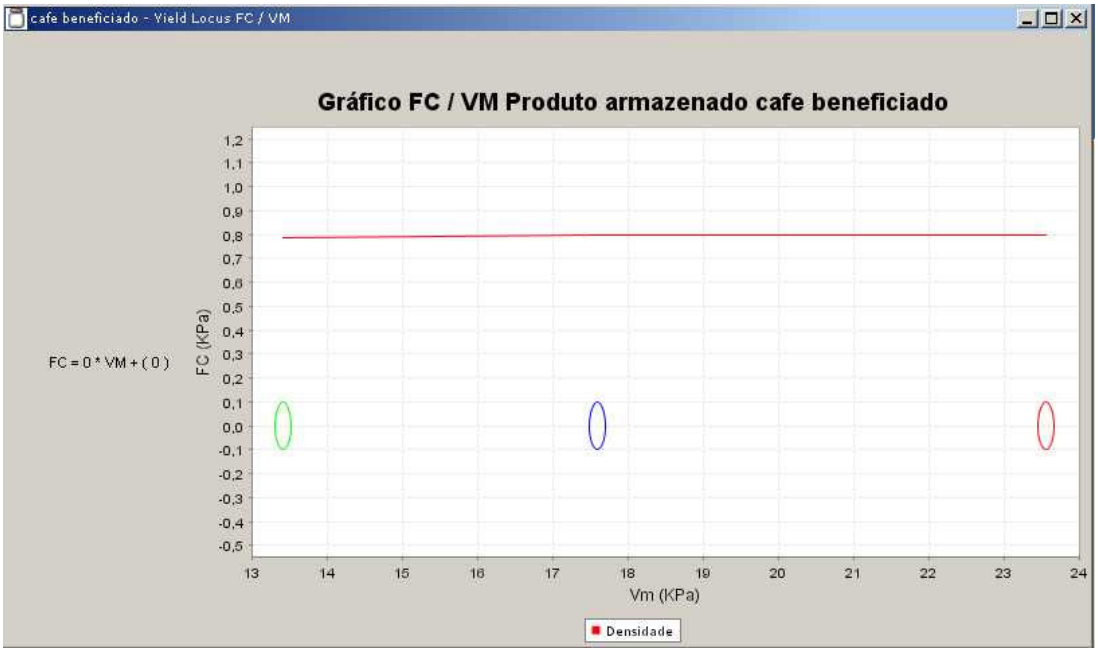


Figura A.8 – Gráfico FC / VM obtido para a entrada de dados ilustrada na Figura 4.3.

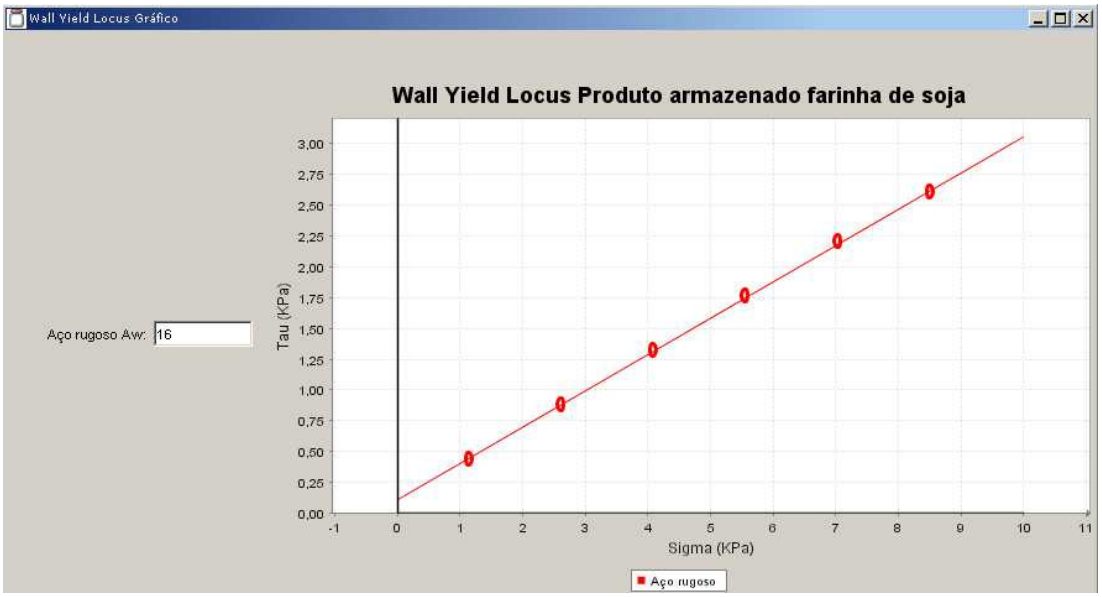


Figura A.9 – O gráfico de *Wall Yield Locus* obtido do processamento da entrada ilustrada na Figura 4.6

Wall Yield Locus
Produto armazenado: farinha de soja
Data: 1\5\2005

Teste	Wtm(KgF)	Ww(KgF)	Sw(KgF)	Sigma(w)	Tau(w)	Phi(w)
1	1	5	1.77	8.49	2.6	45.03
2	1	4	1.5	7.02	2.2	45.03
3	1	3	1.2	5.55	1.76	45.03
4	1	2	.9	4.07	1.32	45.03
5	1	1	.6	2.6	0.88	45.03
6	1	0	.3	1.13	0.44	45.03

Figura A.10 – As variáveis obtidas com o processamento dos dados da Figura 4.6.

Figura A.11 – O gráfico de Cisalhamento Direto obtido da entrada dos dados da tabela ilustrada na Figura 4.7

APÊNDICE B

Neste apêndice é mostrada a descrição de todos os casos de uso ilustrados na Figura A.1.

Caso de uso	Descrição
CriarEntradaDados	Mostrar uma tela de entrada de dados para digitação pelo usuário
CriarYieldLocus	Mostrar uma tela de entrada de dados de *Yield Locus*
CriarWallYieldLocus	Mostrar uma tela de entrada de dados de *Wall Yield Locus*
CriarCisalhamentoDireto	Mostrar uma tela de entrada de dados de Cisalhamento Direto
CalcularVariáveis	Calcular as variáveis a serem usadas nos gráficos e mostradas na tela
SalvarDadosArquivo	Salvar em arquivo os dados digitados na entrada de dados
MostrarArquivodeVariáveisCalculadas	Mostrar o arquivo com as variáveis calculadas usadas nos gráficos
DesenharGráficoPadrão	Desenhar na tela o gráfico padrão para a entrada de dados selecionada
AbrirArquivo	Abrir um arquivo onde estão salvos dados digitados pelo usuário
AbrirArquivoYieldLocus	Abrir um arquivo onde estão salvos dados de entradas de *Yield Locus*
AbrirArquivoWallYieldLocus	Abrir um arquivo onde estão salvos dados de entradas de *Wall Yield Locus*
AbrirArquivoCisalhamentoDireto	Abrir um arquivo onde estão salvos dados de entradas de Cisalhamento Direto
MostrarAjudaCPFP	Mostrar o arquivo de ajuda do software
MostrarSobreCPFP	Mostrar informações sobre o software
EditarTiposParedes	Editar os tipos de paredes disponíveis
EditarTiposCelulas	Editar os tipos de células disponíveis
DesenharGráficoFC/VM	Desenhar na tela o gráfico FC/VM para a entrada de dados *de Yield Locus* selecionada
DesenharGráficoDeltas	Desenhar na tela o gráfico de Deltas para a entrada de dados de *Yield Locus* selecionada
DesenharGraficoDensidade	Desenhar na tela o gráfico de Densidade para a entrada de dados de *Yield Locus* selecionada