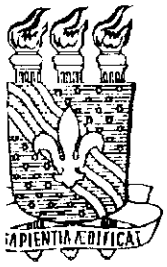


Universidade Federal da Paraíba – UFPB

Centro de Ciências e Tecnologia - CCT

Departamento de Sistemas e Computação - DSC

Coordenação de Pós-Graduação em Informática – COPIN

Mestrado em Informática

Suporte de Ferramenta de Software para o Padrão Pedagógico Aula em Mapa de Conceitos

por

JOSÉ AUGUSTO DE OLIVEIRA NETO

Prof.:Dr. Francisco Vilar Brasileiro
Orientador

Campina Grande, Fevereiro de 2000

Universidade Federal da Paraíba – UFPB
Centro de Ciências e Tecnologia - CCT
Departamento de Sistemas e Computação - DSC
Coordenação de Pós-Graduação em Informática – COPIN
Mestrado em Informática

Suporte de Ferramenta de Software para o Padrão Pedagógico Aula em Mapa de Conceitos

José Augusto de Oliveira Neto

Dissertação submetida à avaliação como
requisito parcial para a obtenção do grau de
Mestre em Ciência da Computação

Francisco Vilar Brasileiro
Orientador

Ficha Catalográfica

OLIVEIRA NETO, José Augusto de

O48S

Suporte de Ferramenta de Software para o Padrão Pedagógico Aula em Mapa de Conceitos.

Dissertação (Mestrado) – UFPB/CCT/COPIN, Campina Grande, fevereiro de 2000.

71p. Il.

Orientador: Framcisco Vilar Brasileiro

1. Informática Aplicada à Educação
2. Padrões Pedagógicos,
3. Mapa de Conceitos,
4. Ferramenta,
5. Software Educativo.

CDU: 681.3.01:37

SUPORTE DE FERRAMENTA DE SOFTWARE PARA O PADRÃO PEDAGÓGICO AULA EM MAPA DE CONCEITOS

JOSÉ AUGUSTO DE OLIVEIRA NETO

DISSERTAÇÃO APROVADA EM 25.02.2000

PROF. FRANCISCO VILAR BRASILEIRO, Ph.D
Orientador

PROF. MARCELO ALVES DE BARROS, Dr.
Examinador

PROF. EDILSON FERNEDA, Dr.
Examinador

PROF. ARTURO HERNÁNDEZ DOMÍNGUEZ, Dr.
Examinador

CAMPINA GRANDE – PB.

Agradecimentos

Aos meus pais. Não só pelo apoio permanente, incondicional e confortante. Mas, principalmente, pelas condições, exemplos e orientações que me guiaram através do longo caminho até aqui.

Aos meus irmãos (precisaria de mais um capítulo para agradecer uma a um!) por me apoiarem e formarem essa maravilhosa legião de amigos que a vida me presenteou.

Ao meu décimo irmão, Luiz Geraldo pelo exemplo, cumplicidade, amizade e sintonia. Esta última, desde o creme dental.

Aos amigos de Campina Grande, que estiveram ao meu lado nesta caminhada, destacando Gustavo, Érica, Lívia, Fellipe (LSD *team*), Sandro, André Barbosa, Cacá e Ivete.

À **mafia-l** pela companhia remota (via correio eletrônico) que me fizeram no período de trabalho em Salvador.

Aos amigos de Salvador, que foram companheiros nos momentos mais tensos, e parceiros nos mais descontraídos: Tatiana Simas, Maria Luiza, Ana Isabel e minha nova irmã/corretora ortográfica/assistente de redação, Trícia Santos. Ainda em Salvador, meu agradecimento especial a Milena Michele, pela parceria, companheirismo e fundamental apoio nesta jornada.

Aos professores, alunos e coordenação de curso da Faculdade Ruy Barbosa, pela atenção e imensa colaboração.

A toda SDS (TRE-BA) pela ajuda, compreensão e colaboração nesse período de dedicação ao Mestrado.

Aos que fazem o NATO-CG (TRE-PB) pela solidariedade nos momentos iniciais desta caminhada.

A toda COPIN pela presteza, lisura, apoio e imediatez com que sempre fui atendido e tratado no decorrer deste curso.

Ao meu orientador **Francisco Vilar Brasileiro** (em negrito, como prometido) pelas lições, pelo apoio, companheirismo e boa vontade prestados a mim durante todo o período deste trabalho.

Resumo

Padrões Pedagógicos são descrições estruturadas para formatos de aula. Catalogados através de um projeto de colaboração entre professores da área de Orientação a Objetos, estes padrões tendem a demandar maior esforço por parte de professores e alunos no desempenho das atividades que determinam. Além disso, a preparação das atividades do padrão ocupa uma fatia considerável do tempo de aula disponível.

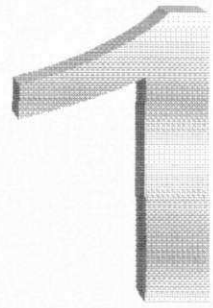
Neste trabalho, experimentamos alguns dos padrões catalogados e selecionamos um deles para receber uma ferramenta de software que o auxiliasse. *Class Concept Map* (traduzido para Aula em Mapa de Conceitos) foi o padrão selecionado. Este padrão apresenta os mesmos problemas (maior esforço e consumo do tempo de aula) detectados nos demais padrões catalogados. Para suportar o Padrão Pedagógico Aula em Mapa de Conceitos, desenvolvemos a ferramenta Mapa Fácil. Em seguida, realizamos vários experimentos com o padrão sendo auxiliado pela ferramenta desenvolvida. Destes experimentos, levantamos resultados para avaliar a eficácia da ferramenta como solução dos problemas detectados no padrão Aula em Mapas de Conceito.

Abstract

Pedagogical Patterns (PP) are structured discriptions to organize a lecture. These patterns have been catalogued by a project based on collaboration among teachers from Object Orientation community. Pedagogical Patterns tend to demmand a greater effort from teachers and stundents when attendding the tasks estabilished by the pattern. In addition, the preparation for the PP activities consumes a notable slice of the available time to the class.

In this work, we experimented some of catalogued patterns, and selected one of them to receive a software tool which supports his activities. Class Concept Map was the selected pattern. This pattern carries the same problems (greater efforts and time consuming) we found in the other catalogued PP. We developed Easy Map software tool to support Class Concept Map Pedagogical Pattern. After this, we performed several experiments with the pattern being supported by the software tool we developed. From these experiments, we extracted results to evaluate the tool as an efficient solution to the problems we had detected in the Class Concept Map pattern.

SUMÁRIO



Introdução

1.1 - Padrões pedagógicos

Em praticamente todas as áreas do conhecimento é comum nos depararmos com problemas recorrentes, que com freqüência estão nos levando a reelaborar soluções semelhantes e de mesmo efeito prático. Catalogar tais soluções é um meio eficaz de se poupar o tempo e o esforço desprendidos para refazê-las, obtendo assim um considerável ganho de produtividade.

A idéia de catalogar soluções que apresentaram bons resultados para sistemática reutilização foi inicialmente adotada com sucesso na construção civil. Nesta área, determinadas modalidades de projetos já não passam por processo detalhado de cálculos, uma vez que as quantidades de material, dimensões e até prazo de elaboração já estão devidamente

catalogados e podem, com a mais completa segurança, ser readotados. Tal idéia foi apresentada no final da década de setenta, quando o arquiteto austríaco Christopher Alexander propôs a utilização de **padrões** (*patterns*) para registrar soluções aplicadas com êxito. Alexander, em [Alex77], ainda definiu como Linguagem de Padrões *(Pattern Language)* o método de reunir soluções dos problemas menores e relacioná-las de uma forma estruturada e bem organizada. Desta maneira pode-se compor o corpo da solução para o problema mais complexo.

Considera-se imensurável a abrangência de aplicações onde os padrões podem ser utilizados, trazendo benefícios notáveis. Procedimentos cirúrgicos, linhas de argumentação do advogado para determinada natureza de processo, procedimentos de laboratório para obtenção de certo tipo de substância química, etc., são todos "experiências que podem ser registradas para posterior aplicação em contexto semelhante" [Alex77].

Um padrão é a descrição clara da solução para um problema. Certo grau de formalização no registro da solução é fundamental para permitir melhor comunicação entre aqueles que recorrerão às informações registradas. Baseado no uso de padrões, Alexander propôs em [Alex79] um novo paradigma para arquitetura, composto por três componentes básicos:

Qualidade

Consiste no conjunto de benefícios (conforto, espaço, liberdade, ar puro, etc.) necessários para propiciar uma melhor qualidade de vida às pessoas.

Portão

É o mecanismo que tornará possível atingir a Qualidade. É o conjunto de padrões que descreverá o contexto necessário para suprir as necessidades que propiciarão a qualidade desejada.

Caminho

Conjunto de técnicas aplicadas para implementar os Padrões utilizados no portão, a fim de atingir a qualidade.

A estrutura (Qualidade-Portão-Caminho) é a principal fonte da transferência das idéias de Alexander para a informática, dando origem à catalogação de soluções para pro-

blemas comuns na fase projeto de baixo nível do ciclo de elaboração de um software, através dos **Padrões de Design** (*Design Patterns*).

Após alguns anos sem evidência, os padrões foram trazidos novamente à tona pela Engenharia de Software, mais especificamente na comunidade de Orientação a Objetos (OO). De 1990 a 1992 Erich Gamma, Richard Helm, Ralph Johnson, e John Vlissides, hoje denominados "Gangue dos Quatro" (*Gang of Four*), realizaram uma coleta de soluções para contextos específicos na elaboração de software. Tais soluções já eram apresentadas num formato disciplinado, com seções bem definidas e padronizadas. O catálogo composto foi então apresentado no livro *Design Patterns: Elements of Reusable Object-Oriented Software* [Gamma95]. O grande nível de aceitação atingido por este trabalho foi responsável pela plena popularização dos padrões de design, fazendo deste assunto um dos temas atualmente mais discutidos na comunidade OO.

Um padrão de design (PD) é composto da combinação de três elementos:

Problema;

Solução;

Contexto.

A reaplicação da **solução** proposta demanda que **problema** e **contexto** sejam os mesmos, caracterizando assim a situação abordada pelo PD descrito.

É crescente a quantidade de PD registrados em livros e páginas Web. Cabe aos engenheiros de software manter uma lista precisa destes catálogos, pois, à medida que forem fluentes tanto no registro quanto na recuperação destas experiências, a concepção de soluções inéditas se tornará um fato cada vez mais raro.

A utilização de PD trouxe dois benefícios de notável importância para a Engenharia de Software e comunidade OO [Kirk98]. O primeiro foi estabelecer uma interface padronizada de comunicação entre desenvolvedores, que através dos formatos bem definidos dos PD podem trocar soluções com mais fluência. O segundo, e mais notável, foi estender o poderio de reutilização de soluções, inato da abordagem OO. Em vez de reutilizar apenas código, os PD permitem reutilização no nível de projeto e, portanto, aumentam ainda mais a produtividade da elaboração de software sob este paradigma.

Como descreveremos a seguir, os PD foram ainda a fonte para os esforços de combate a um dos mais desconfortáveis entraves para a propagação do paradigma OO [Lilly96]: a dificuldade de ensiná-lo e treinar profissionais nas tecnologias que o adotam. Este problema é uma das razões para que o paradigma encontre resistência em vários ambientes de desenvolvimento, onde o legado de ferramentas e aplicações baseadas na Decomposição Funcional [Barnet68] permanece dominante.

Comparando com a Abordagem Estruturada [Dahl72], que no momento da sua popularização encontrou um contexto bem mais adverso (sem as facilidades de comunicação da Internet, maior dificuldade para distribuição de livros e artigos, menor evidência da Informática), o paradigma OO ainda se propaga de forma mais lenta. Esta lentidão decorre do fato do paradigma OO ser sensivelmente mais complexo, demandar uma maior capacidade de abstração e abordar uma maior gama de conceitos e princípios, cuja explicação consiste numa árdua missão pedagógica, somada à demanda de atenção e maior capacitação dos aprendizes [Gehringer95].

Foi pensando nesta dificuldade de ensinar o paradigma que Susan Lilly propôs em [Lilly96] uma adaptação dos PD para o contexto do ensino - aprendizagem, a fim de catalogar experiências de treinamentos e aulas que obtiveram um nível de aproveitamento satisfatório. Os **Padrões Pedagógicos** (*Pedagogical Patterns*) são modelos de aulas aplicados em contextos e assuntos bem definidos, cujas descrições seguem um formato estruturado e padronizado. Os Padrões Pedagógicos são uma derivação dos PD, agora aplicados em outro contexto. Em vez de descrever soluções comprovadamente eficazes para problemas encontrados na elaboração do projeto de um software, os Padrões Pedagógicos descrevem soluções para problemas relativos às atividades pedagógicas (ensino – aprendizagem).

O trabalho de Lilly deu origem ao **Projeto Padrões Pedagógicos**, que partiu em busca da colaboração de educadores e instrutores do paradigma para compor um catálogo de formatos de aula (Padrões Pedagógicos). O objetivo do projeto é que este catálogo sirva de fonte de pesquisa para aqueles que enfrentarão a árdua tarefa de ensinar os conceitos OO. O projeto já encontra-se com dezenas de experiências registradas, e o catálogo de padrões apresentado em sua página (web www-lifia.info.unlp.edu.ar/ppp/) continua em expansão.

A estrutura de um padrão pedagógico (PP), descrita abaixo, apresenta flagrante semelhança com os PD. Cada item apresenta a descrição de um aspecto do PP.

Formato de um Padrão Pedagógico (PP)

- **Nome**

Apresenta o nome do PP (Missão Impossível, Aparelh de CD, EPIC, etc.)

- **Objetivo**

O que se deseja ensinar, ou evitar que o aluno confunda, ou que habilidade se deseja desenvolver, etc.

- **Idéia**

Como o PP atingirá este objetivo.

- **Motivação**

O que leva a crer que essa idéia será eficaz em atingir o objetivo.

- **Aplicabilidade**

Enumera os contextos onde este PP pode ser reaplicado.

- **Contra-indicações**

Situações onde o PP gera problemas ou não funciona adequadamente.

- **Estrutura**

É a descrição do formato da aula, dos procedimentos a serem adotados e os papéis de professor e alunos neste procedimento

- **Conseqüências**

O resultado obtido após a utilização do PP.

- **Pontos a considerar**

São dicas sobre detalhes a serem providenciados e evitados na implementação do PP.

- **Dependências Culturais**

Aspectos que tornam o PP menos ou mais recomendável para determinada cultura.

- **Recursos necessários**

Material que professor/alunos precisam dispor para colocar o PP em prática.

- **Exemplos de utilização deste PP**

Experiências onde o PP já foi adotado.

- **Padrões relacionados**

Outros padrões que apresentam alguma relação de forma, contexto, objetivo ou estrutura, com este.

Visitando-se o *site* do Projeto de Padrões Pedagógicos, pode-se observar que todo o esforço que motivou a elaboração, como já mencionado, origina-se da tentativa da comunidade OO de facilitar as atividades de ensino do paradigma. Entretanto, o formato de aula apresentado mediante a estrutura acima é utilizável em várias outras naturezas de conhecimento, cabendo ao padrão apenas a adição de mais um item na relação de **Exemplos de utilização deste PP.**

Este fato implica na promoção do estudo sobre os PP de uma atividade referente apenas aos conceitos OO, para o âmbito do ensino (e puramente ensino), aplicável em qualquer natureza de conhecimento que se deseje. Sob esta ótica, é profundamente encorajador elaborar trabalhos de pesquisa que venham a contribuir com os esforços deste projeto, aparando suas arestas e lapidando os padrões para utilização em mais ampla escala.

1.2 - Motivação

Tanto através da leitura de Padrões Pedagógicos catalogados (vide, por exemplo, padrões #8, #11, #13, #23, #27, #33, #35 e #38), como nas experiências em sala de aula, verificamos que os PP apresentam dois fatores que consistem em limitações de suas qualidades didáticas:

- *A utilização dos PP implica num maior esforço de elaboração da aula.*

  A preparação da aula, seguindo a dinâmica apresentada no PP, demanda do professor um esforço prévio de preparação de material e organização de recursos a serem utilizados.

- *A implementação da dinâmica da aula consome o tempo das explicações.*

> O professor precisa ser ágil, caso contrário passará mais tempo montando e monitorando as atividades propostas pelo PP, que transmitindo os conhecimentos daquela aula.

A disponibilização de ferramentas de software que suportem as atividades de preparação de aulas segundo determinados PP pode ser uma medida eficaz no combate ao primeiro problema acima mencionado. Tais ferramentas podem evitar resistência do professor em usar o PP, dado o trabalho de elaboração de material, uma vez que tornariam menos árdua a atividade de preparação de aulas.

Nesta mesma linha, ferramentas de software que suportem as dinâmicas de apresentação de aulas podem aliviar este encargo do professor e, por conseguinte, lhe dar mais tempo hábil para a transmissão dos conhecimentos abordados por aquela aula.

O objetivo do nosso trabalho é fornecer um suporte de ferramenta de software para um Padrão Pedagógico e, em seguida, avaliar a influência causada por esta ferramenta na aplicação do padrão.

A utilização de ferramentas de software em atividades pedagógicas não é um fato novo. Softwares do tipo CSCL – *Computer Suported Colaborative Learning* (Aprendizado Colaborativo Suportado por Computador) [Ellis91] e tutores inteligentes [Brown90] já se encontram amplamente divulgados e utilizados em várias partes do mundo. Entretanto, o projeto de Padrões Pedagógicos não menciona nenhum esforço no sentido de prover algum tipo de ferramenta de software que combata as limitações que detectamos.

No trabalho apresentado neste documento, experimentamos uma série de Padrões Pedagógicos em sala de aula, e propomos alguns procedimentos para elaboração de ferramentas para PP. Em seguida, elaboramos a ferramenta Mapa Fácil, que suporta as atividades relacionadas ao Padrão Pedagógico *Class Concept Map* (traduzido para Aula em Mapas de Conceito) utilizando estes procedimentos. Por fim, utilizamos esta ferramenta em uma nova fase de experiências em sala de aula, e investigamos a eficácia da ferramenta no combate aos problemas que havíamos detectado no uso do padrão sem a ferramenta. As dificuldades apresentas pelo PP Aula em Mapa de Conceitos têm a mesma natureza dos problemas que motivaram o nosso trabalho.

1.3 - Estrutura do documento

Cada uma das fases do trabalho que desenvolvemos será detalhada no restante deste documento em 5 capítulos assim estruturados:

No capítulo 2, propomos uma metodologia que seleciona um Padrão Pedagógico para receber uma ferramenta de software. Em seguida, descrevemos como utilizamos a metodologia proposta em experiências que elegeram o padrão Aula em Mapas de Conceitos para receber um software que suportasse suas atividades. Ainda neste capítulo, descrevemos como técnicas da Engenharia de Software foram utilizadas para levantamento dos requisitos da ferramenta que suporta o padrão selecionado.

No capítulo 3, descrevemos a dinâmica prevista pelo Aula em Mapa de Conceito, através de uma aula exemplo. No decorrer desta descrição apresentamos uma série de sugestões de adaptação para o padrão descrito. As sugestões apresentadas se originaram de práticas utilizadas em experiências bem sucedidas com o padrão Aula em Mapa de Conceitos.

O capítulo 4 trás a descrição da ferramenta Mapa Fácil. Os requisitos levantados junto aos usuários do padrão Aula em Mapa de Conceitos (professores e alunos) são enumerados e, em seguida, os quatro módulos que compõem a ferramenta (Distribuição de Aulas, Elaboração de Mapas, Composição de Mapa Geral *standalone* e Composição de Mapa Geral em rede) são detalhadamente descritos. A exemplo do capítulo 2, as funcionalidades destes módulos são apresentadas em uma aula exemplo, onde se passam todas as atividades do Padrão Pedagógico.

No capítulo 5, apresentamos uma nova fase de experiências com o PP Aula em Mapa de Conceitos. Nesta fase, o padrão é utilizado com o apoio da ferramenta Mapa Fácil, tanto na versão *standalone* como na versão em rede. Adicionalmente, descrevemos as técnicas de entrevistas que utilizamos para coletar os resultados obtidos pelo padrão nas aulas de experiência com a ferramenta. Em seguida, apresentamos a sumarização dos resultados obtidos nas entrevistas. Esta sumarização compara o desempenho do padrão nesta fase de experiências com a sua atuação na primeira etapa dos experimentos, descritos no capítulo 2, quando não foi usada nenhuma ferramenta de software para apoiá-lo. No final do capítulo, analisamos os dados levantados nas entrevistas com professores e alunos.

Por fim, o capítulo 6 apresenta as conclusões que obtivemos no trabalho desenvolvido, e sugere uma série trabalhos que podem ser desenvolvidos a partir das nossas propostas e dos nossos resultados.

Experiências com Padrões Pedagógicos em Sala de Aula

Como já mencionado no capítulo introdutório, o objetivo do nosso trabalho é adicionar o aspecto ferramenta de software às aulas com Padrões Pedagógicos (PP) e, em seguida, avaliar os efeitos gerados por esta adição. A abordagem descrita abaixo é a metodologia que definimos como adequada para chegar aos objetivos traçados. Esta metodologia consiste num conjunto de atividades que formam o caminho entre o primeiro contato com os Padrões Pedagógicos catalogados, e a especificação de uma ferramenta que suporta um dos padrões utilizados. Três fases marcaram esta etapa do trabalho:

1 - Utilização de vários PP com turmas, disciplinas, professores, alunos e cursos diferentes.

2 - Seleção de um PP, dentre os experimentados, para elaboração de uma ferramenta de software que o suporte.

3 – Realização de aulas com o padrão selecionado, seguidas de levantamento de requisitos para a ferramenta.

Parte da metodologia (passos 1 e 2) é uma proposta nossa, que se originou da necessidade de atacarmos os problemas detectados no catálogo de Padrões Pedagógicos de forma organizada e estruturada. Esta parte passou por discussões acadêmicas envolvendo professores e alunos, e estas discussões serviram para aperfeiçoar a proposta e, mesmo que de uma forma preliminar, validá-la.

A outra parte da metodologia (passo 3) está espelhada e apoiada em métodos já consagrados na bibliografia de Engenharia de Software que trata de levantamento de requisitos [Thayer98].

O objetivo da metodologia que propomos não é garantir com extrema precisão que o padrão selecionado, a partir dos dois passos iniciais, seja o mais indicado para receber suporte de um ferramenta de software. O seu objetivo é apenas tornar a seleção menos aleatória e, portanto, com maior chance de sucesso nas atividades de especificação e implementação que seguem. A elaboração de uma forma precisa de escolha, além de requerer mais estudos e experiências, não é parte das metas do nosso trabalho.

A seguir, detalharemos os passos citados acima, apresentando as peculiaridades de cada um deles.

2.1 – Utilização de Padrões Pedagógicos

Esta fase da metodologia consiste em utilizar os Padrões Pedagógicos catalogados em disciplinas, cursos, nível de alunos e professores diferentes. O seu objetivo é criar o universo mais abrangente e heterogêneo possível para a tomada de decisão que será feita nas fases seguintes do processo. Ambientes assim favorecem a qualidade do experimento, por-

que diminuem a influência de aspectos particulares às amostras nos resultados gerais obtidos [Lipschutz93] .

No nosso trabalho, este universo foi montado no decorrer das atividades letivas dos cursos de Ciências da Computação e Processamento de Dados da Faculdade Ruy Barbosa [www.frb.br].

Quanto maior acesso ao quadro de professores do ambiente onde se dão esta fase dos experimentos, maiores serão as chances de criar um contexto mais abrangente para o desempenho deste passo do processo. No nosso caso, conseguimos montar um universo satisfatório por ser bastante abrangente e diversificado. Nossas experiências se deram nas disciplinas listadas abaixo, e a diversidade foi assegurada pelo fato destas disciplinas serem de cursos, turnos, período, turmas, professores, e assuntos diferentes.

- Curso Ciências da Computação
  - Algoritmos – duas turmas.
  - Introdução a Computação – uma turma.
  - Linguagem Técnica 2 – duas turmas.
- Curso Processamento de Dados.
  - Linguagem Técnica 1 – duas turmas.
  - Sistemas Operacionais – uma turma.

O fator que determina quais disciplinas farão parte dos experimentos é a disponibilidade dos professores consultados, a menos que se consiga algum comprometimento com as autoridades acadêmicas dos cursos em que se dará o experimento. Este comprometimento demanda um processo de interação mais difícil, por isso recomendamos que a decisão de envolver a disciplina caiba a cada professor titular, de acordo com sua disponibilidade e interesse pelo o projeto desenvolvido. Este foi o procedimento utilizado nas nossas experiências, e através dele conseguimos definir o seguinte universo de atuação para a utilização dos PP em salas de aula:

5 Disciplinas

6 Professores

8 Turmas

246 Alunos (matriculados)

Depois de definido o universo, os professores envolvidos nos experimentos devem consultar o catálogo de padrões. Nesta consulta, eles identificarão quais padrões se encaixam com as suas atividades acadêmicas. Cabe a quem estiver coordenando esta fase de seleção planejar a duração do período de experiências com os padrões. O tempo deve ser longo o bastante para utilizar os padrões escolhidos, e deve estar de acordo com as demais atividades acadêmicas dos cursos como provas, recessos, etc.

O ideal seria que a determinação deste período se desse após os professores fornecerem a lista de PP que estão interessados em utilizar, pois o número de padrões selecionados é uma variável importante nesta decisão. Infelizmente, na prática, os cursos têm seus cronogramas determinados e bastante rígidos, e a extensão do período de experiências com padrões se torna inconveniente.

No nosso trabalho, definimos 2 semanas de experiências para a utilização dos padrões. Em seguida, cada professor forneceu uma pequena lista dos padrões que se adequariam ao conteúdo abordado em suas disciplinas naquele momento do período letivo. A partir destas listas, definimos os padrões que participariam das experiências. Como a lista de padrões recebida dos professores ainda era extensa para o período, adotamos alguns critérios para seleção dos padrões:

1 – Limitações detectadas – padrões que apresentam os problemas de utilização citados no Capítulo 1. Demos preferência a estes padrões porque eles estão de acordo com nossos objetivos e deixam mais nítidos os efeitos da utilização de uma ferramenta em sua aplicação.

2 – Repetição – padrões presentes em maior número de listas.

O resultado deste processo foi a escolha dos seguintes padrões pedagógicos :

- *What did you eat for breakfast* [www-lifia.info.unlp.edu.ar/ppp/pp12.htm]
- *Brainstorming* [www-lifia.info.unlp.edu.ar/ppp/ pp4.htm]
- *Round Robin* [www-lifia.info.unlp.edu.ar/ppp/pp6.htm]
- *Class Concept Map* [www-lifia.info.unlp.edu.ar/ppp/pp38.htm]

- *Mission Impossible* [www-lifia.info.unlp.edu.ar/ppp/pp21.htm]
- *EPIC* [www-lifia.info.unlp.edu.ar/ppp/pp11.htm]
- *RCLAP* [www-lifia.info.unlp.edu.ar/ppp/pp2.htm]

O anexo 3 contem o texto integral destes padrõs.

A utilização dos padrões deve ser monitorada para que se tenha informações de quantos padrões, alunos, turmas e disciplinas passaram pelas experiências. No nosso caso, após as duas semanas de experiências, obtivemos a distribuição de utilização dos PP demonstrada na tabela 2.1. A montagem desta tabela finaliza a primeira fase das atividades determinadas pela metodologia que propomos e adotamos. O passo seguinte nesse processo, descrito a seguir, é a seleção de um dos padrões utilizados para ser suportado por uma ferramenta de software.

Padrão Pedagógico	**Disciplina**	**N. de Turmas**	**N. de Alunos**[1]
What did you eat for breakfast	Linguagem Técnica 1 Linguagem Técnica 2	4	97
Brainstorming	Linguagem Técnica 2	2	16
Round Robin	Introdução a Computação	2	85
Class Concept Map	Introdução a Computação Linguagem Técnica 1 Sistemas Operacionais Linguagem Técnica 2	7	198
Mission Impossible	Algoritmos Linguagem Técnica 1	4	152
EPIC	Algoritmos	2	80
RCLAP	Linguagem Técnica 1	2	72

Tabela 2.1 – Quadro de utilização dos PP nas duas semanas de experiências

[1] A soma dos valores da coluna *N. de Alunos* excede o universo de alunos que trabalhamos, por haver alunos matriculados em mais de uma das disciplinas em que os PP foram experimentados.

A montagem da tabela acima finaliza a primeira fase das atividades determinadas pela metodologia que propomos e adotamos. O passo seguinte nesse processo, descrito a seguir, é a seleção de um dos padrões utilizados para ser suportado por uma ferramenta de software.

2.2 – Selecionando um Padrão Pedagógico para Receber Suporte de uma Ferramenta de Software.

A segunda seleção a ser realizada deve contar com grande participação dos professores que experimentaram os PP em suas aulas. A experiência prática obtida na fase de utilização dos padrões é que dá origem às informações não reportadas no texto dos padrões utilizados.

A escolha do padrão mais indicado para receber uma ferramenta de software deve ser feita baseada em critérios bem definidos e previamente estipulados. A obediência a esses critérios deve ser exigida de todos os professores participantes dos experimentos.

A falta de literatura e referências que tratassem de um contexto tão específico e aparentemente inédito (seleção de PP mais adequando para construirmos um software que o suportasse) dificulta a determinação destes critérios. No nosso trabalho, elaboramos um mecanismo de escolha que utiliza critérios intuitivos. O que poderá ser percebido a seguir é que, embora intuitivos, estes critérios são baseados em heurísticas bastante consistentes.

2.2.1 – Critérios Obedecidos na Escolha do Padrão Pedagógico

O mecanismo de avaliação que utilizamos é composto por dois elementos simples:

- Um conjunto de critérios.
- Uma seqüência de aplicações dos critérios definidos.

O conjunto de critérios está dividido em dois grupos que apresentamos e detalhamos a seguir.

1. ***Critérios Pedagógicos***

   Elegem o padrão baseado em aspectos inerentes ao processo ensino – aprendizagem. Neste grupo foram utilizados os três critérios abaixo:

- **Qualidade Didática** Avalia os resultados obtidos pelos alunos em termos de aprendizado do conteúdo ministrado sob a orientações do PP.
- **Facilidade de Aplicação** Avalia o esforço realizado por alunos e professores para montagem das atividades que compõem o PP. Um segundo aspecto também avaliado nesse critério é a infra-estrutura necessária para desempenhar estas atividades.
- **Versatilidade** Avalia a capacidade do PP de ser aplicado em disciplinas e cursos que abordam conteúdos de natureza diversificada.

2. *Critérios Tecnológicos*

Questiona a complexidade do software necessário para suportar o padrão selecionado, e infere o esforço necessário para elaboração deste software.

Para este grupo de critérios, estipulamos dois questionamentos básicos a serem respondidos para cada padrão:

É possível?

Aqui analisamos se os recursos computacionais existentes são suficientes para construirmos um software que suporte as atividades que compõem o PP. Em alguns casos, o que impossibilita o uso de ferramenta é a própria natureza de atividades do padrão, como por exemplo deslocar os alunos até uma empresa de desenvolvimento de software [www-lifia.info.unlp.edu.ar/ppp/pp15.htm]. Neste caso, é inimaginável uma ferramenta que substituísse (ou mesmo diminuísse) o esforço de alunos e professores no desempenho da tarefa.

Vale a pena?

Este critério avalia a relação custo- benefício oferecida pela construção de uma ferramenta para o padrão. O resultado obtido pelo PP neste critério depende da forma que ele responder às seguintes perguntas:

- Quais as vantagens que a aula sob o controle da ferramenta concebida teria sobre a aula convencional?
- É fácil manipular uma ferramenta que suporte este padrão?
  Este questionamento infere se o *overhead* de utilização de uma ferramenta para aquele PP é pequeno o bastante para valer a pena aplicá-la.
- Os benefícios que se espera receber com o uso da ferramenta valem o esforço estimado para sua construção?

Completando o mecanismo que criamos para a eleição do PP mais adequado, definimos a forma apropriada de adoção dos critérios estabelecidos. De acordo com esta forma, os padrões utilizados passam inicialmente pelos critérios pedagógicos. Aquele que for eleito como mais adequado é então submetido aos critérios tecnológicos. Estes critérios podem indicar o PP como inadequado, reiniciando a avaliação com o PP classificado na posição imediatamente inferior ao eleito. O processo se repete até que um dos PP passe por todos os critérios.

2.2.2 – O Padrão Pedagógico Selecionado

Os resultados obtidos na avaliação baseada no mecanismo da seção 2.2.1 apontaram o padrão ***Class Concept Map*** (**Aula em Mapa de Conceitos** [2]) como o mais adequado para se avaliar os efeitos da aplicação de uma ferramenta de software no suporte das atividades de um PP. Este padrão foi bem avaliado em todos os critérios que adotamos na seleção.

Qualidade Didática – o padrão se revelou capaz de melhorar o aproveitamento dos alunos nos assuntos em que foi utilizado. Há alguns registros de utilização dos ele-

[2] Adaptamos o título para português.

mentos do padrão pelos estudantes, independente de recomendação ou solicitação dos professores, como forma de organização dos seus estudos.

Em experiências de utilização deste padrão, os alunos que abordaram um assunto aplicando o Aula em Mapa de Conceitos tiveram aproveitamento até 30% melhor que outros alunos do mesmo período, professor e disciplina que não utilizaram o padrão.

Facilidade de Aplicação – o padrão é oneroso para o professor no momento de sua aplicação, pois a forma de comunicação proposta costuma tumultuar a aula. Outro aspecto negativo é a dificuldade do aluno em registrar todas as informações trocadas no decorrer da aula.

Versatilidade – o padrão foi utilizado com sucesso nas seguintes situações:

Disciplinas diferentes e do mesmo curso.

Disciplinas de cursos diferentes.

Assuntos diferentes de uma mesma disciplina.

Em todos os casos, a sua aplicação foi positiva e demandou pouquíssimas adaptações (descritas no Capítulo 3).

Quanto aos critérios tecnológicos, o PP também não sofreu restrições, pois como constataremos a seguir, as atividades do padrão são perfeitamente suportáveis por uma ferramenta de software (é possível). Quanto ao fator custo – beneficio, o padrão Aula em Mapas de Conceito carrega os problemas inicialmente detectados nos PP catalogados, e serve como um ótimo exemplo de superação destes problemas pelo auxílio de um software adequado.

No que se refere ao segundo critério técnico (**vale a pena?**), o padrão responde de forma satisfatória a todas as perguntas levantadas.

Como tem boa parte das atividades do padrão que pode ser auxiliada ou substituída por um software, a aula com o uso do software tende a ser mais fluente e produtiva que no modelo original (o uso de ferramenta traria grande vantagem em relação a aula tradicional).

Como as atividades do padrão demandam bastante esforço e organização do professor e alunos, assumimos que a manipulação de uma ferramenta que substitua e auxilie as ati-

vidades do padrão é bem menos oneroso que as próprias atividades (a ferramenta é de fácil manipulação).

As atividades que compõem o padrão já são suportadas por ferramentas de software, como *e-mail* para distribuição de tarefas, editores de mapa de conceitos [Reader94] para criação dos mini-mapas e do mapa geral (vide capítulo 3), ferramentas de FTP para troca de arquivos dos mapas elaborados. Isto nos leva a crer que elaborar uma ferramenta que reuna todas as funcionalidades não é uma tarefa de complexidade desencorajadora (os benefícios valem o esforço).

Depois de eleger o PP que receberá uma ferramenta de software para apoiá-lo, o passo seguinte (passo 3) é partir para uma série de experiências com o padrão selecionado. Estas experiências formarão a base das informações buscadas na fase de levantamento de requisitos, que precede o início da construção da ferramenta para o PP. Em nosso trabalho, essa etapa deu origem aos requisitos da ferramenta Mapa Fácil, que abordamos em detalhes no Capítulo 4 deste documento.

2.2.3 – Levantamento de Requisitos

Os requisitos levantados têm como origem duas fontes principais de informação:

1 – Entrevistas com os professores que utilizaram o padrão no período de avaliação.

2 – Entrevistas com um amostra de alunos selecionados aleatoriamente de todas as turmas participantes das experiências.

Na nossa experiência, foram ouvidos 19 alunos nesta fase.

Uma terceira fonte de informações é o próprio texto do padrão catalogado. Como a catalogação de todos eles foi precedida de experiências em sala, é possível que o texto já apresente algumas limitações que possam ser supridas com o apoio de ferramentas.

Como os processos de levantamento de requisitos recomendam [Lipschutz93], as informações levantadas devem ser registradas. Essas informações guiaram as atividades de construção do software para o padrão pedagógico.

Para o nosso caso, as entrevistas que mencionamos nos itens 1 e 2 acima foram registradas nos formulários exibidos nas figuras 4.1 e 4.2. Além de registrar os dados levantados, estes formulários também foram importantes para definir um foco para as entrevistas. Desta forma, podemos otimizar a fase que investigou as necessidades da ferramenta que, poste

Levantamento de requisitos para uma ferramenta que suporte o padrão Aula em Mapa de Conceitos

Enumere as dificuldades percebidas na elaboração do padrão nas atividades a seguir:

1 - Elaboração da lista de conceitos

1	4	7	...
2	5	8	
3	6	9	

2 - Distribuição dos conceitos

1	4	7	...
2	5	8	
3	6	9	

3 - Aula de montagem do mapa geral

a - Coleta das participações dos alunos

1	4	7	...
2	5	8	
3	6	9	

b - Desenho do mapa

1	4	7	...
2	5	8	
3	6	9	

Figura 2.1 – Formulário de coleta de requisitos para professores.

Levantamento de requisitos para uma ferramenta que suporte o padrão Aula em Mapa de Conceitos

Enumere as dificuldades percebidas na elaboração do padrão nas atividades a seguir:

1 - Elaboração do mini-mapa.

1	4	7	...
2	5	8	
3	6	9	

2 - Aula de montagem do mapa geral.

a - Apresentação das sugestões.

1	4	7	...
2	5	8	
3	6	9	

b - Registro das informações apresentadas pelos demais alunos.

1	4	7	...
2	5	8	

Figura 2.2 – Formulário de coleta de requisitos para alunos.

A interação com professores e alunos, registradas nos formulários acima, formaram um conjunto de requisitos da ferramenta Mapa Fácil. A descrição - tanto dos requisitos como da especificação da ferramenta - é melhor compreendida após o conhecimento das atividades desenvolvidas nas aulas regidas pelo padrão **Aula em Mapa de Conceitos**. Por este motivo, apresentaremos e exemplificaremos estas atividades no capítulo a seguir.

Experiências com Padrões Pedagógicos em Sala de Aula

Como já mencionado no capítulo introdutório, o objetivo do nosso trabalho é adicionar o aspecto ferramenta de software às aulas com Padrões Pedagógicos (PP) e, em seguida, avaliar os efeitos gerados por esta adição. A abordagem descrita abaixo é a metodologia que definimos como adequada para chegar aos objetivos traçados. Esta metodologia consiste num conjunto de atividades que formam o caminho entre o primeiro contato com os Padrões Pedagógicos catalogados, e a especificação de uma ferramenta que suporta um dos padrões utilizados. Três fases marcaram esta etapa do trabalho:

1 - Utilização de vários PP com turmas, disciplinas, professores, alunos e cursos diferentes.

2 - Seleção de um PP, dentre os experimentados, para elaboração de uma ferramenta de software que o suporte.

3 – Realização de aulas com o padrão selecionado, seguidas de levantamento de requisitos para a ferramenta.

Parte da metodologia (passos 1 e 2) é uma proposta nossa, que se originou da necessidade de atacarmos os problemas detectados no catálogo de Padrões Pedagógicos de forma organizada e estruturada. Esta parte passou por discussões acadêmicas envolvendo professores e alunos, e estas discussões serviram para aperfeiçoar a proposta e, mesmo que de uma forma preliminar, validá-la.

A outra parte da metodologia (passo 3) está espelhada e apoiada em métodos já consagrados na bibliografia de Engenharia de Software que trata de levantamento de requisitos [Thayer98].

O objetivo da metodologia que propomos não é garantir com extrema precisão que o padrão selecionado, a partir dos dois passos iniciais, seja o mais indicado para receber suporte de um ferramenta de software. O seu objetivo é apenas tornar a seleção menos aleatória e, portanto, com maior chance de sucesso nas atividades de especificação e implementação que seguem. A elaboração de uma forma precisa de escolha, além de requerer mais estudos e experiências, não é parte das metas do nosso trabalho.

A seguir, detalharemos os passos citados acima, apresentando as peculiaridades de cada um deles.

2.1 – Utilização de Padrões Pedagógicos

Esta fase da metodologia consiste em utilizar os Padrões Pedagógicos catalogados em disciplinas, cursos, nível de alunos e professores diferentes. O seu objetivo é criar o universo mais abrangente e heterogêneo possível para a tomada de decisão que será feita nas fases seguintes do processo. Ambientes assim favorecem a qualidade do experimento, por-

que diminuem a influência de aspectos particulares às amostras nos resultados gerais obtidos [Lipschutz93] .

No nosso trabalho, este universo foi montado no decorrer das atividades letivas dos cursos de Ciências da Computação e Processamento de Dados da Faculdade Ruy Barbosa [www.frb.br].

Quanto maior acesso ao quadro de professores do ambiente onde se dão esta fase dos experimentos, maiores serão as chances de criar um contexto mais abrangente para o desempenho deste passo do processo. No nosso caso, conseguimos montar um universo satisfatório por ser bastante abrangente e diversificado. Nossas experiências se deram nas disciplinas listadas abaixo, e a diversidade foi assegurada pelo fato destas disciplinas serem de cursos, turnos, período, turmas, professores, e assuntos diferentes.

- Curso Ciências da Computação
  - Algoritmos – duas turmas.
  - Introdução a Computação – uma turma.
  - Linguagem Técnica 2 – duas turmas.
- Curso Processamento de Dados.
  - Linguagem Técnica 1 – duas turmas.
  - Sistemas Operacionais – uma turma.

O fator que determina quais disciplinas farão parte dos experimentos é a disponibilidade dos professores consultados, a menos que se consiga algum comprometimento com as autoridades acadêmicas dos cursos em que se dará o experimento. Este comprometimento demanda um processo de interação mais difícil, por isso recomendamos que a decisão de envolver a disciplina caiba a cada professor titular, de acordo com sua disponibilidade e interesse pelo o projeto desenvolvido. Este foi o procedimento utilizado nas nossas experiências, e através dele conseguimos definir o seguinte universo de atuação para a utilização dos PP em salas de aula:

5 Disciplinas

6 Professores

8 Turmas

246 Alunos (matriculados)

Depois de definido o universo, os professores envolvidos nos experimentos devem consultar o catálogo de padrões. Nesta consulta, eles identificarão quais padrões se encaixam com as suas atividades acadêmicas. Cabe a quem estiver coordenando esta fase de seleção planejar a duração do período de experiências com os padrões. O tempo deve ser longo o bastante para utilizar os padrões escolhidos, e deve estar de acordo com as demais atividades acadêmicas dos cursos como provas, recessos, etc.

O ideal seria que a determinação deste período se desse após os professores fornecerem a lista de PP que estão interessados em utilizar, pois o número de padrões selecionados é uma variável importante nesta decisão. Infelizmente, na prática, os cursos têm seus cronogramas determinados e bastante rígidos, e a extensão do período de experiências com padrões se torna inconveniente.

No nosso trabalho, definimos 2 semanas de experiências para a utilização dos padrões. Em seguida, cada professor forneceu uma pequena lista dos padrões que se adequariam ao conteúdo abordado em suas disciplinas naquele momento do período letivo. A partir destas listas, definimos os padrões que participariam das experiências. Como a lista de padrões recebida dos professores ainda era extensa para o período, adotamos alguns critérios para seleção dos padrões:

1 – Limitações detectadas – padrões que apresentam os problemas de utilização citados no Capítulo 1. Demos preferência a estes padrões porque eles estão de acordo com nossos objetivos e deixam mais nítidos os efeitos da utilização de uma ferramenta em sua aplicação.

2 – Repetição – padrões presentes em maior número de listas.

O resultado deste processo foi a escolha dos seguintes padrões pedagógicos :

- *What did you eat for breakfast* [www-lifia.info.unlp.edu.ar/ppp/pp12.htm]
- *Brainstorming* [www-lifia.info.unlp.edu.ar/ppp/ pp4.htm]
- *Round Robin* [www-lifia.info.unlp.edu.ar/ppp/pp6.htm]
- *Class Concept Map* [www-lifia.info.unlp.edu.ar/ppp/pp38.htm]

- *Mission Impossible* [www-lifia.info.unlp.edu.ar/ppp/pp21.htm]
- *EPIC* [www-lifia.info.unlp.edu.ar/ppp/pp11.htm]
- *RCLAP* [www-lifia.info.unlp.edu.ar/ppp/pp2.htm]

O anexo 3 contem o texto integral destes padrõs.

A utilização dos padrões deve ser monitorada para que se tenha informações de quantos padrões, alunos, turmas e disciplinas passaram pelas experiências. No nosso caso, após as duas semanas de experiências, obtivemos a distribuição de utilização dos PP demonstrada na tabela 2.1. A montagem desta tabela finaliza a primeira fase das atividades determinadas pela metodologia que propomos e adotamos. O passo seguinte nesse processo, descrito a seguir, é a seleção de um dos padrões utilizados para ser suportado por uma ferramenta de software.

Padrão Pedagógico	**Disciplina**	**N. de Turmas**	**N. de Alunos**[1]
What did you eat for breakfast	Linguagem Técnica 1 Linguagem Técnica 2	4	97
Brainstorming	Linguagem Técnica 2	2	16
Round Robin	Introdução a Computação	2	85
Class Concept Map	Introdução a Computação Linguagem Técnica 1 Sistemas Operacionais Linguagem Técnica 2	7	198
Mission Impossible	Algoritmos Linguagem Técnica 1	4	152
EPIC	Algoritmos	2	80
RCLAP	Linguagem Técnica 1	2	72

Tabela 2.1 – Quadro de utilização dos PP nas duas semanas de experiências

[1] A soma dos valores da coluna *N. de Alunos* excede o universo de alunos que trabalhamos, por haver alunos matriculados em mais de uma das disciplinas em que os PP foram experimentados.

A montagem da tabela acima finaliza a primeira fase das atividades determinadas pela metodologia que propomos e adotamos. O passo seguinte nesse processo, descrito a seguir, é a seleção de um dos padrões utilizados para ser suportado por uma ferramenta de software.

2.2 – Selecionando um Padrão Pedagógico para Receber Suporte de uma Ferramenta de Software.

A segunda seleção a ser realizada deve contar com grande participação dos professores que experimentaram os PP em suas aulas. A experiência prática obtida na fase de utilização dos padrões é que dá origem às informações não reportadas no texto dos padrões utilizados.

A escolha do padrão mais indicado para receber uma ferramenta de software deve ser feita baseada em critérios bem definidos e previamente estipulados. A obediência a esses critérios deve ser exigida de todos os professores participantes dos experimentos.

A falta de literatura e referências que tratassem de um contexto tão específico e aparentemente inédito (seleção de PP mais adequando para construirmos um software que o suportasse) dificulta a determinação destes critérios. No nosso trabalho, elaboramos um mecanismo de escolha que utiliza critérios intuitivos. O que poderá ser percebido a seguir é que, embora intuitivos, estes critérios são baseados em heurísticas bastante consistentes.

2.2.1 – Critérios Obedecidos na Escolha do Padrão Pedagógico

O mecanismo de avaliação que utilizamos é composto por dois elementos simples:

- Um conjunto de critérios.
- Uma seqüência de aplicações dos critérios definidos.

O conjunto de critérios está dividido em dois grupos que apresentamos e detalhamos a seguir.

1. ***Critérios Pedagógicos***

   Elegem o padrão baseado em aspectos inerentes ao processo ensino – aprendizagem. Neste grupo foram utilizados os três critérios abaixo:

- **Qualidade Didática** Avalia os resultados obtidos pelos alunos em termos de aprendizado do conteúdo ministrado sob a orientações do PP.
- **Facilidade de Aplicação** Avalia o esforço realizado por alunos e professores para montagem das atividades que compõem o PP. Um segundo aspecto também avaliado nesse critério é a infra-estrutura necessária para desempenhar estas atividades.
- **Versatilidade** Avalia a capacidade do PP de ser aplicado em disciplinas e cursos que abordam conteúdos de natureza diversificada.

2. Critérios Tecnológicos

Questiona a complexidade do software necessário para suportar o padrão selecionado, e infere o esforço necessário para elaboração deste software.

Para este grupo de critérios, estipulamos dois questionamentos básicos a serem respondidos para cada padrão:

É possível?

Aqui analisamos se os recursos computacionais existentes são suficientes para construirmos um software que suporte as atividades que compõem o PP. Em alguns casos, o que impossibilita o uso de ferramenta é a própria natureza de atividades do padrão, como por exemplo deslocar os alunos até uma empresa de desenvolvimento de software [www-lifia.info.unlp.edu.ar/ppp/pp15.htm]. Neste caso, é inimaginável uma ferramenta que substituísse (ou mesmo diminuísse) o esforço de alunos e professores no desempenho da tarefa.

Vale a pena?

Este critério avalia a relação custo- benefício oferecida pela construção de uma ferramenta para o padrão. O resultado obtido pelo PP neste critério depende da forma que ele responder às seguintes perguntas:

- Quais as vantagens que a aula sob o controle da ferramenta concebida teria sobre a aula convencional?
- É fácil manipular uma ferramenta que suporte este padrão?
  Este questionamento infere se o *overhead* de utilização de uma ferramenta para aquele PP é pequeno o bastante para valer a pena aplicá-la.
- Os benefícios que se espera receber com o uso da ferramenta valem o esforço estimado para sua construção?

Completando o mecanismo que criamos para a eleição do PP mais adequado, definimos a forma apropriada de adoção dos critérios estabelecidos. De acordo com esta forma, os padrões utilizados passam inicialmente pelos critérios pedagógicos. Aquele que for eleito como mais adequado é então submetido aos critérios tecnológicos. Estes critérios podem indicar o PP como inadequado, reiniciando a avaliação com o PP classificado na posição imediatamente inferior ao eleito. O processo se repete até que um dos PP passe por todos os critérios.

2.2.2 – O Padrão Pedagógico Selecionado

Os resultados obtidos na avaliação baseada no mecanismo da seção 2.2.1 apontaram o padrão ***Class Concept Map*** (**Aula em Mapa de Conceitos** [2]) como o mais adequado para se avaliar os efeitos da aplicação de uma ferramenta de software no suporte das atividades de um PP. Este padrão foi bem avaliado em todos os critérios que adotamos na seleção.

Qualidade Didática – o padrão se revelou capaz de melhorar o aproveitamento dos alunos nos assuntos em que foi utilizado. Há alguns registros de utilização dos ele-

[2] Adaptamos o título para português.

mentos do padrão pelos estudantes, independente de recomendação ou solicitação dos professores, como forma de organização dos seus estudos.

Em experiências de utilização deste padrão, os alunos que abordaram um assunto aplicando o Aula em Mapa de Conceitos tiveram aproveitamento até 30% melhor que outros alunos do mesmo período, professor e disciplina que não utilizaram o padrão.

Facilidade de Aplicação – o padrão é oneroso para o professor no momento de sua aplicação, pois a forma de comunicação proposta costuma tumultuar a aula. Outro aspecto negativo é a dificuldade do aluno em registrar todas as informações trocadas no decorrer da aula.

Versatilidade – o padrão foi utilizado com sucesso nas seguintes situações:

Disciplinas diferentes e do mesmo curso.

Disciplinas de cursos diferentes.

Assuntos diferentes de uma mesma disciplina.

Em todos os casos, a sua aplicação foi positiva e demandou pouquíssimas adaptações (descritas no Capítulo 3).

Quanto aos critérios tecnológicos, o PP também não sofreu restrições, pois como constataremos a seguir, as atividades do padrão são perfeitamente suportáveis por uma ferramenta de software (é possível). Quanto ao fator custo – benefício, o padrão Aula em Mapas de Conceito carrega os problemas inicialmente detectados nos PP catalogados, e serve como um ótimo exemplo de superação destes problemas pelo auxílio de um software adequado.

No que se refere ao segundo critério técnico (**vale a pena?**), o padrão responde de forma satisfatória a todas as perguntas levantadas.

Como tem boa parte das atividades do padrão que pode ser auxiliada ou substituída por um software, a aula com o uso do software tende a ser mais fluente e produtiva que no modelo original (o uso de ferramenta traria grande vantagem em relação a aula tradicional).

Como as atividades do padrão demandam bastante esforço e organização do professor e alunos, assumimos que a manipulação de uma ferramenta que substitua e auxilie as ati-

vidades do padrão é bem menos oneroso que as próprias atividades (a ferramenta é de fácil manipulação).

As atividades que compõem o padrão já são suportadas por ferramentas de software, como *e-mail* para distribuição de tarefas, editores de mapa de conceitos [Reader94] para criação dos mini-mapas e do mapa geral (vide capítulo 3), ferramentas de FTP para troca de arquivos dos mapas elaborados. Isto nos leva a crer que elaborar uma ferramenta que reuna todas as funcionalidades não é uma tarefa de complexidade desencorajadora (os benefícios valem o esforço).

Depois de eleger o PP que receberá uma ferramenta de software para apoiá-lo, o passo seguinte (passo 3) é partir para uma série de experiências com o padrão selecionado. Estas experiências formarão a base das informações buscadas na fase de levantamento de requisitos, que precede o início da construção da ferramenta para o PP. Em nosso trabalho, essa etapa deu origem aos requisitos da ferramenta Mapa Fácil, que abordamos em detalhes no Capítulo 4 deste documento.

2.2.3 – Levantamento de Requisitos

Os requisitos levantados têm como origem duas fontes principais de informação:

1 – Entrevistas com os professores que utilizaram o padrão no período de avaliação.

2 – Entrevistas com um amostra de alunos selecionados aleatoriamente de todas as turmas participantes das experiências.

Na nossa experiência, foram ouvidos 19 alunos nesta fase.

Uma terceira fonte de informações é o próprio texto do padrão catalogado. Como a catalogação de todos eles foi precedida de experiências em sala, é possível que o texto já apresente algumas limitações que possam ser supridas com o apoio de ferramentas.

Como os processos de levantamento de requisitos recomendam [Lipschutz93], as informações levantadas devem ser registradas. Essas informações guiaram as atividades de construção do software para o padrão pedagógico.

Para o nosso caso, as entrevistas que mencionamos nos itens 1 e 2 acima foram registradas nos formulários exibidos nas figuras 4.1 e 4.2. Além de registrar os dados levantados, estes formulários também foram importantes para definir um foco para as entrevistas. Desta forma, podemos otimizar a fase que investigou as necessidades da ferramenta que, poste

Levantamento de requisitos para uma ferramenta que suporte o padrão Aula em Mapa de Conceitos

Enumere as dificuldades percebidas na elaboração do padrão nas atividades a seguir:

1 - Elaboração da lista de conceitos

1	4	7	...
2	5	8	
3	6	9	

2 - Distribuição dos conceitos

1	4	7	...
2	5	8	
3	6	9	

3 - Aula de montagem do mapa geral

a - Coleta das participações dos alunos

1	4	7	...
2	5	8	
3	6	9	

b - Desenho do mapa

1	4	7	...
2	5	8	
3	6	9	

Figura 2.1 – Formulário de coleta de requisitos para professores.

Levantamento de requisitos para uma ferramenta que suporte o padrão Aula em Mapa de Conceitos

Enumere as dificuldades percebidas na elaboração do padrão nas atividades a seguir:

1 - Elaboração do mini-mapa.

1	4	7	...
2	5	8	
3	6	9	

2 - Aula de montagem do mapa geral.

a - Apresentação das sugestões.

1	4	7	...
2	5	8	
3	6	9	

b - Registro das informações apresentadas pelos demais alunos.

1	4	7	...
2	5	8	

Figura 2.2 – Formulário de coleta de requisitos para alunos.

A interação com professores e alunos, registradas nos formulários acima, formaram um conjunto de requisitos da ferramenta Mapa Fácil. A descrição - tanto dos requisitos como da especificação da ferramenta - é melhor compreendida após o conhecimento das atividades desenvolvidas nas aulas regidas pelo padrão **Aula em Mapa de Conceitos**. Por este motivo, apresentaremos e exemplificaremos estas atividades no capítulo a seguir.

O Padrão Pedagógico Aula em Mapa de Conceitos

3.1. Mapas de Conceito

Como o título sugere, o padrão Aula em Mapa de Conceito baseia-se na utilização de mapas de conceitos [Reader94] para melhor estruturar e organizar o conjunto de informações apresentadas ao aluno no decorrer de um curso. Construindo e utilizando o mapa de conceitos, o aluno é capaz de ter uma visão geral e didática de todo o conteúdo recebido, através da clara determinação das relações existentes entre cada um dos tópicos que lhe foram apresentados.

Mapa de conceitos, a estrutura fundamental utilizada por este padrão, consiste num grafo onde os nodos apresentam um rótulo relativo a um conceito, e as ligações entre os nodos (também rotuladas) descrevem a relação entre os conceitos. A figura 3.1 apresenta um mapa de conceitos contendo informações da disciplina Sistemas Operacionais, comum nos currículos de graduação em Ciências da Computação.

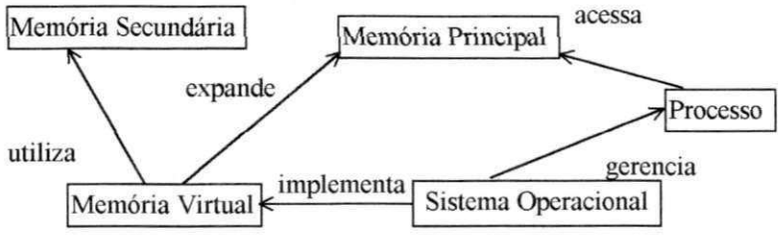


Figura 3.1 - Mapa de Conceitos para a disciplina Sistemas Operacionais

Uma das possíveis leituras do mapa da figura 3.1 trás as seguintes informações: o **sistema operacional** gerencia os **processos**, que por sua vez acessam a **memória principal**. A **memória virtual**, que é implementada pelo **sistema operacional,** utiliza a memória secundária e expande a **memória principal**.

A grande contribuição do Mapa de Conceitos como recurso didático pode ser percebida no exemplo apresentado. Cada um dos conceitos presentes no mapa da figura 3.1 são exaustivamente detalhados no decorrer de um curso de Sistemas Operacionais. Esse detalhamento se dá em momentos bem distintos da disciplina, de forma que cada um dos conceitos é tratado por vez. A conseqüência disto é que o aluno perde a noção do todo, por estar demasiadamente concentrado nas minúcias de cada parte no decorrer do curso. Ou seja, o aluno domina todos os tópicos abordados no curso de Sistemas Operacionais, se tornando capaz de relatar os mais peculiares aspectos de implementação dos mecanismos de gerência de memória, comunicação entre processos, resolução de *deadlocks*, etc (as partes). Por outro lado, ele sente dificuldade para descrever como funciona um Sistema Operacional (o todo), que era a meta alvejada quando se estava estudando cada um de seus componentes e mecanismos.

O padrão Aula em Mapa de Conceitos, descrito a seguir, explora as potencialidades didáticas dos mapas de conceitos para elaborar aulas baseadas em pesquisa, revisão de

assuntos abordados, intensa participação dos alunos e, sobretudo, interação aluno-professor e vice-versa.

3.2 Dinâmica do Padrão Aula em Mapa de Conceito

A dinâmica que descreveremos a seguir está catalogada e disponível no site do projeto de PP (www-lifia.info.unlp.edu.ar/ppp/), embora na forma sucinta que determina a estrutura dos PP. Em nossa descrição, ressaltamos alguns detalhes observados na utilização do padrão em sala de aula. Destas observações se origina uma série de modificações e recomendações que propomos para a melhor aplicação do padrão Aula em Mapa de Conceitos.

De acordo com o padrão pedagógico Aula em Mapa de Conceitos, o primeiro passo a ser dado pelo professor é distribuir entre os alunos uma lista de conceitos abordados em sua disciplina. Por exemplo:

Memória secundária.

Memória principal.

Memória virtual.

Processo.

Sistema operacional.

A experiência de utilização do padrão mostra que o final da aula é o momento mais propício para esta atividade de distribuição de conceitos por, sobretudo, dois motivos:

- os alunos já têm a cultura de receber atividades para a próxima aula no final da aula corrente. É um comportamento também observado nas experiências em sala, onde os alunos costumam entrar num processo crescente de dispersão da atenção a medida que o final da aula se aproxima. Essa dispersão se desfaz quando o professor, encerrado o assunto da aula corrente, anuncia uma atividade para a aula seguinte. Neste momento, o nível de atenção da sala é reestabelecido e o professor terá o ambiente propício para especificar a atividade. Como

observaremos a seguir, os conceitos distribuídos serão matéria de uma atividade a ser desenvolvida em casa e trazida na aula seguinte, ou seja, é uma 'atividade para a próxima aula', portanto melhor anunciá-la no final da aula corrente.

- quando os conceitos são distribuídos no início da aula, os alunos terão sua atenção desviada para o assunto da aula corrente por pelo menos uma hora, tempo mais que bastante para esquecerem a atividade especificada no início. Nas experiências utilizando este padrão, nós detectamos que esta dispersão dos alunos leva à uma segunda explicação da tarefa, desta vez no final da aula.

Na distribuição, cada aluno deve receber dois ou três conceitos desta lista, de forma que, dependendo da quantidade de alunos, alguns conceitos serão recebidos por mais de um aluno. Abaixo descrevemos uma possível distribuição dos conceitos exemplificados acima para dois alunos hipotéticos.

Aluno A – Memória Secundária, Processo, Sistema Operacional
Aluno B – Memória Virtual, Memória principal, Processo

Outro fato verificado nas experiências de sala de aula é que o número máximo de alunos participantes da aula de montagem do mapa geral, descrita adiante, deve estar entre 15 e 20 alunos. Acima, disto dois fatos tendem a surgir, comprometendo o aprendizado da turma e, portanto, diminuindo a qualidade da aula:

1 - As participações se tornam desnorteadas, com 3 ou 4 alunos se pronunciando simultaneamente.
2 - As participações passam a ficar restritas a um pequeno grupo de alunos mais voluntariosos, enquanto os demais se retraem por não estarem dispostos a disputar a atenção do professor para suas sugestões.

Como solução para esta limitação, foi adotado com sucesso a formação de grupos para composição de mini-mapas. Os grupo de dois ou três alunos receberão seus conceitos

e procederão os demais passos do padrão. Desta forma o padrão ganha em versatilidade, podendo assim ser aplicado em turmas de até sessenta ou setenta alunos. Embora não tenhamos experimentado, os resultados com grupos nos leva a crer que o padrão é escalável para turmas ainda maiores, bastando apenas o aumento do número de participantes por grupo.

De posse dos seus conceitos, o passo seguinte a ser dado por cada aluno[1] é pesquisar os conceitos apresentados, a fim de dominá-los, e escrever uma definição para cada um deles. Para efeito de exemplificação, descrevemos abaixo um possível resultado da pesquisa realizada sobre a lista de conceitos apresentada.

- Aluno A

  Memória secundária - Meio permanente (não volátil) armazenamento de programas e dados.

  Processo - É um programa em execução.

  Sistema Operacional - Software responsável pela gerência dos recursos do computador e do controle de seu funcionamento.

- Aluno B

  Memória virtual - Técnica de gerência de memória que combina memória principal e secundária, provendo ao usuário uma expansão da memória principal.

  Memória principal - Parte do computador onde são armazenados instruções e dados.

  Processo - Estrutura de dados composta por dados, instruções e pilha de execução.

Encerrada a pesquisa sobre os tópicos, o aluno partirá para a segunda fase do padrão Aula em Mapa de Conceitos: compor seu mapa, relacionando os conceitos pesquisados. A importância desta atividade se dá pelo fato de que neste momento o aluno buscará a relação entre os conceitos estudados, e esse é um dos objetivos do padrão. A utilização do padrão mostra que neste instante o aluno terá um ganho de aprendizado dificilmente alcançado no

[1] Embora a descrição da atividade a seguir se refira sempre a um aluno, todos os passos se aplicam para um grupo de alunos, caso a divisão em grupo tenha sido necessária.

estudo de cada tópico separadamente, pois para identificar as relações entre os conceitos ele precisará, além de pesquisá-los mais profundamente, perceber aspectos gerais como:

- Relevância que um tópico estudado tem para outro.
- Encadeamento dos conhecimentos estudados.
- Dependências entre os conceitos .

Na imensa maioria dos casos em que aplicamos o padrão Aula em Mapa de Conceitos os alunos chegam a resultados diferentes, mesmo quando recebem conceitos coincidentes. Esta pluralidade de soluções foi observada em todas as experiências de utilização do padrão, independentemente dos tópicos abordados ou mesmo da natureza da disciplina lecionada. Para uma descrição mais realista da aplicação do padrão, apresentaremos dois mapas diferentes (figuras 3.2 e 3.2). Estes mapas foram feitos por dois alunos hipotéticos, montados a partir dos conceito recebidos por cada um deles.

Aluno A

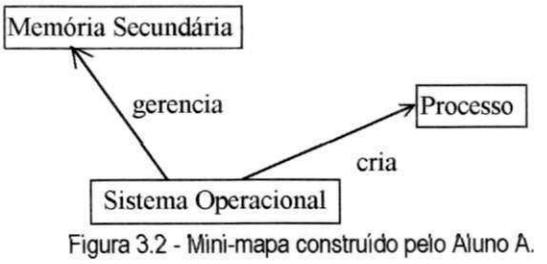


Figura 3.2 - Mini-mapa construído pelo Aluno A.

Aluno B

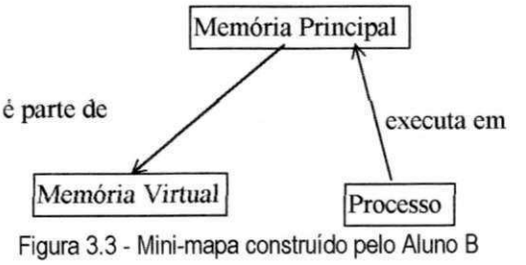


Figura 3.3 - Mini-mapa construído pelo Aluno B

Os mapas construídos pelos alunos se relacionam e se completam, formando um conjunto rico de informações pertinentes àquela parte do curso de onde o professor extraiu a lista de conceitos distribuída. A conexão entre os mini-mapas (elaborados pelos alunos) num mapa de conceitos geral se dará na última fase de atividades do padrão, em uma aula que exigirá a maior desenvoltura do professor para que a interação com os alunos se dê de forma ágil, porém organizada e didática. O próximo passo da aplicação do padrão é a

realização desta aula, onde haverá uma compilação dos resultados obtidos por cada aluno, gerando um mapa que sintetizará as pesquisas individuais em um único mapa. Este mapa é fatalmente mais completo que cada um concebido individualmente.

Para iniciar a aula que coletará os resultados dos alunos, o professor escreverá no quadro um dos conceitos contidos na lista distribuída na aula anterior.

Figura 3.4 - Conceito selecionado para iniciar a aula

A experiência mostra que conceitos fundamentais de cada área são melhores candidatos para iniciar o mapa. Melhores porque instigam a participação de vários alunos logo no início das atividades de montagem do mapa geral, quebrando a inibição da turma e fazendo as participações seguintes surgirem mais facilmente. Da nossa lista exemplo escolhemos o conceito **Processo** (figura 3.4), por ser um conceito fundamental no estudo de sistemas operacionais. Outros exemplos de boa escolha, dentre vários verificados nas aulas onde aplicamos este padrão, são:

Objeto - em uma aula de disciplinas que abordam a teoria OO.

Computador – em aulas da disciplina de introdução `a Computação.

Linguagem – em aulas da disciplina que aborda aspectos de linguagens de programação.

Feito isso, o padrão manda que o professor apresente a definição do conceito escrito no quadro e pergunte se algum dos alunos tem, em seu mapa, um conceito relacionado com o exposto.

Processo - Estrutura de dados montada pelo SO (a partir de um programa que entrará em execução) composta por uma área de código + área de pilha + área de dados.

Os alunos começarão a participar da aula apresentando os conceitos que se relacionam com o inicialmente sugerido pelo professor. Uma prática que se apresenta bastante proveitosa neste momento é o professor solicitar que, além do conceito que se relaciona com o que está escrito no quadro (**Processo**), o aluno apresente verbalmente:

- A sua definição de **Processo.**

- A relação entre **Processo** e um outro conceito.
- A definição do conceito que se relacionará com **Processo.**

Assim, os alunos terão suas definições corrigidas e comparadas, além de verificar a corretude das relações encontradas.

O resultado da participação dos alunos exemplo (Aluno A e Aluno B) no primeiro momento da aula será o mapa descrito na figura 3.5.

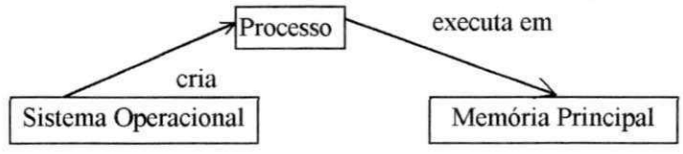


Figura 3.4 - Mapa de conceitos resultante da participação dos alunos exemplo.

O processo de coleta dos resultados dos alunos se repete, tendo agora como ponto de partida o conceito **Sistema Operacional**. A cada conceito adicionado no mapa que está sendo montado pelo professor, segue uma solicitação das relações e das definições dos conceitos sugeridos pelos alunos. Ao final desta atividade, um mapa geral (figura 3.6) estará desenhado no quadro, contendo os conceitos inicialmente listados para pesquisa e todas as relações identificadas pelos alunos na pesquisa realizada entre a aula corrente e a anterior.

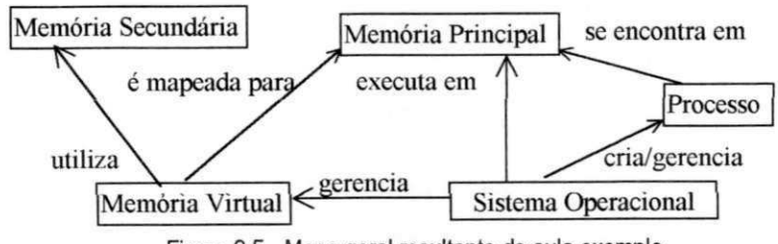


Figura 3.5 - Mapa geral resultante de aula exemplo.

Como o exemplo demonstra, o mapa final é mais completo e detalhado que cada um dos mapas construídos pelos alunos. Ao final desta aula, o aluno terá uma visão mais ampla e geral dos conceitos pesquisados, uma gama de definições para cada um dos conceitos e um mapa contendo várias relações entre cada par de conceito pesquisado. A cada participação dos alunos o professor deve avaliar se a relação sugerida é cabível e se a definição apresentada para o conceito está correta. Desta forma os alunos têm seus mapas

corrigidos e, ao mesmo tempo, a garantia de que a informação adquirida pela interação com seus colegas na aula de composição do mapa geral está correta.

De acordo com a especificação do padrão, as atividades são encerradas com a montagem do mapa geral, entretanto após repetidas aplicações em sala de aula, passamos a dedicar alguns minutos do final da aula para que os alunos anotem o mapa geral construído. Desta maneira, eles obtêm uma fonte de estudo para compreensão global dos tópicos recebidos em forma de conceitos. Esta prática se originou de solicitações por parte dos alunos verificadas em praticamente todas as aulas com o padrão. O registro do mapa geral é a última atividade do padrão pedagógico Aula em Mapa de Conceitos.

A Ferramenta Mapa Fácil

4.1 Requisitos

A ferramenta que criamos tem como objetivo geral suportar as atividades inerentes ao Padrão Pedagógico Aula em Mapa de Conceitos, descrito em detalhes no Capítulo 3.

4.1.1 - Controle de Turmas

A ferramenta deve armazenar as informações das turmas que assistirem aulas ministradas de acordo com o Padrão Pedagógico Aula em Mapa de Conceitos. As informações das turmas consistem em sua identificação (curso, série, período, etc.) e na sua composição, que corresponde a lista dos alunos que fazem parte desta turma.

4.1.2 - Distribuição Automática dos Conceitos

A ferramenta deve oferecer uma forma prática e facilitada para o professor fazer a distribuição de conceitos que, como determina o padrão, serão pesquisados pelos alunos.

4.1.3 - Edição de Mapa de Conceitos

A construção e edição de mapas de conceitos deve ser suportada pela ferramenta. Este editor deve suportar duas modalidades de construção de mapa:

Construção individual

Permite construir o mapa a partir de uma lista de conceitos pesquisados.

Construção interativa

1. Com comunicação oral. A ferramenta deve prover uma forma esquemática e organizada de construção de um mapa de conceito, através da colaboração entre os alunos e o professor. Para este caso, a colaboração se dá oralmente.
2. Com comunicação via rede. O objetivo a ser suprido pela ferramenta é o mesmo da comunicação oral, sendo que o meio utilizado para prover a colaboração é a interação automática, via uma rede de computadores.

Em ambos os casos o professor deve dispor de um mecanismo para marcar determinado estado do mapa e, depois de uma série de modificações, retornar para o estado marcado. Este retorno deve desfazer os efeitos das alterações, devolvendo o mapa para o estado original do momento da marcação. Este mecanismo é útil para o professor tomar uma linha de raciocínio na montagem do mapa, mesmo sabendo que é equivocada. Posteriormente, ele demonstra o equívoco à turma, e retorna o mapa para o estado que tinha antes de progredir naquela vertente.

4.1.4 - Distribuição Automática dos Resultados

A ferramenta deve ser capaz de, finada a elaboração do mapa geral (vide capítulo 3), distribuir os resultados para os alunos de forma fácil e rápida.

4.1.5 - Histórico das Aulas

Uma base de dados deverá ser mantida pela ferramenta, permitindo ao professor compor um histórico das atividades realizadas na preparação, apresentação e avaliação dos mapas de conceito no decorrer da aplicação do padrão Aula em Mapa de Conceitos. Dentre as informações constantes deste histórico destacam-se:

- Preparação
  - Conceitos distribuídos.
  - Mini-mapas produzidos pelos alunos
  - Erros dos mini-mapas
- Aula de montagem do mapa geral
  - Conceitos abordados
  - Turma em que aplicada
  - Data
  - Mapa geral resultante

Este histórico será composto para monitorar as atividades didáticas desenvolvidas sob o auxílio da ferramenta e, baseado nos dados coletados, gerar avaliações e estatísticas do desempenho dos alunos no decorrer das aulas.

4.2 Descrição da Ferramenta

A ferramenta que criamos está estruturada em módulos, cujas funcionalidades se combinam para atender aos requisitos levantados nas experiências com o PP Aula em Mapa de Conceitos. De acordo com a especificação apresentada a seguir, professor e alunos podem desenvolver as atividades que lhes concernem na aplicação do padrão de forma prática, organizada e, conseqüentemente, mais produtiva.

Apesar da absoluta obediência às orientações determinadas pelo padrão, a ferramenta tem como característica marcante a flexibilidade, que permitirá ao professor uma maior versatilidade no desempenho das suas atividades pedagógicas. Esta flexibilidade se apresenta através de várias opções de configuração e de mecanismos para controle do andamento das tarefas em sala de aula.

Os quatro módulos exibidos na figura 4.1 compõem a ferramenta que propomos, e cada um deles responde por uma das atividades contidas no padrão. Nós denominamos a ferramenta com o nome de **Mapa Fácil (MF)**, em alusão ao objetivo de facilitar a utilização do padrão Aula em Mapa de Conceitos.

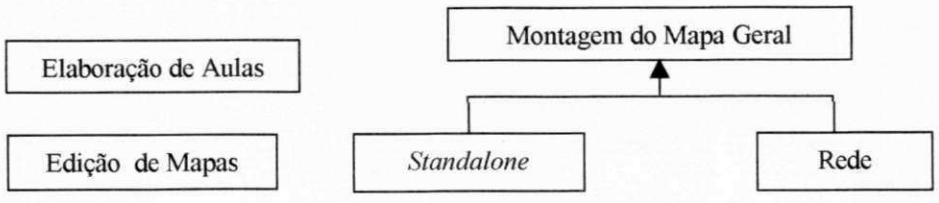


Figura 4.1 – Estrutura da ferramenta Mapa Fácil

As funcionalidades básicas dos módulos são descritas a seguir:

Elaboração de Aulas	- Suporta as atividades de distribuição de conceitos.
Edição de Mapas	- Permite a criação e edição de mapas de conceitos.
Montagem do Mapa Geral	- Responsável pela composição do mapa geral, resultante da compilação dos mapas elaborados pelos alunos .
Standalone	- Suporta a elaboração do mapa geral em uma sala com apenas um computador. As atividades desenvolvidas por este módulos são acompanhadas através da projeção da tela operada pelo professor em um *datashow*.
Rede	- Apresenta a mesma funcionalidade do módulo *Standalone*, sendo que os alunos interagem com o professor através de uma rede de computadores. As partes do módulo em rede estão conectadas, ajudando a comunicação professor - alunos e auxiliando a montagem do mapa.

Mais detalhes das funcionalidades de cada um dos módulos serão apresentadas no decorrer de uma aula hipotética, que utilizaremos como exemplo da aplicação do padrão com o auxílio da ferramenta. O nosso exemplo é de uma aula introdutória que será apresentada à turma do curso de Programação Orientada a Objetos, formada por quatro alunos.

4.2.1 Distribuindo os Conceitos

Para realizar a distribuição dos conceitos, o professor utiliza o módulo **Elaboração de Aulas**. Este módulo automatiza a distribuição de conceitos, de acordo com a interface apresentada na figura 4.4, abaixo:

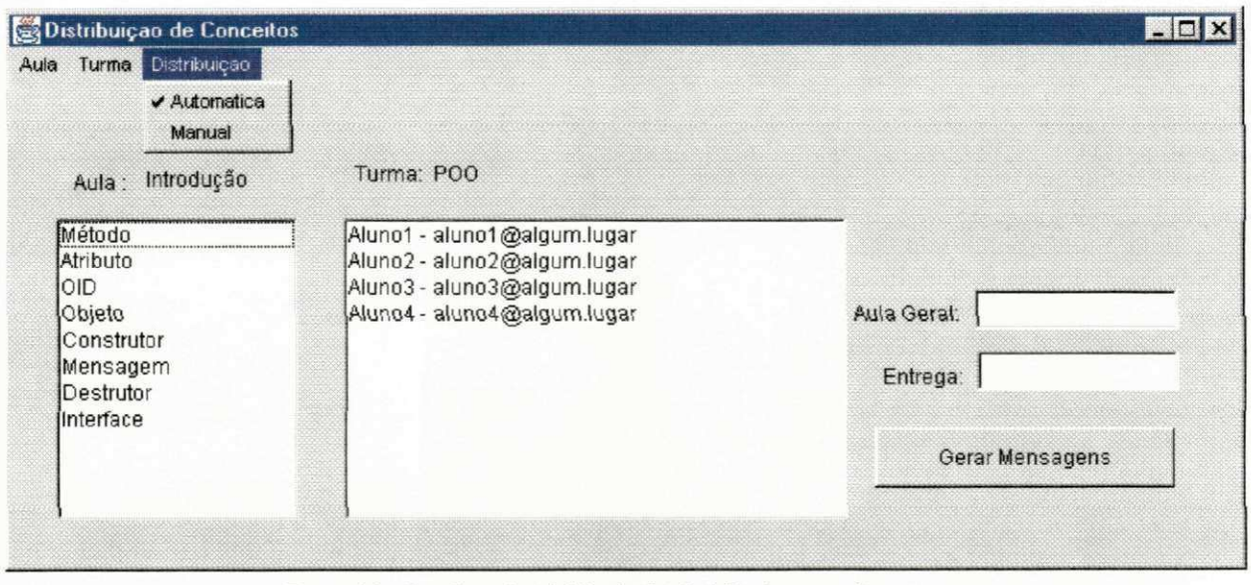


Figura 4.2 – Interface do módulo de distribuição dos conceitos

Acessando opções de menu, o professor seleciona uma turma e a aula que será apresentada àquela turma. O cadastro e edição de aulas e turmas também são atividades suportadas por este módulo. Turma consiste numa lista de alunos, identificados pelo nomes e o endereços eletrônicos correspondentes, enquanto que Aula é uma lista de conceitos, geralmente relacionados, que são abordados nas atividades do padrão pedagógico. A aula a ser apresentada no nosso exemplo é uma introdução ao paradigma OO.

Feita a seleção de Turma e Aula, o professor procede a distribuição dos conceitos, de acordo com as seguintes opções:

1 - Manual : neste caso, o professor seleciona o conceito com um clique de *mouse* e arrasta até o nome do aluno que o receberá. A ferramenta destaca o conceito selecionado, indicando que já foi remetido a algum dos alunos. O aluno que recebe o conceito também tem seu nome destacado, indicando que já recebeu um dos conceitos daquela aula.

2 – Aleatória: de acordo com esta opção, a ferramenta faz a distribuição automática e aleatória dos conceitos para os alunos. Para tanto, o professor precisa indicar os seguintes parâmetros:

Parâmetro 1 : número máximo de conceitos por aluno;

Parâmetro 2 : número de vezes que cada conceito pode ser distribuído.

A ferramenta Mapa Fácil (MF) faz a distribuição respeitando os parâmetros e tratando para que um aluno não receba um mesmo conceito mais de uma vez. Antes disso, a ferramenta verifica a consistência dos parâmetros fornecidos pelo professor. Caso os valores impossibilitem a distribuição (Parâmetro 2 igual a zero, por exemplo), a MF sinaliza o erro e aborta a distribuição.

Para a nossa aula, o professor selecionou a opção 2 (aleatória), atribuindo 3 como número máximo de conceitos por aluno e 3 como o número de vezes que um conceito pode ser distribuído, resultando na seguinte distribuição:

Aluno 1 : Método, Construtor, Objeto.

Aluno 2 : Atributo, Mensagem, OID.

Aluno 3 : OID, Destrutor, Atributo.

Aluno 4 : Objeto, Interface, Destrutor.

Terminada a distribuição, o professor preenche os dados referentes ao dia em que a aula de elaboração do mapa geral (prevista pelo padrão) será realizada e a data de entrega das pesquisas que cada aluno realizará sobre os conceitos recebidos. Ao pressionar o botão <Gerar>, a ferramenta monta um mapa para cada aluno, contendo os conceitos selecionados para ele. Em seguida estes mapas são anexados a mensagens contendo as informações fornecidas e, automaticamente, a ferramenta envia cada uma das mensagens para os alunos correspondentes. Em instantes, salvo algum problema de transmissão, o Aluno 1 (por exemplo) receberá via *e-mail* uma mensagem contendo:

- as datas informadas pelo professor no momento da distribuição dos conceitos.

- um arquivo anexado.

O arquivo anexado contém um mapa formado apenas pelos conceitos **Método**, **Construtor** e **Objeto** (sem definições e relações).

4.2.2 Elaboração dos Mini-Mapas

Seguindo a sequência de atividades contidas no Padrão Pedagógico Aula em Mapa de Conceitos, o próximo passo é cada aluno compor o seu mini-mapa. Este mini-mapa contém as definições dos conceitos recebidos e as relações entre tais. Esta etapa do padrão é suportado pela ferramenta Mapa Fácil através do módulo **Edição de Mapas** (Figura 4.3). As funcionalidades deste módulo serão descritas através da elaboração do mapa de conceitos do Aluno 1 (método, construtor, objeto) da nossa aula exemplo.

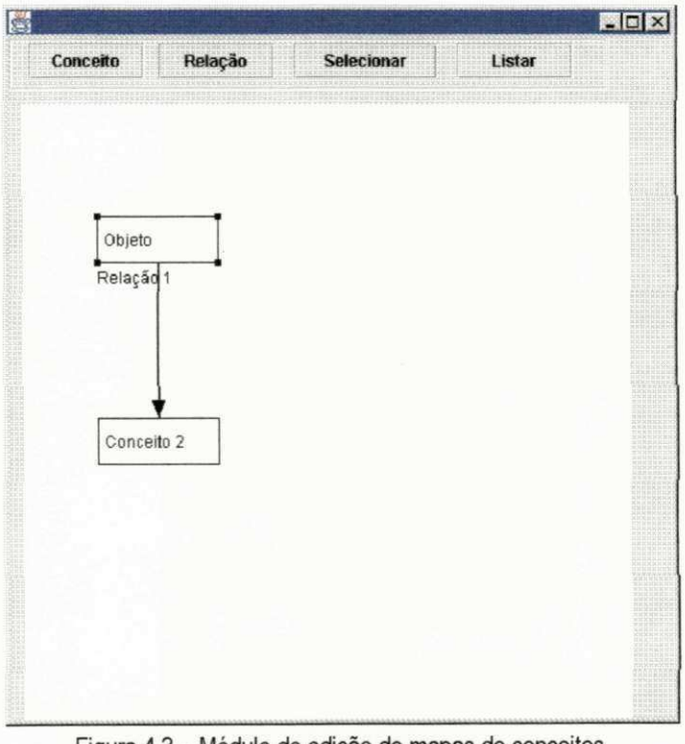


Figura 4.3 – Módulo de edição de mapas de conceitos

4.2.2.1 - Inserindo um Conceito

Para realizar esta operação de inserção de um conceito no mapa em construção, deve-se proceder da seguinte maneira:

Clicar sobre o botão <CONCEITO>.

Clicar na região de desenho do Mapa de Conceitos, na posição onde deseja que o conceito seja inserido. A ferramenta exibe a tela da figura 4.3.

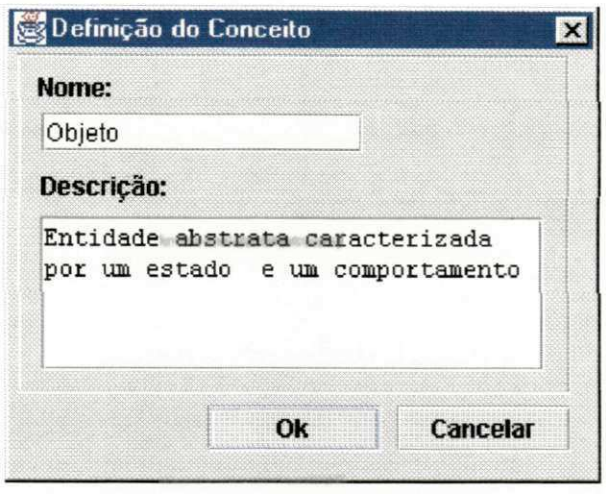


Figura 4.4 – Tela de Inserção de conceito.

No campo **Título,** deve-se digitar o rótulo que será exibido no retângulo que representa este conceito no mapa.

No campo **Definição,** o aluno digita a definição pesquisada para aquele conceito (esta definição não é exibida no mapa). Em seguida, o botão <OK> é clicado e o mapa é novamente exibido na tela inicial.

4.2.2.2 - Inserindo uma Relação

A exemplo da inserção de um conceito, a operação de inserção de uma relação é bastante simples. Ela se dá em quatro etapas, descritas a seguir:

- Clicar sobre o botão <RELAÇÃO>.
- Clicar sobre conceito origem.
- Mover o *mouse* até o destino.
- Clicar sobre o conceito destino.

Após este clique de *mouse*, a tela da figura 4.5 é apresentada. No exemplo abaixo (figura 4.6) o aluno clicou sobre o conceito **Construtor** e arrastou o *mouse* até o conceito **Método**

Figura 4.5 - Tela de inserção de uma relação.

No campo Título, o aluno digita a relação encontrada em sua pesquisa. Este título aparecerá como rótulo na reta que une os dois conceitos.

Caso o aluno, no decorrer de sua pesquisa, tenha encontrado mais que uma relação entre os conceitos **Construtor** e **Método**, deverá digitá-los no campo Outros Títulos.

Após confirmar a operação, com um clique no botão <OK>, a tela da figura 4.6 é exibida.

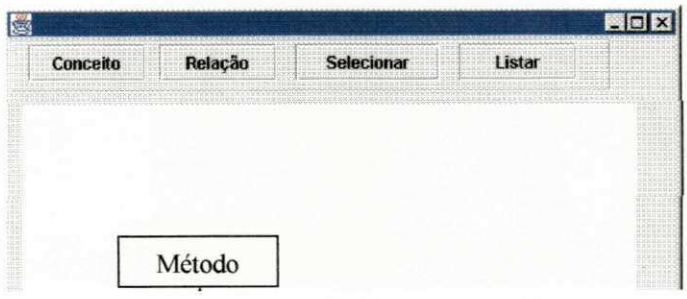


Figura 4.6 - Resultado da inserção de uma relação.

4.2.2.3 - Excluindo uma Relação

- Com um clique, selecionar a relação.
- Pressionar a tecla <DEL>.

A aplicação exibe uma mensagem pedindo confirmação da operação. Caso confirmada, a relação é excluída.

4.2.2.4 - Excluindo um Conceito

- Clicar sobre o Conceito, selecionando-o.
- Pressionar a tecla <DEL>.

A aplicação, após confirmação, exclui o conceito e todas as relações que partem dele ou chegam a ele.

4.2.2.5 - Editando um Objeto (Conceito/Relação)

A operação para edição de um objeto inserido no mapa é a mesma, tanto para conceitos como para relações:

- Fazer um clique duplo sobre o objeto.

Os dados são exibidos em uma tela igual a que foi usada para inserção deste objeto (figura 4.7 para conceitos, figura 4.8 para relações).

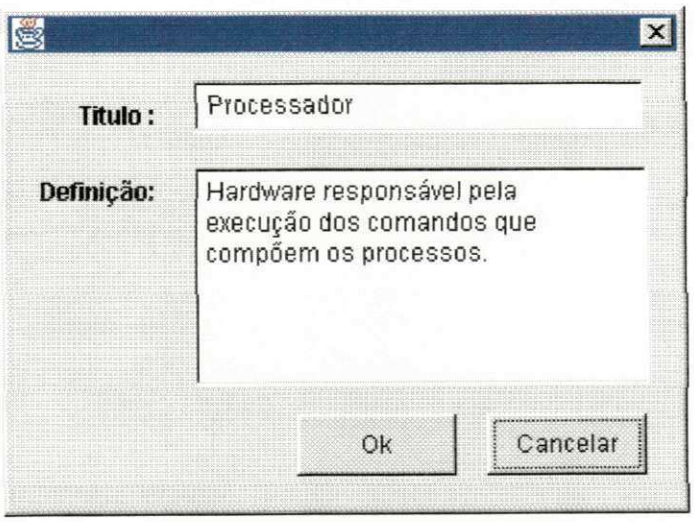


Figura 4.7 – Edição de um conceito.

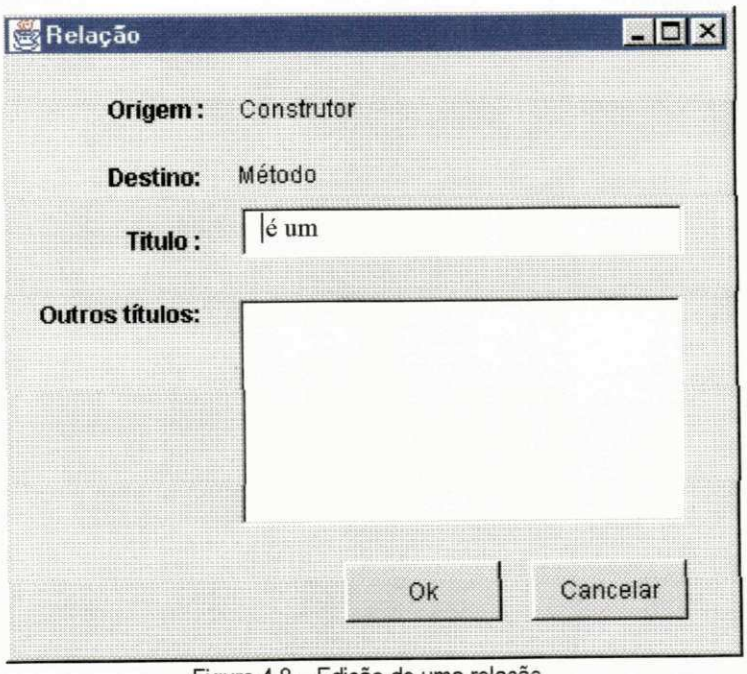


Figura 4.8 – Edição de uma relação.

- Modificar as informações desejadas e clicar em <OK>.
- Caso o aluno desista da modificação realizada, clicar no botão <CANCELAR>.

4.2.2.6 - Reaproveitando um Mapa Gravado

A ferramenta permite que o aluno aproveite as relações e os conceitos contidos em um mapa gravado na elaboração de um novo mapa. O mecanismo utilizado para prover esta funcionalidade é idêntico ao já bastante disseminado nas ferramentas que suportam algum tipo de edição gráfica como MS Paint™, MS Word™, Corel Draw™, etc.

Suponha que o aluno deseje aproveitar as relações e conceitos contidos no mapa gravado em disco com o nome de "OO.map". Para isso ele deve proceder os seguintes passos:

- selecionar no menu Arquivo a opção Abrir

  A tela padrão de abertura de arquivo é exibida, e nela o aluno navegará ao diretório/pasta que contem o arquivo "objeto.map".
- selecionar o arquivo e pressionar no botão Abrir

  O mapa "objeto.map" é exibido na área Mapa de Conceitos.
- selecionar (com <SHIFT> + clique de *mouse*) os conceitos e relações que deseja aproveitar para um novo mapa
- copiar os conceitos/relações selecionados (<CTRL> + C)
- selecionar no menu Arquivos a opção Novo Mapa
- clicar sobre a área Mapa de Conceitos
- colar os conceitos/relações (<CTRL> + V)

4.2.3 – Compondo o Mapa Geral

- *Definições*

Como já descrito no capítulo 3, após cada aluno elaborar as definições para seus conceitos e montar seus mini-mapas de conceitos, um nova aula é realizada para a montagem do mapa geral. A realização desta aula se dá auxiliada pelo módulo de **Montagem do Mapa Geral,** que descreveremos a seguir. Antes, entretanto, apresentaremos alguns conceitos que auxiliarão no entendimento das operações realizadas com a ferramenta:

Conceito corrente – é o conceito alvo das participações dos aluno. De acordo com as orientações do Padrão Pedagógico implementado pela ferramenta, o primeiro conceito corrente é determinado arbitrariamente pelo pro-

fessor. A cada rodada de participação dos alunos, quando o professor torna a perguntar : "Alguém tem, em seu mapa, um conceito relacionado com **Conceito X**?". Isto significa que **Conceito X** é o conceito corrente.

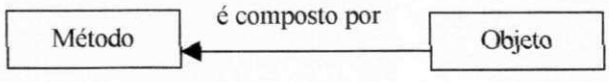
Sugestão – é a dupla formada pelo conceito que o aluno indica como relacionado com o **conceito corrente** e a relação que os une.

Sentença - é a formulação composta por **Conceito origem** + rótulo da relação + **Conceito destino**. Como ilustrado no exemplo abaixo, uma relação pode gerar várias sentenças.

- **Exemplo:**

O professor pergunta: "Alguém tem, em seu mapa, um conceito relacionado com **Objeto**?"

Neste instante o Aluno F, que tem a vez de participar, verifica que em seu mapa aparece a relação:



Baseado nesta relação, o Aluno F sugere o conceito **Método,** juntamente com a relação:

é composto por

Embora a relação exiba apenas um rótulo ("é um"), ela pode conter outros não exibidos como "contém"," é formado pôr", etc. No exemplo acima, de acordo com as definições, temos:

Conceito Corrente - Objeto.

Sugestão - Método + *é composto por* ←

Sentenças - **Objeto** é composto por **Método.**

Objeto contém **Método.**

Objeto é formado por **Método** .

Conceito origem e

Conceito destino - na sentença "**Objeto** é composto por **Método**", temos:

Objeto – conceito origem.

Método – conceito destino.

De posse destas definições, segue nossa aula hipotética para a última fase de suas atividades : compilação + correção dos mapas individuais para composição do mapa geral.

A ferramenta Mapa Fácil suporta duas modalidades de aulas geral (figuras 4.9 e 4.10) para composição do mapa determinada pelo padrão. A modalidade é aplicada de acordo com os recursos disponíveis no ambiente de ensino.

4.2.3.1 - Modalidade 1 : Sala com Apenas um Computador

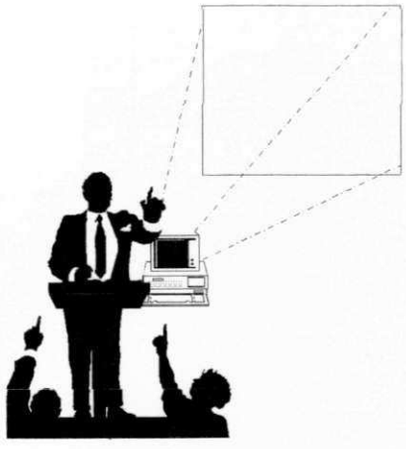


Figura 4.9 – Ambiente com apenas um computador

Para esta modalidade, o professor compila as sugestões apresentadas pelo alunos através do Módulo *Standalone* (figura 4.10).

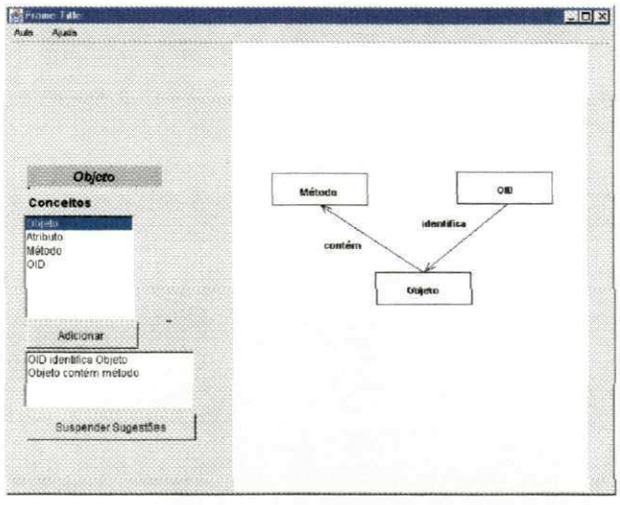
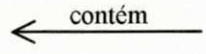


Figura 4.10 – Tela principal do módulo standalone.

Na aula em que se baseia a nossa descrição, o professor selecionou o conceito **Objeto** como o primeiro conceito corrente. Esta seleção foi feita através de uma opção do menu *pop up*, acionado quando o botão direito do *mouse* é clicado sobre um dos conceitos da lista exibida no quadro **Conceitos**.

Em seguida, o Aluno 1 apresentou a sua sugestão: **Método** + 

Para registrar esta sugestão, o professor procede os seguintes passos:

1. Clica sobre o conceito **Método;**
2. \ Digita a palavra "contém" na caixa de texto RELAÇÃO;
3. Clica no botão 

e desta forma dá origem a sentença:

Objeto contém **Método.**

O processo de geração de sentenças se repete até que todos os alunos tenham suas sugestões devidamente registradas pelo professor. Feito isto, o professor analisará cada sentença formada para selecionar as que farão parte do mapa de conceitos geral.

A seleção da sentença para o mapa geral se dá através de um clique sobre a sentença e outro sobre o espaço designado para o mapa geral, na posição onde deseja-se que o novo conceito seja inserido. Quando o professor clica numa sentença e clica sobre o mapa geral, a ferramenta executa uma das seguintes ações:

1 - Insere um conceito e uma relação: quando o conceito destino da sentença ainda não estiver no mapa;

2 - Insere uma relação: quando o conceito destino da sentença já está no mapa geral, mas até então não estava relacionado como o conceito corrente, que sempre é o conceito origem das sentenças catalogadas;

3 - Insere mais um rótulo na relação: quando o conceito origem e o conceito destino da sentença já estão relacionados no mapa geral, mas não apresentam a relação contida na sentença selecionada.

Uma outra forma de inserir um conceito no mapa geral é clicando na lista de conceitos e clicando no mapa, na posição escolhida. Esta funcionalidade é importante para os casos em que a lista de conceitos, distribuída entre os alunos, contém algum conceito que não se relaciona com nenhum dos demais. Isto significa que este conceito aparecerá isolado no mapa geral, não apresentando nenhuma relação que o conecte aos outros. No exemplo da figura 4.11, o conceito **Java** foi selecionado da lista de conceitos da aula e, com um clique sobre a posição em que é mostrado, foi inserido no mapa.

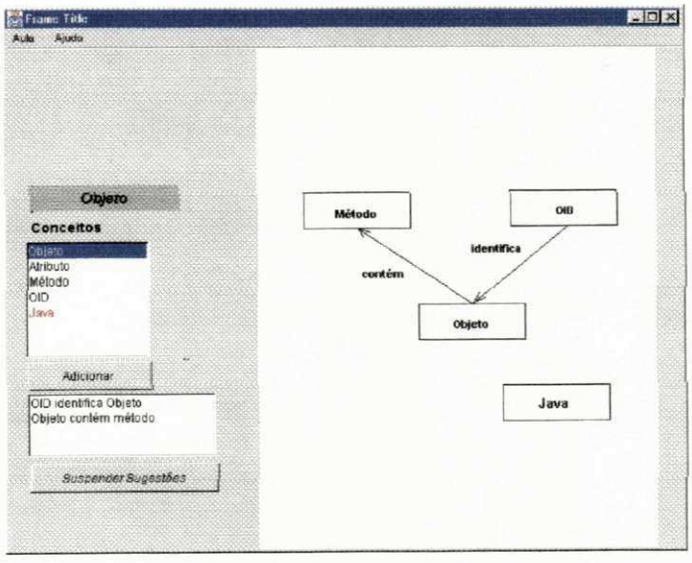


Figura 4.11 – Inserção de um conceito da lista direto para o mapa.

4.2.3.1.1 - Encerrando a Aula

Ao terminar a composição do Mapa Geral, o professor grava o mapa completo e remete aos alunos, através de um processo idêntico àquele que foi utilizado para a distribuição dos conceitos.

Outra prática sugerida para esta modalidade de aula é a reunião de todos os mini-mapas compostos pelos alunos e distribuição dentre eles, para que desta forma cada aluno tenha acesso a todos os mini-mapas elaborados. Desta forma, a quantidade de informação que os alunos levarão (em arquivos) da aula é uma grande fração de tudo aquilo que foi apresentado em sala, tanto quando se definiu cada conceito tratado como quando se discutiu as relações sobre eles. Antes de distribuir os mini-mapas para todos os alunos, o professor deve corrigi-los.

4.2.3.2 - Modalidade 2 : Sala com Vários Computadores em Rede.

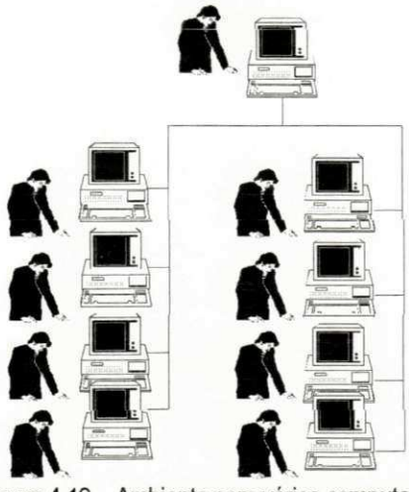


Figura 4.12 – Ambiente com vários computadores.

O que difere esta modalidade da primeira é a forma que os alunos enviam as suas sugestões para o professor. Enquanto que na primeira o professor seleciona o conceito corrente e, a cada sugestão oral dos alunos, digita a relação correspondente, gerando as sentenças, aqui as sentenças serão geradas automaticamente pelo envio das sugestões dos alunos via rede (figura 4.12). Cabe ao professor apenas selecionar as sentenças que farão parte do mapa geral.

A tela da aplicação que executa na máquina operada pelo professor (figura 4.13) é semelhante à apresentada na modalidade 1, distinguindo apenas pela ausência dos controles para cadastro de sentenças.

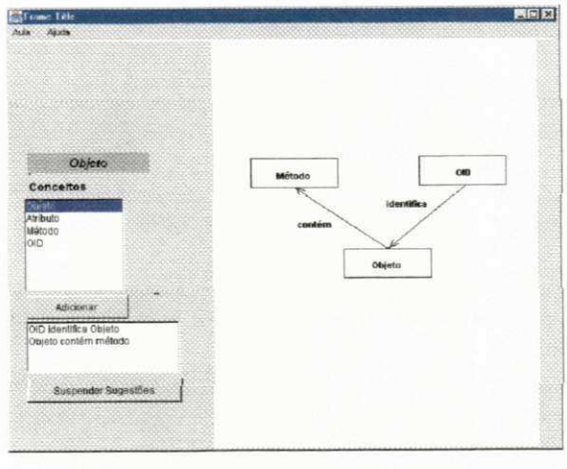


Figura 4.13 – Aplicação do professor.

Cada aluno(ou pequeno grupo de alunos) executa um aplicação que exibe o mini-

A aplicação do aluno (figura 4.14) exibe dois mapas de conceitos: o local, previamente elaborado por ele, e o mapa geral. Este último é um espelho do mapa montado pela aplicação do professor e é atualizado a cada passo do processo de montagem.

O professor dispõe de um botão para interromper a coleta de sugestões, que é utilizado para ele não dividir com a ferramenta o foco das atenções da turma. Este botão deve ser acionado quando o professor desejar explicar mais demoradamente determinado conceito (ou relação entre conceitos).

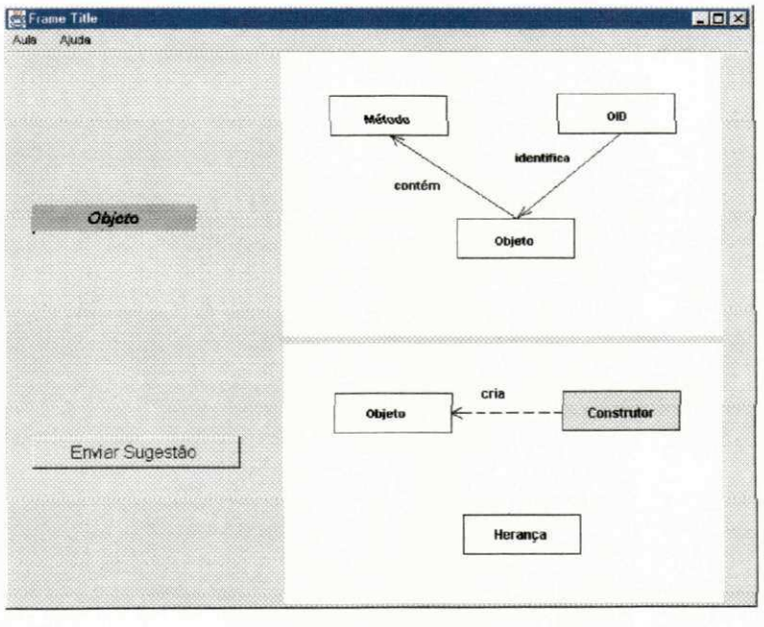


Figura 4.14 – Tela do aluno na modalidade em rede.

Quando o professor desabilita o envio de sugestões, a aplicação do aluno é sinalizada por uma marcação de sinal vermelho em sua tela, seguida da desativação do botão <ENVIAR>.

O aluno envia os conceitos através de uma operação simples de clique no conceito a ser sugerido, seguido de clique no botão enviar. O processamento desencadeado por esta operação depende da relação existente entre o conceito selecionado e o conceito corrente.

- Primeiro caso (figura 4.15) : o conceito selecionado está ligado (tem uma relação) com o conceito corrente.

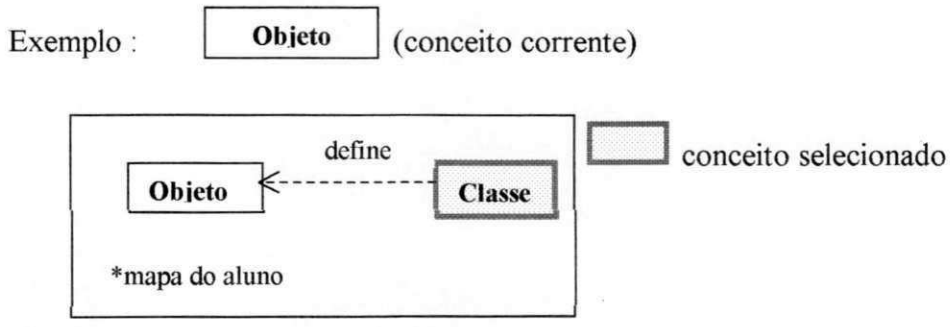


Figura 4.15 – Sugestão de conceito relacionado com o conceito corrente.

Nesta caso a ferramenta insere todas as sentenças geradas por esta relação (**Objeto** define **Classe,** e outras, caso a relação contenha mais rótulos além do exibido) nas lista de sentenças da aplicação do professor.

- Segundo caso (figura 4.16) : o conceito selecionado não apresenta relação com o conceito corrente no mapa do aluno.

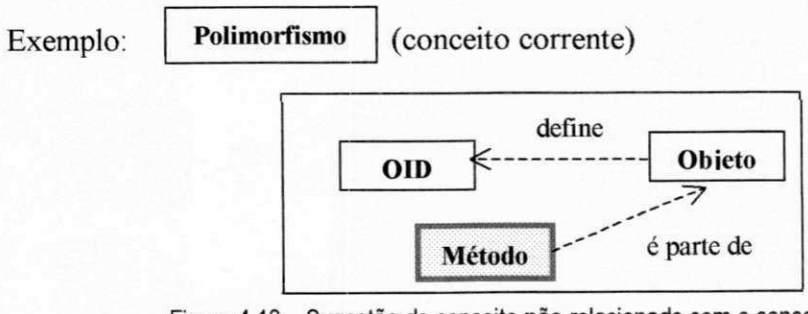


Figura 4.16 – Sugestão de conceito **não** relacionado com o conceito corrente.

Neste caso, ao clicar no botão <Enviar> a ferramenta solicita que o aluno digite a relação em uma caixa de diálogo (figura 4.17).

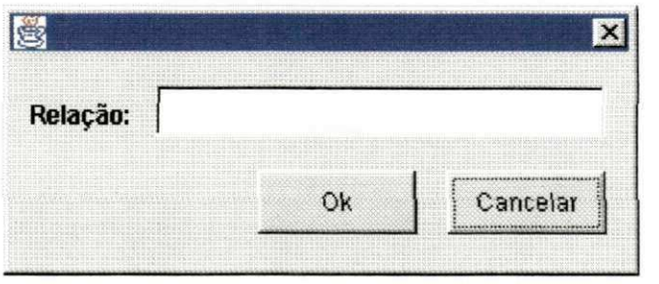


Figura 4.17 - Caixa de diálogo para digitação da relação sugerida.

Após a confirmação, a sugestão composta por conceito selecionado + relação digitada + conceito corrente , é inserida na lista de sugestões da aplicação do professor.

O processo de coleta de sugestões se processa até que o professor sinalize e interrompa o recebimento, seja pela necessidade de explicar a seleção das sentenças geradas pelas sugestões dos alunos, seja por ter finalizado o mapa geral.

Para que o professor possa distribuir o mapa geral construído para os alunos participantes da aula, a ferramenta disponibiliza uma opção de menu <Distribuir Mapa geral> . Esta opção abre a lista de alunos participantes da aula, de onde o professor selecionará as os alunos receptores (todos, geralmente), e confirmará o envio no botão <Enviar Mapa > (figura 4.18)

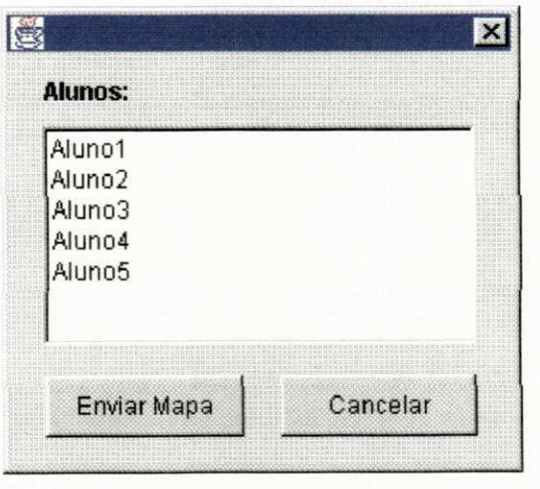


Figura 4.18 : Distribuição do mapa de conceitos geral.

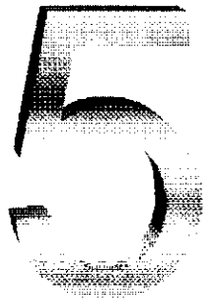
Essa distribuição do mapa geral encerra as atividades da aula utilizando o Padrão Pedagógico Aula em Mapa de Conceito, com o suporte da ferramenta Mapa Fácil.

4.3 - Aspectos de implementação

A ferramenta Mapa Fácil foi implementada parcialmente em um prótotipo. Este protótipo apresenta um subconjunto das funcionalidades que apresentamos no decorrer deste capítulo. A seleção das características implementadas é baseada na importância que cada uma delas tem para o desenrolar das atividades do Padrão Pedagógico e, como conseqüência disto, temos um software capaz de mostrar as potencialidades da ferramenta Mapa Fácil para o melhor andamento das aulas. Os resultados obtidos na utilização do protótipo (vide capítulo 5) atestam esta afirmação, e demonstram que os módulos implementados são suficientes para os objetivos do trabalho. O Anexo 1 apresenta o estado de implementação em que se encontra cada funcionalidade especificada para a ferramenta Mapa Fácil.

Utilizando técnicas de programação orientada a objetos, mais especificamente componentização de software, implementamos todos os módulos da ferramenta em linguagem Java™. A tecnologia de componentes Java Swing™ foi explorada tanto na composição dos elementos gráficos (conceitos e relações) como na construção da interface de navegação da ferramenta (botões, menus, janelas, etc.). O mecanismo *RMI – Remote Method Invocation* (Invocação Remota de Método), implementado pela linguagem utilizada, foi de fundamental utilidade para a implementação do módulo **Montagem do Mapa Geral em Rede**. Através deste mecanismo, podemos elaborar um modelo eficiente de cooperação entre as aplicações dos alunos e a aplicação do professor. O Anexo 2 apresenta o diagrama de classes que compõem a estrutura de todos os módulos da ferramenta Mapa Fácil, enquanto que o Anexo 3 apresenta o código fonte dos componentes Java ™ construídos para suportar as funcionalidades das aplicações.

A seguir, descrevemos as experiências com a ferramenta Mapa Fácil suportando o Padrão Pedagógico Aula em Mapa de Conceitos.



Experiências em Sala de Aula com a Ferramenta Mapa Fácil

A avaliação que realizamos e descrevemos neste capítulo tem seu foco em um aspecto fundamental : a eficácia da ferramenta Mapa Fácil no combate aos problemas identificados na fase de levantamento de requisitos. Outros aspectos que contribuem para a qualidade do software [Ghezi91] foram relevados nos nossos experimentos.

A exemplo da fase de experiências com Padrões Pedagógicos, descrita no capítulo 2, nós elaboramos um planejamento de ações para experimentar e avaliar a eficácia da ferramenta que criamos. Mais uma vez, a decisão de desenvolver os experimentos de forma estruturada e pré-definida foi o motivo que demandou a elaboração deste planejamento.

O planejamento a que nos referimos acima é composto por quatro fases, realizadas seqüencialmente:

1. Experiências em sala de aula.
   Realização de aulas com uso da ferramentas.
   Realização de aulas **sem** uso da ferramentas.
2. Entrevistas junto aos participantes.
3. Sumarização e apresentação dos resultados.
4. Análise dos resultados obtidos.

Estas fases serão explicadas a seguir e, junto às explicações, nós descreveremos como os nossos experimentos se passaram em cada uma das etapas planejadas.

5.1 Realizando as Experiências em Sala de Aula

Esta fase segue os mesmos passos apresentados na seção 2.1 deste documento. A exemplo dos experimentos tratados no capítulo 2, a realização dos experimentos em sala demanda uma definição prévia do universo de atuação. Pelas mesmas razões que apresentamos na seção citada acima, a maior amplitude e heterogeneidade deste universo contribuirão para a qualidade dos resultados.

Vale aqui também o critério da disponibilidade e interesse dos professores como fator determinante na inclusão ou não da disciplina no universo das experiências.

A faculdade Ruy Barbosa [www.frb.br] foi novamente o ambiente de realização dos nosso experimentos. Apesar de termos encontrado menor disponibilidade por parte dos professores que na primeira fase de experiências em sala, conseguimos mais uma vez atuar em disciplinas de cursos, professores, turmas, assuntos e turnos diferentes.

Para experimentar a ferramenta Mapa Fácil, foram dadas aulas nas seguintes disciplinas:

Curso de Ciências da Computação

Linguagem Técnica 2 – duas turmas.

Paradigmas de Linguagem de Programação – duas turmas.

Sistemas Operacionais – duas turmas

Curso de Processamento de Dados

Sistemas Operacionais – duas turmas

Nestas aulas foram envolvidas 190 pessoas, das quais:

4 professores.

186 alunos.

De acordo com a metodologia que utilizamos, o próximo passo após a definição do universo de atuação das experiências é agendar aulas em cada turma das disciplinas selecionadas. Estas aulas passam pelas seguintes fases.

Aula utilizando o padrão nas condições originais.

Aula utilizando o padrão com o auxílio da ferramenta.

No nosso trabalho, como as atividades do padrão suportado pela ferramenta Mapa Fácil se passam em duas aulas, cada um das turmas passou por 2 aulas nesta fase de experimentos.

- Uma aula de montagem do mapa geral **com** o auxílio da ferramenta *standalone*.
- Uma aula de montagem do mapa geral **com** o auxílio da ferramenta em de rede.

Não foi necessário utilizar uma aula para distribuição de conceitos com a ferramenta, já que esta atividade se processa via *e-mails* enviados do professor para os alunos. As aulas com o PP Aula em Mapa de Conceitos sem a ferramenta foram realizadas na fase de levantamento de requisitos, descrita no capítulo 2.

O período necessário para a realização destas aulas depende da quantidade de disciplinas participantes. No nosso trabalho, as aulas levaram o períodos de duas semanas, onde foram realizadas 16 aulas das quatro disciplinas envolvidas.

Terminadas as experiências com a ferramenta, o plano de ações que elaboramos nos leva a fase de coleta de resultados. As pessoas que tomaram parte das aulas são a fonte desta coleta, e ela deve ser feita através de entrevistas, que descreveremos a seguir.

5.2 Entrevistas com os Participantes dos Experimentos

Para a fase de entrevistas com as pessoas (professores e alunos) participantes das experiências com a ferramenta, definimos duas modalidades de entrevistas.

- **Estimulada.**

  Nesta modalidade, a entrevista é conduzida pelo mesmo roteiro de perguntas presente nos formulários utilizados na fase de levantamento de requisitos (figuras 2.1 e 2.2). Desta forma, podemos fazer um paralelo entre as respostas dadas na primeira fase de entrevistas (sem ferramenta), e as respostas obtidas após a utilização da ferramenta. A forma de contabilização dos resultados destas respostas está descrita na seção seguinte (5.3).

- **Espontânea.**

  Nesta modalidade o entrevistado discursa livremente sobre a experiência que realizou com a ferramenta. Este tipo de entrevista tem como objetivo principal destacar as características mais marcantes que os usuários identificaram na ferramenta experimentada.

  Outro benefício desta modalidade é a possibilidade de identificar algum aspecto importante no funcionamento da ferramenta, que tenha fugido do roteiro de perguntas constantes nos formulários utilizados.

No nosso trabalho estas duas modalidades de entrevista foram adotadas num período de uma semana, abrangendo um total de 27 pessoas, sendo:

4 professores.

23 alunos.

Os alunos foram selecionados aleatoriamente e em quantidades igualitárias nas oito turmas envolvidas (4 de cada turma). Por alguns problemas relacionados a horário e indisponibilidade dos alunos no período agendado para as entrevistas, nem todos os 32 alunos selecionados compareceram. Esta ausência reduziu o universo de alunos de 32 (selecionados) para 23 (entrevistados). A primeira rodada de entrevistas adotou a modalidade 2 (espontânea) enquanto que a segunda e última utilizou o método direcionado, descrito acima.

Encerrada a fase de entrevistas, o próximo passo é a coleta e apresentação dos resultados, que demostrarão o desempenho da ferramenta utilizada na solução dos problemas que se propunha a sanar. Esta fase é descrita na seção a seguir.

5.3 Sumarização e Apresentação dos Resultados Obtidos

Em nosso planejamento, estabelecemos métodos distintos para compor e apresentar os resultados de cada uma das modalidades de entrevistas.

5.3.1 - Resultados das Entrevistas Estimuladas

Para a modalidade estimulada, o método utilizado busca deixar o mais evidente possível a atuação da ferramenta nos problemas que motivaram o desenvolvimento de suas funcionalidades.

De acordo com esse método, os resultados são apresentados através de uma relação entre as respostas recebidas e as dificuldades enumeradas nas experiências para levantamento de requisitos. Para cada requisito identificado na primeira fase de experiências com o padrão (sem o uso de ferramentas), nós construímos uma tabela com as colunas descritas no exemplo a seguir:

Dificuldade Encontrada	**RA – fase 1**	**RP – fase 1**	**RA – fase 2**	**RP – fase 2**
Como escolher o primeiro conceito corrente.	5	83 %	0	0

Onde :

Dificuldade encontrada se refere aos problemas encontrados na utilização do padrão, na primeira fase de experiências. Estes problemas formam a base dos requisitos que a ferramenta Mapa Fácil busca atender.

Fase 1 – Rodada de entrevistas realizadas para levantamento de requisitos, após o primeiro período de experiências com o padrão Aula em Mapa de Conceitos.

Fase 2 – Rodada de entrevistas realizadas após as aulas que utilizaram a ferramenta Mapa Fácil.

RA – Resultado Absoluto. Indica quantas vezes aquele requisito foi identificado nas entrevistas.

RP – Resultado Percentual. Indica o percentual de ocorrência daquele requisito, em relação ao total de entrevistas realizadas.

Em nosso exemplo, temos os seguintes resultados:

Na primeira fase de entrevistas, 5 professores alegaram dificuldade na escolha do primeiro conceito corrente (vide definição no capítulo 4), o que representa 83% dos professores entrevistados. Já na segunda fase, nenhum professor alegou este problema (0%).

Demonstraremos a seguir os resultados levantados para a modalidade **espontânea**, após o período de entrevistas. Os resultados estão agrupados por cada um dos quesitos dos formulários 2.1 (utilizado nas entrevistas com professores) e 2.2 (utilizado nas entrevistas com os alunos).

5.3.1.1 – Resultado da Pesquisa Estimulada com Professores – Formulário 2.1.

Formulário 2.1. – Quesito 1 : Elaboração da lista de conceitos.

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Reutilização das listas elaboradas.	2	33%	0	0%
Falta de algum tipo de identificação para a lista como data, assunto, disciplina, etc.	1	17%	0	0%

Formulário 2.1. – Quesito 2: Distribuição dos conceitos.

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Demanda muito tempo de aula, ou alguma preparação prévia.	3	50%	0	0%
Tumultua a aula quando a turma é grande, e se precisa formar pequenos grupos.	6	100%	0%	0%
Gera discussões sobre a quantidade de conceitos recebidos, ou sobre a complexidade do conceito.	4	66%	1	25%

Formulário 2.1. – Quesito 3, letra a: Coleta das sugestões dos alunos.

- **Modalidade *Standalone***

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Gera tumulto na sala.	3	50%	1	25%
Um pequeno grupo de alunos monopoliza a participação.	6	100%	4	100%
Muitas sugestões tornam o registro delas impraticável.	5	83%	1	25%
É cansativo anotar no quadro, ou fazer o desenho equivalente a cada sugestão.	2	33%	0	0%
Requer muito esforço para atentar às sugestões, criticá-las e manter a ordem na turma.	4	66%	1	25%
Alunos mais tímidos não participam.	6	100%	4	100%
É constrangedor refutar as sugestões erradas.	1	17%	0	0%

Formulário 2.2. – Quesito 3, letra a: Coleta das sugestões dos alunos.

- **Modalidade em Rede**

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Gera tumulto na sala.	3	50%	0	0%
Um pequeno grupo de alunos monopoliza a participação.	6	100%	1	25%
Muitas sugestões tornam o registro delas impraticável.	5	83%	0	0%
É cansativo anotar no quadro, ou fazer o desenho equivalente a cada sugestão.	2	33%	0	0%
Requer muito esforço para atentar às sugestões, criticá-las e manter a ordem na turma.	4	66%	0	0%
Alunos mais tímidos não participam.	6	100%	0	0%
É constrangedor refutar as sugestões erradas.	1	17%	0	0%

Formulário 2.2. – Quesito 3, letra b: Desenho do mapa geral.

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Requer alguma habilidade de desenho que o professor não tem.	3	50%	0	0%
É desgastante quando a aula flui dinamicamente.	6	100%	0	0%
O quadro não oferece o espaço necessário.	5	83%	1	25
Não há espaço para registrar vários rótulos de uma mesma relação.	2	33/%	0	0%
A partir de um certo ponto, fica impraticável modificar o mapa.	6	100%	0	0%
O mapa fica quase ininteligível.	3	50%	1	25%

5.3.1.2 – Resultado da Pesquisa Estimulada com Alunos – Formulário 2.2.

Formulário 2.2. – Quesito 1 : Elaboração do mini-mapa.

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Falta de estímulo.	4	21%	1	0%

Formulário 2.2. – Quesito 2, letra a : Apresentação das sugestões.

- **Modalidade *Standalone*.**

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Vários alunos disputando a vez de falar.	11	58%	9	39%
O professor muda o conceito muito rapidamente.	0	0%	2	9%
O professor atenta para a sugestões de um pequeno grupo de alunos.	1	5%	0	0%

Formulário 2.2. – Quesito 2, letra a : Apresentação das sugestões.

- **Modalidade em Rede.**

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Vários alunos disputando a vez de falar.	11	58%	0	0%
O professor muda o conceito muito rapidamente.	0	0%	0	0%
O professor atenta para a sugestões de um pequeno grupo de alunos.	1	5%	0	0%

Formulário 2.2. – Quesito 2, letra b : Registro das informações apresentadas pelos demais alunos.

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Registro das sugestões dadas por outros alunos.	5	22%	7	37%
Registro das definições apresentadas por outros alunos	11	48%	0	0%

Formulário 2.2. – Quesito 3, letra c: Registro do mapa geral.

Dificuldade Encontrada	RA – fase 1	RP – fase 1	RA – fase 2	RP – fase 2
Requer alguma habilidade de desenho que o aluno não tem.	5	22%	0	0%
É difícil de entender o rótulo das relações.	14	61%	0	0%
É difícil de entender o rótulo dos conceitos.	6	83%	0	0%
O mapa é muito grande.	4	26%	0	0%
Pouco tempo para desenhar.	18	78%	0	0%

5.3.2 - Resultados das Entrevistas Espontâneas.

Para acompanhamento das respostas dadas nesta modalidade de entrevista, elaboramos dois formulário que atribuem conceitos a atuação da ferramenta nas aulas de experiência. Estes formulários (exibidos na figuras 5.1 e 5.2, respectivamente) seguem a seqüência de atividades do padrão Aula em Mapa de Conceitos, a exemplo dos formulários das figuras 2.1 e 2.1.

Acompanhamento das Entrevistas Espontâneas

(atividades do professor)

1 - Elaboração da lista de conceitos.
(a) Muito mais fácil.
(b) Mais fácil.
(c) Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

2 - Distribuição dos conceitos.
(a) Muito mais fácil.
(b) Mais fácil.
(c) Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

3 - Aula de montagem do mapa geral.

3.1 - Coleta das participações dos alunos.
(a) Muito mais fácil.
(b) Mais fácil.
(c) Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

3.2 - Desenho do mapa geral.
(a) Muito mais fácil.
(b) Mais fácil.
(c) Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

Figura 5.1 – Formulário para acompanhamento das entrevistas espontâneas com os professores.

As entrevistas com os alunos foram monitoradas com a utilização do formulário figura 5.2, apresentado abaixo.

Acompanhamento das Entrevistas Espontâneas (atividades dos alunos)

1 - Recepção dos conceitos.

(d) Muito mais fácil.
(e) Mais fácil .
(f) Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

2 - Elaboração do mini-mapa.

(a)Muito mais fácil.
(b)Mais fácil .
(c)Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

3 - Aula de montagem do mapa geral.

a - Apresentação de sugestões.

(a)Muito mais fácil .
(b)Mais fácil .
(c)Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

b - Registro de informações.

(a)Muito mais fácil .
(b)Mais fácil .
(c)Equivalente.
(d) Mais difícil.
(e) Muito mais difícil.
(f) Não citou.

Figura 5.2 – Formulário para acompanhamento das entrevistas espontâneas com os alunos.

Após a realização das entrevistas, buscamos uma forma didática e clara de apresentação dos resultados que obtivemos para esta modalidade de entrevista. A forma escolhida foi a apresentação dos valores obtidos em tabelas que reflitam cada um dos quesitos dos formulários aplicados. As tabelas 5.1 (professores) e 5.2 (alunos) trazem esses dados. Assim como os resultados obtidos nas entrevistas estimuladas, estes dados são base de boa parte das conclusões do nosso trabalho. As letras de **a** até **f** representam a classificação no quesito de acordo com os critérios do formulário (muito mais fácil, mais fácil, etc.). Os valores numéricos indicam a quantidade de professores/alunos que atribuíram aquela classificação ao quesito avaliado. Os valores expressos em percentagem indicam a proporção destes valores em relação ao número total de entrevistados (quatro para os professores e 23 para os alunos).

- **Professores.**

Quesito avaliado	a	a%	b	b%	c	c%	d	d%	e	e%	f	f%
Elaboração da lista de conceitos.	0	0%	0	0%	2	50%	0	0%	0	0%	2	50%
Distribuição dos conceitos.	4	100%	0	0%	0	0%	0	0%	0	0%	0	0%
Coleta das participações dos Alunos – *Standalone.*	0	0%	1	25%	1	25%	0	0%	0	0%	2	50%
Coleta das participações dos Alunos – Rede.	3	75%	0	0%	0	0%	0	0%	0	0%	1	25%
Desenho do Mapa.	1	25%	1	25%	0	0%	0	0%	0	0%	0	0%

Tabela 5.1 – Resultado das entrevistas espontâneas com os professores.

- **Alunos.**

Quesito avaliado	a	a%	b	b%	c	c%	d	d%	e	e%	f	f%
Elaboração do mini-mapa.	5	22%	3	13%	0	0%	0	0%	0	0%	15	65%
Apresentação das sugestões – *Standalone.*	0	0%	0	0%	1	4%	1	4%	0	0%	22	96%
Apresentação das sugestões – Rede.	14	61%	2	9%	0	0%	2	9%	0	0%	5	22%
Registro do mapa geral.	11	48%	2	9%	0	0%	0	0%	0	0%	10	43%
Registro das informações apresentadas pelos demais alunos.	4	17%	1	4%	0	0%	0	0%	0	0%	18	70%

Tabela 5.1 – Resultado das entrevistas espontâneas com os alunos.

Considerado os resultados apresentados, tanto das entrevistas estimuladas quanto nas entrevistas espontâneas, já temos os dados necessários para avaliar o desempenho da ferramenta Mapa Fácil no suporte às aulas sob o Padrão Pedagógico Aula em Mapa de Conceitos. Como previa nosso planejamento, esta análise é o próximo passo deste fase de experiências com a ferramenta.

5.4 – Análise dos Resultados Obtidos

Esta é a etapa final do planejamento que elaboramos para experimentar a ferramenta Mapa Fácil. Nela e apresentamos algumas conclusões que os dados nas tabelas da seção 5.3 apresentam. A análise é apresentada na ordem em que os dados foram descritos.

5.4.1 – Entrevistas Estimuladas

Os resultados obtidos nas entrevistas com os professores apontam uma diminuição na ocorrência de dificuldades na primeira atividade do PP (elaboração da lista de conceitos). Neste caso, o maior serviço fornecido pela ferramenta é a capacidade de reutilização das listas elaboradas.

Quanto à distribuição dos conceitos, os resultados apontam que a ferramenta economiza o tempo de aula utilizado para esta atividade. Além disso as queixas dos alunos referentes à justiça da distribuição também diminuem.

A ferramenta Mapa Fácil tem desempenho importante na atividade de coleta das sugestões apresentadas pelos alunos. Na modalidade *standalone,* as dificuldades ainda aparecem em grande percentagem, exceto na maior agilidade do professor para registrar um grande número de sugestões. Por outro lado, na modalidade em rede, o desempenho da ferramenta é muito positivo, superando todas as dificuldades citadas pelos professores nas experiências com o PP sem a ferramenta.

A atividade de desenho do mapa geral foi fortemente favorecida com o uso da ferramenta. Menor esforço e habilidade por parte do professor, além da qualidade do desenho obtido, foram os aspectos de destaque na atuação da Mapa Fácil neste quesito.

Os dados levantados das entrevistas com alunos também apontam para a eficácia da ferramenta Mapa Fácil. Na atividade de elaboração de mini-mapas o uso da ferramenta Mapa Fácil estimulou a realização da tarefa.

Na aula de montagem do mapa geral, a ferramenta favoreceu a apresentação das sugestões dos alunos. A modalidade em rede apresentou um desempenho de destaque no

combate aos problemas da primeira fase de experiências. A única ressalva à atuação da ferramenta na apresentação das sugestões foi encontrada na modalidade *standalone*. Alguns alunos se queixaram da rapidez com que o professor muda o conceito corrente.

Outro serviço provido pela ferramenta Mapa Fácil é a capacidade de registrar uma maior fração do total de informações trocadas na aula. De acordo com os dados, a ferramenta facilita o registro de boa parte das informações passadas pelos alunos e pelo professor.

Por último, a ferramenta resolve completamente um problema dos mais incômodos enfrentado pelos alunos: o registro do mapa geral. Enquanto nas aulas convencionais eles costumam desistir do desenho pela dificuldade, nas aulas com a ferramenta esta dificuldade não é sequer citada.

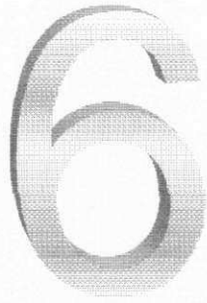
5.4.2 - Entrevistas Espontâneas

As entrevistas espontâneas com os professores indicam uma boa atuação da ferramenta nas atividades do PP Aula em Mapa de Conceitos. O uso da ferramenta facilitou os trabalhos em todas as etapas do padrão, exceto na montagem da lista do conceito (equivalente). O destaque é a coleta das sugestões dos alunos na modalidade em rede. Neste quesito a ferramenta foi extremamente eficaz no apoio ao professor.

Quanto aos alunos, as entrevistas espontâneas apontam para um desempenho satisfatório da ferramenta. Mais uma vez , de acordo com as respostas, todas as atividades foram facilitadas com o emprego da ferramenta Mapa Fácil. A única ressalva é a atividade de apresentação de sugestões, na modalidade *standalone*. Os destaques positivos foram a melhora das atividades de apresentação de sugestões na (modalidade em rede) e o registro do mapa geral.

5.4.3 – Análise geral

Os resultados apresentados nas entrevistas com professores e alunos foram satisfatórios, e indicam que os objetivos do nosso trabalho foram atingidos. Em praticamente todas as atividades do padrão, a ferramenta Mapa Fácil facilita o desempenho das tarefas e elimina as dificuldades encontradas na primeira fase de experimentos (sem padrão).



Conclusões e Trabalhos Futuros

6.1 – Sumário

Padrões Pedagógicos são formatos de aula bem estruturados, inspirados nas idéias dos Padrões (*Patterns*) [Alex77] e Padrões de Design (*Design Pattens*) [Gamma95], que buscam facilitar a troca de experiências de sucesso entre educadores. O projeto que coordenou a colaboração entre professores da comunidade de Orientação a Objetos teve como resultado a catalogação de dezenas destes padrões. Todo o esforço deste projeto se

direciona para a catalogação dos padrões, sem nenhuma referência a utilização de ferramentas como forma de apoio aos Padrões Pedagógicos.

Neste trabalho, desenvolvemos um estudo sobre os Padrões Pedagógicos, catalogados pelo projeto citado acima. Contando com a colaboração de alguns professores dos cursos de graduação em Ciência da Computação e Processamento de Dados da Faculdade Ruy Barbosa [www.frb.br], nós realizamos uma série de experiências com os padrões em sala de aula. Estas experiências tiveram como objetivo selecionar um padrão para receber uma ferramenta de software que auxiliasse a sua utilização.

O resultado das experiências apontaram o Padrão Pedagógico (PP) *Class Concept Map* (Aula em Mapa de Conceitos) como o mais adequando para o qual desenvolvêssemos uma ferramenta de apoio.

Após uma nova fase de experiências com o padrão Aula em Mapa de Conceitos, nós levantamos um conjunto de requisitos que este padrão demanda de um software que o suporte. Baseados neste conjunto de requisitos, especificamos a ferramenta Mapa Fácil, que se apresenta em duas versões : *standalone* e em rede.

Utilizando a linguagem de programação Java™, implementamos um protótipo para cada uma das versões da ferramenta Mapa Fácil e os experimentamos em sala de aula. Desta fase de experimentos coletamos resultados para comparamos com a primeira fase de experiências com o padrão, realizada sem o auxílio da ferramenta que criamos.

6.2 – Conclusões

Das nossas experiências com os padrões catalogados, concluímos que estes padrões são adaptáveis para aulas sobre várias naturezas de assunto. Concluímos também que a estrutura usada para descrever os Padrões Pedagógicos é clara e didática, permitindo a sua compreensão e aplicação.

Da seleção realizada entre os PP experimentados, concluímos que o padrão Aula em Mapa de Conceito tem alta versatilidade e, por isso, aplica-se para um maior número de disciplinas/cursos. Esta versatilidade foi demonstrada na sua aplicação com sucesso em várias disciplinas.

Com base nos resultados obtidos na segunda fase dos experimentos, concluímos que a ferramenta Mapa Fácil facilita sensivelmente a aplicação do Padrão Pedagógico Aula em Mapa de Conceitos. A avaliação positiva da ferramenta usada nos leva a crer que a metodologia utilizada para sua elaboração é eficaz, e propicia a construção do software correto para o padrão. Como o padrão que abordamos em nossos experimentos apresenta os problemas que motivaram nosso trabalho, os nosso resultados indicam que ferramentas de software podem melhorar os Padrões Pedagógicos. Estes resultados indicam também que o procedimento utilizado na aplicação dos padrões é um bom caminho para detecção de suas limitações.

6.2 - Trabalhos Futuros

Como continuidade do trabalho que desenvolvemos, levantamos alguns aspectos a serem investigados, elaborados e experimentados. Dentre eles destacam-se:

- Experiências em sala de aula com outros Padrões Pedagógicos catalogados;
- Repetição do trabalho de especificação e implementação para outros Padrões Pedagógicos catalogados;
- Extensão da ferramenta Mapa Fácil para suportar aulas a distância, sob o padrão Aula em Mapa de Conceitos;
- Alteração do formato utilizado pelos Padrões Pedagógicos, incluindo seções que tratem de ferramentas de software;
- Elaboração, catalogação e utilização de novos Padrões Pedagógicos, que atuem em outras áreas além dos conceitos do paradigma de Orientação a Objetos.

Referências Bibliografia

[Alex77] Alexander, C., Sara Ishikawa, e Murray Silverstein. **A Pattern Language: Towns, Buildings, Construction**. Oxford University Press, New York, New York, 1977.

[Alex79] Alexander, C.. **The Timeless Way of Building**, Oxford University Press, , New York, New York, 1979.

[Barnet68] T. O. Barnet. **Modular Programing**. Information & System Press, Cambridge, 1968

[Brown90] .Brown, John S., **Toward a New Episemology for Learning**, In: Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education, Eds. C. rasson and G. Gauthier, Ablex Publishing, Norwood, NJ, 1990.

[Dahl72] O. Dahl. **Structure Programing** . Academy Press, New York, 1972

[Dansereau84] J. G. C, D. F. Dansereau, D. R. Cross, S. B. Reynolds. **Multirelational Semantic Maps**. Educational Psychology Review, 1(4), 1984, pp. 331-367.

[Ellis91] Ellis, C. A., Gibbs, S. J., Rein, G. L. **Groupware : Some issues and experiences.** Communications of ACM, 1991, 34(1), 38-58.

[Gamma95] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. **Design Patterns, Elements of Reusable Object-Oriented Software**. d, Addison-Wesley 1995.

[Gehringer95] Gehringer, Edward F. **OOA/OOD/OOP: What Programmers and**

Managers Believe We Should Teach. OOPSLA Educators' Symposium. Austin, Texas, outubro de 1995.

[Ghezzi91] Ghezzi, C., Jazayeri, M., Mandrioli, D. **Fundamentals of Software Engineering**. Praticce Hall, p. 17-36, Upper Sandle River, New Jersey, 1991

[Kirk98] Kirk, D. ***Design Patterns and ViewKit.*** MutifDeveloper On Line Magazine, 1998.

http://www.motifzone.com/tmd/articles/VK_Patterns/patterns.html

[Lilly96] Lilly, Susan, 'Patterns for Pedagogy' Object Magazine, Vol. 5(8), janeiro de 1996, pp. 93-96.

[Lipschutz93] Lipschutz, Seynour. **Probabilidade**. McGraw-Hill, São Paulo, 1993

[Reader94] Reader, W., & Hammond, N. (1994**). Computer-based tools to support learning from hypertext: Concept mapping tools and beyond. Computer Education**, 22(1/2), 99-106

[Thayer98] Thayer, Richard H. e Dorfmanm M. Software Requirements Engineering. Capítulo 2. segunda edição, McGraw-Hill, 1998.

Bibliografia

Thinking in Java	Bruce Eckel , Prentice Hall PTR, 1998
Java Networking Proraming	Merlin e Conrad Hughes, Micel Shoffner e Maria Winslow. Mannin Publications Co. 1997
Java in a Nut Shell Secound Edition	David Flanagan, O'Reilly & Associates 1997.
Patterns in Java	Mark Grand, John Willey Publications, 1998.
Distilling UML	Martin Fowler, Addison-Wesley, 1997.
JBuilder2 Developers Guide	Borland International. 1998.
JBuilder2 Bible	Erick Amstrong, IDG Book Worldwide, 1998.

Anexo 1

Estágio de Implementação do protótipo utilizado nas experiências com a ferramenta Mapa Fácil.

Legenda :

Não implementado	☹
Em fase de implementação	😐
Implementado	☺
Implementado e experimentado	[

- Módulo de Elaboração de Aulas

Funcionalidade	Estágio de implementação
Interface e estrutura de navegação.	[
Carga dos alunos de uma turma.	[
Composição das mensagens.	☺
Distribuição manual dos conceitos.	😐
Distribuição aleatória dos conceitos.	😐
Envio das mensagens.	☺

- Módulo de Elaboração de Mapas

Funcionalidade	Estágio de implementação
Interface e estrutura de navegação	[
Inserção de conceitos	[
Exclusão de conceitos	[
Edição de conceitos	[
Inserção de relações	[
Edição de relações	[
Exclusão de relações	[

Anexo 1 – Estágio de Implementação do Protótipo Utilizado nas Experiências com a Ferramenta Mapa Fácil

Gravação do mapa	[
Carga de um mapa gravado	[
Reaproveitamento de um mapa gravado	😐

- Módulo de Elaboração do Mapa Geral - Standalone

Funcionalidade	Estágio de implementação
Interface e estrutura de navegação.	[
Carga dos conceitos de uma aula.	[
Formação das sentenças.	[
Inserção das sentenças no mapa geral.	[
Gravação do mapa geral.	[
Inserção de um conceito diretamente da lista para o mapa.	[
Opção de marcar um estado do mapa, e retornar para ele após alguma modificações.	😐

- Módulo de Elaboração do Mapa Geral em Rede (Aluno).

Funcionalidade	Estágio de implementação
Interface e estrutura de navegação.	[
Sugestão de conceito relacionado com o conceito corrente.	[
Sugestão de conceito não relacionado pelo conceito corrente.	[
Espelho do mapa geral.	[

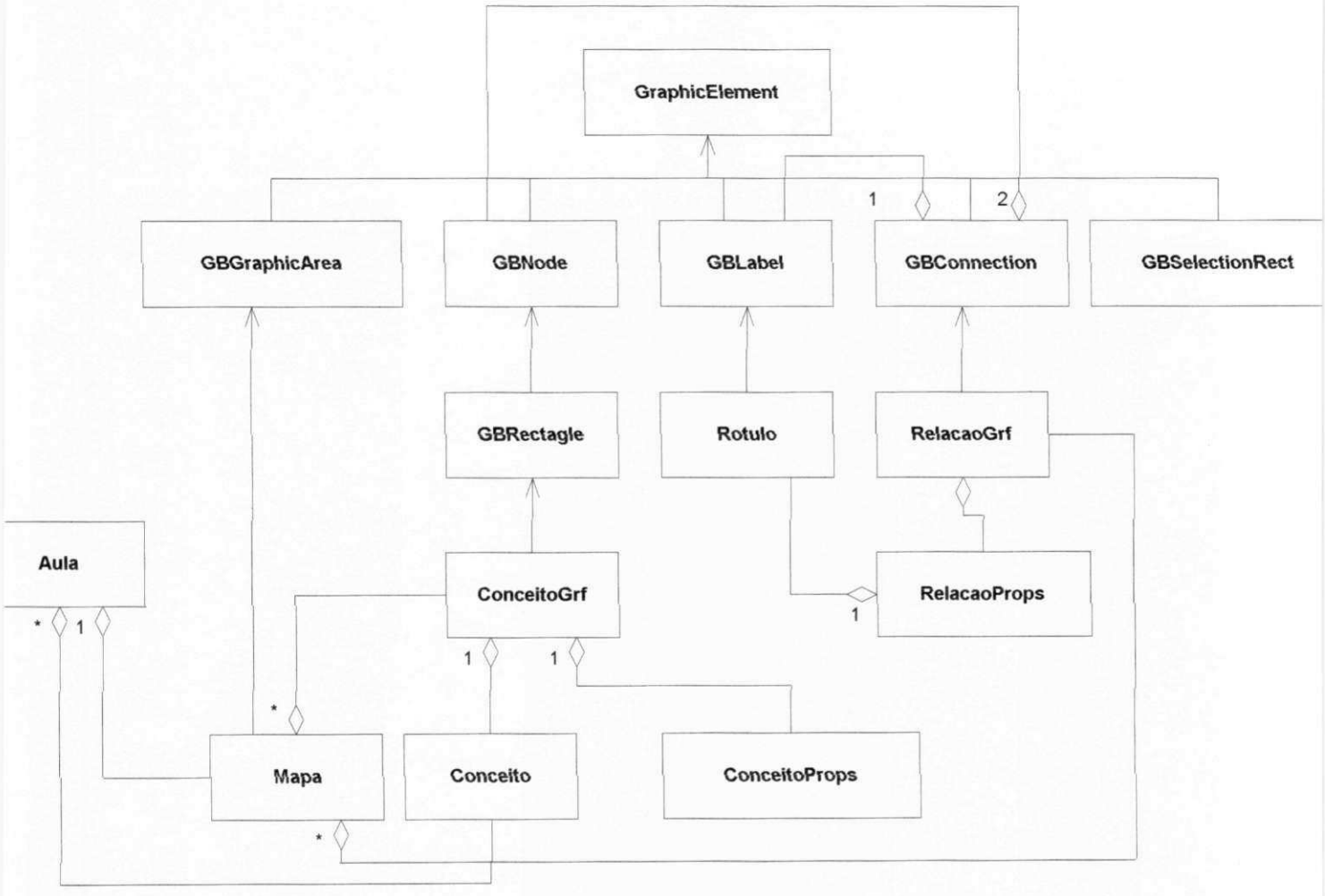
- Módulo de Elaboração do Mapa Geral em Rede (Professor).

Funcionalidade	Estágio de implementação
Interface e estrutura de navegação.	[
Recepção de sugestões.	[
Gravação do mapa geral.	[
Bloqueio das sugestões dos alunos.	😐
Opção de marcar um estado do mapa, e retornar para ele após alguma modificações.	😐
Distribuição do mapa geral.	☹
Manutenção da base de dados com o histórico das aulas.	☹

Anexo 2

Estrutura de Classes Utilizada na Implementação da Ferramenta Mapa Fácil

Anexo 2 – Estrutura de Classes Utilizada na Implementação da Ferramenta Mapa Fácil



Anexo 3

Código Fonte dos Componentes Utilizados na Implementação das Aplicações

```
package GraphicBricks;

import java.awt.event.*;

public class Conceito extends GBRectangle {
  public String nome;
  public String descricao;
  protected static ConceitoProps props;

  static {
    props = new ConceitoProps();
    setPropertiesDlg(props);
  }

  public Conceito() {
    super();
    nome = "Conceito";
    descricao = "";
    setCaption(nome);
  }

  protected void nodeMouseClicked(MouseEvent e) {
    super.nodeMouseClicked(e);
  }

  public void beforeShowProps() {
    props.jTextFieldNome.setText(nome);
    props.jTextAreaDescricao.setText(descricao);
  }

  public void afterShowProps() {
    if (props.getDlgResult() == props.DR_BUTTON_OK) {
      nome = props.jTextFieldNome.getText();
      descricao = props.jTextAreaDescricao.getText();
      setCaption(nome);
      repaint();
    }
  }
}
```

```
  }

  void this_componentShown(ComponentEvent e) {
    jTextFieldNome.requestFocus();
  }

  public void centerWindow() {
    Dimension screenSize;
    Dimension frameSize;

    screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    frameSize = getSize();

    if (frameSize.height > screenSize.height)
      frameSize.height = screenSize.height;
    if (frameSize.width > screenSize.width)
      frameSize.width = screenSize.width;
    setLocation((screenSize.width - frameSize.width) / 2, (screenS
ize.height - frameSize.height) / 2);
  }

  public void setVisible(boolean visible) {
    if (visible) {
      centerWindow();
    }
    super.setVisible(visible);
  }

  public int getDlgResult() {
    return dlgResult;
  }
}
```

```
    });
    this.setTitle("Definição do Conceito");
    jPanelConceito.setLayout(null);
    jPanelConceito.setBorder(BorderFactory.createEtchedBorder());
    jPanelConceito.setBackground(Color.lightGray);
    jPanelConceito.setBounds(new Rectangle(4, 3, 269, 165));
    jLabelNome.setText("Nome:");
    jLabelNome.setForeground(Color.black);
    jLabelNome.setBounds(new Rectangle(11, 9, 41, 15));
    jTextFieldNome.setBounds(new Rectangle(12, 29, 156, 19));
    jLabelDescricao.setText("Descrição:");
    jLabelDescricao.setForeground(Color.black);
    jLabelDescricao.setBounds(new Rectangle(12, 57, 64, 15));
    jTextAreaDescricao.setBounds(new Rectangle(12, 79, 244, 75));
    jTextAreaDescricao.setLineWrap(true);
    jButtonOk.setText("Ok");
    jButtonOk.setBounds(new Rectangle(96, 173, 85, 23));
    jButtonOk.addActionListener(new java.awt.event.ActionListener(
) {
      public void actionPerformed(ActionEvent e) {
        jButtonOk_actionPerformed(e);
      }
    });
    jButtonCancelar.setText("Cancelar");
    jButtonCancelar.setBounds(new Rectangle(188, 173, 85, 23));
    jButtonCancelar.addActionListener(new java.awt.event.ActionLis
tener() {
      public void actionPerformed(ActionEvent e) {
        jButtonCancelar_actionPerformed(e);
      }
    });
    this.getContentPane().add(jPanelConceito, null);
    jPanelConceito.add(jLabelNome, null);
    jPanelConceito.add(jTextFieldNome, null);
    jPanelConceito.add(jLabelDescricao, null);
    jPanelConceito.add(jTextAreaDescricao, null);
    this.getContentPane().add(jButtonOk, null);
    this.getContentPane().add(jButtonCancelar, null);
  }

  void jButtonOk_actionPerformed(ActionEvent e) {
    dlgResult = DR_BUTTON_OK;
    setVisible(false);
  }

  void jButtonCancelar_actionPerformed(ActionEvent e) {
    dlgResult = DR_BUTTON_CANCEL;
    setVisible(false);
```

```
//Title:        Your Product Name
//Version:
//Copyright:    Copyright (c) 1998
//Author:       Your Name
//Company:      Your Company
//Description:  Your description

package GraphicBricks;

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class ConceitoProps extends JDialog {
  public static final int DR_BUTTON_CANCEL = 0;
  public static final int DR_BUTTON_OK = 1;

  JPanel jPanelConceito = new JPanel();
  JLabel jLabelNome = new JLabel();
  JTextField jTextFieldNome = new JTextField();
  JLabel jLabelDescricao = new JLabel();
  JTextArea jTextAreaDescricao = new JTextArea();
  JButton jButtonOk = new JButton();
  JButton jButtonCancelar = new JButton();
  private int dlgResult;

  public ConceitoProps() {
    try  {
      jbInit();
    }
    catch (Exception e) {
      e.printStackTrace();
    }
  }

  private void jbInit() throws Exception {
    dlgResult = DR_BUTTON_CANCEL;
    setModal(true);
    this.setSize(287, 230);
    this.getContentPane().setLayout(null);
    this.setResizable(false);
    this.getContentPane().setBackground(Color.lightGray);
    this.addComponentListener(new java.awt.event.ComponentAdapter(
) {
      public void componentShown(ComponentEvent e) {
        this_componentShown(e);
      }
```

```
    drawSegment(gc, initialPoint, currentPoint);
    drawSegment(gc, currentPoint, finalPoint);
    currentPoint.x = pointSelected.x + e.getX();
    currentPoint.y = pointSelected.y + e.getY();
    drawSegment(gc, initialPoint, currentPoint);
    drawSegment(gc, currentPoint, finalPoint);
  }

  protected void rectSelectionReleased(MouseEvent e) {
    pointSelected.setLocation(currentPoint);
    updateSelectionRectsPos(idRectSelected);
    hasArrow = true;
    setVisibleSelectionRects(true);
    graphicArea.repaint();
    graphicArea.setState(GBGraphicArea.SELECTING);
  }

  public static void setElementsList(Vector newElementsList) {
    elementsList = newElementsList;
  }

  public void destroy() {
    label.destroy();
    label = null;
    initialNode.removeConnectionOut(this);
    finalNode.removeConnectionIn(this);
    elementsList.remove(this);
    graphicArea.graphicElementsList.remove(this);
  }
}
```

```
      }
      ((JPanel) graphicArea).add(selectionRect, 0);
      selectionRects.addElement(selectionRect);
    }
  }

  public void setVisibleSelectionRects(boolean visibleSelectionRec
ts) {
    int rectsCount;

    rectsCount = selectionRects.size();
    for (int cont = 0; cont < rectsCount; cont++) {
      ((GBSelectionRect) selectionRects.elementAt(cont)).setVisibl
e(visibleSelectionRects);
    }
  }

  public void updateSelectionRectsPos(int rectIndex) {
    Point point;
    int rectCenter;
    GBSelectionRect selectionRect;

    rectCenter = GBSelectionRect.RECT_SIZE/2;
    point = (Point) points.elementAt(rectIndex);
    selectionRect = (GBSelectionRect) selectionRects.elementAt(rec
tIndex);
    selectionRect.setLocation(point.x - rectCenter, point.y - rect
Center);
  }

  protected void rectSelectionPressed(MouseEvent e) {
    idRectSelected = selectionRects.indexOf(e.getComponent());
    pointSelected = (Point) points.elementAt(idRectSelected);
    currentPoint = new Point(pointSelected);
    initialPoint = (Point) points.elementAt(idRectSelected-1);
    finalPoint = (Point) points.elementAt(idRectSelected+1);
    hasArrow = false;
    setVisibleSelectionRects(false);
    graphicArea.setState(GBGraphicArea.MOVINING);
    graphicArea.requestFocus();
  }

  protected void rectSelectionDragged(MouseEvent e) {
    Graphics gc;

    gc = graphicArea.getGraphics();
    gc.setXORMode(Color.white);
    gc.setColor(Color.black);
```

```
  protected void beforeShowProps() {
  }

  protected void afterShowProps() {
  }

  public static void setPropertiesDlg(Window newPropertiesDialog)
{
    propertiesDialog = newPropertiesDialog;
  }

  public static Window getPropertiesDlg() {
    return propertiesDialog;
  }

  public void createSelectionsRects() {
    Point point;
    int pointsCount, rectCenter;
    GBSelectionRect selectionRect;

    rectCenter = GBSelectionRect.RECT_SIZE/2;
    pointsCount = points.size();
    for (int cont = 0; cont < pointsCount; cont++) {
      point = (Point) points.elementAt(cont);

      selectionRect = new GBSelectionRect();
      selectionRect.setBounds(point.x - rectCenter, point.y - rect
Center, GBSelectionRect.RECT_SIZE, GBSelectionRect.RECT_SIZE);
      if (cont != 0 && cont != pointsCount-1) {
        selectionRect.setCursor(new Cursor(Cursor.MOVE_CURSOR));
        selectionRect.addMouseListener(new java.awt.event.MouseAda
pter() {
          public void mousePressed(MouseEvent e) {
            rectSelectionPressed(e);
          }
        });
        selectionRect.addMouseListener(new java.awt.event.MouseAda
pter() {
          public void mouseReleased(MouseEvent e) {
            rectSelectionReleased(e);
          }
        });
        selectionRect.addMouseMotionListener(new java.awt.event.Mo
useMotionAdapter() {
          public void mouseDragged(MouseEvent e) {
            rectSelectionDragged(e);
          }
        });
```

```
      Ry = PontoFin.y - PontoIni.y;
      escalarP = ((double) (Hx * Rx + Hy * Ry)) / ((double) (Rx * R
x + Ry * Ry));
      Px = escalarP * Rx;
      Py = escalarP * Ry;
      distance = Math.sqrt(Hx * Hx + Hy * Hy - Px * Px - Py * Py);
      if (distance <= TOLERANCE)
         return true;
      else {
           distance = Hx * Hx + Hy * Hy - Px * Px - Py * Py;
           if (distance <= (TOLERANCE * TOLERANCE))
              return true;
           else
              return false;
      }
   }

   ////////////////////////////////////////////////
   // Verifica se um ponto pertence ao arco
   public GBGraphicElement containsPoint(Point point) {
     boolean Achou;
     int cont;
     Point PontoIni, PontoFin, PontoAux;

    Achou = false;
     cont = 0;
     while ((!Achou) && (cont < points.size()-1)) {
       PontoIni = (Point) points.elementAt(cont);
       PontoFin = (Point) points.elementAt(cont+1);
       if (PontoIni.x > PontoFin.x) {
         PontoAux = PontoIni;
         PontoIni = PontoFin;
         PontoFin = PontoAux;
       }
       if (PontoIni.y > PontoFin.y) { // Arco Subindo
         if ((point.x >= PontoIni.x) && (point.x <= PontoFin.x) &&
            (point.y <= PontoIni.y) && (point.y >= PontoFin.y))
           Achou = overSegment(point.x, point.y, cont);
       } else if ((point.x >= PontoIni.x) && (point.x <= PontoFin.x
) &&
                  (point.y >= PontoIni.y) && (point.y <= PontoFin.y)
)
                 Achou = overSegment(point.x, point.y, cont);
       cont = cont + 1; // Arco Descendo
     }
     return (Achou)?this:null;
   }
```

```
          PontoFinal.y = (int) Math.ceil(PontoFinal.y - 10*Math.si
n(teta));
//      }
      angulo1 = 2.79;     // 160 Graus -> 20 Graus
      angulo2 = 3.49;     // 200 Graus -> 20 Graus

      // Calcula os pontos da seta do arco
      PontoX[0] = (int) Math.ceil((d1*Math.cos(teta) + PontoFinal.x
 + 0.5));
      PontoY[0] = (int) Math.ceil((d1*Math.sin(teta) + PontoFinal.y
 + 0.5));
      PontoX[1] = (int) Math.ceil((d1*Math.cos(teta + angulo1) + Po
ntoX[0] + 0.5));
      PontoY[1] = (int) Math.ceil((d1*Math.sin(teta + angulo1) + Po
ntoY[0] + 0.5));
      PontoX[2] = (int) Math.ceil((d1*Math.cos(teta + angulo2) + Po
ntoX[0] + 0.5));
      PontoY[2] = (int) Math.ceil((d1*Math.sin(teta + angulo2) + Po
ntoY[0] + 0.5));

//      CorAtual = gc.getBackground();
//      ModoAtual = gc.getForeground();
      gc.setXORMode(Color.white);
      gc.setColor(Color.black);
//      gc.drawPolygon(PontoX, PontoY, 3);
      gc.fillPolygon(PontoX, PontoY, 3);
//       g.setColor(getBackground());
//       g.fillRect(1, 1, width-2, height-2);
//      CanvasForm.Brush.Color := CorAtual;
//      CanvasForm.Pen.Mode:= ModoAtual;
  }
*/
  ////////////////////////////////////////////////////////////////
/////
  // Verifica se o click do mouse foi próximo a um segmento do arc
o
  private boolean overSegment(int XMouse, int YMouse, int Indice)
{
      int Hx, Hy, Rx, Ry, TOLERANCE;
      double escalarP, Px, Py, distance;
      Point PontoIni, PontoFin;

      TOLERANCE = 4;
      PontoIni = (Point) points.elementAt(Indice);
      PontoFin = (Point) points.elementAt(Indice+1);
      Hx = PontoIni.x - XMouse;
      Hy = PontoIni.y - YMouse;
      Rx = PontoFin.x - PontoIni.x;
```

```
    pointX[2] = pointX[2] + PontoFinal.x;
    pointY[2] = pointY[2] + PontoFinal.y;

//      gc.setXORMode(Color.white);
    gc.fillPolygon(pointX, pointY, 3);
//     System.out.println("Coefiente Angular: " + coefAngular);
//     System.out.println("Teta: " + Math.toDegrees(teta));
//    System.out.println("X Arrow 2: " + xArrow2);
//     System.out.println("Y Arrow 2: " + yArrow2);
  }

/*
  ////////////////////////////
  // Desenha a seta do arco
  protected void drawArrow(Graphics gc) {
    int PontoX[];
    int PontoY[];
    Point PontoInicial, PontoFinal;
    double coefAngular, teta, d1, angulo1, angulo2;
    Color CorAtual;
    Color ModoAtual;

    PontoX = new int[3];
    PontoY = new int[3];

    PontoInicial = (Point) points.elementAt(points.size()-2);
    PontoFinal = (Point) points.elementAt(points.size()-1);
    if (PontoInicial.x == PontoFinal.x)
        if (PontoInicial.y < PontoFinal.y)
           teta = 1.57;    // Teta = 90 Graus
        else
           teta = 4.71;    // Teta = 270 Graus
     else {
          coefAngular = ((double) (PontoFinal.y - PontoInicial.y))
 /
                        ((double) (PontoFinal.x - PontoInicial.x))
;
          if (PontoInicial.x > PontoFinal.x)
             teta = Math.atan(coefAngular) + Math.PI;
          else if (PontoInicial.y <= PontoFinal.y)
               teta = Math.atan(coefAngular);
          else
               teta = Math.atan(coefAngular) + 2*Math.PI;
     }
//     if (FatorEscalaX == 1) {
          d1 = 11.49;         // Hipotenusa
          PontoFinal.x = (int) Math.ceil(PontoFinal.x - 10*Math.co
s(teta));
```

```
//  public static void drawArrow() {
    int pointX[], pointY[], xArrowIni, yArrowIni;
    Point PontoInicial, PontoFinal;
    double coefAngular, teta;

    pointX = new int[3];
    pointY = new int[3];

//    PontoInicial = new Point(200, 100); // Penultimo ponto do ar
co
//    PontoFinal = new Point(100, 100); // Ultimo ponto do arco
    teta = 0;
    coefAngular = 0;
    PontoInicial = (Point) points.elementAt(points.size()-2); // P
enultimo ponto do arco
    PontoFinal = (Point) points.elementAt(points.size()-1); // Ult
imo ponto do arco
    if (PontoInicial.x == PontoFinal.x)
        if (PontoInicial.y > PontoFinal.y)
           teta = 1.57;    // Teta = 90 Graus
        else
           teta = 4.71;    // Teta = 270 Graus
    else {
      coefAngular = ((double) (PontoFinal.y - PontoInicial.y)) /
                          ((double) (PontoFinal.x - PontoInicial.x
));
      if (PontoInicial.x < PontoFinal.x)
        teta = Math.atan(coefAngular) + Math.PI;
      else
              teta = Math.atan(coefAngular);
    }

    xArrowIni = 12; // X do Ponto da seta superior
    yArrowIni = 5; // Y do Ponto da seta superior
    pointX[0] = PontoFinal.x;
    pointY[0] = PontoFinal.y;
    pointX[1] = (int) Math.round(xArrowIni*Math.cos(teta) - yArrow
Ini*Math.sin(teta));
    pointY[1] = (int) Math.round(xArrowIni*Math.sin(teta) + yArrow
Ini*Math.cos(teta));
    pointX[1] = pointX[1] + PontoFinal.x;
    pointY[1] = pointY[1] + PontoFinal.y;
    xArrowIni = 12; // X do Ponto da seta inferior
    yArrowIni = -5; // Y do Ponto da seta inferior
    pointX[2] = (int) Math.round(xArrowIni*Math.cos(teta) - yArrow
Ini*Math.sin(teta));
    pointY[2] = (int) Math.round(xArrowIni*Math.sin(teta) + yArrow
Ini*Math.cos(teta));
```

```
int));
  }

  public void draw(Graphics gc) {
    gc.setColor(Color.black);
    int lastIndex;

//    gc = graphicArea.getGraphics();
//    gc.setXORMode(Color.white);
    lastIndex = points.size()-1;
    for (int cont = 0; cont < lastIndex; cont++) {
      drawSegment(gc, (Point) points.elementAt(cont), (Point) poin
ts.elementAt(cont+1));
    }
    if (hasArrow) {
      gc.setPaintMode();
      drawArrow(gc);
    }
  }

  public void drawSegment(Graphics gc, Point firstPoint, Point las
tPoint) {
    gc.drawLine(firstPoint.x, firstPoint.y, lastPoint.x, lastPoint
.y);
  }

  public void drawInitialSegment(Graphics gc) {
    Point initialPoint, finalPoint;

    initialPoint = (Point) points.elementAt(0);
    finalPoint = (Point) points.elementAt(1);
    drawSegment(gc, initialPoint, finalPoint);
  }

  public void drawFinalSegment(Graphics gc) {
    Point initialPoint, finalPoint;

    initialPoint = (Point) points.elementAt(points.size()-2);
    finalPoint = (Point) points.elementAt(points.size()-1);
    drawSegment(gc, initialPoint, finalPoint);
    if (hasArrow) {
      gc.setPaintMode();
      drawArrow(gc);
    }
  }
  ////////////////////////////
  // Desenha a seta do arco
  protected void drawArrow(Graphics gc) {
```

```
  public void setInitialNode(GBNode node) {
    initialNode = node;
    initialNode.addConnectionOut(this);
    points.addElement(node.getLocation());
  }

  public GBNode getInitialNode() {
    return initialNode;
  }

  public void setFinalNode(GBNode node) {
    finalNode = node;
    finalNode.addConnectionIn(this);
    points.addElement(node.getLocation());
    updateFirstPoint();
    updateLastPoint();
    initLabel();
    label.setVisible(true);
  }

  public GBNode getFinalNode() {
    return finalNode;
  }

  public void initLabel() {
      label.setGraphicArea(graphicArea);
      label.setSize(70, 25);
      label.setLocation(200, 200);
      graphicArea.add(label, null);
      label.createSelectionsRects();
  }

  public void updateFirstPoint() {
    Point firstPoint, secondPoint;

    firstPoint = (Point) points.elementAt(0);
    secondPoint = (Point) points.elementAt(1);
    firstPoint.setLocation(initialNode.getInterceptionPoint(second
Point));
  }

  public void updateLastPoint() {
    Point firstPoint, secondPoint;

    firstPoint = (Point) points.elementAt(points.size() - 1);
    secondPoint = (Point) points.elementAt(points.size() - 2);
    firstPoint.setLocation(finalNode.getInterceptionPoint(secondPo
```

```
package GraphicBricks;

import java.awt.*;
import java.awt.event.*;
import java.util.Vector;
import javax.swing.*;

public class GBConnection implements GBGraphicElement {
  protected GBGraphicArea graphicArea;
  protected Vector selectionRects;
  protected boolean deletable;
  public static Vector elementsList = null;
  protected static Window propertiesDialog = null;
  protected int idRectSelected;

  protected GBNode initialNode;
  protected GBNode finalNode;
  protected GBLabel label;
  protected Vector points;
  protected boolean hasArrow;
  protected Point pointSelected, currentPoint, initialPoint, final
Point;

  public GBConnection() {
    selectionRects = new Vector();
    idRectSelected = -1;
    deletable = true;

    hasArrow = true;
    initialNode = null;
    finalNode = null;
    points = new Vector();
    pointSelected = null;
    label = new GBLabel();
    label.setVisible(false);
  }

  public boolean isDeletable() {
    return deletable;
  }

  public void setGraphicArea(GBGraphicArea graphicArea) {
    this.graphicArea = graphicArea;
  }

  public void addPoint(Point newPoint) {
    points.addElement(newPoint);
  }
```

```
    super.paint(gc);
    drawConnections(gc);
  }

  private void drawConnections(Graphics gc) {
    int connectionCount;
    GBGraphicElement graphicElement;

    connectionCount = graphicElementsList.size();
    for (int cont = 0; cont < connectionCount; cont++) {
      graphicElement = (GBGraphicElement) graphicElementsList.elem
entAt(cont);
      if (graphicElement instanceof GBConnection)
        ((GBConnection) graphicElement).draw(gc);
    }
  }

  protected void graphicAreaKeyPressed(KeyEvent e) {
    if (e.getKeyCode() == KeyEvent.VK_DELETE) {
      if (currentElement != null) {
        if (currentElement.isDeletable()) {
          currentElement.destroy();
          setCurrentElement(null);
        }
      }
    }
  }
}
```

```
    catch (IllegalAccessException e) {
      e.printStackTrace();
      return;
    }
    catch (InstantiationException e) {
      e.printStackTrace();
      return;
    }
    newConnection.setGraphicArea(this);
  }

  protected void graphicAreaMouseClicked(MouseEvent e) {
    if (e.getClickCount() >= 2) {
      GBConnection connection;

      connection = onPressedConnection(e.getPoint());
      if (connection != null) {
        setCurrentElement(connection);
        connection.setVisibleSelectionRects(true);
        if (connection.propertiesDialog != null) {
          connection.beforeShowProps();
          connection.propertiesDialog.setVisible(true);
          connection.afterShowProps();
        }
      }
    }
  }

  private GBConnection onPressedConnection(Point mousePoint) {
    int elementsCount;
    GBGraphicElement graphicElement;
    GBConnection connection;

    elementsCount = graphicElementsList.size();
    for (int cont = 0; cont < elementsCount; cont++) {
      graphicElement = (GBGraphicElement) graphicElementsList.elem
entAt(cont);
      if (graphicElement instanceof GBConnection) {
        connection = (GBConnection) graphicElement;
        if (connection.containsPoint(mousePoint) != null) {
          return connection;
        }
      }
    }
    return null;
  }

  public void paint(Graphics gc) {
```

```
  public int getState() {
    return state;
  }

  public void setCurrentElement(GBGraphicElement currentElement) {
    if (this.currentElement != null)
      this.currentElement.setVisibleSelectionRects(false);
    this.currentElement = currentElement;
//    if (this.currentElement != null) {
//      this.currentElement.setVisibleSelectionRects(true);
//    }
  }

  public GBGraphicElement getCurrentElement() {
    return currentElement;
  }

  protected void setConnectionNodes(GBNode node) {
    if (initialNode == null) {
        initialNode = node;
        initialNode.setForeground(Color.red);
        createConnection();
        newConnection.setInitialNode(initialNode);
        initialPoint = new Point(-1, -1);
    } else if (node != initialNode) {
      finalNode = node;
      initialNode.setForeground(Color.black);
      newConnection.setFinalNode(finalNode);
      initialNode = null;
      finalNode = null;
      graphicElementsList.addElement(newConnection);
      GBConnection.elementsList.addElement(newConnection);
//      externalElementsList.addElement(newConnection);
      newConnection.createSelectionsRects();
//      connection.setVisibleSelectionRects(true);
      setCurrentElement(newConnection);
      newConnection.setVisibleSelectionRects(true);
      newConnection.draw(getGraphics());
      newConnection = null;
    }
  }

  protected void createConnection() {
    try {
      newConnection = (GBConnection) graphicElementClass.newInstan
ce();
    }
```

```
  }

  public void initNode(GBNode node, Point mousePosition) {
      node.setGraphicArea(this);
      node.setSize(90, 35);
      node.setLocation(mousePosition.x - (node.getWidth()/2), mous
ePosition.y - (node.getHeight()/2));
      add(node, null);
      node.createSelectionsRects();
      node.setVisibleSelectionRects(true);
  }

  private void createLabel(Point mousePosition) {
    GBLabel label;

    try {
      label = (GBLabel) graphicElementClass.newInstance();
    }
    catch (IllegalAccessException e) {
      e.printStackTrace();
      return;
    }
    catch (InstantiationException e) {
      e.printStackTrace();
      return;
    }

    initLabel(label, mousePosition);
    graphicElementsList.addElement(label);
    GBLabel.elementsList.addElement(label);
    setCurrentElement(label);
    label.setVisibleSelectionRects(true);
    initialPoint.setLocation(mousePosition);
  }

  public void initLabel(GBLabel label, Point mousePosition) {
      label.setGraphicArea(this);
      label.setSize(90, 35);
      label.setLocation(mousePosition.x - (label.getWidth()/2), mo
usePosition.y - (label.getHeight()/2));
      add(label, null);
      label.createSelectionsRects();
      label.setVisibleSelectionRects(true);
  }

  public void setState(int state) {
    this.state = state;
  }
```

```
    Point mousePoint;

    mousePoint = e.getPoint();
    initialPoint.setLocation(mousePoint);
    currentPoint.setLocation(mousePoint);
    if (state == CREATING) {
      if (graphicElementType == ET_NODE)
        createNode(mousePoint);
      else if (graphicElementType == ET_LABEL)
        createLabel(mousePoint);
      else if (graphicElementType == ET_CONNECTION)
        if (newConnection != null)
          newConnection.addPoint(new Point(mousePoint));
    } else if (state == SELECTING) {
      GBConnection connection;

      connection = onPressedConnection(mousePoint);
      if (connection != null) {
        setCurrentElement(connection);
        connection.setVisibleSelectionRects(true);
      }
    }
    requestFocus();
  }

  private void createNode(Point mousePosition) {
    GBNode node;

    try {
      node = (GBNode) graphicElementClass.newInstance();
    }
    catch (IllegalAccessException e) {
      e.printStackTrace();
      return;
    }
    catch (InstantiationException e) {
      e.printStackTrace();
      return;
    }

    initNode(node, mousePosition);
    graphicElementsList.addElement(node);
    GBNode.elementsList.addElement(node);
//    externalElementsList.addElement(node);
    setCurrentElement(node);
    node.setVisibleSelectionRects(true);
    initialPoint.setLocation(mousePosition);
//    repaint();
```

```
      return;
    }
    catch (IllegalAccessException e) {
      e.printStackTrace();
      return;
    }
    catch (InstantiationException e) {
      e.printStackTrace();
      return;
    }
  }

  public int getGraphicElementType() {
    return graphicElementType;
  }

  public void graphicAreaMouseMoved(MouseEvent e) {
    if (currentElement == null)
      return;

    if (state == CREATING)
      if (graphicElementType == ET_CONNECTION)
        if (newConnection != null) {
          Graphics gc;

          gc = getGraphics();
          gc.setXORMode(Color.white);
          gc.setColor(Color.black);
          if (newConnection.points.size() == 1) {
            Point pointInterception;

            pointInterception = (newConnection.getInitialNode()).g
etInterceptionPoint(e.getPoint());
            if (! ((initialPoint.getLocation()).equals(pointInterc
eption))) {
              initialPoint.setLocation(pointInterception);
              currentPoint.setLocation(pointInterception);
            }
          }
          newConnection.drawSegment(gc, initialPoint, currentPoint
);
          currentPoint.setLocation(e.getPoint());
          newConnection.drawSegment(gc, initialPoint, currentPoint
);
        }
  }

  public void graphicAreaMousePressed(MouseEvent e) {
```

```
        public void mouseMoved(MouseEvent e) {
          graphicAreaMouseMoved(e);
        }
      });
      addMouseMotionListener(new java.awt.event.MouseMotionAdapter
() {
        public void mouseDragged(MouseEvent e) {
          graphicAreaMouseMoved(e);
        }
      });
      addMouseListener(new java.awt.event.MouseAdapter() {
        public void mousePressed(MouseEvent e) {
          graphicAreaMousePressed(e);
        }
      });
      addKeyListener(new java.awt.event.KeyAdapter() {
        public void keyPressed(KeyEvent e) {
          graphicAreaKeyPressed(e);
        }
      });

/*    jPanel1.addMouseListener(new java.awt.event.MouseAdapter() {

      public void mouseReleased(MouseEvent e) {
        jPanel1_mouseReleased(e);
      }
    });*/
        }

  public void setGraphicElementType(String className) {
    try {
      GBGraphicElement graphicElement;

      graphicElementClass = Class.forName(className);
      graphicElement = (GBGraphicElement) graphicElementClass.newI
nstance();
      if (graphicElement instanceof GBNode)
        graphicElementType = ET_NODE;
      else if (graphicElement instanceof GBLabel)
        graphicElementType = ET_LABEL;
      else if (graphicElement instanceof GBConnection)
        graphicElementType = ET_CONNECTION;
      else
        graphicElementType = ET_NONE;
      graphicElement = null;
    }
    catch (ClassNotFoundException e) {
      e.printStackTrace();
```

```
package GraphicBricks;

import java.awt.*;
import java.awt.event.*;
import java.util.Vector;
import javax.swing.*;

public class GBGraphicArea extends JPanel {
  public static final int CREATING = 1;
  public static final int SELECTING = 2;
  public static final int MOVINING = 3;
  public static final int RESIZING = 4;
  public static final int ET_NONE = 0;
  public static final int ET_NODE = 1;
  public static final int ET_LABEL = 2;
  public static final int ET_CONNECTION = 3;

  protected GBNode initialNode;
  protected GBNode finalNode;
  protected GBGraphicElement currentElement;
  protected GBConnection newConnection;
  protected Point initialPoint;
  public Point currentPoint;
  protected int state;
  protected int graphicElementType;
  protected Class graphicElementClass;
  protected Vector graphicElementsList;

        public GBGraphicArea() {
                  super();
      graphicElementType = ET_NONE;
      currentElement = null;
      initialNode = null;
      finalNode = null;
      newConnection = null;
      graphicElementsList = new Vector();
      setState(SELECTING);
      initialPoint = new Point(30, 30);
      currentPoint = new Point(0, 0);
        setBackground(Color.white);
        setLayout(null);
      addMouseListener(new java.awt.event.MouseAdapter() {
        public void mouseClicked(MouseEvent e) {
          graphicAreaMouseClicked(e);
        }
      });
      addMouseMotionListener(new java.awt.event.MouseMotionAdapter
() {
```

```
package GraphicBricks;

import java.awt.*;
import java.util.Vector;

public interface GBGraphicElement {
  public void setVisibleSelectionRects(boolean visibleSelectionRec
ts);
  public void destroy();
  public boolean isDeletable();
}
```

```
}
```

```
    for (int cont = 0; cont < rectsCount; cont++) {
      ((GBSelectionRect) selectionRects.elementAt(cont)).setVisibl
e(visibleSelectionRects);
    }
  }

  public void updateSelectionRectsPos() {
    int rectsCount, rectCenter, xDiff, yDiff;
    GBSelectionRect selectionRect;

    rectCenter = GBSelectionRect.RECT_SIZE/2;
    selectionRect = (GBSelectionRect) selectionRects.elementAt(0);
    xDiff = getX() - (selectionRect.getX() + rectCenter);
    yDiff = getY() - (selectionRect.getY() + rectCenter);
    rectsCount = selectionRects.size();
    for (int cont = 0; cont < rectsCount; cont++) {
      selectionRect = (GBSelectionRect) selectionRects.elementAt(c
ont);
      selectionRect.setLocation(selectionRect.getX() + xDiff, sele
ctionRect.getY() + yDiff);
    }
  }

  public static void setElementsList(Vector newElementsList) {
    elementsList = newElementsList;
  }

  public void setLinkedElement(GBGraphicElement linkedElement) {
    this.linkedElement = linkedElement;
  }

  public void destroy() {
//    elementsList.remove(this);
//    graphicArea.graphicElementsList.remove(this);
    graphicArea.remove(this);
    graphicArea.repaint();
  }

  public void paintComponent(Graphics g) {
    int height, posCaptionY;
                FontMetrics fm;

    super.paintComponent(g);
                fm = getFontMetrics(getFont());
    height = getHeight();
    posCaptionY = (height - fm.getHeight())/2 + fm.getAscent();
    g.drawString(caption, 5, posCaptionY);
  }
```

```
;
    selectionRect.setBounds(nodeX - rectCenter, nodeY - rectCenter
 + getHeight() - 1, GBSelectionRect.RECT_SIZE, GBSelectionRect.REC
T_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);
  }

  protected void rectSelectionPressed(MouseEvent e) {
    idRectSelected = selectionRects.indexOf(e.getComponent());
  }

  protected void nodeMousePressed(MouseEvent e) {
    if (graphicArea.getState() == GBGraphicArea.SELECTING) {
      if (graphicArea.getCurrentElement() == this)
        setVisibleSelectionRects(false);
      else
        graphicArea.setCurrentElement(this);
      initialPoint.setLocation(e.getPoint());
    }
    graphicArea.requestFocus();
  }

  protected void nodemouseReleased(MouseEvent e) {
    if (graphicArea.getState() == GBGraphicArea.SELECTING) {
      updateSelectionRectsPos();
      setVisibleSelectionRects(true);
    }
  }

  protected void nodeMouseDragged(MouseEvent e) {
    if (graphicArea.getState() == GBGraphicArea.SELECTING)
      setLocation(getX() + (e.getX() - initialPoint.x), getY() + (
e.getY() - initialPoint.y));
  }

  public void setVisibleSelectionRects(boolean visibleSelectionRec
ts) {
    int rectsCount;

    rectsCount = selectionRects.size();
```

```
, GBSelectionRect.RECT_SIZE, GBSelectionRect.RECT_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);

    selectionRect = new GBSelectionRect();
    if (resizable)
      selectionRect.setCursor(new Cursor(Cursor.NE_RESIZE_CURSOR))
;
    selectionRect.setBounds(nodeX - rectCenter + getWidth() - 1, n
odeY - rectCenter, GBSelectionRect.RECT_SIZE, GBSelectionRect.RECT
_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);

    selectionRect = new GBSelectionRect();
    if (resizable)
      selectionRect.setCursor(new Cursor(Cursor.SE_RESIZE_CURSOR))
;
    selectionRect.setBounds(nodeX - rectCenter + getWidth() - 1, n
odeY - rectCenter + getHeight() - 1, GBSelectionRect.RECT_SIZE, GB
SelectionRect.RECT_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);

    selectionRect = new GBSelectionRect();
    if (resizable)
      selectionRect.setCursor(new Cursor(Cursor.SW_RESIZE_CURSOR))
```

```
    caption = "Label";
    addMouseListener(new java.awt.event.MouseAdapter() {
      public void mousePressed(MouseEvent e) {
        nodeMousePressed(e);
      }
    });
    addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseReleased(MouseEvent e) {
        nodemouseReleased(e);
      }
    });
    addMouseMotionListener(new java.awt.event.MouseMotionAdapter()
 {
      public void mouseDragged(MouseEvent e) {
        nodeMouseDragged(e);
      }
    });
//    createSelectionsRects();
  }

  public boolean isDeletable() {
    return deletable;
  }

  public void setGraphicArea(GBGraphicArea graphicArea) {
    this.graphicArea = graphicArea;
  }

  public void setCaption(String caption) {
    this.caption = caption;
  }

  public String getCaption() {
    return caption;
  }

  public void createSelectionsRects() {
    int nodeX, nodeY, rectCenter;
    GBSelectionRect selectionRect;

    nodeX = getX();
    nodeY = getY();
    rectCenter = GBSelectionRect.RECT_SIZE/2;
    selectionRect = new GBSelectionRect();
    if (resizable)
      selectionRect.setCursor(new Cursor(Cursor.NW_RESIZE_CURSOR))
;
    selectionRect.setBounds(nodeX - rectCenter, nodeY - rectCenter
```

```
//Title:        Your Product Name
//Version:
//Copyright:    Copyright (c) 1998
//Author:       Your Name
//Company:      Your Company
//Description:  Your description

package GraphicBricks;

import java.awt.*;
import java.awt.event.*;
import java.util.Vector;
import javax.swing.*;

public class GBLabel extends JComponent implements GBGraphicElemen
t {
  protected GBGraphicArea graphicArea;
  public static Vector elementsList = null;
  protected boolean deletable;
  protected boolean moveable;
  protected boolean selection;
  protected Vector selectionRects;
  protected String name;
  protected int idRectSelected;

  protected int maxSize;
  protected int minSize;
  protected boolean resizable;
  protected String caption;
  protected Point initialPoint;
  protected GBGraphicElement linkedElement;

  public GBLabel() {
    setForeground(Color.black);
    setBackground(Color.white);
    initialPoint = new Point(0, 0);
    deletable = false;
    moveable = true;
    selection = true;
    selectionRects = new Vector();
    name = "";
    idRectSelected = -1;

    linkedElement = null;
    maxSize = 1000;
    minSize = 10;
    resizable = false;
```

```
    }
}
```

```
;
      connection.updateLastPoint();
      connection.updateSelectionRectsPos(connection.points.size()
- 1);
    }
  }

  protected void updateConnectionsOut() {
    GBConnection connection;

    for (int cont = 0; cont < connectionOutList.size(); cont++) {
      connection = (GBConnection) connectionOutList.elementAt(cont
);
      connection.updateFirstPoint();
      connection.updateSelectionRectsPos(0);
    }
  }

  public abstract Point getInterceptionPoint(Point connectionPoint
);

  public static void setElementsList(Vector newElementsList) {
    elementsList = newElementsList;
  }

  public void removeConnectionOut(GBConnection connection) {
    connectionOutList.remove(connection);
  }

  public void removeConnectionIn(GBConnection connection) {
    connectionInList.remove(connection);
  }

  public void destroy() {
    GBConnection connection;

    while (connectionOutList.size() > 0) {
      connection = (GBConnection) connectionOutList.elementAt(0);
      connection.destroy();
    }
    while (connectionInList.size() > 0) {
      connection = (GBConnection) connectionInList.elementAt(0);
      connection.destroy();
    }
    elementsList.remove(this);
    graphicArea.graphicElementsList.remove(this);
    graphicArea.remove(this);
    graphicArea.repaint();
```

```
    rectsCount = selectionRects.size();
    for (int cont = 0; cont < rectsCount; cont++) {
      selectionRect = (GBSelectionRect) selectionRects.elementAt(c
ont);
      selectionRect.setLocation(selectionRect.getX() + xDiff, sele
ctionRect.getY() + yDiff);
    }
  }

  public void addConnectionOut(GBConnection connection) {
    connectionOutList.addElement(connection);
  }

  public void addConnectionIn(GBConnection connection) {
    connectionInList.addElement(connection);
  }

  protected void drawConnectionsIn() {
    Graphics gc;
    GBConnection connection;

    gc = graphicArea.getGraphics();
    gc.setXORMode(Color.white);
    for (int cont = 0; cont < connectionInList.size(); cont++) {
      connection = (GBConnection) connectionInList.elementAt(cont)
;
      connection.drawFinalSegment(gc);
    }
  }

  protected void drawConnectionsOut() {
    Graphics gc;
    GBConnection connection;

    gc = graphicArea.getGraphics();
    gc.setXORMode(Color.white);
    for (int cont = 0; cont < connectionOutList.size(); cont++) {
      connection = (GBConnection) connectionOutList.elementAt(cont
);
      connection.drawInitialSegment(gc);
    }
  }

  protected void updateConnectionsIn() {
    GBConnection connection;

    for (int cont = 0; cont < connectionInList.size(); cont++) {
      connection = (GBConnection) connectionInList.elementAt(cont)
```

```
      if (graphicArea.getCurrentElement() == this)
        setVisibleSelectionRects(false);
      else
        graphicArea.setCurrentElement(this);
      initialPoint.setLocation(e.getPoint());
    }
    graphicArea.requestFocus();
  }

  protected void nodemouseReleased(MouseEvent e) {
    if (graphicArea.getState() == GBGraphicArea.SELECTING) {
      updateSelectionRectsPos();
      drawConnectionsIn();
      drawConnectionsOut();
      updateConnectionsIn();
      updateConnectionsOut();
      setVisibleSelectionRects(true);
      graphicArea.repaint();
//      drawConnectionsIn();
//      drawConnectionsOut();
    }
  }

  protected void nodeMouseDragged(MouseEvent e) {
    if (graphicArea.getState() == GBGraphicArea.SELECTING)
      setLocation(getX() + (e.getX() - initialPoint.x), getY() + (
e.getY() - initialPoint.y));
  }

  public void setVisibleSelectionRects(boolean visibleSelectionRec
ts) {
    int rectsCount;

    rectsCount = selectionRects.size();
    for (int cont = 0; cont < rectsCount; cont++) {
      ((GBSelectionRect) selectionRects.elementAt(cont)).setVisibl
e(visibleSelectionRects);
    }
  }

  public void updateSelectionRectsPos() {
    int rectsCount, rectCenter, xDiff, yDiff;
    GBSelectionRect selectionRect;

    rectCenter = GBSelectionRect.RECT_SIZE/2;
    selectionRect = (GBSelectionRect) selectionRects.elementAt(0);
    xDiff = getX() - (selectionRect.getX() + rectCenter);
    yDiff = getY() - (selectionRect.getY() + rectCenter);
```

```
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);

    selectionRect = new GBSelectionRect();
    selectionRect.setCursor(new Cursor(Cursor.SE_RESIZE_CURSOR));
    selectionRect.setBounds(nodeX - rectCenter + getWidth() - 1, n
odeY - rectCenter + getHeight() - 1, GBSelectionRect.RECT_SIZE, GB
SelectionRect.RECT_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);

    selectionRect = new GBSelectionRect();
    selectionRect.setCursor(new Cursor(Cursor.SW_RESIZE_CURSOR));
    selectionRect.setBounds(nodeX - rectCenter, nodeY - rectCenter
 + getHeight() - 1, GBSelectionRect.RECT_SIZE, GBSelectionRect.REC
T_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);
  }

  protected void rectSelectionPressed(MouseEvent e) {
    idRectSelected = selectionRects.indexOf(e.getComponent());
  }

  protected void nodeMousePressed(MouseEvent e) {
    if (graphicArea.getState() == GBGraphicArea.CREATING) {
      if (graphicArea.getGraphicElementType() == GBGraphicArea.ET_
CONNECTION) {
        graphicArea.setConnectionNodes(this);
      }
    }
    else if (graphicArea.getState() == GBGraphicArea.SELECTING) {
```

```
      afterShowProps();
    }
  }

  protected abstract void beforeShowProps();

  protected abstract void afterShowProps();

  public static void setPropertiesDlg(Window newPropertiesDialog)
{
    propertiesDialog = newPropertiesDialog;
  }

  public static Window getPropertiesDlg() {
    return propertiesDialog;
  }

  public void createSelectionsRects() {
    int nodeX, nodeY, rectCenter;
    GBSelectionRect selectionRect;

    nodeX = getX();
    nodeY = getY();
    rectCenter = GBSelectionRect.RECT_SIZE/2;
    selectionRect = new GBSelectionRect();
    selectionRect.setCursor(new Cursor(Cursor.NW_RESIZE_CURSOR));
    selectionRect.setBounds(nodeX - rectCenter, nodeY - rectCenter
, GBSelectionRect.RECT_SIZE, GBSelectionRect.RECT_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
    });
    ((JPanel) graphicArea).add(selectionRect, 0);
//    getParent().add(selectionRect, 0);
    selectionRects.addElement(selectionRect);

    selectionRect = new GBSelectionRect();
    selectionRect.setCursor(new Cursor(Cursor.NE_RESIZE_CURSOR));
    selectionRect.setBounds(nodeX - rectCenter + getWidth() - 1, n
odeY - rectCenter, GBSelectionRect.RECT_SIZE, GBSelectionRect.RECT
_SIZE);
    selectionRect.addMouseListener(new java.awt.event.MouseAdapter
() {
      public void mousePressed(MouseEvent e) {
        rectSelectionPressed(e);
      }
```

```
    resizable = false;
    caption = "";
    connectionOutList = new Vector();
    connectionInList = new Vector();
    addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseClicked(MouseEvent e) {
        nodeMouseClicked(e);
      }
    });
    addMouseListener(new java.awt.event.MouseAdapter() {
      public void mousePressed(MouseEvent e) {
        nodeMousePressed(e);
      }
    });
    addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseReleased(MouseEvent e) {
        nodemouseReleased(e);
      }
    });
    addMouseMotionListener(new java.awt.event.MouseMotionAdapter()
 {
      public void mouseDragged(MouseEvent e) {
        nodeMouseDragged(e);
      }
    });
//    createSelectionsRects();
  }

  public boolean isDeletable() {
    return deletable;
  }

  public void setGraphicArea(GBGraphicArea graphicArea) {
    this.graphicArea = graphicArea;
  }

  public void setCaption(String caption) {
    this.caption = caption;
  }

  public String getCaption() {
    return caption;
  }

  protected void nodeMouseClicked(MouseEvent e) {
    if (e.getClickCount() >= 2 && propertiesDialog != null) {
      beforeShowProps();
      propertiesDialog.setVisible(true);
```

```
//Title:        Your Product Name
//Version:
//Copyright:    Copyright (c) 1998
//Author:       Your Name
//Company:      Your Company
//Description:  Your description

package GraphicBricks;

import java.awt.*;
import java.awt.event.*;
import java.util.Vector;
import javax.swing.*;

public abstract class GBNode extends JComponent implements GBGraph
icElement {
  protected GBGraphicArea graphicArea;
  public static Vector elementsList = null;
  protected boolean deletable;
  protected boolean moveable;
  protected boolean selection;
  protected Vector selectionRects;
  protected static Window propertiesDialog = null;
  protected String name;
  protected int idRectSelected;

  protected int maxSize;
  protected int minSize;
  protected boolean resizable;
  protected String caption;
  protected Vector connectionOutList;
  protected Vector connectionInList;
  protected Point initialPoint;

  public GBNode() {
    setForeground(Color.black);
    setBackground(Color.white);
    initialPoint = new Point(0, 0);
    deletable = true;
    moveable = true;
    selection = true;
    selectionRects = new Vector();
    name = "";
    idRectSelected = -1;

    maxSize = 1000;
    minSize = 10;
```

```
    else if (connectionPoint.y < (yRect + heightRect)) {
      if (connectionPoint.x < xRect) {
        interceptionPoint.x = xRect - 1;
        interceptionPoint.y = yRect + heightRect / 2;
      }
      else {
        interceptionPoint.x = xRect + widthRect;
        interceptionPoint.y = yRect + heightRect / 2;
      }
    }
    else {
      interceptionPoint.x = xRect + widthRect / 2;
      interceptionPoint.y = yRect + heightRect;
    }

    return interceptionPoint;
  }
}
```

```
    else if (connectionPoint.y < (yRect + heightRect)) {
      if (connectionPoint.x < xRect) {
        interceptionPoint.x = xRect - 1;
        interceptionPoint.y = yRect + heightRect / 2;
      }
      else {
        interceptionPoint.x = xRect + widthRect;
        interceptionPoint.y = yRect + heightRect / 2;
      }
    }
    else {
      interceptionPoint.x = xRect + widthRect / 2;
      interceptionPoint.y = yRect + heightRect;
    }

    return interceptionPoint;
  }
}
```

```
package GraphicBricks;

import java.awt.*;

public class GBRectangle extends GBNode {

  public GBRectangle() {
    super();
  }

  public void paintComponent(Graphics g) {
    int width, height, posCaptionY;
                FontMetrics fm;

    super.paintComponent(g);
                fm = getFontMetrics(getFont());
    width = getWidth();
    height = getHeight();
    if (true) {
      g.setColor(getBackground());
      g.fillRect(1, 1, width-2, height-2);
    }
//    g.setColor(Color.black);
    g.setColor(getForeground());
    g.drawRect(0, 0, width-1, height-1);
    posCaptionY = (height - fm.getHeight())/2 + fm.getAscent();
    g.drawString(caption, 5, posCaptionY);
  }

  protected void beforeShowProps() {
  }

  protected void afterShowProps() {
  }

  public Point getInterceptionPoint(Point connectionPoint) {
    Point interceptionPoint;
    int xRect, yRect, widthRect, heightRect;

    interceptionPoint = new Point();
    xRect = getX();
    yRect = getY();
    widthRect = getWidth();
    heightRect = getHeight();
    if (connectionPoint.y < yRect) {
      interceptionPoint.x = xRect + widthRect / 2;
      interceptionPoint.y = yRect - 1;
    }
```

```
package GraphicBricks;

import java.awt.*;
import javax.swing.*;

public class GBSelectionRect extends JComponent {
  public static final int RECT_SIZE = 5;

  public GBSelectionRect() {
    setVisible(false);
    setBackground(Color.black);
  }

  public void paintComponent(Graphics g) {
    int width, height;

    super.paintComponent(g);
    width = getWidth();
    height = getHeight();
    if (true) {
      g.setColor(getBackground());
      g.fillRect(0, 0, width-1, height-1);
    }
    g.setColor(Color.black);
    g.drawRect(0, 0, width-1, height-1);
  }
}
```

```
            titulo = (String) (relacao.titulos.elementAt(cont2));
            System.out.println("  Titulo " + (cont2 + 1) + ": " + titu
lo);
        }
      }
    }
}
```

```
    this.getContentPane().add(jPanel1, null);
    jPanel1.add(jButton1, null);
    jPanel1.add(jButton2, null);
    jPanel1.add(jButton3, null);
    jPanel1.add(jButton4, null);
  }

  void jButton1_actionPerformed(ActionEvent e) {
    graphicArea.setState(GBGraphicArea.CREATING);
    graphicArea.setGraphicElementType("GraphicBricks.Conceito");
//    graphicArea.setGraphicElementType("GraphicBricks.GBLabel");
    System.out.println("Conceito !");
//    System.out.println("Label !");
  }

  void jButton2_actionPerformed(ActionEvent e) {
    graphicArea.setState(GBGraphicArea.CREATING);
    graphicArea.setGraphicElementType("GraphicBricks.Relacao");
    System.out.println("Relação !");
  }

  void jButton3_actionPerformed(ActionEvent e) {
    graphicArea.setState(GBGraphicArea.SELECTING);
    System.out.println("Selection !");
  }

  void jButton4_actionPerformed(ActionEvent e) {
    int numConceitos, numRelacoes;
    Conceito conceito;
    Relacao relacao;

    System.out.println();
    System.out.println("Conceitos:");
    numConceitos = conceitos.size();
    for (int cont = 0; cont < numConceitos; cont++) {
      conceito = (Conceito) conceitos.elementAt(cont);
      System.out.println(conceito.nome);
    }

    System.out.println();
    System.out.println("Relacoes:");
    numRelacoes = relacoes.size();
    for (int cont = 0; cont < numRelacoes; cont++) {
      relacao = (Relacao) relacoes.elementAt(cont);
      System.out.println("Relacao " + (cont + 1) +":");
      for (int cont2 = 0; cont2 < relacao.titulos.size(); cont2++)
{
        String titulo;
```

```
    this.setBounds(100, 50, 500, 600);
    this.getContentPane().setLayout(null);

    Conceito.setElementsList(conceitos);
    Relacao.setElementsList(relacoes);
    graphicArea = new GBGraphicArea();
    graphicArea.setBounds(10, 50, 450, 500);

    jPanel1.setBorder(BorderFactory.createEtchedBorder());
    jPanel1.setBounds(new Rectangle(1, 2, 442, 38));
    jPanel1.setLayout(null);
    jButton1.setText("Conceito");
    jButton1.setBounds(new Rectangle(14, 3, 86, 27));
    jButton1.addActionListener(new java.awt.event.ActionListener()
{
      public void actionPerformed(ActionEvent e) {
        jButton1_actionPerformed(e);
      }
    });

    jButton2.setText("Relação");
    jButton2.setBounds(new Rectangle(110, 3, 86, 27));
    jButton2.addActionListener(new java.awt.event.ActionListener()
{
      public void actionPerformed(ActionEvent e) {
        jButton2_actionPerformed(e);
      }
    });

    jButton3.setText("Selecionar");
    jButton3.setBounds(new Rectangle(210, 3, 106, 27));
    jButton3.addActionListener(new java.awt.event.ActionListener()
{
      public void actionPerformed(ActionEvent e) {
        jButton3_actionPerformed(e);
      }
    });

    jButton4.setText("Listar");
    jButton4.setBounds(new Rectangle(330, 3, 86, 27));
    jButton4.addActionListener(new java.awt.event.ActionListener()
{
      public void actionPerformed(ActionEvent e) {
        jButton4_actionPerformed(e);
      }
    });

    this.getContentPane().add(graphicArea, null);
```

```
//Title:        Your Product Name
//Version:
//Copyright:    Copyright (c) 1998
//Author:       Your Name
//Company:      Your Company
//Description:  Your description

package GraphicBricks;

import java.awt.*;
import java.util.Vector;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.border.*;

public class GBTest extends JFrame {
        GBGraphicArea graphicArea;
  Vector conceitos;
  Vector relacoes;
  JPanel jPanel1 = new JPanel();
  JButton jButton1 = new JButton();
  JButton jButton2 = new JButton();
  JButton jButton3 = new JButton();
  JButton jButton4 = new JButton();
  TitledBorder titledBorder1;

  public GBTest() {
    try {
      jbInit();
    }
    catch (Exception e) {
      e.printStackTrace();
    }
  }

  public static void main(String[] args) {
    GBTest gbTest = new GBTest();

    gbTest.setVisible(true);
//      GBConnection.drawArrow();
  }

  private void jbInit() throws Exception {
    conceitos = new Vector();
    relacoes = new Vector();

    titledBorder1 = new TitledBorder("");
```

```
[Browser]
Create=1
Visible=1
State=2
Left=0
Top=33
Width=973
Height=582
ClientWidth=1024
ClientHeight=598
NavigationPane=1
StructurePane=1
SplitNavigationView=246
SplitNavigationPView=598
SplitBrowseView=232
SplitTabsStatus=171
ivSplitClientMessages=138
Curtain=0

[Tab0]
NavigationURL=.\GBGraphicArea.java
URLCount=1
URL0=.\GBGraphicArea.java

[Tab0@.\GBGraphicArea.java]
CursorX=1
CursorY=132
TopLine=108
LeftCol=1
```