Interface de Autoria para o Modelo Pedagógico do MathTutor

Eliane Pozzebon, Luciana Bolan Frigo, Adriana Postal, Guilherme Bittencourt

[1]Universidade Federal de Santa Catarina
Departamento de Automação e Sistemas
Caixa Postal 476 – CEP 88040-900
Florianópolis, SC Tel.: 331-7597

(eliane,lu,apostal, gb) @das.ufsc.br

Resumo. *Propõe-se uma abordagem para o modelo pedagógico baseado em Redes de Petri, que são utilizadas para modelar as Unidades Pedagógicas visando facilitar a tarefa do especialista na organização dos conteúdos e ao mesmo tempo fornecer flexibilidade e adaptatividade na apresentação. A interação entre cada agente tutor responsável por um subdomínio e o aprendiz é representado por um objeto da Rede de Petri e é automaticamente traduzido para regras de produção que formam o sistema especialista.*

Abstract. *We propose a new approach to pedagogical model based on Petri Net, which is used to modelling the Pedagogical Units to facilitate the teacher task of organizing the contents of a tutorial system and at the same time to provide adaptiveness and flexibility in the presentation. The interaction between the apprentice and each agent of the system is controlled by an object Petri net,automatically translated into a rule- based expert system. The teacher has several available Petri net modules during the course preparation. The idea to transform the task of designing a course easier is to develop a compiler that accepts Petri net definitions and automatically generates the expert system rules that implement them.*

1. Introdução

Este artigo apresenta um Sistema Tutor Inteligente (STI) chamado MathTutor, que utiliza técnicas cognitivas para apoiar o ensino-aprendizagem. MathTutor é formado por agentes distribuídos que interagem entre si possibilitando ao aprendiz ter uma navegação guiada ou ainda navegar livremente, sem a ajuda dos agentes. Sua arquitetura segue um modelo para concepção e desenvolvimento de ambientes de aprendizagem assistidos por computador, chamado MATHEMA. Este modelo necessita de um ambiente computacional onde seja possível construir uma sociedade de agentes cognitivos, isto é, agentes que incorporem um sistema especialista.

Neste artigo descreve-se como o modelo pedagógico pode ser gerado a partir das informações obtidas pelo mecanismo de autoria. A interface de autoria permite a construção e visualização da estrutura do modelo pedagógico gerada pelo especialista humano.

O artigo segue a organização: na seção 2 introduz-se uma definição sobre os Sistemas Tutores Inteligentes e sua respectiva arquitetura; na seção 3 apresenta-se o MathTutor; na seção 4 é detalhada a interface de autoria e na última seção, apresentam-se as conclusões.

2. Sistemas Tutores Inteligentes

Um Sistema Tutor Inteligente (STI) é definido, segundo Auberger [Auberger, 1998], como um sistema instrucional baseado em computador, que ensina o estudante de maneira interativa, usando os conceitos de Inteligência Artificial. Murray [Murray, 1999] ressalta como um dos objetivos dos STIs ser capaz de modelar comportamentos complexos de ensino, os quais se adaptam às necessidades do estudante, à situação de aprendizagem e ao assunto da instrução.

As funções operacionais básicas de um STI são determinadas por quatro componentes principais ou modelos conforme mostra a figura 1:

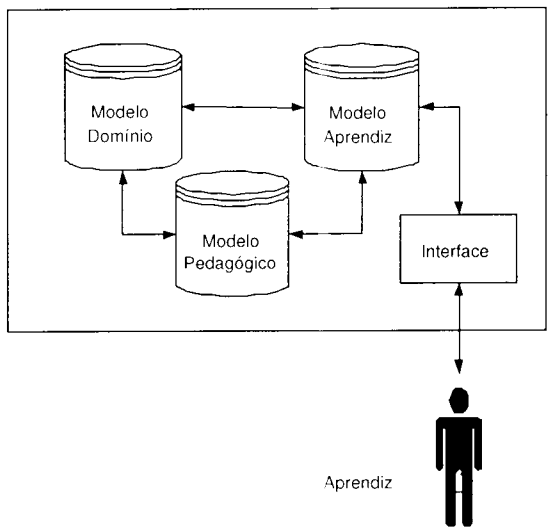

Figura 1: Arquitetura Clássica do STI

- Modelo do Especialista ou Domínio
  Segundo Auberger [Auberger, 1998], este modelo contém o conhecimento do domínio do sistema, e os mecanismos de inferência. O modelo do especialista é fundamentalmente uma base de conhecimento, contendo informações sobre um determinado domínio, que é organizada de alguma maneira para representar o conhecimento de um especialista ou professor. É, geralmente, considerado o componente central de qualquer STI. Em essência, este modelo incorpora a maior parte da "inteligência" do sistema na forma do conhecimento necessário para solucionar problemas do domínio [Park, 1998].

- Modelo do Aprendiz
  Este modelo descreve o conhecimento e crenças do aprendiz. É, na verdade, uma representação abstrata do aprendiz no sistema. A principal função deste modelo é permitir que o conteúdo instrucional seja adaptado para um aprendiz individual. Com conhecimento sobre o aprendiz, o tutor pode controlar a ordem e a dificuldade do material a ser apresentado, bem como prover remediações [Auberger, 1998]. A chave para um ensino personalizado e inteligente em um sistema tutorial é, sem dúvida, o conhecimento que o sistema deve ter de seu próprio usuário.

- Modelo Pedagógico
  Também conhecido como tutor ou instrucional, usa as informações do modelo do estudante para determinar quais aspectos do conhecimento do domínio devem ser apresentados para o Aprendiz. Representa os métodos e técnicas didáticas utilizadas no processo da comunicação de conhecimento. Este modelo contém o conhecimento necessário para tomar decisões sobre quais táticas de ensino devem ser empregadas dentre aquelas disponíveis no sistema. O modelo pedagógico diagnostica as necessidades do aprendiz com base nas informações do modelo do aprendiz e na solução do especialista. Em geral, as decisões são sobre qual informação apresentar ao estudante, quando e como apresentá-la.
  As decisões pedagógicas são tomadas no contexto de um ambiente educacional que determina o grau de controle sobre a atividade e sobre a interação possuídos respectivamente pelo sistema tutorial e pelo estudante [Wenger, 1987].

- Modelo da Interface
  É responsável por processar o fluxo de informações com o meio externo e o sistema. Traduz a representação interna do sistema numa linguagem compreensível para o aprendiz. Na engenharia de software, a interface do usuário tem sido a primeira preocupação dos projetistas quando estão discutindo a criação de uma nova aplicação, pois como afirmam Hix e Hartson [Hix and Hartson, 1993]: "Para os usuários, a interface é o próprio sistema".
  Muitos princípios baseados em teorias cognitivas têm sido propostos para projetos de interface como resultado de pesquisas na área da interação homem-máquina. Entretanto, a meta da maioria destas pesquisas é que o usuário não necessite se adaptar à interface do sistema e sim, a interface deve ser projetada para que seja intuitiva tornando seu aprendizado natural para o usuário.
  Apesar da interface operar em estreita cooperação com ambos os modelos, pedagógico e do estudante, suas decisões são de natureza distinta, requerendo um tipo diferente de conhecimento.

3. Sistema MathTutor

O MathTutor é STI por ser um programa capaz de adaptar suas estratégias de ensino mediante a interação com o aprendiz e atualiza as informações necessárias conforme as intervenções do aprendiz com o sistema. A missão do MathTutor é a mesma missão inerente aos STIs que é permitir que o processo ensino-aprendizagem seja personalizado e utilizado como suporte ao ensino tradicional.

A arquitetura do MathTutor é baseada no modelo conceitual MATHEMA [de Barros Costa, 1997] que é um modelo para concepção e desenvolvimento de ambientes de aprendizagem assistidos por computador e consiste basicamente de três módulos: a sociedade de agentes tutores artificiais (SATA), a interface do aprendiz e a interface de autoria, como pode-se verificar na figura 2.

A implementação do MathTutor partiu inicialmente da busca de ferramentas de domínio público que possuíssem as características desejadas. A partir deste princípio foram escolhidos, o JESS (Java Expert System Shell) para o suporte a Inteligência Artificial,

JATLite (Java Agent Template Lite) para a comunicação entre os agentes e Java Servlets e HTML para o suporte multimídia.

A interface fornece acesso ao sistema através de qualquer navegador de Internet. O módulo da interface de autoria permite a definição da estrutura e conteúdos do curso e é discutido em maiores detalhes na seção 4. Finalmente, a SATA consiste de um SMA onde cada agente, ao lado das capacidades de comunicação e cooperação, contém um STI completo enfocado em um subdomínio do domínio alvo.

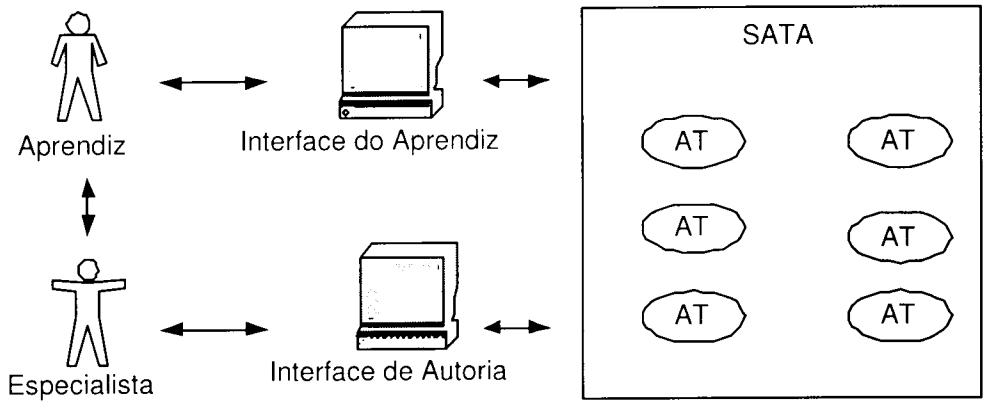

Figura 2: Arquitetura do Sistema

Para facilitar o entendimento de como o MathTutor funciona escolheu-se um exemplo cujo domínio de aplicação é apresentar os principais conceitos de abstração de dados e de procedimentos aos alunos de Fundamentos da Estrutura da Informação, disciplina oferecida no curso de Engenharia de Controle e Automação da Universidade Federal de Santa Catarina [Bittencourt, 2004].

A modelagem do conhecimento sobre um dado domínio é abordada segundo duas visões, uma externa e outra interna:

Visão Externa: é um esquema de particionamento baseado em suposições epistemológicas, estando comprometido com alguma visão particular do domínio. Consiste de uma perspectiva tridimensional [Frigo, 2002]:

- **Contexto (C)**: compõe-se de diferentes interpretações a um domínio de conhecimento, constituindo-se de representações ou abordagens diferentes de um mesmo objeto de conhecimento.
- **Profundidade (P)**: relativa a um contexto particular, refere-se a um refinamento na linguagem de percepção.
- **Lateralidade (L)**: referente aos conhecimentos intimamente relacionados ao domínio de aplicação, proveniente de uma visão particular de contexto e profundidade.

Para o domínio de conhecimento – Estrutura da Informação – existem dois contextos: teórico e prático; e cada contexto é trabalhado em duas profundidades: abstração procedural0 e abstração de dados. Desta forma, a SAT consiste de quatro agentes, cada um responsável por um dos seguintes subdomínios:

- PT – abstração procedimental teórica;
- PP – abstração procedimental prática;
- DT – abstração de dados teórica;
- DP – abstração de dados prática.

Cada um dos quatro Agentes Tutores (AT) tem a arquitetura mostrada na figura 3, e seu comportamento é controlado pelo módulo *Coordenador*, para conhecer as atividades desenvolvidas pelo Coordenador, ver [Postal et al., 2004].

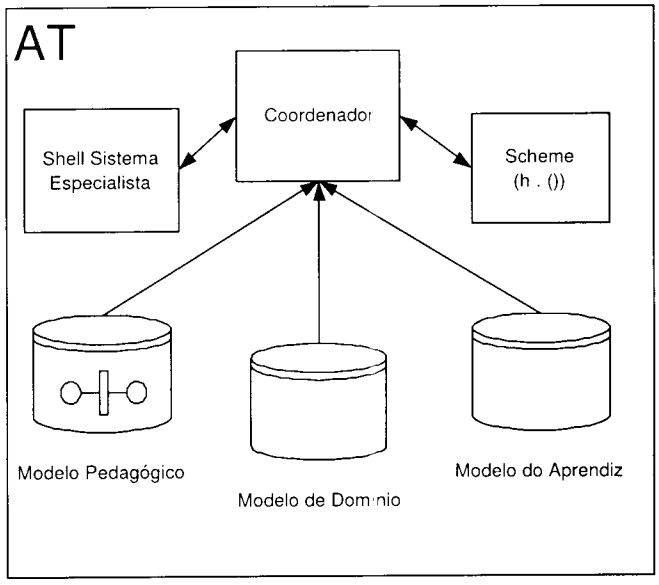

Figura 3: Arquitetura de um Agente Tutor

Visão Interna: o conhecimento associado a cada subdomínio (para este exemplo: PT, PP, DT e DP) é organizado em um ou mais *currículos*. Cada currículo consiste de um conjunto de *unidades pedagógicas* (UPs) e cada UP é associada a um conjunto de *problemas*.

O Modelo Pedagógico do Mathtutor controla a interação entre o aprendiz e cada agente do sistema. Ele é implementado por uma Rede de Petri a Objeto (RdPO), automaticamente traduzida em um sistema especialista baseado em regras. As fichas da RdP são compostas por um objeto que identifica o aprendiz. As transições da RdPO são controladas por condições lógicas que se referem ao modelo do aprendiz e disparam estas transições produzindo ações que atualizam o modelo do aprendiz. Durante a preparação do curso, o especialista tem uma interface em três níveis que permite a definição das seqüências e conteúdos do curso. Esta interface gera automaticamente os Modelos da RdPO que inclui a seqüência de controle e as funções de atualizações do modelo do aprendiz, dentro do qual ela tem apenas que ligar explicações, exemplos e exercícios, de acordo com as especificações dadas (veja seção 4). Embora o especialista não especifique como os modelos da RdPO poderão adaptar-se a diferentes aprendizes, esta capacidade está embutida nas condições do modelo do aprendiz inclusive pela interface na transições da RdPO [Postal et al., 2004].

3.1. Exemplo de Unidades Pedagógicas

De acordo com a visão interna do modelo conceitual MATHEMA, o conhecimento associado com cada sub-domínio (PT, PP, DT e DP) é organizado em um ou mais currículos. Cada currículo consiste de um conjunto de UPs ordenados de acordo com pré-requisitos. As UPs associadas com os sub-domínios de Estrutura da Informação no sistema MathTutor são apresentadas abaixo [Bittencourt, 2004]. No exemplo, ambos contextos – teórico e prático – têm as mesmas UPs, o que muda é o ponto de vista sobre o assunto.

- Abstração Procedimental(PT e PP):
  - Procedimentos Primitivos (UP1)
  - Procedimentos Compostos (UP2)
  - Interação e recursão (UP3)
  - Procedimentos de Alta Ordem (UP4)
  - Procedimentos como Argumentos (UP5)
  - Procedimentos como Métodos Gerais (UP6)
  - Procedimentos como Valores Retornados (UP7)
- Abstração de Dados (DT e DP):
  - Dados Compostos
  - Barreiras de Abstração
  - Dados Hierárquicos
  - Seqüências como Interfaces Convencionais
  - Dados Simbólicos
  - Múltiplas Representações para Dados Abstratos
  - Sistemas com Operações Genéricas

Um grafo de pré-requisitos para um possível currículo para o domínio "abstração procedimental teórico" é representado na figura 4. Cada UP está associada com um conjunto de problemas, no sentido que, se o aprendiz está apto a resolver os problemas associados a uma dada UP, o sistema considera que o conteúdo da unidade foi aprendido. Os problemas também são ordenados de acordo com pré-requisitos. Cada problema está associado com um conjunto de páginas HTML interativa que suportam as atividades de resolução de problemas [Postal et al., 2004]. No exemplo, os problemas associados com a UP1, *procedimentos primitivos*, são:

- O que são computadores e linguagens de programação?
- O que são programas e processos?
- Quais são os tipos de dados primitivos?
- Como implementar um procedimento?

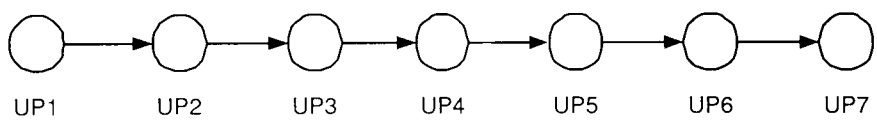

Figura 4: Grafo de Pré-Requisitos para o domínio *abstração procedimental teórico.*

Um possível grafo de pré-requisitos para estes problemas é representado na figura 5. As páginas interativas que implementam um problema têm um formato padrão com alguns controles de navegação, tais como *sair*, *prosseguir*, *repetir*, etc. Estas páginas são de

três tipos: explicações, exemplos e exercícios. Explicações e exemplos são apenas páginas de textos em HTML a serem lidas pelo aprendiz e tipicamente seriam disponibilizadas em diferentes níveis de complexidade para cada conteúdo. Exercícios podem ser questões de (múltipla) escolha, questões que perguntam por alguma resposta simbólica ou numérica ou questões perguntando por alguma implementação *Scheme* [DrScheme, 2004], que precisa da interação com o programa para ser testada [Postal et al., 2004].

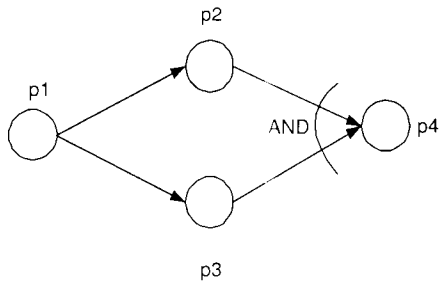

Figura 5: Grafo de Pré-Requisitos para os problemas da UP1 *procedimentos primitivos.*

4. A Interface de Autoria

O propósito da ferramenta de autoria é ajudar o especialista nas seguintes tarefas:

- propor, para cada subdomínio, um conjunto de currículos e especificar cada um deles de acordo com a visão interna, definindo as UPs, os problemas associados e seus pré-requisitos.
- especificar os conteúdos de cada interação suportada pela unidade.

Os mecanismos de uso e atualização do modelo do aprendiz são automaticamente integrados dentro do modelo pedagógico, deixando ao especialista somente a tarefa de oferecer as explicações associadas, exemplos e exercícios. Depois que os modelos de domínio e pedagógico de cada AT são definidos, o Coordenador assume o controle do STI associado.

O mecanismo de autoria proposto é mostrado na figura 6. Inicialmente, o especialista especifica os currículos do curso. Cada currículo é composto por um conjunto de UPs e suas possíveis seqüências de execução. Para especificar cada currículo, o especialista dispõe de uma interface gráfica que permite a construção de grafos. Cada grafo é associado com um currículo e cada nó de um grafo é associado com uma UP. Uma aresta do nó `up1` ao nó `up2` significa que a UP `up2` tem o nó `up1` como pré-requisito. Cada nó pode ter as seguintes arestas de entrada:

Nenhuma: a UP não tem pré-requisitos e pode ser executada em qualquer tempo.

Uma: a UP tem apenas uma UP como pré-requisito e esta uma deve ser executada antes de torná-la habilitada para execução.

Duas ou mais arestas necessárias: a UP tem várias UPs como pré-requisitos e todas devem ser executadas, em qualquer ordem, antes de torná-la habilitada para execução.

Duas ou mais arestas alternativas: a UP tem várias UPs como pré-requisitos mas somente uma delas deve ser executada antes de torná-la habilitada para execução.

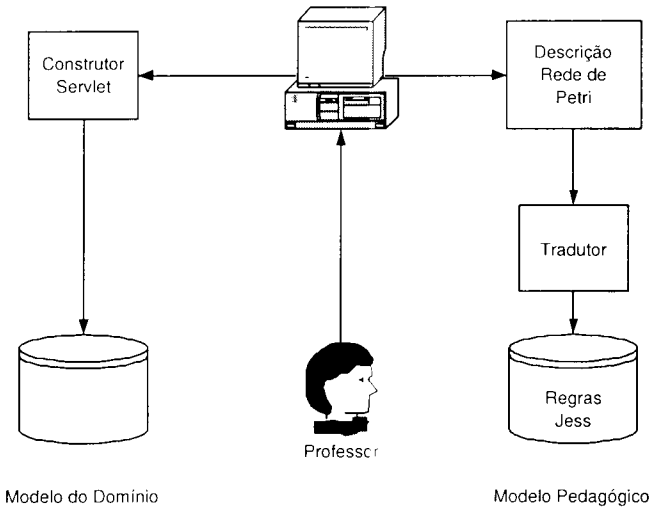

Figura 6: Interface de Autoria

Os nós e os diferentes tipos de arestas podem ser combinadas em um grafo complexo, de acordo com as seqüências pretendidas do curso. A saída do primeiro nível da interface consiste de um grafo representado como uma expressão na linguagem formal definida na figura 7.

```
<graph>::= <prerequisite-list>
<prerequisite-list>::= <prerequisites> | <prerequisite>, <prerequisite-list>
<prerequisites>::= <node>   [<or-list>]
<or-list>::= <or-pred> | <or-pred>, <or-list>
<or-pred>::= <node> | (<and-list>)
<and-list>::= <node> {, <node>}*
```

Figura 7: Linguagem Formal para Definição do Grafo de Pré-Requisitos

O segundo nível da interface permite a definição das UPs. Cada UP consiste de um conjunto de problemas cujas definições são especificadas através de uma interface de texto. A especificação do problema inclui uma questão que o aprendiz deverá estar apto a responder depois que a interação com o problema for terminada e a especificação do número de explicações, exemplos e exercícios que serão associados com o problema. A ordenação do pré-requisito entre os problemas da mesma UP é definida através da mesma interface gráfica usada para definir a ordenação das UPs e é explícitada por uma expressão na mesma linguagem formal da figura 7.

A interface de autoria, baseado na informação obtida nos dois primeiros níveis, gera uma descrição de RdP das seqüências do curso. Na RdP, cada problema é associado com um lugar e cada UP com uma sub-rede. A descrição de RdP é usada como entrada para um compilador que gera automaticamente as regras do sistema especialista que é implementado com a ferramenta JESS. O compilador incorpora na RdP os caminhos condicionais que definem os diferentes tipos de arestas e os procedimentos para atualizar o modelo do aprendiz.

No terceiro nível da interface consta as explicações, exemplos e exercícios que são especificados diretamente dentro da interface ou copiado de um arquivo previamente prepa-

rado pelo especialista. Os textos são incorporados em páginas WEB num formato padrão.

Cada grafo de pré-requisito deve ser transformado em uma RdP que representa o modelo pedagógico. O modelo pedagógico define a estratégia de controle da interação entre o aprendiz e o AT associado com cada subdomínio e especificado pelo especialista na forma de currículo e grafos de UP.

4.1. A estrutura da RdPO - Interface de Autoria

Para representar o modelo pedagógico, foi utilizado uma RdPO. Os lugares da RdPO podem representar qualquer unidade pedagógica. A estrutura da rede é obtida diretamente do grafo especificado pelo professor e a estrutura de dados da ficha é definida pelo modelo do aprendiz adotado. As fichas têm a classe `Apprentice` que está associada com a atividade atual do aprendiz e também com suas possíveis atividades futuras. Uma transição dispara quando o aprendiz termina as interações associadas com seus nós de entrada. A ficha é então movida para os lugares de saída da transição, significando que o aprendiz está pronto para iniciar as atividades associadas a eles.

Para ilustrar a transformação de um grafo de pré-requisitos para a estrutura de uma RdP, observe a figura 8, onde tem-se as possíveis topologias de grafos.

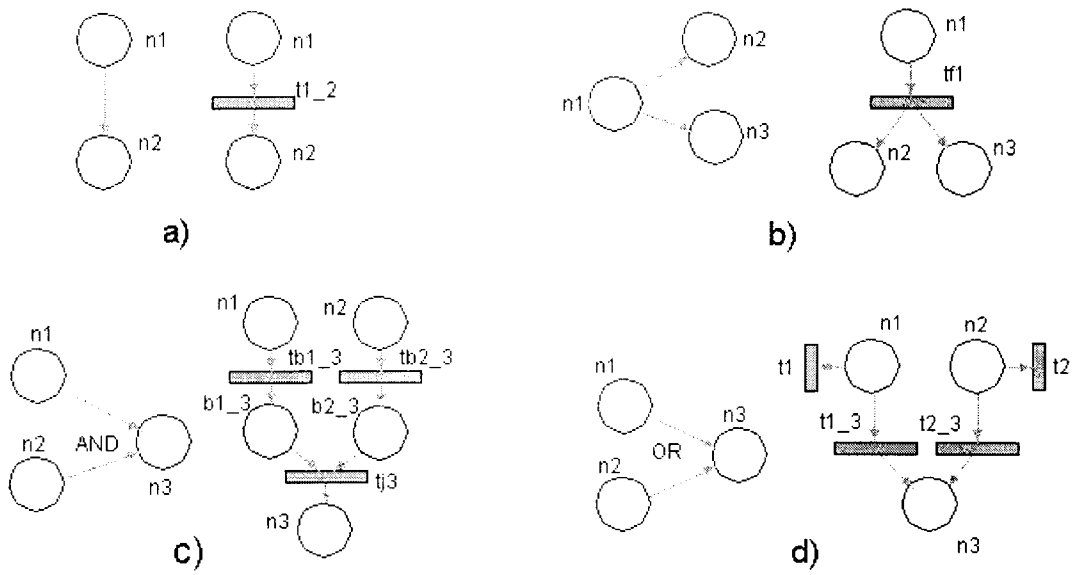

Figura 8: Topologias de grafos

Por exemplo, uma seqüência simples de nós, figura 8(a), é implementada por uma transição (`t_1_2`), um lugar de entrada (`n1`) e um lugar de saída (`n2`). Um nó com duas arestas de saída, figura 8(b), é implementada por uma transição *fork* (`tf_1`) com um lugar de entrada (`n1`) e dois lugares de saída, (`n2` e `n3`). Depois que a atividade associada com o lugar de entrada está terminada, o aprendiz pode fazer a atividade associada com um ou outro lugar de saída.

Um nó com duas arestas de entrada necessárias, figura 8(c), é implementada por uma transição *join* (`tj_3`) com dois lugares de entrada, (`b_1_3` e `b_2_3`), e um lugar de saída,

(n3). Os lugares (b_1_3 e b_2_3) são lugares de *buffers* e não fazem correspondência com UPs ou problemas. Eles são necessários porque depois que a atividade associada ao lugar n1 ou n2 for terminada, a informação deve ser armazenada no buffer correspondente e somente quando a atividade associada com o outro lugar de saída estiver terminada (e também armazenada no buffer correspondente) a atividade associada com o lugar de saída (n3) pode ser realizada. Para isto, nós precisamos duas transições extras, (tb_1_3 e tb_2_3), conectando, respectivamente, os lugares de entrada n1 e n2 aos buffers (b_1_3 e b_2_3).

Finalmente, veja a figura 8(d), um nó com dua s arestas de entrada alternativas é representado por duas transições (t_1_3 e t_2_3), com lugares de entrada n1 e n2, respectivamente, e ambos com lugar de saída n3. Transições t_1 e t_2 são usadas para remover a ficha do aprendiz dos lugares n1 e n2, respectivamente, no caso das atividades associadas a n3 já terem sido realizadas, evitando deste modo a repetição de n3.

4.2. Um Exemplo de Grafo de pré-requisitos para RdP

Seja o exemplo dado pelo grafo de pré-requisitos da figura 9.

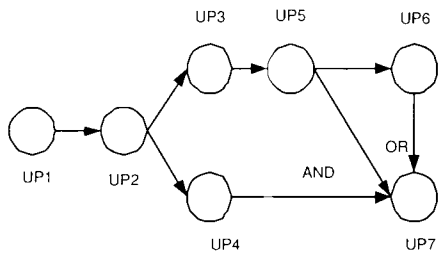

Figura 9: Exemplo de Grafo de Pré-requisitos

Este grafo, pode ser representado pelo seguinte arquivo de entrada:

```
up-pu1 <- [] up-pu2 <- [up-pu1] up-pu3 <- [up-pu2] up-pu4 <-
[up-pu2] up-pu5 <- [up-pu3] up-pu6 <- [up-pu5] up-pu7 <-
[(up-pu4,up-pu5),up-pu6]
```

A RdP obtida, após a aplicação do algoritmo neste arquivo, é dada pela figura 10. Esta rede é obtida a partir das estruturas de dados que o algoritmo obtém (não é o algoritmo que "desenha" a RdP).

Após transformar um grafo de pré-requisitos para uma RdP, o próximo passo é transformar estas regras em uma base de conhecimento. Esta base servirá para gerenciar a interação do aprendiz com o sistema. Para maiores detalhes da implementação da interface de autoria ver [Postal et al., 2004].

5. Agradecimento

As autoras Eliane Pozzebon e Luciana Bolan Frigo agradecem o CNPq Conselho Nacional de Desenvolvimento Científico pela bolsa de doutorado.

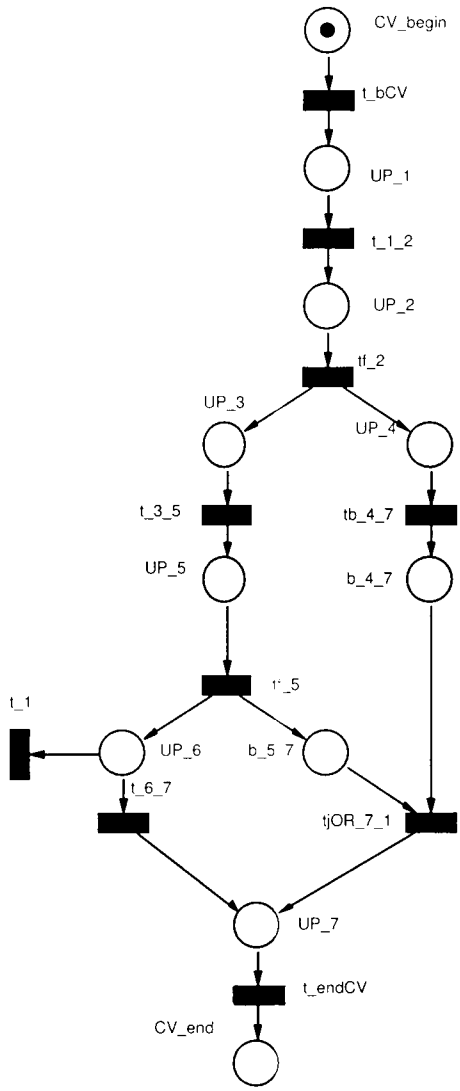

Figura 10: RdP Obtida para o grafo da figura 9

6. Conclusão

O artigo mostra uma alternativa para a estruturação do modelo pedagógico em um Sistema Tutor Inteligente, através de um mecanismo de autoria, cujo objetivo é diminuir a complexidade e conseqüentemente o esforço para a construção de STI. Tal facilidade traduz-se na redução do tempo de desenvolvimento de um curso e do nível de conhecimento especializado.

A principal contribuição da interface de autoria é permitir que o especialista construa um curso independente do domínio de conhecimento, concentrando seus esforços no conteúdo através de uma interface gráfica ou textual. O uso das Redes de Petri trouxeram uma facilidade visual para a modelagem e controle dos processos/ regras que ocorrem paralelamente, revelando informações importantes sobre a estrutura e o comportamento dinâmico do sistema modelado.

A próxima etapa é fazer com que o modelo do aprendiz e o modelo pedagógico trabalhem conjuntamente o que permitirá uma personalização do curso a cada aprendiz além, de uma expansão da interface de autoria alcançando assim os objetivos de uma ferramenta de autoria para STI completa.

Referências

Auberger, M. (1998). Student modeling in educational multimedia titles using agents. Disponível em: http://aif.wu-wien.ac.at/usr/geyers/, Acesso em: 18 maio 1998.

Bittencourt, G. (2004). Fundamentos da estrutura da informação. Página da Internet – www.das.ufsc.br/ gb/fei.

de Barros Costa, E. (1997). *Um Modelo de Ambiente Interativo de Aprendizagem Baseado numa Arquitetura Multi-Agentes*. Tese, Universidade Federal da Paraíba – Engenharia Elétrica, Campina Grande, PB.

DrScheme (2004). Drscheme. Página da Internet – http://www.drscheme.org/.

Frigo, L. B. (2002). Mathtutor - um ambiente interativo multiagente para o ensino de estrutura da informação. Master's thesis, Universidade Federal de Santa Catarina - Engenharia Elétrica, Florianópolis.

Hix, D. and Hartson, H. (1993). *Developing User Interfaces*. Wiley.

Murray, T. (1999). Authoring intelligent tutoring systems: An analysis of the state of the art. In *Journal of Artificial Intelligence and Education*, volume 10.

Park, O. (1998). Functional characteristics of intelligent computer-assisted instruction: Intelligent features. In *Educational Technology*, pages 7–14.

Postal, A., Pozzebon, E., Frigo, L. B., Cardoso, J., and Bittencourt, G. (2004). Mathtutor: A multi-agent intelligent tutoring systems. Artigo enviado para submissão ao AIAI 2004 – Artificial Intelligence Applications and Innovations. Aguardando notificação de aceite.

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. Morgan Kaufmann Publishers.