Integração Disciplinar em um Curso de Graduação com o uso de Ontologias

André Bernardes Pezza[1], Nizam Omar[1], Vilmar Pedro Votre[1]

[1]Universidade Presbiteriana Mackenzie
Rua da Consolação, 930 – 01302-907 – São Paulo – SP – Brasil
andrepezza@andrepezza.com, omar@mackenzie.com.br, fcidir@mackenzie.br

Abstract. *This article describes the use of ontologies to create a broad view of a Computer Science curriculum to promote a higher level of integration between the planned disciplines syllabus and the realized lectures during the completion of a graduation course. The proposed ontology makes it possible to construct a dynamic knowledge base of the course, so it can be constantly updated and adapted to the student's needs.*

Resumo. *Este artigo descreve o uso de ontologias para criar uma ampla visualização de um currículo de Ciência da Computação, capaz de promover um maior nível de integração entre a ementa planejada das disciplinas e o que é efetivamente ensinado no decorrer do curso. Através da ontologia proposta torna-se possível construir uma base de conhecimentos dinâmica sobre o curso, de forma que este possa ser constantemente atualizado e adaptado às necessidades dos alunos.*

1. Introdução

A disseminação da informação, seja por meios eletrônicos ou não, torna cada dia o ensino uma tarefa bastante complexa devido à bagagem inicial dos alunos. Um problema que afeta a qualidade e o aproveitamento nos cursos em geral é o tratamento individual do aluno, necessário para complementar a sua formação. Geralmente os cursos são oferecidos para um conjunto de pessoas. O conteúdo do curso costuma ser baseado em um currículo previamente planejado englobando novos conceitos e considerando uma série de conceitos como pré-requisitos para o acompanhamento do curso. Essa série de conceitos que forma o currículo muitas vezes difere da que é efetivamente ensinada e mais ainda da série aprendida pelos alunos, ou seja, aqueles conceitos que deverão ter sido corretamente assimilados ao término do curso. A questão deixada de lado nessa abordagem é o alto grau de heterogeneidade existente entre os alunos de uma turma. Mesmo alunos que tenham tido uma formação acadêmica semelhante costumam possuir uma gama de conhecimentos diferente, de forma que pode não ser adequado pensar que todos estarão completamente nivelados somente por atenderem a um conjunto básico de pré-requisitos.

Segundo Cardoso e Jandl (1998), alunos de uma mesma classe, submetidos às mesmas condições de ensino, apresentam resultados distintos. Portanto as diferenças existem, e descobrir uma forma de levar em conta essas diferenças ao planejar e ministrar um curso pode ser uma contribuição válida à melhoria da qualidade de ensino e talvez até ao melhor aproveitamento do curso por parte dos alunos.

Segundo Gagné et al. (1992), conhecimento causa uma mudança observável no estudante, habilidades e técnicas devem ser aprendidas separadamente, uma por vez; novas habilidades e/ou técnicas aprendidas devem ser construídas a partir de uma habilidade e/ou técnica aprendida previamente e, portanto, aprendizado e conhecimento são ambos hierárquicos por natureza.

O problema é que algumas vezes fatores como falta de tempo, divergências de horários de trabalho e até mesmo dificuldades de relacionamento interpessoal podem fazer com que a ementa das disciplinas não seja discutida e trabalhada em conjunto, o que acaba causando um aspecto de falta de integração ao curso e com isso prejudicando o processo de construção do conhecimento proposto.

Podem ocorrer ainda problemas tais como falhas na aderência entre as ementas, programas e realização das disciplinas, resultando em disciplinas de um mesmo curso com tópicos desnecessariamente repetidos, e outras por sua vez com tópicos relacionados e que poderiam ser trabalhados em conjunto mas não o são.

Nem sempre é possível o cumprimento do programa disciplinar na sua íntegra por parte dos professores, justamente pelo fato de a sala de aula ser um ambiente de natureza heterogênea, onde não raramente um conjunto de alunos encontra-se em uma situação de desnível em relação aos outros e acaba atrasando o andamento normal do curso, fazendo com que o professor tenha que gastar mais tempo do que o previsto em determinados tópicos e com isso não trabalhe o programa completo, prejudicando assim os alunos que estariam em condições de acompanhar o curso na sua totalidade.

A somatória destes problemas pode acabar causando um efeito negativo sobre o processo de aprendizagem, fazendo algumas vezes com que os alunos venham a se desinteressar pelas disciplinas, dediquem-se menos do que deveriam e em alguns casos cheguem até mesmo a desistir do curso, justamente por terem a impressão de que este não foi suficientemente elaborado e possui falhas em demasia.

A solução proposta consiste no uso de ontologias para construção de uma base de dados dinâmica contendo todo o conhecimento ensinado no decorrer de um curso e, paralelamente a isto, um processo de avaliação contínua e personalizada dos alunos de forma a tornar possível que seja feita uma constante reestruturação da grade curricular a fim de melhor suprir suas necessidades. Através da ontologia proposta seria possível saber quais tópicos presentes no programa disciplinar foram efetivamente ensinados e quais tópicos foram deixados de lado ou até mesmo ensinados mas não aprendidos pelos alunos, já que, segundo Moreira e Masini (2001), o fator que mais influencia a aprendizagem é justamente aquilo que o aluno já sabe.

Com base nesses dados torna-se possível fazer uma análise precisa e constante do andamento do curso, visando sempre o seu aprimoramento e principalmente a sua adaptação para com as necessidades dos estudantes. Em outras palavras, o que se propõe aqui é que o curso deixe de ser uma simples estrutura estática para tornar-se uma estrutura dinâmica, dando ao professor condições de realizar melhor o seu trabalho e concentrar seus esforços para atender a essas necessidades.

O artigo está organizado da seguinte forma. A seção 2 apresenta uma breve explanação sobre ontologias, partindo de uma definição e mostrando posteriormente o processo de construção. Na seção 3 discute-se a ontologia que foi desenvolvida para representar a grade curricular do curso de Ciência da Computação da Universidade

Presbiteriana Mackenzie. Na seção 4 é feito um breve estudo de caso que tem por objetivo exemplificar os benefícios práticos da utilização da ontologia proposta. Na seção 5 comenta-se os resultados esperados, e na seção 6, por fim, são apresentadas algumas sugestões para trabalhos futuros sobre o assunto.

2. Ontologias

Ontologias vêm sendo utilizadas para representação de conhecimento dentro de um determinado domínio.

Segundo Gruber (1993), ontologia pode ser definida como a especificação formal de uma conceituação, ou seja, fornece a descrição de conceitos e relações existentes em um dado domínio com o objetivo de definir um modelo conceitual que reduza ou elimine confusões terminológicas e ofereça uma estrutura de trabalho unificada de entendimento comum sobre o domínio.

O processo de construção de uma ontologia geralmente tende a ser lento e complexo, pois exige um alto nível de conhecimento sobre o domínio a ser representado.

Segundo Noy e McGuinness (2002), a elaboração de uma ontologia envolve os seguintes passos:

- Determinar o domínio e o escopo da ontologia;
- Investigar o reuso de ontologias existentes;
- Listar termos importantes;
- Definir as classes;
- Identificar a hierarquia de classes;
- Definir propriedades das classes.

Uma ontologia é formada por classes, sendo que estas classes podem possuir atributos (também conhecidos como *slots*) e restrições (também chamado de *facets* ou *role restrictions*). Uma classe descreve um conceito dentro de um determinado domínio. As classes também podem possuir subclasses, sendo que estas são utilizadas para representar um conceito mais específico dentro do domínio. Por exemplo, poderíamos imaginar uma classe pessoa que possui as subclasses autor e professor, já que ambos são tipos de pessoas.

Os atributos descrevem as propriedades que poderão ser particulares para cada instância de uma classe. No caso de uma classe pessoa, por exemplo, poderíamos ter os atributos nome, data de nascimento, documento de identidade etc.

Nota-se que é muito fácil fazer uma analogia da ontologia com os conceitos de orientação a objetos utilizados nas linguagens de programação modernas, mas não podemos deixar de salientar uma diferença fundamental entre as duas. Enquanto na orientação a objetos as classes possuem atributos e métodos, que definem respectivamente as propriedades e o comportamento de uma instância, na ontologia as classes não possuem métodos. A princípio pode parecer estranho, mas analisando mais a fundo vemos que isto faz grande sentido, já que a função da ontologia é representar

um conhecimento, e para uma simples representação não precisamos de métodos (que definem ações), mas sim de atributos (que definem as propriedades).

3. O Modelo de Conhecimento de um Curso de Graduação

Para desenvolver este modelo de conhecimento foi utilizada como base a grade curricular do curso de Ciência da Computação da Universidade Presbiteriana Mackenzie, procurando dar ênfase às disciplinas específicas da área de computação, especialmente programação. Mas é importante ressaltar que, por se tratar de um modelo genérico de conhecimento, pode facilmente ser utilizado para representar praticamente qualquer curso que pudermos imaginar.

Durante o desenvolvimento ficou evidente que a criação de uma ontologia é um processo extremamente iterativo, pois por várias vezes foi necessário voltar ao início do processo para corrigir o modelo de acordo com as necessidades que iam surgindo.

Para criação do modelo foi utilizado o software Protege 2000 [The Protege Project, 2000], devido principalmente à sua interface intuitiva, excelente documentação e capacidade de exportar a ontologia para um arquivo padrão XML (no formato RDF Schema), o que aumenta a sua capacidade de integração com outros sistemas.

Uma premissa importante é de que seja possível com este modelo representar as interdependências existentes entre as diversas disciplinas do curso, de forma que possamos relacioná-las entre si e que seja possível descobrir, através de um tópico, quais tópicos dependem deste (ou seja, só podem ser ensinados após o aluno ter o conhecimento deste tópico) e quais tópicos são pré-requisitos para que este possa ser ensinado e devidamente compreendido.

O modelo é composto pelas seguintes classes: `Disciplina`, `Software`, `Tópico`, `Pessoa`, `Obra`, `Organização`, `NumeroTelefone`, `Curso`, `Titulação`, `Credencial` e `Local`. A classe `Pessoa` possui as subclasses `Autor` e `Professor`. Esta separação é interessante, pois normalmente teremos um conjunto comum de atributos para professores e autores (como por exemplo, o nome) e um conjunto de atributos específico para os professores (como por exemplo uma lista das disciplinas ensinadas, a faculdade em que ele leciona, etc). A classe `Obra` possui as subclasses `Livro` e `DocumentoCientifico`, sendo que esta por sua vez possui as subclasses `TeseDoutorado` e `TeseMestrado`. As classes `TeseMestrado` e `TeseDoutorado` possuem inicialmente os mesmos atributos, mas a divisão é interessante pois no futuro pode surgir a necessidade de diferenciá-las, e sempre que fazemos uma modelagem temos que pensar em todos os problemas e necessidades que poderão surgir (e provavelmente surgirão) no futuro. A classe `Organização` possui as subclasses `InstituiçãoGovernamental`, `Empresa` e `EntidadeDeEnsino`, sendo que esta por sua vez possui a subclasse `Universidade` (outras podem ser facilmente acrescentadas conforme necessidade futura). A classe `Local` trata-se de uma adaptação de uma classe de mesmo nome que acompanha o pacote de exemplos do software Protege 2000. Esta adaptação consiste basicamente na tradução para a língua portuguesa e modificação de alguns atributos de forma a regionalizar o conteúdo.

Vamos analisar agora os atributos das principais classes para entender como é feita a armazenagem do conhecimento e como as classes se relacionam entre si.

A classe `Disciplina` possui os seguintes atributos: `software_apoio`, `código`, `aulas_lab_semanais`, `nome`, `bibliografia_complementar`, `carga_horaria`, `professores`, `aulas_teoria_semanais`, `ementa`, `etapa`, `cursos`, `blibliografia_basica` e `topicos`.

nome: contém o nome da disciplina.

software_apoio: lista de instâncias da classe Software, utilizado para indicar os programas que são utilizados na disciplina.

código: código da disciplina, utilizado para controle da faculdade.

aulas_lab_semanais: número de aulas de laboratório em uma semana.

aulas_teoria_semanais: número de aulas teóricas em uma semana.

bibliografia_basica: Lista de instâncias da classe Obra contendo a bibliografia básica da disciplina.

bibliografia_complementar: Lista de instâncias da classe Obra contendo a bibliografia complementar da disciplina.

carga_horaria: número de horas aula.

professores: Lista de instâncias da classe Professores contendo os professores que lecionam na disciplina.

ementa: Um texto com a descrição da disciplina.

etapa: Número indicando em qual semestre ou ano do curso a disciplina é ensinada.

cursos: Lista de cursos que possuem esta disciplina.

topicos: Lista de instâncias da classe Tópico, representando assim os conceitos que são ensinados na disciplina.

A classe `Topico` possui os seguintes atributos: `nome`, `disciplinas`, `professores`, `depende_de`, `pre_requisito_para`.

nome: O nome do Tópico.

disciplinas: Lista de instâncias da classe Disciplina contendo as disciplinas que possuem este tópico.

depende_de: Lista de instâncias da classe Tópico indicando os tópicos que são pré-requisitos para este.

pre_requisito_para: Lista de instâncias da classe Tópico indicando os tópicos que são dependentes deste.

Nota-se que é a partir dos atributos `depende_de` e `pre_requisito_para` que se torna possível montar toda a estrutura de interdependência de conceitos do curso, o que nos permite ter uma visão geral das relações existentes entre as disciplinas e com isso realizar um estudo detalhado de todo o curso, visando detectar possíveis falhas e com isso melhorar a qualidade geral do ensino e o nível de integração entre as matérias.

A classe `Curso` possui os seguintes atributos: `nome`, `tipo`, `area_concentracao`, `organizacao`, `credenciais`, `disciplinas`.

`nome`: O nome do curso.

`area_concentracao`: A área de concentração do curso, como por exemplo exatas, computação etc.

`organizacao`: Instância da classe Organização, indica a qual organização (Colégio, Faculdade, Universidade etc) o curso pertence.

`credenciais`: Lista de instâncias da classe Credencial, contendo os certificados emitidos pelas entidades reguladoras de ensino.

`disciplinas`: Lista de instâncias da classe disciplina, contém todas as disciplinas existentes no curso.

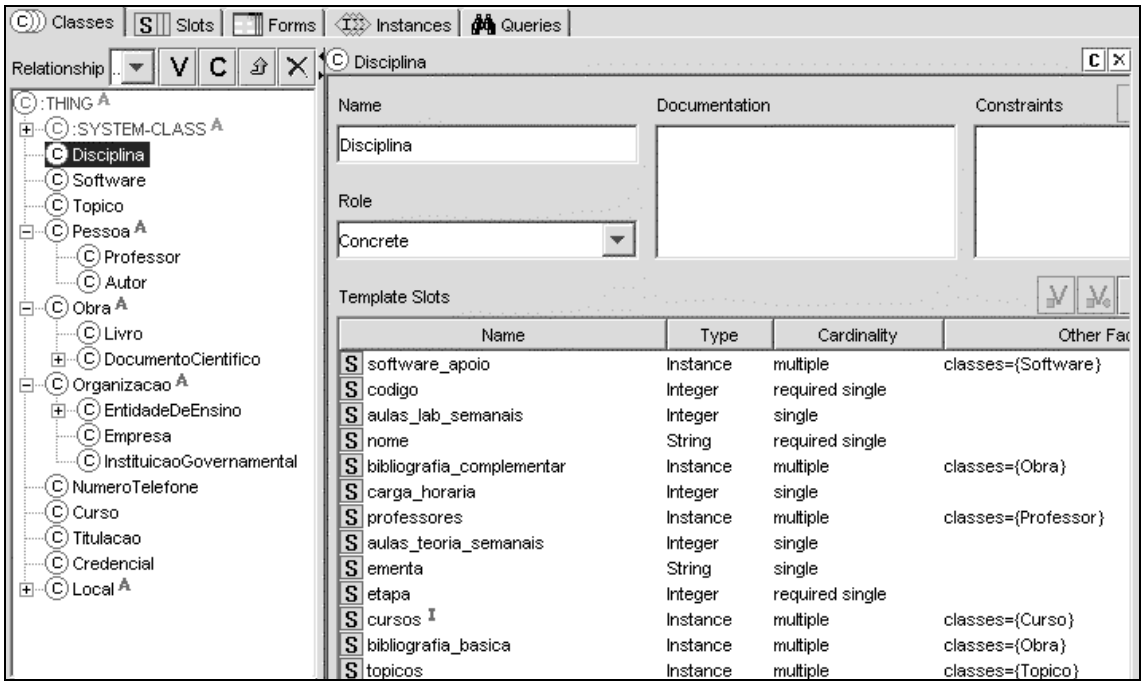

Figura 1 - Tela do Protege 2000 [PRO 2000] exibindo a hierarquia de classes.

4. Estudo de Caso – Aprendizagem de Linguagens de Programação

Como exemplo de uma cadeia de disciplinas interdependentes e inter-relacionadas, podemos citar o aprendizado de linguagens de programação que ocorre direta e indiretamente nas seguintes disciplinas, que normalmente fazem parte de grade curricular de um curso de Ciência da Computação.

Disciplinas Diretas:

- Desenvolvimento de Software Básico
- Desenvolvimento de Software I, II, III
- Desenvolvimento Orientado a Objetos I, II, III
- Linguagem de programação I, II

Disciplinas Indiretas:

- Análise de algoritmos I, II
- Análise Numérica
- Estrutura de dados
- Introdução a Computação
- Lógica matemática

Em cada uma das disciplinas citadas são ensinados diversos conceitos, e é importante ressaltar que estes conceitos dificilmente são vistos de forma isolada. É comum que um mesmo conceito seja trabalhado em diversas disciplinas, geralmente em diversos níveis de profundidade. Seria interessante podermos visualizar as relações de interdependência existentes entre estes conceitos e suas respectivas disciplinas de forma visual, utilizando para tal uma representação em forma de árvore como a sugerida na Figura 2.

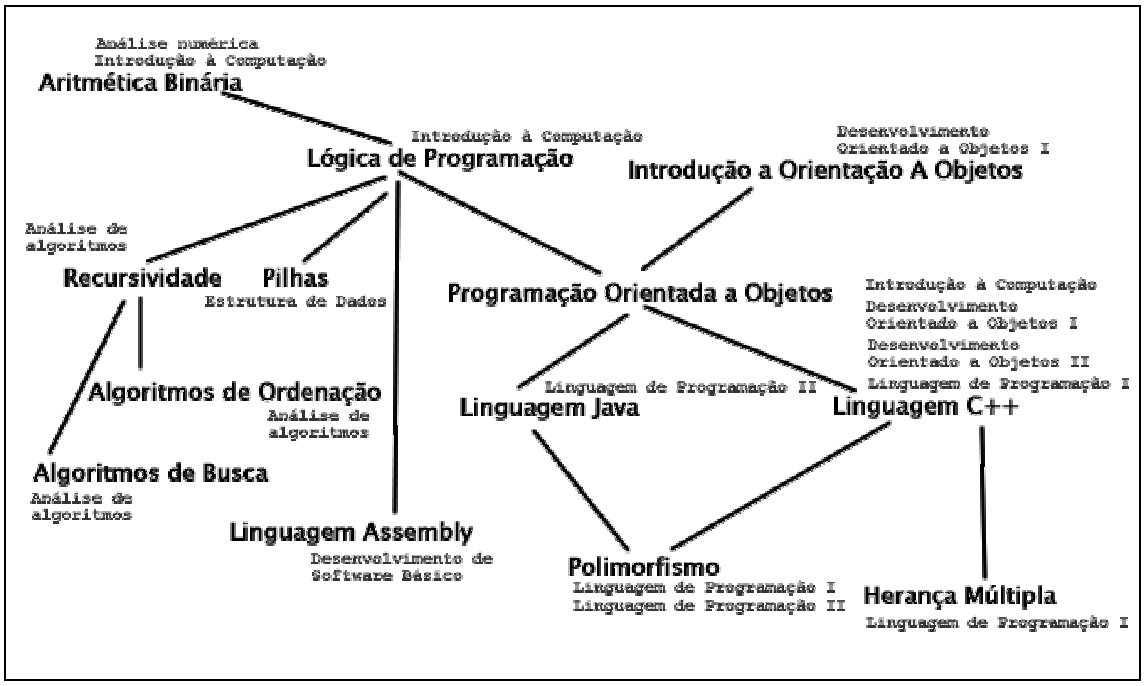

Figura 2 - Árvore de conceitos relacionando os tópicos e as disciplinas.

Na Figura 2 observa-se que as linhas indicam uma relação de dependência, ou seja, cada conceito depende dos conceitos que estão acima deste ligados por uma linha. Em letra de tamanho menor temos uma lista de disciplinas onde o conceito é ensinado. Obviamente trata-se apenas de um exemplo didático, e a quantidade de relações e conceitos foi extremamente reduzida por questões de espaço e simplicidade. Fazendo uma análise subjetiva da figura fica fácil perceber que a Linguagem C++ é ensinada em diversas disciplinas no decorrer do curso. Só com base nesta informação já é possível propor melhorias na integração do curso, promovendo discussões entre todos os professores envolvidos nas disciplinas que ensinam C++ de forma que estes possam coordenar da melhor maneira possível o seu trabalho, procurando não repetir demasiadamente tópicos que já foram completamente assimilados e concentrar seus esforços naqueles tópicos que ainda são obscuros para os alunos.

Mas isso é só o começo. É possível ir mais longe, e propor um sistema que armazene informações sobre cada aluno, de forma que ao final de cada etapa do curso seja possível saber de forma individualizada o que cada aluno realmente aprendeu e quais tópicos não foram aprendidos. Para isso é necessário construir uma base de dados contendo o nível de aquisição de conhecimento de cada um dos alunos do curso, relacionando cada tópico a ser ensinado / aprendido com o nível de habilidade do aluno. A Tabela 1 apresenta um exemplo simplificado de como poderia ser armazenado o nível de aquisição de conhecimento dos alunos. Note que existe um relacionamento entre os tópicos a serem ensinados e cada aluno, associando para tal um determinado valor. Neste exemplo foi utilizada uma escala de 0 a 5, onde 0 significa o desconhecimento total do tópico em questão e 5 indica o domínio pleno e total do mesmo, mas esta escala poderia ser alterada de acordo com as necessidades específicas de cada problema, partindo desde um simples sim ou não (ou seja, sabe ou não sabe) e podendo chegar a um nível de detalhamento tão grande quanto for necessário.

Tabela 1 – Nível de aquisição de conhecimento dos alunos para cada tópico.

	Marta	**Talita**	**Timóteo**
Lógica de Programação	5	4	2
Recursividade	1	2	4
Pilhas	3	0	3
Polimorfismo	2	1	3
Herança Múltipla	1	0	1

Ao término de cada etapa o aluno deve ser avaliado e o seu nível de aquisição de conhecimento deve ser atualizado de acordo com o resultado desta avaliação, de forma que seja possível ter um controle preciso daquilo que cada aluno realmente aprendeu e daquilo para o qual não foi atingido um nível satisfatório de conhecimento. As etapas posteriores do curso devem então ser trabalhadas de acordo com as necessidades de cada um dos alunos, fazendo com que eles recebam conteúdo de forma personalizada e otimizada de acordo com as suas habilidades e dificuldades.

É evidente que nas aulas expositivas torna-se impraticável oferecer um tratamento individual para cada aluno, portanto essas aulas poderiam ser planejadas para melhor atender a maioria dos alunos. Mas nas aulas de laboratório e exercícios é possível alcançar uma individualidade muito grande, sendo possível sugerir exercícios e atividades específicas para cada aluno, de acordo com o seu nível de conhecimento que estaria registrado em um banco de dados.

Na prática este sistema funcionaria da seguinte forma: primeiramente o aluno se identifica ao sistema através do uso de um nome de usuário e senha; o sistema então localiza qual o curso em que este aluno está matriculado e acessa a base de dados contendo todos os tópicos a serem ensinados; o passo seguinte é acessar a base de dados do aluno, que deverá conter o seu nível de aquisição de conhecimento para cada tópico presente no currículo. Com base nesses dados já é possível selecionar as atividades mais indicadas ao aluno em questão, pois o sistema "conhece" os seus pontos fracos e fortes. Então, o sistema deve se concentrar nas atividades em que o aluno possui um menor nível de conhecimento, sem perder tempo com aquilo que o aluno já domina completamente. Isso deverá tornar o processo de aprendizagem mais interessante, pois a

repetição de tópicos já trabalhados e entendidos pode se tornar uma atividade frustrante e acabar causando um certo desinteresse por parte do aluno. Ao término da atividade, o nível de aquisição de conhecimento do aluno é atualizado para refletir o progresso do aprendizado.

5. Considerações Finais e Aplicações no Ensino a Distância

O uso de ontologias mostrou-se apropriado para representar a grade curricular de um curso de graduação, tornando possível visualizar todo o curso de uma forma simples e integrada. Com isso pode-se realizar estudos detalhados visando melhorar a qualidade do curso e minimizar eventuais problemas. Diversas queixas muito comuns, como falta de integração entre as disciplinas, repetição de tópicos durante o curso, tópicos que deveriam ter sido vistos mas não foram, e muitas outras, podem ser facilmente detectadas e corrigidas utilizando o modelo de representação proposto.

Nota-se também que é extremamente animadora a possibilidade de manter uma avaliação contínua e individual de cada aluno, e através disso poder constantemente readaptar o curso para que este melhor atenda às suas necessidades. Afinal de contas os alunos são a razão da existência de um curso e, portanto, qualquer esforço no sentido de melhor atendê-los nunca será em vão.

Além das aulas de laboratório e exercícios, o ensino a distância via Internet também parece ser uma área promissora para aplicação do sistema proposto, visto que, ao contrário do que normalmente ocorre em uma aula presencial, pode-se ter vários alunos realizando a mesma disciplina simultaneamente e ainda assim recebendo conteúdo completamente personalizado para suas necessidades. E melhor ainda, ao final do curso pode-se visualizar todo o progresso que foi obtido pelo aluno durante a sessão, comparando o seu nível de aquisição de conhecimento inicial com o final e vendo exatamente em quais tópicos o aluno obteve melhor aproveitamento e quais tópicos ainda deixaram a desejar. Se por um lado perde-se um pouco com a falta do contato pessoal entre professor e aluno, por outro lado ganha-se um nível de individualização que não seria viável em aulas presenciais, a não ser, é claro, que esteja-se falando de uma aula particular. E é justamente com esta individualização do conteúdo de forma específica para cada aluno é que torna-se possível minimizar as possíveis dificuldades existentes em cursos não-presenciais e com isso contribuir de certa forma para uma melhor qualidade de ensino e um melhor aproveitamento dos cursos por parte dos alunos. A tendência é de que a quantidade de cursos oferecidos via Internet venha a crescer cada vez mais nos próximos anos, portanto qualquer trabalho no sentido de melhorar a qualidade desses cursos é mais do que nunca necessário e bem-vindo.

6. Sugestões para Trabalhos Futuros

O modelo de ontologia proposto foi desenvolvido levando-se em conta as necessidades de um curso de graduação, mais especificamente do curso de Ciência da Computação da Universidade Presbiteriana Mackenzie. Mas nada impede que este modelo seja aprimorado (acrescentando-se novos relacionamentos) ou até mesmo expandido e adaptado (criando-se novas classes e atributos) para que possa ser aplicado a outros domínios de interesse.

Atualmente existem algumas linguagens baseadas em XML para a representação de ontologias, e um grande esforço para criar uma especificação padrão que possa ser

entendida por qualquer sistema e compartilhada pela Internet. Seria interessante desenvolver um estudo mais aprofundado sobre este assunto a fim de que a ontologia criada possa ser utilizada nos mais diversos sistemas de acordo com as necessidades e particularidades de cada situação.

Outro ponto interessante seria utilizar o nível de aquisição de conhecimento do aluno como uma forma de direcionamento profissional, já que é possível saber em quais tópicos do curso o aluno se saiu melhor e com isso orientá-lo para que desenvolva cada vez mais suas habilidades nas áreas com as quais possui maior facilidade de entendimento.

Referências Bibliográficas

Cardoso, S. M. V. e Jandl, Peter (1998) Estilos de Aprendizagem: Aprender a Aprender.

Gagné, Robert M., Briggs, Leslie J. Wager, Walter W. Principles of Instructional Design. Fort Worth, Harcourt Brace & Company, 1992.

Gruber, T.R. (1993). A Translation Approach to Portable Ontology Specification.

Moreira, Marco A., Masini, Elcie F. S. (2001). Aprendizagem Significativa: A Teoria de David Ausubel. São Paulo, Centauro.

Noy, Natalya F., McGuinness, Deborah L. (2002) "Ontology Development 101: A Guide to Creating Your First Ontology". Disponível em: <http://protege.stanford.edu/publications/ontology_development/ontology101.pdf>. Acesso em: 10 março 2003.

The Protege Project (2000). Disponível em: <http://protege.stanford.edu/>. Acesso em: 10 março 2003.