Um Modelo de Referência para Ambientes Virtuais de Aprendizagem na Web: Aproximando as Perspectivas do Autor e do Desenvolvedor

Hemilis Joyse Barbosa Rocha[1]

[1]Instituto de Computação - Universidade Federal do Alagoas (UFAL)

Evandro de Barros Costa[1] (Orientador)

Patrick Henrique da Silva Brito[1] (Orientador)

Abstract. *In this work we study the learning management systems (LMSs) and propose a reference model, serving to both analysis and synthesis of LMSs. Such model has been implemented and evaluated, being composed of two representations and facilities specially oriented to two important actors in LMS development. These actors are the author and the developer, where to the first a concept map is offered in order to help educators to select services that will compose the indended LMS. A flexible reference architecture is dedicated to the developers in order to help them to implement the user´s requirements. Additionally, a knowledge-based recommend system is ready to automatically approximate these two representations. Hence, through rules that map the needs of the author in Project decisions to developers, the system may recommend an existing LMS to the author or otherwise even recommend a new architecture. Furthermore, the proposed model was evaluated in the task of constructing a specific LMS by showing positive results.*

Resumo. *Neste trabalho investiga-se os ambientes virtuais de aprendizagem e propõe-se um modelo de referência, servindo tanto ao papel de análise de AVAs quanto o de síntese, servindo para construção de AVAs específicos. Tal modelo foi implementado e avaliado, sendo constituído de duas representações e facilidades especialmente orientadas a dois atores importantes no desenvolvimento de um AVA, quais sejam: autor e desenvolvedor, onde ao primeiro dedica-se um mapa conceitual destinado a auxiliar educadores a escolher serviços que irão compor o AVA que atenda suas necessidades, e ao segundo, uma arquitetura de referência flexível que se presta a ajudar os desenvolvedores a implementar as demandas dos usuários. Ademais, oferece-se um sistema de recomendação baseado em conhecimento que busca automaticamente aproximar essas duas representações. Assim, através de regras que mapeiam as necessidades do autor em decisões de projetos para o desenvolvedor, o sistema pode recomendar ao autor um AVA já existente ou uma nova arquitetura caso não exista uma que atenda o conjunto de funcionalidades escolhidos pelo autor. Além disso, o modelo proposto foi avaliado na construção de um AVA específico e mostrou resultados positivos.*

1. Introdução

Ambientes Virtuais de Aprendizagem (AVA) são sistemas baseados na web, destinados a apoiar atividades educacionais nas modalidades presencial e a distância. Atualmente várias plataformas de AVA já foram desenvolvidas e disponibilizadas, tendo com isso atraído uma quantidade significativa e diversificada de usuários. Parte desses usuários são educadores já experientes e possuem um entendimento bem refinado e crítico sobre a efetividade de um AVA, ou mesmo tem expectativas sobre quais requisitos um determinado AVA deveria satisfazer, principalmente considerando os avanços tecnológicos, possibilitando a viabilização de diferentes propostas pedagógicas. Nesse sentido, observam-se lacunas importantes no cenário dos AVA existentes, levando-se a insatisfações de seus usuários, tanto com alguns dos recursos disponíveis quanto com os que são interessantes, mas ainda não oferecidos.

Por outra perspectiva, observa-se que ainda não há facilidades e ferramentas de software apropriadas para orientar uma participação efetiva dos educadores no desenvolvimento destes ambientes. Faz-se, então, necessário organizar e representar apropriadamente o mencionado entendimento e conhecimento sobre AVA e suas possibilidades, investindo-se, por exemplo, na elaboração de uma arquitetura de referência, baseada nos AVA, para dar suporte ao processo de desenvolvimento desses ambientes.

Para motivar este tema, algumas questões emergem imediatamente, como por exemplo: 1- Qual seria a representação mais adequada em termos de expressividade e capacidade de leitura de um modelo conceitual do AVA para apoiar os educadores no desenvolvimento de um AVA? 2- Qual seria a representação mais adequada da arquitetura de AVA para apoiar o desenvolvedor de software? 3- Como aproximar a representação do educador da representação do desenvolvedor de software?

Nesse sentido, um possível resposta para o questionamento 1, é o trabalho proposto em Crespo et al (1998) focalizou tais aspectos, procurando caracterizar em um modelo conceitual uma representação que servia como uma referência para os AVAs existentes naquele momento. Entretanto, apesar do mérito dessa proposta, atualmente ela já não reflete o estágio corrente desses ambientes, nem do arsenal tecnológico do momento, além disso, apresenta apenas um modelo conceitual e não se preocupa com requisitos. Quanto ao questionamento 2, uma resposta encontrada na literatura é a proposta por Sweeney (2008). No entanto, este trabalho não considera requisitos não-funcionais e não faz uma análise nos AVAs existentes. Nesta dissertação o detalhamento interno dos componentes arquiteturais, foi feito de maneira sistemática, utilizando o processo UML Components, com base no mapa conceitual. Já esse trabalho relacionado não deixou claro como os "sub-componentes" foram identificados. Além do mais, vale ressaltar que uma das contribuições mais importantes desta dissertação é o envolvimento efetivo do educador no processo de desenvolvimento do AVA, enquanto que neste trabalho relacionado não houve essa preocupação. Apesar existirem trabalhos relacionados aos dois primeiros questionamentos, até o momento, não foi encontrado na literatura um trabalho que se proponha a aproximar representação do educador da representação do desenvolvedor de software.

O foco desta pesquisa concentra-se em abordar os problemas relacionados à falta de elementos para compreensão de recursos e a pouca de flexibilidade dos AVAs atuais

com respeito à necessidade de customização em diversos aspectos e adaptação de soluções particulares, dificuldade para integração com outras ferramentas e dados externos aos AVAs. Além disso, o ambiente deve oferecer segurança de acesso tanto aos alunos quanto aos professores, deve permitir escalar facilmente o número de usuários do sistema e deve ser acessível a partir de qualquer sistema operacional. Tais requisitos podem ser encarados como desafios a serem alcançados por sistemas educacionais, para atender de fato ao que se espera deles: apoio ao ensino de qualidade e em larga escala.

Assim, propõe-se nesta dissertação o uma resposta para amenizar parte dos problemas existentes e para solucionar os três questionamentos levantados anteriormente. Para atender o problema abordado no 1 e 2 questionamentos, definiu-se um modelo de referência que conta com um mapa conceitual e uma arquitetura de software. O mapa conceitual é destinado a auxiliar os educadores na avaliação e desenvolvimento de um AVA que atenda suas necessidades. A arquitetura de referencia se presta a ajudar os desenvolvedores a implementar as demandas dos usuários. Assim, pretende-se dispor de um arcabouço adequado para realizar avaliações e comparações entre AVAs, servindo para orientar e viabilizar o desenvolvimento de novos ambientes atendendo demandas específicas de educadores. Este modelo foi avaliado sob três perspectivas complementares: (1) Análise comparativa de um conjunto de AVAs; (2) Síntese de um AVA ideal, através de um estudo de caso envolvendo a refatoração da arquitetura do AVA sakai, em função das necessidades do autor; e (3) Recomendação de um AVA mais adequado, dentre os analisados em função dos requisitos de autor.

Quanto ao questionamento 3, esta dissertação propõe um sistema de recomendação para viabilizar a transformação da visão do educador (o mapa conceitual) na visão do desenvolvedor (a arquitetura de referência e seus detalhamentos). Este sistema mapeia cada escolha do usuário educador, uma arquitetura de software com seus detalhamentos, para o desenvolvedor.

2. Modelo de Referência para Ambientes Virtuais de Aprendizagem

Devido aos questionamentos levantados na seção anterior, neste capítulo será discutido um modelo de referência para AVA. Este modelo é composto por um mapa conceitual, que contém os requisitos funcionais de um AVA, e uma arquitetura de software, que foi elabora seguindo tanto os requisitos não-funcionais quanto os funcionais. Assim, o modelo servirá como base de conhecimento para o sistema de recomendação que através das escolhas feitas pelo professor, recomenda uma arquitetura de software e seus detalhamentos em um diagrama de componentes utilizando o processo de UML components (CHEESMAN e DANIELS, 2000). A arquitetura recomendada será utilizada no desenvolvimento de um novo AVA.

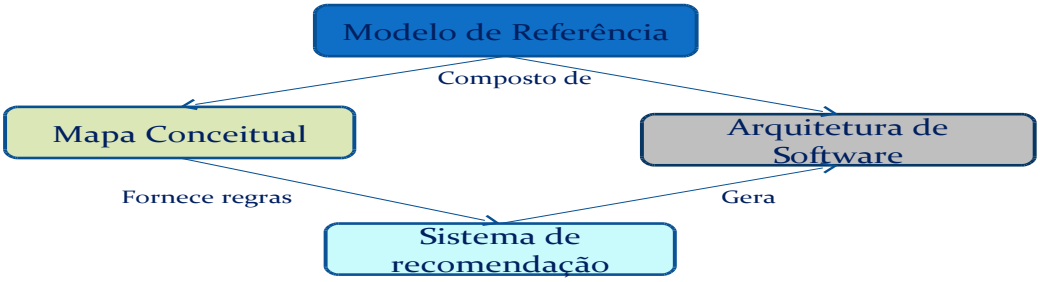

Figura 1. Visão Geral da Dissertação

2.1 Mapa Conceitual de Referência

A concepção do mapa conceitual proposto seguiu uma metodologia que, em relação aos objetivos específicos, é classificada como uma pesquisa exploratória. Este tipo de pesquisa tem como objetivo proporcionar maior familiaridade com o problema (GIL, 1991). Além disso, ela pode utiliza fontes que darão base ao assunto abordado, como é o caso da pesquisa bibliográfica e das entrevistas com pessoas que tiveram experiências práticas com o problema pesquisado. Quanto ao delineamento, a pesquisa classificou-se em: pesquisa bibliográfica, levantamento, a pesquisa experimental (GIL, 1991).

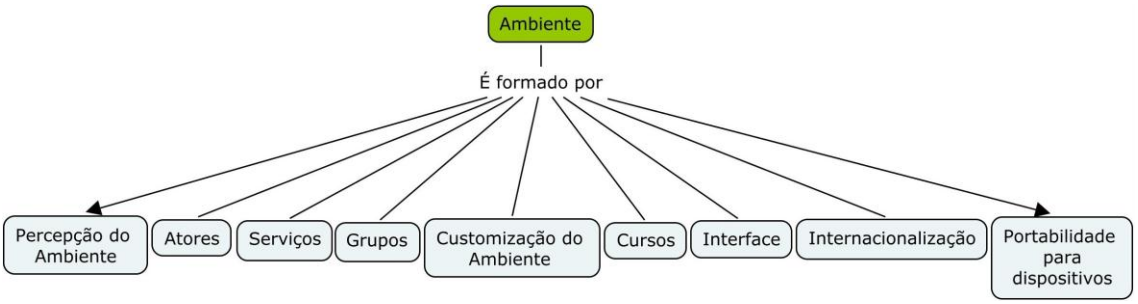

Figura 2. Modelo Conceitual do Ambiente

O mapa conceitual apresentado na Figura 2 é formado por nove conceitos que idealmente deveriam ser considerados no desenvolvimento de um AVA. Cada conceito pode ser decomposto em subconceitos.

i. **Conceito Atores:** Este conceito considera a existência dos atores que interagem com o ambiente, como, por exemplo, professor, estudante e tutor.

ii. **Conceito Interface:** Responsável por avaliar a interface do AVA no que diz respeito a aspectos de usabilidade e design.

iii. **Conceito Cursos:** Busca analisar se o ambiente oferece suporte a estrutura e organização de cursos.

iv. **Conceito Percepção do Ambiente:** Neste conceito são tratados aspectos de percepção de tarefas e atividades, individuais e do grupo, como também a percepção social dentro do ambiente.

v. **Conceito Customização do Ambiente:** Este subconceito abrange todas as mudanças que podem ocorrer tanto na organização quanto no conteúdo e interface do ambiente.

vi. **Conceito Portabilidade de Dispositivos:** Indica se o ambiente pode ser acessado através de diferentes dispositivos, por exemplo, dispositivos móveis, celulares e TV Digital.

vii. **Conceito Internacionalização:** Trata-se da capacidade do ambiente oferecer suporte para que seus cursos sejam adaptados para diferentes localidades, que possuam, por exemplo, formatos de data e moeda diferenciados, e para diferentes idiomas.

viii. **Conceito Serviços:** São os recursos utilizados no desenvolvimento do curso, podem ser classificados como: serviços administrativos, serviços didáticos, serviços de avaliação: possuem como finalidade avaliar o desempenho do aluno,

serviços de comunicação, serviços individuais, serviços de grupo. Os serviços também podem ser classificados como: externo ou interno, por exemplo, um serviço pode ser didático e interno ao mesmo tempo. Além disso, cada tipo de serviço citado anteriormente pode ser categorizado como: serviço interno ou externo ao AVA.

ix. **Conceito Grupos:** Este conceito trata da possibilidade de formação e organização de grupos entre os usuários do ambiente. Neste conceito são considerados quatro aspectos: definição do grupo: simples (não possui subgrupos) ou hierárquico (possui subgrupos); controle de visibilidade de grupos; e interação intergrupos: possibilidade de interação, cooperação e colaboração entre grupos.

Como pode ser observado, esta seção apresentou o mapa conceitual que reflete as características funcionais do AVA. No entanto, essas características não são suficientes no desenvolvimento de um AVA. Por isso, na próxima seção, será discutido o desenvolvimento de uma arquitetura de software de referência para AVA, que foi elabora seguindo os requisitos não-funcionais. Com isso, serão consideradas características tão importantes como escalabilidade.

2.2 Arquitetura de Referência para Ambientes Virtuais de Aprendizagem

O projeto arquitetural para a solução proposta objetivou inicialmente a especificação de uma arquitetura de referência que atendesse os requisitos de qualidade definidos pelo modelo ISO 9126, que se são tratados na Seção 2.2.1. Para isso, foi seguido um processo sistemático, baseado na ideia de catálogos de estilos e padrões arquiteturais.

Inicialmente, foram analisados estilos e padrões arquiteturais clássicos relatados na literatura (Shaw e Garlan, 1996), (Bass, Clements e Kazman, 2003), (Buschman et al., 1996). Dadas às características de cada estilo e padrão arquitetural, foi feita uma associação entre as características de cada estilo e os requisitos de qualidade que os mesmos favorecem e prejudicam. Em seguida, foi selecionado um estilo arquitetural como referência básica para iniciar o projeto da arquitetura. No contexto da solução proposta nesse trabalho, foi adotado o padrão arquitetural MVC (Shaw e Garlan, 1996) como arquitetura inicial. A principal razão para adoção desse padrão como arquitetura inicial é o fato dele ser fortemente indicado para sistemas de informação (Buschman et al., 1996), isto é, sistemas cujas funcionalidades normalmente envolvem consulta e atualização de banco de dados. Finalmente, foi realizado um refinamento progressivo dessa arquitetura inicial.

2.2.1. Requisitos não-funcionais

Do ponto de vista dos requisitos de qualidade do sistema, foi realizada uma análise dos requisitos não-funcionais definidos pelo modelo ISO 9126, que refere-se ao desenvolvimento de software. Em seguida, os requisitos foram classificados em três níveis de prioridade: prioridade alta, prioridade média e prioridade baixa. Para ilustrar melhor os requisitos de qualidade, assim como a importância de cada um no contexto de educação, será apresentado um cenário exemplificando cada requisito.

Alguns requisitos de qualidade com prioridade alta, considerados essenciais, para o modelo de referência proposto foram:

i) **Flexibilidade:** o ambiente deve ser personalizável de acordo com o usuário. Um exemplo dessa adaptação poderia ser o fato de, enquanto o AVA apresenta para um aluno a ferramenta de Fórum em destaque, para outro, apresenta um chat.

ii) **Interoperabilidade:** o ambiente deve ser capaz de estabelecer comunicação com outros ambientes ou com outros aplicativos. Essa é uma das grandes tendências e um dos principais desafios para a área de AVA: integração entre o AVA e o ambiente externo, tais como redes sociais e outros AVA.

Um exemplo de requisito de qualidade com prioridade média, considerados importantes, para o modelo de referência proposto foi:

i) **Facilidade de evolução:** a lógica do negócio costuma ser alterada com frequência. O custo de se trocar algum componente específico do sistema deve ser minimizado. Um cenário no contexto de educação poderia ser o caso onde a funcionalidade de submissão de tarefas tem sua lógica alterada.

Um exemplo de requisito de qualidade com prioridade baixa, considerados desejáveis, para o modelo de referência proposto foi:

i) **Confinamento de falhas**: Caso ocorra alguma falha no ambiente, elas devem ser isoladas de modo que a correção deve ser feita apenas naquele módulo da falha.

2.2.2 Arquitetura de Referência

Nesta seção é apresentada a arquitetura de referência fruto da execução do processo sistemático. A Figura 2 apresenta a visão geral da arquitetura de referência final. Como pode ser percebido, a arquitetura definida possui um estilo arquitetural heterogêneo, baseado no MVC e que combina diversos estilos arquiteturais de maneira complementar. A seguir, é apresentada uma descrição de como, onde e porque cada estilo e padrão arquitetural foi utilizado na arquitetura de referência proposta:

Padrão arquitetural MVC. Esse padrão arquitetural foi adotado como base para iniciar o projeto da arquitetura de referência para AVAs. O padrão MVC apresenta três módulos básicos (Buschman et al., 1996): model, controller e view. Sendo assim, considerou-se que esse padrão arquitetural favorece a implantação do requisito de qualidade “Usabilidade”.

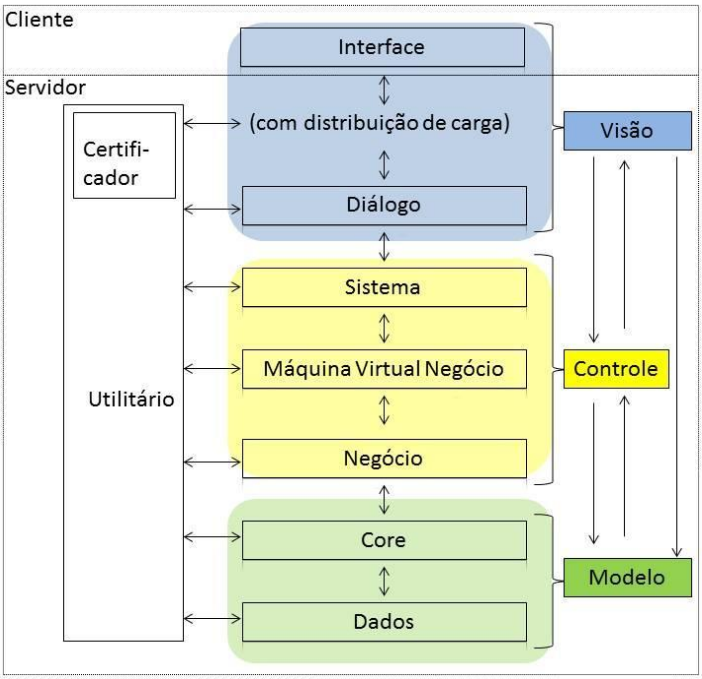

Figura 3. Visão Geral da Arquitetura de Referência Proposta

Componentes Independentes (Shaw e Garlan, 1996). Através da utilização deste padrão é possível obter uma alta escalabilidade. Porém, para que esse potencial seja aproveitado, a arquitetura baseada em componentes independentes deve dispor de conectores apropriados que, quando necessário, desempenhem o papel de distribuidores de carga. Outra potencialidade desse estilo arquitetural refere-se ao requisito de "Segurança de acesso", mas para isso os conectores também possuem um papel de destaque, desempenhando papel de controle de acesso e criptografia centralizados. Finalmente, o fato dos componentes possuírem interfaces bem definidas e interagirem entre si exclusivamente através dessas suas interfaces proporciona um baixo acoplamento. Esse baixo acoplamento beneficia direta e indiretamente outros requisitos de qualidade muito relacionados entre si: "Flexibilidade (funcionalidade flexível)", "Facilidade de evolução" e "Confinamento de falhas". Quanto menos acoplado for o sistema, mais fácil será trocar suas partes e, consequentemente, flexibilizar e evoluir suas funcionalidades.

Camadas. A arquitetura de referência proposta também foi influenciada pelo estilo arquitetural em camadas. Como pode ser observado na Figura 2, basicamente, o estilo em camadas atuou em três aspectos complementares do sistema: (1) Definir uma camada específica para definir a forma de interação do sistema com o usuário que o está utilizando no momento (camada de Diálogo). Sendo assim, favorece os requisitos de qualidade de "Usabilidade" e "Flexibilidade (usabilidade flexível)"; (2) Separar funcionalidades básicas do domínio (camada de Negócio) das funcionalidades específicas do sistema (camada de Sistema). Essa separação visa tanto promover a reusabilidade e facilidade de evolução; (3) Além do mais, foi especificada uma camada voltada para sistemas legados (camada Core). Esta camada possibilita que funcionalidades existentes e implementadas em outros sistemas ou linguagens de programação sejam encapsuladas por funcionalidades básicas do Negócio. Essa

representação explícita dos sistemas legados facilita a integração com ferramentas externas, tais como redes sociais e outros AVAs e promove o requisito de qualidade de "Interoperabilidade".

Máquina Virtual. Para facilitar a evolução de forma considerável e favorecer inclusive a atividade de implantação dessas alterações. Como pode ser observado na Figura 2, a ideia geral por traz do estilo arquitetural de máquina virtual é: (1) localizar o ponto crítico da lógica de negócio; e (2) desacoplar essa lógica dos componentes do sistema, fazendo-a acessível por especialistas do domínio e não necessariamente de computação.

A arquitetura de software apresentada nesta seção é considerada ortogonal com os requisitos funcionais de referência apresentados na Seção 2.2.1, ou seja, a arquitetura de software não deve interferir nas funcionalidades do sistema, mas interfere na qualidade do sistema com atributos especificados como requisitos não-funcionais. No contexto do método proposto, a arquitetura é apoiada por um sistema de recomendação que será apresentado na próxima seção.

3. Sistema de Recomendação

Como já foi anunciado nas seções anteriores, o desenvolvimento do mapa conceitual justificou-se pela inserção do educador no desenvolvimento de um AVA que atendesse suas necessidades. Porém, apesar da relevância desse primeiro auxílio, ainda há uma distância considerável entre as decisões de preferência no mapa conceitual e as decisões de projeto na arquitetura de software. Por essa razão, nesta seção será apresentado um sistema de recomendação baseado em conhecimento, cujas regras foram extraídas do mapa conceitual apresentado na Seção 2.1 e dos requisitos não-funcionais apresentados na Seção 2.2.1. O objetivo desse sistema é facilitar o mapeamento entre as decisões do educador e o impacto disso para o desenvolvedor de software.

Ao ser executado, o sistema de recomendação apresenta uma série de perguntas, que deverão ser respondidas pelo usuário. As primeiras perguntas estão relacionadas aos requisitos não-funcionais, portanto a partir delas será gerada arquitetura similar a da Seção 2.2.2. No entanto, a arquitetura resultante do sistema poderá atender a qualquer sub-conjunto de requisitos, dependendo do que foi respondido ao sistema. Após a definição da arquitetura abstrata, derivada da arquitetura de referência, as perguntas seguintes são voltadas aos conceitos relativos a requisitos funcionais, extraídos no mapa conceitual, e têm o propósito de gerar a arquitetura detalhada, sendo esta um diagrama em UML *components*. Assim, baseado na base de conhecimento e nas informações passadas pelo usuário o sistema inferirá tanto a arquitetura abstrata quanto o detalhamento em UML *components* com todos os componentes, conectores e interfaces que o diagrama de componentes deve possuir. Então, uma vez recomendada à arquitetura e seus devidos detalhamentos, tal estrutura servirá como subsidio para o desenvolvedor durante as demais fases do desenvolvimento do LMS.

Para o desenvolvimento do sistema de recomendação foi utilizado o Inabit, um framework que oferece recursos para a criação de sistemas baseados em conhecimento. Sendo assim, a seguir, serão apresentadas algumas telas do sistema. Primeiramente, algumas telas que servem para recomendar a arquitetura e depois as que recomendam o detalhamento.

Figura 4. Sistema de Recomendação - Tela 1

Na tela apresentada na Figura 4, o sistema deseja saber do usuário se deve acrescentar à arquitetura o requisito Facilidade de evolução da Seção 2.2.1 a arquitetura. Caso a resposta seja positiva, ele adiciona a camada de máquina virtual à arquitetura. Em outra pergunta, caso essa característica precise ser combinada com alto desempenho, à arquitetura também tem que conter um sistema de cache das regras de negócio contidas na máquina virtual.

Figura 5. Sistema de Recomendação - Tela 2

A tela da Figura 5 é apresentada ao usuário o sistema que saber se deve acrescentar o recurso de grupo ao diagrama de componentes. Caso o usuário clique em "Yes", o sistema adicionará os componentes GrupoMgr, OpGrupo, os conectores connGrupoPer e connOpGrupoCore e as interfaces IGrupoReq, IGrupoMgr, IOpGrupo à arquitetura detalhada.

```
r0: if valid=true then true=ask("OpComunicao","Deseja utilizar serviços de comunicação?", "componente");
r1: if OpComunicao = "componente" then ConnOpComunicaCore="conector";
r2: if ConnOpComunicaCore = "conector" then ComunicacaoMgr ="componente";
r3: if ComunicacaoMgr ="componente" then connComunicaPer = "conector";
r4: if connComunicaPer = "conector" then IOpComunicacao = "interface";
r5: if IOpComunicacao = "interface" then IOpComunicaReq = "interface";
r6: if IOpComunicaReq = "interface" then IcomunicacaoMgr = "interface";
r7: if IcomunicacaoMgr = "interface" then IComunicacaoReq = "interface";
r8: if valid=comunicacao then comunicacao=ask("OpGrupo","Deseja utilizar formação de grupos no ambiente?", "componente");
r9: if OpGrupo = "componente" then GrupoMgr="componente";
r10: if GrupoMgr="componente" then connGrupoPer="conector";
r11: if connGrupoPer="conector" then connOpGrupoCore="conector";
r12: if connOpGrupoCore="conector" then IGrupoReq="interface";
r13: if IGrupoReq="interface" then IGrupoMgr="interface";
r14: if IGrupoMgr="interface" then IOpGrupo="interface";
r15: if IOpGrupo="interface" then hierarquia=ask("IOpHierarquia","Deseja utilizar hierarquias entre grupos?", "interface");
r16: if IOpGrupo="interface" then visibilidade=ask("IOpVisibilidade","Deseja que os grupos possuam controle de visibilidade?", "interface");
r17: if IOpGrupo="interface" then moderar=ask("IOpModerar","Deseja que apenas os professores possam criar grupos?", "interface");
```

Figura 6. Base de Conhecimento

A Figura 6 mostra algumas regras, criadas a partir do mapa conceitual, utilizadas nas inferências realizadas pelo *Inabit.*

4. Considerações Finais

Um dos mais importantes objetivos deste trabalho foi aproximar mais o educador do desenvolvimento do AVA que ele irá utilizar. Assim, para que isso fosse possível, neste trabalho foi apresentado, um mapa conceitual contemplando características dos principais AVA. Assim, este mapa conceitual pode servir de guia para obtenção de requisitos funcionais para uma grande combinação de potenciais AVA. No entanto, este buscou dar uma visão em amplitude dos conceitos de um AVA, pois a verticalização, ou

seja, os detalhamento em subconceitos podem não ser suficientes para descrever cada conceito. Além disso, não se espera que este trabalho consiga abranger todas as possíveis características de um AVA, uma vez que o universo destes ambientes é muito extenso, sendo este um dos motivos para que o mapa conceitual fosse elaborado com a composição de conceitos. Com isso, caso surja alguma nova característica ou deseje-se considerar outros AVA, o modelo conceitual pode ser evoluído, assim como a solução arquitetural proposta. Vale ressaltar que o impacto arquitetural após uma adição, remoção ou alteração de um conceito é considerado baixo, uma vez que só afetaria a fase de detalhamento dos componentes.

Visando aproximar a visão do educador com a visão do desenvolvedor, foi desenvolvido um sistema de recomendação que mapeia as escolhas do educador, em relação às funcionalidades do AVA, em o que o desenvolvedor precisa para desenvolvê-lo. Quanto à arquitetura de referência proposta, ela foi desenvolvida de uma forma que suporta qualquer combinação de requisitos não-funcionais. Por exemplo, caso o requisito não-funcional escalabilidade seja de baixa prioridade ou não seja necessário, pode-se remover da arquitetura a distribuição de carga. Bastando para uso, substituir um elemento arquitetural de forma localizada.

Um trabalho futuro para a solução proposta nesta dissertação, que já está sendo desenvolvido, é a utilização de lógica Fuzzy nas inferências do sistema de recomendação, para que com isso fosse possível inferir um ordem de prioridade nos requisitos não- funcionais.

Referências

Bass, Len; Clements, Paul; and Kazman, Rick**. Software Architecture in Practice (SEI Series in Software Engineering),** 2nd Edition. Addison-Wesley, 2003.

Buschmann, Frank; Meunier, Regine; Rohnert, Hans; Sommerlad, Peter; and Stal, Michael**. Pattern-Oriented Software Architecture**, Volume 1: A System of Patterns. Wiley, 1996.

Cheesman, John and Daniels, John. **UML Components: A Simple Process for Specifying Component-Based Software**. Addison-Wesley, 2000.

Crespo, S.; Fontoura, M.F. M. C. e Lucena, C. J. P. de. **Um Modelo Conceitual Compatível com a Plataforma EDUCOM/IMS para Comparação de Ambientes de Educação na WEB**. 1998. Disponível em: < http://fontoura.org/papers/sbie98.pdf >. Acesso em: 24 abril 2011.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa**. São Paulo. Atlas. 1991

Shaw, Mary and Garlan, David. **Software architecture: perspectives on an emerging discipline**. Prentice Hall, 1996.

Sweeney, Alan**. Agile e Open E-Learning Systems Architecture**. Athabasca. 2008. Dissertação (Master of Science in Information Systems). Athabasca University.