

# Framework Baseado em Padrões Abertos para Construção de Ambientes CSCW/CSCL na Web

José Marques Pessoa  
ICLMA/UFMT, PPGEE/UFES  
Av. Fernando Ferrari, s/n, Vitória-Es, Brasil  
jmpessoa@npd.ufes.br

Crediné Silva de Menezes  
PPGEE/UFES  
Av. Fernando Ferrari, s/n, Vitória-Es, Brasil  
credine@inf.ufes.br

## Resumo

*Este trabalho apresenta uma proposta de framework, chamado FAmCorA, para o desenvolvimento de aplicações do tipo CSCW/CSCL na Web. O framework está baseado em padrões abertos, com ênfase no reuso de aplicações executáveis na Web, e não de código fonte, serviços (APIs) ou componentes dependentes de um middleware. O FAmCorA faz a construção de groupwares na Web possível com o simples reuso do modelo de hipertexto. A composição e o reuso de aplicações no FAmCorA utiliza apenas conhecimento declarativo, escrito em HTML ou XML, gerado automaticamente por um ambiente visual de desenvolvimento, tornando, assim, a construção de portais CSCW/CSCL na Web, uma simples atividade de autoria.*

## 1. Introdução

As redes de computadores, em particular a Internet, ampliaram as possibilidades de interações, visando a troca e a aquisição de conhecimento, entre pesquisadores, professores e estudantes. Segundo Bonk & King citados em [1] a rede pode: i) mudar a maneira com que aprendizes e professores interagem; ii) aumentar as oportunidades para a aprendizagem colaborativa; iii) facilitar as discussões; iv) deslocar o estudo de um processo individual em direção à aprendizagem social e participativa.

A utilização da informática na educação, no Brasil, é tida como uma experiência promissora. Ao contrário de outros países, onde a introdução dos computadores no contexto da aprendizagem se deu a partir de pressupostos puramente tecnológicos, a comunidade científica nacional tem reafirmado a primazia dos aspectos sócio-culturais-cognitivos sobre estes.

A comunidade brasileira de pesquisadores da área de informática na educação participa desse esforço e tem feito importantes contribuições para o desenvolvimento de ambientes telemáticos de apoio ao ensino e aprendizagem, tanto em aspectos pedagógicos quanto no desenvolvimento de ferramentas computacionais. [2], [3].

Entretanto, ainda que o número de ambientes propostos seja significativo, a realidade qualitativa aponta que típicos ambientes telemáticos de apoio à aprendizagem na Web, relatados, dão suporte a um conjunto padrão de

funcionalidades: documentos em hipertexto, áreas específicas para compartilhamento de arquivos (upload/download), distribuição de mensagens (e-mail), conversação síncrona (chat), fóruns de discussão, murais (quadro de avisos), serviço de notificação de novidades, notificação de presença para troca de mensagem instantâneas (IM) e serviço de esclarecimento de dúvidas (FAQ) [4].

Apesar da nítida intersecção do conjunto básico de aplicações/ferramentas utilizadas nesses sistemas, devido a questões tecnológicas e do escopo do domínio de problema, a construção de um ambiente novo quase sempre requer a re-implementação de cada uma dessas ferramentas. Paralelamente, existem muitas outras funcionalidades que são originais a cada sistema e que não são passíveis de reaproveitamento. Esse cenário apresenta algumas sinalizações importantes: primeiro indica que muito do esforço da comunidade é despendido "reinventando a roda"; segundo indica que a satisfação de uma determinada abordagem pedagógica pode não ser alcançada por nenhum sistema isoladamente, acarretando a invenção de um novo sistema, alimentando o círculo vicioso.

Dado este cenário, este trabalho propõe uma arquitetura de desenvolvimento voltada para a interoperabilidade entre aplicações, na Web, com ênfase no reuso de aplicações. O reuso aqui proposto é do tipo ready to run, diferentemente das abordagens clássicas tais como: reuso de código fonte (módulos, classes, etc.), reuso de componentes pré-compilados e reuso de middlewares (EJB, COM, CORBA, etc.).

O objetivo é fazer com que a geração de portais do tipo CSCW/CSCL (Computer Supported Cooperative Work /Computer Supported Cooperative Learning), em particular os de apoio ao ensino-aprendizagem deixe de ser uma tarefa exclusiva de experts em programação para se tornar uma atividade de autoria, ao alcance de educadores, pedagogos e professores das mais diversas áreas.

O texto está organizado da seguinte maneira: a seção 2 discute componentes de software com ênfase nos modelos de composição; a seção 3 apresenta os requisitos dos ambientes CSCL/CSCW na Web e aspectos considerados relevantes para a partição do domínio visando a construção de um framework; a seção 4 apresenta uma proposta de framework para construção de ambientes CSCW e CSCL na web (FAmCorA); a seção 5 exemplifica com uma

aplicação do framework proposto: o próprio site do FAmCorA; a seção 6 discute trabalhos correlatos e a seção 7 apresenta as considerações finais.

## 2. Modelo de Composição na Web

Considerando o paradigma de hipertexto da WEB e tomando App1, App2 e App3 como sendo típicas aplicações utilizadas em ambientes virtuais de aprendizagem, a figura 1 apresenta uma possível composição destas aplicações, dando origem a um portal de apoio a aprendizagem.

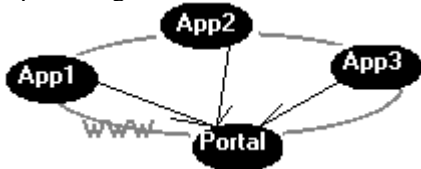


Figura 1. Portal CSCL na Web.

No mais genuíno estilo hyperlink da Web, a aplicação "Portal CSCL" é um ponto de unificação do acesso aos serviços das aplicações App1, App2 e App3.

A Web faz composição de aplicações e não de serviços. Esse paradigma sinaliza uma orientação inovadora em que a composição utiliza apenas o conhecimento declarativo (HTML) e não requer qualquer tipo de glue-code.

## 3. Ambientes CSCW/CSCL

A inspiração para desenvolver os ambientes CSCL (Computer Supported Collaborative Learning) originou das pesquisas em CSCW (Computer Supported Cooperative Work). O Trabalho Cooperativo é uma atividade em que cada pessoa é responsável pelo desenvolvimento/resolução de uma parte do problema/tarefa. A colaboração envolve o compromisso mútuo dos participantes num esforço coordenado, visando à conclusão/resolução da tarefa/problema.

Um ambiente CSCL visa dar suportes para as principais atividades da aprendizagem colaborativa: diálogos; compartilhamento de idéias e documentos; observações e sugestões; coordenação e controle; planejamento e execução de tarefas; iniciativa e supervisão. Os recursos (ferramentas) disponíveis em típicos ambientes CSCL para dar suporte a essas atividades são: canais de diálogos; espaços e objetos compartilhado e espaço e objetos pessoais. A finalidade última de um ambiente CSCL é facilitar a construção coletiva/cooperativa do conhecimento.

Segundo [6], os ambiente CSCL devem dar suporte a: Comunicação/interação entre os participantes; Compartilhamento da informação; Análise das atividades/interações; Coordenação e Percepção (awareness): o entendimento/percepção das atividades do

outro, que proporciona um contexto para a realização da própria atividade [7].

### 3.1 Aspectos em CSCW/CSCL

Visando a construção de um framework orientado a aplicações na Web, os aspectos de CSCW/CSCL observados, foram agrupados em duas categorias: aspectos referentes à organização de comunidades virtuais e aspectos referentes à de interoperabilidade entre aplicações.

### 3.2 Aspectos Referentes à Organização de Comunidades Virtuais

Portal: Segundo [5], "a premissa básica do ensino online deve ser a construção de comunidades de aprendizes para facilitar a troca de idéias, informações e sentimentos". O termo "Portal" designa um ponto compreensível de acesso à informação (categorização e busca), aplicações (desktop integrado) e pessoas (colaboração) em uma comunidade virtual na Web.

Grupo: o componente básico da abordagem baseada na aprendizagem colaborativa é a aprendizagem em grupo. Os seguintes termos também são entendidos com o mesmo significado: projeto, turma, disciplina, evento, etc.

Papel: O conjunto de permissões de um indivíduo, em relação ao sistema e que corresponde a responsabilidade do indivíduo na organização. Como nas sociedades reais, os indivíduos das comunidades virtuais têm o seu comportamento (preferências, atribuições, responsabilidades, direitos, etc.) segundo a função que desempenham na organização (ex: professor, estudante, coordenador, tutor, monitor, membro, convidado, colaborador, aprendiz, etc), nesse sentido as aplicações de um ambiente CSCL/CSCW precisam ser adaptativas a cada um desses papéis pedagógicas.

Autenticação: A maioria dos ambientes CSCL/CSCW requer que o usuário se identifique.

### 3.3 Aspectos de Interoperabilidade entre Aplicações

Os seguintes aspectos em relação a interoperabilidade entre aplicações observados:

Notificação: Visando manter as aplicações e seus usuários ciente de fatos, preferências, acontecimentos e eventos que transcorrem no ambiente, os sistemas CSCL/CSCW implementam algum mecanismo de notificação.

Log: [6] argumentam que é fundamental que os ambientes virtuais de aprendizagem guardem o histórico das atividades de cada indivíduo para monitoramento das interações, coordenação e análises.

Composição: Um dos objetivos da proposta aqui relatada é que uma aplicação desenvolvida para um ambiente/portal possa ser utilizada (plug-and-play) em um outro, de modo transparente, apenas apontando para um hyperlink. A composição diz respeito ao acoplamento entre uma aplicação e outra aplicação, entre aplicação e o framework.

#### 4. A proposta FAMCorA - Um Framework para Construção de Ambientes CSCW/CSCL na web

A proposta FamCorA, um acrônimo para Framework para Construção de Ambiente Cooperativos de Apoio à Aprendizagem, aborda a questão dos aspectos do ponto de vista da arquitetura. A proposta consiste de: um modelo de componentes/aplicações; os padrões que guiam a composição e interação entre componentes (protocolos) e as restrições nesses padrões (estilo arquitetônico).

##### 4.1 Estendendo o Paradigma Cliente/Servidor da Web

Considerando o modelo clássico cliente/servidor apresentado na figura 2, observa-se que o script CGI pode oferecer uma cópia personalizada/customizada do "doc2.htm" ao Ambiente B, através da leitura dos parâmetros (e seus valores apropriados) passados via hyperlink. Resta, entretanto a questão do feedback do Ambiente A para o Ambiente B. Isto é, como adequar os valores dos parâmetros ou até mesmo o próprio "doc1.htm" do Ambiente B de acordo com, digamos, a performance do usuário na interação com o documento "doc2.htm" do ambiente A? Esta é uma questão crucial para os Ambientes CSCL que precisam dessa informação de feedback visando o controle e o monitoramento das interações.

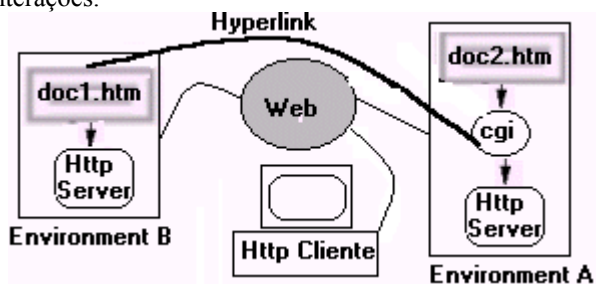


Figura 2. O "doc1.htm" do Ambiente B contém um hyperlink para uma aplicação script CGI do Ambiente A.

A figura 3 apresenta um esquema que possibilita o feedback entre os dois ambientes.

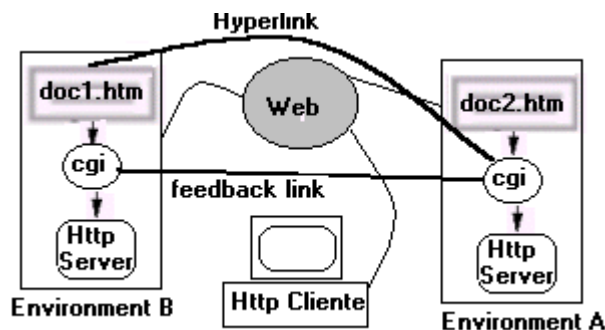


Figura 3. O Ambiente A oferece um feedback ao Ambiente B através de um feedbacklink.

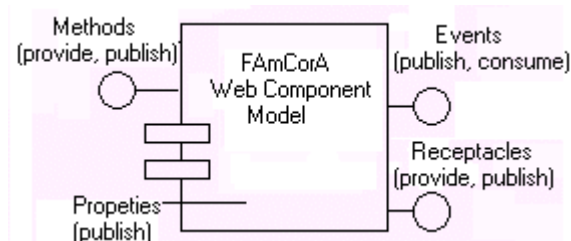
Agora, o ambiente B também está dotado de uma espécie de "inteligência" oferecida por um script CGI. Nesse cenário o feedback entre os ambiente (feedback link) é viabilizado pela comunicação direta entre as partes "inteligentes" dos sistemas, os scripts CGIs.

Uma generalização imediata do esquema proposto na figura 3 é a noção de que o feedback link pode ser usado nos dois sentidos da comunicação, abrindo assim amplas possibilidades para: 1) a troca de informações entre ambientes 2) Suporte a notificação e configuração dinâmica ("on the fly") entre ambientes.

Nessa proposta está implícita uma modificação importante no papel desempenhado pelo programa CGI. Em geral, na arquitetura cliente/servidor clássica do HTTP, o CGI tem sido, quase sempre, apenas um valioso auxiliar do lado servidor e até se confundido com este. O papel de cliente HTTP tem sido unicamente dado ao programa chamado browser. Esta restrição não existe no HTTP. A implementação do script CGI também como um cliente do protocolo HTTP abre novas e excelentes possibilidades nessa arquitetura já prodigiosa chamada Web e está na base do modelo de componentes da proposta apresentada neste trabalho.

##### 4.2 Modelo de Componentes do FAMCorA

Os componentes, aqui chamados de *famcoralets*, são os módulos básicos desenvolvidos por colaboradores especialistas em programação e são registrados em um repositório (catálogo) do framework. Um componente publica métodos, propriedades, eventos e receptáculos, conforme figura 4. Uma publicação é um compromisso de configuração. O termo receptáculo está relacionado com a composição da interface gráfica (GUI). Isto é, um componente (*famcoralet*) que publica um receptáculo passa a ser também um contêiner GUI.



#### Figura 4. Modelo de Componente.

Uma FAMCorAlet é um componente/aplicação, conforme o modelo de componentes do FamCorA, comprometido com as seguintes responsabilidades: 1) participar do processo de registro em um diretório, 2) estabelecer uma comunicação com outras FAMCorAlet conforme os protocolos estabelecidos pela proposta FamCorA e 3) implementar as interfaces de um cliente HTTP.

O diagrama de classes da figura 5 mostra a hierarquia geral das classes de aplicação/componente (FamCorAlet) compatíveis com a proposta.

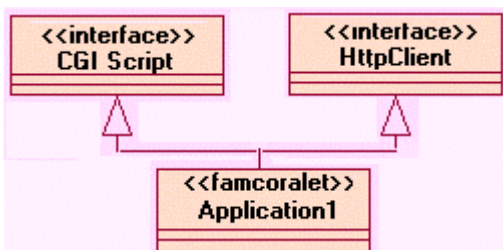


Figura 5. Hierarquia de classes de aplicação/componente do FAMCorA.

Na proposta aqui relatada um componente é qualquer aplicação cujo front-end é um cliente HTTP (o browser ou outro componente), podendo ser implementado em qualquer tecnologia de geração dinâmica de páginas (scripts) em conformidade com o protocolo CGI (exemplos: cgi-bin, ASP, ASP.NET, Java servlet, PHP, ISAPI DLL, Apache DSO, etc.) com as seguintes responsabilidades: participar do processo de registro ao framework e estabelecer uma comunicação com outras aplicações conforme os protocolos estabelecidos pelo framework. O diagrama de classe da figura 5 mostra a hierarquia geral das classes de aplicação/componente. A figura 5 mostra que um componente/aplicação (famcoralet) compatível com o framework é um tipo de aplicação *script* CGI que implementa as interfaces de um cliente HTTP.

#### 4.3 Modelo de Composição

A composição de componentes (seja de serviços ou GUI), segue o paradigma da autoria. O framework define um protocolo baseado em XML, chamado Passport, como mediador básico dos aspectos não funcionais, tais como: organização, coordenação, autenticação (portal, grupo, usuário), configuração e composição e propõe uma arquitetura no estilo *blackboard* para mediar a comunicação entre componentes. Esta decisão não restringe a comunicação direta entre as aplicações (como acontece no estilo *Broker*) e evita que cada desenvolvedor tenha que implementar um parser/Interpretador para o protocolo, mediar a interação e composição de componentes e ainda certificação, evitando chamadas não autorizadas entre aplicações e o chamado “uso malicioso”.

A figura 6 ilustra a arquitetura de composição do framework centrada no *Passport Provider* que é o guardião do *passport*.

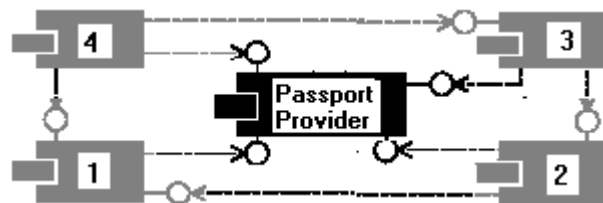


Figura 6. Arquitetura de composição.

No FAMCorA, componente e aplicação são conceitos intercambiáveis.. Um componente/aplicação pode ser implementada em qualquer tecnologia de geração dinâmica de páginas (script) em conformidade com o protocolo CGI (exemplos: cgi-bin, asp, servlet, PHP, isapi/nsapi, asp.net, etc.).

## 4.4 Protocolo Passport

O protocolo Passport é o mediador básico da arquitetura do FamCorA, principalmente para a modelagem não-funcional: organização, coordenação, autenticação (portal, grupo, usuário) e configuração e composição, podendo servir também para algumas funcionalidades gerais, como a manutenção de uma sessão no http. Um Passport é codificado em XML e seus elementos básicos são <authentication>, <groups> e <applications>.

Na prática o passport funciona também como uma espécie de linguagem de configuração da arquitetura. O passport do portal é a sua configuração básica. As configurações particulares para cada um dos seus usuários são mantidas pelo passport do indivíduo, isto é, cada usuário recebe o seu próprio passport ao logar no portal.

O trecho do passport mostrado na figura 7 configura o comportamento de uma aplicação/componente, possivelmente um gerenciador de arquivos, especificamente para uma chamada. O passport indica que nessa configuração somente os métodos Upload, Zip, Delete e Help (entre todos aqueles que foram registrados/publicados) devem estar disponíveis ao usuário. Das propriedades publicadas apenas TitleFont, TitleFontSize e Background precisam ser personalizadas. Por exemplo, o método Help, possivelmente abstrato, deve ser chamado no link `http://myip/scripts/help` (composição por extensão). A propriedade TitleFont deve receber o valor "Verdana" e TitleFontSize o valor 3. Background deve apontar para a URL `http://myip/scripts/myhelp`.

```
.....
<Component2>
  <behavior>
    <methods attr="valuearray">
      Upload|Zip|Delete|Help
    </methods>
    <properties attr="valuearray">
      TitleFont|TitleFontSize|Background
    </properties>
    <receptacles attr="valuearray">
      TOPROW|BOTTOMROW
    </receptacles>
    <events>onDelete</events>
    <handlers>
      <Help attr="url">http://myip/scripts/help</Help>
      <TitleFont attr="value">Verdana</TitleFont>
      <TitleFontSize attr="value">3</TitleFontSize>
      <Background attr="url">
        http://myip/scripts/myhelp
      </Background>
    </handlers>
  </behavior>
</Component2>
```

Figura 7. FamCorA Passport: Composição baseada em conhecimento declarativo.

Tendo uma referência (um GUID) para um passport, o conteúdo (ou atributo) de qualquer tag XML pode ser obtido em tempo de execução diretamente de um serviço do Passport Provider, via método GET do HTTP. Por exemplo, o pseudocódigo da figura 8 recupera o valor da propriedade TitleFont do passport mostrado na figura 7, isto é "Verdana".

```
String tf = Component2.httpGet(
  "http://passporthost/passportprovider/getInfo?node=
  portal.GUID.group.component2.behavior.handlers.TitleFont");
```

Figura 8. Acesso a um serviço do FamCorA Passport Provider.

Um passport descreve ainda muito outros aspectos não-funcionais oferecidos pelo framework, tais como o portal e o grupo que originou a chamada, a autenticação do usuário naquela chamada, etc. Na prática, algumas dessas informações, por exemplo, o host provedor do passport e a referência GUID do passport, precisam ser antecipadas, cabendo então ao componente que solicita um serviço de um outro passar esses parâmetros na mensagem de chamada GET do http, via uma querystring. De fato existe uma ontologia para essa requisição. O pseudocódigo da figura 9 exemplifica essa situação para os hipotéticos Componente1 e Componente2.

```
Component1.httpRedirect(
  "http://component2host/component2/init?portal=LaWEB
  &group=Authors
  &PassportProvider=http://200.137.66.74/passportservice
  &GUID=P17032003171414" );
```

Figura 9. Ontologia de uma requisição.

## 4.5 Protocolo xLidex

O Extensible Learning Interactions Data Exchange (xLidex) é um protocolo baseado em XML para a troca de dados relativa às atividades do usuário (log) entre aplicações, oferecendo mecanismos (estatísticas) para a coordenação e o monitoramento do trabalho individual e do grupo. O xLidex é extensível e aberto, permitindo retratar qualquer tipo de atividade proposta em uma ferramenta.

## 4.6 Protocolo Agent-handler

O Protocolo Agent-handler governa a comunicação/interação entre uma aplicação e um agente em tempo de execução. Ao se acoplar a uma aplicação um agente se anuncia como sendo do tipo *redirect* ou *interactive*. Um agente do tipo *redirect* desvia a execução de uma funcionalidade da aplicação para outra localidade, sem nada retornar. Por outro lado, um agente do tipo

*interactive* retorna uma das seguintes diretivas: *redirect* (em que o agente pede que a aplicação seja redirecionada para outra localidade), *response* (em que o agente formata a informação que deve ser retornada ao usuário da aplicação) ou *continue* (em que o agente solicita que a aplicação siga o curso normal do processamento).

## 5. Estudo de caso: o portal do FAmCorA

Como estudo de caso, para exemplificar a aplicação framework aqui relatado, apresentamos o site do FAmCorA. O Portal FAmCorA é o site construído para abrigar a comunidade de desenvolvedores (implementadores e registradores de componentes) e autores (criadores e administradores de portais). Isto é, a fábrica de CSCW/CSCL.

A figura 10 apresenta sua estrutura geral. O wizard Portal Builder é o módulo responsável pela geração e catálogo de portais. O módulo Portal Application é um portal gerado pelo wizard. O módulo Portal Builder Wizard é o responsável pelo catálogo de portais (portal aspecto). O módulo Portal Application é o responsável pelo catálogo de grupos (grupo aspecto), usuários (aspecto autenticação) e papéis (aspecto papel). Todos esses aspectos, somados a aqueles relativos à interoperabilidade (composição) entre aplicações são resolvidos no nível da arquitetura de componente e do protocolo Passport. O Log das atividades (aspecto log) é mediado pelo protocolo xLidex (Extensible Learning Interactions Data Exchange). A interação entre uma aplicação e um agent-handler (aspecto notificação) é mediada por um protocolo específico (protocolo Agent-handler).

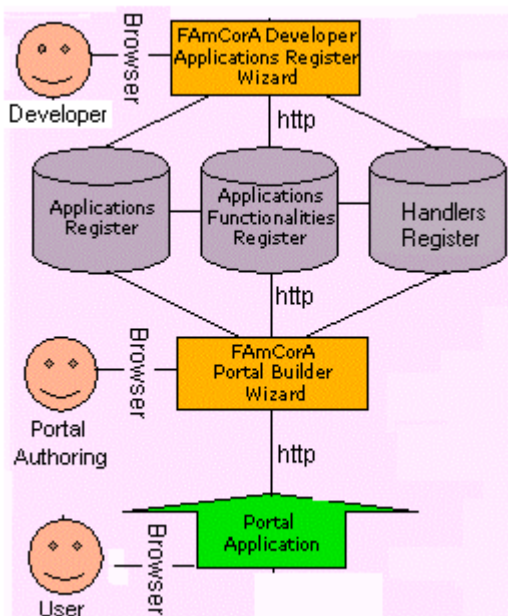


Figura 10. Visão da Geral das funcionalidades do Portal do FAmCorA.

No framework FAmCorA, um Portal pode conter qualquer quantidade de grupos (que podem conter subgrupos). Um grupo/subgrupo pode conter qualquer quantidade de papéis. Todo usuário inscrito em um grupo tem um papel. Para cada papel pode ser disponibilizado um conjunto de ferramentas (FAmCorAlets), e cada ferramenta pode ser configurada, de modo transparente (click and play!), para operar segundo um papel.

### 5.1 Ambiente do desenvolvedor

Um desenvolvedor é um colaborador do FAmCorA que desenvolve aplicações (FAmCorAlets) e está inscrito no grupo “Developers” no papel específico de “developer”. A figura 11 mostra que o desenvolvedor utiliza uma ferramenta de nome Register para: 1) Registrar Aplicação; 2) Registrar Funcionalidade; 3) Registrar Handler; 4) Registrar Temas (o tema padrão é “summer”).

A figura 15 apresenta ainda a atividade de registrar um handler (um handler é um termo genérico no FAmCorA, podendo significar deste um tratador de eventos até mesmo um valor atribuído a uma propriedade). O desenvolvedor escolhe a aplicação “NewsBoard” (lista de seleção) e pode assim, para cada funcionalidade da aplicação, ou seja, “New” (method), “OnNew” (event), SPONSOR\_1 (receptacle) e “BackGround” (property) propor um handler. No exemplo, o desenvolvedor registra um agent-handler, do tipo “interactive”, nomeado “NotifyUser”, que pode ser utilizado para interceptar o evento “OnNew” da aplicação. O handler nesse caso é um serviço (script CGI) cuja localização (URL) precisa ser informada (Handler:). Além disso, o desenvolvedor precisa entrar com a documentação (Help Url:) para orientar futuras utilizações. A seção seguinte exemplifica uma utilização desse handler que acabou de ser registrado.

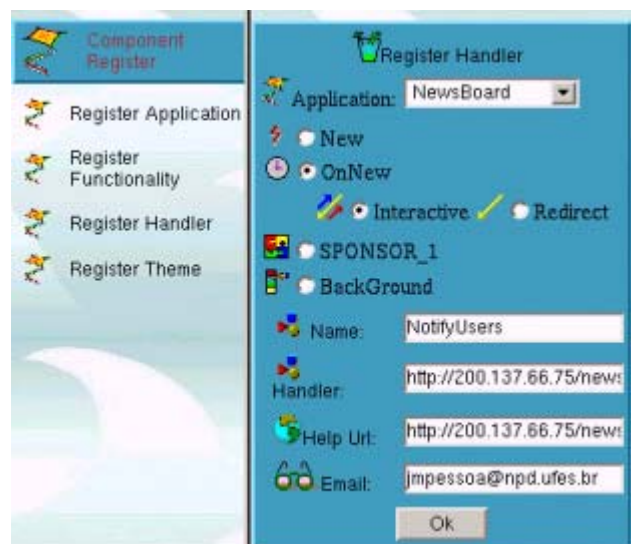


Figura 11. FAmCorA Component Register.

## 5.2 Ambiente de Autoria de Portais

Segundo [8], com o rápido crescimento do Internet, muitos usuários estão em condições de “recortar” e “colar” serviços. Tais usuários mesmo que possuindo uma perícia técnica limitada, ainda assim exigirão garantias de que as partes coladas irão funcionar corretamente, segundo as suas expectativas.

Uma das premissas básicas do projeto FAmCorA é fazer com que o desenvolvimento de um ambiente CSCL/CSCW deixe de ser uma tarefa de experts em programação e passe a ser uma tarefa de autoria, dando às várias categorias de profissionais da educação a oportunidade de criarem ambientes virtuais de suporte ao ensino-aprendizagem. A autoria de um portal educacional com o framework FAmCorA e assistida por um *wizard*, seguindo o paradigma *RAD*. Um click de mouse gera o núcleo básico do sistema, prontamente instalado e funcionando. Isto é, um Web portal.

Ao logar no ambiente, figura 12, o autor do portal já tem à sua disposição três grupos para ser administrado: "Community" é o grupo onde estão reunidos todos os usuários do portal. O "espaço\_individual" quando ativado possui um nome igual ao do email do usuário. Em se tratando do autor do portal, esse espaço já vem equipado com uma ferramenta básica de administração e configuração do portal: o GroupManager. Com ela o autor pode: 1) criar grupos e os seus papéis, 2) Incluir novos usuários em um grupo, 3) escolher quais ferramentas estarão a disposição dos usuários do grupo, em função do papel assumido por cada um deles.

Quando o autor escolhe uma ferramenta, por exemplo, "NewsBoard", é oferecido a ele a oportunidade para "personalizar" ou adaptar o seu uso, através de um menu que apresenta as várias funcionalidades que estão disponíveis. Para cada funcionalidade (eventos, métodos, propriedades e receptáculos), opcionalmente, pode ser selecionado um handler. Isto é, um valor de configuração.



**Figura 12. GroupManager: Configurando um agent-handler.**

Ainda na figura 12, o menu está aberto na paleta de eventos e o autor pode, se desejar, adicionar o agent-handler "NotifyUser", como um ouvinte (listener) do evento "OnNew" para todo usuário inscrito com o papel "Participant".

Um portal criado com o framework do FAmCorA é sempre uma obra "em construção". A qualquer momento o autor pode criar novos grupos e papéis, configurar e reconfigurar as ferramentas, sem nenhum impacto sobre o ambiente, porque o trabalho de autoria é uma "simples" geração de conhecimento declarativo (DK), isto é, uma configuração de (re)uso.

## 6. Trabalhos relacionados

De fato, as tecnologias para o desenvolvimento de aplicações groupware com ênfase no reuso têm apontado basicamente em três direções: componentes (dependentes de um middleware), bibliotecas de classes, e composição de serviços (Web services).

Alguns frameworks são específicos para um dado ambiente (ou linguagem) de programação, por exemplo, GroupKit [9] oferece um framework de desenvolvimento para ambientes groupware baseado na linguagem de programação Tcl. [10] apresenta a arquitetura COGAM (Component-based Groupware Architectural Model) voltada especificamente para a plataforma Windows e seus objetos COM (Common Object Model), ZOPE [11] oferece um ambiente extensível baseado em Python. Habanero [12] é um framework baseado em Java. No framework Habanero, um wizard auxilia a reescrita do código incluindo as APIs em Java do Habanero. O Java

oferece portabilidade para diversas plataformas de sistemas operacionais, entretanto o Habanero possui cerca de 2 Megabytes de código de classes que devem ser carregados para o browser do cliente para que o sistema seja executado. Além disso, o suporte assíncrono completo requer a instalação de versões específicas de múltiplos pacotes de software associados (Jigsaw, Jini, object databases, etc.), tornando uma instalação do framework um processo complexo e a utilização bastante comprometida em locais onde o acesso/conexão a Internet não possui uma performance elevada.

[13] descreve um WebFramework que permite a colaboração entre serviços e a composição de serviços por meio de roles para a geração de aplicações na Web. Uma composição de serviços define o perfil de uma aplicação a ser gerada. Um serviço poderá ter vários roles permitindo assim visões diferenciadas a partir de diversas aplicações. Uma linguagem de script (WebCompose), proporciona a colaboração e o reuso de serviços já existentes na plataforma. WebCompose também oferece mecanismos para composição de serviços de forma a gerar outros com mais funcionalidade.

Outros exemplos de arquiteturas orientadas a serviços incluem: E-Speak (HP), Jini (Sun) e ONE (Sun) e ainda o .NET (Microsoft). Recentemente, tais tecnologias têm convergido para o padrão W3C Web Service.

O problema das abordagens orientadas a middleware (EJB, CORBA, COM, etc) e/ou serviços é que a instanciação de um ambiente a partir desses framework requer um grande esforço técnico, pois a integração entre os componentes ocorre no nível de programação. Isto é, a aplicação a ser construída incorpora os componentes do framework durante a fase da programação.

A proposta apresentada aqui neste artigo é significativamente diferente dessas abordagens porque o seu ambiente de execução é a própria Web, ou seja, o FamCorA na requer um servidor de aplicação específico. As aplicações residem e executam em seus próprios ambientes.

## 7. Considerações finais

Para [14] uma tecnologia de reuso pode reduzir a distância cognitiva (esforço intelectual) de dois modos: (1) reduzindo a distancia, em termos de esforço intelectual, entre as de fases concepção e especificação e (2) reduzindo a distancia, em termos de esforço intelectual, entre as fases de especificação e a produção do sistema executável.

O FAMCorA é um ambiente visual de autoria tornando, assim, a construção de portais de apoio a aprendizagem colaborativa na Web, uma simples atividade do tipo click and play ao alcance da grande maioria dos usuários, não especialistas em programação de computadores.

O framework do FAMCorA está baseado em padrões da Web, com ênfase no reuso de aplicações ready to run, e não de código fonte (módulos, bibliotecas de classes, etc)

ou componentes binários, dependentes de um middleware (EJB, CORBA, .Net, etc). A arquitetura do FAMCorA está voltada para o reuso e a interoperabilidades entre aplicações Web. Isto tem um significado que julgamos importante: uma aplicação desenvolvida para um portal pode ser utilizada em outros, apenas apontando para um hyperlink. Essa estratégia é fundamental para o surgimento de novas abordagens para ambientes CSCL/CSCW, porque agora é fácil construí-los, utilizando um ambiente de composição visual.

## 8. Referências

- [1] Lehtinen, Erno; Hakkarainen, Kai; Lipponen, Lasse; Marjaana, Rahikainen; Muukkonen, Hanni. Computer Supported Collaborative Learning: A Review <http://www.kas.utu.fi/papers/clnet/clnetreport.html>
- [2] Santoro, F. N.; Borges, M. R. S.; Santos, N. Um Framework para estudos de ambientes de suporte 'a aprendizagem cooperativa. Revista Brasileira de Informática na Educação, No. 4, Abril de 1999.
- [3] Menezes, C.; Pessoa, J.M., Netto, H. V.; et all; Educação a distância no Ensino Superior - Uma proposta baseada em Comunidades de Aprendizagem usando Ambientes Telemáticos, XIII Simpósio Brasileiro de Informática na Educação São Leopoldo/RS 2002.
- [4] Rowley, Kurt. A design approach for the engineering and construction of CSCL-ITS environments. In proceedings of the AI-ED 97 Workshop on Computer-supported collaborative learning environments and AI, Kobe, Japan, August 20, 1997.
- [5] Hiltz, Starr Roxanne. Collaborative Learning in Asynchronous Learning Networks: Building Learning Communities, In "WEB98", Orlando/Florida, November, 1998.
- [6] Stahl, Gerry. Contributions to a Theoretical Framework for CSCL Proceedings of the International Conference on Computer Support for Collaborative Learning, 2002. <http://orgwis.gmd.de/~gerry/publications/conferences/2002/CSCL2002/index.html>
- [7] Lopes, P G., Skarmeta, A. F. G. ANTS Framework for Cooperative WorkEnvironments, IEEE-Computer, March/2003.
- [8] Garlan, David. Software Architecture: a Roadmap. The Future of Software Engineering, A. Finkelstein, ed., ACM Press, 2000.
- [9] Roseman, M., Greenberg, S. Building Real-Time Groupware with GroupKit, A Groupware Toolkit. ACM Transactions on Computer-Human Interaction 3, (1), 66-106, 1996.
- [10] Rees, Michael J.; Herring, Charles. A Component-Based Groupware Architecture Model (COGAM) <http://comet.it.bond.edu.au/borg>
- [11] ZOPE. [www.zope.org](http://www.zope.org)

- [12] Chabert, A., Grossman, E., Jackson, L., Pietrowicz, S., and Seguin, C., Java Object-Sharing in Habanero. Communications of the ACM, Vol. 41 # 6, June 1998.
- [13] Crespo, Sérgio C S Pinto. Tese de Doutorado. Composição em WebFrameworks, PUC-Rio, Agosto 2000.  
<http://www.inf.unisinos.br/~crespo/publicacoes.html>
- [14] Krueger, Charles W. Software Reuse. ACM Computing Surveys, Vol. 24, No. 2, June 1992.