# Uso da técnica de planejamento hierárquico no sequenciamento automático de atividades colaborativas

**Geiser Chalco Challco**[1], **Marco Aurélio Gerosa** [1], **Leliane Nunes de Barros**[1]

[1]Instituto de Matemática e Estatística – Universidade de São Paulo (USP)
Caixa Postal 66.281 – São Paulo – SP – Brasil

{geiser,gerosa,leliane}@ime.usp.br

***Abstract.*** *The automatic sequencing of activities in a collaborative learning system is a task that can be modeled as a planning problem in the field of Artificial Intelligence. In this article, we modeled the automatic sequencing of collaborative learning activities, this task involve the modeling of educational goals, the teaching domain, the student model and the general tasks of instructional design. The proposed model can be input of the JSHOP2 system, a hierarchical domain independent planner.*

***Resumo.*** *O sequenciamento automático de atividades num sistema de aprendizagem colaborativa é uma tarefa que pode ser modelada como um problema de planejamento automático da área de Inteligência Artificial. Neste artigo, é modelado o sequenciamento automático de atividades de aprendizagem colaborativa envolvendo a modelagem dos objetivos pedagógicos, do domínio que se deseja ensinar, do modelo do estudante e das tarefas genéricas de um projeto instrucional. A modelagem proposta pode ser dada como entrada para o sistema JSHOP2, um planejador hierárquico independente do domínio.*

## 1. Introdução

O sequenciamento automático de atividades de aprendizagem, conhecido também como sequenciamento de curso [Brusilovsky and Vassileva 2003] tem uma longa história em informática na educação no que diz respeito à adaptação de cursos, lições, módulos e exercícios às características e necessidades dos estudantes. Com os trabalhos de [Peachey and McCalla 1986] e [Marcke 1992] surgiram propostas de sequenciamento automático de atividades de aprendizagem que utilizam diversas técnicas de planejamento em inteligência artificial [Santos and Boticario 2007], [Vrakas et al. 2007]. Contudo, no sequenciamento de atividades colaborativas as propostas atuais encontradas são baseadas em workflows e padrões de colaboração [Hernández-Leo et al. 2006] [Pinkwart 2003], que oferecem pouco suporte à adaptação de atividades a cada participante, tornando este processo uma tarefa complexa.

Neste artigo apresentamos uma proposta de sequenciamento automático de atividades colaborativas, adaptadas a cada participante, que utiliza a técnica de planejamento hierárquico. Na Seção 2, os conceitos fundamentais de planejamento hierárquico são apresentados. Na Seção 3 apresentamos a modelagem do sequenciamento automático de atividades colaborativas como problema de planejamento hierárquico e finalizamos com a Seção 4, apresentando um exemplo do processo de planejamento hierárquico de uma tarefa de projeto instrucional.

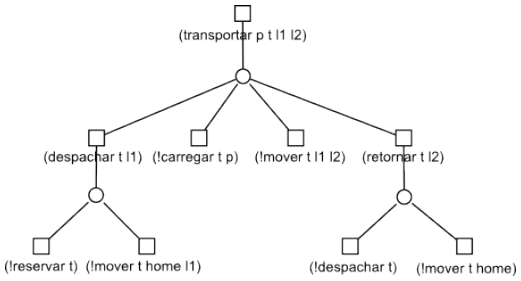
## 2. Planejamento hierárquico

Planejamento hierárquico é uma técnica de Inteligência Artificial na qual o problema de busca e sequenciamento de um conjunto de ações, chamado de plano, é gerado mediante a decomposição de *tarefas compostas* (tarefas com nível de abstração) em sub-tarefas até atingir um nível de *tarefas primitivas* (tarefas que não podem ser decompostas em sub-tarefas) [Nau et al. 2004]. Assim, o plano deve ser executado num determinado estado inicial ($S_0$) e alcançar um estado meta para atingir o objetivo que é efetuar a *tarefa inicial* ($T$) (tarefa composta, de maior nível de abstração).

Neste trabalho, utilizamos o sistema de planejamento hierárquico JSHOP2 (*Java Simple Hierarchical Ordered Planner 2*) [Ilghami 2006], planejador independente de domínio de aplicação desenvolvido pela Universidade de Maryland, no qual o conjunto de tarefas, métodos e operadores são representados com base na lógica de predicados de primeira ordem mediante variáveis anotadas com o prefixo *"?"* e átomos (expressões lógicas) representados como as n-uplas *(p* $t_1$ ... $t_n$*)*. As tarefa são representadas com a expressão *(*$p$ $t_1$ ... $t_n$*)* na qual $p$ é o nome da tarefa e os argumentos $t_1$ ... $t_n$ são termos (as tarefas primitivas são anotadas com o prefixo *"!"* no nome). Os métodos são representados utilizando a forma *(:method* $h$ $pre$ $t$*)* na qual $h$ é o nome a tarefa composta, $pre$ é a expressão lógica que define a aplicabilidade e $t$ representa a lista de sub-tarefas em que a tarefa composta $h$ pode ser decomposta. As tarefas primitivas são representadas mediante operadores utilizando a forma *(:operator* $h$ $pre$ $del$ $add$*)* na qual $h$ é o nome da tarefa primitiva, $pre$ é a pre-condição, $del$ e $add$ representam os efeitos de uma ação como a lista de átomos a serem removidos (efeitos negativos) e adicionados (efeitos positivos) na representação do estado atual.

```
1  (:method (transportar ?p ?t ?l1 ?l2)
2    ((disponivel ?t)        ;; precondições
3     (em ?p ?l1))
4    ((despachar ?t ?l1)     ;; sub-tarefas
5     (!carregar ?t ?p)
6     (!mover ?t ?l1 ?l2)
7     (retornar ?t ?l2)))
8  (:operator (!mover ?t ?l1 ?l2)
9    ((em ?t ?l1))       ;; precondições
10   ((em ?t ?l1))       ;; efeitos negativos
11   ((em ?t ?l2)))      ;; efeitos positivos
```

**(a)** Domínio de planejamento em JSHOP2




**(b)** Decomposição da tarefa transportar

**Figura 1: Exemplificação do problema de transporte de pacotes com planejamento hierárquico**

A Figura 1 exemplifica parte de um domínio de planejamento para o problema de transporte de pacotes de uma localização. Nesse exemplo, a representação do estado é feita com os átomos *(em* $t$ $l$*)*, significa que o pacote ou veículo $t$ está localizado em $l$ e *(disponivel* $t$*)*, significa que o veículo $t$ está disponível. As tarefas primitivas são: reservar um veículo *(!reservar ?t)*, mover o veículo de uma localização para outra *(!mover ?t ?l1 ?l2)*, carregar um pacote no veículo *(!carregar ?t ?p)* e descarregar um pacote do veículo *(!descarregar ?t ?p)*. As tarefas compostas são: despachar um veículo *(despachar ?t ?l1)*, que envolve as tarefa reservar e mover; retornar o veículo *(retornar* $t$ $l1$*)*, que envolve despachar e mover; e transportar um pacote *(transportar ?p ?t ?l1 ?l2)* que envolve despachar, carregar, mover e retornar. A Figura 1a apresenta o método de decomposição da tarefa transportar um pacote *(transportar ?p ?t ?l1 ?l2)* (linhas 1-7) que é aplicável se as precondições, o pacote *?p* está localizado em *?l1* e o transporte *?t* está disponível (linhas

3-4), são satisfazíveis. Esse método é decomposto nas sub-tarefas: *despachar*, *!carregar*, *!mover* e *retornar* (linhas 4-7). O operador *(!mover ?t ?l1 ?l2)* (linhas 8-11) representa a ação de mover um veículo *?t* de *?l1* para *?l2* com a remoção do átomo *(em ?t ?l1)* (linha 10) e a adição do átomo *(em ?t ?l2)* (linha 11). A Figura 1b ilustra a decomposição da tarefa transportar o pacote $p$ da localização $l1$ a $l2$. O plano solução para o problema é dado pela sequência de tarefas primitivas: *(!reservar* $t$*)*, *(!mover* $t$ $home$ $l1$*)*, *(!carregar* $t$ $p$*)*, *(!mover* $t$ $l1$ $l2$*)*, *(!despachar* $t$*)* e *(!mover* $t$ $l2$ $home$*)*.

## 3. Sequenciamento automático de atividades colaborativas como um problema de planejamento hierárquico

Um sistema de planejamento hierárquico aplicado ao domínio de aprendizagem colaborativo é inicializado pelo educador mediante a seleção dos participantes e dos objetivos pedagógicos que definem a tarefa inicial ($T$) e o estado inicial ($S_0$). Essas definições são feitas usando-se uma ferramenta de edição dos modelos de domínio e de estudante, Figura 2. A sequência de atividades colaborativas que será gerada pelo planejador hierárquico JSHOP2 é especificada no padrão IMS-LD (*IMS - Learning Design*) para ser executada num ambiente virtual de aprendizagem (LMS).

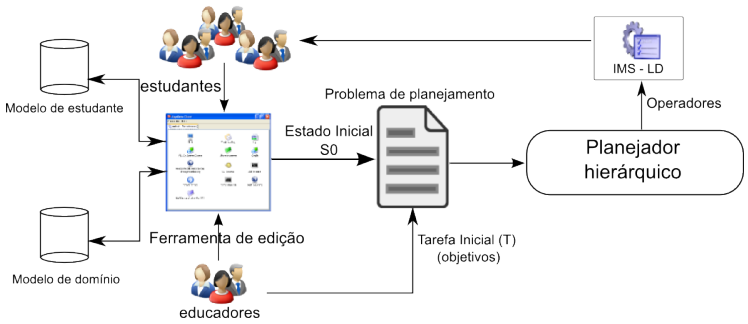



**Figura 2: Sequenciamento automático de atividades colaborativas como planejamento hierárquico**

Nas subseções a seguir apresentamos a representação da tarefa inicial (Subseção 3.1), estado inicial (Subseção 3.2), operadores (Subseção 3.3) e as tarefas e métodos hierárquicos (Subseção 3.4) utilizadas no sequenciamento das atividades de aprendizagem colaborativas, em termos da linguagem do planejador JSHOP2 [Ilghami 2006].

### 3.1. Tarefa inicial: objetivos pedagógicos e participantes

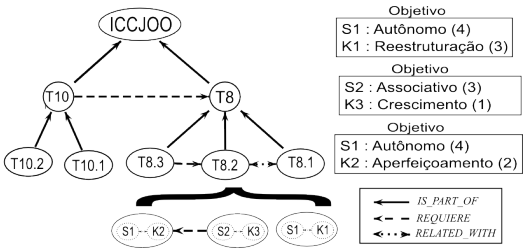
A tarefa inicial *(createCLUoL* $l$ $p$*)* é definida utilizando a formalização dos objetivos pedagógicos $l$ como 4-uplas $(s_i\ l_{s_i}\ k_i\ l_{k_j})$ que representam os níveis de competência desejados para os estudantes em termos de desenvolvimento de habilidade, $s_i$, e aquisição de um conhecimento específico, $k_j$. Os níveis de competência utilizam valores qualitativos do modelo de aprendizagem do estudante (LGM [Isotani and Mizoguchi 2006]) em que $l_{s_i}$ assume valores: i) sem habilidade (*nothing*), ii) inicial (*rough*), ii) explicativo (*explanatory*), iii) associativo (*associative*) e iv) autônomo (*autonomous*); e $l_{k_i}$ assume valores: i) sem conhecimento (*nothing*), ii) crescimento (*accretion*), iii) aperfeiçoamento (*tuning*) e iv) reestruturação (*restructuring*).

A Figura 3b ilustra a definição dos objetivos pedagógicos para o tópico 8.2 “O Laço While” (T8.2) na disciplina “Introdução à Ciência da Computação com Java e

Orientação a Objetos" (ICCJOO) [Kon and Goldman 2004] na qual o educador definiu como objetivos pedagógicos: *(s1 autônomo k1 reestruturação)*, *(s2 associativo k3 crescimento)* e *(s1 autônomo k2 aperfeiçoamento)*. A partir da definição dos objetivos pedagógicos de um determinado tópico. Por exemplo, no Tópico 8.2, a Fígura 3a) define a tarefa inicial, *createCLUoL*, que envolve a seleção dos estudantes que participam das atividades (*student01*, *student02*, ..., *student08*).

```
1 (createCLUoL ((s2 associative k3 accretion)
2               (s1 autonomous k1 restructuring)
3               (s1 autonomous k2 tuning))
4               (student01 student02 student03 student04
5                student05 student06 student07 student08))
```

**(a)** Tarefa inicial




**(b)** Ilustração da definição dos objetivos pedagógicos

**Figura 3: Definição de objetivos no tópico 8.2 do curso ICCJOO**

## 3.2. Estado inicial: domínio que se deseja ensinar e modelo do estudante

O estado inicial modela o domínio que se deseja ensinar em termos de conhecimento, conceitos e recursos instrucionais; e o modelo do estudante envolve os níveis de competência que o estudante possui.

A modelagem do domínio a ser ensinado é feita com a definição dos elementos: habilidade *(skill $s$)*, conhecimento *(knowledge $k$)*, conceito *(concept $c$)* e recurso instrucional *(resource $r$)*; tudo dado em termos do conhecimento específico do domínio. Além disso, esse conhecimento, é definido através de relações entre os elementos do tipo: composição *(isPartOf $e_1$ $e_2$)*, ordem *(requieres $e_1$ $e_2$)* e associação *(relatedWith $e_1$ $e_2$)*.

Aos elementos *skill* e *knowledge* é associada a relação denominada **definição de competências** para representa a habilidade de aplicar um conhecimento específico num determinado conceito. A Figura 4 apresenta a representação formal da definição de competências para o Tópico 8.2 da disciplina ICCJOO.

```
1 (skill s1)       (relatedWith s1 k1)      (isPartOf s1 concept8_2)
2 (knowledge k1)   (relatedWith s1 k2)      (isPartOf s1 concept8_2)
3 (knowledge k2)                            (isPartOf k1 concept8_2)
4                                           (isPartOf k2 concept8_2)
5
6 (skill s2)       (relatedWith s2 k3)      (isPartOf s2 concept8_2)
7 (knowledge k3)   (requires k3 k2)         (isPartOf k3 concept8_2)
```

**Figura 4: Representação da definição de competências**

As relação associada aos elementos *concept* e *resource* é denominada **estrutura de conceitos** e **estrutura de recursos** respectivamente, utilizados para dividir o conteúdo a ser aprendido em cursos, módulos e lições. Os recursos instrucional são associados à *estrutura de conceitos* utilizando-se relações que definem o papel instrucional de cada recurso. Por exemplo, exercício *(exercise $r$ $c$)* é o papel que o recurso instrucional $r$ desempenha para o aprendizado do conceito $c$. Se aplica o mesmo aos papéis instrucionais: questionário *(questionarie $r$ $c$)* e leitura *(lecture $r$ $c$)*. Definimos também uma medida de

benefício esperado na utilização de um recurso instrucional para o desenvolvimento de uma habilidade $d$ ou aquisição do conhecimento $d$ que é definida por *(competence* $r$ $d$ $l$*)*.

O **modelo do estudante** define os níveis de competência possuídos pelo estudante no desenvolvimento de habilidades e aquisições de conhecimento, definidos por *(skill* $s_i$ $p$ $l_{s_i}$*)* e *(knowledge* $k_i$ $p$ $l_{k_i}$*)*, que relacionam o estudante $p$ com um nível $l_{s_i}$ no desenvolvimento de uma habilidade $s_i$ e com um nível $l_{k_i}$ na adquisição de conhecimento $k_i$. A Figura 5 apresenta a formalização do modelo de estudante para as competências definidas no Tópico 8.2 da disciplina ICCJOO.

```
1 (skill s1 student01 explanatory) (knowledge k1 student01 tuning)
2 (skill s1 student04 explanatory) (knowledge k1 student04 restructuring)
3 (skill s1 student05 associative) (knowledge k1 student05 tuning)
4 (skill s1 student08 explanatory) (knowledge k1 student08 restructuring)
```

**Figura 5: Representação do modelo de estudante**

### 3.3. Operadores de planejamento

Os operadores de planejamento são utilizados para representar elementos pertencentes à especificação IMS-LD, denominados daqui em diante simplesmente como elemento(s) LD. Por exemplo: A criação de um ato *(elemento <act> em IMS-LD)*, criação de um cenário *(elemento <play> da IMS-LD)* ou a crição de uma atividade de aprendizagem (elemento <learning-activity> em IMS-LD). No sequenciamento das atividades colaborativas, os operadores de planejamento *!createLDElement* e *!closeLDElement* (Figura 6) definem a criação de elementos LD com variáveis *?type* e *?params* que representam o tipo de elemento LD e a lista de parametros do elemento LD.

```
1 (:operator (!createLDElement ?type ?params) ()()())
2 (:operator (!closeLDElement ?type) ()()())
```

**Figura 6: Operadores para adicionar elementos LD**

O operador *!insertResource* (Figura 7) é utilizado para definir a inclusão de um recurso instrucional (elemento <*resource*> em IMS-LD) com variáveis *?resource* e *?role*. Ao ser inserido o recurso instrucional, a execução do operador de planejamento adiciona os átomos *(inserted ?resource ?learner)* ao modelo do estudante que desempenhe o papel *?role*.

```
1 (:operator (!insertResource ?resource ?role)
2   ()()((forall (?learner) ((role ?learner ?role)) ((inserted ?resource ?learner)))))
```

**Figura 7: Operadores para adicionar um recurso instrucional**

Os operadores *!addUserToRole* e *!removeUserToRole* apresentados na Figura 8 são utilizados para definir a atribuição e remoção de papéis para cada estudante.

### 3.4. Tarefas é métodos de um projeto instrucional

Em aprendizagem colaborativa, **os scripts colaborativos** são receitas utilizadas pelos educadores para definir a formação de grupos e o sequenciamento de atividades colaborativas que podem ser divididas em macro e micro-scripts, representadas naturalmente

```
1 (:operator (!addUserToRole ?user ?role)
2   ()()((role ?user ?role)))
3 (:operator (!removeUserFromRole ?user ?role)
4   ()((role ?user ?role))())
```

**Figura 8: Operadores de atribuição e remoção de papéis**

por tarefas e sub-tarefas de planejamento hierárquico. A Tabela 1 resume os conceitos desenvolvidos por [Villasclaras-Fernández et al. 2009] que apresentam a integração de macro e micro-scripts através da definição de uma sessão colaborativa.

**Tabela 1: Conceitos desenvolvidos por [Villasclaras-Fernández et al. 2009]**

| Conceito | Descrição |
|---|---|
| Fases | As fases efetuam a formação de grupos e a distribuição dos grupos nas atividades colaborativas com a restrição de que os grupos somente mudam entre cada fase. |
| Atividades de grupo | As atividades de grupo apresentam um conjunto de sessões sequênciadas pelo macro-scripts que definem o trabalho em grupo de acordo com um conjunto de objetivos de grupo. Definimos o objetivo de grupo como o par de estrategias de aprendizagem que definem a interação entre participantes utilizada para atingir os objetivos individuais. |
| Sessão colaborativa | Atividade colaborativas efetuada pelos participantes do grupo a fim de atingir um objetivo específico definindo a sequência de interações através de um padrão de interação. |
| Padrão de interação | O padrão define as estratégias de aprendizagem a serem efetuadas pelos participantes na atividade, formalizando como um conjunto de interações necessárias e complementares. |
| Interação (Evento IL) | A interação é um evento instrucional e um evento de aprendizagem (Evento IL) definido por um ator, uma ação (ou conjunto de ações), benefícios esperados (para o efetuador das ações) e os recursos instrucionais necessários. |

A partir dos conceitos apresentados na Tabela 2 propomos quatro tarefas hierárquicas principais de projeto instrucional que definem o sequenciamento das atividades colaborativas detalhadas na Tabela 1 com o mapeamento dos elementos LD correspondentes. A modelagem desses métodos são apresentadas na Seção 4.

## 4. Exemplo de planejamento instrucional no sequenciamento de atividades colaborativas

Fazendo uso das modelagem da Seção 3 foi desenvolvida uma aplicação WEB que faz uso das tarefas de projeto instrucional para dar apoio ao educador no planejamento de atividades colaborativas. Neste exemplo, utilizamos como entrada: a tarefa inicial (T) da Figura 3 e o estado inicial ($S_0$) definido nas Figuras (4-6).

A Figura 9 apresenta o processo de planejamento da tarefa inicial *createCLUoL* (1) na qual a primitiva *(sortByKnowledge ?competences ?r)* (linha 2) efetua a ordenação dos objetivos pedagógicos. Na decomposição da tarefa *planningCLUoL* (1), é definida a inclusão de um elemento LD <*play*> (linhas 11,16) por cada objetivo pedagógico. Assim, no Tópico 8.2 da disciplina ICCJOO três elementos LD <*play*> são adicionados.

A tarefa *planningWithCLSteps* (2) estabelece objetivos individuais a cada participante $p_i$ adicionando as relações *(goal ?learner ?skill ?goalSkill)* e *(goal ?learner ?knowledge ?goalKnowledge)* para a definição dos objetivos individuais. A tarefa *createCLPhase* é inicializada para atingir os objetivos individuais e a tarefa *planningWithCLSteps* é executada enquanto o número de elementos LD ato (<*atos*>) gerados for menor ou igual ao máximo permitido (linhas 15, 18-19). No exemplo da Figura 9 a tarefa é efetuada

**Tabela 2: Tarefas principais de projeção instrucional**

| Tarefa Hierárquica | Descrição |
|---|---|
| (createCLUoL $l$ $p$)<br>$l$ = ($c_1$ $c_2$ ... $c_m$)<br>com $c_i$ = ($s_i$ $l_{s_i}$ $k_i$ $l_{k_i}$)<br>$p$ = ($p_1$ $p_2$ ... $p_n$) | Tarefa de projeção instrucional de mais alto nível na qual se definem os objetivos pedagógicos $l$=($c_1$ $c_2$ ... $c_n$) a serem atingidos pelos participantes $p$ = ($p_1$ $p_2$ ... $p_n$).<br>Elemento IMS-LD: <method>, <ld> |
| (createCLPhase $l$ $p$)<br>$l$ = ($s$ $l_s$ $k$ $l_k$)<br>$p$ = ($p_1$ $p_2$ ... $p_n$) | A tarefa define o agrupamento dos dos participantes e o sequênciamento das atividades de grupo entre os participantes $p$ = ($p_1$ $p_2$ ... $p_n$) efetuada para atingir o conjunto de objetivos pedagógico $l$=($s$ $l_s$ $k$ $l_k$).<br>Elemento IMS-LD: <act> |
| (createCLGroupActivity $l$ $p$)<br>$l$ = ($g_1$ $g_2$ ... $g_n$)<br>$p$ = ($p_1$ $p_2$ ... $p_n$) | A tarefa define a criação de uma atividade de grupo e o sequenciamento das sessões correspondentes a fim de que os participantes $p$ = ($p_1$ $p_2$ ... $p_n$) posam atingir o conjunto de objetivos de grupo ($g_1$ $g_2$ ... $g_n$). Desta maneira por cada objetivo $g_i$ de grupo uma sequência de sessões colaborativas é criada.<br>Elemento IMS-LD: <role-part> |
| (createCLSession $l$ $p$)<br>$l$ = ($s$ $k$ $pStrat$ $sStrat$)<br>$p$ = ($p_1$ $p_2$ ... $p_n$) | A tarefa define a criação de uma sessão colaborativa como uma atividade colaborativa a fim de atingir o objetivo de grupo ($s$ $k$ $pStrat$ $sStrat$). De forma que as estratégias $pStrat$ e $sStrat$ definem o fluxo de interações entre os participante.<br>Elemento IMS-LD: <activity-structure>, <learning-activity> |

em três situações gerando assim 3 elementos.

A tarefa *createCLPhase* (1) (Figura 10) define a criação de um elemento LD ato <*act*> (linhas 5,7) e o planejamento das atividades a serem efetuadas no ato através da tarefa *planningCLPhase* (linha 6) na qual os objetivos de grupos são identificados a partir dos objetivos individuais pela tarefa *getGroupGoals* (linha 3). A tarefa *planningCLPhase* (2) efetua a formação de grupos procurando participantes que possam trabalhar em atividades colaborativas atingindo assim os objetivos de grupo. A tarefa *getLearnersWithStrategy* (linhas 5-8) efetua a seleção dos participantes utilizando a *definição de competências* e o *modelo do estudante*. Os estudantes *student01* e *student02* foram selecionados utilizando a estratégia de aprendizagem *learning by discussion* enquanto *student05* e *student06* são selecionados utilizando as estratégias *learning by reflection* e *learning by selfexpression*. Isso mostra uma adaptação às necessidades específicas de cada estudante.

A tarefa de planejamento *planningByTheory* (2) apresenta dois métodos de decomposição que efetuam o agrupamento de estudantes efetuando a divisão do grupo em sub-grupos, tarefa *divideGroupInSubGroups* (linhas 28-29), se o número de participantes é maior do que o máximo permitido. A tarefa *createByInstruction* (3) efetua a criação de uma atividade de grupo mediante a atribuição de papéis utilizando a tarefa *addUserToRole* (linhas 7-8). Assim, como é ilustrada na Figura 10, os estudantes *student02* e *student01* são agrupados numa atividade colaborativa *la-distribute-cognition*; enquanto os estudantes *student06* e *student05* são agrupados em outra atividade *la-cognitive-flexibility*.

A tarefa *createCLGroupActivity* (1) (Figura 11) efetua a criação de um novo papel *createRoleGroup* (linha 2) utilizando os elementos LD <*role-part*> e <*role-ref*> (linhas 4-6, 8) aos quais será atribuída uma sessão colaborativa. A tarefa *createCLSession* (2) define a criação de uma atividade colaborativa de aprendizagem utilizando o elemento LD <*learning-activity*> (linhas 5-6,8).

O processo de planejamento completo da tarefa de projeção instrucional gera o

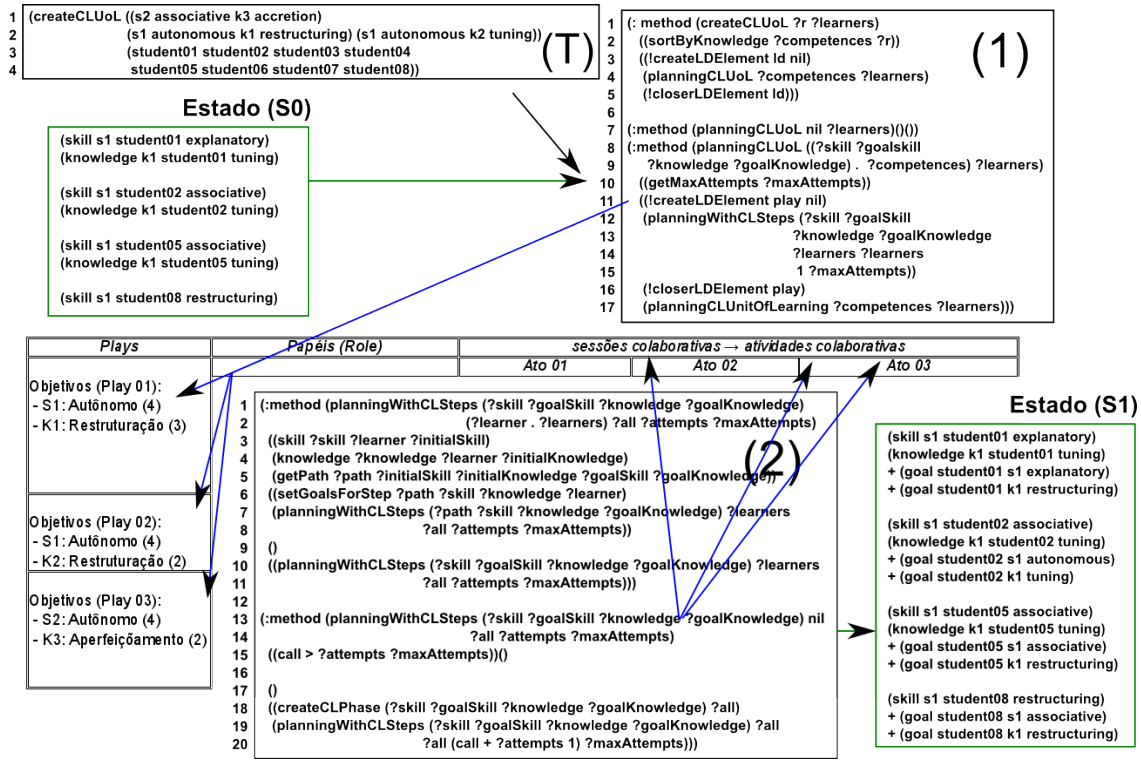



**Figura 9: Exemplificação das tarefas *createCLUoL* e *planningWithCLSteps***

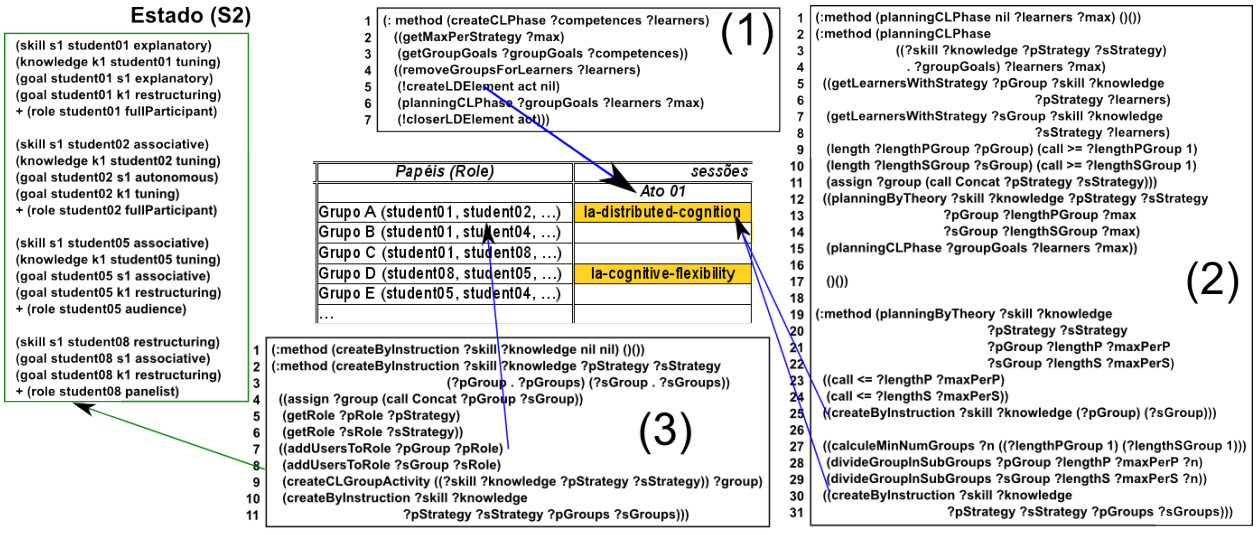



**Figura 10: Exemplificaçao das tarefas *createCLPhase*, *planningCLPhase* e *createByInstruction***

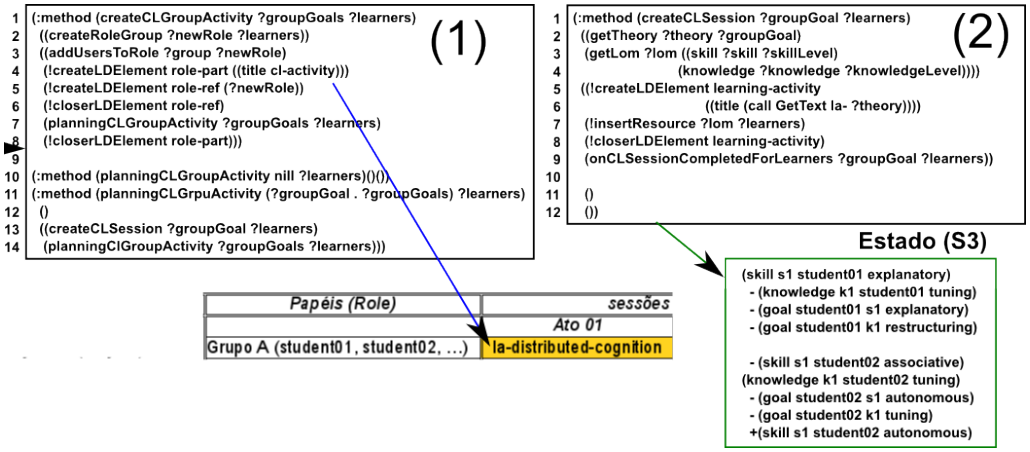



**Figura 11: Exemplificaçao das tarefas *createCLGroupActivity* e *createCLSession***

sequenciamento de atividades colaborativas apresentado na Figura 12 na qual é possível visualizar a individualização de atividades colaborativas. Por exemplo: O estudantes *student01* é agrupado com *student02* para realizar a atividade *la-distributed-cognition* e em seguida é agrupado com *student04* para realizar a atividade *la-anchored-instruction*; e finalmente com *student06* para efetuar a atividade *la-cognitive-apprenticeship*. Da mesma forma, visualizamos que o estudante *student06* realiza primeiro a atividade *la-cognitive-flexibility* com *student05* realizando em seguida a atividade *la-cognitive-apprenticeship* com *student01* atingindo assim os benefícios pedagógicos definidos na tarefa inicial.

| *Plays* | *Papéis (Role)* | *sessões colaborativas → atividades colaborativas* | | | |
|---|---|---|---|---|---|
| | | *Ato 01* | *Ato 02* | *Ato 03* | |
| Objetivos (Play 01): | Grupo A (student01, student02, ...) | la-distributed-cognition | | | |
| - S1: Autônomo (4) | Grupo B (student01, student04, ...) | | la-anchored-instruction | | |
| - K1: Restruturação (3) | Grupo C (student01, student08, ...) | | | la-cognitive-apprenticeship | |
| | Grupo D (student08, student05, ...) | la-cognitive-flexibility | | | |
| | Grupo E (student05, student04, ...) | | | la-cognitive-apprenticeship | |
| | ... | | | | |
| | | *Ato 01* | *Ato 02* | *Ato 03* | *Ato 04* |
| Objetivos (Play 02): | Grupo F (student01, student06, ...) | | | la-distributed-cognition | |
| - S1: Autônomo (4) | Grupo G (student06, student08, ...) | la-peer-tutoring | | | |
| - K2: Restruturação (2) | ... | | | | |
| | | *Ato 01* | | *Ato 02* | *Ato 03* |
| Objetivos (Play 03): | Grupo H (student01, student03, ...) | la-anchored-instruction | | | |
| - S2: Autônomo (4) | Grupo I (student04, student07, ...) | | | la-peer-tutoring | |
| - K3: Aperfeiçoamento (2) | Grupo J (student05, student04, ...) | la-cognitive-apprenticeship | | | |
| | Grupo K (student04, student06, ...) | | | | la-peer-tutoring |
| | ... | | | | |

**Figura 12: Exemplificaçao do processo de planejamento**

## 5. Conclusões

Neste artigo, a geração automática de curso colaborativos é modelada como um problema de planejamento hierárquico através da definição dos elementos do problema e domínio de planejamento como tarefas de projeto instrucional definidas em termos dos conceitos e elementos principais pertencentes aos scripts colaborativos. Dessa maneira, a principal contribuição de nossa proposta de sequenciamento automático de atividades colaborativas é o agrupamento de estudantes e a definição das atividades colaborativas com fundamentos nesses scripts.

A utilização de operadores que definem elementos da especificação IMS-LD possibilita sua execução em quaisquer ambientes virtuais de aprendizagem compatíveis. Além disso, as modelagens efetuadas no domínio que se deseja ensinar e domínio do

estudante apresenta relações e dados que podem ser representados com as especificações IMS-RDCEO, IMS-MD e IMS-LIP. Os próximos passos deste trabalho são: a integração com outras especificações IMS e a adição de mais métodos de decomposição nas tarefas de projeto instrucional efetuando a modelagem de *scripts colaborativos* como *Jigsaw* e *Pyramid* [Hernández-Leo et al. 2008].

## Referências


Brusilovsky, P. and Vassileva, J. (2003). Course sequencing techniques for large-scale webbased education. *International Journal of Continuing Engineering Education and Lifelong Learning*, 13:75–94.

Hernández-Leo, D., Villasclaras-Fernández, E., Asensio-Pérez, J., and Dimitriadis, Y. (2008). Diagrams of learning flow patterns solutions as visual representations of refinable IMS Learning Design templates. *Handbook of visual languages for instructional design: Theories and practices*.

Hernández-Leo, D., Villasclaras-Fernández, E., Asensio-Pérez, J., Dimitriadis, Y., Jorrín-Abellán, I., Ruiz-Requies, I., and Rubia-Avi, B. (2006). COLLAGE: A collaborative Learning Design editor based on patterns. *Jornal of Educational Technology and Society*, 9(1):58.

Ilghami, O. (2006). Documentation for jshop2. Technical report, Department of Computer Science, University of Maryland.

Isotani, S. and Mizoguchi, R. (2006). An integrated framework for fine-grained analysis and design of group learning activities. In *Proceeding of the 2006 conference on Learning by Effective Utilization of Technologies: Facilitating Intercultural Understanding*, pages 193–200, Amsterdam, The Netherlands, The Netherlands. IOS Press.

Kon, F. and Goldman, A. (2004). Introdução à ciência da computação com java e orientação a objetos.

Marcke, K. V. (1992). A generic task model for instruction. In Dijkstra, S., Krammer, H. P., and Merrienboer, J. G. V., editors, *Instructional Models for Computer-Based Learning Environments*, volume 104, pages 171–194. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Nau, D., Ghallab, M., and Traverso, P. (2004). *Automated Planning: Theory and Practice*. Morgan Kaufmann, San Francisco, CA.

Peachey, D. R. and McCalla, G. I. (1986). Using planning techniques in intelligent tutoring systems. *International Journal of Man-Machine Studies*, 24(1):77–98.

Pinkwart, N. (2003). A plug-in architecture for graph based collaborative modeling systems. In *Shaping the future of learning through intelligent technologies. Proceedings of the 11th Conference on Artificial Intelligence in Education*, pages 535–536. Citeseer.

Santos, O. C. and Boticario, J. G. (2007). Supporting learning design via dynamic generation of learning routes in adaptaplan. In *Proceeding of the 2007 conference on Artificial Intelligence in Education*, pages 638–640, Amsterdam, The Netherlands, The Netherlands. IOS Press.

Villasclaras-Fernández, E., Isotani, S., Hayashi, Y., and Mizoguchi, R. (2009). Looking into collaborative learning: Design from macro-and micro-script perspectives. *Frontiers in Artificial Intelligence and Applications*, 200.

Vrakas, D., Tsoumakas, G., Kokkoras, F., Bassiliades, N., Vlahavas, I., and Anagnostopoulos, D. (2007). PASER: a curricula synthesis system based on automated problem solving. *International Journal on Teaching and Case Studies, Special Issue on Information Systems: The New Research Agenda, the Emerging Curriculum and the New Teaching Paradigm*, 1:159–170.