3
O modelo PoliFacets

Este capítulo descreve e especifica o modelo conceitual de *design* para documentos ativos concebido nesta tese. Para facilitar a compreensão do leitor, apresentamos no capítulo de Introdução (p. 21-25) uma ilustração da instância mais completa deste modelo (o PoliFacets-AS) e da ferramenta de programação com a qual se ensina e aprende raciocínio computacional (o AgentSheets) no projeto SGD-Br. No capítulo 4 o leitor encontrará uma descrição detalhada da instanciação deste modelo, já discutida em função dos elementos conceituais descritos neste capítulo.

O modelo apresentado neste capítulo é resultado de dois processos principais que têm como interseção sucessivas instanciações do modelo, como provas de conceito. Um dos processos envolve vários estudos empíricos que ocorreram com as escolas parceiras no decorrer do projeto SGD-Br. O outro processo é a elaboração teórica do modelo construído, com base na Engenharia Semiótica.

Primeiramente, apresentamos os conceitos da Engenharia Semiótica que foram utilizados na elaboração do modelo e são essenciais para o seu entendimento. Em seguida, reportamos os estudos empíricos realizados para, então, relatar o processo de elaboração teórica, os componentes estruturais e uma visão completa do modelo.

3.1.
Conceitos de Engenharia Semiótica

Esta tese está fundamentada na teoria da Engenharia Semiótica. Segundo Umberto Eco (1976, 1984), a Semiótica é a disciplina que estuda a significação e a comunicação. Antes de apresentar e definir os conceitos de Engenharia Semiótica, definimos conceitos da própria Semiótica que são importantes para o que vem a seguir: signo, significação e comunicação.

Um signo é tudo aquilo que, sob certo aspecto ou em certa medida, representa algo para alguém (Peirce, 1992-1998). Os conceitos de significação e comunicação que nós adotamos foram definidos por Eco (1976). Segundo o autor, significação é o processo através do qual certos sistemas de signos são estabelecidos a partir de associações entre expressões e conteúdos resultantes de convenções sociais e culturais adotadas pelos intérpretes e produtores de signos. Comunicação é o processo através do qual, para uma variedade de propósitos, produtores de signos expressam os significados pretendidos, explorando as possibilidades de sistemas de significação existentes. Ocasionalmente, eles podem até recorrer a signos não convencionais, por exemplo, ao inventar um novo signo ou ainda usar um signo conhecido de modo imprevisível para realizar certo efeito retórico e alcançar sua intenção de comunicação de maneira mais eficaz ou eficiente. Ou seja, sempre que a cultura produz uma convenção que relaciona um conjunto expressivo de elementos a um conjunto de conteúdos, um novo sistema de significação é constituído e, um novo código torna-se disponível para comunicação (Eco, 1976).

No processo de comunicação, Eco (1976) distingue três elementos fundamentais: intenção, conteúdo e expressão. A intenção, atribuída a um comunicador tipicamente humano, se refere ao objetivo a ser alcançado com a comunicação. O conteúdo é a informação que faz parte da comunicação. Por fim, a expressão se refere a quais meios e formas foram utilizados na comunicação.

A Engenharia Semiótica (de Souza, 2005) é uma teoria de IHC em que a interação é vista como um tipo especial de comunicação mediada por computador. Essa comunicação acontece entre *designers* e usuários através sistemas em tempo de interação. O sistema tem a voz do *designer* e disponibiliza todos e somente os caminhos para conversas que foram antecipados pelo *designer* e projetados durante o desenvolvimento do sistema. Neste contexto, dizemos que o sistema é o preposto do *designer*. O preposto representa o *designer* e age em nome dele no processo de comunicação.

O principal objeto de estudo da Engenharia Semiótica é o processo de metacomunicação, a comunicação sobre como, quando, onde e por que se comunicar com o sistema. Esse tipo específico de metacomunicação é definido como uma mensagem dos *designers* para os usuários, uma peça de comunicação

completa e complexa cujos significados se revelam à medida que o usuário interage com o sistema (Salgado *et al.*, 2013).

A Engenharia Semiótica propõe que o *designer* preencha o seguinte *template* para construir a mensagem de metacomunicação: "*Está é a minha interpretação sobre quem você é, o que eu entendi que você quer ou precisa fazer, de que formas prefere fazê-lo e por quê. Este é portanto o sistema que eu projetei para você, e esta é a forma que você pode ou deve usá-lo para atingir os objetivos incorporados na minha visão*". Esta é uma forma abstrata de representar o conteúdo completo da mensagem que os *designers* comunicam através da interface para os usuários e que estes vão recebendo à medida que interagem com o sistema. Quando o *designer* descreve a sua interpretação sobre quem o usuário é, ele deve apontar as necessidades, crenças, oportunidades, preferências e contexto em que estas características fazem sentido. O *designer* deve esclarecer as atividades e metas que o usuário deve poder realizar, descrevendo o modo como a comunicação deve ocorrer dependendo do tipo de tecnologia que está disponível para o sistema. Apesar de o *designer* ter construído o sistema de determinada forma para que o usuário alcance os objetivos previamente definidos, o usuário tem liberdade para explorar os caminhos de comunicação habilitados no sistema de acordo com a sua própria intenção.

A Figura 3.1 ilustra o esquema geral do processo de metacomunicação em IHC de acordo com a Engenharia Semiótica. Na emissão da mensagem, o *designer* está contando aos usuários a sua visão sobre quem eles são. Na recepção da mensagem, os usuários percebem a mensagem do *designer* na medida em que interagem com o sistema. O usuário pode reagir à comunicação do *designer* mesmo que não esteja consciente de quem está comunicando a mensagem recebida. O *designer* tem a escolha de deixar sua presença mais ou menos explícita através do sistema (Salgado *et al.*, 2013). Essa característica está relacionada à questão que defendemos da autoexpressão através de *software*.

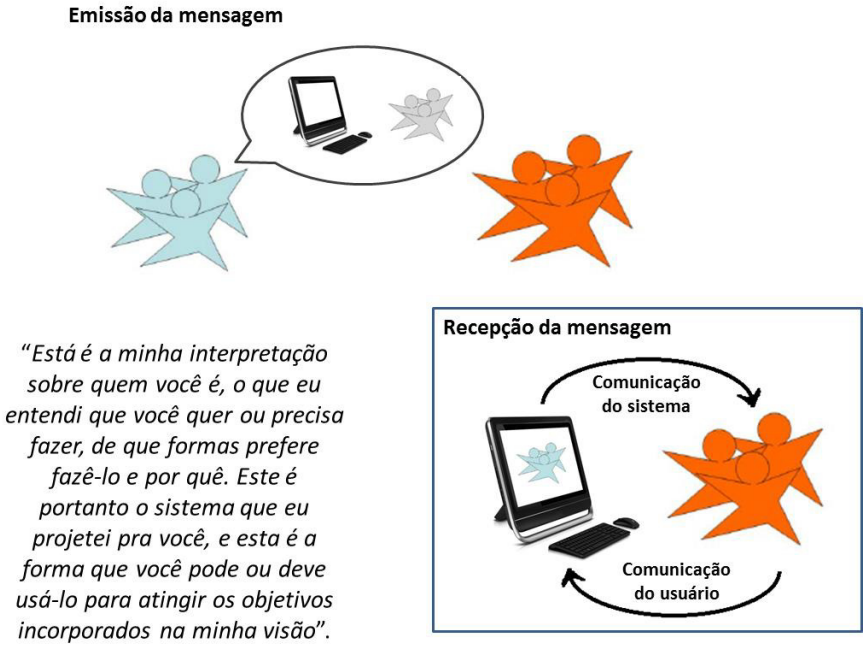


Figura 3.1 - Esquema geral de metacomunicação em IHC
Fonte: adaptada de Salgado *et al.* (2013)

A mensagem de metacomunicação é transmitida pelos *designers* aos usuários através de signos da interface do sistema que podem ser definidos em três classes (de Souza e Leitão, 2009): signos estáticos, signos dinâmicos e signos metalinguísticos.

Os signos estáticos têm o seu significado interpretado independentemente do tempo e de relações causais. O contexto de interpretação é limitado aos elementos que estão presentes na interface em um instante determinado no tempo, como por exemplo, barras de ferramenta e itens de menu.

Os signos dinâmicos surgem da interação e devem ser interpretados em relação ao desdobramento da interação no tempo. Eles são dependentes de relações causais, como por exemplo a ação em consequência de um botão clicado ou a sequência de ações para salvar um arquivo. Outros exemplos são signos animados como a ampulheta de tempo e uma barra de progressão de um *download*.

Os signos metalinguísticos fazem referência aos signos estáticos, dinâmicos, e até mesmo metalinguísticos, tipicamente com o objetivo de explicar os outros signos. Geralmente eles se apresentam em forma de dicas, explicações, mensagens de erro, alertas e mecanismos de ajuda. Os signos metalinguísticos podem ser, eles mesmos, estáticos ou dinâmicos.

Estes diferentes tipos de signos são disponibilizados no meio computacional. Para que sejam codificados, precisam ter um significado definido no sistema. Esta é uma limitação que deve ser levada em consideração na produção dos signos que fazem parte da comunicação mediada por computador.

A qualidade do sistema que verifica a eficiência e a efetividade da metacomunicação é a comunicabilidade, a habilidade do sistema de significar e comunicar a intenção do *designer* em relação ao usuário, que normalmente é a de satisfazer suas necessidades, anseios e preferências (de Souza, 2013).

Com esses conceitos básicos da Engenharia Semiótica em mente, podemos apresentar alguns conteúdos mais avançados da teoria, necessários para o entendimento do modelo proposto. A Engenharia Semiótica apresenta, como ferramenta epistêmica para o *design* de aplicações customizáveis e extensíveis, taxonomias com modificações que podem ser feitas em sistemas de significação, tanto da perspectiva do sistema quanto da perspectiva do usuário. Por um lado, a perspectiva do sistema analisa as dimensões léxicas, sintáticas e semânticas. Por outro lado, a perspectiva do usuário explora as dimensões semióticas de intenção, expressão e conteúdo. Essas taxonomias sugerem que diferentes classes de modificações correspondem a diferentes graus de complexidade semiótica para computadores e para pessoas.

Nesta tese, fazemos uma apropriação das taxonomias propostas na Engenharia Semiótica (de Souza, 2005; de Souza e Barbosa. 2006). Ao lidar com processamento de linguagem de programação, podem ser realizadas modificações considerando as dimensões léxicas, sintáticas e semânticas. A dimensão léxica está relacionada com o vocabulário usado para formar sentenças significativas. A dimensão sintática se refere aos diferentes tipos de combinações e estruturas que podem ser formadas com o vocabulário. A dimensão semântica trata do significado de itens lexicais e estruturas sintáticas.

Uma modificação ocorre quando a dimensão é reduzida, expandida ou alterada. A Tabela 3.1 apresenta as combinações possíveis de acordo com modificações em uma ou mais dimensões (léxica, sintática e semântica). As células pintadas indicam as dimensões alteradas para cada tipo definido. As combinações em cinza claro foram descartadas por de Souza (2005). As linhas com células em cinza escuro são particularmente importantes para o modelo proposto nesta tese.

Tabela 3.1 - Operações simbólicas na perspectiva do sistema

	Tipo	Léxico (vocabulário)	Sintático (gramática)	Semântico (significado)
Modificação léxica preservando o significado	I	■		
Modificação sintática preservando o significado	II		■	
Expansão estrutural preservando o significado	III	■	■	
Modificação léxica e semântica	IV	■		■
Expansão estrutural alterando o significado	V		■	■
Expansão semântica	VI			■
Expansão linguística com tipos controlados	VIIa	■	■	■
Expansão linguística com tipos livres	VIIb	■	■	■

Fonte: adaptada de (de Souza, 2005)

Note que os tipos I, II e III não apresentam mudança na dimensão semântica; isso que dizer que eles mantêm o significado original (no tocante ao programa), apesar das mudanças em outras dimensões. Este aspecto específico diferencia uma aplicação customizável (que preserva o significado) de uma aplicação extensível (que pode mudar o significado), considerando-se a semântica "de programa" (e não a possível interpretação de tais mudanças por parte de uma pessoa). Exemplos típicos de customização e extensão podem ser notados em navegadores de Internet. A customização é, por exemplo, a mudança na aparência, na posição dos elementos e a configuração de barras de ferramentas. A extensão ocorre através dos *plug-ins* que podem ser instalados, eles fornecem uma funcionalidade adicional ao navegador. Em qualquer caso, os navegadores devem manter certos signos cujas formas e significados não podem ser alterados por fazerem parte da identidade da aplicação.

O tipo IV apresenta novo vocabulário, mantém a gramática e introduz nova semântica na especificação original da linguagem. Na interpretação da autora (de Souza, 2005), estes novos itens do vocabulário estão previstos na geração de estruturas já existentes na gramática da linguagem, e são, por isto, mapeados em novas instâncias de categorias semânticas pré-existentes. Este tipo de extensão está presente, por exemplo, em todo tipo de aplicação que permite aos usuários criar e dar (novos) nomes a objetos como documentos, imagens, projetos, etc. Em outras palavras, embora se trate de uma "extensão", no sentido estrito do conceito, é uma extensão intrínseca à própria funcionalidade de grande parte dos aplicativos utilizados pelos usuários e a característica não ajuda a criar distinções relevantes na teoria.

De Souza (2005) também descarta o tipo V de combinações entre mudanças lexicais, sintáticas e semânticas na perspectiva de sistema. Isto porque a

introdução de novos significados pela mera reestruturação de elementos sem modificação léxica é uma condição desfavorável. A restrição de não marcar diferença de significado com diferenças lexicais (além de outras) poderia introduzir dificuldades desnecessárias de interação.

Finalmente, em relação ao descarte do tipo VI, isto é proposto porque a mudança ocorre apenas na dimensão semântica, sem afetar o vocabulário e a gramática. Ou seja, seria como se, para um mesmo programa de computador, um mesmo comando pudesse passar a dizer coisas diferentes sem que isto estivesse associado a qualquer marcação sintática. Esse tipo de mudança não é desejável porque pode comprometer a identidade da aplicação, além de introduzir ambiguidades semânticas que, do ponto de vista de IHC, não são desejáveis.

Percorrendo então as linhas relevantes da Tabela 3.1, o tipo I está relacionado à mudança do vocabulário sem alteração sintática e semântica. Na maior parte das vezes, consiste no mecanismo de renomear signos que já existem na aplicação, ou ainda, criar sinônimos para denominar o mesmo objeto. No tipo II, há mudança apenas na dimensão sintática. Nesse caso, o vocabulário e o significado são mantidos e os mesmos signos podem ser reordenados sem mudança no significado. Por exemplo, algumas aplicações permitem reordenar itens de menu. A combinação de mudanças léxicas e sintáticas preservando o mesmo significado define o tipo III, que geralmente consiste na criação de novos signos estruturados em uma ordem diferente, como o que costuma acontecer na gravação de macros.

O tipo VII representa mudança em todas as dimensões. Por causa da complexidade desse tipo de mudança, ele é subdividido entre VIIa (expansão controlada) e VIIb (expansão livre). No tipo VIIa, apenas um grupo dos tipos de itens léxicos, sintáticos e semânticos pode ser alterado. Por outro lado, no tipo VIIb, em teoria, qualquer tipo de item léxico, sintático e semântico pode ser alterado. Na nossa apropriação para esta tese, desconsideraremos esta diferença. O *designer* do documento ativo em construção em última análise o único responsável por limitar alterações que possam, a seu ver, prejudicar a identidade da aplicação ou introduzir descontinuidades semióticas no sistema de significação.

As combinações na perspectiva do sistema podem então ser simplificadas em dois grandes tipos: reformulação (tipos I, II e III), combinações de dimensões

léxicas e sintáticas (sem alteração semântica); e reescrita (VII), alterações nas três dimensões (com alteração semântica).

Da perspectiva do usuário, as dimensões consideradas na taxonomia proposta pela Engenharia Semiótica, conforme já mencionado, são: expressão, conteúdo e intenção. A Tabela 3.2 apresenta todas as possiblidades de mudanças. Assim como na tabela anterior, as combinações em cinza claro foram descartadas por de Souza (2005) e as combinações na cor cinza escuro são as mais importantes no contexto desta tese. Uma divisão importante na tabela é a manutenção ou expansão na dimensão da intenção.

Tabela 3.2 - Operações semióticas na perspectiva do usuário

		Tipos	Expressão (forma)	Conteúdo (informação)	Intenção (ação)
Independente da intenção	Mudança de forma	A			
	Mudança de conteúdo	B			
	Mudança de forma e conteúdo	C			
Expandindo o escopo da intenção	Paráfrase	D			
	Figuras de linguagem	E			
	Ressignificação	F			
	Expansão linguística	G			

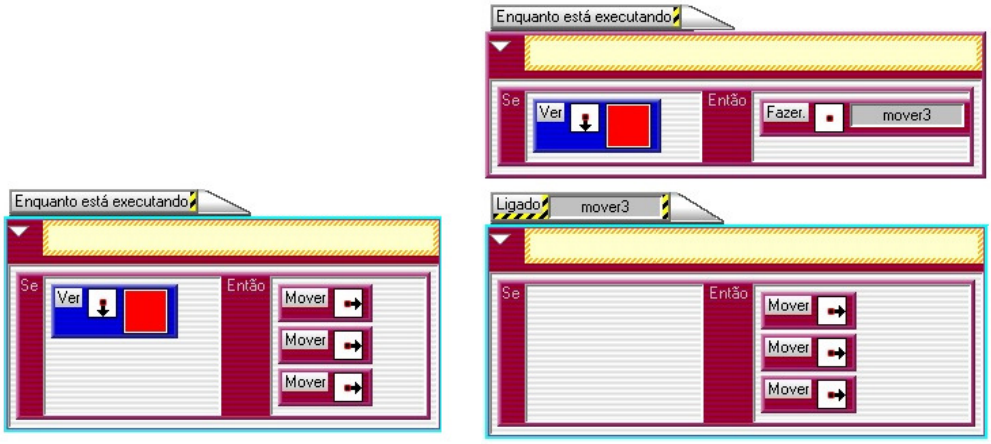
Fonte: de Souza (2005)

De maneira geral, é difícil imaginar uma modificação que não seja motivada por uma nova intenção. Toda modificação que um usuário realiza habitualmente vai ao encontro de um novo objetivo ou intenção de comunicação. Porém, nesta tese, vamos considerar que existem novas intenções apenas quando ocorre a criação de novas metas em uma hierarquia de um modelo de tarefas.

De acordo com o *design* de IHC, os modelos de tarefas apresentam a estrutura hierárquica das tarefas do usuário que compõem um objetivo e as estruturas de sequência e iteração. Os modelos também exibem tarefas alternativas, independentes de ordem, opcionais e ubíquas (Barbosa e Silva, 2010), tornando-se o equivalente a uma representação de planos racionalizados para realizar, computacionalmente, um conjunto de metas a eles associadas. Por exemplo, no domínio de ensino de programação, podemos dizer que apresentar um mesmo conceito de outra forma com a intenção de explicar ou esclarecer, criando uma paráfrase, faz parte do modelo de tarefas na medida em que se pode traçar, racionalmente, planos para realizar estes objetivos, os quais são implementados computacionalmente.

Desse modo, foram desconsiderados na taxonomia os tipos A, B e C, onde as combinações formadas não possuem nova intenção. No contexto desta tese, os tipos que não possuem novas metas especificadas no modelo de tarefas com a intenção de auxiliar no processo de ensino e aprendizado não são importantes.

O tipo D engloba, além de expansão da intenção, mudanças na expressão, isto é, na forma. Em outras palavras, o tipo D é definido como uma paráfrase, uma espécie de tradução de um mesmo conteúdo, que é expressa pelo usuário para contemplar uma mudança em suas intenções quanto à interação com a aplicação extensível. A Figura 3.2 apresenta um exemplo de uma paráfrase no AS. As imagens (a) e (b) refletem um mesmo conteúdo expresso de formas diferentes. Em (a), há uma sequência de ações 'mover'. Enquanto que (b) propõe uma refatoração do código transformando a sequência de ações em um novo método (mover3) que poderá ser chamado em outros pontos do programa.



(a) (b)

Figura 3.2 - Exemplo de paráfrase no AS

O tipo E é caracterizado como uma figura de linguagem (tipicamente metáfora ou metonímia), onde um novo conteúdo é introduzido apesar de manter a dimensão vocabular da expressão anterior (metáforas e metonímias são formadas por combinações literalmente "agramaticais" de vocábulos pertencentes à linguagem). Por exemplo, suponha que um método 'saltar' seja anunciado no programa do AS conforme a Figura 3.3 (a), isso significa que qualquer agente que saiba responder a 'saltar' executará as ações correspondentes. Apesar do nome do comando ser o mesmo, cada agente tem liberdade para ser programado de forma diferente. Note que o salto é diferente nas imagens (a) e (b) da Figura 3.3.

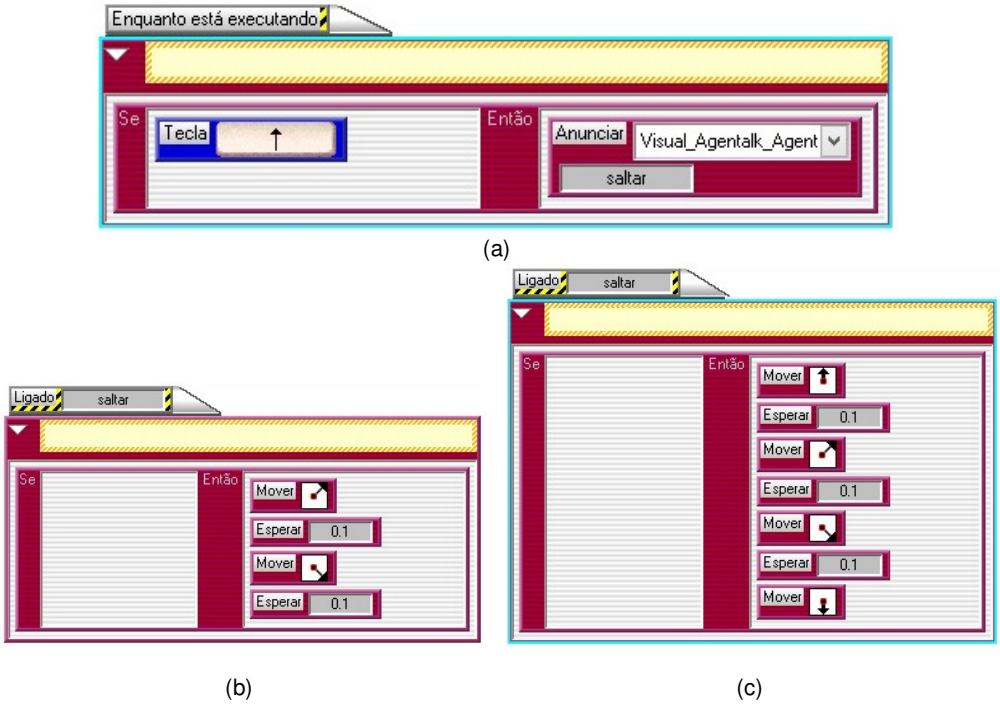


Figura 3.3 - Exemplo de figura de linguagem no AS

O tipo F consiste apenas na mudança da intenção, tipicamente uma funcionalidade da aplicação é usada com certa intenção para um determinado fim, mas passa a ser usada com outra intenção e para outro fim, caracterizando uma ressignificação. Um exemplo de ressignificação no AS é o uso de máscara de cores em aparências dos agentes. Na Figura 3.4, considere o agente da imagem (a), que tem um contorno amarelo. Se escolhermos mascarar a cor amarela, o agente ficará como na imagem (b), com o contorno transparente. Isto é usado para melhorar o visual do jogo ou simulação quando os agentes estão sobrepostos. Assim, quando o agente da imagem (b) ficar por cima do agente da imagem (c) o resultado é a imagem (d). Podemos atribuir outro significado à funcionalidade de máscara quando escolhemos colorir um agente com uma cor única e mascarar essa cor; desse modo o agente ficará completamente transparente. Isso pode ser usado para sensorear uma determinada área da planilha e permitir o uso do comando 'mover aleatoriamente em', que necessita de um agente de parâmetro; só é possível mover aleatoriamente em cima de outro agente, o que está sendo mostrado na regra da imagem (e).

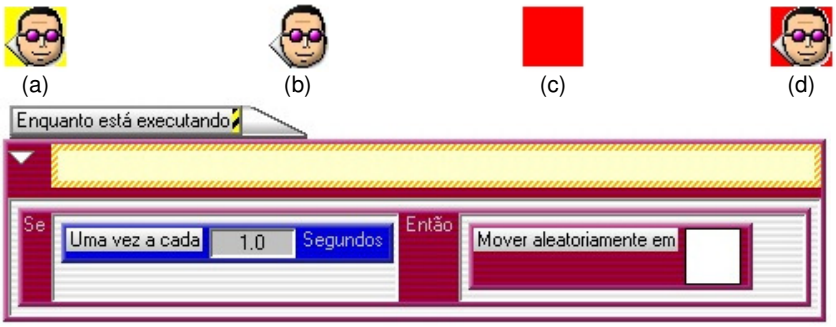


Figura 3.4 - Exemplo de Ressignificação no AS

O tipo G inclui uma mudança mais complexa. O usuário precisa inserir novas expressões e novos conteúdos para expressar nova intenção. Esse tipo de modificação requer mais conhecimento do usuário, em alguns casos, inclusive, conhecimento de linguagens de programação. No AS, a expansão linguística acontece através da criação de novos métodos (como na Figura 3.3), mas o reuso de métodos é limitado aos agentes e ao projeto envolvidos. Além disso, é possível adicionar recursos construídos em Java no AS, desde que sejam seguidos os requisitos impostos pela ferramenta.

Retomando os principais conceitos abordados, temos: a metacomunicação, a classificação dos signos e as taxonomias de modificações. O entendimento do processo de metacomunicação é importante porque está em linha com a definição de raciocínio computacional que nós adotamos para esta tese. Nós acreditamos que a programação deve ser vista como uma forma de autoexpressão. O processo de metacomunicação tem como foco a mensagem que o *designer* envia para o usuário através do sistema.

Os signos são a forma de representar a metamensagem construída pelos *designers* no processo de metacomunicação. A combinação e a escolha adequada dos signos (estáticos, dinâmicos e metalinguísticos) são fundamentais para que a metamensagem seja comunicada e recebida com sucesso. Além disso, esta combinação de classes de signos também pode despertar a atenção dos designers para novas possibilidades de expressão de acordo com a intenção a ser alcançada.

Já em relação à taxonomia de modificações, uma das vantagens de classificar as dimensões semióticas de aplicações customizáveis e extensíveis é chamar a atenção do *designer* para as possibilidades de expansão mais significativas dentro do contexto de sua aplicação. Além disso, estudar a correspondência entre as perspectivas do usuário e do sistema faz com que o

designer possa avaliar mais claramente os custos e benefícios de explorar mecanismos de interpretação mais elaborados. O *designer* pode entender melhor as motivações dos usuários e o modo que eles escolhem para se expressar ou expandir o sistema de significação, incluindo novas expressões, conteúdos e intenções. Conhecer mais sobre as possibilidades de modificações da perspectiva do sistema e do usuário ajuda os *designers* a decidir se, e como, eles querem expandir as possibilidades comunicativas permitidas pelo sistema.

No contexto desta tese, podemos explorar essas taxonomias para ajudar o *designer* a criar facetas com combinações da perspectiva do usuário e do sistema. As facetas apresentarão paráfrases, figuras de linguagem, ressignificações e expansões linguísticas que ajudarão os professores e alunos a compreenderem como os programas podem incorporar novas intenções, expressões ou conteúdos. Ao mesmo tempo, as facetas projetadas pelo *designer* envolverão reformulações (sem alterações semânticas) e reescritas (com alterações semânticas). Elas são importantes para despertar nos aprendizes a noção de como as facetas são representadas do ponto de vista do sistema. Os programadores devem ser capazes de perceber a correspondência entre o que está sendo codificado e o resultado final do produto, pois o modo como os conteúdos foram expressos comunica uma intenção para o usuário.

3.2. Estudos empíricos

Durante os últimos quatro anos do projeto SGD-Br, acompanhamos diversos grupos de professores e estudantes no processo de ensino e aprendizado de raciocínio computacional com o AS. No entanto, vamos selecionar apenas alguns dos estudos realizados com o foco específico em resultados relacionados ao processo incremental de elaboração do modelo conceitual PoliFacets e de instanciação do PoliFacets-AS.

Como anteriormente introduzido na seção 1.2, os estudos empíricos foram organizados em três fases. A Fase 1 foi realizada exclusivamente no contexto da tese, enquanto que as Fases 2 e 3 aconteceram como parte da pesquisa realizada no projeto SGD-Br. A Fase 1 correspode ao primeiro estudo empírico (Mota *et al.*, 2012) com o PoliFacets-AS (v0), realizado com professores do ensino fundamental e médio e com especialistas em AS, no primeiro semestre de 2012.

Em seguida, em julho de 2012, aconteceu o primeiro *workshop* de professores de escolas convidadas, tendo por objetivo principal apresentar as ferramentas disponíveis: o AS e o PoliFacets-AS (v1). Em agosto de 2013, já com uma nova versão do PoliFacets-AS (v2), desenvolvida com resultados colhidos em 2012, ocorreu o segundo *workshop* de professores de escolas parceiras do projeto e outros professores convidados. Nesse segundo momento, o uso do PoliFacets-AS foi reforçado, inclusive com uma estratégia diferenciada de apresentação. Entre os *workshops* foram realizados estudos sobre as regras e narrativas de programas e sobre a conexão entre os agentes. A Tabela 3.3 exibe a organização dos estudos empíricos exploratórios que foram relevantes para o desenvolvimento do modelo conceitual de *design* proposto.

Tabela 3.3 - Organização dos estudos entre as fases da pesquisa

Fase	**Versão do PoliFacets-AS**	**Estudos**	**Mês/ano de realização**
FASE 1 Pesquisa preliminar sobre a extensão de representações do AS	PoliFacets-AS v0	Estudo Inicial	06/2012
FASE 2 Estudos empíricos ampliados	PoliFacets-AS v1	I *workshop* para professores	07/2012
		Regras e narrativas de programas	08-11/2012
		Expressão da conexão entre os agentes	02/2013
FASE 3 Estudos empíricos finais	PoliFacets-AS v2	II *workshop* para professores	08/2013
		Estratégias de comunicação	10/2013

Três escolas parceiras participaram do projeto SGD-Br. Para manter a anonimização e consistência na apresentação dos estudos nos referirmos às escolas como A (pública), B (particular-nacional) e C (particular-internacional). Os participantes serão identificados como P<numeral> em cada estudo. Apesar de alguns participantes estarem presentes em mais de um dos estudos, o escopo das identificações se restringe a cada estudo separadamente. Cada um dos estudos contribuiu de alguma forma para a construção do modelo PoliFacets. Ao apresentar os resultados específicos de cada estudo, mostramos como o aprendizado adquirido serviu de insumo para a construção do modelo.

3.2.1. Estudo inicial

Objetivo

O objetivo do estudo inicial foi explorar e aprofundar nosso conhecimento sobre como um tipo diferente de extensão *Web* para o AS poderia trazer benefícios aos aprendizes durante a exploração dos significados dos programas (Mota *et al.*, 2012).

Procedimento para coleta de dados e investigação

Considerando que dois grupos de participantes estavam envolvidos, as etapas da pesquisa para cada grupo foram diferenciadas a fim de melhor aproveitar o conhecimento e a experiência desses grupos. O procedimento para o grupo 1 foi consistiu das seguintes etapas:

- Apresentação guiada no PoliFacets-AS, destacando os pontos principais em cada faceta;
- Apresentação de uma versão simplificada do jogo *Pacman* (como na Figura 3.5), com os agentes sem regras;
- Solicitação de que completassem a programação do jogo para cumprir as metas de: (1) fazer o agente 'comer' as bolinhas verdes pelo caminho que percorresse ao se movimentar sobre elas; (2) fazer o agente chegar a um ponto na planilha, quando então exibiria uma mensagem; (3) e, imediatamente após isto, realizar uma mudança de fase;
- Realização, como etapa final, de um grupo de foco onde coletamos dados sobre a experiência dos participantes em sala de aula, basicamente relacionada às principais dificuldades de ensino e aprendizado com suas turmas e às expectativas de um futuro uso do PoliFacets-AS em sala.

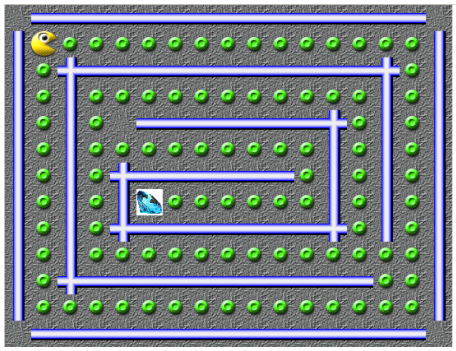


Figura 3.5 - Projeto *Pacman* simplificado

Enquanto isso, para o grupo 2, nós realizamos as seguintes etapas:

- Apresentação guiada no PoliFacets-AS, destacando os pontos principais em cada faceta (semelhante à apresentação para o grupo 1);
- Realização de uma entrevista com a intenção de saber a impressão deles sobre o PoliFacets-AS e os principais problemas enfrentados no ensino.

Perfil dos participantes

Os participantes voluntários foram divididos em dois grupos: o grupo 1 era formado de professores; e o grupo 2 era formado por especialistas (professores dos professores) envolvidos no ensino de programação da escola A. Os três professores participantes do grupo 1 eram formados em biologia (P1), matemática (P2) e pedagogia (P3), este último ministrando o curso de AS na disciplina de matemática. Eles não estavam suficientemente familiarizados com a questão do ensino de programação. Já os dois especialistas do grupo 2 eram da área de computação, um estudante de mestrado (P4) e uma doutora (P5).

Principais resultados

Os resultados deste estudo reforçaram a necessidade de se construir mecanismos de ajuda para os aprendizes, seja para os professores no primeiro contato com a ferramenta, seja para os alunos das turmas.

Os dados coletados foram organizados em quatro categorias: problemas de abstração; lógica da máquina de execução do AS; reconhecimento do valor do PoliFacets-AS; diferentes aplicações e novas funcionalidades para o PoliFacets-AS.

Problemas de abstração

Entre os problemas relacionados ao entendimento dos programas, a observação dos professores ao tentar executar a tarefa solicitada levou-nos a notar a falta de segurança deles sobre que agentes devem conter as regras de comportamento e quais são os sujeitos da ação. O especialista P5 relatou que "*na criação de jogos novos, os alunos sentem dificuldades para atravessar a 'ponte' entre a definição do jogo e sua respectiva programação. Por onde começar é uma dúvida recorrente"*, quando falava das principais dificuldades dos alunos. O PoliFacets-AS pode ajudar a "*atravessar essa ponte*" por meio da disponibilização de exemplos de jogos e "*de sua respectiva programação*", afinal as facetas

apresentam uma nova representação que funciona como uma desconstrução do jogo.

Outra questão de falta de abstração é observada na dificuldade de compreensão ampla dos conceitos de representação do agente e classe do agente. Durante a execução da tarefa solicitada, notamos que os próprios professores têm problemas para perceber as diferenças entre classe (agente) e instância (representação). No AS, nós programamos o comportamento de um agente e inserimos instâncias deste agente na planilha de trabalho, cada instância pode estar em um contexto diferente e tem um comportamento especial por causa disso.

Lógica da máquina de execução do AS

A lógica de execução do AS não está absolutamente clara (as regras são testadas na ordem em que foram dispostas). Mesmo que todos os agentes fossem testados ao mesmo tempo, cada um deles tem as regras em uma sequência específica. Ocorre que, se dois agentes testam uma mesma condição, o resultado pode não parecer consistente, pois o processamento realizado pelo interpretador dos programas em AS não é de fato paralelo. Ou seja, tratando-se de uma "simulação" de processamento paralelo, o interpretador do AS introduz certas características de execução nas regras programadas pelos aprendizes e professores que são difíceis de entender, sem um sofisticado nível de alfabetismo computacional (que é justamente o que o raciocínio computacional visa promover). Quando os professores enfrentam esse tipo de caso, eles acabam trocando a ordem das regras como uma resposta irrefletida, como tentativa que vem à mente, sem pensar muito na solução, porque em alguns casos, por razões que o programador desconhece, a estratégia de alterar ordem das regras já funcionou. Alguns até percebem o problema, como P1 explicitou a respeito de certo contexto de programação "*eu acho que isso vai causar conflito*". No entanto, P1 não conseguiu explicar o que seria exatamente esse conflito.

Reconhecimento do valor do PoliFacets-AS

Em relação ao PoliFacets-AS, os participantes perceberam o valor que a ferramenta poderia agregar ao processo de ensino e aprendizado. P2 e P3 comentaram sobre a faceta 'planilhas', que apresenta o posicionamento dos agentes na planilha de trabalho do AS, dizendo que era importante saber quantos agentes existiam e onde eles estavam localizados, pois, por mais de uma vez, eles

foram forçados a reconstruir suas planilhas porque criaram agentes empilhados ou que ficaram perdidos (fora da visualização), mas não sabiam quantos eram e onde estavam. P4 disse: "*Acho que pode ser uma excelente ferramenta para mostrar aos alunos uma outra visão do jogo. Muitas pessoas, inicialmente, não enxergam um determinado problema, mas não por não o conhecerem, mas por não visualizarem ele. O [PoliFacets-AS], trazendo esta outra visão, pode ajudá-lo a encontrar estes erros e descobrir novas possibilidades para o jogo*". Esse é um dos resultados esperados do PoliFacets-AS: permitir que o aprendiz reflita sobre seus projetos usando as diferentes facetas disponíveis.

Diferentes aplicações e novas funcionalidades para o PoliFacets-AS

Três novas funcionalidades foram solicitadas: um sistema de perguntas recorrentes respondidas, como FAQ (Frequently Asked Questions); uma especificação para cada célula na grade da planilha, seguindo o padrão usado em *software* como o Microsoft EXCEL, por exemplo (letras nas colunas e números nas linhas); e um modo de visualização das relações entre os agentes. A segunda foi incorporada facilmente na faceta 'planilhas'. Com relação à última, P4 disse: "*O [PoliFacets] poderia permitir a visualização da relação entre agentes. Por exemplo, se quisermos ver que agentes trocam mensagens entre si, quais agentes podem se encontrar na planilha, entre outros."*. Essa sugestão foi a semente para a realização do estudo que resultou na criação da faceta 'conexões'.

Contribuição para a elaboração do Modelo PoliFacets

Quanto à construção do modelo PoliFacets, o estudo ressaltou a importância das facetas construídas até aquele momento e forneceu requisitos para melhorias nos conteúdos das facetas existentes, assim como requisitos para novas facetas criadas posteriormente. As avaliações e melhorias para a instância PoliFacets-AS viabilizaram os estudos empíricos posteriores e a elaboração teórica do modelo.

3.2.2. I *workshop* para professores

Objetivo

Do ponto de vista do SGD-Br, o objetivo do primeiro *workshop* para professores foi apresentar o projeto para novas escolas e convidá-las a formar uma parceria com o SGD-Br. A escola A já era parceira desde 2010, as escolas B e C

firmaram a parceria a partir deste *workshop*. Assim, em julho de 2012, com a realização deste *workshop*, o projeto consolidou a formação de um grupo heterogêneo de parceiras, com uma escola pública, uma privada nacional e uma privada internacional.

O objetivo deste estudo para esta tese foi apresentar o PoliFacets-AS (v1) e perceber como os participantes reagiam a suas funcionalidades, utilidades e possibilidades de uso nas suas escolas. Esta foi a primeira apresentação oficial do PoliFacets-AS (v1) no projeto SGD-Br como ferramenta auxiliar no processo de ensino e aprendizado de programação.

Procedimento para coleta de dados e investigação

O cronograma de atividades do primeiro *workshop* para professores está detalhado no Quadro 3.1. O PoliFacets-AS foi apresentado como ferramenta de apoio, mas apenas no último turno os participantes fizeram *upload* dos jogos para realizar a apresentação do resultado final aos colegas.

Quadro 3.1 - Cronograma do primeiro *workshop*

Dia - Turno	Programação
Dia 1 - Manhã	Jogos! O AgentSheets Criação de agentes Movimento de agentes Como apagar agentes Como mudar a representação de agentes
Dia 1 - Tarde	Agentes criando agentes Movendo agentes em cima de outros agentes Exibição de mensagens e troca de planilha
Dia 2 - Manhã	Juntando as peças: O Frogger e o Ghost Atributos Métodos
Dia 2 - Tarde	Construindo jogos com o AgentSheets
Dia 3 - Manhã	Subindo seus jogos para o PoliFacets Apresentação de projetos Discussões finais (experiências)

Observamos os participantes durante 20h de *workshop*, mas os dados coletados estão concentrados em três momentos: um questionário anterior ao *workshop* (Anexo - Questionário A do 'I *workshop* para professores'), um questionário posterior (Anexo - Questionário B do 'I *workshop* para professores') e o momento de apresentação dos projetos dos participantes. O questionário final trazia questões sobre o *workshop*, sobre o AS e sobre o PoliFacets-AS.

Perfil dos participantes

Membros das três escolas foram convidados a comparecer. Além de seis professores de ensino médio e fundamental, também participaram três coordenadores. A área de atuação de cada participante pode ser consultada no Quadro 3.2.

Quadro 3.2 - Participantes do primeiro *workshop*

Escola	Participante	Atuação/disciplinas
A	P1	Ensino fundamental / Ciências e biologia
B	P2	Coordenação de informática
	P3	Ensino fundamental e médio / Literatura e produção de textos
	P4	Coordenação do ensino fundamental
C	P5	Ensino fundamental / Ciências físicas, químicas, biológicas e astronomia
	P6	Ensino fundamental / Ciências e arte digital
	P7	Ensino fundamental / Animação e produção de vídeo
	P8	Ensino médio / Manipulação de imagens e produção de filme digital
	P9	Ensino fundamental / *Computer skills*

Resultados

Dos nove participantes do *workshop*, três (P2, P3 e P8) tornaram-se professores em turmas de ensino de raciocínio computacional com o AS no segundo semestre de 2012. P6 também iniciou uma turma no ano seguinte. Seguindo o formato do próprio *workshop*, pesquisadores do projeto acompanharam os professores durante os 12 meses seguintes, em suas aulas semanais, com a missão de coletar dados para pesquisa e para a continuidade do processo de *design* do PoliFacets-AS.

A análise dos dados relacionados ao PoliFacets-AS levaram-nos a identificar as seguintes categorias: fator motivacional; reconhecimento do valor do PoliFacets-AS e suas aplicações; e limitações e dificuldades de uso.

Fator motivacional

Quanto ao fator motivacional, no questionário pré-workshop, os participantes responderam perguntas sobre sua atuação profissional e sobre o uso que fazem de computadores em suas atividades de ensino. A pergunta com respostas mais interessante foi: "*Você tem algum conhecimento anterior sobre raciocínio computacional? Poderia escrever algum comentário rápido sobre seu interesse pelo assunto?*". Seis participantes disseram que não tinham conhecimento prévio. P1 disse: "*Não tenho, é justamente o que me motiva a participar do evento*". Três participantes disseram que tinham algum

conhecimento. Entre eles P2, que tem formação acadêmica em informática. P8 disse que fez "*alguns semestres em ciência da computação, onde teve a oportunidade de estudar lógica da computação*", enquanto P6 relatou que usava o Scratch com seus alunos da sétima série.

No questionário pós-workshop, ao responder sobre pontos positivos, P3 disse: "*já estou pensando nas inúmeras possibilidades que ele [AS] me dá pra trabalhar com os alunos"*. P8 também mostrou motivação dizendo: *"Estou super-animada! Esse é o sentimento que esta ferramenta [AS] me trouxe, pois consigo perceber uma forma quase ilimitada de uso pelos alunos e já posso imaginar o bem que este aprendizado possa lhes oferecer!*".

Reconhecimento do valor do PoliFacets-AS e suas aplicações

Durante o *workshop*, o PoliFacets-AS foi utilizado basicamente para exposição dos projetos. Entretanto, ao perguntarmos aos participantes que tipo de uso do PoliFacets-AS, eles responderam que pensavam que a ferramenta seria útil e as respostas foram bem variadas. P8 antecipou que seu uso do PoliFacets-AS seria "*na 'dissecagem' da simulação para concluir o entendimento dos conceitos*". Já outros participantes comentaram sobre o fato de o PoliFacets-AS ser um ambiente de aprendizado e discussão, que contém referências para os conceitos básicos, assim como a apresentação de diferentes maneiras de criação de um mesmo jogo.

Sobre os pontos positivos, o valor do sistema foi reconhecido. O PoliFacets-AS foi definido como uma ferramenta importante para entender como as regras foram construídas, além de servir como um banco de exemplos para o ensino e a aprendizagem e ajudar no planejamento dos projetos construídos com o AS. P9 falou sobre a ferramenta: "*Fundamental. Ao explorarmos as regras e planilhas dos outros jogos apresentados no PoliFacets[AS], podemos aprender muito, e ver detalhadamente como funcionam diversos programas e ações*". P7 elogiou a ferramenta, apontando-a como uma referência: "*Excelente – usei muito como referência especialmente para rememorar as regras e como utilizá-las para servir no meu plano de jogo; como referência de exemplo, para entender melhor a lógica dos jogos"*.

Grande parte dos participantes criou projetos com jogos e simulações relacionados à sua área de atuação. Por exemplo, P1, que é professor de biologia,

criou uma simulação apresentando a criação e o consumo de oxigênio e gás carbônico (Figura 3.6a). P5, professora de ciências, fez uma espécie de cópia do *Frogger*, mas modificou completamente o contexto, o personagem principal era Homer Simpson[31], os caminhões eram *cupcakes*, a travessia do rio era constituída de verduras e frutas. Se o Homer tocasse no *cupcake*, mudava sua representação para um Homer acima do peso e o jogo terminava. Ele precisava evitar os *cupcakes* e ir por cima das verduras (comê-las) para chegar à bandeirinha, então mudava sua representação para um Homer mais magro e o jogo terminava (Figura 3.6b). P8 trabalha com manipulação de imagens e utilizou recursos de importação de figuras no AS para criar um jogo com um visual atrativo (Figura 3.6c). Apesar do *design* arrojado, não houve tempo hábil para a finalização do jogo que tinha uma lógica bem complexa, era preciso movimentar as bolinhas para juntar três da mesma cor e elas desapareciam fazendo com que as bolinhas de cima descessem enquanto houvesse vaga livre na planilha. P3 criou um jogo onde objetos caiam das janelas, o personagem devia se movimentar para alcançar as maçãs, desviando dos gatos e dos vasos de flores (Figura 3.6d). Os jogos apresentados mostram a riqueza da autoexpressão a partir de conteúdos básicos ensinados durante o *workshop*, onde já podemos observar as variações na expressão, conteúdo e intenção dos professores participantes.

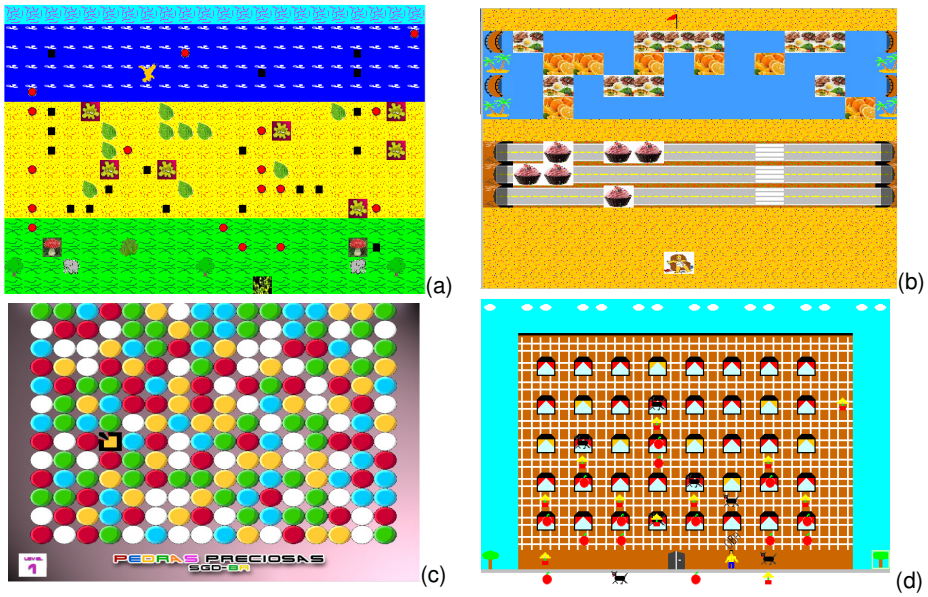


Figura 3.6 - Alguns jogos do primeiro *workshop*

[31] Homer Simpson é um personagem de desenho animado (Os Simpsons) popular nos Estados Unidos. Homer é conhecido por seus hábitos pouco saudáveis.

Esse último jogo (Figura 3.6d) foi escolhido no *workshop* como exemplo de uso do PoliFacets-AS. Em discussão com a turma, nós apresentamos a grade do jogo na faceta 'planilha', mostrando os agentes que estavam perdidos na planilha. Como os participantes foram criando os jogos conforme aprendiam os comandos, em algum momento, P3 começou a gerar os agentes que caíam das janelas em sua planilha de trabalho, mas não havia uma barreira que apagasse esses agentes.

No AS, a planilha é visualizada com um tamanho limitado, o máximo que o monitor permitir. Não há barra de rolagem e os agentes podem ocupar uma posição distante para direita e para baixo. Essa posição não fica visível, os agentes ficam perdidos e não há motivação para suspeitar que há agentes fora da tela visualizada. Certos comportamentos dos agentes podem deslocar agentes para fora da tela de visualização. Por exemplo, considere que um agente A seja programado para criar um agente B para o lado direito a cada meio segundo. O agente B é programado para se mover para direita a cada meio segundo. Desse modo, após iniciar a execução, o agente B vai se mover para direita indefinidamente, enquanto o agente A estará criando agentes B. Em pouco tempo, vários agentes B estarão acumulados no fim da planilha no lado direito. A Figura 3.7 ilustra o processo de geração de B's a partir de A. A cada instante de tempo (de t1 a t5) o agente B move-se para direita e, em t4, sai do campo de visão. Assim, o aprendiz não percebe que novos agentes B continuam sendo gerados.

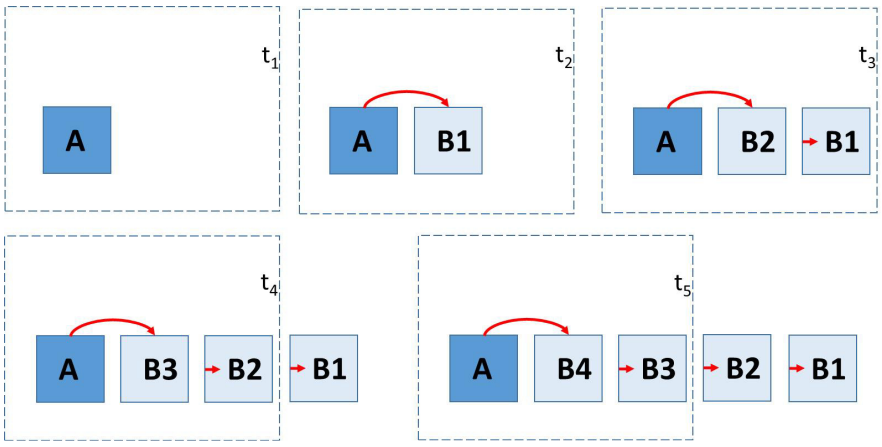


Figura 3.7 - Simulação de agentes fora da janela de visualização

Esse foi o problema de P3, pois ele criava os agentes que iam caindo indefinidamente, sem uma condição para serem apagados. No AS, o único efeito estranho causado na planilha foi a demora para abrir o projeto, sintoma que, no entanto, não foi associado ao fato de que havia muitos agentes perdidos no limite

da planilha. Só o PoliFacets-AS, através da faceta 'planilhas', possibilitou a visualização do problema. Diante do diagnóstico possibilitado pela visualização dessa faceta pelo grupo, P6 perguntou: "*mas e qual foi a solução?*". P3 respondeu: "*a solução é criar um chão para apagar os agentes [que estão] caindo"*. Esse foi um momento de revelação para os participantes, no qual perceberam que o número de agentes presente na planilha era muito grande e que isto não aparecia na visualização do *applet* (ver quantidade de instâncias nos agentes marcados na Figura 3.8). Não é possível ver os agentes apenas usando o AS. Por conta disso, o carregamento do jogo fica lento e, nestes casos, a solução mais comum costuma ser *resetar* e redesenhar a planilha.

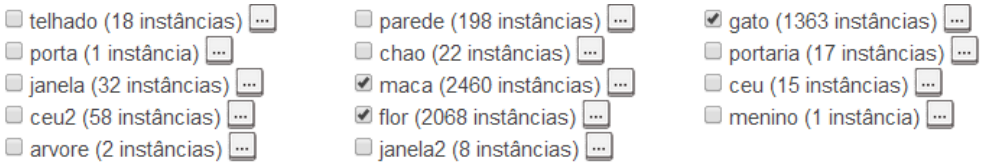


Figura 3.8 - Lista de agentes na planilha do projeto Apartamentos

Limitações e dificuldades de uso do PoliFacets-AS

Podemos considerar que o *workshop* uma oportunidade de estudo produtiva. Todos os participantes concordaram que a finalidade de apresentar e transmitir os conceitos básicos do AS foi plenamente atingida. Vários participantes comentaram como um ponto positivo a grande disponibilidade de monitores (ajudantes) durante o *workshop*. Apesar de eles terem considerado isto um ponto positivo, podemos dizer que a necessidade da presença de monitores era uma grande limitação na nossa pesquisa, isto reforçou ainda mais a ideia do PoliFacets-AS atuar como documento ativo e mediador no processo de ensino e aprendizado. Cumprindo a meta do que preconiza a própria Engenharia Semiótica, o documento ativo é o preposto dos monitores em caso de sua ausência física, supondo-se que os *designers* do PoliFacets-AS pudessem capturar e comunicar adequadamente o discurso destes monitores, tão apreciado pelos participantes do *workshop*.

Entre os pontos negativos do *workshop*, outra dificuldade bastante comentada foi a falta de material didático para consulta. Este caso pode ser observado de duas perspectivas. Por um lado, os professores viram o PoliFacets-AS como um ambiente de ensino tradicional e esperavam encontrar apostilas sobre como utilizar o AS. Por outro lado, nós observamos a necessidade de

expandir a documentação ativa que estávamos propondo, indo além da disponibilização de materiais tradicionais.

Contribuição para a elaboração do Modelo PoliFacets

Este estudo colaborou para o modelo PoliFacets na medida em que os benefícios das facetas existentes foram reconhecidos. Além disso, a descoberta de dificuldades no entendimento dos professores também serviu para percebermos a necessidade de melhorias na 'engenharia semiótica do PoliFacets -AS e de criação de novas facetas que preenchessem as lacunas no processo de compreensão dos programas.

3.2.3. Regras e narrativas de programas

Objetivo

O objetivo do estudo foi melhorar a construção das verbalizações geradas automaticamente para a faceta 'regras'. Esta faceta apresenta uma tradução em linguagem pseudonatural das regras programadas em linguagem visual no AS. Precisávamos saber mais sobre como as pessoas verbalizavam naturalmente seus jogos criados no AS para criar melhores verbalizações.

Procedimento para coleta de dados e investigação

O estudo ocorreu por durante um semestre em uma das escolas parceiras do projeto. O professor queria explorar o desenvolvimento de jogos como construção de narrativas. Coletamos a verbalização dos alunos em várias fases do aprendizado, antes e depois de conhecerem os comandos do AS.

Perfil dos participantes

O estudo contou com a participação de cinco alunos de ensino médio, quatro alunos do primeiro e um aluno do segundo ano. O professor de literatura foi o responsável pelas atividades com a turma.

Principais resultados

A análise dos dados foi realizada considerando como foco o uso da faceta 'regras' no PoliFacets-AS e as categorias que emergiram foram: a diferença entre as verbalizações no decorrer do aprendizado; e novas facetas de regras.

Diferença entre as verbalizações

Percebemos a diferença de verbalização dos alunos no início e no final do curso. Eles fizeram uma redação sobre o jogo que desejavam construir, antes de conhecer o AS, e outra, quando finalizaram os seus jogos. Em um caso interessante, o P1 escreveu a primeira redação com foco na história em torno do jogo e terminou o curso com uma redação completamente técnica, falando dos agentes e seus comportamentos (Apêndice - Material do estudo 'Regras e narrativas de programas'). Enquanto isso, P2, que já conhecia um pouco de programação, fez exatamente o contrário, iniciou com o foco na programação, descrevendo o comportamento dos personagens e entregou uma redação final com o foco na história sobre o jogo que ele havia construído.

Para exemplificar essa categoria, observe trechos das narrativas de P1 e P2. Na primeira etapa de coleta de narrativas, P1 disse: "*O objetivo do jogo é coletar todo o lixo do chão, levar até a lixeira e receber o prêmio em menos tempo*". Enquanto que na segunda etapa, P1 disse: "*O objetivo do jogo é fazer o agente criança coletar todos os agentes lixo (garrafavidro, papelamassado, lata, sacola, papelão, garrafaplastico, panela) e levá-los aos agentes lixeira (lixeirametal, lixeirapapel, lixeiraplastico e lixeiravidro)"*. O segundo trecho está completamente ligado à programação realizada com o AS.

Na mesma atividade, P2 em sua primeira etapa disse: "*No jogo o agente Louis tem o comportamento de ao pressionar a tecla 'para esquerda' ele se move para esquerda, ao pressionar a tecla 'para direita' ele se move para direita, ao pressionar 'para cima' ele se move para cima, ao pressionar 'para baixo' ele se move para baixo, podendo ter combinações se movendo para diagonal*". Na segunda etapa de coleta de narrativas, P2 relatou: "*O jogo The adventurer tem como seu personagem principal o Louis que tem o objetivo de achar a "chave-mestra" do castelo de seu rival Marty para detê-lo de seu plano maligno. (...) O Louis como herói tem sua movimentação baseada nas setas do teclado*". Podemos notar que ele se desprendeu da narrativa ligada à programação.

Novas facetas de regras

A experiência de uso mais intenso dessa faceta explicitou novos requisitos que originariam diferentes tipos de faceta, ainda sobre o mesmo contexto. Por exemplo, a geração de parágrafos que envolvessem o comportamento recíproco de

dois ou mais agentes fortemente conectados com foco em um conjunto de elementos do programa.

O caso de diferenças entre as verbalizações nos alertou para a possibilidade de geração de texto em linguagem pseudonatural explicando as regras de modo a contemplar a mediação entre um texto fortemente relacionado à programação e um texto mais ligado ao domínio do jogo. Neste caso, a exploração dos textos, considerando o domínio de programação e do jogo em si, também resultaria em novas facetas.

Contribuição para a elaboração do Modelo PoliFacets

Este estudo foi importante para observarmos a relevância da faceta 'regras' e seus possíveis desdobramentos com relação à abrangência do escopo das verbalizações e diferenças considerando as perspectivas do usuário e do sistema.

Apesar de termos requisitos para novas facetas, esta ideia ainda não foi colocada em prática no PoliFacets-AS, porque, decidimos explorar o estudo da diversidade das facetas. Ainda assim, a questão da mediação entre o ponto em que o usuário conhece programação e o outro extremo onde ele não conhece programação (inclusive o PoliFacets-AS e o próprio AS) foi utilizada em uma fase posterior da pesquisa e será descrita na subseção 3.2.6, que apresenta uma nova estratégia de comunicação para o PoliFacets-AS.

3.2.4. Expressão de conexões entre agentes

Objetivo

O objetivo deste estudo foi a construção da faceta 'conexões' motivada pela necessidade de representar o relacionamento entre os agentes (Mota *et al.*, 2013). Esta necessidade foi percebida em nossas observações durante o uso do AS nos cursos ministrados pelos professores das escolas participantes do projeto. Até então, não havia como visualizar uma representação regional do programa, considerando a influência que cada agente tinha sobre os outros.

Procedimento para coleta de dados e investigação

Para um levantamento das primeiras impressões, construímos um protótipo da faceta 'conexões' e o utilizamos com participantes com perfil de professores de cursos de programação de jogos com o AS. O protótipo era intencionalmente

rebuscado, no sentido de que a programação dos efeitos desejados e visíveis na tela do jogo (interações explicadas mais adiante entre: as fábricas e a fumaça; a fumaça e as nuvens; e a configuração de nuvens e fumaça e o final da simulação) seguia um padrão com indireções não muito típicas, mas perfeitamente corretas, nas relações causais entre o comportamento dos agentes. Isto foi feito justamente para o pesquisador perceber se, em tal situação não óbvia de programação, a representação das conexões entre agentes ajudaria ou não o participante a raciocinar sobre o problema. O estudo foi dividido em dois ciclos e cada ciclo foi composto pelas seguintes etapas:

- Elaboração do protótipo da faceta;
- Teste com usuários para capturar a recepção da mensagem;
- Análise dos resultados.

No protótipo, cada agente do projeto do AS tinha seu diagrama de conexões. Um exemplo desse diagrama pode ser visualizado na Figura 3.9., por meio da qual podemos observar que o agente A está relacionado aos agentes D e E. Além disso, para conhecer as regras que formam a relação, os aprendizes podem acessar os links disponíveis e escolher a visualização das regras na linguagem pseudonatural automaticamente gerada na faceta 'regras' ou a consulta de um relatório que o AS disponibiliza com uma espécie de pseudocódigo ainda muito próximo da linguagem visual do AS.

Figura 3.9 - Relações do agente A com outros agentes

Para o teste de recepção do primeiro ciclo, criamos uma simulação sobre o desequilíbrio ambiental (Figura 3.10), com proposital alto nível de complexidade e diversas ligações implícitas entre os agentes. Na simulação, o agente C ( ) cria novos agentes do tipo D ( ) de acordo com uma variável controlada pelo agente B ( ). Quando o agente D ( ) encontra o agente E ( ), o agente E muda sua representação para uma nuvem mais escura, gradativamente (,). Quando o agente E muda sua representação para a nuvem mais escura ( ), o agente A muda sua representação (de para). Os agentes F (,) são responsáveis por criar e apagar os agentes do tipo E (, ,). Enquanto o agente A está no seu estado inicial ( ), ele faz o agente D ( ) executar um método para apagar a si mesmo dentro de um determinado tempo. Depois de algum tempo, os agentes do tipo A ( ) estarão todos modificados ( ) e os agentes D ( ) ficarão acumulados acima do agente C ( ). Haverá, ainda, uma mudança no plano de fundo da planilha e a simulação acabará. Os agentes não foram nomeados de forma significativa para não interferir na interpretação dos participantes sobre a simulação.

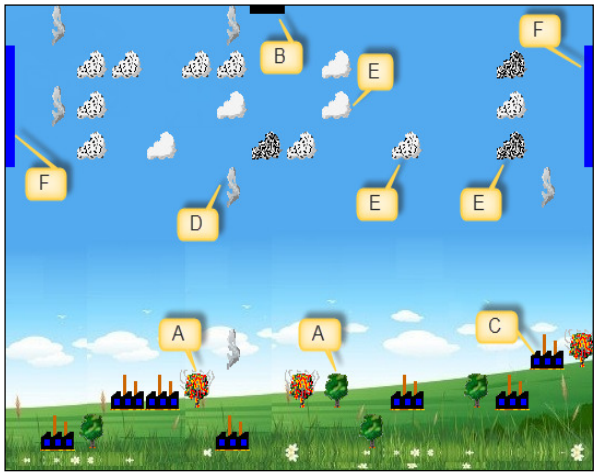


Figura 3.10 - Agentes da simulação

As atividades (Apêndice - Material do estudo 'Expressão de conexões entre agentes') do estudo foram divididas em três partes. Primeiramente, nós pedimos aos participantes para observar a simulação e explicar o que acontecia nela, tentando descrever detalhadamente o seu funcionamento. Em seguida, pedimos a eles que respondessem qual a relação entre os agentes B ( ) e C ( ) e, depois qual a relação entre os agentes A (ou) e C ( ), consultando apenas o material existente no PoliFacets-AS e no AS. Por último, nós apresentamos o protótipo da faceta 'conexões' e pedimos que os participantes comentassem o diagrama.

Depois de eles terem compreendido a intenção e objetivos que diagrama expressa,, nós perguntamos qual era a relação entre os agentes D ( ) e F(,). Eles deveriam responder consultando preferencialmente o material que fazia parte do protótipo da futura faceta 'conexões'.

Perfil dos participantes

Este estudo teve a participação de sete voluntários. Nós os classificamos segundo uma definição de Turkle (2005) que, ao falar sobre o significado de programação, distingue usuários de recursos computacionais em dois grupos: *soft masters* e *hard masters*. Os *soft masters* interagem com o computador e eventualmente compõem programas com padrões de significado que emergem da interação. Os *hard masters* fazem parte do grupo que tem um plano em mente e trabalham racionalmente para implementar esse plano no computador. Nos dois casos, é preciso lidar com um alto nível de complexidade, mas os caminhos para alcançar os objetivos são diferentes. Os participantes estão classificados no Quadro 3.3.

Quadro 3.3 - Organização dos participantes

Perfil	Participante	Ciclo 1	Ciclo 2
Soft masters	P1	X	X
	P2	X	X
	P3	X	X
Hard masters	P4	X	X
	P5	X	X
	P6	X	
	P7		X

Principais resultados

Depois de realizar o segundo ciclo de avaliação e *redesign* do protótipo da faceta, melhoramos o protótipo e perguntamos aos participantes sobre modificações realizadas na simulação, considerando as possíveis consequências com base na relação existente entre os agentes. Os resultados indicaram que os *hard masters* sabiam responder a questão corretamente, enquanto os *soft masters* tiveram problemas. Esses resultados são tão mais relevantes ao associá-los ao fato de que a maior parte dos professores participantes do projeto não é da área de computação, e não há uma restrição quanto aos conhecimentos mínimos de programação que eles devem ter inicialmente para se engajar em trabalhar com o

AS. Em decorrência disto, as facetas devem ajudar especialmente aos *soft masters*, que mais precisam de apoio.

A faceta ‘conexões’ ainda precisa, portanto, de mais *scaffolds* para ser plenamente compreendida e útil a todos os usuários do PoliFacets-AS. Mesmo assim, os resultados obtidos foram suficientes para criarmos a primeira versão da faceta no sistema PoliFacets-AS. Observe, na Figura 3.11, como disponibilizamos as funcionalidades elaboradas no protótipo.

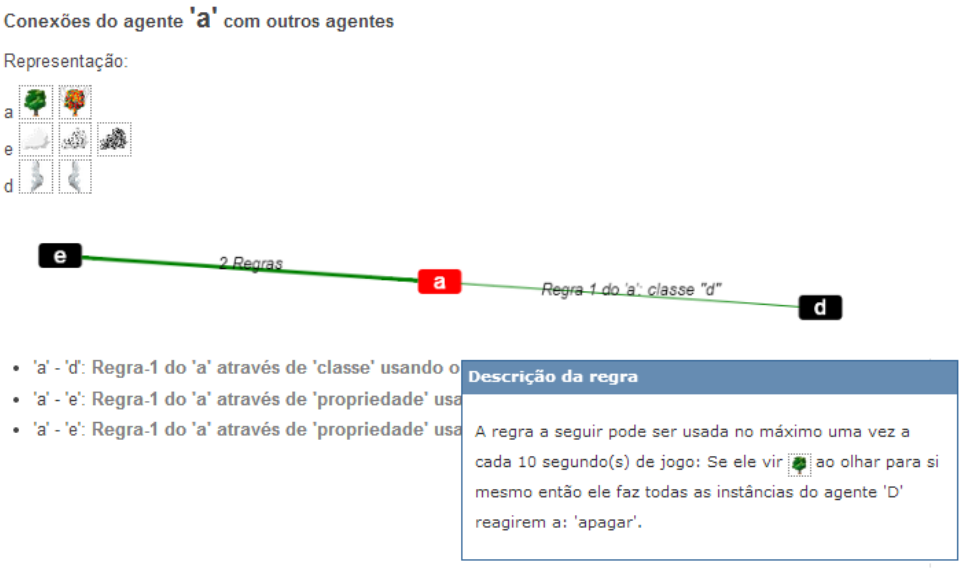


Figura 3.11 - Conexões do agente 'a' com outros agentes

A análise dos dados coletados resultou em quatro categorias: primeira impressão, iconicidade, função de linguagem dominante e transição entre funções de linguagem.

Primeira impressão

Em relação ao que denominamos primeira impressão, quatro dos seis participantes tiveram boas reações quando olharam pela primeira vez para o protótipo da faceta ‘conexões’. P1, no seu primeiro contato, exclamou: “*eu já gostei disso! Eu adoro ícones, gráficos [...] e tem figuras! Eu já gostei!”*. Por outro lado, P6 reagiu mal no primeiro contato dizendo “*você pegou algo simples e deixou tão complicado*”, mas depois de entender melhor como os diagramas funcionavam mudou de ideia e disse: “*isso é bom; as pessoas entendem quando [e] o que vai acontecer [...] isso é muito útil [...] é um mapa [...]*”. Essa categoria é importante porque uma boa impressão da faceta causa uma pré-disposição positiva no esforço dos aprendizes para sua compreensão e uso.

No segundo ciclo de interação, refizemos o protótipo da faceta considerando os resultados encontrados no primeiro ciclo. Pedimos aos participantes que respondessem a um questionário, consultando a nova versão do protótipo. Quanto à primeira impressão, desta vez, o participante P2 que havia dito que o diagrama era muito complicado de entender da primeira vez, admitiu que: "*dado o que olhei ao longo do estudo, digo que a nova versão é mais provocativa do aprendizado, aguça mais a cadeia epistêmica ao falar, anunciar de cara, nas relações, os nomes dos atributos e propriedades, por exemplo. Você saca de cara e apenas aprofunda seu conhecimento clicando nos detalhes. Mais instigante, acho eu.".* Isto é importante porque ajuda a prender a atenção do aprendiz e ele fica mais engajado a refletir sobre o programa.

Iconicidade

Os participantes apresentaram uma forte tendência a interpretar as relações entre os agentes de um modo físico, concreto. Chamamos isso de iconicidade. Quando perguntamos a respeito das relações, eles focaram no comportamento perceptível dos agentes durante a simulação e procuraram observar como os agentes entravam em contato "físico" com os outros. No entanto, nas relações perguntadas, não havia contato direto entre os agentes. O participante P6 nos forneceu uma forte evidência de como a iconicidade e a questão da abstração das representações computacionais podem causar conflitos no seu entendimento. P6 disse: "*alguma coisa caiu na árvore [...] alguma coisa caiu e eu não reparei? Eu só vi ela ficando laranja [...] eu não sei se caiu [Foi] tão rápido que eu não reparei, deixa eu ver a simulação de novo...*". Em outra circunstância, P6 tentava responder qual a relação entre o agente B (¯) e C (🏭), mas não conseguiu completar o raciocínio, ele disse: "*visualmente, nada acontece, deixa eu ver [...] mas ele [agente B] é um contador, você sabe, isso é o que está dito aqui [nas regras do agente], ele está contando alguma coisa*". Apesar de saber que o agente B (¯) funcionava como contador, P6 não conseguia explicar claramente qual a sua função.

Cinco dos seis participantes do primeiro ciclo comentaram algo sobre uma "chuva tóxica" ou "fim do mundo" no final da simulação. De fato, tudo que acontece é uma mudança na imagem de fundo. Mais uma vez, não há nenhuma referência ao final trágico imaginado pelos participantes, isso foi uma inferência

completamente relacionada ao domínio da aplicação. O participante P4 foi o único que não comentou sobre o 'fim do mundo', quando perguntamos sobre a relação entre B () e C (), ele respondeu: "*a relação entre B e C é que B conta enquanto C está checando o contador, quando B conta, C checa se B é igual a 2 então C faz algo, e quando contador é igual a 3 C vai fazer outra coisa*". P4 responde sem nenhuma informação relacionada ao domínio da simulação e fornece uma resposta correta baseada, apenas, em como as regras estão programadas.

A iconicidade é importante porque ressalta a necessidade de um signo, uma representação, que concretize as relações entre os agentes, independente do domínio. As relações podem ser encontradas no programa, mas é difícil porque estão espalhadas e não há, na interface do AS, um mecanismo que possibilite a visão regional. As explicações feitas pelos participantes sobre as relações que levaram a esta categoria foram baseadas no *applet*.

Função de linguagem dominante

As duas últimas categorias função e transição entre funções de linguagem estão ligadas às funções de linguagem definidas por Jakobson (1960): "expressiva", quando o foco é no emissor; "fática", quando é direcionado ao canal de comunicação; "conativa", quando é direcionado ao receptor; "metalinguística", quando o foco é na linguagem; "poética", quando o foco é na mensagem; e "referencial", quando o foco é no contexto de comunicação.

Os participantes P1, P5 e P6 usaram predominantemente a função referencial, enquanto que os participantes P2, P3 e P4 usaram mais a função metalinguística. Por exemplo, P5 estava tão focado no contexto da simulação que, ao responder sobre a relação entre B () e C (), disse: "*Eles estão se limitando, certo? Esse é um limite para não ficar cheio de fumaça aqui*". Mesmo depois de verificar as regras de comportamento dos agentes, ele concluiu "*certamente, o papel de B é limitar a emissão [da fumaça] do C*". A resposta não está errada, mas apenas a consequência foi explicada. É importante desenvolver a capacidade de diferenciar o que o programa realmente faz para que o efeito seja o esperado na simulação.

Por outro lado, P4 tem um discurso fortemente metalinguístico (focado na linguagem do AS). Ele afirmou que A e C () não tinham nenhuma relação após olhar as regras do agente A (ou) e do agente C () separadamente. Na

verdade, o agente D ( ) fazia a conexão entre eles. Com alguma atenção na simulação, era possível ver essa conexão, como os outros participantes fizeram. O agente D ( ) era criado pelo agente A (ou) e o agente A enviava uma mensagem para que o agente D ( ) se apagasse. Apesar disto, P4 guiou-se estritamente pelas especificações do código do programa (na linguagem visual do AS) e concluiu que ali não havia qualquer representação (explícita) da relação questionada.

Transição entre funções de linguagem

Foi possível perceber que não há uma relação sistemática entre os *soft* e *hard masters* e os participantes que optaram preferencialmente pela função referencial e metalinguística. Apesar de detectarmos uma função de linguagem predominante, observamos que os participantes transitavam de uma função de linguagem a outra durante suas explicações. É, contudo, difícil apresentar uma evidência deste resultado a partir de falas dos participantes, especialmente porque é o tipo resultado extraído de uma observação dinâmica, que se desdobra ao longo de todo o estudo.

Todos os participantes foram escolhidos porque eram habilitados a ministrar um curso para iniciantes usando o AS. O discurso de um professor é tipicamente repleto de transições entre as funções de linguagem, essa é uma habilidade que dá aos estudantes múltiplas perspectivas sobre um assunto que está sendo estudado. Esse pensamento despertou nosso interesse para um objetivo de *design* que ainda não estava claro. Era preciso reforçar os caminhos de conversa que facilitassem essas transições entre as funções de linguagem nas facetas do sistema.

Contribuição para a elaboração do Modelo PoliFacets

Em relação ao modelo PoliFacets, este estudo foi importante porque destacou a necessidade de facetas com o escopo maior do que um único agente e menor do que o projeto completo. Além disso, notamos que a ligação entre as facetas para facilitar as transições entre as funções de linguagem é um requisito essencial.

3.2.5. II *workshop* para professores

Objetivo

O objetivo desse estudo foi, após mostrar a professores a riqueza de informações disponíveis no PoliFacets-AS, explorar como eles poderiam compreendê-las e chegar até elas sozinhos. Notamos que os professores dos estudos anteriores haviam tido problemas para se apropriar do PoliFacets-AS e tirar proveito de todas as funcionalidades disponíveis. Depois de um ano de lançamento oficial do PoliFacets-AS, vários estudos e melhorias foram realizados, mas percebemos que o PoliFacets-AS ainda estava subutilizado.

Procedimento para coleta de dados e investigação

O segundo *workshop* para professores foi realizado em agosto de 2013, em dois sábados consecutivos com carga horária de 16h. Comparado ao I *Wokshop*, utilizamos uma abordagem diferente. A programação (Quadro 3.4) foi planejada para demonstrar mais o valor que vemos no (e tentamos comunicar através do) PoliFacets-AS com transições gradativas do nível de complexidade sem sobrecarregar os participantes.

Quadro 3.4 - Cronograma do segundo *workshop*

Dia - Turno	Programação
Dia 1 - Manhã	Boas Vindas e Apresentação do projeto SGD-Br O que é Raciocínio Computacional PoliFacets: Apresentando suas facetas AgentSheets: Interface e Movimentos Básicos
Dia 1 - Tarde	AgentSheets: Outras regras de programação Encerramento do dia
Dia 2 - Manhã	AgentSheets: Elaborando jogo ou simulação AgentSheets: Finalização dos jogos e envio para PoliFacets
Dia 2 - Tarde	Apresentação dos jogos criados durante o *workshop* Compartilhando experiências PoliFacets: Refletindo sobre as facetas Encerramento do w*orkshop*

Diante dos problemas relacionados à apropriação do PoliFacets-AS por parte dos professores, pensamos em uma nova estratégia de apresentação. Com essa finalidade, fizemos uso estendido das metáforas de exploração das perspectivas culturais no *design* de sistemas multiculturais de Salgado (2011). A Figura 3.12 apresenta as cinco metáforas que partem desde o isolamento cultural até a imersão cultural. O trabalho de Salgado (2011) apresenta uma ferramenta epistêmica que tem como objetivo ajudar os *designers* que estão preocupados em encorajar e apoiar experiências entre culturas distintas, tanto em tempo de *design*

quanto de avaliação. Embora não estejamos tratando de "cultura" no mesmo sentido que Salgado (2011) trata, vemos um paralelo interessante no qual estaríamos tentando "aproximar" uma cultura que desconhece programação de uma cultura que conhece.

Figura 3.12 - Progressão de acordo com as perspectivas culturais
Fonte: Salgado (2011)

Essa ideia nasceu no estudo das regras e narrativas de programas (subseção 3.2.3), mas foi reaproveitada nesse novo contexto. Naquela ocasião, observamos como os alunos criaram suas narrativas partindo da sua cultura nativa (sem contato com programação) para a cultura estrangeira (mundo da programação) e pensamos em gerar automaticamente textos em linguagem pseudonatural que fizessem a mediação entre as duas culturas. Um dos conceitos centrais utilizados por Salgado (2011) em sua proposta é o da "mediação cultural". Em nosso domínio, temos o professor como "mediador" entre o aprendiz e aquilo que ele aprende. Neste momento, com a ajuda da professora que ministrou o *workshop*, que atuou como agente mediador, nós guiamos os participantes na navegação pelo PoliFacets-AS, partindo da sua cultura nativa (programação no AS) para um cultura estrangeira, (PoliFacets-AS). O planejamento do segundo *workshop* de acordo com as metáforas está descrito no Quadro 3.5.

Quadro 3.5 - Planejamento de acordo com as metáforas

Metáfora	Programação detalhada
Observador à distância	"AgentSheetês" Apresentação do PoliFacets-AS Apresentação inicial das facetas Apresentação simples das facetas (dinâmica com perguntas) - Fish tank (Na Prática, Regras e *Tags*) - Apartamentos (Na prática e planilha)
Visitante guiado	Exploração projetos mais complexos (dinâmica com perguntas) - Reviver (Agente fantasma na planilha) - Desequilíbrio ambiental (Lógica global e relações entre agentes) *Upload* dos projetos atuais (devem conter os exemplos de comandos construídos durante o dia) Sugestão de consulta do PoliFacets-AS para observação dos exemplos e material didático Apresentação de jogo de exemplo com conceitos estudados
Estrangeiro com tradutor	*Upload* dos projetos finais e apresentação Provocar o uso do PoliFacets-AS além da faceta 'na prática': - Perguntar sobre comportamentos/relações interessantes, agentes empilhados/fantasmas - Perguntar sobre que modificações eles fariam para melhorar/completar seus jogos e como Apresentação final da mensagem do *design* das facetas

Neste momento, consideramos apenas as metáforas de mediação. Não serão considerados os dois extremos nos quais não há mediação. Em um extremo, a metáfora do 'viajante doméstico' refere-se à manutenção do usuário do sistema onde ele se sente em casa, no ambiente do AS, sem provocar o uso do Polifacets-AS, sistema que ele desconhece. No outro, o 'estrangeiro sem tradutor' provocaria o uso totalmente livre do PoliFacets-AS, sem nenhum tipo de mediação ou orientação ao usuário. Já na metáfora mediada do 'observador à distância', o objetivo é guiar o usuário pelo sistema, fornecendo muito apoio, seguindo passo a passo de modo que ele se sinta bem à vontade, mais 'perto de casa' do que do 'desconhecido'. Na metáfora do 'visitante guiado', o uso do PoliFacet-AS deveria ser ainda mediado por um guia que, no entanto, explica a cultura desconhecida em um nível médio entre o 'observador à distância' e o 'estrangeiro com tradutor'. Por último, no 'estrangeiro com tradutor', o conteúdo deveria ser parafraseado para uma linguagem mais simplificada. A inspiração nas metáforas para a elaboração da programação do *workshop* não foi comunicada aos participantes.

Observamos os participantes durante as dezesseis horas de *workshop*, mas os dados coletados estão concentrados em três momentos: um questionário anterior ao *workshop* (Apêndice - Questionário A do 'II *workshop* para professores'), um questionário ao final (Apêndice - Questionário B do 'II *workshop* para professores') e a apresentação dos projetos dos participantes.

Perfil dos participantes

Membros de quatro escolas foram convidados. As escolas A, B e C eram parceiras do projeto. A escola identificada como D é um grupo especial formado por uma aluna de doutorado em educação na PUC-Rio que também ministra aulas de língua portuguesa para o ensino fundamental em uma escola pública do Rio de Janeiro e uma estudante de pedagogia da PUC-Rio que se interessou pelo projeto. O participante P2 também esteve presente no primeiro *workshop*. A área de atuação de cada participante pode ser consultada no Quadro 3.6.

Quadro 3.6 - Participantes do segundo *workshop*

Escola	Participante	Atuação/disciplinas
A	P1	Ensino médio / Matemática
B	P2	Professora de turma de programação com o AgentSheets
	P3	Ensino fundamental / Matemática
C	P4	Ensino fundamental / Matemática e tecnologia
	P5	Ensino médio / Communication Arts, Computer Literacy e Digital Arts
D	P6	Ensino fundamental / Língua portuguesa / Aluna de doutorado em educação
	P7	Aluna de pedagogia

Principais resultados

Neste estudo os professores reconheceram a utilidade das diferentes representações e perceberam, através do exemplo presenciado, algumas formas de utilizar os recursos disponíveis. As análises foram realizadas considerando como foco o PoliFacets-AS e as seguintes categorias foram identificadas: fator motivacional; reconhecimento do valor do PoliFacets-AS e suas aplicações; novo formato de apresentação do conteúdo; e a consciência sobre o que foi implementado no jogo.

Fator motivacional

No questionário anterior ao *workshop*, entre os sete participantes, apenas dois responderam que já tinham ouvido falar de raciocínio computacional. P2 foi um deles, cujo conhecimento do assunto era esperado uma vez que era membro ativo no projeto e participou do primeiro *workshop*. P3, que também já tinha noções a respeito, era da mesma escola que P2, já havia assistido a algumas aulas de P2 e, apesar de ser professora de matemática, tem adicionalmente uma formação de mestrado na área de computação. Ao pedir que os participantes comentassem o que entendiam por raciocínio computacional, obtivemos respostas interessantes. P6 respondeu: "*A capacidade de lidar com a forma de pensar*

hipertextual. Isso não é novo. O pensamento do ser humano é hipertextual e agora há uma tecnologia que permite e promove – talvez o termo melhor seja concretize – essa forma de pensar". P1 disse: "*Acredito ser uma forma de produzir e repassar, por meios computacionais, ações que queremos ver realizadas. Uma forma de elaborar a realidade*".

Como parte do fator motivacional, a capacidade de atrair os participantes a iniciarem e continuarem no projeto é importante aspecto do *workshop* e das ferramentas que nele são apresentadas. P1 disse: "*Tenho interesse por entender que [o ensino de raciocínio computacional] estimula formas de pensar usando a criatividade*". P6 explica porque tem interesse no assunto dizendo: "*Estou começando (começando mesmo) a estudar jogos digitais e tenho muito interesse em tudo o que diz respeito à construção do conhecimento através das novas mídias*".

Entre os comentários finais do questionário, P6 declara: "*Estou encantada com o que aprendi. Minha cabeça não para de pensar em possibilidades*". Todos são convidados a se tornarem parceiros na pesquisa e P1 sugere: "*[...] que tal dar continuidade ao workshop, à distância? Seria um aprofundamento "parcial" durante um período curto (1 ou 2 meses). O que acham?"*. É interessante que esse pedido já tinha entrado em pauta depois que turmas de programação com o AS terminaram, mas os alunos ainda queriam continuar construindo seus jogos.

<u>*Reconhecimento do valor do PoliFacets-AS e suas aplicações*</u>

Mais uma vez, os participantes compreenderam o valor do PoliFacets-AS e suas possibilidades de aplicação. P3 definiu o PoliFacets-AS como: "*Ferramenta que orienta, organiza e ajuda a compreender o jogo*". Mais do que isso, os participantes começaram a perceber o PoliFacets-AS como um mecanismo de reflexão para melhoria dos projetos, como afirma P6: "*Uma plataforma para visualizar todos os procedimentos desenvolvidos na criação de um jogo. É também um recurso de autoavaliação e de estudo constante*". P5 ratifica essa ideia: "*Acho o [PoliFacets-AS] superinteressante pois nele [você] pode testar e jogar os jogos criados além de analisar e ver as regras criadas em cada agente e suas conexões. Com isso, podemos modificar além de melhorar o [que] já produzimos*". Durante a exposição da faceta 'conexões', P6 fez uma pergunta para saber se esses recursos então sendo usados em sala de aula e complementou: "*eu*

estou vendo uma espécie de mapa mental onde conseguimos ver as relações entre os termos. Eu fico imaginando uma série de coisas, toda essa construção, ver isso, fazer aquilo etc. e tal, eu já estou pensando na estrutura sintática [...], quando você vai construir um texto e vai organizar o sujeito, o predicado, [...] você tem uma construção".

Quanto ao uso do PoliFacets-AS em classe, os participantes reafirmaram essa oportunidade. P4 elogiou dizendo: "*[...] Vai ser a parte principal do curso de programação que estou ensinando*". P1 lembrou que a ferramenta ajuda na busca pelo conhecimento, afirmando: "*Acredito que independente do objetivo do professor, o [PoliFacets-AS] por si só, estimula o aprendizado e complementa qualquer finalidade de uso com o desenvolvimento da persistência e da busca pelo conhecimento*". Como professora de língua portuguesa, P6 pensou no potencial da ferramenta com relação à produção textual, afirmando: "*[O PoliFacets-AS] facilita releitura e reavaliação do que se produziu, com possibilidade de percepção direta dos erros e acertos, a plataforma se torna uma ferramenta concreta para a produção textual, avaliação das partes componentes [...]*".

Novo formato de apresentação do conteúdo

A professora que ministrou o *workshop* tinha experiência anterior no ensino de AS em turmas de ensino fundamental e usou o PoliFacets-AS mesmo antes de ser integrante da equipe de pesquisa do projeto. Nós, equipe técnica do projeto, e essa professora trabalhamos em conjunto para que a nova abordagem de navegação guiada no PoliFacets-AS fosse considerada e seguida. Vale ressaltar que as habilidades de expressão e de didática da professora estão envolvidas nos comentários a seguir, mas também podemos relacioná-los ao modo como o PoliFacets-AS, o AS e os conceitos foram abordados. P6 revelou entre os comentários gerais sobre o *workshop*: "*Foi muito didática a apresentação. Conheci uma nova possibilidade de trabalho*". P7 destacou que: "*A criação de jogos foi muito divertida. Os apresentadores ensinaram claramente*". Como um dos pontos positivos, P1 comentou sobre "*a forma cadenciada com que o conteúdo foi compartilhado*". Esta "forma cadenciada" de apresentação do conteúdo, percebida pelo participante, parece ser um indício de que o uso das metáforas para mediarem a inserção gradativa dos usuários na cultura do

Polifacets-AS surtiu o efeito desejado apesar de o uso dessas metáforas não ter sido explicitado aos participantes.

Consciência sobre o que foi implementado no jogo

Um ponto de destaque no discurso dos participantes, e, de certa forma, o mais importante de todos os resultados deste estudo sobre o desenvolvimento da capacidade de leigos se expressarem via *software*, foi a consciência que os participantes ganharam sobre os jogos que criaram. Havia, para cada jogo, um planejamento, não se tratava do resultado de tentativa e erro. Quando os participantes realizaram a apresentação dos projetos deles, várias perguntas foram feitas para provocar explicações mais elaboradas. P7 criou o jogo da Figura 3.13 e explicou como fez a finalização do jogo. Ele disse: "*fiz a mudança no pano de fundo. Na verdade, tem uma barreira invisível aqui embaixo e quando a barreira vê a comida, o pano de fundo muda, é quando encosta, mas o comando que se usa é ver*". A resposta dada é bastante sofisticada tendo-se em conta que o comando para mudar o pano de fundo das planilhas não foi ensinado durante o *workshop*. No evento, apenas o comando de troca de planilhas para criação de novas fases foi abordado e, por isso a professora se interessou por perguntar como P7 havia feito, já que era a mesma planilha.

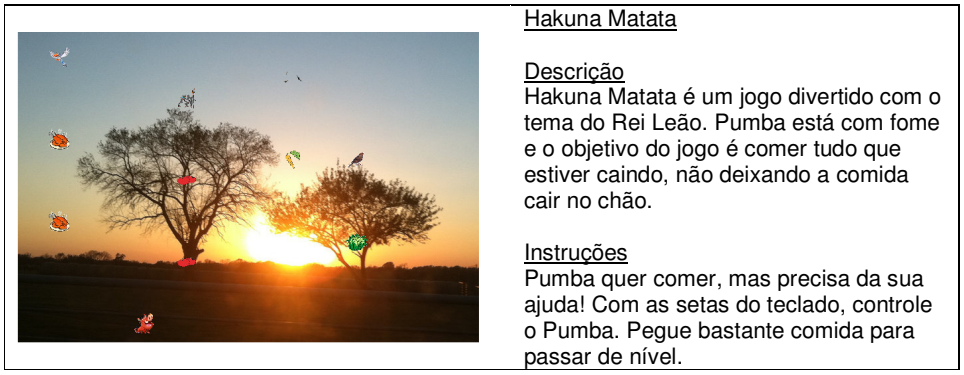


Figura 3.13 - Projeto do participante P7

O participante P5 criou o jogo com um tesouro escondido por baixo do "terreno" que cobria toda a planilha (Figura 3.14). Quando questionado sobre a posição exata do tesouro, primeiramente respondeu: "*eu lembro onde está*". Quando indagado sobre o que faria caso esquecesse a posição, rapidamente afirmou: "*então eu ia ter que jogar*". Um dos professores antigos do projeto (convidado para participar de um momento do *workshop* no qual professores experientes compartilharam experiências com os professores em formação)

sussurrou logo em seguida "*acho que o PoliFacets ajuda [a descobrir onde está o tesouro]*". E isso de fato ocorreu pois, enquanto P5 estava navegando no PoliFacets-AS, foi possível observar que outro participante já tinha encontrado a localização do tesouro na grade da faceta 'planilhas'. Depois do apoio dos colegas, P5 percebeu, então, como poderia apontar a localização exata do agente na grade.

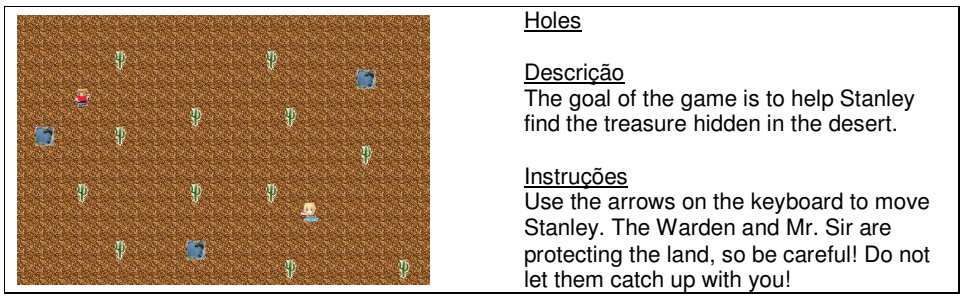


Figura 3.14 - Jogo do Participante P5

O participante P6 fez um jogo sobre acentuação. Sua ideia era tratar palavras proparoxítonas e paroxítonas e oxítonas que deviam cair cada uma em um balde específico. P6 não se prendeu a representação visual das palavras, uma vez que esse plano era de difícil concretização através da ferramenta de desenho do AS. Assim, alternativamente, P6 criou um padrão de quadradinhos coloridos, temporariamente, para expressar sua ideia (Figura 3.15). Enquanto apresentava o projeto, mostrou sua consciência sobre os problemas já reconhecidos. P6 disse: "*já descobri inúmeros erros, esse azul, por exemplo, não está explodindo em cima do bonequinho. Eu percebi que eu fiz a regra certinho com o empilhado pro verdinho. Mas o azulzinho eu percebi que fiz errado, porque usei o se vir [...], quando eu estava lendo as regras eu percebi que tinha esquecido*".

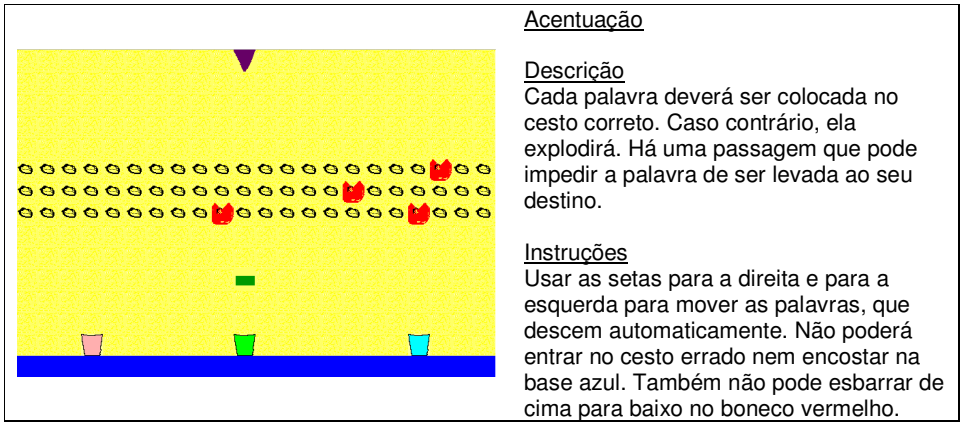


Figura 3.15 - Jogo do Participante P6

P6 também falou do problema de visualização da planilha: "*eu percebi também que a minha [planilha] tá errada porque tem um quadradinho lá embaixo, um que sobrou de quando eu não tinha feito a barreira*". A professora pediu para o P6 mostrar onde estava o agente perdido e ela o fez através da tela inicial das facetas, na qual a miniatura do jogo estava visível, contendo propositalmente os problemas da planilha. Veja na Figura 3.16 a tela que apresenta as facetas com uma seta vermelha indicando o agente2 perdido que está na última posição abaixo e à direita da planilha.

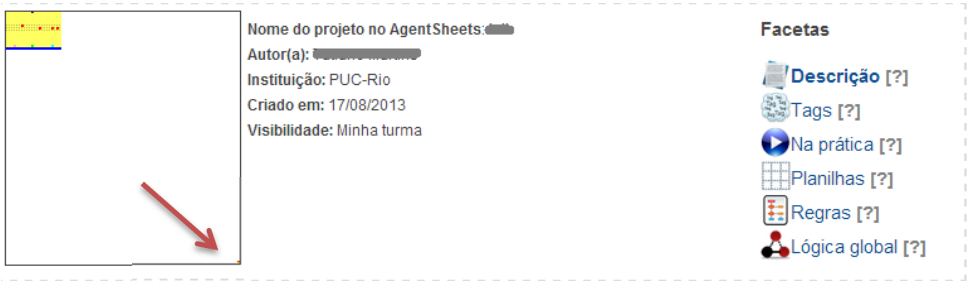


Figura 3.16 - Cabeçalho das facetas do projeto de P6

Contribuição para a elaboração do Modelo PoliFacets

Em relação ao modelo PoliFacets, este estudo foi importante porque os resultados mostram como as facetas são utilizadas com sucesso durante o processo de ensino e aprendizado de programação, fomentando novas possibilidades de uso e expressão através de *software*. Os participantes perceberam como as facetas ajudavam a perceber situações que eram difíceis de notar usando apenas o AS. A paráfrase dos conteúdos do AS via PoliFacets-AS ajudou a aumentar a consciência dos participantes sobre os programas que eles estavam construindo.

3.2.6. Estratégias de comunicação

Objetivo

O objetivo deste estudo foi entender como seria a aplicação da abordagem de mediação utilizada no segundo *workshop* se o processo de mediação fosse introduzido no PoliFacets-AS diretamente. No segundo *workshop*, a professora agiu como mediadora, neste estudo usamos uma ferramenta computacional para simular a mediação, como se a professora estivesse falando por meio do PoliFacets-AS.

Procedimento para coleta de dados e investigação

Este estudo consistiu na avaliação do protótipo de um *design* guiado no PoliFacets-AS usando as metáforas de mediação cultural aplicadas no estudo anterior. A ferramenta usada para simular a ideia de mediação foi o SideTalk (Monteiro e de Souza, 2012) que é um recurso de comunicação entre pessoas por meio de sites disponíveis na internet. Ele funciona como uma extensão para o Firefox, tornando-se disponível através de uma barra lateral no navegador que age como uma conversa paralela sobre páginas na *Web*, construída a partir de um *script* previamente gravado de interações nas páginas. O SideTalk já foi utilizado com outros propósitos, como em casos de acessibilidade, auxiliando no ensino e explorando a autoexpressão (Alves *et al.*, 2013, Monteiro e de Souza, 2012 e Monteiro *et al.*, 2013).

Neste estudo, utilizamos o SideTalk como uma ferramenta para criar um protótipo de *design* da interação no PoliFacets-AS. Um exemplo de como é a visualização do SideTalk com o PoliFacets pode ser observado na Figura 3.17. Os *scripts* usados no estudo estão disponíveis no Apêndice - *Scripts* do estudo 'Recepção das facetas'. O uso do SideTalk foi importante para facilitar o processo de criação do protótipo de *design* da interação. Ele foi um mecanismo necessário para avaliar a viabilidade da mediação simulada no PoliFacets-AS.

Figura 3.17 - SideTalk durante a navegação no PoliFacets

Como preparação para a avaliação com os participantes que representariam os usuários finais, foram criadas três versões de conversas com o SideTalk, cada uma correspondendo a um nível de mediação diferente segundo as metáforas que

guiaram o segundo *workshop*. Depois disso, contamos com o apoio da professora do segundo *workshop* para atestar a qualidade da mediação criada com o SideTalk. Após exibirmos os *scripts* e diálogos de mediação, a professora constatou que o protótipo realmente demonstrava o que havia acontecido, fazendo-a se sentir representada através dos diálogos.

O procedimento do estudo foi dividido em quatro etapas: uma explicação sobre como o SideTalk funcionava, na qual era explicitado que ele não fazia parte da avaliação, sendo apenas uma ferramenta para auxiliar a demonstração da ideia; a leitura do enunciado da tarefa que consistia do acompanhamento do *script* proposto, e da leitura das mensagens na barra lateral do navegador com atenção durante os três níveis propostos; a execução de três *scripts*, em que cada um representava um nível de mediação (nenhuma explicação sobre essa diferença foi explicitada na condução do estudo); e por último, a entrevista para entender como os participantes avaliaram a mediação.

O foco da entrevista era obter uma avaliação sobre a navegação. Por isso, fizemos perguntas abertas sobre o que os participantes acharam da experiência. A caracterização dos diferentes níveis de mediação foi identificada pela maioria dos participantes.

Perfil dos participantes

Convidamos dois grupos: participantes do segundo *workshop* que tinham passado pelo processo de mediação com a professora; e pessoas que não participaram do *workshop*, mas conheciam superficialmente o projeto SGD-Br e PoliFacets-AS (não conheciam detalhes das facetas). A divisão dos participantes nos grupos pode ser observada no Quadro 3.7.

Quadro 3.7 - Participantes

Grupo	Participante
Participantes do II w*orkshop* para professores	P1
	P2
	P3
Apenas conhecem a superficialmente o PoliFacets-AS	P4
	P5
	P6
	P7

Principais resultados

Este estudo foi importante para começarmos a perceber que tipo de melhoria devemos fazer no PoliFacets-AS para que os professores consigam se apropriar

melhor do sistema. O estudo mostrou que pode ser interessante explorar diferentes níveis de mediação durante a apropriação do PoliFacets-AS pelos usuários finais.

Os dados coletados foram classificados em três categorias: a identificação dos diferentes níveis de mediação; avaliação geral dos participantes sobre a estratégia de comunicação; e a estratégia de comunicação preferida.

Identificação dos diferentes níveis de mediação

Quanto à identificação dos diferentes níveis de mediação, os participantes conseguiram perceber as características predominantes dos diálogos apresentados. P2 disse: "*elas [as conversas] vão avançando no nível do [PoliFacets-AS], não é?! A parte III era mais complexa que a parte I, a parte I era mais básica*". Já P1 descreveu o primeiro caso: "*O primeiro [script] procura te explicar mais como funciona o [PoliFacets-AS], [...] como você pode se movimentar ali dentro*". O participante P5 percebeu o aumento no grau de dificuldade na mediação, ele disse: "*o primeiro tem uma linguagem bem simples, o segundo não é tão simples, mas também vai fazendo um passo a passo*". P6 reforçou a mesma noção: "*a [parte] dois e a três achei complexo, [a parte I] a apresentação dos conceitos é bem tranquila*".

O participante P4 concluiu que: "*a parte I seria uma visão topdown da estrutura da aplicação, não da forma da gente pensar, meio que apresenta o [PoliFacets-AS] seguindo a estrutura da interface. A segunda [parte] mostra um exemplo de onde você obtém uma informação ou outra. A terceira [parte] misturou um pouco as duas."*. P4 mostrou dúvida em relação à última parte, dizendo: "*na parte III, notei que estava apresentando de novo as coisas, eu estava meio que revendo o que já tinha visto de uma forma ou de outra."*. Isto é verdade, o que mudou foi apenas a forma de apresentação do conteúdo que tentava demonstrar o uso da metáfora 'estrangeiro com tradução'.

P7 caracterizou as diferenças entre o segundo e terceiro *scripts*: "*é diferente porque no segundo [script] apresenta [o sistema], mas o terceiro [script] é mais como você pode analisar"*. P7 reafirmou o incentivo a descoberta e reflexão: *"na terceira [parte] mostra mais aquelas abas [facetas], vai de um em um, conexões, você até acaba interagindo mais, [o diálogo] manda você clicar [...]"*.

Avaliação geral dos participantes sobre a estratégia de comunicação

Os participantes acharam a possibilidade de mediação interessante. P1 comentou que o primeiro *script* foi muito cansativo e aconselhou que esse tipo de informação devesse ser adicional. P1 disse: "*essa parte mais detalhada e cansativa, eu acho [que] isso devia ser uma informação extra, caso eu queira saber mais [...]*".

Como um dos participantes do segundo *workshop*, P2 comentou: "*eu achei interessante [ver] onde cada coisa está e isso vai me ajudar melhor para quando eu for fazer o jogo [...] deu uma boa revisada [...], não sabia que os vídeos existiam, acho que não procurei direito*". Mesmo tendo participado do *workshop*, P2 não se lembrava do momento em que os vídeos foram mostrados e ficou feliz em saber que eles existiam, já que valoriza muito esse tipo de mídia no ensino. P3 concordou que era uma possibilidade didática interessante, mas manifestou preocupação com a ideia do ensino passo a passo e nos mínimos detalhes. P3 disse: "*O [PoliFacets-AS] é bem didático, acho que é uma dificuldade das pessoas hoje em dia de querer tudo muito mastigadinho. É o professor que te pega pela mão e vai, lê isso, te obriga a responder. Não acho que é um problema no [PoliFacets-AS]*".

O participante P5 começou a imaginar que tipo de público seria mais adequado para cada nível de mediação. Como uma opinião, P5 relatou: "*achei a primeira [parte] mais didaticazinha, mais fácil, parece até que estava falando com criança. Já a última [parte], é mais direta [...] uma criança já não ia conseguir pensar tão rápido. Mas, para gente que é adulto, que tem mania de mexer em tudo, coisas diretas são melhores*".

Dois participantes defenderam uma maior liberdade de interação com o sistema e a possibilidade de solicitar ajuda quando fosse preciso. P5 disse: "*de repente era legal não ter o [SideTalk] o tempo todo conversando com você, poder navegar livremente e numa área que você não está entendendo, tem a possibilidade de ter o [SideTalk] para te ajudar no tour [...] se o sistema tivesse preparado pra isso, seria bem melhor*". Apesar da inclusão do SideTalk no discurso, P5 estava claramente se referindo à estratégia de mediação no sistema. P7 também reafirma a necessidade de liberdade do aprendiz, P7 relatou que: "*acho interessante sim, mas às vezes a pessoa quer ficar mais livre, brincar*

depois, voltar [...] acho a ideia de ir acompanhando é boa, mas é preciso dar mais liberdade e feedback".

O participante P3 participou do segundo *workshop*, mas ainda não tinha experiência ministrando cursos de programação com o AS. Não sabia do problema atual de dificuldade de apropriação do PoliFacets-AS por parte dos professores. Em sua avaliação geral, P3 comentou: "*eu gostei muito disso aqui, engraçado porque depois do workshop a gente ficou pensando que estava completo e não tinha mais nada pra mexer nisso, e a gente vê que essa ferramenta ajuda bastante, o roteirinho foi ótimo*".

Estratégia de comunicação preferida

Quanto à estratégia preferencial, P2 disse: "*eu sou muito independente, eu aprendo fuxicando mesmo*". Várias vezes P2 observou mais do que o diálogo e ficou explorando outros detalhes na página. P2 revela sua preferência: "*eu iria mais para a parte III, pelo meu jeito de ser mesmo, eu quase nunca leio as instruções*". P5 compartilha a mesma opinião: "*eu não sou uma pessoa que usa manual, eu sou daquelas que tenta, tenta, procura no Google, se eu não achar, vou pro manual. Pra mim é muito difícil, só ir clicando, é legal que dá um guia, mas tem coisas aqui que se eu não lesse na lateral eu não ia achar, por exemplo, essas histórias das facetas, não tem um menu com as facetas[...]*". Mas, P5 reconhece que o PoliFacets-AS atualmente tem problemas , como, por exemplo, a comunicabilidade baixa sobre o funcionamento de cada faceta.

Mesmo reconhecendo o valor da mediação com muito apoio, P7 também segue a linha de que precisa de liberdade na interação acompanhada da possibilidade de ativar mais mecanismos de ajuda. P7 afirma: "*se o sistema fosse dizer isso passo a passo, quero que tenha como desligar, tem hora que eu quero brincar. Depois, quero poder escolher quando eu quero a ajuda [...]*".

P3 comentou que a interação seria interessante com um esquema de mediação mais interativo. P3 revelou: "*eu acho que quando ele começa a responder, ele começa a interagir, deixa de ser só um leitor e passa a ter uma postura mais ativa. Isso faz você falar em voz alta, então você encontra erros [...]*". Já P4, apesar de ter caracterizado as facetas de forma adequada, não demonstrou preferências pessoai. P4 explica: "*não estou sabendo julgar se tem alguma forma mais importante*". O participante P6, por sua vez, mostrou sua

preferência pelo primeiro nível de mediação. P6 disse: "*a primeira [parte] achei bem tranquila, assim, boa de acompanhar, mas também de ser gostosa, de me sentir bem conduzido*".

Contribuição para a elaboração do Modelo PoliFacets

Em relação ao modelo PoliFacets, a principal contribuição deste estudo é a necessidade de criar ligações entre as facetas. Essas ligações ajudam o aprendiz a navegar e se apropriar do PoliFacets-AS, pois criam caminhos que auxiliam a comunicação da mensagem por parte do *designer*.

Ainda não é possível concluir que tipo de mediação seria mais adequado, pois há muitas variáveis que precisam ser melhor analisadas, entre preferências pessoais e de ensino. Por essa razão, a abordagem da mediação ainda não será integrada ao modelo. Não queremos impor uma metodologia de ensino, mas apresentar as ferramentas e oportunidades de exploração disponíveis.

3.3. Elaboração teórica do modelo

Esta é a Fase 4 da pesquisa desta tese. O modelo PoliFacets foi elaborado com base na fundamentação teórica (seção 3.1) e na fundamentação empírica (seção 3.2). Cada um dos estudos empíricos relatados anteriormente colaborou diretamente para a construção de sucessivas versões da instância PoliFacets-AS e indiretamente para a construção do modelo.

A elaboração teórica do modelo corresponde a uma construção progressiva de uma estrutura conceitual para nortear o *design* de conteúdos de documentação ativa destinada a apoiar o ensino e aprendizado de programação. A Tabela 3.4 apresenta um resumo sobre os resultados dos estudos empíricos que formaram parte do aprendizado necessário para a abstração do modelo.

Tabela 3.4 - Estudos e resumo da contribuição para a construção do modelo

Estudos	Contribuição para a construção do modelo
Estudo inicial	As avaliações e melhorias nos conteúdos das facetas do PoliFacets-AS viabilizaram os estudos empíricos posteriores e a elaboração teórica do modelo.
I workshop para professores	A descoberta de dificuldades no entendimento dos professores também serviu para percebermos a necessidade da criação de novas facetas que preenchessem as lacunas no processo de compreensão dos programas.
Regras e narrativas de programas	Possíveis desdobramentos com relação à abrangência do escopo das verbalizações da faceta 'regras' e novos mecanismos de verbalização considerando as perspectivas do usuário e do sistema foram considerados.
Expressão da conexão entre os agentes	A necessidade de facetas com o escopo maior do que um único agente e menor do que o projeto completo foi identificada A ligação entre as facetas é um requisito essencial, nesse caso entre 'Conexões' e 'Regras'.
II workshop para professores	Os participantes perceberam como as facetas ajudavam a perceber situações que eram difíceis de notar usando apenas o AS. Parafrasear os conteúdos do AS ajudou a aumentar a consciência dos participantes sobre os programas que eles estavam construindo, além de fomentar novas possibilidades de uso e expressão através de *software*.
Estratégias de comunicação	A necessidade de criar ligações entre as facetas foi identificada.

Os dois primeiros estudos foram como uma grande apresentação da ideia do sistema de documentação com múltiplas representações dos programas. Eles foram importantes para entendermos as dificuldades dos aprendizes e percebermos as possibilidades de uso do PoliFacets-AS no ensino. Logo concluímos que criar extensões do AS em forma de facetas para adicionar significados ou reexplicar conteúdos expressados de modo diferente era promissor no contexto de ensino de programação como forma de autoexpressão. No plano teórico, essas questões estão em linha com as taxonomias de extensões e customizações propostas pela Engenharia Semiótica. As melhorias de conteúdos nas facetas estavam relacionadas ao modo como apresentávamos as informações e, nesse caso, as classes de signos que compõem a ontologia da Engenharia Semiótica apoiam a abstração teórica do modelo. A escolha do tipo de signo que deve comunicar a mensagem do *designer* através do sistema deve ser considerada durante o *design* da faceta.

No estudo das regras e narrativas de programas, ao observarmos as verbalizações dos alunos, percebemos que seria útil criarmos alternativas de textos. Por exemplo, uma alternativa seria gerar um texto com foco em um conjunto de elementos do programa, envolvendo um agente principal e os agentes que se relacionam com ele; ou ainda, gerar um texto com foco em todos os

elementos do programa, que representaria um texto completo com todos os agentes existentes. Textos mais elaborados exigem novos algoritmos que não apenas façam reformulações nos dados disponíveis, mas com alterações da dimensão semântica, que correspondem a reescrita da perspectiva do sistema de acordo com as taxonomias descritas anteriormente.

O estudo das conexões entre os agentes reforçou a necessidade de explorar representações em diferentes escopos dos programas, considerando um ou mais agentes conectados. Além disso, aprendemos que os usuários do PoliFacets-AS podem ter um discurso predominantemente metalinguístico, referencial ou outro, e que a transição entre as funções de linguagem presentes no discurso precisa ser apoiada. No plano teórico, as funções de linguagem da comunicação apoiam a reflexão sobre as ligações que o modelo deve oferecer para facilitar as transições entre os diferentes discursos. As variações ajudam os aprendizes a parafrasearem o conhecimento, explicando os mesmos conceitos com focos diferentes de acordo com suas intenções.

A nova estratégia de comunicação usada no segundo *workshop,* e depois no estudo específico, reforçou a importância de criar ligações entre as facetas. Outro ponto relevante do segundo *workshop* foi a consciência critica dos professores sobre seus próprios jogos. Com abordagem diferenciada que apresentou o PoliFacets-AS logo no início das atividades, como um mecanismo de reflexão sobre os projetos, os participantes já iniciaram a criação dos jogos com possíveis problemas em mente. Ao apresentar os projetos, eles demonstraram o conhecimento dos seus problemas e suas limitações. Entenderam que precisavam ajustar os projetos para conseguir construí-los dentro do tempo estabelecido no *workshop* e com os recursos que tinham aprendido. Mesmo assim, tinham noção do que era preciso para modificar o comportamento dos agentes e realizar determinadas ações para melhorar os projetos.

3.4. Componentes estruturais e suas relações

O modelo proposto nesta tese tem como objetivo o *design* da metacomunicação de documentos ativos para apoiar o processo de ensino e aprendizado de programação. O usuário do modelo é o *designer* do documento

ativo. Os usuários finais do documento ativo são os alunos e professores envolvidos no processo de ensino e aprendizado de programação.

A Figura 3.18 mostra o modelo de modo mais abrangente e como ele está situado no processo de *design*. As entradas são os programas e sua ferramenta de construção, assim como suas representações e a ontologia de domínio. Essas entradas passam por um analisador que gera os signos de programa e programação necessários para a construção de facetas e ligações. Depois que as facetas e ligações são modeladas de acordo com a ontologia de metacomunicação, ocorre o projeto e desenvolvimento de *software* que resultará em um sistema de documentação ativa. Os componentes em destaque na cor laranja são o foco desta tese e são detalhados nesta seção.

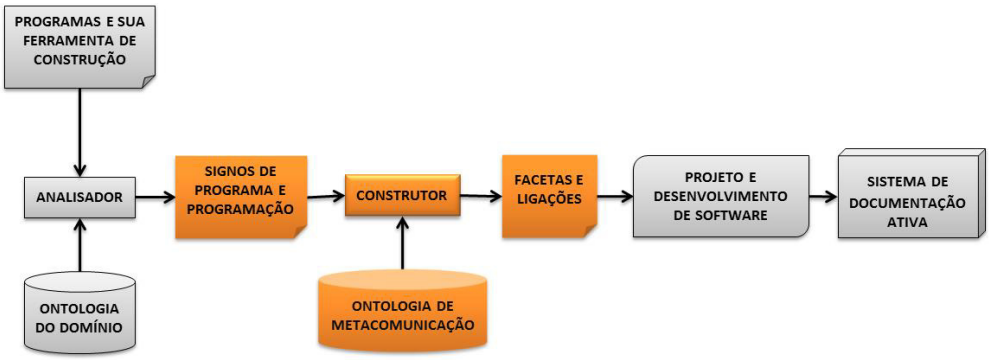


Figura 3.18 - Modelo PoliFacets no contexto de *design* da tecnologia

Os programas e sua ferramenta de construção formam o sistema de significação de entrada. Conceitualmente, a entrada é formada por um código ou um conjunto de códigos sobrepostos. Pode ser formada, por exemplo, por um código fonte do programa ou uma representação que passa por um analisador para que as informações sejam organizadas do ponto de vista léxico, sintático e semântico.

A ontologia do domínio é extraída de dados da aplicação sobre a qual o documento ativo será criado no domínio de ensino e aprendizado de programação. Ela representa tudo o que podemos conhecer sobre a aplicação e que será utilizado pelo analisador em conjunto com os programas e sua ferramenta de construção, com a finalidade de ensinar programação.

O analisador interpreta as informações a partir dos programas, da ferramenta de construção e da ontologia do domínio. Há casos em que as representações da ferramenta de construção de programas destinada ao ensino de programação não

destacam informações importantes. A análise e interpretação dessas informações permite uma posterior explicitação dos conceitos escondidos através da criação de novas facetas. Por exemplo, vamos supor que a ontologia de domínio indique a necessidade de ensinar a fatorar códigos. Nesse caso, o analisador deve ser capaz de apontar que trechos dos programas podem ser fatorados. Ele deve conter elementos da linguagem de programação e do modelo de tarefas embutido no ambiente de construção dos programas, sendo ou não um ambiente voltado para o ensino de programação.

3.5. Ontologia de metacomunicação

Nesta subseção vamos apresentar a construção da ontologia de metacomunicação do modelo de *design* proposto nesta tese. Esta ontologia foi dividida em três perspectivas: do usuário, do sistema e da interação. As duas primeiras são baseadas, principalmente, na taxonomia da Engenharia Semiótica para aplicações customizáveis e extensíveis. A perspectiva da interação foi incluída no modelo após evidências coletadas durante a execução de estudos empíricos realizados com o PoliFacets-AS somados aos aspectos teóricos da classificação dos signos.

A ontologia de metacomunicação é a parte principal do modelo. Ela é essencial para definir categorias de facetas de programas que podem existir no *design* de uma documentação ativa voltada para o processo de ensino e aprendizado de programação visando a autoexpressão através de *software*.

3.5.1. Perspectivas do usuário e do sistema

Parte da ontologia de metacomunicação composta pelas perspectivas do usuário e do sistema é resultado da síntese das combinações entre as mudanças previstas na taxonomia de customizações e extensões da Engenharia Semiótica apresentada anteriormente. Não existe uma correspondência direta entre as dimensões das perspectivas do usuário e do sistema, mas algumas relações podem ser destacadas. As alterações na expressão se refletem em mudanças léxicas e sintáticas, enquanto alterações no conteúdo representam mudanças na dimensão semântica e, por fim, alterações na intenção também espelham mudanças

semânticas. A complexidade de realizar esse mapeamento cruzado é um dos motivos que reforça a dificuldade de se criar aplicações extensíveis com qualidade (de Souza, 2005).

O objetivo original dessas taxonomias descritas na seção 3.1 é orientar o *design* de customização e de extensão de aplicações. Entretanto, realizamos uma apropriação das taxonomias para um novo contexto, que é o de apoiar ao *design* de uma documentação ativa com a finalidade de explorar significados presentes nos programas.

A Tabela 3.5 é uma síntese das combinações de taxonomias com sete diferentes tipos de extensões em ordem crescente de complexidade. Estamos lidando com extensões de um ponto de vista diferente, uma vez que não se trata da customização ou extensão de uma aplicação pelo usuário final para seu próprio uso. Um sistema resultante da aplicação do modelo age como mediador entre o *designer* principal da aplicação considerada e seus usuários finais. Queremos utilizar o conteúdo que a aplicação disponibiliza para gerar documentação. A ideia é apresentar novas representações do que foi construído na aplicação. Essas novas representações são desenvolvidas com base nas mudanças definidas nas perspectivas do usuário e do sistema. Ou seja, é uma documentação sobre o produto construído pelo usuário no ambiente estendido.

Tabela 3.5 - Matriz com as perspectivas do usuário e do sistema

Para o usuário	**Para o sistema**	**Tipo**
Ressignificação (*intenção*)		**1**
Paráfrase (*expressão e intenção*)	Reformulação (*léxico e sintático*)	**2**
	Reescrita (*léxico, sintático e semântico*)	**3**
Figuras de linguagem (*conteúdo e intenção*)	Reformulação (*léxico e sintático*)	**4**
	Reescrita (*léxico, sintático e semântico*)	**5**
Expansão linguística (*expressão, conteúdo e intenção*)	Reformulação (*léxico e sintático*)	**6**
	Reescrita (*léxico, sintático e semântico*)	**7**

Todas as combinações foram realizadas, exceto quando a mudança é apenas na intenção. Isto acontece porque o que caracteriza a mudança somente na intenção do usuário é a inexistência de modificações no sistema. A partir do momento em que ocorrem mudanças léxicas, sintáticas ou semânticas, também ocorrem mudanças na expressão ou no conteúdo.

Vale lembrar que quando falamos de mudanças ou alterações, pode ser até mesmo apenas uma redução ou ampliação em qualquer uma das dimensões. Estas mudanças são pensadas pelos *designers* para provocar efeitos sobre a experiência dos usuários finais.

1. A ontologia de metacomunicação de acordo com as perspectivas do usuário e do sistema é descrita nas sete subdivisões a seguir: **Ressignificação**. *Para o usuário: intenção.* Esse caso inclui apenas modificações na intenção do usuário. É quando o usuário atribui novos significados ao sistema, ou seja, quando um programa que serve para determinado objetivo passa a ser utilizado para outro fim, sem mudanças no programa. Exemplo: A linguagem Lua oferece a possibilidade de criação de bibliotecas que são chamadas de 'rocks'. Cada pacote do tipo '*rock*' é formado por uma coleção de arquivos e um deles deve ter a extensão '*rockspec*' que contém a especificação do pacote. Isto é, o *rockspec* é um arquivo de definições com a especificação de documentação e requisitos para a instalação do pacote. O objetivo dele é fornecer o detalhamento das especificações técnicas para o funcionamento do pacote. Por outro lado, o *rockspec* pode ser utilizado como *template* na criação de outros pacotes, funcionando como uma espécie de documentação baseada em exemplos. Esta é uma forma de usar um programa que tem um determinado fim com outro objetivo.

2. Paráfrase - Reformulação. *Para o usuário: expressão e intenção. Para o sistema: léxico e/ou sintático.* Este caso inclui novas representações em que há mudança na expressão ou forma de comunicação da nova intenção. Para o sistema, deve acontecer uma alteração no vocabulário e/ou na gramática da linguagem. Exemplo: em linguagem Lua, a principal estrutura de dados é a tabela. Para criar uma tabela com índices numéricos podemos deixar os índices implícitos, como "`local tabela = {'a', 'b', 'c'}`", onde a, b e c são elementos dos índices (implícitos) 1, 2 e 3, respectivamente. Considerando que haja intenção de padronizar os códigos deixando sempre os índices explícitos

para facilitar o entendimento dos programas, podemos escrever "`local tabela = {[1] = 'a', [2] = 'b', [3] = 'c'}`". Esse é um caso de escrever o mesmo de forma diferente, quando o sistema não precisa modificar sua semântica.

3. Paráfrase - Reescrita. *Para o usuário: expressão e intenção. Para o sistema: léxico, sintático e semântico.* Além das mudanças previstas no caso 2, deve ocorrer mudança na dimensão semântica, ou seja, novos signos devem introduzir novos significados. Exemplo: seguindo a utilização de tabelas em Lua, considere uma função que retorne uma tabela como resultado. Podemos fazer: "`local function a () return {'a', 'b', 'c'}; end`". Podemos parafrasear o código da seguinte forma: "`local function a () local tabela = {}; tabela[1] = 'a'; tabela[2] = 'b'; tabela[3] = 'c'; return tabela; end`". Esse código retorna exatamente a mesma tabela, mas foi criada uma variável desnecessária, modificando a semântica do sistema. O modo como o sistema interpreta o código causa desperdício de recursos de memória.

4. **Figuras de linguagem - Reformulação**. *Para o usuário: conteúdo e intenção. Para o sistema: léxico e/ou sintático.* Este caso engloba mudança na informação transmitida, pois o conteúdo é alterado para compor uma nova intenção. Do ponto de vista do sistema, a dimensão semântica é mantida, mas deve haver modificação na dimensão léxica e/ou sintática. Exemplo: um programa simples em linguagem Lua, estruturado de modo procedural, pode ser composto de várias funções e de um conjunto de código que eventualmente realiza chamadas a essas funções. Considere os dois trechos de código abaixo, o primeiro apresenta a criação de três funções e o corpo do programa contém a chamada de três funções. O segundo trecho contém quatro funções e a última função faz a chamada das três funções anteriores. Neste trecho, o corpo do programa é formado pela chamada apenas da última função. Os dois trechos de programas têm o mesmo efeito final, mas o segundo usa um mecanismo de figura de linguagem para organizar melhor o código. Ao criar a função ABC, o programador preparou o programa para receber uma chamada em forma de metonímia, onde é possível chamar o 'todo' (a função ABC) pelas 'partes' do programa (funções A, B e C separadamente).

Quadro 3.8 - Exemplo de figuras de linguagem com reformulação

`local function A () print('a'); end`
`local function B() print('b'); end`
`local function C() print('c'); end`
`-- corpo do programa`
`A(); B(); C();`
	`local function A () print('a'); end`
	`local function B() print('b'); end`
	`local function C() print('c'); end`
	`local function ABC()`
	`  A(); B(); C();`
	`end`
	`-- corpo do programa`
	`ABC();`

5. Figuras de linguagem - Reescrita. *Para o usuário: conteúdo e intenção. Para o sistema: léxico, sintático e semântico.* Apesar de não ocorrer mudança na forma, o conteúdo e a intenção são diferentes. Além disso, do ponto de vista do sistema, há mudança nas três dimensões semióticas. Exemplo: em linguagem Lua temos como recurso a criação de *metatables*. Entre as funcionalidades das *metatables* está a possibilidade de dizer como a tabela deve se comportar quando um índice que não existe é chamado. Em uma tabela normal, caso um índice inexistente seja chamado como "`local valor = tabela[10]*5`", esse código vai resultar em um erro porque o valor de 'tabela[10]' é nulo. Se transformarmos 'tabela' em uma *metatable* podemos especificar que sempre que um índice inexistente for chamado, o resultado será '1'. Então, se 'tabela' é uma *metatable* ao solicitar o valor de 'tabela[10]' haverá um resultado se a tabela já tinha valor atribuído e outro resultado se o índice '10' não existia. A chamada de 'tabela[10]' é a mesma, mas internamente o programa interpreta de um modo diferente. Este exemplo consiste em uma metáfora porque o tratamento de índices da *metatable* prepara o programa para tratar índices não existentes como índices existentes, além de esse mecanismo funcionar como um tratamento de erros no caso de acesso a índices inexistentes.

6. Expansão linguística - Reformulação. *Para o usuário: expressão, conteúdo e intenção*. *Para o sistema: léxico e/ou sintático*. Nesse caso, há mudança nas três dimensões da perspectiva do usuário. Há nova forma, conteúdo e intenção, bem como mudanças no léxico e no sintático mas não há mudança na semântica. Exemplo: em linguagem Lua, temos a função 'print' que imprime uma string passada como parâmetro e insere uma quebra de linha. As funções podem ser sobrescritas. Podemos fazer "`print = function (str)`

`print('\n\n string na tela:'..str..'\n\n')`". Esse caso faz com que sempre que a função *print* seja chamada, ela resultará em duas quebras de linha, o valor 'string na tela:', concatenado com a *string* enviada como parâmetro e duas quebras de linha adicionais sejam exibidas. A função *print* foi expandida mas, apesar das mudanças léxicas e sintáticas, a mensagem continua sendo exibida na tela.

7. Expansão linguística - Reescrita. *Para o usuário: expressão, conteúdo e intenção*. *Para o sistema: léxico, sintático e semântico*. Esse é o caso mais complexo porque conta com modificações em todas as dimensões tanto da perspectiva do usuário quanto da perspectiva do sistema. Há mudança na forma, na informação e na intenção, assim como no vocabulário, na gramática e na semântica da linguagem. Ainda assim, é importante lembrar que há limites nesse caso, em especial, para evitar a alteração dos signos que podem causar inconsistências na identidade da aplicação. Exemplo: o caso de criações de funções novas é típico nesse contexto. Um caso de adicionar uma função que ainda não existe na linguagem é uma expansão linguística com reescrita porque houve a adição de um nome do ambiente. Essa nova função poderá ser chamada de qualquer lugar do código. Os trechos de código abaixo ilustram um exemplo de fatoração do código. Do lado esquerdo, temos um exemplo de chamada de função onde parte do código aparece repetida. Do lado direito, a função ABC foi criada com o código repetido e a função F pode chamar a função ABC. Nesse caso, a nova intenção é fatorar o código para facilitar o entendimento.

Quadro 3.9 - Exemplo de expansão linguística com reescrita

local function F () print(‘a’); print(‘b’); print(‘c’); print(‘a’); print(‘b’); print(‘c’); end F();	local function ABC () print(‘a’); print(‘b’); print(‘c’); end local function F() ABC(); ABC(); end F();

3.5.2. Perspectiva da interação

A outra perspectiva que faz parte da ontologia de metacomunicação do modelo proposto é baseada nos tipos de signos estudados na Engenharia Semiótica e seu escopo dentro da aplicação. As representações propostas como extensões em uma aplicação destinada ao ensino e aprendizado de programação têm como objetivo apoiar o entendimento de programas construídos pelos aprendizes, especialmente os programas mais complexos. Uma parte essencial para a elaboração da metacomunicação é a escolha de signos, simples e compostos, e a definição de como podem ser usados para comunicar o que os *designers* pretendem dizer aos usuários. Tratamos aqui de dois aspectos destes signos.

Primeiramente é preciso que o nível de conhecimento necessário para compreender os conceitos seja crescente, sem que sejam criados abismos entre as representações com relação ao nível de conhecimento necessário para entendê-las. Para suprir essa necessidade, percebemos que as representações devem considerar o escopo local, regional e global dos programas.

A fundamentação teórica dessa ideia está ligada aos diferentes tipos de representação de sistemas propostos por Stenning e Oberlander (1995). Os autores estudaram os aspectos representacionais das linguagens em geral, comparando os tipos de raciocínios apoiados em representações textuais e gráficas. Stenning e Oberlander (1995) classificaram os sistemas de representação em: *Minimal Abstraction Representation System* (MARS), *Limited Abstraction Representation System* (LARS), e *Unlimited Abstraction Representation System* (UARS). No sistema mínimo (MARS), um elemento do sistema representacional corresponde a um único elemento do sistema semântico. No sistema limitado (LARS), um

elemento do sistema representacional pode corresponder a um ou mais elementos no modelo semântico. No sistema ilimitado (UARS), como o próprio nome sugere, não há limite para as correspondências entre o sistema representacional e semântico. É possível, para os UARS, estabelecer relações entre grupos ou estruturas de elementos representacionais e grupos ou estruturas de elementos semânticos. Nesse estudo, os autores concluíram que as manipulações e expressões de abstrações limitadas (LARS) são criticamente importantes para apoiar o raciocínio durante os processos de aprendizado.

Em segundo lugar, combinados com os contextos locais, regionais e globais, estão os aspectos ligados à natureza do signo: estáticos, dinâmicos e metalinguísticos. Quando consideramos os signos em uma perspectiva mais abrangente, onde a própria aplicação como um todo é considerada um signo, o sistema de ajuda é um signo metalinguístico. O documento ativo resultante da aplicação do modelo é um mecanismo de ajuda. Desse modo, o documento ativo é, por si só, um grande signo metalinguístico, pois a maior parte dos signos está fazendo referência à aplicação "principal" com o objetivo de explicar, explicitar ou detalhar informações.

Sendo assim, para esta classificação, vamos considerar apenas os signos estáticos e dinâmicos, combinados com as opções de escopo local (foco em um elemento), regional (foco em um conjunto de elementos) e global (foco em todos os elementos). A Tabela 3.6 apresenta a matriz de combinações possíveis da perspectiva da interação de acordo com os tipos de signos e escopo. A variação entre os tipos de signos e escopo permite isolar partes da aplicação para uma melhor exploração e comunicação dos seus significados. Além disso, pode ser criado um nível de gradação entre as representações, considerando a complexidade de cada representação.

Tabela 3.6 - Perspectiva da interação

	Tipos					
Metacomunicação principalmente através de que tipos de signos?	Estáticos			Dinâmicos		
Foco em quantos elementos do programa?	um (local)	um conjunto (regional)	todos (global)	um (local)	um conjunto (regional)	Todos (global)

Em geral, as aplicações voltadas para o desenvolvimento de programas apresentam pelo menos uma representação. Por exemplo, o código de um módulo específico em linguagem Lua, que é incluído em um programa maior, seria considerado como uma metacomunicação através de signos estáticos com foco em um elemento do programa (escopo local); e o programa principal que inclui os outros módulos e contém o código principal seria como uma metacomunicação através de signos estáticos com foco em todos os elementos do programa (escopo global). Nesse caso, há um espaço entre escopo local e global que pode ser ocupado para apoiar o entendimento mais concreto do comportamento do programa. Para isso, suponha que fosse criada uma representação que explicasse como a inclusão dos módulos interfere no funcionamento do programa como um todo, como ocorre a passagem de parâmetros, como é feito o controle de escopo das variáveis e mais todo tipo de informação que ajuda a refletir sobre o programa desenvolvido. Este caso seria como uma metacomunicação através de signos estáticos com foco em um conjunto de elementos do programa (escopo regional).

3.6. Visão completa do modelo

O modelo PoliFacets está representado na Figura 3.19. Os signos de programa e programação são gerados pelo analisador com base na ontologia de domínio, nos programas e sua ferramenta de construção. O construtor que compõe o modelo faz uso da ontologia de metacomunicação e de domínio da aplicação para gerar conceitualmente as facetas e ligações. Em outras palavras, a construção das facetas e suas ligações é o processo de engenharia semiótica em ação.

Figura 3.19 - Modelo PoliFacets

A ontologia de metacomunicação detalhada na seção anterior serve como base de conhecimento para o processo de engenharia semiótica representado pelo construtor no modelo. Esta ontologia explora os signos de programas e programação para criar novas facetas.

O construtor utiliza a ontologia de metacomunicação e os signos de programa e programação para gerar as facetas e ligações entre elas. Uma alternativa de visualização da ontologia de metacomunicação é apresentada na Tabela 3.7. Esta tabela serve como ferramenta epistêmica para ressaltar as oportunidades de *design* das novas facetas. Ela tem como intenção apresentar todo o espaço de criação disponível na ontologia de metacomunicação.

As facetas e suas ligações são o resultado da aplicação do modelo. Esse componente do modelo é uma estrutura conceitual que determina quais facetas compõem o documento e como elas devem estar conectadas umas às outras. Depois disso, é preciso realizar o processo de desenvolvimento de *software*, que está fora do modelo, com a modelagem e projeto de *software* que resultará na instância do documento ativo (Figura 3.18).

O *designer* define cada faceta de acordo com a ontologia de domínio e metacomunicação como uma espécie de *template*, considerando quaisquer tipos de programas. Afinal, a ideia é poder gerar facetas de significados sobre

programas específicos. Cada novo código de entrada deve gerar um novo conjunto de facetas relacionadas.

Do ponto de vista teórico, a geração das facetas é realizada pelo construtor a partir dos signos de programa e programação e da ontologia de metacomunicação. Do ponto de vista prático, é importante que as facetas sejam geradas, o máximo possível, de modo automatizado. Isto seria interessante, especialmente, por dois motivos: a confiabilidade da informação disponibilizada; e a praticidade. As facetas podem ser complexas, por isso é ideal que sejam construídas a partir de algoritmos confiáveis que processem a informação e criem as representações finais de acordo com os requisitos previamente definidos. A praticidade é essencial porque se a criação de facetas desenvolvidas para auxiliar no processo de ensino e aprendizado for trabalhosa demais, pode desmotivar os aprendizes,inviabilizando o estudo dos benefícios desse tipo de modelo e documentação ativa resultante.

As células na Tabela 3.7 são espaços de reflexão a serem preenchidos de acordo com as perspectivas da ontologia de metacomunicação e conforme as decisões de *design*. As ligações entre as células são os elos que o *designer* decidiu construir entre as células que preencheu, ou seja, constituem as ligações entre as facetas. Cada faceta representa o *designer* comunicando aos aprendizes os significados dos programas sob determinados aspectos relacionados à autoexpressão via *software*.

Construtor de ligações

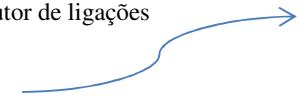


Tabela 3.7 - Grade da ontologia de metacomunicação

			Metacomunicação através de Signos ESTÁTICOS (principalmente)			**Metacomunicação através de Signos DINÂMICOS (principalmente)**		
	Operações Semióticas @ Usuário	**Operações Simbólicas @ Sistema**	LOCAL foca um elemento de programa	REGIONAL foca um conjunto de elementos de programa	GLOBAL foca todos os elementos de programa	LOCAL foca um elemento de programa	REGIONAL foca um conjunto de elementos de programa	GLOBAL foca todos os elementos de programa
Operações sobre o sistema de significação do OBJETO do Documento Ativo	RESSIGNIFICAÇÃO dá novos significados aos signos do programa	-						
	PARÁFRASE expressa os signos do programa de maneira alternativa	REFORMULAÇÃO sem alteração semântica						
		REESCRITA com alteração semântica						
	FIGURAS DE LINGUAGEM faz combinações inéditas com os signos do programa e revela novos significados	REFORMULAÇÃO sem alteração semântica						
		REESCRITA com alteração semântica						
Outro Sistema de Significação	EXPANSÃO LINGUÍSTICA dá novos significados aos signos do programa	REFORMULAÇÃO sem alteração semântica						
		REESCRITA com alteração semântica						

As operações semióticas do ponto de vista do usuário e as operações simbólicas do lado do sistema estão classificadas de acordo com o sistema de significação a que se referem. As operações de ressignificação, paráfrase e figuras de linguagem estão com suas respectivas subdivisões de reformulação e reescrita. Elas são operações sobre o sistema de significação do próprio objeto do documento ativo. A expansão linguística com reformulação ou reescrita ocorre por operações sobre outro sistema de significação. Ao projetar as facetas também é preciso considerar se a metacomunicação será construída principalmente através de signos estáticos ou dinâmicos e, além disso, decidir se o escopo da faceta será local, regional e global.

Há possibilidade de usar o modelo de modo recursivo. Por exemplo, imagine que criássemos dois documentos ativos com base no modelo. O primeiro é uma aplicação A, voltada para o ensino de Java. Ela tem um sistema de significação próprio, uma análise específica e uma ontologia de domínio relativa ao Java. O documento ativo resultante é composto por facetas que explicitam conceitos e fundamentos da linguagem Java. O segundo documento ativo é uma aplicação B, voltada para o ensino da linguagem Lua. O sistema de significação, a análise e a ontologia de domínio teriam especificidades da linguagem Lua. Logo, o documento ativo resultante também seria diretamente relacionado aos conceitos importantes para a linguagem Lua. Conceitualmente, os documentos ativos resultantes da aplicação A e B formam um novo sistema de significação que pode ser analisado gerando uma nova ontologia de domínio que contém dados importantes para as duas linguagens, Java e Lua. O *designer* do documento ativo pode definir o construtor para usar essa ontologia de domínio e a ontologia de metacomunicação para gerar um documento ativo que tem como objetivo tornar a transição entre as duas linguagens mais suave. Ou ainda, o *designer* pode usar as ferramentas disponíveis para ressaltar as diferenças entre as linguagens.