3
Ferramentas de apoio à pesquisa

Conforme antecipado na Introdução, os estudos empíricos realizados durante a pesquisa envolveram o uso de basicamente três sistemas: AgentSheets, PoliFacets e SideTalk. Este capítulo apresenta os detalhes sobre eles.

3.1
AgentSheets

O AgentSheets é uma ferramenta de programação visual por manipulação direta, indicada para a criação de jogos e simulações por usuários sem conhecimento de programação (REPENNING e IOANNIDOU, 2004). Desde sua origem, tem sido usado em iniciativas e pesquisas sobre o ensino de raciocínio computacional, especialmente para alunos de nível fundamental e médio. Trata-se de um ambiente de EUD, em que os usuários "arrastam e soltam" blocos de condições e ações que formam "regras de comportamento" para os personagens do projeto (agentes). Os usuários então criam um programa sem precisar codificar de forma textual. A Figura 3-1 apresenta a interface do AgentSheets.

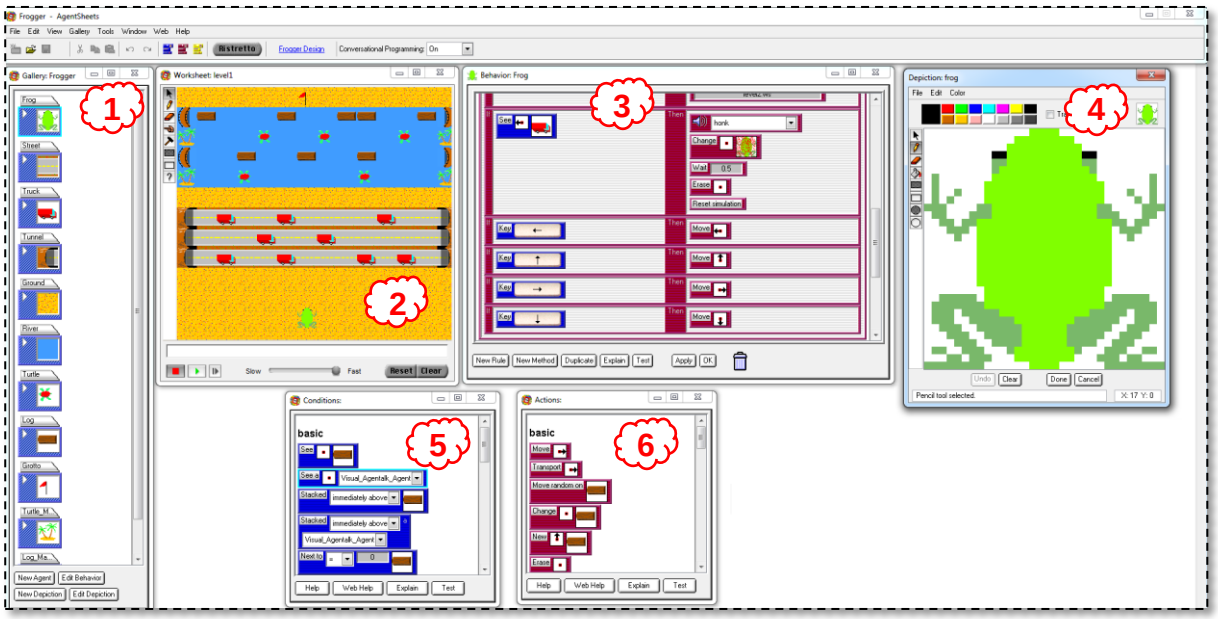


Figura 3-1. Interface do AgentSheets

Os projetos criados no AgentSheets possuem os seguintes componentes:

- Agentes: são os "objetos" no projeto, que podem ser personagens com participação ativa, objetos "inanimados" em uma cena, mas com alguma função específica, ou objetos usados meramente para decoração. Os agentes criados ficam dispostos na galeria de agentes (item 1 da Figura 3-1). Os agentes podem ter uma ou mais *depictions* (aparência ou representação). Cada *depiction* é uma representação visual do agente (item 4 da Figura 3-1). Por exemplo, um agente que se move nas quatro direções pode ter uma *depiction* para representar cada uma delas.
- Planilhas: são as áreas de trabalho, os "mundos" do jogo ou simulação (item 2 da Figura 3-1). Os agentes são posicionados nas planilhas para compor o cenário do projeto, que pode ter uma ou mais delas. Em geral, diferentes planilhas são usadas para diferentes níveis de um jogo.
- Regras: cada agente em um projeto pode ter seu comportamento programado com regras na forma de expressões "se-então". Há comandos que se aplicam aos agentes em si e comandos que se aplicam às *depictions* dos agentes. Por exemplo, é possível criar uma regra para mudar a aparência de um agente quando ele se mover para uma determinada direção. Na Figura 3-1, o item 3 é a janela de comportamento do agente *frog*, onde estão todas as suas regras. Para adicionar os comandos a uma regra, o usuário do AgentSheets "arrasta e solta" o bloco correspondente das janelas de condições (item 5) e ações (item 6) para a janela de comportamento (item 3). A Figura 3-2 mostra as regras para programar o movimento do agente *frog*. Por exemplo, a primeira regra quer dizer: "SE a tecla 'seta para esquerda' do teclado for pressionada, ENTÃO o agente move-se para a esquerda".

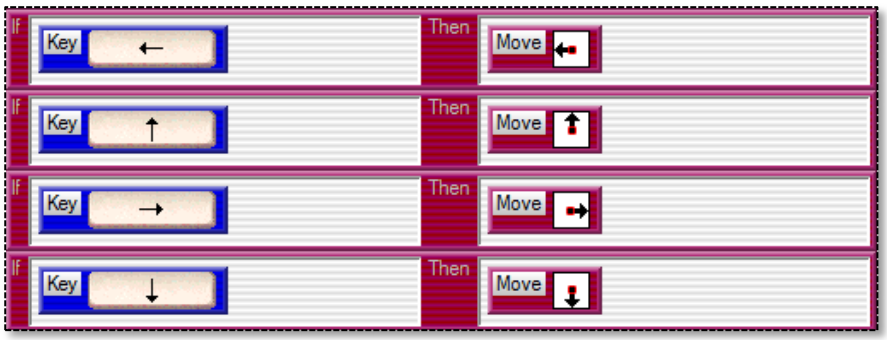


Figura 3-2. Regras de movimento pelas setas

O AgentSheets é uma das ferramentas utilizadas no projeto *Scalable Game Design* (SGD), liderado pelo professor e pesquisador Alexander Repenning, da Universidade do Colorado em Boulder (EUA). Este projeto tem como missão: "*Reinventing computer science in public schools by motivating & educating all students including women and underrepresented communities to learn about computer science through game design starting at the middle school level*"[1]. Além

[1] http://sgd.cs.colorado.edu/wiki/Scalable_Game_Design_wiki

dos Estados Unidos, o SGD atua em outros países, entre eles o Brasil, cuja versão chama-se *Scalable Game Design Brasil* (SGD-Br)[2] e tem sido coordenada e executada pelo SERG, grupo de pesquisa em Engenharia Semiótica da PUC-Rio.

O projeto SGD-Br iniciou-se em 2010, a partir da parceria com uma escola pública de Niterói-RJ. A partir de 2012, duas escolas particulares (uma nacional e uma internacional) da cidade do Rio de Janeiro passaram a integrar o grupo de escolas participantes. O objetivo do projeto é obter uma sólida compreensão dos desafios, oportunidades e requisitos tecnológicos específicos para um programa mais amplo de educação para a computação em escolas brasileiras. A maior motivação do SGD-Br é ajudar a promover a aquisição do raciocínio computacional para que as pessoas possam entender melhor o significado social da computação, dominar conceitos computacionais básicos e aprender a se expressar através de programas de computador (DE SOUZA, GARCIA, *et al.*, 2011) (DE SOUZA, SALGADO, *et al.*, 2014). O SGD-Br, assim como a versão original americana, adotou o AgentSheets como principal ferramenta de aprendizado de programação nas escolas parceiras. Alguns jogos criados no AgentSheets por alunos do ensino fundamental participantes do SGD-Br fizeram parte dos dados coletados no último estudo.

Um dos resultados da pesquisa realizada no SGD-Br, alinhado à inclinação semiótica do SERG, foi o desenvolvimento do PoliFacets, uma ferramenta de estímulo à reflexão sobre os significados explícitos e implícitos presentes nos jogos e simulações construídos com o AgentSheets.

3.2 PoliFacets

O PoliFacets surgiu dentro da pesquisa do SERG e do projeto SGD-Br originalmente como um sistema de apoio a alunos e professores do projeto, que tinham o AgentSheets como ferramenta de aprendizado. Posteriormente, ele foi estendido e redefinido como um modelo conceitual mais abstrato, cujo foco é "*o design da metacomunicação de documentos ativos para apoiar o processo de ensino e aprendizado de programação*" (MOTA, 2014). Apresentaremos aqui os

[2] http://www.serg.inf.puc-rio.br/wiki/index.php/P%C3%A1gina_principal

detalhes práticos da aplicação do modelo para o AgentSheets, proposto por Mota como uma instância do modelo PoliFacets e referido pela autora como PoliFacets-AS. Nesta tese, nos referiremos ao PoliFacets-AS apenas como PoliFacets, pois não entraremos nos detalhes do modelo conceitual. Além disso, é por "PoliFacets" que os alunos e professores do projeto conhecem a ferramenta, na forma como está disponível hoje para acesso[3]. O PoliFacets, segundo os seus *designers* é:

> [...] um mediador que atua entre o AgentSheets e seus usuários, com o objetivo de facilitar a comunicação e explicitar conceitos, tornando a mensagem do usuário (designer dos jogos) mais clara. O nome faz referência às múltiplas "facetas" que representam um projeto do AgentSheets. (PoliFacets)

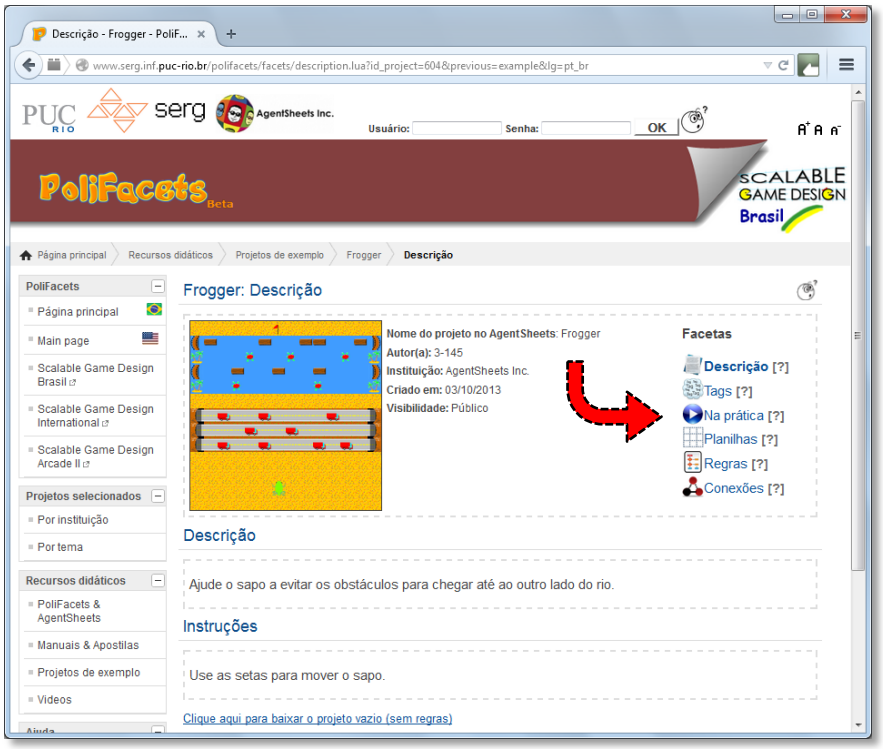


Figura 3-3. "Frogger" no PoliFacets (faceta descrição)

Para analisar um projeto no PoliFacets, o usuário deve enviá-lo ao sistema e fornecer alguns detalhes sobre ele. O PoliFacets então gera as páginas correspondentes às facetas, cujo conteúdo é obtido de duas formas: automaticamente através de *parser* dos dados do projeto e manualmente através das informações passadas durante o seu envio ao sistema. A versão atual do PoliFacets (a mesma considerada nos estudos desta tese) possui seis facetas: descrição, *tags*, na prática, planilhas, regras e conexões. A Figura 3-3 mostra a tela de abertura do jogo Frogger no PoliFacets, que corresponde à faceta

[3] http://www.serg.inf.puc-rio.br/polifacets

descrição. À direita da seta vermelha, vê-se o menu de acesso às seis facetas. As subseções a seguir apresentam uma descrição mais detalhada de cada uma delas.

3.2.1
Faceta *descrição*

A faceta descrição apresenta duas partes: uma descrição geral do projeto e o conjunto de instruções sobre ele, ambas fornecidas pelo usuário (normalmente o autor do jogo) durante o processo de *upload*. Esta é a faceta inicial apresentada ao usuário quando ele acessa qualquer projeto (Figura 3-3). Como o conteúdo principal não é gerado automaticamente pelo sistema, segundo Mota (2014, p. 122), "*esta é a melhor oportunidade do criador do projeto explicar em linguagem textual a sua intenção de comunicação, ele pode contar uma história ou explicar o comportamento dos agentes de acordo com o seu desejo*".

3.2.2
Faceta *tags*

A faceta tags mostra um diagrama com os comandos utilizados no projeto proporcionalmente à intensidade de uso. Através do tamanho das palavras (apresentadas numa nuvem de *tags*) é possível saber quais comandos são mais utilizados. Quanto mais frequente um comando, maior ele aparece na nuvem. Além disso, é possível conhecer que agentes usam cada comando. A Figura 3-4 apresenta as *tags* Frogger. Nela, vê-se a lista de agentes que utilizam o comando "apagar". Ao clicar no link com o nome do agente, o usuário é direcionado à faceta *regras*, mais precisamente à seção específica do agente escolhido.

Figura 3-4. Tags do projeto "Frogger"

3.2.3
Faceta *na prática*

A faceta na prática corresponde ao módulo executável do projeto, na forma de um *applet* Java, gerado e disponibilizado pelo próprio AgentSheets. Conforme esclarece Mota (2014, p. 123), a faceta *na prática* "*é o produto resultante da programação desenvolvida no AS*". Nesta faceta, além de o usuário poder jogar, ele pode (re)ver a descrição e as instruções do projeto. A Figura 3-5 apresenta a tela da faceta *na prática* para o jogo Frogger.

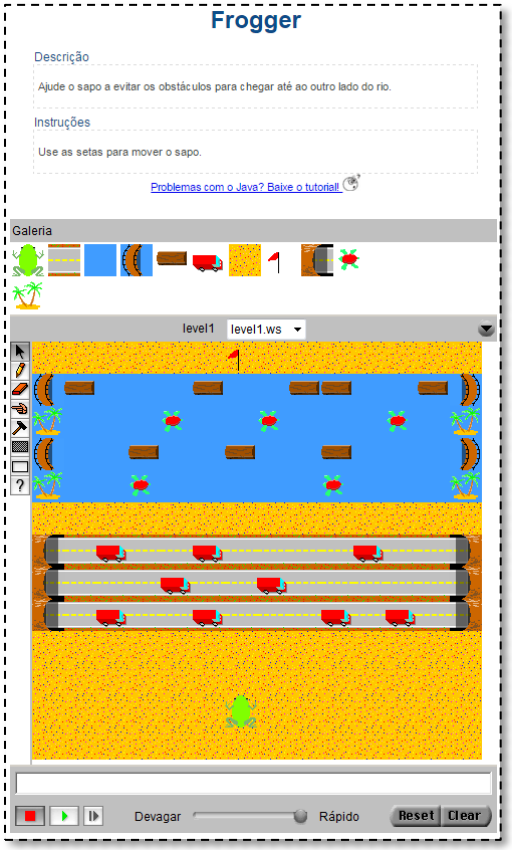


Figura 3-5. Applet do jogo "Frogger"

3.2.4
Faceta *planilhas*

A faceta planilhas permite um verdadeiro escrutínio das planilhas do projeto. Através dela, é possível investigar vários aspectos relativamente ocultos no AgentSheets. Por exemplo, não é possível visualizar facilmente quando agentes estão sobrepostos uns aos outros, ou quais componentes visuais fazem parte de uma imagem de fundo, ou ainda quantas instâncias de um mesmo agente estão distribuídos numa planilha. Na faceta, essas e outras informações são

prontamente obtidas, a partir de uma grade que representa a disposição espacial dos elementos visuais do jogo. A Figura 3-6 apresenta a visualização da planilha "level1" do jogo Frogger, com a demonstração de algumas dessas funcionalidades. As setas vermelhas menores mostram que os agentes *street* e *river* estão escondidos. Comparando a área da planilha da Figura 3-6 com a planilha original na Figura 3-5, é possível notar a ausência dos agentes azuis na área do rio e dos agentes formando a estrada por onde passam os caminhões. Além disso, percebe-se que há agentes *ground* onde "deveria" estar o rio. A seta vermelha maior aponta para a área de exibição dos agentes empilhados posicionados na célula B2. Nesta pequena janela, são listados todos os agentes da célula, empilhados na ordem de exibição: *ground*, *river* e *log*. Mota (2014, p. 127) ressalta a faceta *planilhas* como uma boa vitrine para o modelo de documentação ativa proposto: "*O aprendiz interage com os signos da interface para explorar a planilha dos jogos e simulações, compreendendo detalhes que não estavam disponíveis em outras visualizações. Essa interação como uma 'conversa' com o sistema é o que caracteriza a documentação ativa.*"

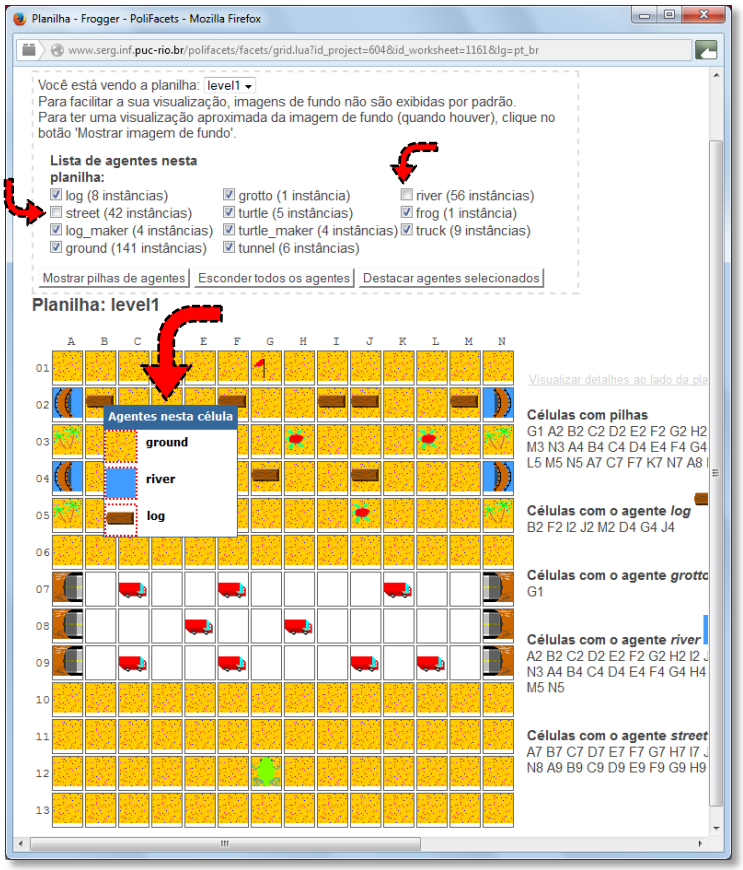
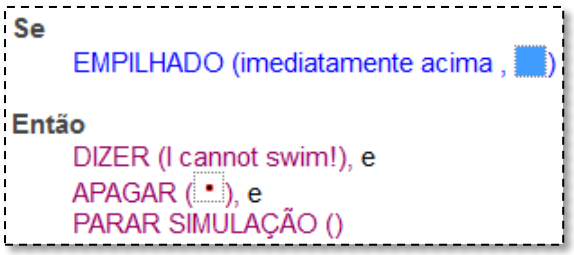
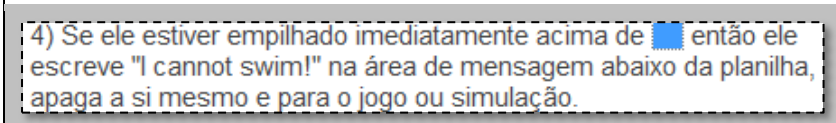


Figura 3-6. Visualização em grade da faceta "planilhas"

3.2.5
Faceta *regras*

A faceta regras tem o objetivo de explicitar a programação realizada no AgentSheets, disponibilizando duas formas alternativas: a) uma listagem do código semelhante aos blocos de comandos no AgentSheets e b) cada regra é uma sentença com uma tradução próxima à linguagem natural. A Tabela 3-1 traz um exemplo de regra representada nestas três formas possíveis.

Tabela 3-1. Formas de representar o código de programas do AgentSheets

Tipo	Representação
Código em blocos (AgentSheets)	
Código em texto similar aos blocos (PoliFacets)	
Código traduzido para linguagem pseudonatural (PoliFacets)	

A relação entre as três representações é assim esclarecida por Mota:

> A linguagem de código do AS [AgentSheets][segunda linha da Tabela 3-1] é bastante icônica (refere-se diretamente aos comandos e imagens do AS), pode ser vista como um ponto de mediação entre as regras construídas através da linguagem visual do AS [primeira linha] e o texto em linguagem pseudonatural do PoliFacets-AS [terceira linha]. (MOTA, 2014, p. 128)

3.2.6
Faceta *conexões*

A faceta conexões informa ao usuário como os agentes estão interligados entre si através das regras. Para cada agente, é apresentado um diagrama de conexões, indicando com quais outros agentes ele se relaciona.

A Figura 3-7 apresenta o diagrama de conexões do agente *log*. A espessura da linha que faz a ligação entre os agentes representa a quantidade de regras que

garantem essa conexão. Abaixo do diagrama há uma listagem dessas regras. Ao clicar em uma delas, é exibida uma caixa (seta vermelha) com a descrição da regra, em linguagem pseudonatural, de forma semelhante à sua apresentação na faceta *regras.* O diagrama também apresenta ligações indiretas entre os agentes: se um agente "A" faz referência a um agente "B", no diagrama de conexões do agente "A", aparece uma linha verde em direção ao agente "B", representando uma ligação direta (todas as ligações da Figura 3-7 são diretas); já no diagrama do agente "B", aparece uma linha cinza em direção ao agente "A", representando uma ligação indireta e expressando para o usuário que ele deve (caso ele deseje) se dirigir ao agente A para verificar qual regra liga os dois agentes.

A faceta *conexões* tem um papel privilegiado entre as facetas no que diz respeito à exploração dos significados decorrentes da programação:

> Os diagramas de conexões entre os agentes são importantes porque apoiam a exploração dos limites de influência que um agente tem sobre o comportamento de outros agentes. Usando essa faceta, os estudantes e professores têm mais opções para "dividir e conquistar", no que diz respeito à compreensão da complexidade lógica dos jogos e simulações. Além disso, a significação e comunicação das relações entre os agentes na estrutura do programa são essenciais para o domínio da complexidade cognitiva nas tarefas de programação. (MOTA, 2014, p. 132)

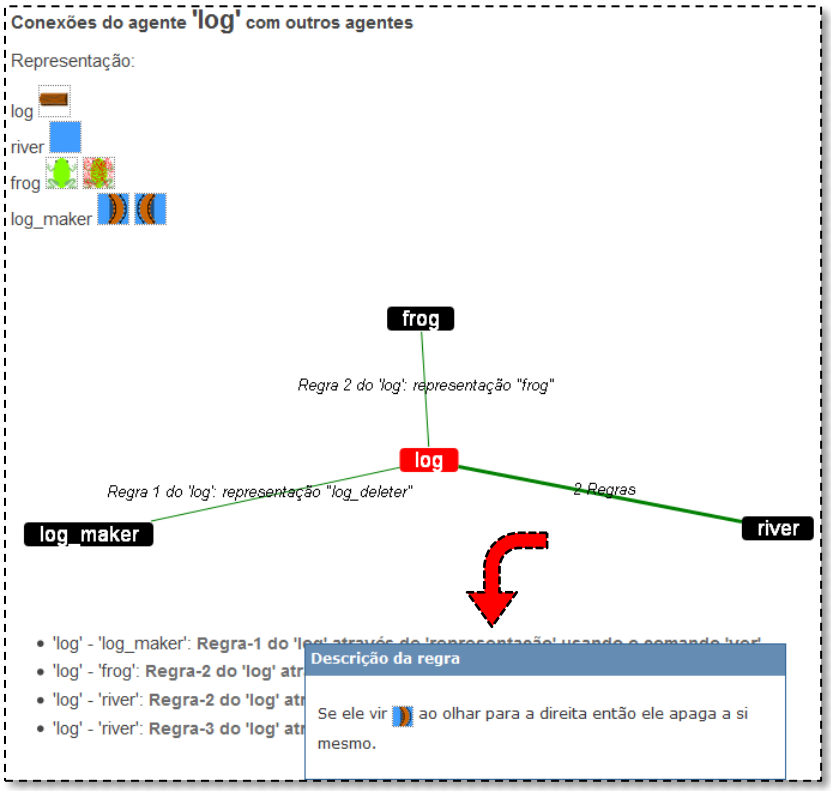


Figura 3-7. Diagrama de conexões do agente *log*

No último estudo realizado para esta pesquisa, os alunos utilizaram a infraestrutura do PoliFacets para construir uma comunicação mediada com o SideTalk, ferramenta descrita na próxima seção.

3.3 SideTalk

As primeiras pesquisas com o SideTalk iniciaram-se ainda em 2007, quando ele se chamava WNH – *Web Navigation Helper*. Tecnicamente o SideTalk é uma extensão para o navegador Firefox, construída a partir do gravador de macros CoScripter (LESHED, HABER, *et al.*, 2008). O SideTalk "herdou" do CoScripter a capacidade de gravar interações diversas em páginas na Web, gerando *scripts* de navegação. É em sintonia com as ações gravadas no *script* que os diálogos de mediação aparecem. A seguir descrevemos as três fases de pesquisa que levaram o SideTalk à sua configuração atual.

3.3.1 Primeira fase

Os primeiros trabalhos com o SideTalk deram-se no contexto de uma pesquisa de mestrado do Departamento de Informática da PUC-Rio. O foco da pesquisa era o uso do então WNH como uma ferramenta de promoção da acessibilidade na Web. A pesquisa empreendida no período, os experimentos realizados e uma ferramenta inicial desenvolvida foram abordados na dissertação de mestrado (INTRATOR, 2009) e em publicações relacionadas (INTRATOR e DE SOUZA, 2008) (INTRATOR e DE SOUZA, 2009).

À época, foram realizados experimentos a fim de comprovar a viabilidade técnica do WNH. Os experimentos foram feitos com usuários finais de dois perfis: deficientes visuais e analfabetos funcionais. Para esses experimentos, ainda não havia uma ferramenta para a criação dos diálogos, que foram elaborados pelos próprios pesquisadores. Ou seja, não havia o envolvimento de usuários atuando como *designers* para estes testes iniciais. Apesar de alguns obstáculos decorrentes principalmente das necessidades especiais de cada perfil, em linhas gerais, demonstrou-se que a abordagem de diálogos de mediação criados a partir de *scripts* era uma alternativa promissora para a acessibilidade na Web.

As figuras abaixo apresentam capturas de tela da primeira versão do WNH, obtidas na dissertação de Intrator (2009).

Figura 3-8. Tela inicial do WNH (primeira versão) (INTRATOR, 2009, p. 34)

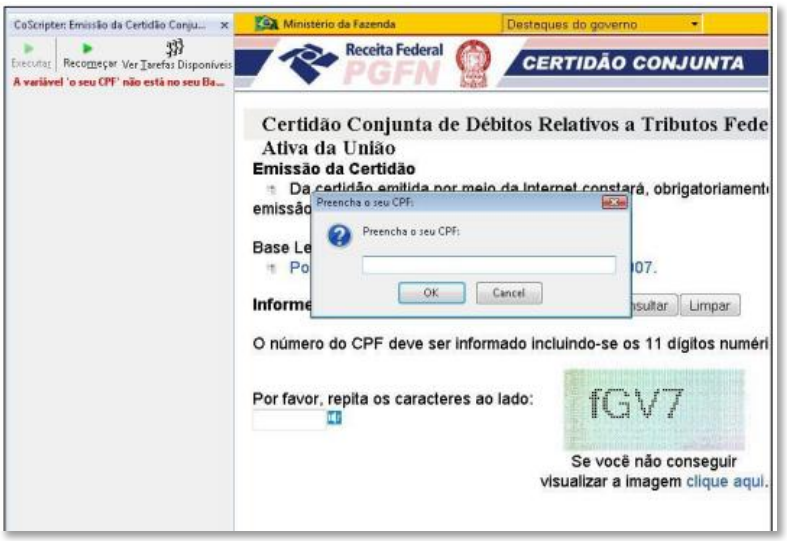


Figura 3-9. Navegação com WNH (primeira versão) (INTRATOR, 2009, p. 48)

3.3.2 Segunda fase

A partir de 2009, iniciou-se a segunda etapa de pesquisa com o WNH, já durante o nosso mestrado. A pesquisa concentrou-se em torno do desenvolvimento de um editor de diálogos que permitisse uma elaboração altamente customizada dos diálogos de mediação. Tecnicamente, esta foi a maior diferença entre as duas versões: a partir de agora era possível para qualquer usuário auxiliar alguém com necessidades especiais a navegar na web, com a mediação dos diálogos do WNH.

Em termos de pesquisa, a maior contribuição desta etapa foi o aprendizado sobre o modelo ideal de acessibilidade a ser aplicado no WNH. Inicialmente, acreditava-se que um grupo de usuários voluntários se disponibilizaria a criar *scripts* genéricos de tarefas comuns da web para usuários quaisquer, com necessidades especiais. Entretanto, a partir de estudos empíricos relatados na nossa dissertação (MONTEIRO, 2011) e em outras publicações (DE SOUZA, MONTEIRO e INTRATOR, 2010) (DE SOUZA e MONTEIRO, 2010) (MONTEIRO e DE SOUZA, 2011) (MONTEIRO, DE SOUZA e LEITÃO, 2013), viu-se que o mais indicado seria contar com um modelo de comunicação "um pra um", ou seja, um usuário com necessidades especiais teria o auxílio do WNH, mas com diálogos criados por alguém próximo a ele, que teria condições de construir uma mensagem mais adequada a suas necessidades.

As imagens a seguir mostram algumas telas do editor de diálogos do WNH, desenvolvido durante a segunda fase de pesquisa.

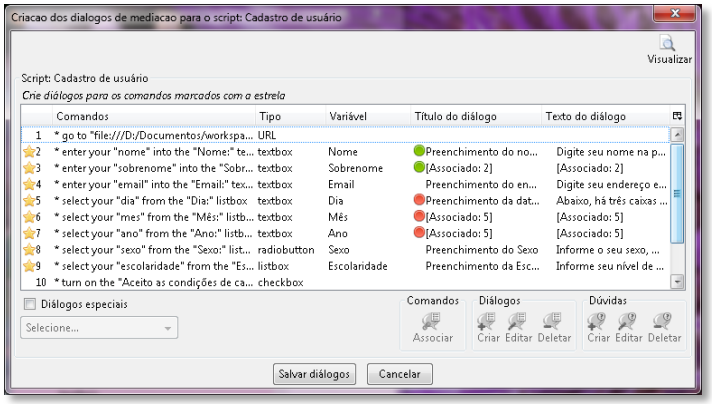


Figura 3-10. Tela inicial do editor de diálogos

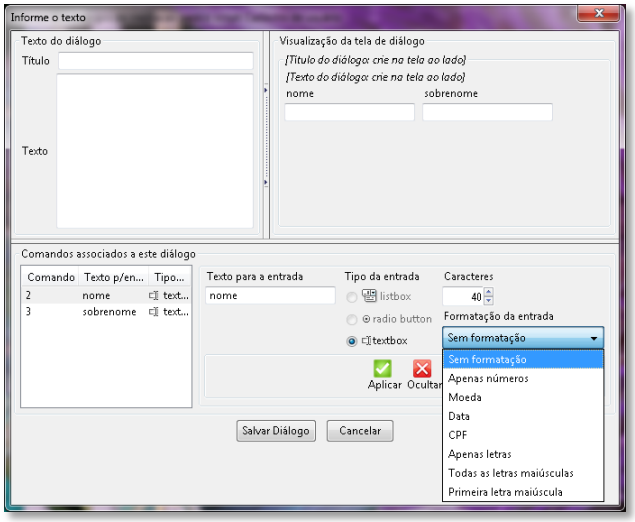


Figura 3-11. Tela de edição de um diálogo

3.3.3
Terceira fase

A pesquisa com o WNH teve continuidade durante o nosso doutorado. O principal salto foi o alargamento do significado do WNH, que nasceu como uma ferramenta dedicada à acessibilidade e que seguiu amadurecendo para um completo sistema de EUD voltado para comunicação em um sentido mais amplo. Esta evolução se refletiu principalmente no nome da ferramenta, que no fim de 2013 foi rebatizada para SideTalk, palavra que remete à ideia de conversa secundária, alternativa à uma conversa principal.

De acordo com a Engenharia Semiótica, os signos metalinguísticos (seção 2.5) são signos que se referem a outros signos na interface, comumente evidenciados em mensagens de ajuda e de *feedback* do sistema (DE SOUZA, 2005). Seguindo esta visão, os diálogos de mediação do SideTalk são um grande conjunto de signos metalinguísticos que se referem claramente às páginas web que integram a conversa mediada. O autor dos diálogos de mediação, ao construir sua conversa, precisa inicialmente interpretar a mensagem do *designer* original (quem criou a página) e então construir sua própria mensagem, podendo manter ou alterar livremente o que foi inicialmente comunicado. Assim, este usuário entra num interessante processo de *End-User Semiotic Engeneering*, levando uma mensagem de "muitas vozes" ao seu interlocutor.

Este processo de mediação mostra-se um terreno fértil para investigação de fenômenos de comunicação como autoexpressão e autorrepresentação de *end-users*, fazendo do SideTalk uma poderosa ferramenta de pesquisa.

Para ficar mais claro como o SideTalk funciona na prática, vamos apresentar uma conversa que explica como se pesquisa imagens do tipo "ícone" no Google. Após acessar a conversa a partir de uma lista (Figura 3-12), o primeiro diálogo é exibido (Figura 3-13), antes mesmo de o *script* ser executado. Em seguida, o *script* inicia e os demais diálogos da conversa são exibidos em momentos pré-determinados pelo criador. As imagens a seguir (Figura 3-14 a Figura 3-19) apresentam a sequência de exibição. As marcações em verde, feitas de forma automática pelo CoScripter (setas vermelhas), ressaltam os elementos da página associados aos respectivos passos do *script* relacionados a cada diálogo.

Neste momento, não entraremos em detalhes sobre como a conversa é criada (uso do gravador de *script*s e do editor de diálogos). O objetivo é mostrar que qualquer conversa, seguindo qualquer estilo ou retórica, pode ser criada com o SideTalk. Observe que, por exemplo, se o foco fosse acessibilidade, provavelmente, não haveria diálogos intermediários entre a entrada da palavra pra pesquisa (Figura 3-15) e o resultado já filtrado (Figura 3-19). O usuário seria levado diretamente a seu objetivo.

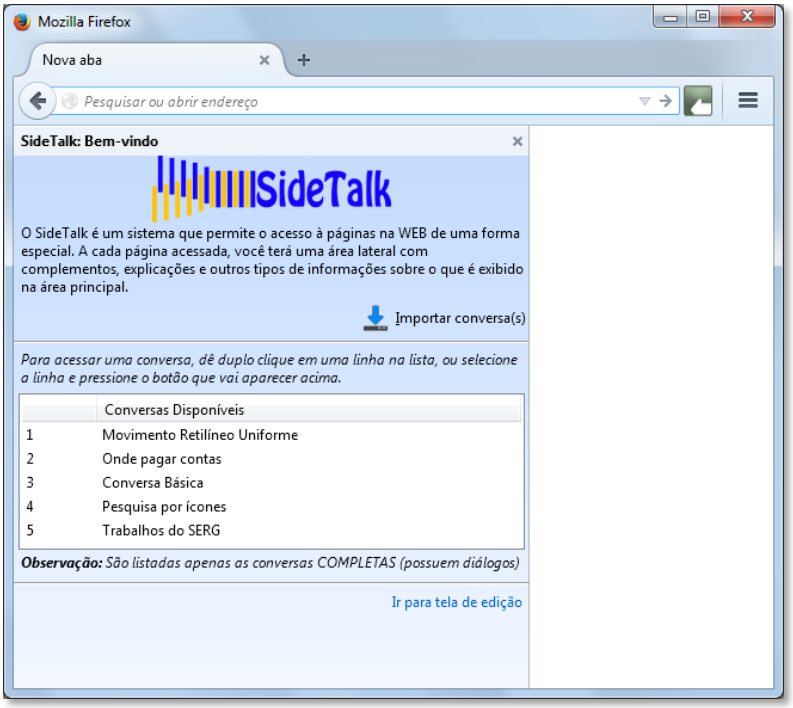


Figura 3-12. Tela inicial do SideTalk com lista de conversas

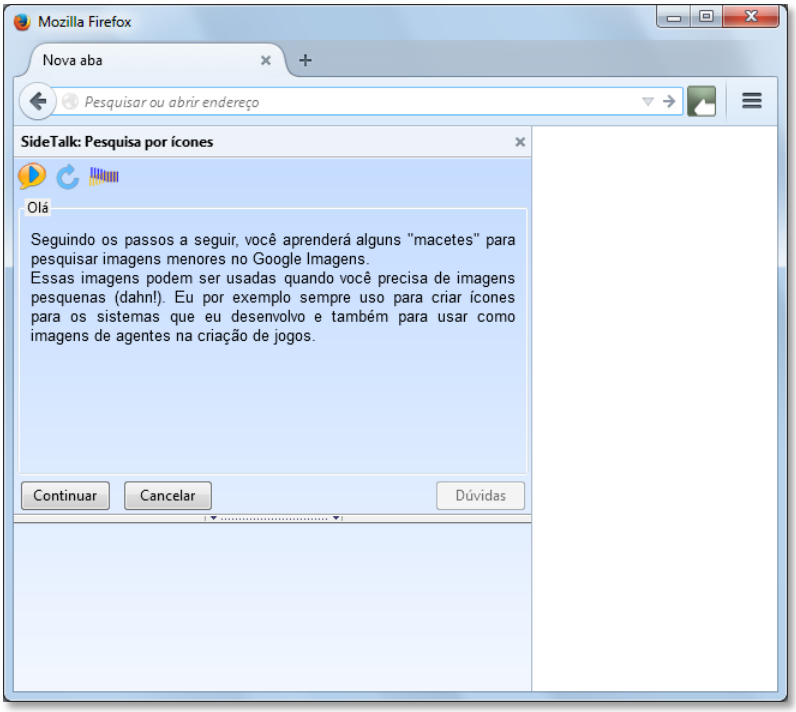


Figura 3-13. Diálogo de abertura

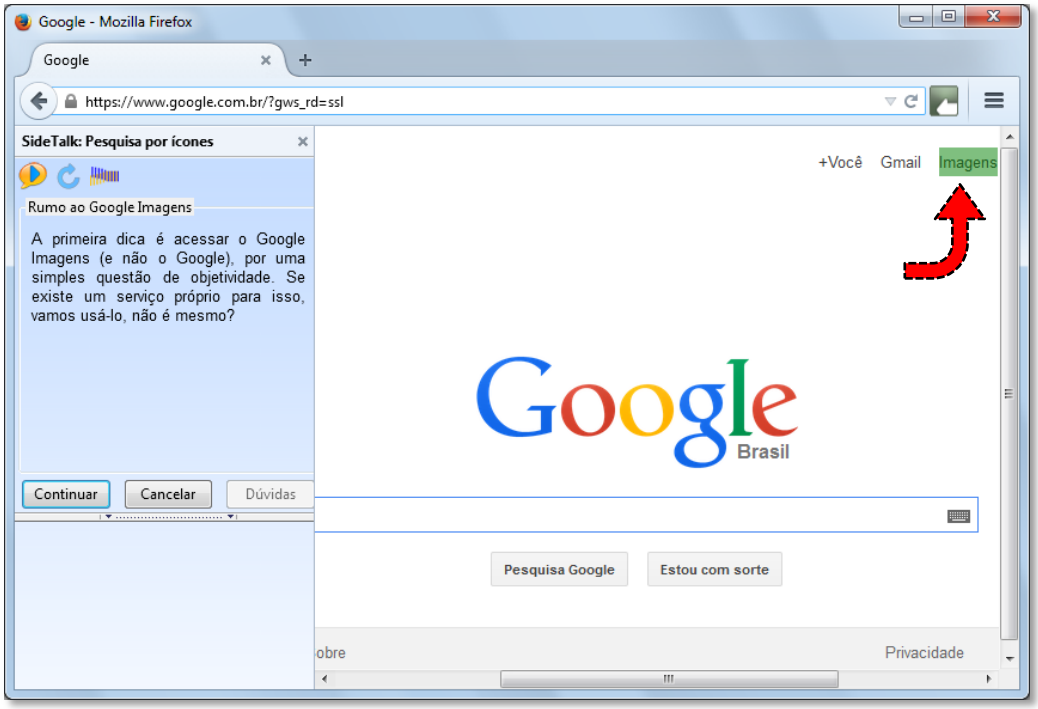


Figura 3-14. Diálogo "Rumo ao Google Imagens"

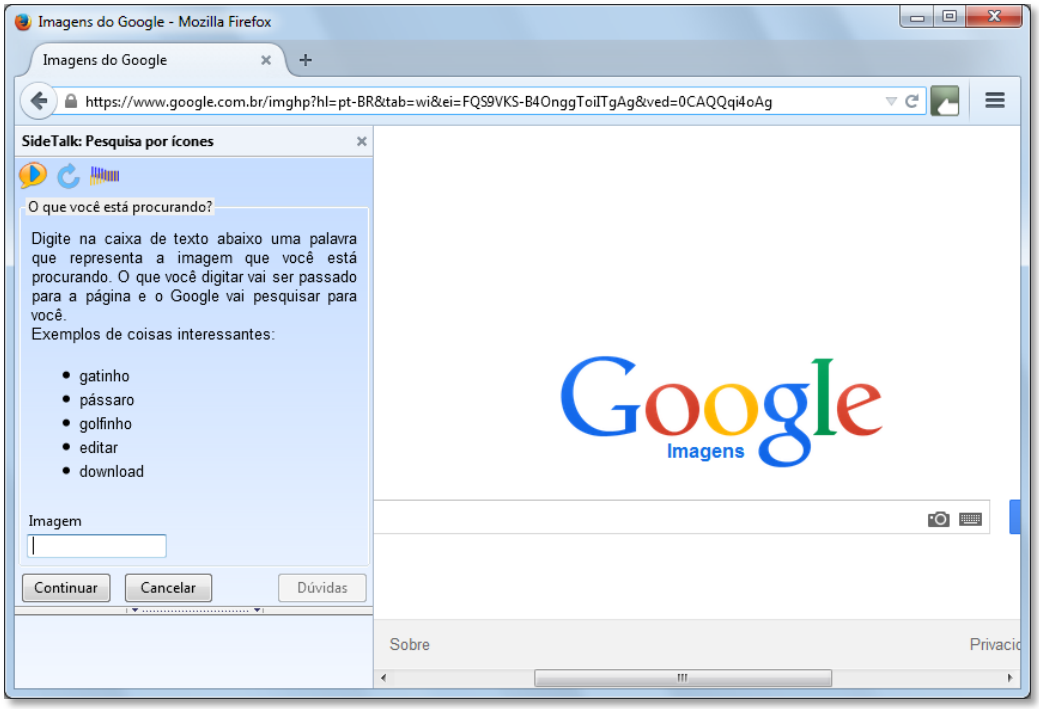


Figura 3-15. Diálogo "O que você está procurando?"

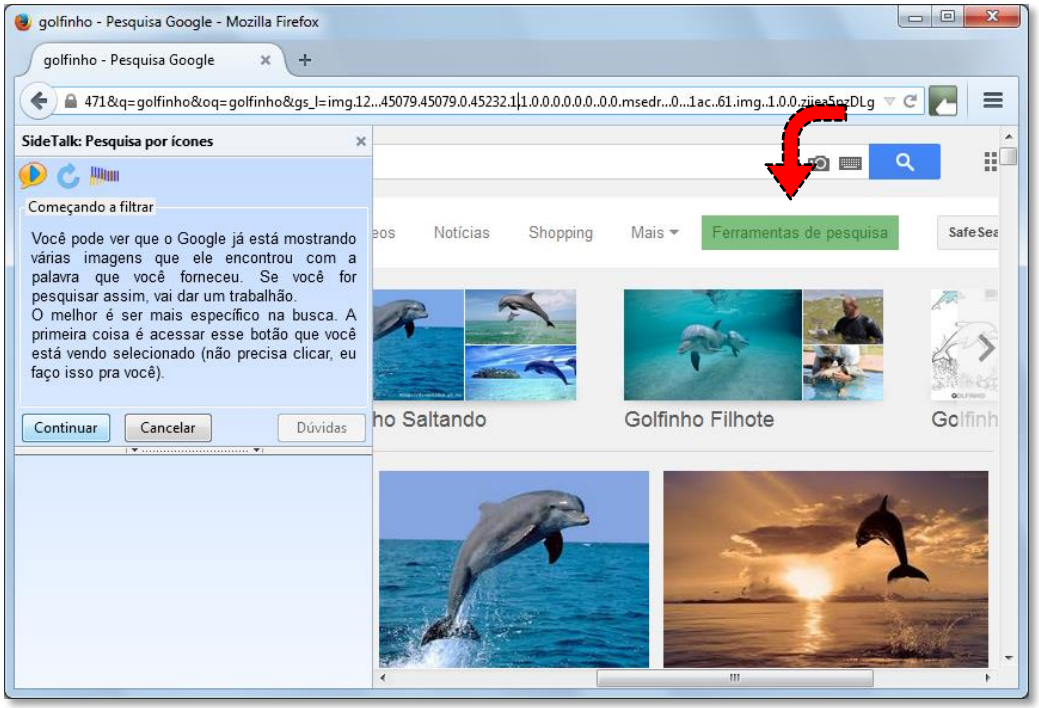


Figura 3-16. Diálogo "Começando a filtrar"

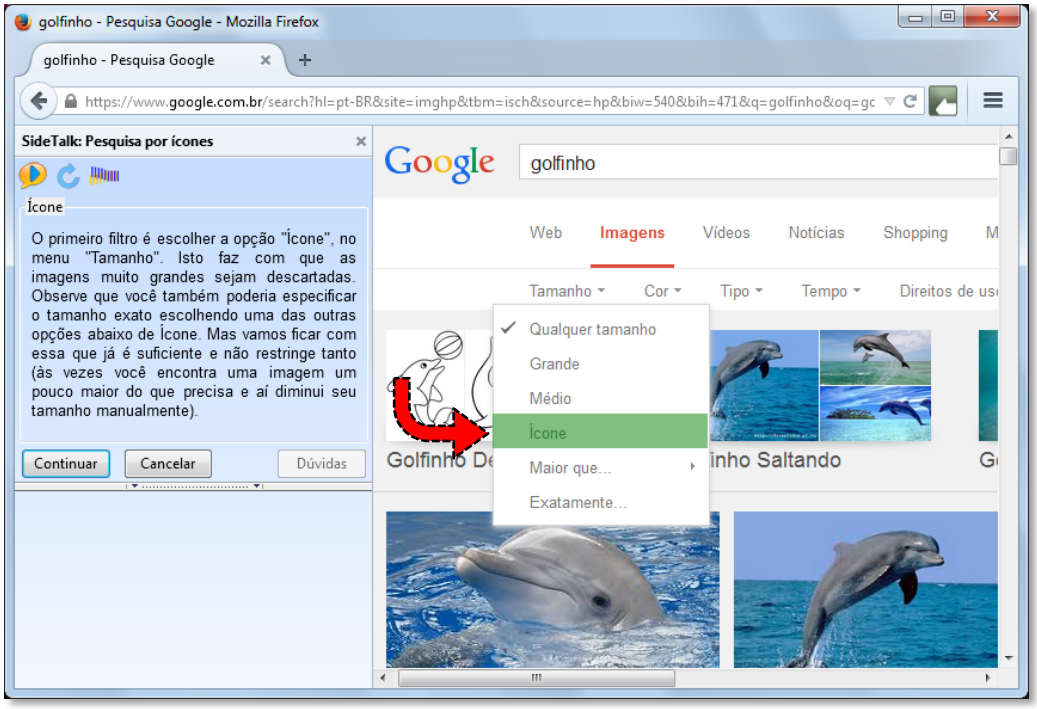


Figura 3-17. Diálogo "Ícone"

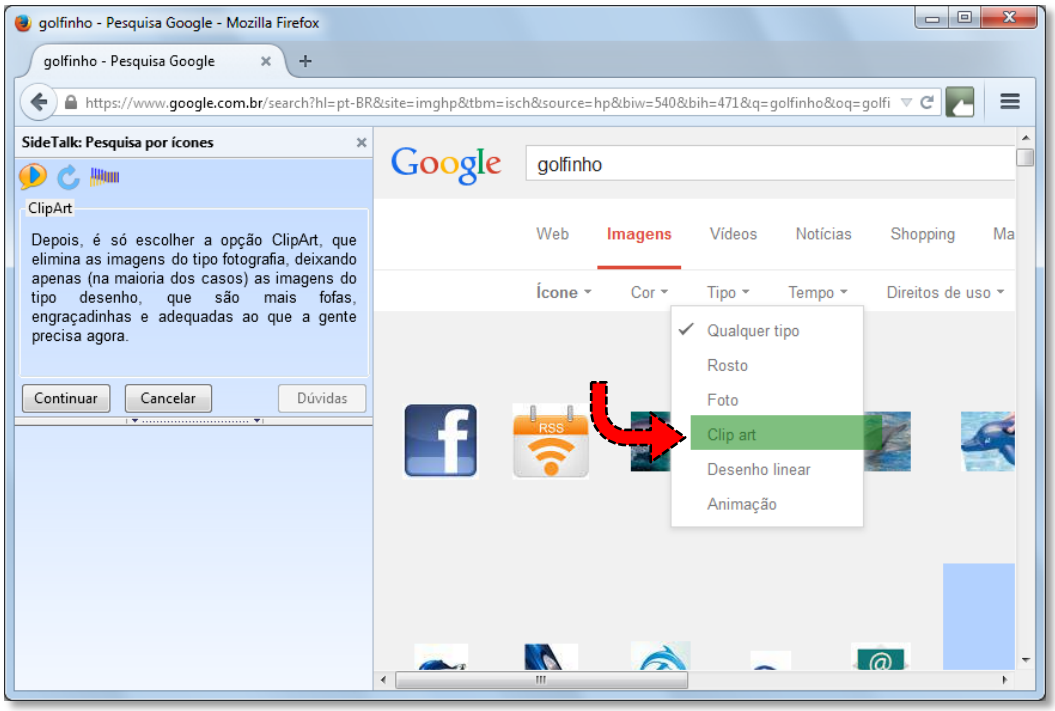


Figura 3-18. Diálogo "ClipArt"

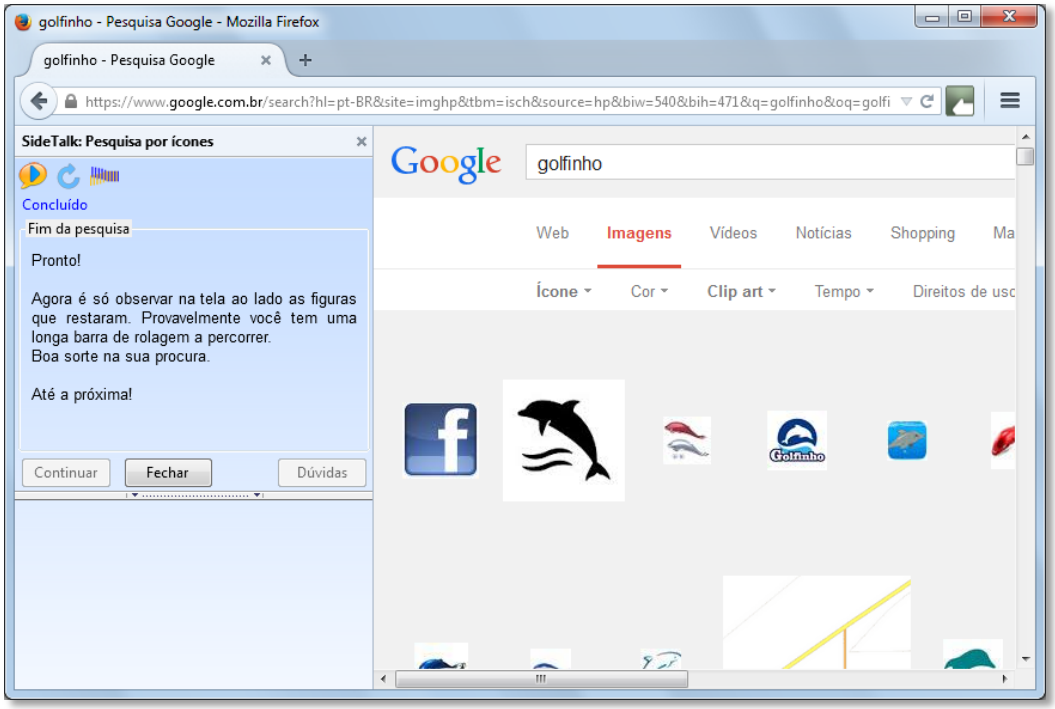


Figura 3-19. Diálogo "Fim da pesquisa"

3.3.4
Criação de conversas mediadas

A criação de conversas mediadas pelo SideTalk possui duas fases bem delimitadas: 1) gravação e edição do *script* de navegação; 2) criação dos diálogos de mediação associados aos passos do *script* previamente gravado.

Como comentado no início deste capítulo, o SideTalk foi construído sobre o CoScripter (LESHED, HABER, *et al.*, 2008), que provê a função de gravação dos *scripts* que compõem as conversas mediadas pelo SideTalk. A Figura 3-20 mostra a tela de edição de *scripts*, com o *script* da conversa "Pesquisa por ícones" aberto.

A criação da conversa começa com a gravação do *script*, que é feita pelo modo "Editar conversas" do SideTalk (ao contrário das imagens exibidas anteriormente, que apresentavam a interação no modo "Acessar conversas"). As funções de gravação (gravar, parar, executar) estão na barra de ferramentas (área 1 da Figura 3-20). Ao iniciar a gravação, todas as ações realizadas no browser serão registradas na área de edição do *script* (área 2), em uma linguagem próxima à natural. Por exemplo, "`go to 'www.google.com.br`" é responsável por abrir a URL corresponde no browser; "`click the 'Imagens' link`" é responsável pelo acionamento do link correspondente; e assim por diante.

A motivação original do CoScripter era a automatização de tarefas repetitivas, ou seja, o usuário gravaria os passos necessários para a execução de uma determinada tarefa na web, salvaria esses passos num *script* e, posteriormente, poderia realizar a tarefa novamente apenas executando o *script*, sem precisar passar por cada passo isoladamente; todos os passos seriam reexecutados automaticamente pelo CoScripter. Dentro da taxonomia específica de EUD, diz-se que a criação de *scripts* pelo CoScripter é um tipo de *programação por demonstração* (CYPHER, 1993), pois o usuário "demonstra" o que ele quer que o sistema faça.

O SideTalk tira proveito da funcionalidade do CoScripter de "quebrar" uma navegação em passos para permitir a um usuário "anotar" esses passos através dos diálogos de mediação. Além do apelo da automatização de tarefas, a equipe do CoScripter também investiu na possibilidade de compartilhamento dos *scripts* criados por uma comunidade de usuários[4]. Uma das principais funcionalidades que viabiliza esse compartilhamento é o uso do banco de dados pessoal (área 3 da Figura 3-20). Funciona da seguinte forma: sempre que o "autor do *script*" quiser que um determinado comando tenha seu significado completado com alguma informação passada pelo "usuário", esta informação deve estar registrada em uma variável pertencente ao banco de dados pessoal do usuário[5]. Na área de edição do *script* (área 2 da Figura 3-20), o terceiro comando (selecionado em verde) demonstra essa funcionalidade. O trecho "`your 'palavra'`" significa que "palavra" é uma variável cujo valor deve ser consultado no BD pessoal antes de se executar o comando, momento em que este passa a ser "`enter 'golfinho' into the 'Pesquisar' textbox`". O resultado da execução deste comando é o preenchimento da caixa de texto correspondente com o valor "golfinho", conforme se vê na área 4 da Figura 3-20. Quando a execução do *script* acontece de forma integrada com os diálogos de mediação, o BD pessoal não é usado e o valor da variável é solicitado diretamente ao usuário (Figura 3-15).

[4] De fato, é dessa motivação que surgiu o nome (Co)Scripter, algo como "roteirizador colaborativo (de tarefas da web)".

[5] Apesar de estarmos nos referindo a duas pessoas (o autor do *script* e o usuário do *script*), nada impede que ambas sejam na verdade o mesmo indivíduo.

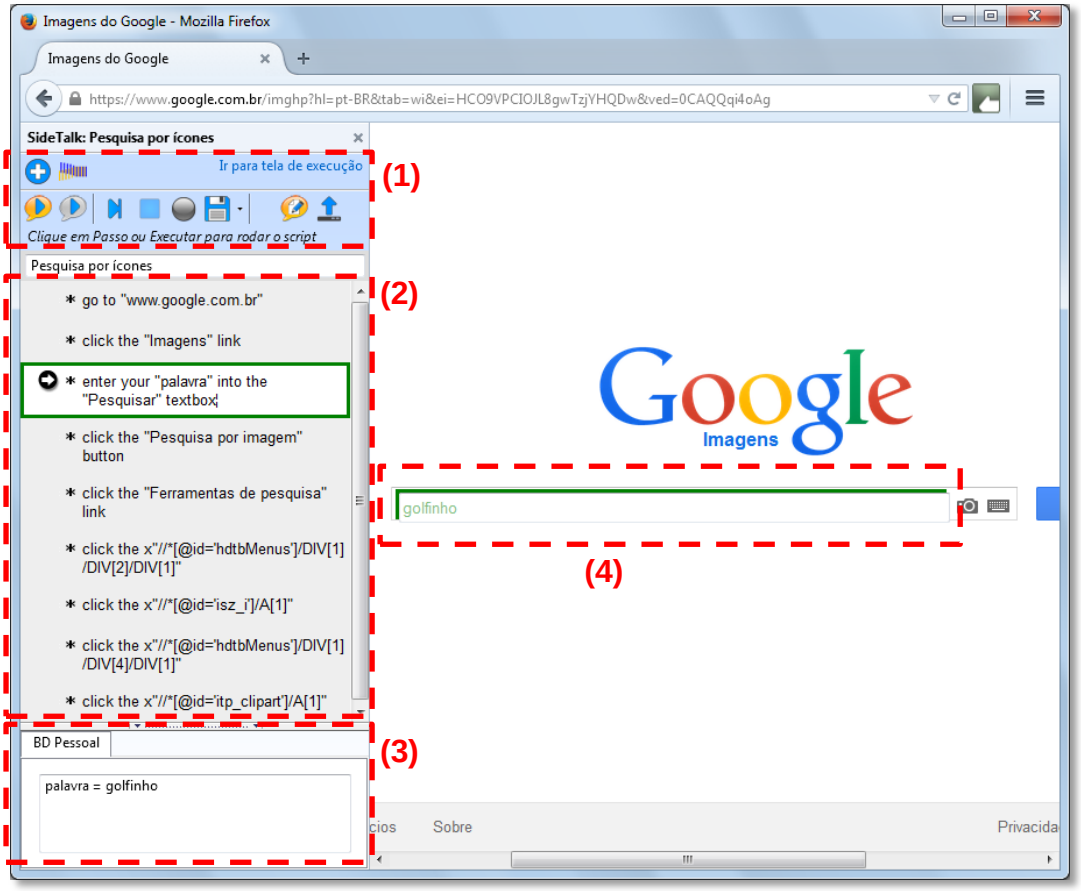


Figura 3-20. Edição do *script* "Pesquisa por ícones"[6]

Uma vez finalizado o *script*, o usuário pode iniciar a criação dos diálogos. O acesso ao editor de diálogos é feito através de um dos botões da barra de ferramentas (área 1). A Figura 3-21 apresenta a tela inicial do editor de diálogos do SideTalk. A área principal da janela traz uma lista numerada com todos os comandos gravados no *script*. A tarefa do usuário é escolher um dos comandos e criar um diálogo relacionado a ele. Isso significa que ao carregar a conversa no SideTalk, quando o *script* alcançar um comando que possui um diálogo, esse diálogo será exibido e após o usuário clicar em "Continuar", o passo será executado (quando for o caso, com o valor da variável preenchido de acordo com o que foi informado pelo usuário).

A Figura 3-22 apresenta a tela de edição do diálogo "O que você está procurando?" (Figura 3-15). Esta janela possui duas partes: 1) um editor HTML para a área principal do diálogo, na qual é possível adicionar textos com as mais diversas formatações, imagens, links etc.; e 2) uma área de formatação do

[6] 1) barra de ferramentas, 2) área de edição do *script*, 3) banco de dados pessoal, 4) caixa de texto associada ao passo selecionado

elemento de entrada (quando for o caso), que permite uma completa customização do item configurado. Por exemplo, é possível adicionar máscaras de formatação à caixa de texto, alterar o número de linhas exibidas; ou até limitar o que pode ser informado na caixa, convertendo a caixa de texto para *listbox* ou *radiobutton*.

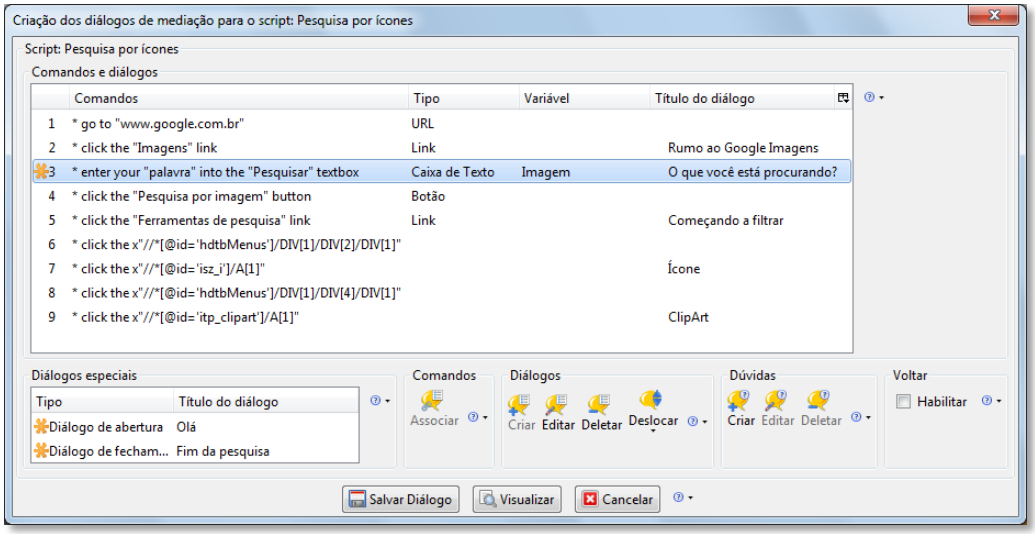


Figura 3-21. Tela inicial do editor de diálogos do SideTalk

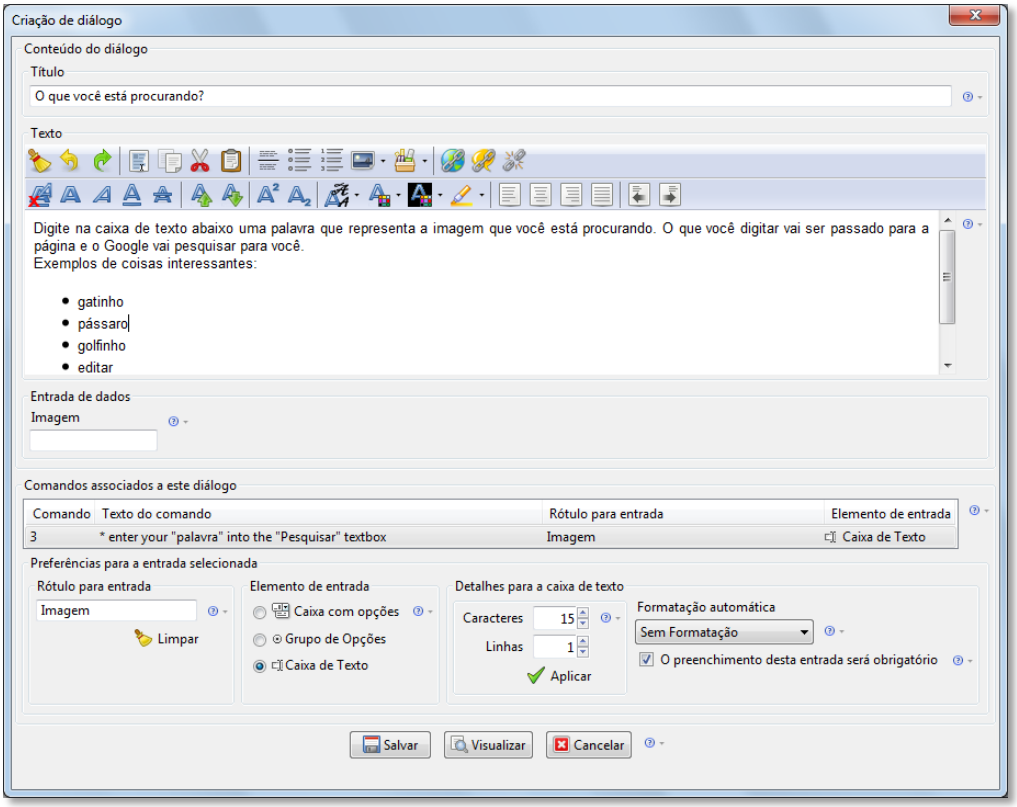


Figura 3-22. Edição do diálogo "O que você está procurando?"

As opções de formatação do editor permitem a construção de um diálogo altamente customizado, apresentando-se como recursos disponíveis para a elaboração da mensagem do *designer*.

3.3.5
Engenharia Semiótica no SideTalk

Apresentamos na seção 2.5 como ocorre a comunicação entre *designer* e usuário segundo a visão da Engenharia Semiótica. Entretanto, a comunicação oferecida pelo SideTalk apresenta uma *mediação* que ocorre em dois níveis: os diálogos realizam a mediação entre o seu criador e seu usuário; o próprio criador da conversa faz uma mediação (através dos diálogos) entre a página e seu usuário. Desta forma, enquanto na interação humano-computador em geral (Figura 2-1) aparecem apenas dois interlocutores, na comunicação com o SideTalk, por sua vez, falamos de quatro participantes: 1) o *designer* da ferramenta SideTalk; 2) o *designer* da(s) página(s) adicionadas em uma determinada conversa; 3) o *designer* da conversa em si (que também é um usuário em relação aos anteriores), responsável por selecionar as páginas, determinar a sequência e outros detalhes de navegação e o conteúdo em si dos diálogos; e, por último, 4) o usuário final, que se encontra no outro lado desta cadeia (Figura 3-23).

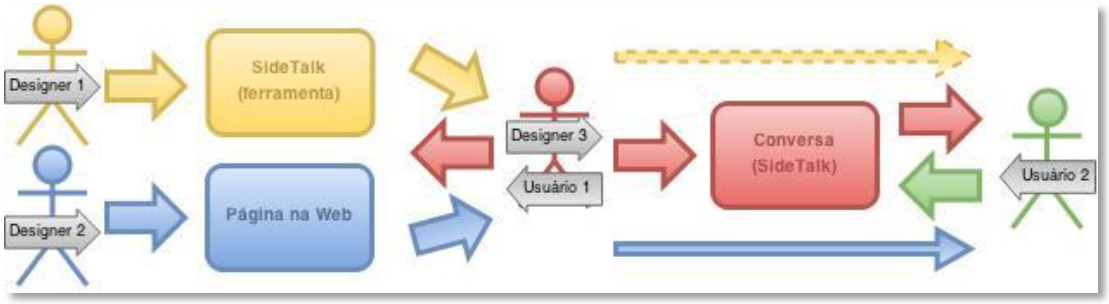


Figura 3-23. Comunicação entre *designers* e usuários no SideTalk

Pela ilustração, percebe-se que a conversa (representada pela caixa vermelha) é criada a partir da interpretação do seu autor (Usuário 1) de duas metacomunicações distintas passadas por duas interfaces bem delimitadas: o SideTalk, enquanto ferramenta de construção (de *scripts* e diálogos) e a(s) página(s) da web que fazem parte da conversa. A partir daí, este mesmo indivíduo, agora em outro papel (Designer 3) ingressa em sua própria engenharia semiótica para estabelecer os signos que comporão uma nova metacomunicação. Outro detalhe importante é que o usuário receptor dos diálogos (Usuário 2) é igualmente usuário do SideTalk, mas desta vez com signos de navegação e não de criação (por isso a seta amarela pontilhada), além de também ser usuário da página (seta azul comprida), pois este usuário tem exatamente o mesmo acesso à página que o Usuário 1, ainda que este procure guiar a navegação daquele. Pela

imagem, a existência de três *designers* revela a polifonia característica da comunicação através do SideTalk. Em outras palavras, a mensagem que chega ao usuário final (Usuário 2) ecoa os enunciados de três vozes.

A Figura 3-24 apresenta uma visualização alternativa deste fenômeno. A área verde representa todo o conjunto de signos disponíveis ao usuário; corresponde à mensagem integral que chega à sua cognição. A área azul representa a página exibida no browser à qual se refere a conversa criada com o SideTalk. Este aparece em amarelo e o diálogo em vermelho. A região em vermelho transparente quer dizer que as decisões de *design* tomadas no SideTalk influenciam a forma de se perceber a área azul de várias formas: visualmente destacando-se elementos da página, direcionando o foco; optando-se por acessar determinados links ou demais elementos em detrimento de outros; textualmente (nos diálogos) resignificando a mensagem original apresentada.[7]

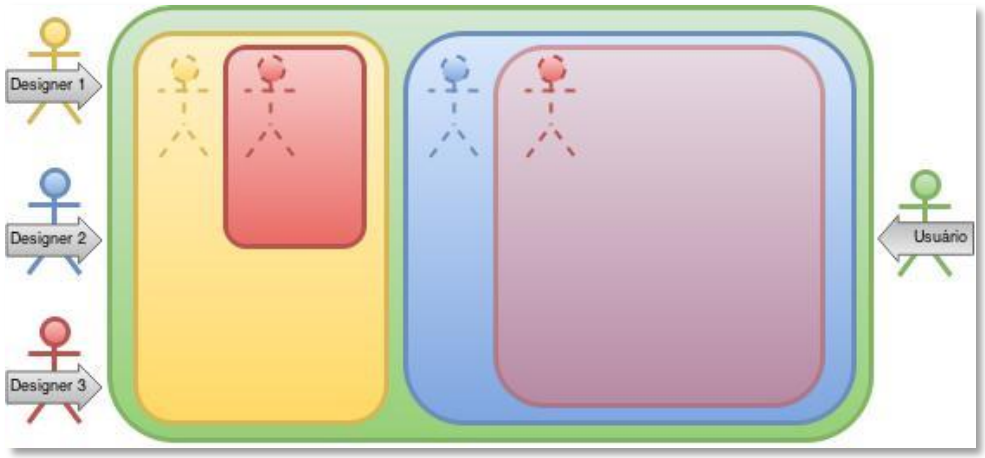


Figura 3-24. Autoria na comunicação com o SideTalk

Os "bonequinhos" tracejados representam o "preposto do *designer*", além de fazer referência ao fenômeno de autoexpressão, já que argumentamos ao longo da tese que as pessoas deixam rastros de suas personalidades quando se comunicam também em discursos digitais.

O próximo capítulo descreve a metodologia adotada na pesquisa, quando, entre outras coisas, detalharemos como cada uma das três ferramentas aqui descrita foi utilizada nos estudos empíricos.

[7] O tamanho das áreas da figura não é significativo em termos de níveis de autoria, apenas reflete o posicionamento da página principal e da barra lateral do SideTalk no navegador.