4
Segunda Edição da MoLIC

Na revisão da primeira edição da MoLIC apresentada no capítulo anterior, diversas questões referentes ao uso da linguagem foram respondidas e deram origem a propostas de extensão à MoLIC. Resta-nos neste capítulo apresentar na íntegra como ficou a segunda edição da MoLIC depois destas discussões. É importante notar que não pretendemos que este texto seja um manual de uso da segunda edição da linguagem. Na verdade, pretendemos apenas que este seja um texto de referência da segunda edição da MoLIC, onde os elementos estejam apresentados com uma semântica bem definida.

O leitor não deve esperar mudanças radicais na segunda edição da MoLIC, nem um grande número de novos elementos do diagrama de interação. De forma resumida, a segunda edição da MoLIC aprimora a semântica dos elementos do diagrama de interação, fornece alguns novos recursos para o detalhamento da interação e dá mais um passo em direção à modelagem de sistemas multi-usuário.

Deste modo, a ferramenta epistêmica MoLIC continua sendo composta por quatro artefatos inter-relacionados: (1) um diagrama de metas, (2) uma ontologia de signos e (3) um diagrama de interação complementado por uma (4) especificação textual, como apresentado no capítulo 3. Os artefatos modificados na segunda edição da MoLIC foram o diagrama de interação e a descrição textual, mas são sugeridos novos requisitos para a ontologia de signos que não faz parte deste trabalho. A Tabela 4 aponta aquilo que foi mantido, incluído e adaptado no diagrama de interação da segunda edição da MoLIC.

Além da modificação de alguns artefatos em si, a segunda edição da MoLIC propõe uma primeira organização, um rascunho, do processo de projeto da interação pela definição de duas etapas no uso da MoLIC, cada qual com foco e abstração bem definidos. Todas as alterações realizadas na MoLIC continuam compromissadas com os fundamentos da primeira edição: a teoria da engenharia semiótica e os princípios de design, apresentados no capítulo 2.2.

A seção 4.1 apresenta o projeto de interação utilizando a segunda edição da MoLIC, descrevendo como o diagrama de interação, entre outros artefatos, pode ser elaborado em duas etapas do projeto de interação, cada qual endereçando um conjunto reduzido de questões de design. A seção 4.1.1 apresenta a primeira etapa do projeto de interação onde o designer estrutura a conversa usuário-preposto. A seção 4.1.2 apresenta a forma como o designer pode detalhar a conversa estruturada na primeira etapa do projeto de interação, no que chamamos de segunda etapa do projeto de interação.

Na seção 4.2, analisamos como os elementos do diagrama de interação se relacionam com o espaço de design de IHC proposto pela engenharia semiótica (seção 2.2.1) para ratificar a correspondência destes elementos com a ontologia desta teoria. Seguindo esta preocupação com a semântica dos elementos do diagrama de interação, a seção 4.3 apresenta o metamodelo do diagrama de interação MoLIC.

Os exemplos apresentados neste capítulo foram baseados em programas de e-mail encontrados no mercado, como, por exemplo, o Eudora® 6.2, o Microsoft Outlook Express® 6.0 e o Mozilla Thunderbird® 1.0.

Elementos do diagrama de interação	**Idêntico à primeira edição**	**Incluído na segunda edição**	**Adaptado na segunda edição**
Cena			*
Diálogos			*
Signos			*
Processo do Sistema		* (novo tipo)	
Fala de Transição			*
Acesso Ubíquo	*		
Ponto de Entrada		*	
Ponto de Saída		*	
Ponto de Contato		*	
Influência da interação do usuário com ator externo		*	

Tabela 4: O que mudou no diagrama de interação da segunda edição da MoLIC.

4.1 Projeto de Interação Humano-Computador

Espera-se que o designer já conheça os elementos envolvidos na metacomunicação designer-usuário – os interlocutores, o canal[1], o código, a mensagem, e o contexto – além dos objetivos (ou metas) de comunicação dos interlocutores antes de começar o projeto de IHC. Com estes conhecimentos adquiridos na análise do problema, o designer será capaz de organizar as metas do usuário no diagrama de metas da MoLIC, caso ainda não as tenha organizado por exemplo, em um modelo de tarefas. Uma vez documentadas e organizadas as metas, será mais fácil fazer a correspondência entre o diagrama de interação e as metas do usuário.

Como dito no capítulo 3, interagir é essencialmente conversar. O projeto de IHC sob esta perspectiva é, na verdade, o projeto da conversa usuário-preposto do designer, tendo como base a metacomunicação designer-usuário mediada pelo sistema. Assim, MoLIC continua se propondo a apoiar o designer na modelagem de todas as possíveis conversas que usuário pode (ou deve) ter com o sistema (preposto do designer) para alcançar seus objetivos. Tal conversa é projetada em um diagrama que privilegia a visão global da interação para facilitar o projeto de um discurso interativo consistente[2] por uma equipe multidisciplinar.

A utilização dos mecanismos de detalhamento da interação fornecido pela segunda edição da MoLIC pode causar dificuldades de entendimento ou de comunicação entre os membros de uma equipe multidisciplinar. Por um lado, é possível que, para alguns profissionais sem formação técnica em Computação, como, por exemplo, psicólogos, sociólogos, artistas e especialistas no domínio da aplicação, esse detalhamento seja desnecessário ou até atrapalhe certos usos do diagrama de interação. Por outro lado, o detalhamento da interação é um importante insumo para o projeto e a implementação da interface, pois algumas decisões ainda precisam ser tomadas para possibilitar a construção da interface.

[1] Conhecer o canal neste caso significa conhecer as características do ambiente computacional (por exemplo, os estilos de interação e as tecnologias disponíveis) no qual a interação se dará, mas não se deve pensar nem representar detalhes de interface na MoLIC.

Caso o designer não tome certas decisões durante o projeto de interação, elas serão tomadas por profissionais sem o conhecimento necessário das questões de IHC envolvidas na solução do problema, que em último caso podem ser tomadas por um programador. Mesmo se não houvesse um desconforto na equipe de projeto, ainda assim seria muito complexo para o designer tratar da interação como um todo (visão geral do discurso interativo consistente), considerando muitos detalhes ao mesmo tempo.

FOCO	ABSTRAÇÃO	QUESTÕES DE PROJETO
Primeira Etapa		
Estrutura de tópicos	Visão global da interação	▪ Conversa usuário-preposto ▪ Recuperação de *breakdowns* ▪ Consistência do discurso interativo (padrões de interação) ▪ Caminhos de interação alternativos
Segunda Etapa		
Detalhamento da conversa	Visão pontual detalhada da interação	▪ Estruturação de diálogos ▪ Restrição no rumo da conversa em função dos diálogos ▪ Definição e estruturação dos signos ▪ Classificação das falas de *breakdown*

Tabela 5: Etapas do projeto de interação na MoLIC.

Seguindo o propósito do projeto baseado em modelos (Hoover et al., 1991), a segunda edição da MoLIC estabelece duas etapas no projeto de interação. A primeira etapa de projeto da interação está focada na estrutura de tópicos da conversa usuário-preposto sob um nível de abstração que favorece a visão global da interação, desconsiderando momentaneamente alguns detalhes da conversa. A segunda etapa do projeto de interação está focada no detalhamento da conversa

[2] Um discurso interativo consistente diz respeito à forma sistemática como a conversa pode se desdobrar. Isto engloba, por exemplo, tratar de modo semelhante trechos semelhantes da conversa e de modo distinto trechos diferentes.

usuário-preposto em um nível de abstração mais baixo, que privilegia uma visão pontual detalhada da interação. A Tabela 5 apresenta um resumo das questões de projeto endereçadas em cada etapa do projeto de interação utilizando a MoLIC.

4.1.1 Primeira Etapa: Estruturação da Conversa

Na primeira etapa do projeto de interação, o designer tem por objetivo especificar todas as possíveis conversas entre o usuário e o sistema (preposto do designer), refletindo sobre questões estruturais da interação como, por exemplo, estrutura de tópicos da conversa usuário-preposto, recuperação de *breakdowns*, consistência do discurso interativo (padrões de interação) e caminhos de interação alternativos. Não cabe nesta etapa entrar em detalhes da conversa, em questões como a definição de signos utilizados na conversa, por exemplo.

Os insumos recebidos da fase de análise − os objetivos dos usuários e a definição de parte dos signos e seus relacionamentos − costumam evoluir à medida que o designer reflete e aprende sobre a interação. O resultado desta primeira etapa deve ser um diagrama de interação ainda sem detalhes da conversa, pois depois de decidido como será a estrutura de tópicos é que a interação será detalhada.

Os elementos básicos do diagrama de interação são:

- cenas;
- processos do sistema;
- falas de transição;
- pontos de acesso;
- correspondência entre o diagrama de metas e de interação; e
- pontos de contato entre diagramas de interação e com sistemas externos.

Cena

Uma *cena* representa uma conversa sobre um determinado assunto, matéria, ou *tópico*. O tópico, na verdade, indica aquilo que o designer considera que o seu preposto e o usuário estarão conversando em determinado momento da interação circunscrito pela cena (Barbosa e da Silva, 2001). O tópico da cena geralmente está associado a uma meta ou a (parte de) uma tarefa do usuário relacionado com o seu objetivo naquela parte da conversa, conforme presumido pelo designer.

Os participantes de uma conversa se alternam como emissor em uma seqüência de turnos de fala organizados em pares conversacionais (Marcuschi, 1999), onde o usuário e o preposto se alternam nos papéis de emissor e receptor. Chamamos de *diálogo* um conjunto de turnos de fala que tratam de um subtópico da conversa. Por exemplo, um tópico "cadastrar cliente" pode ter os subtópicos "informar dados pessoais" e "informar dados profissionais". Assim sendo, a conversa descrita por uma cena é composta por um conjunto de diálogos.

No entanto, algumas vezes um dos participantes detém a maioria dos turnos de fala da conversa; quando, por exemplo, um professor esclarece uma dúvida de um aluno, quando alguém explica um caminho para um visitante ou quando ouvimos um amigo desabafar. Em casos extremos, a conversa pode se resumir em um monólogo, isto é, apenas um participante detém todos os turnos de fala.

No caso do projeto IHC sob a perspectiva mono-usuário, os interlocutores da conversa são o usuário e o preposto do designer. Por um lado, não faz muito sentido existirem conversas onde ocorra um monólogo do usuário ou mesmo uma predominância de turnos de fala do usuário, pelo simples fato de que o usuário necessita de *feedback* do sistema para saber se está sendo ouvido e da maneira esperada. Por outro lado, é razoável existirem conversas onde predominam os turnos de fala do preposto do designer ou até mesmo ocorram monólogos do preposto do designer. Deste modo, o designer deve ser capaz de representar não somente cenas com diálogos de turnos de fala equilibrados, mas também cenas onde predominam os turnos de fala do preposto do designer e onde ocorra um monólogo do preposto.

Uma cena é representada minimamente no diagrama de interação por um retângulo com bordas arredondadas e uma identificação do seu tópico (Figura

47a). O tópico deve ser expresso por um verbo no modo infinitivo sob o ponto de vista do usuário. Se a cena corresponder a um monólogo do preposto, a mensagem do designer com conteúdo extensional ou os signos que compõem sua mensagem com conteúdo intensional devem estar representados num compartimento abaixo do tópico (Figura 47b). No caso mais comum, a cena corresponderá a diálogos de turnos de fala equilibrados. Então, oculta-se o compartimento da mensagem do designer e representa-se apenas os diálogos entre colchetes no seu compartimento logo abaixo do tópico (Figura 47c). Por fim, se a cena corresponder a uma conversa em que predominam os turnos de fala do preposto e o usuário puder travar diálogos que ajustem o código, a sintaxe e o foco da falas do preposto, a mensagem do designer é representada no compartimento logo abaixo do tópico e em seguida os diálogos são representados em seu compartimento (Figura 47d).

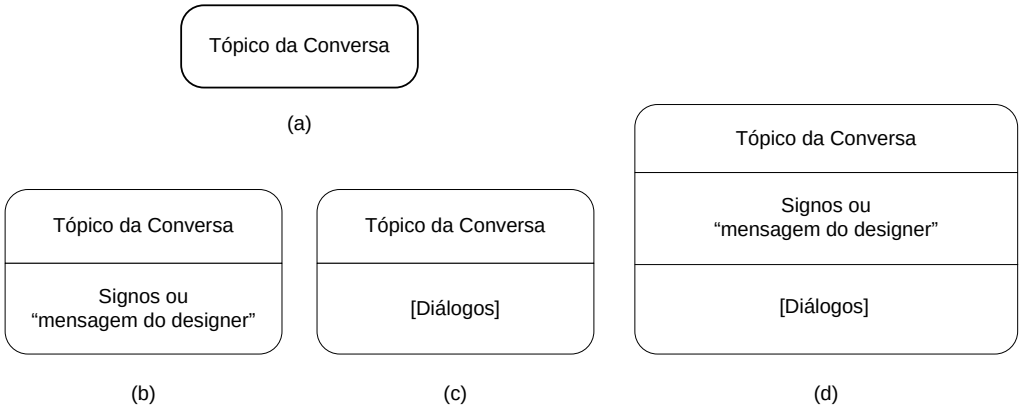


Figura 47: Representação gráfica da cena no diagrama de interação.

A Figura 48 apresenta exemplos de cenas que podem ser encontradas em típicos programas de e-mail: (Figura 48a) a cena de tópico *Examinar caixa de entrada* representada em sua forma mínima, (Figura 48b) a cena *Examinar e-mail recebido* que apresenta uma mensagem do designer com conteúdo intensional indicada pelo signo composto *e-mail*, (Figura 48c) a cena *Escrever e-mail* onde o usuário trava o diálogo *[informar os dados do e-mail]* para depois confirmar o envio, e, por fim, (Figura 48d) a cena *Examinar caixa de entrada* onde o designer apresenta os cabeçalhos dos e-mails recebidos (mensagem do designer com conteúdo intensional representada pelo signo *conjunto(cabeçalho.*)!*) e o usuário trava diálogos com intuito de ajustar o código (por exemplo, apresentar um e-mail como lido), a sintaxe (por exemplo, definição dos atributos do e-mail que compõe

o cabeçalho) e o foco (por exemplo, o conjunto de e-mails sendo exibido) da mensagem que o preposto do designer apresentou.

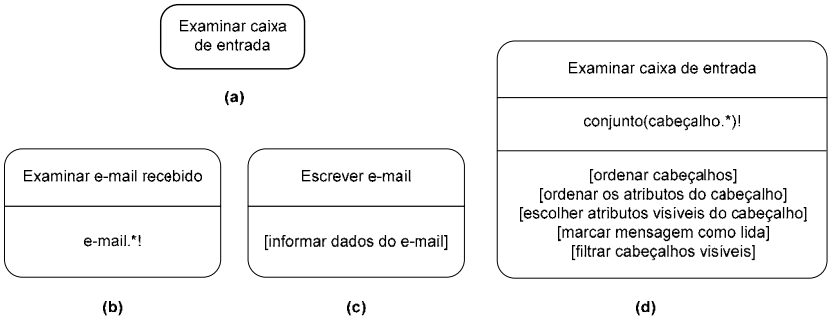


Figura 48: Exemplos de cena em programas de e-mail.

Num próximo passo do desenvolvimento, quando formos construir o modelo de interface, as condições necessárias para a conversa descrita em cena ocorrer poderão ser implementadas em um componente estruturado, tal como uma tela, janela ou página web. Desta forma, um componente estruturado de interface servirá de palco para a conversa descrita em uma cena.

Processos do Sistema

Um processo do sistema representa um momento na interação onde o sistema está "pensando" sobre o que o usuário falou e, conforme o resultado do processamento, o próximo tópico da conversa (a próxima cena) será determinado. No diagrama de interação, um processo do sistema é representado graficamente por uma "caixa preta" (Figura 49a). Usou-se a noção de "caixa preta" pois o usuário não percebe o que está ocorrendo em um processo do sistema até que o processo termine e o preposto do designer comunique seu resultado. Esta forma de representação visa motivar o projeto cuidadoso das falas do preposto do designer emitidas após a conclusão de cada processamento, como *única* forma de comunicar ao usuário o que ocorreu no processamento, suas conseqüências, e possivelmente uma explicação sobre o que tenha ocorrido.

Se o processo do sistema for demorado, não é conveniente que o usuário fique esperando o término do processamento para obter algum retorno do preposto do designer. Nestes casos, é mais conveniente que durante o processamento o preposto do designer comunique ao usuário índices de evolução do processo do

sistema ou mesmo os seus resultados parciais. Além disto, em algumas situações é desejável que o usuário interaja com o sistema durante um processo do sistema para influenciar o próprio processamento ou fazer uso dos resultados parciais.

Como qualquer interação entre o usuário e preposto é descrita em uma cena no diagrama de interação, a interação usuário-preposto durante um processo do sistema é representada por uma cena inter-relacionada com determinado processo do sistema. Esta inter-relação é representada graficamente por duas linhas paralelas que unem o processo do sistema e a cena em questão (a Figura 49b, por exemplo, representa a interação do usuário com o sistema considerando parte dos e-mails recebidos, mesmo antes de o sistema completar a obtenção dos e-mails no servidor). Optou-se pelo uso de duas linhas paralelas para fazer alusão a um canal por onde passam falas do preposto, comunicando a influência da evolução do processo sobre a cena, e falas do usuário, comunicando a influência da conversa ocorrida na cena sobre o processo do sistema. Este canal possui linha de largura maior do que as setas para indicar maior intensidade na interinfluência entre um processo do sistema e uma cena.

Um caso particular da interação usuário-preposto durante um processo do sistema ocorre quando o preposto apenas apresenta ao usuário um índice da evolução do processamento e o usuário pode apenas cancelar (abortar, desistir) o processo do sistema emitindo uma fala de transição. Para este caso particular onde não existam diálogos entre o usuário e o preposto sobre o andamento do processo do sistema, podemos utilizar uma representação de “caixa branca” acoplada ao processo (a Figura 49c, por exemplo, apresenta uma “caixa branca” que comunica ao usuário o andamento da obtenção de e-mail no servidor, sem, no entanto, apresentar e permitir a interação com os resultados parciais). Nela estão descritos os índices de evolução do processo do sistema comunicados pelo preposto, e dela saem as falas de transição do usuário cancelando o processo do sistema.

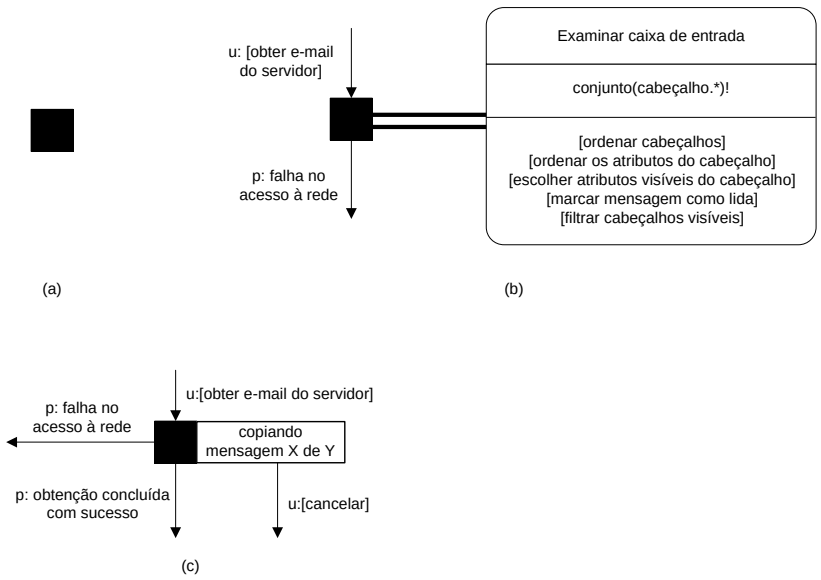


Figura 49: Representação gráfica do processo do sistema no diagrama de interação.

Falas de Transição

Uma fala de transição transmite uma mensagem que manifesta (marca) a mudança de tópico da conversa, ou seja, ela comunica uma mensagem com o objetivo de sair de uma cena e chegar em outra cena, diferente da primeira. Representa-se graficamente uma fala de transição no diagrama de interação por uma seta preta indicando a direção da transição e um rótulo.

Em qualquer momento da interação descrita por uma cena, o usuário pode manifestar seu desejo de mudar o tópico da conversa emitindo uma fala de transição. No caso mais simples, uma fala de transição do usuário o leva imediatamente para a cena de destino, sem que o preposto tenha algo a dizer a respeito e o próximo tópico da conversa não dependa do resultado de um processo de sistema (Figura 50a). No caso mais comum, uma fala de transição do usuário faz com que o sistema "pense" (durante um processo do sistema) sobre o que o usuário falou e determine o próximo tópico da conversa (isto é, a próxima cena ou acesso ubíquo) através de uma fala de transição do preposto que comunica ao usuário o resultado do processo do sistema (ou seja, daquilo que o sistema "pensou") (Figura 50b). Nestes casos, cabe ao designer prever e representar no diagrama de interação os possíveis rumos da conversa em resultado a um processo do sistema.

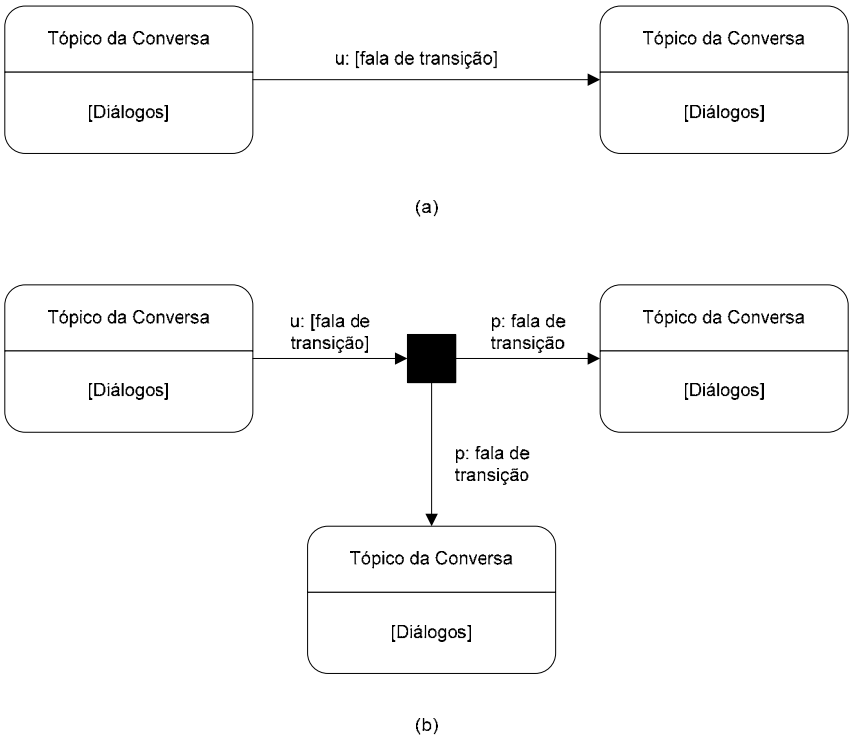


Figura 50: Representação gráfica da mudança de tópico no diagrama de interação.

Durante um processo do sistema, o preposto do designer pode não "compreender" a fala de transição do usuário (i.e., ser capaz de processar ou atribuir um comportamento ao sistema). Tal situação é geralmente causada por uma ruptura de comunicação com origem nos diálogos travados na cena de onde saiu a fala de transição do usuário (por exemplo, o fornecimento de valores inválidos a um ou mais signos). Assim, algumas das falas de transição do preposto em resposta ao processo do sistema correspondem ao esforço do designer para reparar rupturas de comunicação, também chamadas de *breakdowns*. O destino de uma fala de transição de reparo de *breakdown* pode ser a mesma cena de origem da fala de transição do usuário que iniciou o processo do sistema ou uma outra cena (ou acesso ubíquo) qualquer, dependendo de como o designer deseja apoiar o usuário na retomada da conversa em direção aos seus objetivos.

O rótulo de uma transição é composto por três partes:

1. *pressuposição*: condições do contexto da conversa que devem ser satisfeitas para que o usuário ou o preposto possa enunciar a fala de transição correspondente. A descrição da pressuposição é precedida pela expressão *pré:* e normalmente é expressa em linguagem natural.

Por exemplo, *pré: o usuário deve ter permissão para alterar a senha*;

2. *uma ou mais falas do usuário ou do preposto do* designer. No caso do usuário, trata-se de fala(s) com o objetivo de modificar o tópico da conversa ou concluir a conversa. As falas do usuário são descritas no formato *u:[fala do usuário]*. Por exemplo, *u:[confirmar alteração]*. No caso do preposto, trata-se de fala(s) que, ao revelar o(s) resultado(s) de um processo, determinam o próximo tópico da conversa. As falas do preposto são descritas no formato *p:fala do preposto*. Por exemplo, *p:login inexistente*; e
3. *implicatura*: estabelecimento ou modificação dos elementos que caracterizam o contexto da conversa. A descrição da implicatura é precedido pela expressão *pós:*. Por exemplo, *pós: login efetuado*.

A Figura 51 apresenta as formas de representação das falas de transição. Uma fala de transição normal (sem reparo de *breakdown*) é representada por uma seta preta de linha cheia, já a uma fala de transição de reparo de *breakdown* é representada por uma seta preta de linha tracejada. Ambas são descritas por um rótulo com a própria fala e que pode incluir as pressuposições e implicaturas quando necessário.

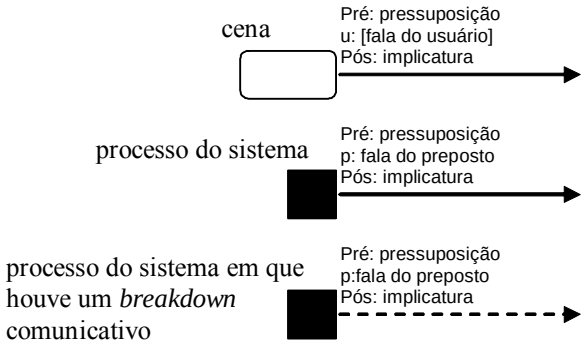


Figura 51: Representação gráfica de falas de transição no diagrama de interação.

Quando a fala de transição fizer referência a algum signo envolvido na conversa na cena de origem, especialmente se for parte de um signo composto, a sua descrição deve deixar clara tal referência. Isto é importante para que no futuro a interface permita enunciar a fala de transição com os signos apropriados (sejam

eles selecionados, marcados, descritos, etc na interface), como, por exemplo, selecionar *um subconjunto de documentos* e depois remover, ou copiar as propriedades de *um objeto* e aplicar sobre *outros objetos* indicados. Uma maneira de deixar mais claro de que signos, partes de signos, ou (sub)conjuntos de signos se está falando durante uma transição seria utilizar descrições das seguintes formas:

Modelo	**Exemplo**
u: "["<verbo (voz ativa)> <signo-composto> <Token> "]"	u:[imprimir o documento D]
u: "["<verbo (voz ativa)> <parte-do-signo> de <signo-composto> <Token> "]"	u:[modificar senha de U]
u: "["<verbo (voz ativa)> < conjunto-de-signo-composto > <subconjunto-de-tokens >[+|*|{m..n}] <restrição>]" * indica zero ou mais tokens + indica um ou mais tokens {m..n} indica números mínimos (m) e máximo (n) de tokens	u:[remover documentos D^+ selecionados] u:[arquivar mensagens M* anteriores à data D]
p: <signo-composto> (<Token>) <verbo (voz passiva)>	p: documento D enviado para a impressora p: documento enviado para a impressora
p: < parte-do-signo > (de <signo-composto>) (<Token>) [<verbo (voz passiva)> | <adjetivo>]	p: CPF inválido
p: < conjunto-signo-composto > <subconjunto-de-tokens > (<restrição>)[<verbo (voz passiva)> | <adjetivo]	p: documentos D removidos p: mensagens anteriores a D arquivadas

Tabela 6: Algumas formas de referenciar signos na fala de transição.

onde termos iniciados com letra maiúscula indicam valores de um *signo da cena de origem* sendo referenciados (semelhante aos possíveis objetos de uma classe).

A Figura 52 apresenta exemplos de transições comuns em programas de e-mail. Quando um usuário decide examinar o e-mail com determinado cabeçalho, ele emite uma fala de transição que sai da cena *Examinar caixa de entrada* e chega a cena *Examinar e-mail* (Figura 52a) sem interceptação de um processo do sistema, pois o designer optou por não tratar erros neste caso. Repare que a descrição desta fala de transição discrimina que um signo cabeçalho está envolvido (*u:[examinar e-mail de cabeçalho C]*). Uma situação um pouco diferente ocorre quando o usuário deseja confirmar o envio do e-mail após ter

informado os dados necessários (Figura 52b). Neste caso, ele emite uma fala de transição *u:[enviar e-mail]* que sai da cena *Escrever e-mail* e é interceptada por um processo do sistema que verifica se os dados do e-mail foram fornecidos corretamente. Caso os dados sejam fornecidos corretamente, o preposto do designer emite uma fala de transição *p: e-mail enviado com sucesso,* que tanto continuará a conversa na cena *Examinar caixa de entrada* quanto afetará a interação do receptor. (Esta bifurcação será tratada mais adiante, quando falarmos de pontos de contato. Por enquanto basta saber que de alguma forma a conversa continua na cena *Examinar caixa de entrada*.) Se em outro caso o processo do sistema encontrar algum signo não obrigatório sem valor definido, mas que normalmente possui valor, o preposto do designer emite a fala de transição de reparo de *breakdown p: dados X sem valor definido* que chega na cena *Confirmar envio*. Note que esta fala de transição discrimina os signos sem valor definido provenientes da cena *Escrever e-mail* e, por se tratar de uma fala de reparo de *breakdown*, a seta é tracejada. Na cena *Confirmar envio*, o usuário pode desistir do envio (fala de transição *u:[desistir do envio]*) ou efetivar o envio mesmo com signos não obrigatórios sem valor definidos (fala de transição *u:[enviar e-mail]*).

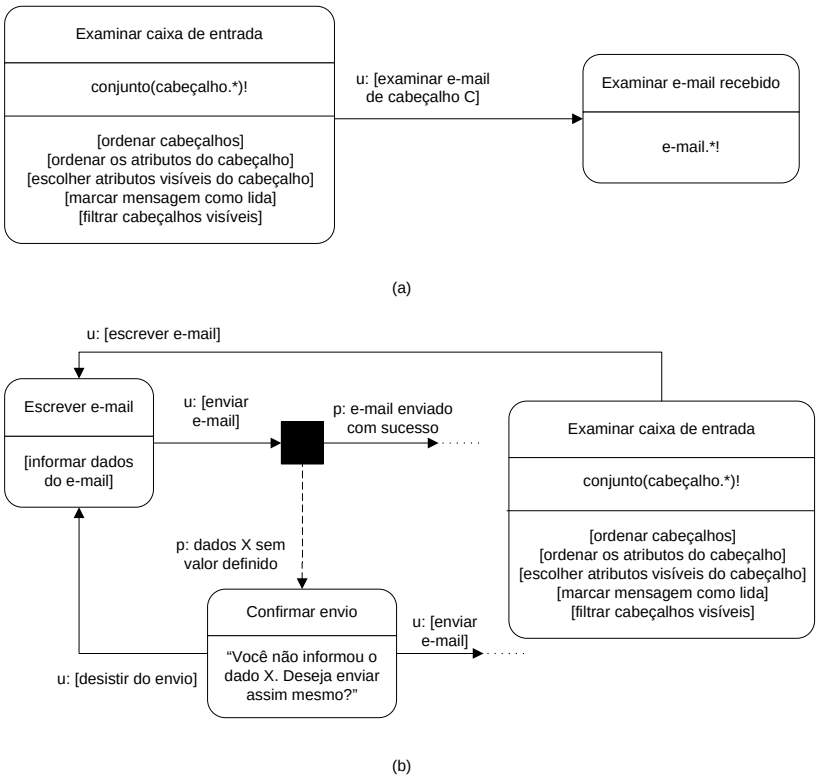


Figura 52: Exemplo de mudança de tópico em programas de e-mail.

Pontos de Acesso

Um ponto de acesso é um elemento do diagrama de interação que pode servir de origem (ou destino) para falas de transição que vão para (ou chegam em) uma cena, um processo do sistema, ou mesmo outro ponto de acesso. Alguns possuem semântica própria, indicando, por exemplo, o início da interação. Outros, porém, são representações mais simples que substituem outros elementos do diagrama com determinado objetivo. Existem três tipos de pontos de acesso: acesso ubíquo, ponto de entrada e ponto de saída.

Acesso Ubíquo

Algumas cenas podem ser o destino de falas de transição provenientes de qualquer outra cena, ou seja, não importa o tópico atual da conversa durante a interação, o usuário pode decidir mudar a conversa para um determinado tópico. Para não representarmos uma fala de transição saindo de cada cena do diagrama de interação e chagando nessas cenas "ubíquas", decidimos definir um ponto de acesso chamado *acesso ubíquo* para simplificar o diagrama de interação. Então, um acesso ubíquo representa qualquer cena do diagrama de interação de onde o usuário pode sair e para onde ele e o preposto podem voltar através de falas de transição. Por alusão à cena, um acesso ubíquo é representado graficamente por um retângulo cinza com bordas arredondadas descrito pela letra U de "ubíqua" seguida de um número (Figura 53a). Na maioria das vezes um acesso ubíquo alcança diretamente uma cena (Figura 53b), mas ele também pode alcançar um processo do sistema (Figura 53c).

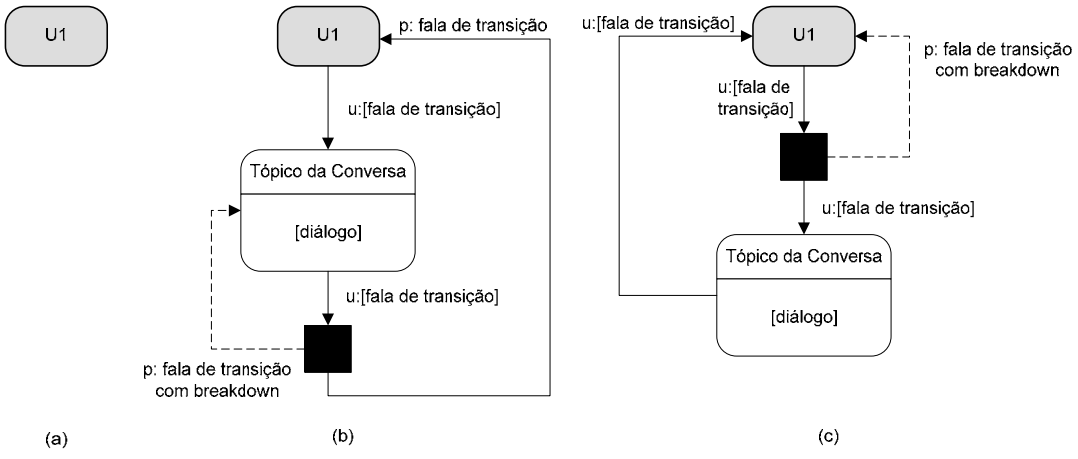


Figura 53: Representação gráfica de acesso ubíquo no diagrama de interação.

Pensando novamente em programas de e-mail, é desejável que o usuário possa escrever e enviar um e-mail de qualquer lugar do sistema, e não somente a partir da cena *Examinar caixa de entrada,* conforme representado na Figura 52b. Esta idéia é simplesmente representada por uma fala de transição que sai de um acesso ubíquo e chega a cena *Escrever e-mail* e por todas as falas de transição que voltam ao mesmo acesso ubíquo concluindo de alguma forma este caminho de interação (Figura 54).

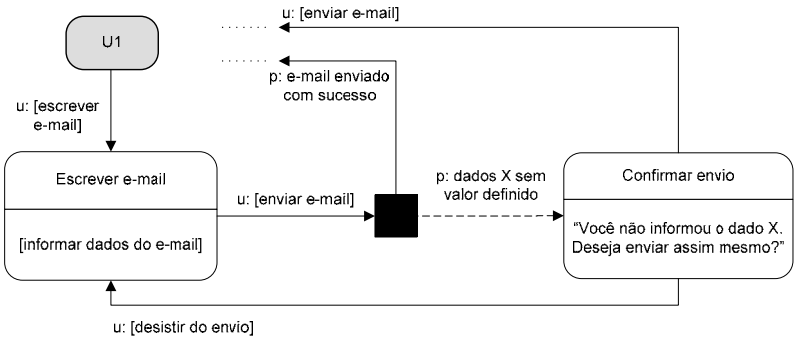


Figura 54: Exemplo de acesso ubíquo em programa de e-mail.

Pontos de Entrada e de Saída

O ponto de entrada representa o momento onde se inicia a conversa usuário-preposto, isto é, a interação do usuário com o sistema. A ação que marca o início da conversa é uma fala do usuário que sai do ponto de entrada em direção a uma cena. Por oposição, o ponto de saída representa o momento onde se encerra a interação. O ponto de saída também é atingido por uma fala de transição geralmente emitida pelo usuário, mas que também pode ser emitida pelo preposto. A representação gráfica do ponto de entrada e do ponto de saída é respectivamente a mesma do estado inicial e final de Statecharts (Harel, 1987), como apresentado na Figura 55. Em programas de e-mail, é comum a fala de transição que sai do ponto de entrada chegar na cena *Examinar caixa de entrada* e o ponto de saída ser acessível de qualquer lugar do sistema (através de um acesso ubíquo).

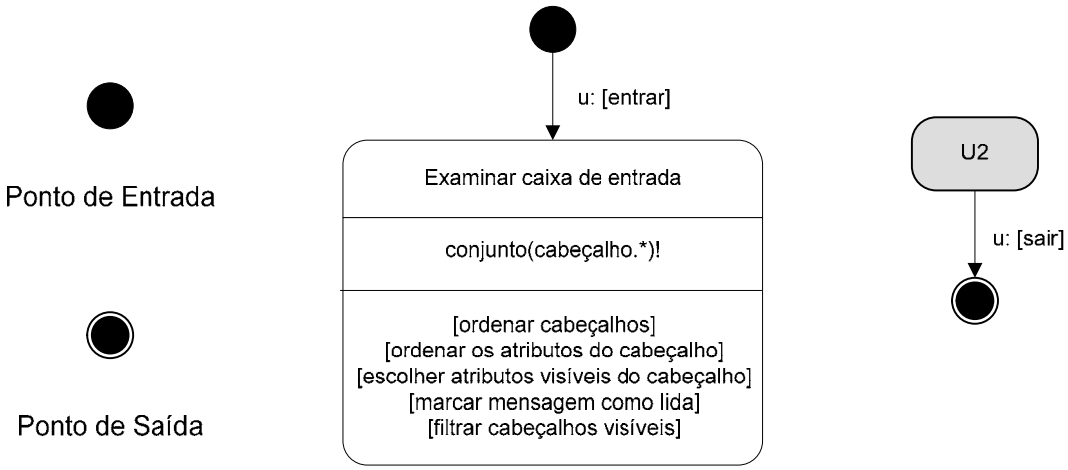


Figura 55: Representação gráfica dos pontos de entrada e de saída no diagrama de interação.

Correspondência entre o Diagrama de Metas e de Interação

Podemos obter a correspondência entre elementos do diagrama de interação e as metas do usuário circunscrevendo com um retângulo cinza os caminhos de interação necessários para que o usuário alcance sua meta (Figura 56). Cada retângulo é identificado pela meta do usuário que pode ser alcançada com a interação representada dentro dele. Na Figura 57, por exemplo, um retângulo delimita os possíveis caminhos de interação para que a meta *enviar e-mail* seja alcançada em um programa de e-mail.

Figura 56: Correspondência entre uma meta do usuário e parte do diagrama de interação.

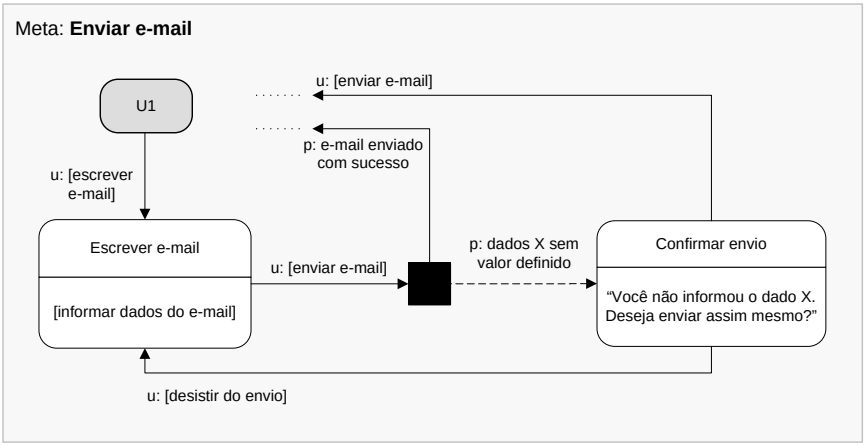


Figura 57: Exemplo de correspondência da meta *Enviar e-mail* e a respectiva parte do diagrama de interação.

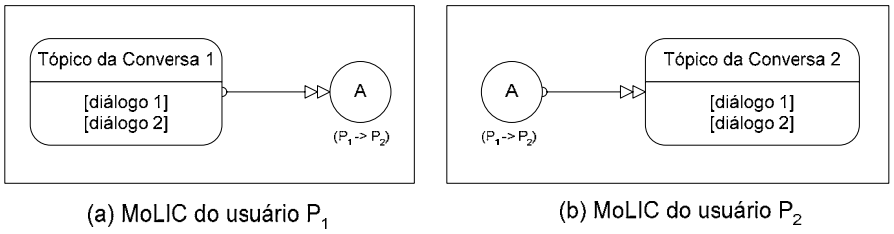
Esta delimitação da meta do usuário no diagrama de interação permite algum grau de rastreabilidade entre os diagramas, facilita a avaliação do impacto de mudanças corretivas ou evolutivas que possam ocorrer, bem como facilita a manutenção da consistência entre os diagramas, a cada nova versão do sistema.

Pontos de Contato entre Diagramas de Interação

Alguns diálogos travados durante a interação usuário-sistema têm como interlocutor, direto ou indireto, um ator externo ao contexto imediato de interação. Quando este interlocutor externo interage ("ouve" ou "fala") com o usuário através do sistema sendo projetado, é preciso que o designer projete a influência da interação de um usuário sobre a interação de outro usuário (externo ao contexto de interação representado) e vice-versa. Deste modo, o designer deve elaborar um diagrama de interação MoLIC para cada ator ou papel de usuário envolvido na interação (considerando um modelo de usuário como, por exemplo, o proposto por Prates (1998)). Em cada diagrama será projetada a interação usuário-sistema sob o respectivo ponto de vista, levando o designer a refletir sobre a interação de cada (papel de) usuário com o sistema separadamente.

Depois de construídos os diferentes diagramas de interação MoLIC para cada usuário, utilizam-se *pontos de contato* entre eles para relacioná-los. Um ponto de contato representa um canal que transmite a influência da interação de um usuário (ou do seu preposto) para a interação de outro(s) usuário(s) externo(s) ao contexto imediato de interação, e vice-versa. Representa-se graficamente um

ponto de contato por um círculo com um rótulo de onde chegam e saem setas duplas representando as influências da interação usuário-ator(es) externo(s). A Figura 58 apresenta o projeto da influência da interação do usuário P_1 sobre o usuário P_2 . No diagrama de interação do usuário P_1 (Figura 58a), a conversa descrita pela cena *Tópico da Conversa 1* influencia o usuário P_2 através do ponto de contato *A*. Já no diagrama de interação do usuário P_2 (Figura 58b) chega uma influência do usuário P_1 através do ponto de contato *A* que afeta a conversa descrita pela cena *Tópico da Conversa 2*.



(a) MoLIC do usuário P_1 (b) MoLIC do usuário P_2

Figura 58: Representação gráfica de pontos de contato entre diagramas de interação (adaptado de Silva e Barbosa, 2004).

É importante notar que as setas associadas a pontos de contato *não* são falas de transição. Na verdade estas setas representam a influência da interação de um usuário sobre um ator externo e vice-versa, sem, no entanto, mudar o tópico da conversa. Uma influência da interação é representada por uma seta preta com um rótulo, cuja origem terá um círculo vazado e o destino uma seta dupla vazada, conforme apresentado no lado esquerdo da Figura 59. Esta representação ainda não diferencia influências síncronas das influências assíncronas.

Caso a influência da interação seja causada por um *breakdown*, a linha da seta será tracejada como na fala de transição de reparo de *breakdown*. Em situações onde é necessário conjugar a mudança de tópico da conversa e a influência da interação sobre um ator externo, uma fala do usuário ou do preposto chega em uma barra preta (semelhante à transição *fork* do diagrama de atividades na UML) e se bifurca em uma fala de transição e uma influência da interação, conforme apresentado no lado direito da Figura 59.

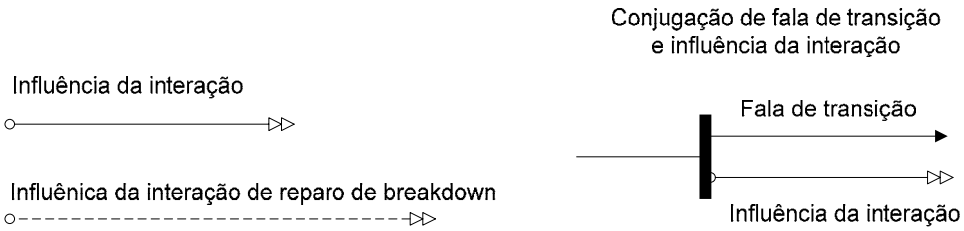
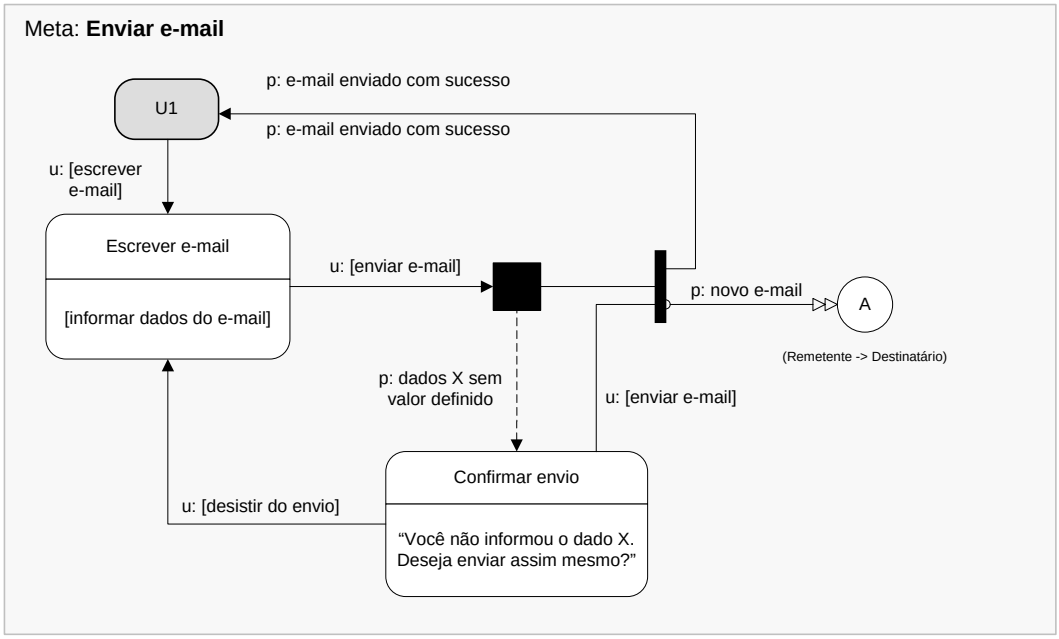
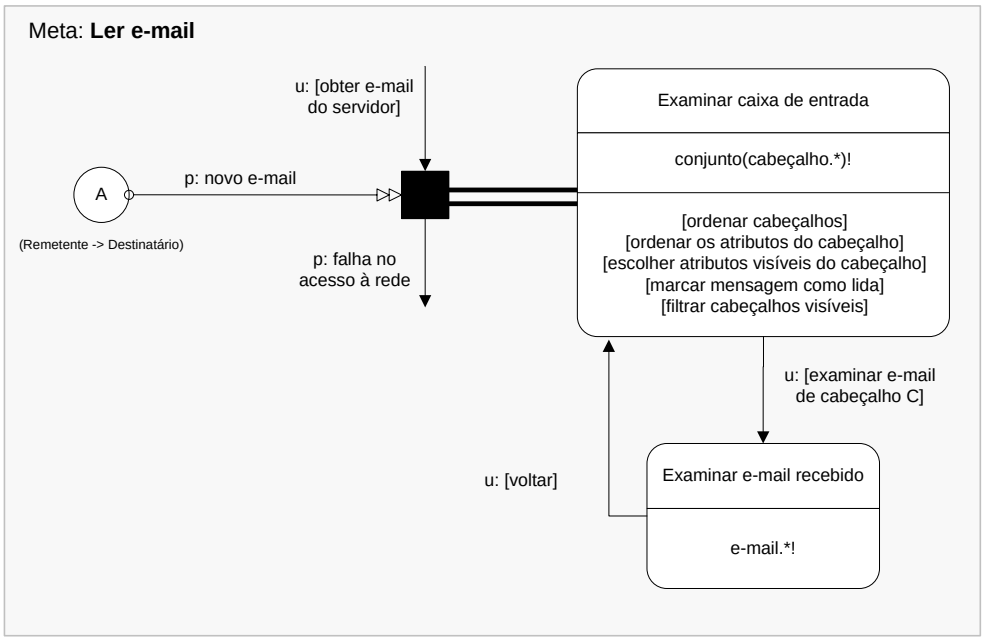


Figura 59: Representação gráfica da influência da interação.

Para facilitar a leitura dos modelos, sugere-se indicar os (papéis de) usuários envolvidos nos pontos de contato, como uma referência ao MoLIC que conterá o ponto de contato correspondente (de origem ou de destino da fala). Por exemplo, se voltarmos à Figura 58, vamos notar que junto ao ponto de contato *A* estão indicados os usuários envolvidos: ($P_1 \rightarrow P_2$), lê-se P_1 interage com P_2 através do sistema. A Figura 60 apresenta um exemplo prático do uso de pontos de contato em um programa de e-mail. Ela representa como a interação do remetente com seu sistema (Figura 60a) influencia a interação do destinatário com a sua respectiva instância do sistema (Figura 60b).



(a) MoLIC do remetente no programa de e-mail



(b) MoLIC do destinatário no programa de e-mail

Figura 60: Exemplo da influência da interação do remetente sobre interação do destinatário do e-mail.

Se as conseqüências da influência da interação do usuário sobre algum ator externo não fizerem parte do sistema sendo projetado, o designer deve usar um ponto de contato com sistema externo para representar essa influência. Um ponto de contado com sistema externo é representado graficamente por um círculo metade branco e metade preto (Figura 61). O lado branco do círculo representa

que o usuário, o preposto e o designer conhecem (algo sobre) a entrada do canal de comunicação (ponto de contato) e aquilo que é transmitido por ele. Já o lado preto do círculo representa o desconhecimento de (exatamente) como/onde é saída deste canal de comunicação e, principalmente, das conseqüências das falas recebidas através dele. Em termos de conversa, esta situação equivale ao caso onde o usuário pode "falar" com algum ator externo através do sistema, sem, no entanto, "ouvi-lo". Por exemplo, o usuário pode editar um documento no Microsoft Word® e depois enviá-lo pelo próprio Word para um programa de e-mail transmiti-lo como anexo para alguém. Neste exemplo, a interação usuário-Word influencia ("fala" com) um programa de e-mail sem esperar resposta através do próprio Word, ou seja, sem ser diretamente influenciada ("ouvir") pelas conseqüências do envio de e-mail.

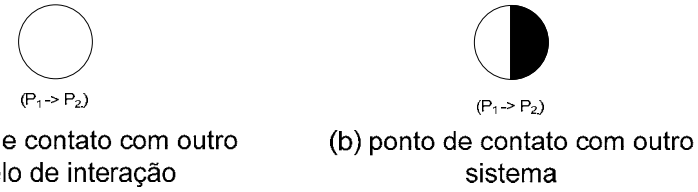


Figura 61: Representação de ponto de contato (a) com outro modelo de interação e (b) com outro sistema.

4.1.2 Segunda Etapa: Detalhamento da Conversa

Na segunda etapa do projeto de interação, o designer concentra esforços no detalhamento da interação, isto é, na especificação dos diálogos e na classificação das falas de transição de reparo de *breakdown*. A especificação dos diálogos envolve (1) a estruturação dos diálogos dentro da cena, (2) as possíveis restrições sobre o rumo da conversa (falas de transição que pressupõem um conjunto diálogos travados na cena de origem), (3) a definição e estruturação dos signos envolvidos nos diálogos, e, por fim, (4) o detalhamento da ontologia de signos.

Conforme o designer detalha a interação, ele acaba refinando o diagrama de interação, definindo mais completamente a ontologia de signos e elaborando uma descrição textual para registrar os detalhes da interação. Estas informações adicionais do projeto de interação servirão de insumo para o projeto e a implementação da interface em etapas futuras do processo de desenvolvimento.

Estrutura de diálogos

A representação padrão dos diálogos dentro do respectivo compartimento na cena não implica em uma ordem pré-estabelecida para o travamento dos diálogos. Na Figura 62, por exemplo, bem como em todos os exemplos da etapa anterior, não existe indicação da ordem em os diálogos serão travados. A seqüência de apresentação dos diálogos dentro desta cena no máximo pode sugerir a ordem em que a interface disponibilizará ao usuário as condições necessárias à ocorrência destes diálogos. Logo, os diálogos desta cena podem ser travados na ordem <1,2,3,4>, <2,1,3,4> ou outra qualquer, dependendo da vontade do usuário.

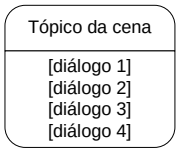


Figura 62: Representação de uma cena na primeira etapa do projeto de IHC.

A forma de representação dos diálogos utilizada na Figura 62 equivale à forma de estruturação mais simples dos diálogos: um grupo onde todos os diálogos podem ser travados em qualquer ordem, ou mesmo ignorados (quando o usuário aceita os valores default sugeridos pelo designer). A forma explícita de agrupar os diálogos independentes de ordem é utilizando o operador *group* antes de um conjunto de diálogos entre colchetes:

```
group {
    [diálogo 1]
    [diálogo 2]
}
```

Ou simplesmente

```
[diálogo 1]
[diálogo 2]
```

O designer pode querer representar explicitamente algum tipo de informação sobre o grupo de diálogos, tais como critério para realizar o agrupamento ou condições do contexto em que os diálogos poderão ocorrer. Na segunda edição da MoLIC, estas informações são fornecidas em texto livre entre parênteses, ao lado da palavra-chave `group` :

```
group (critérios de agrupamento ou condições do contexto) {
   [diálogo 1]
   [diálogo 2]
}
```

Por exemplo:

```
group (pelo menos 2 diálogos travados) {
   [diálogo 1]
   [diálogo 2]
   [diálogo 3]
}

group (usuário é administrador) {
   [diálogo 1]
   [diálogo 2]
   [diálogo 3]
}
```

Quando pensamos que os diálogos de uma cena sem ordem pré-definida poderão ocorrer em interfaces de diferentes plataformas, lembramo-nos de que eles podem ser reorganizados livremente quando as unidades de apresentação (por exemplo, telas, janelas e páginas web) de cada plataforma forem definidas/construídas. Nestes casos, o operador *group* será bastante útil para indicar conjuntos mínimos de diálogos que devem ocorrer em uma única unidade de apresentação.

Ordenação de diálogos

Em alguns casos, no entanto, é importante que o designer informe a ordem em que os diálogos devem ser travados. O propósito de um *Wizard*, por exemplo, é guiar um usuário por uma seqüência de diálogos até alcançar um determinado objetivo, seja ele criar um currículo, criar uma classe Java, configurar os dados da uma conta de e-mail ou outro qualquer. Além de conduzir a interação do usuário, a seqüência de travamento dos diálogos também pode ser importante quando um diálogo depende do outro, isto é, quando os signos envolvidos num diálogo são determinados (ou restringidos) pelos signos envolvidos em outro(s) diálogo(s). Por exemplo, um diálogo onde o usuário informa sua cidade pode depender do diálogo em que ele informou o seu estado.

Quando a seqüência de travamento dos diálogos for importante, o designer pode agrupá-los em um conjunto de diálogos precedido pelo operador *seq*. Por exemplo, o *diálogo 1* e o *diálogo 2* foram agrupados entre colchetes precedidos

pelo operador *seq* para indicar que o *diálogo 2* só pode ser travado depois de ter sido travado o *diálogo 1*:

```
seq {
   [diálogo 1]
   [diálogo 2]
}
```

Vale lembrar que a seqüência dos diálogos não implica necessariamente a determinação (ou dependência) entre os signos envolvidos. Um signo A é dito determinado por um signo B quando o domínio de A varia conforme o valor atribuído a B. A dependência entre os signos será tratada na descrição textual da interação.

Quantidade de Diálogos: todos, um, ou pelo menos um.

Além de indicar se a ordem em que os diálogos serão travados é importante ou não, o designer também deve indicar quantos diálogos daquele conjunto devem travados durante a interação. Logo, o agrupamento dos diálogos deve conter os operadores que indicam a ordem acrescidos dos operadores que indicam a quantidade dos diálogos.

Em casos onde o designer desejar indicar que todos os diálogos de um conjunto devem ser travados, ele deve agrupá-los com o operador *and*. Caso não exista nenhuma indicação sobre a quantidade de diálogos que deverão ser travados, considera-se que sejam todos, ou seja, o operador *and* pode ser omitido. Por exemplo, *group-and*, *group*, *and* ou simplesmente nenhum operador, indica que todos os diálogos daquele conjunto devem ser travados independente de ordem. De forma análoga, *seq-and* ou simplesmente *seq*, indica que todos os diálogos daquele conjunto devem ser travados na ordem indicada pelo designer.

```
and {
    [diálogo 1]
    [diálogo 2]
}
```

Ou simplesmente

```
[diálogo 1]
[diálogo 2]
```

Em certos momentos da interação, primeiro devemos escolher, dentre um conjunto de diálogos, qual diálogo iremos travar, para somente depois travarmos o diálogo escolhido. Os diálogos que pertencem a este conjunto são chamamos de

diálogos mutuamente exclusivos. Por exemplo, para o usuário adicionar um novo contato no MSN Messenger, ele pode escolher um contato no seu catálogo de endereços ou criar um novo contato com base em um endereço de e-mail (Figura 63).

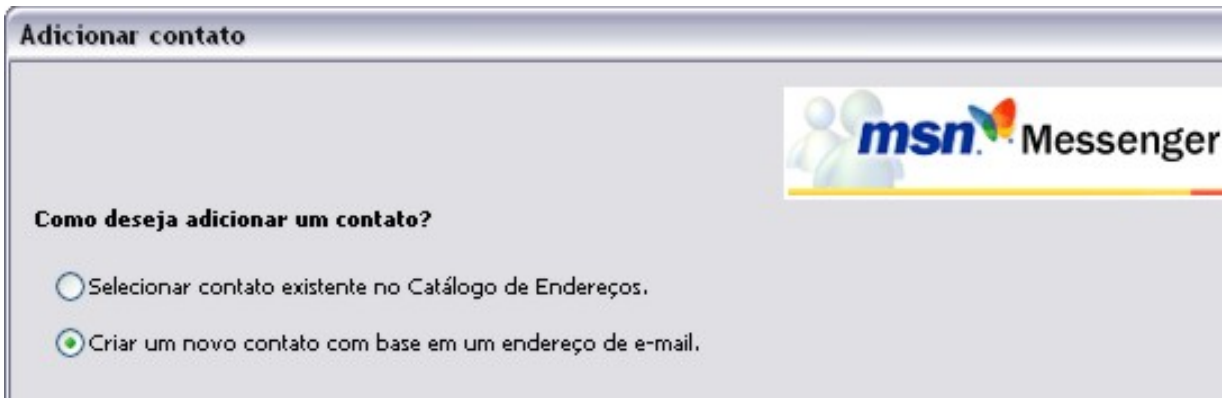


Figura 63: Tela onde o usuário adiciona um novo contato no MSN Messenger.

Caso os diálogos sejam mutuamente exclusivos, são agrupados e precedidos pelo operador `xor`:

```
xor {
   [diálogo 1]
   [diálogo 2]
}
```

Se, de outra forma, o usuário precisar travar pelo menos um diálogo de um conjunto, os diálogos são agrupados e precedidos pelo operador `or`:

```
or {
   [diálogo 1]
   [diálogo 2]
}
```

Em resumo, o designer pode indicar a ordem em que os diálogos devem ser travados agrupando-os com os operadores *group* ou *seq*, e, além disto, indicar quantos diálogos devem ser travados agrupando-os com os operadores *and* (todos), *xor* (um) ou *or* (pelo menos um).

Estruturas compostas

Utilizando-se os operadores apresentados anteriormente, torna-se possível articular diálogos em estruturas mais complexas, como neste exemplo:

```
xor {
   seq {
      [diálogo 1]
      [diálogo 2]
   }
   seq {
      [diálogo 3]
      [diálogo 4]
   }
   or {
      [diálogo 5]
      [diálogo 6]
   }
}
```

Se voltarmos a considerar que uma *estrutura de articulação da interação* proposta pela LEMD (Leite, 1998) é uma forma de manifestação, de execução na interface da conversa descrita por um *diálogo*, podemos pensar nos possíveis mapeamentos entre estas propostas. Assim, um diálogo seria mapeado para um conjunto estruturado de ações ou visualizações sobre signos de interface na LEMD, e a estrutura de diálogos seria uma composição de estruturas de articulação da interação na LEMD. As possíveis formas de estruturar articulações da interação na LEMD são as seguintes: `select`, `repeat`, `sequence`, `join`, e `combine` (p. 163, definição sintática, e 119, definição semântica).

A estrutura de seleção (`select`) indica que o usuário pode escolher alternativas de interação (por interação entenda-se execução de ações sobre a interface, inputs e outputs), o que na nossa estrutura de diálogos equivale ao operador `xor`. Já na estrutura de repetição (`repeat`) as interações devem ser realizadas repetidamente. A repetição de diálogos geralmente é indicada pela descrição no plural dos signos manipulados por ele. Por exemplo, no diálogo *[informar alunos da turma]*, espera-se que o usuário repita as falas necessárias para informar um aluno tantos quantos forem os alunos da turma em questão. A estrutura de seqüência (`sequence`) determina a ordem em que as interações deverão ser executadas, o que na estrutura de diálogos equivale ao operador `seq`. Na estrutura de agrupamento (`join`) todas as interações devem ocorrer, mas em

qualquer ordem, o que na estrutura de diálogos equivale ao operador `group`. Por fim, a estrutura de combinação "requer que duas ou mais interações sejam realizadas de maneira dependente uma da outra ou de maneira síncrona" (p. 119). Quando dizemos que dois diálogos são seqüenciais, isto também pode significar que o segundo diálogo depende do primeiro. No entanto, ainda não foi proposta uma forma para representar sincronismo entre diálogos ou cenas na MoLIC.

Restrições sobre o rumo da conversa

Certas falas de transição podem exigir que anteriormente tenham sido travados alguns diálogos na cena de origem. A indicação da obrigatoriedade deve estar vinculada à fala de transição e não somente aos diálogos, pois um (conjunto de) diálogo(s) pode ser obrigatório para uma transição e não o ser para outra. Sendo assim, podemos agrupar os diálogos obrigatórios em um grupo com nome e depois referenciá-lo na pressuposição da fala de transição apropriada.

A Figura 64a apresenta uma cena com três falas de transição, indicando rumos de conversa distintos. Para o usuário emitir a primeira fala de transição (*u:[primeira fala]*), ele deve antes ter travado os diálogos do grupo 1 (*[diálogoA]* e *[diálogoB]*). Para ele emitir a segunda fala, antes devem ter sido travados os diálogos do grupo 2 (*[diálogoB]* e *[diálogoC]*). Por último, para emitir a terceira fala de transição, não é necessário ter sido travado nenhum diálogo anteriormente. A pressuposição de um conjunto de diálogos terem sido travados antes de uma fala de transição também se aplica no caso da cena *Escrever e-mail* (Figura 64b). Antes de confirmar o envio do e-mail, o usuário deve ter informado os dados do e-mail no diálogo *[informar dados do e-mail]*.

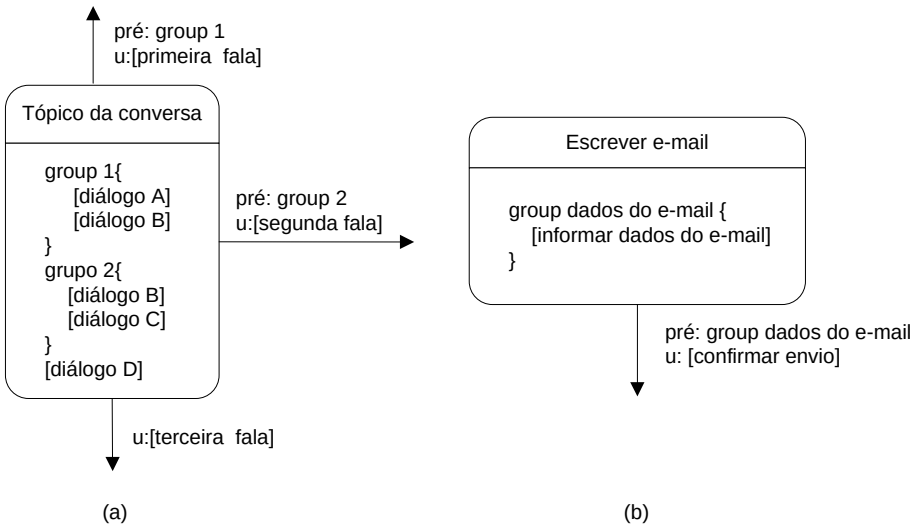


Figura 64: Diálogos obrigatórios para falas de transição.

Definição e estrutura dos signos

Como complemento ao diagrama de interação, cada diálogo deve ser especificado para servir de insumo na construção dos *storyboards* correspondentes. Esta especificação deve definir e estruturar os signos envolvidos (ou referenciados) em cada diálogo. Os signos de um diálogo também podem ser estruturados de forma semelhante aos diálogos (seção Estrutura de diálogos), utilizando-se os operadores *group*, *seq*, *and*, *xor* e *or*.

Quando os signos são estruturados com *group* (Figura 65, cena de tópico 1), eles devem ser apresentados pelo designer ou modificados pelo usuário em qualquer ordem naquele conjunto de signos, mas a ordem de descrição do conjunto sugere uma ordem de apresentação destes signos na interface, que pode ou não ser implementada. Estruturando-se os signos com o operador *seq* (Figura 65, cena de tópico 2), indica-se que os signos deverão ser apresentados pelo designer ou manipulados pelo usuário na ordem representada (primeiro o *signo 1* depois o *signo 2*). Em casos onde os signos estejam estruturados com o operador *and*, todos os signos daquele grupo devem ser modificados pelo usuário ou apresentados pelo designer (Figura 65, cena de tópico 3). Quando os signos forem estruturados com o operador *xor* (Figura 65, cena de tópico 4), eles serão mutuamente exclusivos, ou seja, ou o usuário modifica (ou o designer apresenta) o *signo 1* ou modifica o *signo 2*, nunca os dois no mesmo instante da interação. Por fim, se os signos forem estruturados com o operador *or* (Figura 65, cena de

tópico 5), o usuário ou o preposto do designer poderão informar pelo menos um signo do conjunto dos signos que compõem o diálogo.

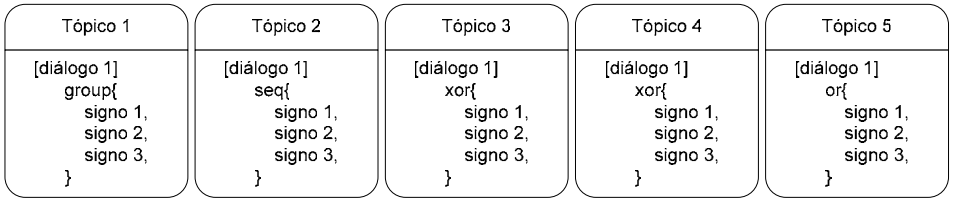


Figura 65: Possibilidades de estruturação dos signos.

A Figura 66 apresenta como o usuário dos programas Outlook, Eudora e Thunderbird pode modificar os valores dos signos que compõem o signo "e-mail". As seqüências de apresentação destes signos nas interfaces podem ser sugeridas pela estruturação dos signos que compõem o signo "e-mail", conforme apresentado na Tabela 11.

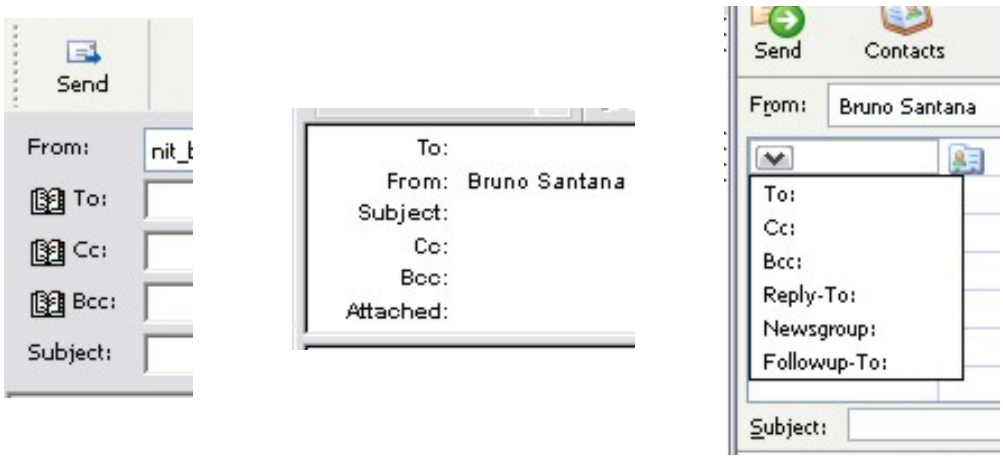


Figura 66: *Screenshots* do Outlook, Eudora e Thunderbird na tela onde o usuário escreve um e-mail *(da esquerda para direita)*.

Outlook	**Eudora**	**Thunderbird**

group {
from
to
cc
bcc
subject
}
	group {
	to
	from
	cc
	bcc
	subject
	}
		group {
		from
		group {
		to
		cc
		bcc
		}
		subject
		}

Tabela 7: Estrutura dos signos do e-mail no Outlook, Eudora e no Thunderbird.

Esta proposta de estruturação dos signos nos faz lembrar da estruturação de articulações da interação na LEMD (Leite, 1998). Como discutido na estruturação dos diálogos (seção Estrutura de diálogos), as formas de agrupamento propostas são semelhantes. Contudo, a diferença fundamental é que a LEMD não agrupa falas sobre signos do domínio, mas sim ações sobre os signos de interface. Uma proposta de estruturação dos signos mais próxima deste trabalho é aquela feita por Dahis (1991), que foi baseada na LEMD e na metáfora de interação como uma conversa. Apesar das semelhanças com este trabalho, a proposta de Dahis se diferencia deste pelo nível de detalhes utilizado para definir a interação. Enquanto este trabalho (que é baseado no trabalho de Paula, 2003) propõe definir apenas os signos sobre os quais se fala algo para contribuir com o diálogo, Dahis propõe definir todas as falas dos participantes da conversa. Na Figura 67, por exemplo, a apresenta a estruturação dos signos que compõem o diálogo [informar dados do cliente]. No lado esquerdo da figura, encontramos a estruturação deste diálogo segundo a proposta de Dahis (Dahis, 1991 p. 104) e no lado direito, a proposta deste trabalho. Preferimos utilizar uma descrição mais enxuta para facilitar a reflexão do designer sobre a concepção da interação.


```
group                                                    group
{                                                        {
     system:<Meio de contato> (Information Request)          e-mail?
     sequence subtopic <e-mail>                              telefone?
     {                                                       endereço?
       system: <E-mail> (Information Request)            }
       user: e-mail (Domain Information)
     }
     sequence subtopic <telefone>
     {
       system: <Telefone> (Information Request)
       user: telefone (Domain Information)
     }
     sequence subtopic <endereço>
     {
       system: <Endereço> (Information Request)
       user: endereco (Domain Information)
     }
}
```

Figura 67: Estrutura dos signos do diálogo *[informar dados do cliente]* na proposta de Dahis (à esquerda) e na proposta deste trabalho (à direita).

Descrição textual da interação

O diagrama de interação privilegia a visão global da interação focada na estrutura de tópicos da conversa. A legibilidade deste diagrama ficaria prejudicada se representássemos nele todos os detalhes da conversa necessários para a construção da interface. Por isto, o diagrama de interação é complementado por uma *descrição textual,* que especifica a interação, incluindo os detalhes da conversa.

Na segunda edição da MoLIC, a descrição textual da interação envolve apenas a especificação dos diálogos e a representação dos tratamentos de *breakdowns* relacionados com cada cena. A especificação dos diálogos está relacionada com (1) a estruturação dos diálogos dentro da cena, (2) a estruturação dos signos envolvidos nos diálogos e (3) a (re)definição dos atributos dos signos em função do seu uso na cena. Os signos (e seus atributos) usados na descrição textual das cenas já deveriam estar definidos na ontologia de signos.

Os tratamentos de *breakdowns* podem estar relacionados ao signo, ao diálogo com um todo ou até mesmo à própria cena. Como cada fala de *breakdown* terá seu tratamento específico, devemos associar expressões do tipo **se** <fala de *breakdown*> **então** <forma de tratamento> a cada signo, diálogo ou cena associada relacionado.

A Figura 68 mostra a gramática em BNF para a descrição textual da MoLIC que detalha os (signos dos) diálogos.

```
cena ::= tópico <nome do tópico>
 [tratamento_de_breakdowns]
 conversa

conversa ::= {monólogo_do_preposto} {estrutura_de_diálogos}

tratamento_de_breakdowns ::= "(" tratamento_de_breakdowns
                                {tratamento_de_breakdowns} ")"

tratamento_de_breakdowns ::=  se <descrição do breakdown>
                              então <forma de tratamento>

operador ::= group | seq | and | xor | or

estrutura_de_diálogos ::= diálogo {diálogo} |
                          operador "{" diálogo {diálogo} "}"
```

```
diálogo ::= [tratamento_de_breakdowns]
            “[” <descrição do diálogo> “]”
            estrutura_de_signos

enunciador ::= “?” | “!”

estrutura_de_signos ::= signo {signo} |
                        operador “{” signo {signo} “}” |
                        estrutura_de_signos {estrutura_de_signos}

estrutura_de_signos_do_preposto ::=
     signo_do_preposto {signo_do_preposto } |
     operador “{” signo_do_preposto {signo_do_preposto} “}” |

estrutura_de_signos_do_preposto{estrutura_de_signos_do_preposto}

signo ::= <descrição do signo> enunciador
          [obrigatório “=” <obrigatório>]
          [ “<” atributos_do_signo “>” ]
          tratamento_de_breakdowns
          [partes_do_signo]

signo_do_preposto ::= <descrição do signo> “!”
                      [obrigatório “=” <obrigatório>]
                      [ “<” atributos_do_signo “>” ]
                       tratamento_de_breakdowns
                      [partes_do_signo]

atributos_do_signo ::= atributo_do_signo |
                       atributo_do_signo “,”atributos_do_signo

atributo_do_signo ::=  descrição “=” <descrição>
                       origem  “=” <origem>
                       tipo    “=”  <tipo do conteúdo> |
                       domínio “=”  <domínio do conteúdo> |
                       default “=”  <valor default> |
                       formato “=”  <formato esperado> |
                       widget “=”   <widget recomendado>

monólogo_do_preposto ::= estrutura_de_signos_do_preposto |
                         “”” < mensagem do designer com conteúdo
                         extensional > “””

signos_do_preposto ::= signo_do_preposto {“,” signo_do_preposto}

partes_do_signo ::= signo [ “,” partes_do_signo]
```

Figura 68: Detalhamento dos diálogos na descrição textual.

Os atributos do signo estarão definidos na ontologia de signos e, se necessário, serão redefinidos na descrição textual da cena.

Detalhamento da ontologia de signos

A segunda etapa do projeto de interação receberia[3] a ontologia de signos inicialmente definida, ou seja, com parte dos signos e seus relacionamentos, sem, no entanto, definir todos os atributos dos signos. À medida que o designer especifica os signos que compõem os diálogos, a ontologia de signos evoluiria e novos signos iriam sendo definidos. Apesar de a ontologia de signos ainda não ter sido proposta, nem por este nem por outros trabalhos anteriores, a especificação dos diálogos proposta aqui gerou alguns requisitos para o detalhamento da ontologia de signos. Por isto, apresentaremos nesta seção uma organização dos atributos dos signos que deverão fazer parte da ontologia de signos.

A ontologia de signos deverá organizar os atributos do signo em função do seu conteúdo, sua expressão e sua identidade. Em linhas gerais, a identidade do signo está relacionada com seu identificador, sua descrição e sua origem (Tabela 8); o conteúdo de um signo está relacionado com seu tipo de valor, com as restrições sobre os valores que ele pode assumir e com o seu valor default, quando houver (Tabela 9); e, por fim, a expressão de um signo está relacionada com a maneira pela qual o enunciador do signo pode (ou deve) expressar seu conteúdo considerando o formato esperado (Tabela 10). Os elementos de interação (*widgets*) que compõem a interface determinam a maneira pela qual o conteúdo de um signo pode ser expresso, segundo determinada intenção de comunicação.

[3] Isto é o que se espera que aconteça, quando houver uma proposta de ontologia de signos para a MoLIC. Hoje o que existe é apenas uma tabela de signos.

INTENÇÃO DE COMUNICAÇÃO

Atributo	*Exemplos*
Identificador (id) O nome que identifica o signo.	aluno matrícula data de cadastramento
Descrição Representa o significado do signo. (Pode conter **"**informação adicional para ser comunicada aos usuários para auxiliá-los compreender e/ou manipular o signo" (Barbosa et al., 2004 p.4))	aluno da universidade matrícula do funcionário na empresa data na qual o documento foi cadastrado
Origem Indica se o signo pertence ao domínio do problema (signo de domínio) ou se foi criado (signo de aplicação) ou modificado por causa da tecnologia (signo transformado). O emprego de signos de aplicação exige um cuidado especial do designer para comunicar ao usuário o seu significado, pois seu significado não está (ainda) estabelecido na cultura do usuário.	signo de domínio signo de aplicação signo transformado

Tabela 8: Atributos relacionados com a intenção de comunicação do signo.

CONTEÚDO

Atributo	*Exemplos*
Tipo O tipo do conteúdo caracteriza os valores que o signo pode assumir.	<u>Tipos Simples</u> Texto Número Data Hora Sim/Não – Booleano Imagem Áudio Vídeo <u>Tipos Compostos</u> Conjunto Lista Tabela Árvore Grafo ou simplesmente composto
Domínio Conjunto dos valores possíveis que o signo pode assumir, ou restrições sobre esses valores. O domínio de um signo pode depender do valor de outro signo.	<u>Independentes de outros signos</u> no máximo 50 caracteres número entre 1 e 250 de 01 de janeiro de 1980 até hoje hora em intervalos de 15 minutos <u>Dependentes de outros signos</u> departamentos da *universidade* cidades do *estado* nome do aluno da *matrícula* autores do livro do *código*

Valor default O valor default é o primeiro valor que o signo assume quando é definido (ou apresentado na interface). Ele é a sugestão do preposto que o usuário pode aceitar ou não.	<u>Independentes de outros signos</u> “casa” 1 data de hoje hora atual sim <u>Dependentes de outros signos</u> departamento de informática (depende da universidade) cidade de Niterói (depende do estado) Bruno Santana (depende da matrícula) Barbosa e de Souza (depende do código do livro)

Tabela 9: Atributos relacionados com o conteúdo do signo.

Ao representar o signo na especificação textual, precisamos informar quem é o enunciador do signos, ou seja, quem determina o seu valor. Usamos “!” para indicar que o preposto determina o valor do signo, sem que o usuário possa dizer algo a respeito do signo, e “?” quando for o caso do usuário determinar o valor do signo, mesmo que o preposto tenha oferecido um valor *default*.

Vale observar que, mesmo que seja o usuário quem deva determinar o valor do signo, presume-se que o preposto indique que este é o caso. Em outras palavras, todo signo marcado com “?” pressupõe uma fala introdutória do preposto, seguida de uma fala do usuário que determine seu valor. Por exemplo, o preposto poderia dizer algo parecido com “– Usuário, você pode escolher qualquer departamento deste conjunto.” e o usuário responderia “– eu escolho ‘Departamento de Informática’ ”.

Como a expressão do signo varia em função do enunciador do signo, a ontologia de signo deve representar duas possíveis expressões: uma quando o usuário enunciar o signo e outra quando o preposto do designer for o enunciador.

EXPRESSÃO

Atributo	*Exemplos*
Elemento de interface (*widget*) recomendado Indica qual é o *widget* recomendado pelo designer para a manipulação do signo. O *widget* recomendado normalmente é abstrato, isto é, independente de implementação e de plataforma, dado que ainda não é indicado se pensar em detalhes de interface. No entanto, vale ressaltar que a implementação da forma de expressão do signo pode fazer uso de outro *widget* equivalente (por exemplo, na hora de projetar ou mesmo de implementar a interface, o designer pode decidir trocar um *widget* de escolha múltipla por outro de escolha simples em função do espaço disponível).	Texto livre Escolha/Seleção simples Escolha/Seleção múltipla Rótulo
Formato esperado O formato esperado é a sugestão do designer para a restrição da expressão do signo na interface.	ddd.ddd.ddd (xx)dddd-dddd dd/mm/aaaa dd de mês-por-extenso de aaaa

Tabela 10: Atributos relacionados com a expressão do signo.

Além dos atributos de cada signo, a ontologia de signos deve definir o relacionamento entre os signos (por exemplo, a hierarquia e a composição de signos a partir de outros signos). O relacionamento entre signos pode determinar a obrigatoriedade de um signo-parte na composição de um signo composto, ou seja, determinar se é necessário atribuir algum valor ao conteúdo de um signo quando este é tido como parte de um signo composto, mesmo que seja aceitar o valor *default*. A obrigatoriedade de um signo pode ser entendida como uma restrição ao seu conteúdo (o conteúdo não pode ser vazio). Por exemplo, o relacionamento entre o signo "pessoa" e o signo "nome" determina que toda pessoa tem um nome, isto é, o valor do signo "nome" que compõe "pessoa" não pode ser nulo.

Todavia, ser um signo-parte obrigatoriamente definido pela relação com um signo composto não implica que o valor do signo parte deverá ser definido em todos os momentos da interação. Por exemplo, um signo-parte pode ser obrigatório durante o cadastramento de um signo composto, mas ser indiferente

quando se realiza uma pesquisa pelo signo composto. Deste modo, a obrigatoriedade intrínseca do signo será definida pelas relações entre os signos na ontologia de signos e poderá ser modificada na descrição textual da cena em função do uso do signo.

Vamos, por exemplo, como seria a ontologia de signos e a descrição textual de cenas que utilizam o signo *e-mail*. A Figura 69 apresenta (parte de) a ontologia de signos que define o signo *e-mail* e os outros signos que o compõem, representada em um diagrama de classes. Segundo as relações entre os signos, podemos observar que o signo *e-mail* é obrigatoriamente composto pelos signos *destinatário* e *remetente*, e pode ser composto pelos signos *assunto*, *anexo*, *corpo do e-mail*, *destinatário de cópia* e *destinatário de cópia oculta*.

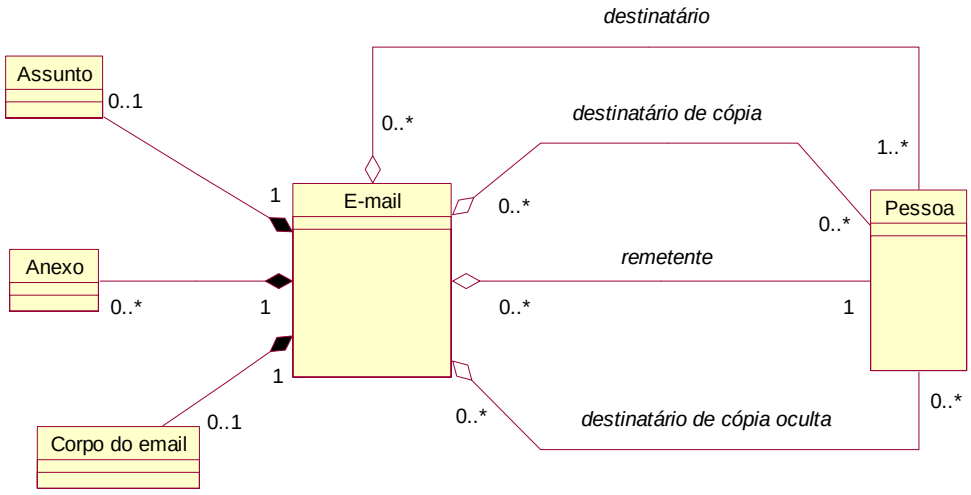


Figura 69: Diagrama de classes representado o signo *e-mail*.

Além da definição dos signos e de seus relacionamentos, a ontologia de signos também deve definir os atributos relacionados com sua identidade, seu conteúdo e sua expressão, conforme discutido anteriormente. Por exemplo, os atributos do signo composto *e-mail* apresentados na Tabela 11, também deveriam fazer parte da ontologia de signos. Note que, até então, não faz sentido definir o domínio nem a expressão de um signo composto.

Signo**: e-mail**		
Dimensão	Atributo	Valor
Identidade	Identificador	e-mail
	Descrição	Qualquer mensagem que pode ser enviada para um endereço eletrônico, semelhante a uma carta.
	Origem	signo transformado
Conteúdo	Tipo	composto
	Domínio	-
	Valor Padrão	-

Tabela 11: Atributos do signo composto e-mail descritos na ontologia de signos.

Os atributos de todos os signos que compõem o signo *e-mail* também deveriam estar definidos na ontologia de signos. Por exemplo, a Tabela 12 apresenta os atributos do signo *remetente* que compõem o signo *e-mail*.

Signo**: remetente**		
Dimensão	Atributo	Valor
Identidade	Identificador	remetente
	Descrição	Representa o remetente do e-mail. Na maioria das vezes o remetente é apenas indicado pelo seu endereço de e-mail, mas é possível que ele também seja identificado pelo seu nome.
	Origem	signo transformado
Conteúdo	Tipo	texto
	Domínio	possíveis endereços de e-mail, algumas vezes acompanhado pelo nome do usuário
	Valor Padrão	-
Enunciador: Preposto		
Expressão	*Widget* recomendado	rótulo
	Formato esperado	nome do remetente <endereço de e-mail> ou, simplesmente, <endereço de e-mail>
Enunciador: Usuário		
Expressão	*Widget* recomendado	texto livre
	Formato esperado	nome do remetente <endereço de e-mail> ou, simplesmente, <endereço de e-mail>

Tabela 12: Atributos do signo *remetente* que compõem o signo *e-mail*, descritos na ontologia de signos.

Uma vez definidos os signos e seus atributos na ontologia de signos, o designer pode especificar diálogos que apresentam ou modificam estes signos. Quando necessário, ele irá redefinir os atributos do signo na descrição textual da interação, conforme o emprego do signo na conversa. Por exemplo, vejamos como o signo remetente, que é parte do signo e-mail, é empregado nas cenas *Escrever e-mail* e *Ler e-mail* (Figura 70). Em típicos programas de e-mail, o designer geralmente irá empregar os signos *remetente* e *e-mail* conforme os atributos definidos na ontologia de signos (Tabela 12 e Tabela 11). Então, o único atributo a ser definido neste caso é o enunciador do signo: na cena *Escrever e-mail* é o usuário quem determina o valor dos signos *remetente* e, por conseguinte, *e-mail* (indicado na descrição textual por "?"), já na cena *Ler e-mail* é o preposto quem determina o valor destes signos (indicado na descrição textual por "!"), conforme apresentado na Figura 71. Com o apoio de uma ferramenta computacional, o designer poderia integrar os atributos definidos na ontologia de signos e as modificações realizadas na descrição textual para visualizar a especificação textual estendida destas cenas, conforme apresentado na Figura 72.

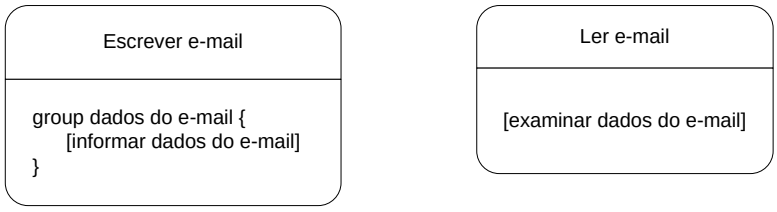


Figura 70: Cenas *Escrever e-mail* e *Ler e-mail*.

Cena: **Escrever e-mail**

```
group dados do e-mail {
    [informar dados do e-mail]
       e-mail.*?
}
```

Cena: **Ler e-mail**

```
group dados do e-mail {
    [examinar dados do e-mail]
       e-mail.*!
}
```

Figura 71: Especificação textual resumida das cenas Escrever e-mail e Ler e-mail.

Cena: **Escrever e-mail**

```
group dados do e-mail {
    [informar dados do e-mail]
       e-mail.*? < identificador = “e-mail”,
                   descrição = “Qualquer mensagem que pode ser enviada
                               para um endereço eletrônico, semelhante a
                               uma carta.”
                   origem = “signo transformado”
                   tipo = “composto”
                   domínio = “”
                   padrão = “” >
            // partes do signo ...
            remetente? < identificador = “remetente”
                         descrição = “Representa o remetente do e-mail. Na
                                     maioria das vezes o remetente é
                                     apenas indicado pelo seu endereço de
                                     e-mail, mas é possível que ele
                                     também seja identificado pelo seu
                                     nome.
                         origem = “signo transformado”
                         tipo = “texto”
                         domínio = “possíveis endereços de e-mail, algumas
                                    vezes acompanhado pelo nome do usuário”
                         padrão = “”
                         widget = “texto livre”
                         formato = “nome do remetente <endereço de e-mail>
                                    ou,simplesmente,<endereço de e-mail>”>
          //partes do signo ...
}
```

Cena: **Ler e-mail**

```
group dados do e-mail {
    [examinar dados do e-mail]
       e-mail.*! < identificador = “e-mail”,
                   descrição = “Qualquer mensagem que pode ser enviada
                               para um endereço eletrônico, semelhante a
                               uma carta.”
                   origem = “signo transformado”
                   tipo = “composto”
                   domínio = “”
                   padrão = “” >
            // partes do signo ...
            remetente? < identificador = “remetente”
                         descrição = “Representa o remetente do e-mail. Na
                                     maioria das vezes o remetente é
                                     apenas indicado pelo seu endereço de
                                     e-mail, mas é possível que ele
                                     também seja identificado pelo seu
                                     nome.
                         origem = “signo transformado”
                         tipo = “texto”
                         domínio = “possíveis endereços de e-mail, algumas
                                    vezes acompanhado pelo nome do usuário”
                         padrão = “”
                         widget = “rótulo”
                         formato = “nome do remetente <endereço de e-mail>
                                    ou,simplesmente,<endereço de e-mail>” >
          //partes do signo ...
     }
```

Figura 72: Especificação textual estendida das cenas Escrever e-mail e Ler e-mail.

O objetivo do modelo diagramático de interação é dar uma visão global aos designers do discurso interativo como um todo, enquanto a especificação textual fornece os detalhes da conversa visando alimentar a especificação do sistema. O modelo de interação facilita a reflexão do designer sobre suas decisões de projeto, pois ele consegue se imaginar tanto no papel do usuário quanto no do seu preposto, ao projetar ou verificar as falas de cada um, nos atos comunicativos.

Normalmente a especificação textual é feita separadamente da representação diagramática. Entretanto, é possível incluir parte da especificação textual no diagrama de interação. Na Figura 73, os diálogos são acompanhados dos referidos signos que é parte da especificação textual. O único problema desta abordagem é que o diagrama pode ficar confuso com tanta informação e o designer pode ser levado a pensar simultaneamente em níveis de detalhes distintos, dificultando certos tipos de reflexão.

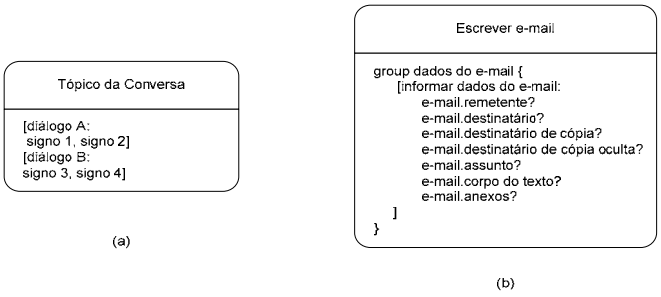


Figura 73: Cenas com signos referenciados.

Classificação das falas de reparo de *breakdown*

Além da especificação dos diálogos registrada na descrição textual de cada cena, o detalhamento da interação também envolve a classificação das falas de reparo de *breakdown*. O designer classifica as falas de reparo de *breakdown* com o respectivo mnemônico escrito entre parênteses no início da descrição da fala (Figura 74).

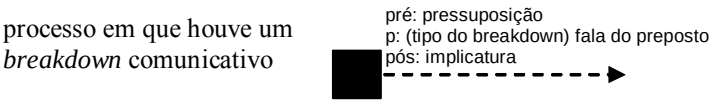


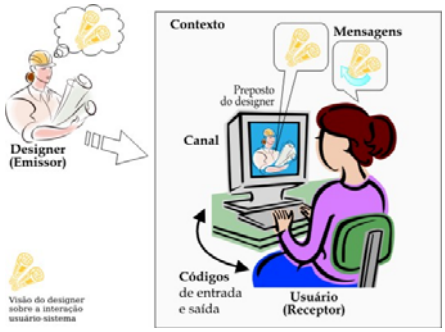
Figura 74: Representação do tipo nas falas de reparo de *breakdown*.

Existem 5 tipos de mecanismos de tratamento ou prevenção de rupturas comunicativas (Barbosa e Paula, 2003) para classificar as falas de reparo de *breakdown*:

- **prevenção passiva (*PP, passive prevention*)**: rupturas que podem ser evitadas por documentação ou instruções *online*. Por exemplo, comunicar previamente que "a expressão deste valor deve ser dd/mm/aaaa" ou que "o usuário não possui acesso a parte X do sistema";
- **prevenção ativa (*AP, active prevention*)**: rupturas que podem ser evitadas pelo sistema (impedindo que o usuário expresse uma fala inválida que o levaria a uma ruptura na comunicação). Por exemplo, impedir que o usuário utilize letras ou símbolos para expressar um conteúdo numérico, ou habilitar ou desabilitar botões de acordo com o *status* atual da aplicação;
- **prevenção apoiada (*SP, supported prevention*)**: situações que o preposto detecta como sendo possíveis rupturas, mas cuja decisão sobre ser ou não uma ruptura deve ser tomada pelo usuário. Por exemplo, quando o usuário expressa uma intenção de armazenar um arquivo com um nome diferente e em seguida expressa o nome utilizado atualmente. Este tipo de apoio geralmente é concretizado na interface por mensagens de confirmação (por exemplo, "Arquivo já existe, deseja sobrescrever?");
- **recuperação apoiada (*SR, supported recovery*)**: Rupturas que devem ser remediadas pelo usuário com apoio do preposto do designer (por exemplo, apresentar uma mensagem de erro sobre o preenchimento de um campo e uma oportunidade para o usuário corrigi-lo); e
- **captura de erro (*EC: error capture*)**: situações em que não há uma ruptura comunicativa, mas sim um erro do sistema. Quando o preposto é capaz de identificar erros do sistema, deve notificá-los ao usuário e, se possível, sugerir formas de se recuperar do erro. No entanto, a conversa de remediação deve ocorrer fora da aplicação. Um exemplo de captura de erro é: "O arquivo está corrompido. É possível que isto se deva a uma interrupção no *download* ou cópia do arquivo de sua localização original.".

O designer se mantém refletindo sobre a interação durante a classificação das falas de transição de reparo de *breakdown*. Em especial, ele tem a oportunidade de revisar as falas de transição (com ou sem reparo de *breakdown*) e as conversas iniciadas a partir delas (no caso de um *breakdown*, espera-se uma forma de recuperação do mal-entendido).

4.2 Espaço de Design na MoLIC



Miniatura da Figura 5: Espaço de *design* de IHC segundo a engenharia semiótica.

O espaço de design de IHC estruturado pela engenharia semiótica decorre de como esta teoria caracterizou a interação humano-computador (de Souza, 2005). Como discutido na seção 2.2.1, a engenharia semiótica caracteriza IHC como um processo particular de comunicação, onde a metacomunicação designer-usuário se manifesta durante e influência a comunicação usuário-sistema (preposto do designer). Deste modo, o espaço de design de IHC envolve os seis elementos do processo de comunicação – emissor, receptor, canal, mensagem, código e contexto – (seção *Processos de Significação e Comunicação*), conforme esquematizado na Figura 5 (veja sua miniatura à direita). Durante a interação existem no mínimo dois[4] interlocutores: o usuário e o preposto do designer, seu representante cristalizado na interface que comunica sua visão sobre a interação durante a própria interação. Assim, podemos dizer que a interação é uma seqüência de turnos de fala onde o usuário e o preposto do designer se alternam no papel de emissor e receptor de mensagens.

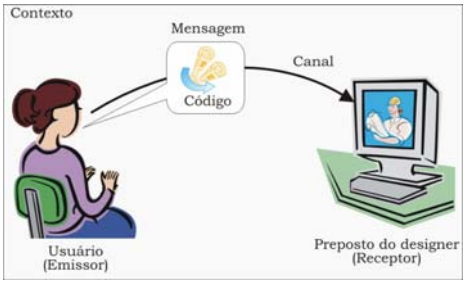
Resta-nos agora compreender como a MoLIC auxilia o designer a refletir sobre os elementos e o processo de comunicação que fazem parte do espaço de design de IHC segundo a engenharia semiótica. Vamos analisar separadamente cada elemento do diagrama de interação.

[4] Se a aplicação for multi-usuário, teremos mais de um usuário como interlocutor.

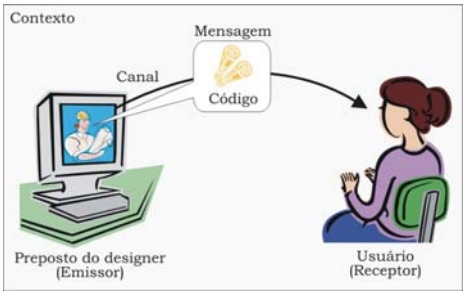
Cena

Durante a conversa descrita por uma cena, o usuário e o preposto do designer se alternam como *emissor* e *receptor* de falas em torno do tópico da cena. Estas falas são organizadas em pares conversacionais (Marcuschi, 1999) para compor os diálogos da cena. Cada fala, seja ela emitida pelo usuário ou pelo preposto, tem o objetivo de transmitir ao *receptor* uma *mensagem* através da interface, isto é, através do *canal*. Portanto, a interface do sistema (o canal) é quem determina os possíveis *códigos* que o emissor pode utilizar para compor sua mensagem. O tópico da conversa, por enquanto, é a única referência ao *contexto* da conversa representada na cena.

As mensagens trocadas durante um diálogo dizem respeito a um conjunto de signos. Para o designer ser capaz de definir os atributos de cada signo, ele deve ter em mente os seis elementos da conversa observados sob o ponto de vista do enunciador do signo, conforme apresentado na Figura 6 e Figura 7 (vide miniaturas à direita). O enunciador do signo depende do emissor da mensagem onde determina o valor do signo. O conteúdo do signo contribui para o conteúdo da mensagem. Por um lado, o emissor necessita que os signos que componham a mensagem possam ser expressos em algum código na interface (o canal) que irá transmiti-lo, ou seja, os atributos do signo relacionados com a expressão dependem de alguma forma dos códigos disponíveis na interface. Por outro lado, o receptor necessita conhecer o código utilizado para expressar os signos que compõem a mensagem transmitida pela interface (o canal), para, assim, ser capaz de interpretá-la corretamente. Por



Miniatura da Figura 6: Espaço de *design* de IHC do ponto de vista do usuário como emissor.



Miniatura da Figura 7: Espaço de *design* de IHC sob o ponto de vista do preposto do designer como emissor.

fim, a identidade de cada signo está relacionada com a intenção de comunicação do emissor da mensagem.

Processo do sistema

É um momento onde o preposto do designer está "pensando" para *emitir* uma fala de transição com a *mensagem* que apresenta ao usuário (o *receptor*) o resultado do processo do sistema e marca a mudança de tópico da conversa.

Falas de Transição

Uma fala de transição é uma *mensagem* do emissor que modifica o tópico da conversa. Tanto o usuário (representado por *u:*) quanto o preposto do designer (representado por *p:*) podem ser o *emissor* da mensagem. As pressuposições das falas de transição fazem referência ao *contexto* da conversa. Já as implicaturas alteram o *contexto* da conversa. Para que uma fala de transição chegue ao *receptor*, o *emissor* deve ser capaz de expressá-la usando algum *código* disponível na interface (o *canal*) para poder transmiti-la.

Pontos de Acesso

O ponto de entrada apenas representa o momento onde a conversa usuário-preposto se inicia. Isto significa que exatamente após este momento, a interface (o *canal*) deve estar disponível para a troca de *mensagens* entre o usuário e o sistema.

O ponto de saída apenas representa um momento em que a conversa usuário preposto terminar. Conseqüentemente, a interface (o *canal*) não precisa mais estar disponível para a troca de *mensagens*. Isto normalmente ocorre em ambientes gráficos quando a janela do programa se fecha, interrompendo o *canal* de comunicação.

O acesso ubíquo representa uma cena como outra qualquer. Naquele momento da interação pode ocorrer qualquer conversa descrita por uma cena.

Correspondência entre o Diagrama de Metas e de Interação

A correspondência entre o diagrama de metas e o diagrama de interação não representa uma comunicação entre os interlocutores da conversa. O seu maior propósito é a rastreabilidade entre os diagramas. Apesar disto, a meta do usuário serve para o designer presumir um caminho de interação que o usuário irá percorrer para alcançá-la. Assim, ele pode pressupor uma seqüência de mensagens necessária para alcançar a meta do usuário.

Pontos de Contato

Um ponto de contato representa um *canal* que transmite *mensagens* do usuário ou do seu preposto (*emissores*) para um ator externo e vice-versa, seja o ator externo um sistema externo ou outro usuário utilizando outra instância do sistema sendo projetado.

É importante notar que o emissor, o receptor, os signos que compõem as mensagens trocadas, uma parte implícita do contexto e os possíveis rumos da conversa são definidos na MoLIC. Porém, a definição do canal (a interface) e dos códigos utilizados não são definidos na MoLIC, apesar de os atributos do signo sugerirem algumas características do canal e do código quando recomenda os *widgets* de interface e o formato esperado.

4.3 Metamodelo do Diagrama de Interação MoLIC

Esta seção apresenta o metamodelo dos elementos do diagrama de interação MoLIC descrito como um diagrama de classes UML, que define as regras e restrições necessárias à construção do diagrama de interação. Como, representar o diagrama de classes do metamodelo em uma única figura dificultaria imensamente sua leitura e compreensão, decidimos apresentá-lo em diferentes figuras, cada qual focando algum aspecto da construção do diagrama de interação. A definição de uma cena (cena, diálogos e signos) é descrita na Figura 75. As possíveis origens de uma fala de transição são descritas na Figura 76, enquanto que os possíveis destinos são descritos na Figura 77. As influências da interação usuário-

ator externo com origem em um ponto de contato são descritas na Figura 78. Já as influências da interação com destino em um ponto de contato são descritas Figura 79.

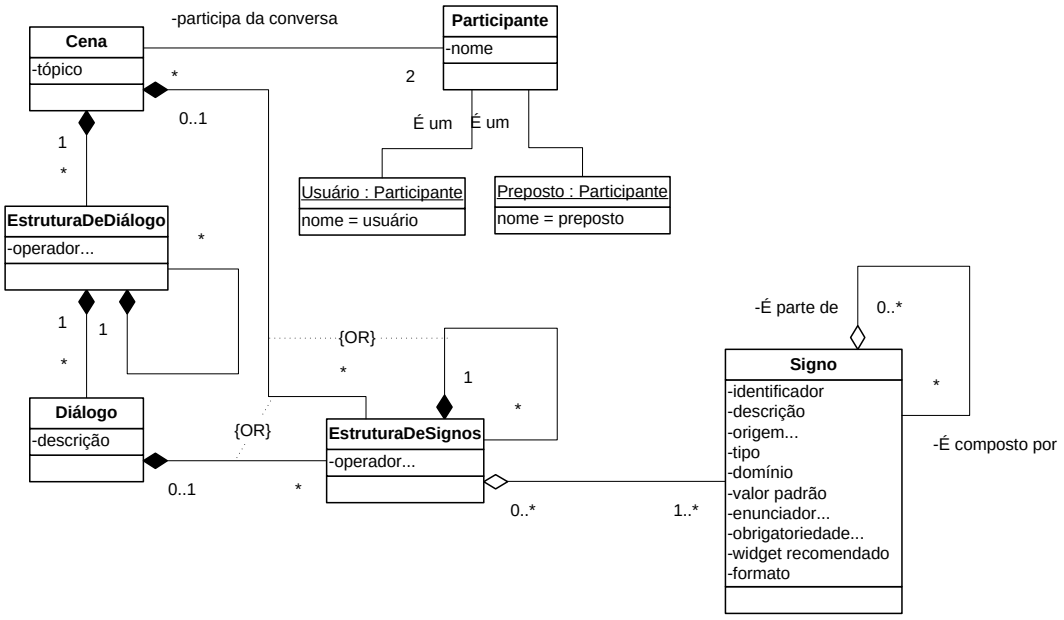


Figura 75: Parte do metamodelo que descreve a definição de uma cena.

Os atributos da Figura 75 que estão seguidos de reticências (por exemplo, EstruturaDeSignos.operador...) possuem restrições associadas, conforme descrito na Tabela 13, abaixo.

Atributo	Valor Default	Valores Possíveis
EstruturaDeDiálogos.operador e EstruturaDeSignos.operador	group	group, seq, and, xor e or
Signo.origem		signo de domínio signo de aplicação signo transformado
Signo.enunciador		os participantes da conversa: o usuário ou o preposto
Signo.obrigatoriedade	Não	Sim/não (obrigatório/não obrigatório)

Tabela 13: Restrições sobre o conteúdo de atributos envolvidos numa cena.

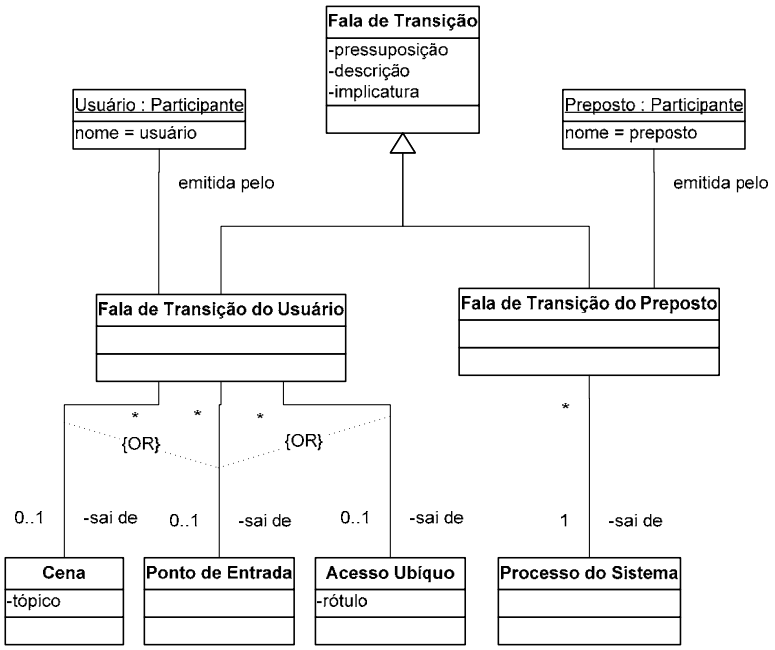


Figura 76: Parte do metamodelo que descreve as possíveis origens da fala de transição.

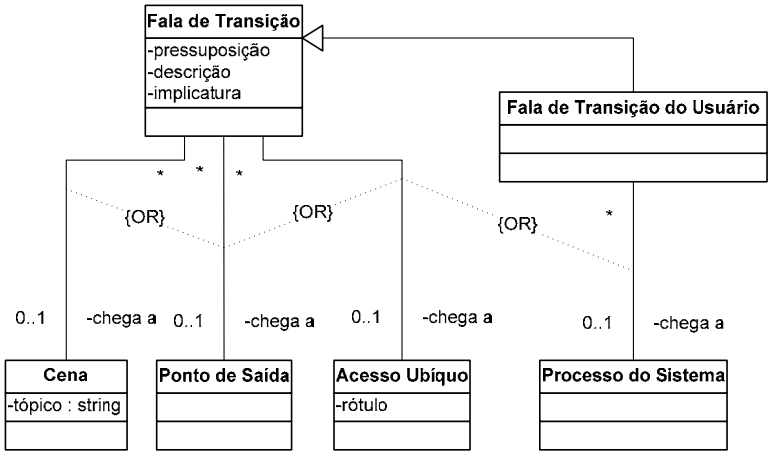


Figura 77: Parte do metamodelo que descreve os possíveis destinos da fala de transição.

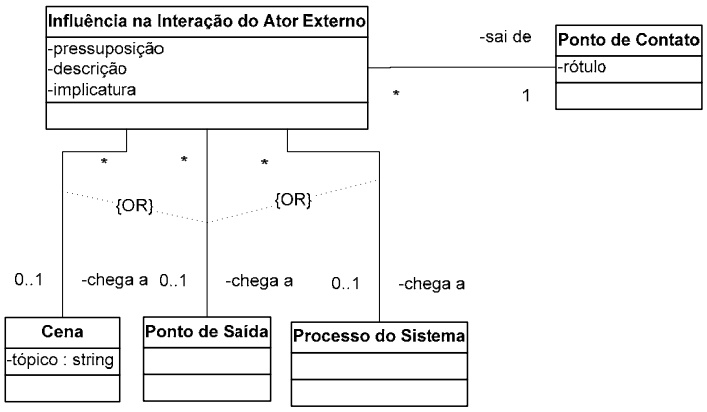


Figura 78: Parte do metamodelo que descreve as influências da interação com origem em um ponto de contato.

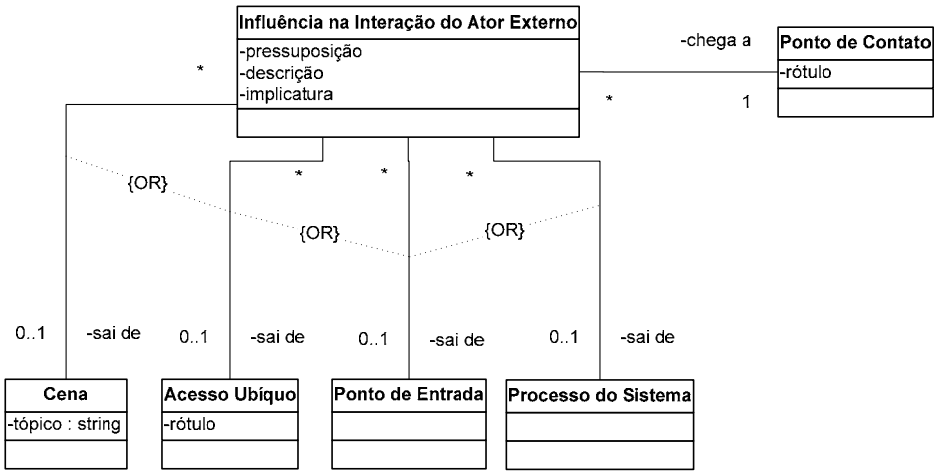


Figura 79: Parte do metamodelo que descreve as influências da interação com destino em um ponto de contato.