

2

Referencial teórico

A pesquisa apresentada nesta tese dá ênfase aos fenômenos comunicacionais envolvidos no processo de construção de artefatos computacionais por usuários finais. Assim, este capítulo discorre sobre teorias e conceitos que permeiam nossas investigações.

2.1

Computação e linguagem

Sistemas computacionais são sistemas simbólicos em sua essência. Através destes sistemas, signos são criados, interpretados, manipulados, modificados etc. para os mais diversos fins. Podemos sumarizar a presença de *linguagem*, em seu sentido mais abrangente (enquanto sistema simbólico), nos sistemas computacionais, conforme os temas a seguir:

- Linguagens formais: a teoria da computação apropriou-se de conceitos específicos da linguística formal para definir a área. A linguística formal pretende descrever as linguagens seguindo o formalismo dos modelos matemáticos. Por exemplo, Noam Chomsky classificou as gramáticas formais em quatro tipos que compõem uma hierarquia (CHOMSKY, 1956). Esta classificação serviu de base para os estudos de teoria da computação, linguagens de programação e compiladores.
- Linguagens de programação: são as formas mais “imediatas” de se pensar em linguagem quando se fala em computação. As linguagens de programação são a forma como os profissionais de computação “se comunicam” com a máquina. São linguagens suficientemente compreensíveis por humanos e que posteriormente são traduzidas (por compiladores) em linguagem de máquina (ao fim e ao cabo, zeros e uns).
- Processamento de linguagem natural (PLN): é uma área de pesquisa cujo objetivo é traduzir a linguagem natural (falada ou escrita) em modelos formais e tratá-la computacionalmente. Particularmente de interesse da Inteligência Artificial, o PLN está presente em aplicações como tradução automática de textos e reconhecimento de fala.
- Comunicação humana em linguagem natural: uma das principais e mais atuais aplicações dos sistemas computacionais em geral é o seu uso como ferramenta de comunicação. Várias tecnologias e sistemas têm sido

desenvolvidos dentro da área de Comunicação Mediada por Computador (CMC), gerando bastante interesse das pessoas em se comunicarem através delas. Proporcionalmente ao interesse dos usuários por ferramentas de CMC, elevou-se também o interesse dos pesquisadores de ciência da computação, linguística, psicologia e educação em investigar os fenômenos particulares deste tipo de comunicação.

- Linguagens de interface e interação: a Engenharia Semiótica (DE SOUZA, 2005) é uma teoria semiótica que define a interação entre humanos e sistemas como um processo bem definido de comunicação. Segundo esta teoria, os signos de uma interface de sistema, além de definirem em si as possibilidades e limites de interação entre o usuário e o sistema, carregam também uma mensagem do produtor desta tecnologia sobre do que ela se trata. Em outras palavras, a interação humano-computador (IHC) é um caso especial de metacomunicação sobre como uma determinada comunicação foi projetada para atingir determinados efeitos. O foco da IHC em comunicação leva à constatação de que a interface corresponde a uma linguagem própria. Existe uma linguagem de interação que deve ser compartilhada entre o projetista e seu usuário para que a comunicação ocorra de forma satisfatória.

A nossa pesquisa segue a visão da Engenharia Semiótica, focando no processo de comunicação envolvido na criação de sistemas computacionais. Ao longo do texto, muito será discutido sobre linguagem, mas com uma exploração diferente em relação aos tópicos listados acima. Os três primeiros não serão abordados e os dois últimos serão constantemente considerados.

2.2 Programação e raciocínio computacional

Tão logo os computadores deixaram de ser artigos restritos a instituições de pesquisa, iniciaram-se os esforços em torno do ensino de programação para além de cientistas e engenheiros da computação. Uma das iniciativas pioneiras foi o desenvolvimento da linguagem Logo, um dialeto de Lisp criado como uma ferramenta de aprendizagem de programação (PAPERT, 1972) (PAPERT, 1980). Desde então, dezenas de linguagens e ferramentas têm sido desenvolvidas para estimular a aprendizagem de programação através da construção de jogos, simulações, animações 2D e 3D, narrativas digitais etc. Algumas ferramentas populares são: variantes do Logo (Scratch¹, NetLogo², StarLogo³), Alice⁴,

¹ <http://scratch.mit.edu/>

² <https://ccl.northwestern.edu/netlogo/>

³ <http://education.mit.edu/starlogo/>

⁴ <http://www.alice.org/>

AgentSheets⁵, Stagecast⁶, Kodu⁷ e Greenfoot⁸. Além de ambientes de programação para *software*, há ainda iniciativas em torno das áreas de robótica e projetos como o *Computer Science Unplugged*⁹, compreendido como uma coleção de atividades que ensina ciência da computação sem o uso real do computador.

Inúmeros artigos científicos têm reportado resultados (positivos principalmente, mas também negativos) da aplicação destas e de outras ferramentas no ensino de programação e de outros conceitos de ciência da computação para os mais diversos públicos: calouros em cursos de ciência da computação (DALY, 2011) (GALLANT e MAHMOUD, 2008) (MISHRA, BALAN, *et al.*, 2014), estudantes de ensino fundamental e médio (BENNETT, KOH e REPENNING, 2011) (SMITH, SUTCLIFFE e SANDVIK, 2014) e meninas (KELLEHER, PAUSCH e KIESLER, 2007) (WERNER, CAMPE e DENNER, 2005). Algumas iniciativas parecem ainda mais ousadas como, por exemplo, ensinar conceitos básicos de computação a estudantes universitários de outras áreas (GKIOLMAS, PAPAConstantinou, *et al.*, 2014) ou até mesmo para crianças menores de cinco anos (FLANNERY, SILVERMAN, *et al.*, 2013).

Dentro deste contexto, Jeannette Wing (2006) definiu o conceito de *Computational Thinking*¹⁰, que foi rapidamente incorporado aos discursos e práticas dos interessados no ensino de conceitos de ciência da computação. Em poucas palavras, para Wing raciocínio computacional significa “pensar como um cientista da computação”. De forma mais completa e numa revisão de sua definição, Wing (2014) afirma: “*Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer – human or machine – can effectively carry out.*”

Raciocínio computacional é visto, por Wing e outros pesquisadores¹¹, como uma habilidade básica e imprescindível para o completo letramento das pessoas (SMITH, SUTCLIFFE e SANDVIK, 2014), junto a ler, escrever e contar. Em outras palavras, para o indivíduo atuar plenamente na sociedade, além dessas três

⁵ <http://www.agentsheets.com/>

⁶ <http://www.stagecast.com/>

⁷ <http://www.kodugamelab.com/>

⁸ <http://www.greenfoot.org/>

⁹ <http://csunplugged.org/>

¹⁰ Apesar de “thinking” significar “pensamento” e “reasoning” significar “raciocínio”, em português traduz-se “computational thinking” para “raciocínio computacional”.

¹¹ Por exemplo, (LU e FLETCHER, 2009)

habilidades (conhecidas como os “três R’s”¹²), ele precisa, hoje, “pensar computacionalmente”. Desta forma, iniciativas de fomento ao ensino de computação para estudantes, por exemplo, visam não necessariamente recrutar futuros profissionais para cursos de computação (o que em muitos casos é uma motivação justificável), mas também capacitá-los para tirar proveito desta habilidade em suas vidas cotidianas.

Alinhados com a visão da Engenharia Semiótica de que criadores de sistemas estão constantemente se comunicando com seus usuários, consideramos que a aquisição de raciocínio computacional por parte do *designer* é uma condição determinante para que a comunicação ocorra efetivamente. Em outras palavras, uma das prerrogativas de uma boa comunicação é o domínio da linguagem adotada; então, também na comunicação via *software* é necessária certa fluência na linguagem computacional, que vai além da habilidade de saber programar.

Assim, consideramos o raciocínio computacional uma habilidade que não se limita à capacidade de solucionar problemas de forma lógica e racional, mas que também, e principalmente, capacita o indivíduo a se comunicar via *software*, tirando o máximo proveito das potencialidades deste tipo de linguagem para construir mensagens computacionais que correspondam satisfatoriamente a suas intenções comunicativas.

Do ponto de vista da linguística, há um recente movimento em direção ao conceitos de letramentos (no plural), letramentos múltiplos ou multiletramentos, que remetem a ideia de que não há (mais) uma forma de ler e escrever textos para se atuar de forma plena na sociedade.

Assim, devido à heterogeneidade das práticas sociais de leitura e escrita que circulam na sociedade atual, entendemos que, ao invés de letramento, teríamos letramentos, ou melhor, letramentos múltiplos, ou seja, as mais variadas práticas existentes de leitura e de escrita que circulam na sociedade, sejam escolares ou não escolares, sejam locais ou globais, valorizadas ou não valorizadas. (LEITE e BOTELHO, 2011, p. 13)

Neste sentido, as pesquisas deste novo movimento consideram, por exemplo, a análise de imagens que compõem os textos (DO NASCIMENTO, BEZERRA e HEBERLE, 2011) ou a leitura hipertextual e hipermidiática comum nos ambientes virtuais (MARTINS e IGNÁCIO, 2013).

¹² Os três R’s vêm das palavras em inglês *Reading*, *wRiting* e *aRithmetic*.

Ao discutirmos sobre o ensino de raciocínio computacional a fim de capacitar os aprendizes a se expressarem através de uma nova linguagem estamos também falando de um novo tipo de letramento a ser considerado entre os múltiplos letramentos já endereçados pelos linguistas e estudiosos da área.

2.3

EU(*)

Uma inevitável consequência da penetração dos computadores nas mais diversas atividades cotidianas é o crescimento do número de usuários que assumem uma posição ativa em relação ao consumo de sistemas computacionais. Muitas vezes, o usuário não consegue concretizar suas intenções de uso apenas através da interface original dos sistemas. Então, ele parte para atividades como customização, personalização e extensão do sistema, até o extremo de, no caso de alguns, criar um *software* específico que atenda a suas necessidades. Em todos os casos, embora em diferentes proporções, ele atua como *produtor* de tecnologia. Cronologicamente, as pesquisas dedicadas a este tipo especial de *end-user activity* foi mudando de nome conforme foram considerados níveis mais abrangentes de atividades. Assim, a área iniciou-se com o termo *End-User Programming* (EUP), dedicado exclusivamente à atividade de programação (codificação). Ko *et al* (2011) definem essa expressão como “*programming to achieve the result of a program primarily for personal, rather than public use.*” (p. 21). E esclarecem que esta distinção reside no fato de que “*the program itself is not primarily intended for use by a large number of users with varying needs.*” (p. 21). Percebe-se então o contraste entre EUP e a programação “clássica”, que, segundo os autores, “*has the goal of producing code for others to use.*” (p. 21)

Em seguida, surgiu *End-User Development* (EUD), um termo mais genérico, como uma alternativa a EUP. Uma definição mais detalhada da expressão é: “*EUD can be defined as a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify, or extend a software artifact.*” (LIEBERMAN, PATERNÒ e WULF, 2006, p. 2). A partir desta visão, compreende-se que “*end users are involved in a system design, but may or may not be involved in its actual coding*” (KO, ABRAHAM, *et al.*, 2011, p. 21).

Repenning e Ioannidou (2006) também questionam essa mudança de termo, acrescentando que, com o alargamento do significado, novas questões emergem:

The shift from “programming” to “development” reflects the emerging awareness that, while the process of adapting a computer to the needs of a user may include some form of programming, it certainly is not limited to it. In that sense, most of the research questions from EUP carry over to EUD but because of the widened scope of EUD new issues need to be explored. (2006, p. 52)

Voltando à questão do *design* para si mesmo versus *design* para terceiros, Ko e colegas (2011) deixam claro que esta não é uma visão tão estável: “Therefore, the moment novice web designers move from designing a web page for themselves to designing a web page for someone else, the nature of their activity has changed.” (p. 21). Neste caso, conforme aumenta a quantidade de usuários visados, por exemplo, da *web page* desenvolvida por um *end-user*, este designer “will have to increasingly consider software engineering concerns in order to satisfy increasingly complex and diverse constraints.” (p. 21)

Esta é então a motivação para o surgimento do terceiro termo, *End-User Software Engineering* (EUSE), em poucas palavras definido como “a highly integrated and incremental concept of software engineering support for end-users” (BURNETT, ROTHERMEL e COOK, 2006, p. 88). Ko e coautores (2011) explicam que o que difere a engenharia de *software* “clássica” de EUSE é justamente a quantidade de atenção dada às questões de qualidade do *software*, envolvendo atividades como testes e manutenção de código. Não é surpreendente então que “people who are engaging in end-user programming rarely have the time or interest in systematic and disciplined software engineering activities.” (KO, ABRAHAM, et al., 2011, p. 21).

Alinhado com os esforços de pesquisa em torno do *end-user developer*, o Grupo de Pesquisa em Engenharia Semiótica (SERG) tem estendido suas investigações para além do *designer* profissional. Referimo-nos a *End-User Semiotic Engineering* (EUME)¹³. Neste caso, os conceitos da engenharia semiótica, segundo a qual a interação humano-computador é um tipo especial de comunicação mediada por computador entre *designers* e usuários, são também aplicáveis ao usuário final que atua como *designer* no contexto de EUD. Isto é, o *designer*, seja ele um desenvolvedor de *software* ou um usuário final, realiza a

¹³ À primeira vista, a melhor sigla para o termo seria EUSE, mas ela já está em uso na expressão *End-User Software Engineering*. Além disso, o “me” de EUME também remete a “eu” em inglês.

construção da metacomunicação através da criação, escolha e disposição dos signos na interface, engajando-se num genuíno processo de *engenharia semiótica*. A Engenharia Semiótica defende que é importante os projetistas terem consciência de que eles estão se comunicando com o usuário em tempo de interação, para terem sucesso em suas intenções comunicativas. Da mesma forma, o usuário final atuando como *designer* também pode se beneficiar deste *design* centrado na comunicação.

2.4

Computador como mídia

Ao longo da história da ciência da computação e em especial da Interação Humano-Computador, a visão dos sistemas computacionais evoluiu da *perspectiva de sistema*, em que a interação era vista como uma mera transmissão de dados entre pessoas e sistemas computacionais, até chegar à *perspectiva de mídia*, segundo a qual, a interação é vista como um processo comunicativo entre pessoas (KAMMERGAARD, 1988). No meio do caminho, outras duas perspectivas também foram consideradas: *perspectiva de parceiro do discurso*, em que a interação era vista como um processo de comunicação entre sistema e usuário, no qual o sistema tenta “se passar” por uma pessoa, e *perspectiva de ferramenta*, segundo a qual os sistemas são vistos como ferramentas que auxiliam os usuários a realizarem suas tarefas.

A *perspectiva de mídia* é tradicionalmente caracterizada em dois tipos (BARBOSA e DA SILVA, 2010): aquela em que os usuários conversam diretamente entre si, através de sistemas como e-mail, fóruns e *chats*; e aquela em que os *designers* conversam indiretamente com os usuários, através da interface; visão especialmente difundida pela Engenharia Semiótica (DE SOUZA, 2005). Estes dois tipos caracterizam-se como “comunicação mediada por computador”.

A visão do computador como mídia está intrinsecamente relacionada à nossa questão de pesquisa discutida ao longo da tese. Nos estudos empíricos realizados, convidamos os participantes a comunicarem-se via *software* com seus interlocutores. Ou seja, falamos de um misto dos dois tipos descritos acima: temos sim dois usuários se comunicando através do computador (primeiro tipo), mas desta vez um destes usuários atua como *designer* (segundo tipo), pois sua mensagem é passada através de um programa que ele mesmo fez.

Em capítulos subsequentes teremos oportunidade de discutir em detalhes como se caracteriza este tipo especial de “comunicação mediada por computador” em que um usuário-final atua como *designer* de uma mensagem interativa.

2.5 Engenharia Semiótica

A Engenharia Semiótica é uma teoria de base semiótica desenvolvida dentro do contexto da área de Interação Humano-Computador (IHC). Esta teoria vê a IHC como uma comunicação especial entre aqueles que desenvolvem sistemas computacionais e aqueles que os utilizam. E é justamente esta comunicação entre *designers* e usuários que é o foco de investigação da Engenharia Semiótica.

Segundo a concepção formal da Engenharia Semiótica (DE SOUZA, 2005, p. 83), IHC é um tipo particular de metacomunicação mediada por computador em dois níveis. Tecnicamente, metacomunicação significa comunicação *sobre* (aspectos de) comunicação. A mensagem passada na metacomunicação nos diz como interpretar esta mensagem. IHC é um tipo particular de metacomunicação mediada por computador em dois níveis na qual *designers* de sistemas computacionais enviam a usuários de sistemas uma mensagem única. A mensagem diz aos usuários como eles devem recebê-la e interpretá-la, a fim de se obter uma certa gama de efeitos. É uma mensagem única porque, do ponto de vista de *design*, ela transmite um conteúdo completo e imutável codificado na e disponibilizado pela interface do sistema. (DE SOUZA, 2005, p. 84). Em outras palavras, a interface está dizendo aos usuários uma importante mensagem sobre como eles podem ou devem usar o sistema, por que e para que efeitos. A mensagem é construída de uma forma tal que comunique aos usuários o que pode ser parafraseado pelo seguinte *template* genérico – *template de metacomunicação*:

Este é o meu entendimento de quem você é, do que aprendi que você quer ou precisa fazer, de que maneiras prefere fazer, e por quê. Este, portanto, é o sistema que projetei para você, e esta é a forma como você pode ou deve utilizá-lo para alcançar uma gama de objetivos que se encaixam nesta visão. (DE SOUZA, 2005, p. 84) (DE SOUZA e LEITÃO, 2009, p. 16)

O “eu” – primeira pessoa do discurso – nesta paráfrase é o *designer* do sistema (ou alguém que fale pela equipe de *design*) e o “você” – segunda pessoa do discurso – refere-se ao usuário (ou à população de usuários) (DE SOUZA e LEITÃO, 2009, p. 16). A metacomunicação é em dois níveis porque ela só é

efetivada se os usuários se comunicarem via mensagem. Se isto não ocorre, a metacomunicação não é alcançada, porque o signo que representa a intenção do *designer* não é interpretado como tal. O nível de metacomunicação *designer*-usuário determina a comunicação usuário-sistema. O *template* de metacomunicação resume o que os *designers* estão comunicando ao usuário através da interface. A Figura 2-1 ilustra essa comunicação.



Figura 2-1. Comunicação entre *designer* e usuário

A mensagem única do *designer* para o usuário é progressivamente desempacotada e interpretada, conforme ele vai se comunicando com o sistema. Para a metacomunicação proceder de forma consistente e coesa, o sistema deve *falar pelo designer* (DE SOUZA, 2005, p. 90), que não está fisicamente presente em tempo de interação. Então, os *designers* não estão, eles mesmos, entregando suas mensagens aos usuários. Por isso, o sistema os representa em tempo de interação (DE SOUZA e LEITÃO, 2009, p. 17), assumindo então o papel de *preposto do designer* – um agente comunicacional que está apto a passar a mensagem do *designer*. Portanto, a distinção fundacional proposta pela Engenharia Semiótica é que IHC não é exatamente sobre como usuários interagem com computadores, mas sobre como usuários se comunicam com *designers* de sistemas computacionais através do seu representante em tempo de interação, o preposto do *designer*. (DE SOUZA e LEITÃO, 2009, p. vii) Consequentemente, o discurso interativo do sistema é uma versão computacional das conversas que os *designers* teriam com os usuários a fim de atingir a principal intenção comunicativa no *design*, que seria o usuário entender, gostar e se beneficiar do produto de *design* (p. 16).

Há três classes distintas de signos no discurso interativo do preposto do *designer* (DE SOUZA e LEITÃO, 2009, p. 19): signos estáticos, signos dinâmicos e signos metalinguísticos. Os primeiros são signos da interface cujo significado é interpretado independentemente de relações temporais ou causais. O contexto de interpretação é limitado aos elementos que são apresentados na interface em um

único instante no tempo. Por exemplo, estruturas de leiaute, opções de menu e botões em barras de ferramentas são todos signos estáticos.

Signos dinâmicos estão ligados a aspectos temporais e causais da interface, ou seja, da própria interação em si. Eles surgem da interação e devem ser interpretados com referência a ela. Signos estáticos e dinâmicos estão intrinsecamente relacionados. Os primeiros estimulam o usuário a se engajar na interação com o artefato; eles ajudam o usuário a antecipar o que a interação vai trazer. Os signos dinâmicos, por sua vez, confirmam ou desconfirmam esta antecipação do usuário. (p. 19)

O significado dos signos estáticos e dinâmicos é explicitamente informado, ilustrado ou explicado por signos de uma outra classe – metalinguísticos. Eles se referem a outros signos da interface, estáticos, dinâmicos ou até outros metalinguísticos (em referência recursiva). Normalmente eles aparecem na forma de mensagens de ajuda e de erros, alertas, diálogos de esclarecimentos, dicas de tela, etc. Com os signos metalinguísticos, os *designers* comunicam explicitamente os significados codificados no sistema e como eles podem ser usados. (p. 19)

É, portanto, através de uma engenharia de signos que os *designers* concebem a interface de forma a comunicar de maneira consistente e efetiva suas intenções aos usuários. Em outras palavras, o *designer* além de *produzir* sistemas interativos, também deve *apresentá-lo* ao usuário durante a interação, através de uma boa comunicabilidade.

A comunicabilidade diz respeito à capacidade da interface de comunicar ao usuário a *lógica de design*: as intenções do *designer* e os princípios de interação resultantes das decisões tomadas durante todo o processo de *design*. Se um usuário for capaz de compreender a lógica utilizada na concepção do sistema interativo, terá maiores chances de fazer um uso criativo, eficiente e produtivo dele (BARBOSA e DA SILVA, 2010, p. 38). Atualmente, a Engenharia Semiótica dispõe de dois métodos para investigar a comunicabilidade dos sistemas: o método de inspeção semiótica (MIS), focado na *emissão* da metacomunicação e o método de avaliação de comunicabilidade (MAC), focado na *recepção* desta mensagem. (DE SOUZA e LEITÃO, 2009; BARBOSA e DA SILVA, 2010)

Ao longo de seu desenvolvimento e amadurecimento, a Engenharia Semiótica tem se mostrado robusta principalmente para a *pesquisa em IHC*, mas também igualmente forte para a *prática de IHC*. Além de todo o ferramental

teórico descrito acima, a Engenharia Semiótica dispõe de métodos e técnicas para o *design* e a avaliação de sistemas computacionais interativos, com o foco na comunicação. Entretanto, todo esse arcabouço teórico e prático tem sido concebido a partir de uma abordagem *profissional*. Ou seja, a Engenharia Semiótica defende que um *design profissional* centrado na comunicação favorece uma boa comunicabilidade entre os *designers* e os usuários, o que, em situações específicas, caracteriza-se como um fator de qualidade dos sistemas (BARBOSA e DA SILVA, 2010, p. 38). Em consequência disto, para os praticantes de IHC (*designers*, avaliadores, pesquisadores etc.) que estão preocupados com este fator de qualidade, é desejável que eles tenham uma formação ou conhecimento mínimo dos conceitos e métodos da Engenharia Semiótica, afim de que seus produtos tenham uma boa comunicabilidade.

A Engenharia Semiótica (substantivo próprio) corresponde, *grosso modo*, a este conjunto de conceitos, ferramentas e métodos disponíveis aos praticantes de IHC. Entretanto, esta mesma teoria prevê a existência da *engenharia semiótica* (substantivo comum), ou engenharia de signos, inerente à construção de qualquer sistema, como um processo nato por parte dos *designers* dos sistemas, justamente porque ela entende a interação humano-computador como uma comunicação entre dois grupos de pessoas (*designers* e usuários). Ora, a comunicação humana está intimamente ligada à geração e interpretação dos signos, pois eles são os elementos “atômicos” de qualquer linguagem, que é um dos recursos humanos que viabilizam a comunicação. Por este raciocínio, é trivial concluir que mesmo aqueles que não possuem qualquer formação em *Engenharia Semiótica* estão sempre realizando uma *engenharia semiótica* quando desenvolvem sistemas computacionais interativos.

Ao longo desta tese, apresentaremos casos de *engenharia semiótica*, mas focando em *designers* específicos, aqueles que naturalmente são usuários finais interagindo com ferramentas de EUD para construírem artefatos interativos com o objetivo de se comunicarem com outra pessoa.

2.6 Autoexpressão

We express ourselves in many ways: through tone of voice, posture, the face, words, and, in more subtle cases, paint, music, sculpture, and other forms of art. (GREEN, 2007, p. 1)

Autoexpressão é um fenômeno ligado à comunicação humana que está intimamente relacionado à personalidade, individualidade e idiossincrasias do indivíduo. Diversas áreas das ciências sociais e humanas investigam autoexpressão. Podemos citar trabalhos em linguística (AAKER, 1999), psicologia (LANDAU, VESS, *et al.*, 2011), educação (TOBIN, 1995), artes (OSKAR, 1997) e até política (SAKR, 2004). Entretanto, devido à visão do computador como mídia, a ciência da computação também tem voltado o olhar para elementos próprios da comunicação, como é o caso da autoexpressão. Como será apresentado adiante, autoexpressão é o principal fenômeno de comunicação investigado nesta tese. É necessário então expormos algumas definições a respeito deste assunto que serão úteis ao longo do texto, especialmente quando formos apresentar as evidências coletadas nos estudos empíricos.

Segundo Green, “*In expressing ourselves we manifest some part of our point of view*” (GREEN, 2007, p. 1). Mais precisamente, este autor defende autoexpressão “*as a matter of signaling one’s thought, affect, or experience*”. (GREEN, 2007, p. 15). Resumidamente, esta é a caracterização formal da autoexpressão trazida pelo autor: “*Where A is an agent and B a cognitive, affective, or experiential state of a sort to which A can have introspective access, A expresses her B if and only if A is in state B, and some action or behavior of A’s both shows and signals her B*”. (p. 43). Este autor apresenta, portanto, uma caracterização genérica de autoexpressão, aplicável a qualquer forma de representação que “sinalize” algo interno a um indivíduo.

A linguagem, como um conjunto de representações simbólicas aplicado à comunicação, é, portanto, um campo fértil de autoexpressão. Neste sentido e retomando o que foi levantado na primeira seção deste capítulo sobre a presença da linguagem na computação em geral, acreditamos que uma aproximação da questão da autoexpressão focada na linguagem pode ser esclarecedora.

Buscamos suporte, então, no trabalho da linguista Barbara Johnstone (1996). Esta autora defende que não se é possível estudar a linguística sem

considerar o indivíduo. Este se apropria da linguagem e constrói sua própria linguística e esta apropriação faz parte da constituição da personalidade do indivíduo: “*People know each other because each has a unique way of sounding, an individual voice*” (p. 7). A autora acredita na exclusividade do uso da língua e explica que “*it is impossible that two people could do things with language the same way.*” (p. 8) Ao nos direcionarmos aos sistemas computacionais, especialmente aqueles que permitem a atuação de usuários finais como *designers* (sistemas EUD), podemos estender a afirmativa da autora e postular provisoriamente que nenhum *end-user* usa a linguagem daquele sistema da mesma forma. Como veremos no decorrer da tese, os produtos construídos pelos participantes dos estudos empíricos carregam toda a individualidade dos seus criadores, refletida na maneira como eles se apropriaram dos recursos de linguagem disponibilizados pelo ambiente EUD.

Johnstone explica que, independente do valor que as culturas dão ao individualismo, sempre há autoexpressão quando as pessoas se socializam. Segundo a autora, uma sociedade cujos membros possuíssem “a mesma voz” seria inconcebível: “*people express themselves with everything they do, whether or not self-expression is at the moment or in the context particularly valued or even though relevant at all. A society in which everyone always talked alike, in which people were linguistically indistinguishable from one another, would be unthinkable*” (p. 7). É com este raciocínio que ela descreve o “indivíduo linguístico”, ou seja, para além das convenções de uma língua comum a uma sociedade, é o uso não convencional que define a linguística individual.

Assim como Johnstone acredita que “*paying attention to individual voice helps us to understand language*” (p. 22), nós também entendemos que o escrutínio das “vozes” daqueles que constroem artefatos computacionais através de sistemas EUD nos permite aprender sobre estes próprios sistemas. De forma análoga às linguagens verbais, para estes sistemas, o usuário deve ser capaz de captar a linguagem proposta e se apropriar dela para construir sua mensagem.

Adicionalmente, no livro de Johnstone, ela define seu interesse de pesquisa da seguinte forma: “*I want then to examine one of the things people use their linguistic resources for: as a way to express their autonomous, unique selves*” (p. 28). Ao longo da pesquisa apresentada nesta tese, foi bastante instigante observar

as pessoas revelando seus *unique selves* nos discursos digitais construídos, como será largamente discutido adiante.

Além da área de linguística, destacamos aqui dois trabalhos mais alinhados com educação e psicologia, que também investigam o fenômeno da autoexpressão no indivíduo. Baraldi (2008), ao analisar crianças, relaciona “self-expression” com “self-socialization”: *“Self-expression may be considered a demonstration of competence in personal choice acquired through self-socialization.”* (p. 239). O autor esclarece como isso ocorre:

Self-socialization becomes socially evident only through self-expression. In communication processes, self-expression provides cues for self-socialization because it indicates that children autonomously create their own meanings, while adaptation to external rules and models indicates that children simply reproduce generalized social meanings. Self-expression indicates that the origins and conditions of actions lie inside the individual. (p. 240)

Esta relação explicita a importância da autoexpressão, pois ela se reflete na vida social do indivíduo, em especial das crianças, no caso do estudo citado. Landau *et al* (2011) referem-se a “*intrinsic self-expression*”, como uma influência no comportamento interpessoal. Segundo eles, “*the intrinsic self [is characterized] as an ‘inner core’ which expands (under facilitating conditions) to express its true nature and responds to perceived social threats by seeking shelter behind external ‘façades.’*” (p. 79)

Trazendo a discussão para mais próximo dos sistemas computacionais, Sherry Turkle (2005) desenvolveu uma pesquisa bastante extensa com o objetivo de investigar o impacto da tecnologia na vida das pessoas, mas em termos mais psicológicos. Ela traz reflexões do tipo: “*That question is not what will the computer be like in the future, but instead, what will we be like? What kind of people are we becoming?*” (p. 19)”. A autora fala claramente da perspectiva de mídia dos sistemas: “*I choose to look at settings where the computer can be taken up as an expressive medium.*” (p. 20). Turkle acredita que, em certa medida, os computadores refletem a mente das pessoas, pois “*the machine can act as a projection of part of the self, a mirror of the mind.*” (p. 20). Em sintonia com a ideia de autoexpressão via *software*, a autora afirma: “*a computer program is a reflection of its programmer’s mind. If you are the one who wrote it, then working with it can mean getting to know yourself differently*” (p. 24).

Investigando o uso dos *blogs* como ferramentas para educação, Deng e Yuen (2011) definem autoexpressão como a seguir: “*Self-expression involves expressing one’s thoughts and emotions, as well as capturing or recording one’s experiences.*” (p. 443). Os autores analisaram *posts* de blogs acadêmicos e cruzaram essas informações com os relatos dos alunos a fim de investigar como os blogs potencializaram a autoexpressão e autorreflexão deles. Outro estudo (VAN HOUSE, 2005), a respeito de fotografias tiradas com o celular, adota uma definição mais focada neste domínio: “*By self-expression, we mean the creation of images that express one’s own view of the world – images that are ‘artistic’, funny, experimental, or otherwise expressive*” (p. 3). Este é um estudo interessante para observar a autoexpressão em linguagem não-verbal.

Os estudos empíricos realizados, em maior ou menor grau, demonstraram como a autoexpressão emergiu dos vários *discursos digitais* construídos pelos participantes. Também pudemos ver como os diversos artefatos computacionais refletiram as mentes de seus criadores, além de também, em alguns momentos, revelar o processo de construção de suas próprias identidades. No capítulo sobre a discussão dos resultados, retomaremos a questão da autoexpressão, inclusive retratando outros estudos sobre este tema.

Para finalizar a seção, é importante fazer uma ressalva retórica: conforme visto nos parágrafos acima, a autoexpressão está potencialmente em qualquer parte, intencionalmente ou não. Assim, em última instância, toda expressão é uma expressão de si mesmo (do emissor), o que tornaria desnecessária a utilização do prefixo “auto”. Entretanto, optamos por mantê-lo ao longo de nosso próprio discurso para enfatizar a presença do “eu” do emissor neste tipo especial de mensagem, construída em sistemas de desenvolvimento pelo usuário final.