O Método de Inspeção Semiótica Aplicado ao Requisito Usabilidade

Elizabeth Suescún Monsalve[1], Vera Maria B. Werneck[2], Julio Cesar Sampaio do Prado Leite[1]

[1] Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Rio de Janeiro, Brasil
[2] Universidade do Estado do Rio de Janeiro (UERJ), Rio de Janeiro, Brasil

emonsalve@inf.puc-rio.br, vera@ime.uerj.br, www.inf.puc-rio.br/~julio

Resumo. O requisito usabilidade é de suma importância para os artefatos de software principalmente na área de jogos computacionais, pois permite medir a capacidade de uso de um artefato de software. Classifica-se esse requisito como um requisito de qualidade ou Requisito Não-Funcional (RNFF). Nosso trabalho centra-se na operacionalização do requisito usabilidade no contexto de um jogo educacional usando o Método de Inspeção Semiótica (MIS), oriundo da área de Interação Humano--Computador (IHC). O SimulES-W é a primeira versão digital do jogo SimulES usado para o ensino da engenharia de software de forma lúdica e colaborativa entre os jogadores.

Abstract. The usability requirement is extremely important to software artifacts especially in the area of computer games. This requirement can be classified as a quality or a non-functional requirement. Our work focuses on the operationalization of usability in the context of an educational game using the Semiotics Inspection method from the area of Human--Computer Interaction. SimulES-W is the first digital version of the game SimulES aiming at teaching software engineering in a fun and collaborative way.

Keywords: Inspeção semiótica, requisitos, requisitos não-funcionais, jogos educacionais. Semiotics inspection, requirements, non-functional requirements, educational games

1 Introdução

Jacob Nielsen em seu trabalho "*Usability: Empiricism or Ideology?*" [25] introduz o conceito da usabilidade como um direito dos usuários. Esse direito significa que as pessoas têm prioridade sobre à tecnologia, ou seja, se houver um conflito entre tecnologia e pessoas, então a tecnologia deve mudar. Para isso, ele nos faz refletir sobre o conceito de controle, onde o usuário tem que entender o que acontece com software e deve ter a capacidade de controlar o resultado. Além disso, o usuário tem direito à simplicidade, ou seja, usuários devem conseguir o que desejam sem desnecessária complexidade e finalmente o direito dos usuários de ter seu tempo respeitado. Por outro lado, a transparência de software, segundo Leite [17], será um direito que a sociedade irá a exigir, no qual o software terá que ser transparente. Um dos elementos necessários para atingir a transparência é que o software deverá atender ao RNF (Requisito Não-Funcional) de usabilidade.

Num jogo computacional com fins educacionais ou para treinamento em situações da prática [20] a interface do usuário é um fator preponderante sendo responsável pela comunicação entre o usuário e o sistema (IHC Interação Humano-Computador) [1]. É por isso que seu desenho cumpre um papel importante na mediação e nas relações cognitivas. O entendimento da relação desenho de interfaces e lógica do usuário possibilita a criação e melhorias nas interfaces que facilitem a realização dos objetivos dos usuários. Conforme descrito em Norman e Draper [1], na IHC se faz necessário simplificar o esforço que o usuário tem que fazer para realizar suas tarefas e para isso as interações devem ser revisadas explicitamente num processo de avaliação do software. Nesse trabalho utilizaremos a inspeção semiótica apresentada na Engenharia Semiótica [2] a qual se propõe estudar o processo de significação e comunicação, onde a interface é considerada a ferramenta mediadora entre usuário e computador.

Como caso de estudo foi utilizado o SimulES-W, um jogo de tabuleiro e cartas, que tem como objetivo proporcionar aos alunos um mecanismo de aprendizado diferente dos utilizados nas aulas tradicionais de ensino da engenharia de software simulando e explorando situações vividas na prática de um ambiente de desenvolvimento de software.Várias versões do SimulES foram criadas [6], [8] e [11] até a concepção mais recente; a versão digital que utiliza a Internet denominada SimulES-W. Este trabalho mostra como foi aplicado o Método de Inspeção Semiótica (MIS) ao SimulES-W permitindo a identificação e verificação da comunicabilidade desse software. Em Peixoto, Prates e Resende [21] é descrito como o MIS pode ser usado para avaliar jogos como o SimulES, podendo ser utilizado como um mecanismo para melhorar este tipo de software.

O artigo está organizado em 6 Seções: a Seção 2 apresenta o MIS. Na Seção 3 é explicado o SimulES-W. Na Seção 4 é mostrado a aplicação do método MIS para avaliar a comunicabilidade do SimulES-W. Na Seção 5 são discutidos os aspectos de melhorias identificados. E finalmente na Seção 6 são apresentados as conclusões e trabalhos futuros.

2 Método de Inspeção Semiótica

A Engenharia Semiótica é uma abordagem da IHC que propõe que o engenheiro de software[1] do sistema computacional seja um ator ativo na comunicação com o usuário. Segundo Souza [2], os engenheiros e usuários são interlocutores de um processo global de comunicação onde está envolvida uma interface que contém palavras, gráficos e comportamentos. Os engenheiros devem informar aos usuários o que eles pretendem comunicar com o artefato que estão criando, e os usuários devem compreender e responder satisfatoriamente ao que está sendo dito através da interface. Esse processo é possível mediante um acoplamento entre a teoria semiótica e engenharia. O foco de Souza [2] para IHC engloba os princípios, os materiais, os processos e as possibilidades para a produção do discurso em sistemas de computador interativos de forma significativa numa perspectiva mais ampla do que as abordagens cognitivas, etnográficas ou ergonômicas. Por isso este método pode ser aplicado especialmente para a avaliação da IHC desde a perspectiva do desenho de sistemas até as ajudas *on-line*, customização e desenvolvimento para o usuário final.

A Engenharia Semiótica se vale de dois métodos a fim de avaliar a comunicabilidade dos sistemas computacionais. O primeiro é o MIS que permite avaliar o artefato computacional desde a perspectiva do engenheiro, ou seja, o engenheiro como um ator ativo através de sua metacomunicação com o usuário. Esta avaliação é realizada através de uma verificação realizada pelo próprio engenheiro ou por um inspetor com objetivo de avaliar a interface, sem ter que envolver o usuário no processo. E por último se tem um segundo método através de uma validação, chamado de MAC (Método de Avaliação de Comunicabilidade) que envolve o usuário e a utilização do artefato; o que leva a desenvolver testes de usuário que avaliem aspectos da interface. Este trabalho é centrado na aplicação do método MIS da Engenharia Semiótica.

Os métodos de avaliação em IHC são propostas que permitem verificar e validar a qualidade dos sistemas interativos e estratégias específicas de arquitetura [3]. O foco do MIS conforme Souza et al [3] permite que um inspetor possa analisar a comunicabilidade da interação dos artefatos baseados em computador. No entanto, as descobertas produzidas com a aplicação do método MIS têm como objetivo melhorar a interface de usuário e por tanto a interação. A idéia de comunicabilidade na Engenharia Semiótica proposta por de Souza et al [3] é apresentada como uma mensagem na interface de usuário enviada pelo engenheiro para o usuário conforme sua visão de engenheiro. Ou seja, como produto atende às necessidades do usuário, baseado nos benefícios e valores que o produto possa trazer para a vida do usuário. A Engenharia Semiótica se baseia na avaliação dos signos os quais estão presentes na interface de usuário mediante ícones, símbolos e índices. Do mesmo modo, estes signos são classificados como: signos estáticos, signos dinâmicos e signos metalinguísticos.

[1] Utilizaremos no artigo a palavra “engenheiro” em substituição à expressão “engenheiro de software”.

Os **signos estáticos** não têm movimento e persistem mesmo que a interação entre o usuário e o computador não esteja acontecendo. Esses signos são independentemente das relações temporais e causais. Os **signos dinâmicos** apresentam uma transformação como resposta à interação do usuário, eles são atualizáveis ao longo do tempo e perdem sua significância fora da dimensão temporal, já que apresentam as transições na interface. E finalmente os **signos metalingüísticos** são signos que representam outros signos sejam estáticos, dinâmicos ou metalinguísticos. Por exemplo, o signo "ajuda", possui a seguinte mensagem "pressione certa tecla para obter ajuda".

Na aplicação do método MIS inicialmente tem se uma fase prévia chamada de preparação, que envolve um levantamento da documentação existente. O método deve ser aplicado em 5 etapas. Na primeira são analisados os signos metalingüísticos, na segunda os signos estáticos, na terceira os signos dinâmicos.

Nestas três primeiras etapas é sugerido que as seguintes frases sejam utilizadas: (i) Aqui está o meu entendimento de quem você é. (ii) O que eu aprendi que você quer ou precisa fazer, de qual jeito você prefere fazer e por que. (iii) Este é o sistema que eu projetei para você e este é o jeito que você pode ou deve usá-lo para satisfazer seus propósitos que casam com esta visão. No decorrer do artigo essas frases serão apresentadas entre colchetes e representam o esquema de metacomunicação.

Na quarta etapa é feita uma comparação entre a mensagem de metacomunicação do engenheiro gerada nos passos anteriores. E finalmente na última etapa se faz uma avaliação da comunicabilidade do sistema inspecionado.

Assim, a aplicação do MIS para avaliar o SimulES-W tem por objetivo analisar a operacionalização do RFN de usabilidade. A seguir será discutido a usabilidade como um requisto não funcional para depois contextualizar o entendimento de SimulES-W e apresentar como o MIS foi aplicado.

3 RFN de Usabilidade

A usabilidade é um requisito que reflete a facilidade de uso do artefato software [4] e o grau em que o desenho desse artefato facilita ou dificulta seu uso.

Concordamos com Cappelli em [27] quando define que a transparência precisa da facilidade de uso para ser atingida e que esta pode ser identificada conforme a avaliação comparativa de práticas que implementem características como uniformidade, intuitividade, simplicidade, amigabilidade e compreensibilidade. Esses elementos definidos no trabalho de Cappelli estão presentes repetidamente nos trabalhos do Nielsen [4] [25].

A Figura 1 apresentada por Cappelli em [27] ilustra a usabilidade e os elementos ou RFNs necessários ou que ajudam para que a usabilidade seja possível.

Uniformidade é a capacidade de manter uma única forma, sendo a operacionalização realizada através da padronização de nomes, uso de estilo de programação, da divisão das partes, das formas de utilização e da apresentação. *Amigabilidade* pode ser definida como a capacidade de uso sem esforço, e para isso técnicas de IHC podem ajudar como a ajuda sensível ao contexto. *Simplicidade* é a capacidade de não apresentar dificuldades ou obstáculos; onde padronizar da documentação, minimizar o uso de variáveis ou termos; cada parte deve possuir uma única função e destacar as funções mais utilizadas são possíveis operacionalizacões para se atingir a simplicidade. *Operabilidade* é a capacidade de estar operacional e para atingi-la o software deve estar hábil para funcionar, permitir diversos níveis de acesso para ser controlado pelo usuário, executar as tarefas previstas nos requisitos funcionais (RF) e ser auto-descritivo e ter capacidade de ser reconhecido como um componente único. *Intuitividade* é a capacidade de ser utilizado sem aprendizado prévio podendo ser utilizado como soluções, por exemplo; uso de padrões conhecidos, utilização de vocabulário mínimo e do domínio ou minimizar tamanho de sentenças ou módulos. *Adaptabilidade* pode ser definida como a capacidade de ser alterado de forma a atender novas necessidades ou mudança de contexto. *Desempenho* está relacionado com a capacidade de operar adequadamente em relação ao tempo e recursos computacionais.

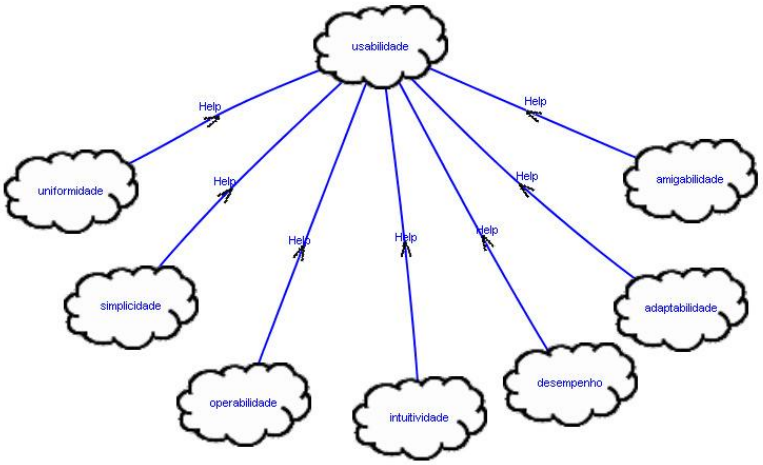

Fig. 1. Usabilidade conforme Cappelli [25].

O objetivo deste trabalho é a operacionalização do RFN de usabilidade. Acreditamos que com o uso do método MIS é possível avaliar como o software está se comunicando; cobrindo assim os atributos da usabilidade que apresentamos na Figura 1.

4 Aplicação da Inspeção Semiótica no SimulES-W

A aplicação da inspeção semiótica no SimulES-W foi realizada através das etapas de preparação e aplicação dos passos do MIS descritos anteriormente na Seção 2 deste trabalho.

4.1 SimulES-W

A seguir o leitor é contextualizado no entendimento do Software sob análise: SimulES-W é um jogo para ensino na engenharia de software. Desde sua concepção a partir do jogo "*Problems and Programmers*" (PnP) [5], o SimulES evoluiu gerando diferentes versões [5][8][11] que englobam sugestões de melhorias e aprimoramentos no jogo. Foi assim que foram identificados requisitos que permitiram uma evolução do SimulES original documentados em [8] e [11]. Relatos da utilização sempre com uma retroalimentação positiva sobre seu potencial de ensino, também, podem ser encontrados em [10][12]. Além disso, a avaliação sobre o uso de SimulES 2.0 [8] que foi publicada em [10] foi utilizada no desenvolvimento da versão do SimulES-W.

O objetivo do SimulES é que cada jogador construa um mesmo produto de software. No jogo o aluno/jogador assume vários papéis, tais como: engenheiro de software, coordenador técnico, controlador de qualidade e gerente de projeto. Dentre as tarefas do jogo, jogador deve lidar com: (i) complexidade e tamanho do produto, (ii) noção de tratamento de qualidade do produto com base na verificação por meio de inspeção, (iii) risco de se ter produtos de má qualidade, (iv) orçamento em termos de valor do projeto, (v) contratação e demissão de engenheiros de software, (vi) visão de recursos em função do valor, produtividade e maturidade das pessoas envolvidas e (vii) construção de diversos artefatos necessários para término do projeto.

A partida termina quando um jogador constrói todos os módulos necessários para completar o projeto com a qualidade estipulada, onde os módulos do projeto podem ter sidos todos inspecionados ou não. A tarefa de inspeção tem um custo relacionado à qualidade do projeto e a habilidade do engenheiro para fazer esta atividade. Se o jogador escolher não inspecionar todos os seus módulos, a inspeção final obrigatória, segundo o critério de qualidade do projeto, poderá ou não encontrar defeitos. Se o número de defeitos encontrado for maior que o permitido, o jogo continua, porque aquele jogador não atingiu a meta. Essa dinâmica onde qualidade, custo e risco são exercitados é um dos pontos principais sob a ótica do aprendizado.

Ou seja, o término do jogo, primeiro jogador a completar o projeto com seus módulos, pode ser usado como a didática para ensino dos conceitos de qualidade, custo e risco.

SimulES-W, implementado em um software que utiliza a Internet, daí o W, é a versão digital do SimulES que incorpora os mesmos cenários, ou seja, "rodadas" que o jogo de tabuleiro. Essas rodadas são apresentadas no jogo digital como telas através das quais os usuários navegam ao longo do jogo. Essas rodadas são: **A jogada de inicio** onde é escolhido aleatoriamente o projeto que deve ser tratado durante todo o jogo, como também o primeiro Engenheiro de Software para cada um dos jogadores. **A jogada de ações** que começa com o lançamento do dado e de acordo com o número tirado, retira 1 a 3 Cartas de Conceitos e Problemas e se o valor do dado for mais de 4 pode pegar também (dado–3) Cartas de Engenheiro de Software. **A jogada de ações** continua e o jogador da vez deve tratar (construir, inspecionar ou empacotar) os artefatos dispostos no tabuleiro individual. Na construção pode retirar novas Cartas de Artefato dependendo da complexidade do projeto e da habilidade de sua equipe de Engenheiros de Software. A habilidade determina quantos "pontos de tempo" ele tem e, portanto, quantas ações ele pode desempenhar por vez. Analogamente ele pode inspecionar dependendo também da habilidade da sua equipe. **Na jogada de conceitos e tratamento de problemas** os jogadores podem melhorar seu jogo (aumentando por exemplo a habilidade da equipe) ou serem atacados ou atacar seus adversários usando as Cartas Conceitos e Cartas Problemas com conceitos típicos da engenharia de software. Esta instância do jogo é importante porque permite que os jogadores analisem tanto as Cartas Conceitos e Cartas Problemas que possuem quanto as que possuem seus adversários para realizar a melhor jogada possível. Essas cartas possuem conhecimento de engenharia de software e devem ser lidas, entendidas e discutidas entre os jogadores. Além disso, o professor (jogando ou "só olhando") pode induzir a uma análise mais profunda através de discussões para fixação dos conceitos envolvidos nas diversas cartas e situações do jogo. Por último, **o produto de software é submetido**, sendo que o primeiro jogador a chegar nesta instância ganha o jogo, se o produto passar pela inspeção final e se que não houver penalidade pendente.

4.2 Passo 1: Preparar Inspeção Semiótica

Esta etapa inicia-se pela preparação onde são listadas as fontes de informação do software tanto *on-line* quando *off-line*. Isso permite ao engenheiro a definição do escopo do problema, do perfil do usuário e do cenário que guiará a análise e aplicação da inspeção semiótica

Como fontes de informação foram utilizados os trabalhos apresentados com as diferentes versões do SimulES [6], [7], [8] e [14], as ajudas fornecidas pelo SimulES-W [9],[10] e informação do SimulES vindo de outras fontes [11] e [12].

Na definição do escopo, o inspetor deverá navegar entre as diferentes páginas do SimulES-W, identificar os diferentes cenários do jogo de tabuleiro e tentar executar as ações dirigidas conforme sua intuitividade.

O perfil dos usuários que no futuro utilizarão o software são alunos de graduação em Ciências da Computação com conhecimentos básicos em Engenharia de Software.

O seguinte Cenário de inspeção foi definido: "Renato é um estudante do curso Ciência da Computação da PUC-Rio e está matriculado na disciplina de PES (Princípios de Engenharia de Software). Nesta disciplina ele aprendeu sobre léxicos e cenários. Cenários como uma técnica que auxilia no entendimento e descrição de uma situação específica de uma aplicação do software, e léxicos para expressar ou descrever a denotação e a conotação dos conceitos da aplicação. Além disso, dentro da programação da disciplina, Renato conheceu o jogo SimulES e ficou muito empolgado com o jogo de tabuleiro. E assim que Renato ficou sabendo que o SimulES estava sendo desenvolvido como uma aplicação Web, de modo que todos os estudantes poderiam usá-lo através do acesso *on-line*, quis conhecê-lo. Porém, como Renato deseja aplicar em seus futuros desenvolvimentos a técnica de léxicos e cenários e como gostou muito de jogar SimulES na versão de tabuleiro, ele quer descobrir na versão Web os cenários e os episódios que ele identificou na documentação e na versão tabuleiro. Com esse conhecimento prévio, Renato espera não ficar muito confuso na navegação do jogo e por isso ele deu uma revisada novamente na documentação e imprimiu uma versão dos cenários para

poder se guiar na versão Web". *Como você faria para realizar a tarefa de Renato no ambiente do SimulES-W?*

Além disso, no MIS é definido a meta mensagem de comunicação inicial do ponto de vista do engenheiro para seu usuário final:

[*Aqui está o meu entendimento sobre quem é você*]: Como estudante de ciência da computação você deve estar habituado a padronização dos ícones e da navegação nas aplicações. Além disso, você está familiarizado com o jogo SimulES de tabuleiro, conhece seus cenários e os episódios que acontecem em cada um deles. Este conhecimento adquirido será vital para uso adequado do SimulES-W. Você deve identificar e executar as ações do jogo no ambiente digital, aprendendo também as novas ações incorporadas para adequar o jogo a esse ambiente.

4.3 Passo 2: Analisar os signos metalingüísticos

Nesta etapa é realizada uma inspeção tanto *on-line* quanto *offline* de documentação disponível sobre o SimulES-W, buscando a meta-mensagem do engenheiro para o usuário. Os usuários podem usar os cenários definidos para a aplicação tanto impressos quanto *on-line* [13] e a opção de ajuda que o sistema fornece. A Figura 2 é a representação de uns dos signos metaliguisticos do SimulES-W que foi utilizados para a análise.

[O que eu aprendi que você quer ou precisa fazer, de que forma e porque] Você precisa de material de ajuda do jogo que esteja disponível tanto dentro da aplicação quanto fora dela para o entendimento do jogo e com informações sobre como utilizar e para que serve cada componente da interface.

[Este é o sistema que eu projetei para você e essa é a maneira pela qual você pode ou deve utilizá-lo de forma que preencha uma variedade de propósitos que coadunam com esta visão] As ajudas aparecem após você clicar no ícone de ajudas na barra de ferramentas como é apresentado na Figura 2, ou realizando alguma ação da mesma forma como apresentado na Figura 2. Se você precisa de informação mais detalhada, os cenários vividos na prática de um ambiente de desenvolvimento de software (exemplo na Figura 3) possuem a informação relacionada com o jogo e as diferentes rodadas, além de ter os requisitos para se fazer as jogadas e se ganhar o jogo.

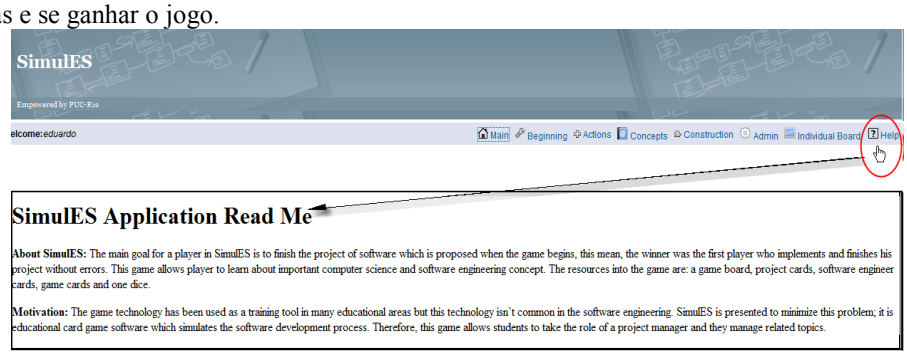

Fig. 2. Ajuda SimulES-W.

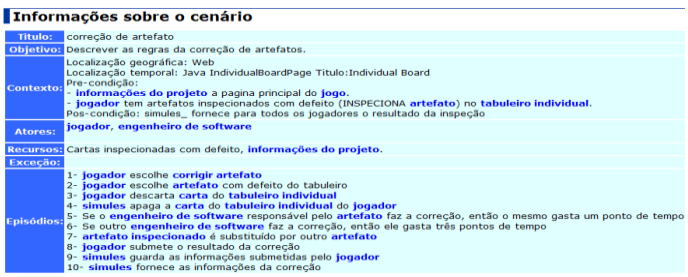

Fig. 3. Cenário de SimulES-W.

4.4 Passo 3: Analisar os signos estáticos

A análise das interfaces da aplicação, com seus botões, menus, ícones e *layout* e imagens estáticas. *[O que eu aprendi que você quer ou precisa fazer, de que forma e porque]* Você precisa ter o mesmo recurso oferecido no jogo do tabuleiro, ou seja, tabuleiro individual do jogo e seus recursos, além disso, você deve poder realizar as mesmas jogadas.

[Este é o sistema que eu projetei pra você e essa é a maneira pela qual você pode ou deve utilizá-lo de forma que preencha uma variedade de propósitos que coadunam com esta visão] No tabuleiro individual (Figura 4) os artefatos brancos e cinzas, que devem ser distribuídos entre os engenheiros de software contratados para construção do produto, foram projetados no desenho da tela, de modo a assemelhar-se ao máximo com o tabuleiro físico do jogo. Dessa forma, todos os componentes da tela tentam levar você a executar adequadamente as ações desta etapa baseadas nas experiências adquiridas com o jogo de tabuleiro. Os elementos dispostos em este tabuleiro devem ser semelhantes aos elementos que já você conhece do jogo do tabuleiro, você deve identificar pelos mesmos nomes e figuras e assim você se poderá guiar na ação que deve executar. Foram identificadas como símbolos estáticos nesta tela artefatos inspecionados e sem inspecionar brancos e cinzas e as etiquetas alusivas a cada tipo de artefato. Os demais *labels* complementam a construção do tabuleiro individual, lembrando que sua condição de signos estáticos é dada porque seu significado é independente das relações causais e temporais. Ou seja, o contexto da interpretação é fornecido em um momento especifico da interface.

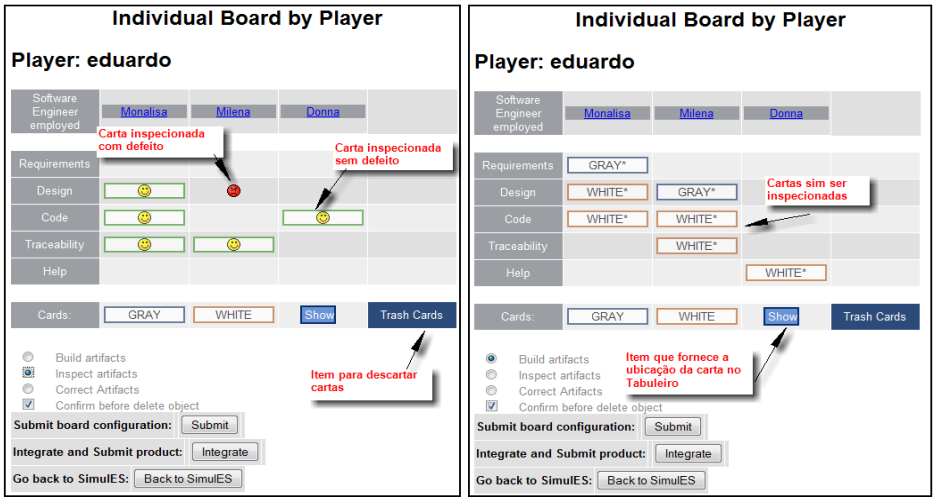

Fig. 4. Signos Estáticos do Tabuleiro Individual.

4.5 Passo 4: Analisar os signos dinâmicos

[O que eu aprendi que você quer ou precisa fazer, de que forma e porque] Você precisa ter os mesmos episódios que conhece do jogo do tabuleiro para que possa começar sua partida, além disso, como você precisa se registrar para que possa participar do jogo e as imagens em forma de botão da Figura 5 devem-lhe indicar isso.

[Este é o sistema que eu projetei pra você e essa é a maneira pela qual você pode ou deve utilizá-lo de forma que preencha uma variedade de propósitos que coadunam com esta visão] Na tela da Figura 5, você encontra disponíveis as imagens para efetuar suas ações que são: registrar-se no jogo, escolher o projeto e escolher seu primeiro engenheiro de software. Você deve clicar sobre a animação do dado para que o resultado do dado apareça, além disso, a execução das demais imagens dispostas na tela oferecem a você os demais recursos necessários para iniciar o jogo, tais como: escolher o projeto e seu primeiro engenheiro. Estes signos são considerados dinâmicos já que oferecem a possibilidade de transição de estado na interface como conseqüência das ações realizadas sobre eles.

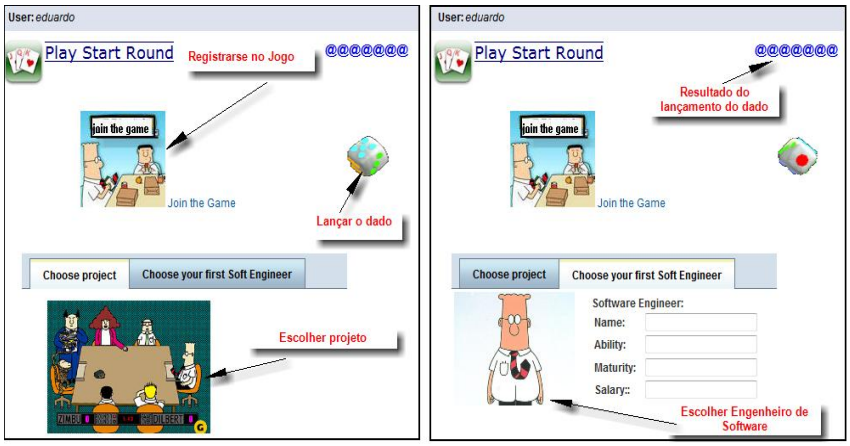

Fig. 5. Signos Dinâmicos Tela de Inicio

4.6 Passo 5: Comparar

Na tela rodada de início identificamos alguns signos que estão associados a um label deixando bem claro seus significados, como é o caso de "*Join the Game*" (Registrar-se no jogo), "*Choose project*" (Escolher projeto) e "*Choose your first soft engineer*" (Escolher o seu primeiro engenheiro de software). O principal problema nesta tela está relacionado ao seguinte signo @@@@@@@, que não deixa bem claro a sua intenção, que na visão do engenheiro seria a espera do resultado do lançamento do dado, este signo aparece também em outras telas do jogo. Um signo dinâmico bem claro é dado, na verdade é uma animação que fica girando até que o usuário interage. Além do mais, são mostradas algumas imagens tratadas pelo sistema também como botões. O comunicado pelo botão para escolher o engenheiro de software parece como um signo estático, com tudo, o engenheiro acabou omitindo a possibilidade de interação através dele. Na tela de início os "*tooltips*" estão em conformidade com os signos dinâmicos além das mensagens de ajuda.

No tabuleiro individual os artefatos (brancos e cinzas), os artefatos inspecionados e artefatos não inspecionados comunicam de modo bem claro o seu significado, principalmente por estar bem próximo dos símbolos usados no jogo de tabuleiro. Um problema a ressaltar: os signos utilizados somente pelo desenvolvedor que foram deixados na interface. O botão *Show*, observado na Figura 6, não possui uma relação direta com o jogo e ainda este é bastante convidativo ao usuário, dada a semântica associada a ele. Os "*optionButtons"* que permitem escolher a ação sobre o tabuleiro tem concordância com os cenários definidos no jogo de tabuleiro, os botões para submeter a jogada ou submeter o produto também são claros. As mensagens fornecidas dentro do tabuleiro individual precisam ser reestruturadas, pois seu conteúdo é bem técnico e comunica pouco ao usuário, contudo, algumas mensagens estão relacionadas a operações que não são importantes dentro do jogo, como é o caso de localização dos artefatos dentro do tabuleiro (Figura 6).

O sistema de ajuda do jogo é apresentado na barra de ferramentas e encontra-se localizado no extremo direito da tela. O conteúdo do sistema de ajudas descreve sobre a filosofia do jogo, mas não aborda como o SimulES-W tem que ser jogado. Logo, se precisa complementar estas informações. Finalmente, alguns dos *tooltips* fornecidos pelo SimulES-W devem ser revistos sendo que alguns tem problemas de conteúdo e cor, fugido dos padrões estabelecidos para aplicações Web.

4.7 Passo 6: Avaliar

Como limitações deste trabalho pode-se dizer que a avaliação qualitativa da aplicação do MIS não foi o suficientemente satisfeita. Conforme descrito em Souza et al [3] neste passo é

recomendável fazer uma triangulação de fontes endógenas e/ou exógenas. Fontes endógenas referem-se ao mesmo modelo de artefato ou artefatos que compartilham o mesmo domínio do modelo, ou seja, é possível triangular a interpretação com outras realizadas por outros inspetores e intérpretes no mesmo contexto, ou o que se pode dizer sobre o mesmo artefato que estamos inspecionando, ou acerca de artefatos do mesmo tipo. Existem outros sistemas de software para ensino na engenharia de software [14], [15] e [16] que compartilham o mesmo domínio, mas a forma de implementação é bem diferente, impossibilitando a realização de uma comparação. Como foi dado a conhecer no inicio deste trabalho, jogos para ensino da engenharia de software é uma área pouco explorada.

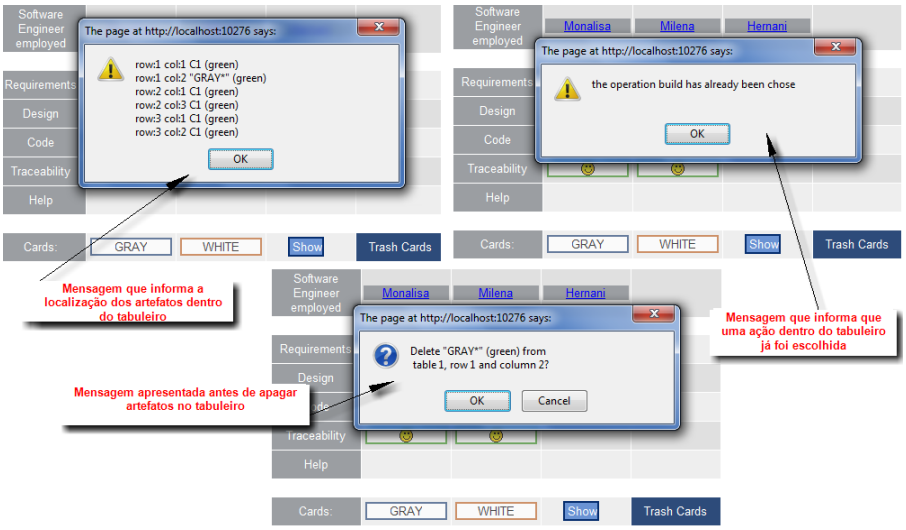

Fig. 6. Mensagens informativas na tela tabuleiro individual

Por outro lado, o MIS propõe também triangular com fontes exógenas as quais referem-se a artefatos que não compartilham o mesmo modelo de domínio, no entanto, compartilham certas características relevantes do projeto. Ou seja, os artefatos cujas interpretações estão sendo triangulados podem fazer coisas completamente diferentes, mas eles devem ter algo em comum que está diretamente relacionada com a interpretação que se pretende validar. Cabe anotar que o SimulES-W foi desenvolvido usando *JavaServer Faces* (JSF) e *Visual Web JavaServer Faces* e que ambos são *frameworks* para criação de elementos de interface de usuário baseados em Java. Porém muito dos ícones, barra de ferramentas e estilos de navegação que foram obtidos destes *frameworks* são usados para criação de aplicações Web para Java, consequentemente, podem ser familiares para a maioria dos usuários Web sendo considerados padrões. Além disso, o software foi projetado incorporando atributos da qualidade vindos da transparência de software [17] que por sua vez têm embutidos atributos de qualidade de aplicações Web.

5 Evolução do SimulES-W a partir da Análise do MIS

O contexto do presente trabalho foi a procura da inter-relação dos atributos da usabilidade que podem ser avaliados com o método MIS e os critérios de usabilidade definidos para criar um software mais transparente definidos em [27] com a finalidade de encontrar mecanismos que ajudem na operacionalização dos mesmos. Cada um dos atributos de usabilidade definidos na Figura 1 será descrito do mesmo modo que o MIS propõe, de uma forma qualitativa e interpretativa. Em outras palavras, foram ponderados de uma forma exploratória tanto os elementos chamados de signos no MIS quanto os critérios de usabilidade, assim, a analise desta sessão relata que critérios foram ou não atendidos pelo Método e de que forma.

Como foi apresentado, a utilização do método MIS permitiu não só a classificação e decomposição dos signos, mas também a identificação das condições relacionadas com usabilidade. A análise geral da avaliação do SimulES-W utilizando o MIS focam várias quebras na comunicabilidade que contribuem negativamente para a usabilidade, assim como também elementos que atingem a usabilidade dentro do SimulES-W.

Se mantivermos as definições a redundância entre símbolos metaliguísticos/estático e metalingüísticos/dinâmico contribui para que tenhamos uma maior **intuitividade** e **comunicabilidade** melhorando a usabilidade. Portanto, as telas do SimulES-W foram revisadas e reestruturadas incorporando mensagens mais significativas para o usuário. Além disso, foi identificado que as imagens não estavam comunicando ao usuário a ação que tinham que executar e por isso elas foram redesenhadas e acompanhadas pelo signo metalinguístico (*tooltips*) como também de seu respectivo *label* (signo estático). Como as imagens não são interativas elas não estavam passando a idéia da ação ao usuário, o que levou a dar a solução de aumentar a quantidade de signos metalinguísticos para suprir essa necessidade.

Através da Inspeção Semiótica sobre a barra de ferramentas identificamos que alguns ícones não refletiam exatamente o conteúdo associado a esse "*menu*", além de que a disposição desses "*menus*" não proporciona uma navegação natural para todas as ações especificadas no jogo. Também foi considerada a mudança neste item. Esses aspectos estão relacionados com a **simplicidade, operabilidade e intuitividade**.

De modo geral, a tela de rodada de ações apresentou basicamente as mesmas dificuldades da tela inspecionada anteriormente, apresentando um detalhe adicional com relação a algumas mensagens de erro cujo conteúdo não era condizente com a cor escolhida para destaque das mensagens. Na tela de rodada de conceitos, ocorreram os mesmos problemas de expressividade dos signos estáticos, o que sugeriu adotar a solução antes mencionada. Além disso, signos sem funcionalidades foram omitidos, para evitar que o usuário fique confuso na hora de tomar ações. Isso significaria um desempenho ruim e maior complexidade (em contrapartida da simplicidade).

Evidencia-se que as ajudas fornecidas pelo software não auxiliam ao jogador em como deve utilizar o jogo, dando somente uma explicação sobre as regras do jogo, mas o que acontece em cada tela e a forma como ele tem que executar as ações não está explicitado nesta ajuda. Dificulta a **operabilidade e fere a intuividade**. Isso porque, conforme [27], uns dos requisitos para um sistema ser operacional é que este seja auto-descritivo e que ele execute as tarefas previstas nos cenários os quais deveriam ser fornecidos aos usuários em forma de ajuda. Outros elementos próprios da operabilidade descritos em [27] como são: *hábil para funcionar, permitir diversos níveis de acesso ou ser reconhecido como um componente único,* que não são possíveis de serem avaliados com este método que está focado na parte semiótica.

Finalmente, nota-se que um jogador experiente (Jogador que conheça bem o jogo do tabuleiro) realmente conseguiria direcionar-se e executar a maioria das ações do SimulES-W, o que não aconteceria com usuários pouco experientes. Isso deve levar a pensar na melhoria das e acompanhamento *on-line* das jogadas de cada jogador, oferecendo em todo momento mensagens que direcionem o usuário na navegação e na execução de suas tarefas. É importante ressaltar que SimulES-W é um jogo para ser acompanhado com um instrutor que forneça e valide os conteúdos relacionados com a Engenharia de Software, entretanto no desenho do SimulES-W deve ser relevante procurar mecanismos que facilitem as tarefas durante o jogo dado que os jogadores não estão no mesmo espaço físico. Essa poderia ser uma melhoria para aumentar **amigabilidade**.

O método por ser uma verificação baseada na interface e na comunicabilidade não avalia o critério de **Desempenho** por estar relacionado com um desempenho adequado. Em relação à Adaptabilidade esta pode ser avaliada somente no foco da transição de estado adequado dos signos dinâmicos. Para estes critérios outro tipo de avaliação deve ser aplicado. A Tabela 1 apresenta um pequeno resumo que permite focar como cada elemento da usabilidade é avaliado através do método MIS, o que nos permite chegar a operacionalizações destes atributos.

Tabela 1. Resumo Atributos da Usabilidade e Signos Avaliados mediante o Método MIS

Atributo da Qualidade	Signo Avaliado com MIS	De que modo pode atingir a transparência
Uniformidade	Signos estáticos, dinâmicos e metalingüísticos.	Redundância e correspondência quando signos estão relacionados.
Simplicidade	Signos estáticos, dinâmicos e metalingüísticos.	A mensagem da metacomunicação não deve ser prolixa.
Operabilidade	Signos estáticos, dinâmicos.	O método avalia que os símbolos sejam auto-descritivos.
Intuitividade	Signos estáticos e signos dinâmicos.	Metáforas bem definidas que comuniquem a funcionalidade correta ao usuário.
Desempenho	-	O método não avalia este atributo.
Adaptabilidade	signos dinâmicos.	Transição de estado através dos signos dinâmicos deve ser coerente.
Amigabilidade	Signos estáticos , signos dinâmicos e metalingüísticos.	O método MIS é forte na avaliação deste atributo.

6 Conclusão

Ao longo deste artigo foi apresentada uma abordagem que analisa a comunicabilidade de um artefato de software como uma conversa entre o engenheiro e o usuário. Contudo, o engenheiro constrói o artefato de forma que na interação o usuário consiga interpretar a informação que ele quer transmitir. Esta comunicação é mediada pelos signos e um contexto em comum que compartilham engenheiro e usuário. Esta abordagem permite entender que requisitos estão presentes não somente no processo de construção do software e que devem ser validados em cada uma das etapas para confirmar que os requisitos elicitados estejam conforme com as necessidades do usuário.

Sendo o engenheiro um ator ativo na comunicação é importante tornar evidente suas escolhas de desenho. O MIS permitiu, no caso de SimulES-W, analisar como essas escolhas e os entendimentos do engenheiro vão influenciar aos usuários nos processos de significação, ou seja, observamos e analisamos os signos e como eles podem interferir no comportamento dos usuários frente ao software. Ficou evidente como em Peixoto, Prates e Resende [21] que o uso do método MIS pode ser uma ferramenta útil para a avaliação do RFN de usabilidade, sendo que ela ajuda ao analisador a fazer um juízo sobre que e como elementos da interface estão comunicando ao usuário sua finalidade. Identificamos, também, que este método foca sua análise em RNFs como uniformidade, simplicidade, intuitividade, amigabilidade, informativo entre outros, presentes na Transparência de Software [19].

Nota-se que o resultado desta análise permitiu antecipar e relatar conseqüências que poderiam ocorrer durante a interação do usuário com SimulES-W. Desdobramentos destas evidências permitiram propor adaptações refinar os requistos já elicitados, analisados, modelados e implementados. Além disso, a aplicação do MIS permitiu identificar potencialidades e limites impostos pela interface durante a comunicação. Todo esse processo de inspeção deve refletir no sucesso da utilização do SimulES-W pelo usuário final bem como torná-lo mais transparente.

Além da análise da comunicabilidade seria própria a aplicação de uma análise desde o foco do usuário, a Engenharia Semiótica também propõe outro método denominado MAC [18] que avalia a comunicabilidade centrada na utilização do artefato do software pelo usuário, ou seja, o MAC envolve testes de usuário que consideram aspectos da interface.

Visando a qualidade da interação e da experiência do usuário, outras operacionalizações com o usuário devem ser realizadas, como por exemplo: testes de usabilidade que permitem identificar os impactos de determinadas escolhas de desenho por meio de experiências reais.

Agradecimentos

Julio Cesar Sampaio do Prado Leite agradece o financiamento de CNPq e FAPERJ.
Elizabeth Suescún Monsalve reconhece o financiamento da CAPES pela bolsa recebida que viabilizou esta pesquisa.

Referências

1. Norman D., Draper S.: User-centered System Design. Direct Manipulation Interfaces. University of California, San Diego. Lawrence Erlbaum Associates, Publishers. Hillsdale, New Jersey London (1986)
2. de Souza, C. S. The Semiotic Engineering of Human--Computer Interaction. MIT Press. ISBN 0-262-04220-7 (2005)
3. de Souza, C.S., Leitão, C.F., Prates, R.O., Bim, S.A., da Silva, E.J.: Can inspection methods generate valid new knowledge in HCI? The case of semiotic inspection. International Journal of Human--Computer Studies, Volume 68, Pages 22--40. Available online in December 2009. doi:10.1016/j.ijhcs.2009.08.006. Issues 1-2, January- February (2010)
4. Introdução à usabilidade de Nielsen http://www.useit.com/alertbox/20030825.html, ultimo acesso outubro de 2010
5. Problems and Programmers http://www.problemsandprogrammers.com/, ultimo acesso outubro de 2010
6. Figueiredo, E., Lobato, C., Dias, K., Leite, J.C.S.P. , Lucena, C.: Um Jogo para o Ensino de Engenharia de Software Centrado na Perspectiva de Evolução, XV Workshop sobre Educação em Computação (WEI), Rio de janeiro. co-alocado ao XXVII Congresso da SBC,. pp. 37--46. (2007)
7. SimulES v 1.0 http://www.teccomm.les.inf.puc-rio.br/emagno/simules/, ultimo acesso outubro de 2010
8. Serrano, M., Serrano, M., Napolitano, F., Soares, B.: Evolução do SimulES Versão 2.0, Monografia Final, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil (2007)
9. Monsalve, E. S.: Construindo um Jogo Educacional com Modelagem Intencional Apoiado em Princípios de Transparência, Dissertação de Mestrado, PUC–Rio, (2010)
10. Monsalve, E., Werneck, V., Leite, J.C.S.P.: Evolución de un Juego Educacional de Ingeniería de Software a través de Técnicas de Elicitación de Requisitos, Proceedings of XIII Workshop on Requirements Engineering (WER'2010), Cuenca, Ecuador, Abril 2010, pp. 12--23,(2010)
11. Napolitano, F.: Uma Estratégia Baseada em Simulação para Validação de Modelos em i*. Dissertação de Mestrado, PUC–Rio. (2009)
12. Blog SimulES por Serrano M http://mileneserrano.wordpress.com/, ultimo acesso outubro de 2010
13. Blog Disciplina em Princípios em Engenharia de Software http://pes.inf.puc-rio.br/cel/, ultimo acesso outubro de 2010
14. Jogo SESAM http://www.iste.uni-stuttgart.de/se/research/sesam/overview/index_e.html, ultimo acesso outubro de 2010
15. Boehm, Jain. B.: SimVBSE: Developing a Game for Value-Based Software Engineering, Proceedings 19th Conference on Software Engineering Education and Training, pp. 103 --114, (2006)
16. Birkhoelzer, T., Navarro, E., van der Hoek, A.: Teaching by Modeling instead of by Models, Proceedings of the 6th International Workshop on Software Process Simulation and Modeling, St. Louis, MO, (2005)
17. Leite, J.C.S.P.: Sistemas de software Transparentes. Palestra convidada en el 20 Simpósio Brasileiro de Engenharia de Software (http://www-di.inf.puc-rio.br/~julio/Slct-pub/transp-sbes.pdf) (2006)
18. Prates, R.O., de Souza, C.S., Barbosa, S.D.J.: Communicability Evaluation Method for User Interfaces. Interactions. New York:, v.7,n.1,p.33-38, http://doi.acm.org/10.1145/328595.328608. (2000)
19. Leite, J.C.S.P., Cappelli, C. Software Transparency. Business & Information Systems Engineering 2(3): 127-139 (2010)
20. Hainey, T. , Connolly, T. M., Stansfield, M. and Boyle, E. A., Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level, In; Computers and Education, 56, 2011, 21-35
21. Peixoto, D. C. C., Prates, R. O., and Resende, R. F., Semiotic Inspection Method in the Context of Educational Simulation Games, SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing, doi>10.1145/1774088.1774342, (2010)