O FUNDAMENTAL NA AVALIAÇÃO DA QUALIDADE DO SOFTWARE EDUCACIONAL

Edla Ramos

Laboratório de Software Educacional - EDUGRAF

Departamento de Informática e Estatística

UFSC

1. Introdução

Muito se tem falado a respeito da uso dos recursos da informática no processo educacional, e quando se tenta vislumbrar qual a forma adequada deste uso, torna-se importante abordar a questão da avaliação do software a ser utilizado. Esta preocupação, deriva da necessidade de optimizar os esforços e recursos despendidos na área, tanto a nível da pesquisa acadêmica, e dos investimentos educacionais públicos ou privados, quanto a nível do trabalho dos professores que atuam diretamente no sistema educacional.

Falar em avaliação de software educacional exige que de imediato se defina um padrão de qualidade para o mesmo, pois avaliar é uma atividade na qual comparamos a "realidade" com um modelo "ideal", designado pelo padrão. Esta questão passa necessariamente pela definição do paradigma educacional subjacente à prática pedagógica levada a efeito. Logo leva ao questionamento, fundamental e imprescindível, da própria escola que temos hoje e de quais as propostas pedagógicas nela em voga. Especificamente ao tentar esclarecer o que é um bom software educativo, deve-se fazer também uma profunda reflexão, tentando vislumbrar qual o papel da tecnologia da informática na educação.

Segundo Carraher[Car90], são dois basicamente os contextos nos quais podem ser inseridos os aspectos observáveis na qualificação de um software educacional: o educacional ou pedagógico e o técnico computacional. A avaliação técnica e computacional é importante, mas deve estar subordinada à educacional e pedagógica. Algumas versões do LOGO seriam com certeza reprovadas por muitas metodologias de avaliação de software educativo, dada a importância que tais metodologias delegam a questões técnicas irrelevantes no contexto. Quer-se aqui salientar que é preciso estar alerta contra os atraentes circos tecnológicos, propagandeados pela indústria da área, pois os mesmos frequentemente retratam visões pedagógicas retrógradas.

Este artigo não se propõe a apresentar uma metodologia acabada para a avaliação de software educacional. O mesmo se propõe a fornecer uma relação de aspectos a serem observados na avaliação do software educacional, e espera-se que a contribuição principal esteja na explicitação da ênfase real a ser atribuída a cada aspecto. Propõe-se uma abordagem, nova nesta área, que é a de realizar a avaliação considerando a grande diversidade de tipos de uso da tecnologia da informática possíveis no processo educacional. Existem características que são peculiares a cada tipo, não fazendo sentido avaliar de forma idêntica um software do tipo instrução programada e uma simulação, como o proposto nas metodologias conhecidas.

2. Aspectos Relativos ao Eixo Pedagógico da Avaliação

Um vasto trabalho já foi realizado no sentido de construir classificações para os softwares educacionais. A bibliografia apresenta várias destas taxionomias, algumas delas serão a seguir sumarizadas.

Considerando as funções que podem ter os materiais educativos computadorizados Galvis[Gal88] identifica as seguintes categorias:

- tutoriais: como o nome indica este tipo de software pretende assumir as funções do bom tutor guiando o aprendiz através das distintas fases da aprendizagem, estabelecendo uma relação coloquial com o mesmo. Tipicamente um tutorial segue as quatro grandes fases descritas por Gagné para o processo de aprendizagem: motivação, retenção, aplicação e retro-alimentação.

- exercitação e prática: estes materiais preocupam-se basicamente com as duas últimas fases da taxionomia de Gagné, a aplicação e a retro-alimentação. São portanto bem menos ambiciosos que os tutoriais.

- simuladores e jogos educativos: este tipo de software tenta apoiar a aprendizagem criando situações que se assemelhem com a realidade. No caso dos jogos introduz-se ainda uma componente lúdica e de entretenimento.

- Linguagens sintonizadas: uma forma particular de interação com ambientes computacionais ocorre com a utilização de linguagens próprias destes ambientes. Uma linguagem sintonizada é aquela que não precisa ser aprendida por alguém que esteja em sintonia com suas instruções usando-a naturalmente para interagir com algum micro mundo no qual seus comandos sejam aplicáveises.

- sistemas especialistas: são sistemas capazes de representar e de arrazoar sobre algum domínio do conhecimento.

Stahl[Sta90] e Galvis[Gal88] citam a classificação proposta por Thomas Dwyer, que considerando a atividade do aprendiz, propõe uma grande divisão em dois grupos, software com enfoque do tipo algorítmico e software com enfoque do tipo heurístico:

- No enfoque do tipo algorítmico é predominante a ênfase na transmissão de conhecimento do sujeito que sabe para o sujeito que deseja aprender, sendo neste caso função

do criador do software, projetar uma seqüência bem planejada de atividades que conduzam o aprendiz ao objetivo esperado. Este enfoque utiliza, basicamente, tutoriais e exercício e prática.

- No enfoque do tipo heurístico, o aspecto predominante é a aprendizagem experimental ou por descobrimento, devendo o software criar um ambiente rico em situações que o aluno deve explorar conjecturalmente. Compartilham deste enfoque as simulações, os jogos, as linguagens sintônicas e os sistemas especialistas.

Segundo Reggini[Reg90] existem duas modalidades para o uso de computadores na educação: a modalidade dura e a modalidade branda. Na primeira o trabalho no computador segue planos traçados anteriormente. A atividade dos alunos resume-se a responder perguntas que aparecem na tela. Não existe nenhuma alegria na interação com os programas. Erros e acertos são registrados e contabilizados. Na segunda, a modalidade branda, a atividade não parece ter um objetivo definido. O aprendiz está no comando do computador, fazendo uma série de coisas interessantes com eles. Erros são fontes para novas reflexões e projetos.

É preciso acrescentar às taxionomias aqui reproduzidas um outro tipo de uso educacional, possível para os recursos computacionais. Moreira[Mor87] questiona: "Um software é educacional em si mesmo ou depende do uso que se faça dele?". Ora, resumidamente poderíamos dizer que os computadores são parte de uma tecnologia que tem como fim o processamento, o gerenciamento e a disseminação da informação/conhecimento. Este é também o objetivo do processo educacional. Na verdade há software construído já visando o uso no processo educativo e que reflete portanto uma concepção educacional e existe software construído para outros fins, tais como: pacotes gráficos, estatísticos e de análise numérica, planilhas eletrônicas, gerenciadores de base de dados, processadores de textos, etc., que podem vir a ser utilizados com sucesso no processo educacional.

Outras taxionomias bastante completas e podem ser encontradas em Campos[Cam89], Coburn[Cob88] e Stahl[Sta90], dentre outros autores.

No contexto da avaliação do software educacional, torna-se importante registrar uma convergência percebida entre estas várias taxionomias. Toda prática pedagógica está profundamente impregnada de uma determinada visão da natureza humana. Percebe-se uma linha divisória clara entre os software educacionais, esta linha é definida por concepções educativas bastante distintas. De um lado da está o paradigma comportamentalista (modalidade dura e enfoque algorítmico) e do outro lado está o paradigma do construtivismo (modalidade branda e enfoque heurístico).

Logo o ato de classificar um software quanto ao tipo de uso educacional a que se destina, é uma das etapas da avaliação do software. Na verdade, a primeira e a principal etapa, pois o tipo de uso a que se destina, reflete a concepção pedagógica do software. Ora a explicitação da concepção pedagógica é fundamental para a definição do padrão de qualidade a ser adotado no processo de avaliação. Conforme Moreira[Mor86], são substancialmente diferentes os conceitos de qualidade decorrentes das visões empirista e racionalista (paradigma comportamentalista), e da visão interacionista (paradigma construtivista).

Sob este ponto de vista fica claro que as metodologias de avaliação de software devem avançar na direção de levar em consideração o paradigma educacional subjacente ao software a ser avaliado, sob pena de barrar a produção e utilização de muitos aplicativos que não atendam os requisitos de uma particular visão pedagógica, imbricada na metodologia adotada. Por exemplo, perguntar numa ficha de avaliação de software se as questões estão bem formuladas e qual o nível de tratamento dado as respostas erradas dos alunos, traria inconsistência para processo adotado, se não houvesse nenhuma questão sendo formulada no software em análise.

Antes de detalhar a relação dos aspectos concernentes a avaliação de cada tipo de software educacional é preciso ressaltar a existência de uma questão geral, que é fundamental neste processo. Concorda-se com Galvis[Gal88] quando afirma:

> "...o computador deve ser usado no processo ensino-aprendizagem, antes de qualquer outra coisa, como um meio para implementar o que com outros meios não seria possível ou seria difícil obter. Diferentemente do que alguns educadores temem, não se trata de implementar com o computador a ação de outros meios educativos cuja qualidade está bem demonstrada. Este raciocínio não é estranho, se considera que o computador é um bem escasso e também custoso, cujo uso deve oferecer o máximo de benefícios, neste caso educativo...".

Também neste sentido Moreira[Mor87] declara "... não é desejável que o software educacional venha substituir situações já resolvidas de modo mais simples...". aposta-se então que o bom software educacional ou generalizando, o bom uso da tecnologia da informática na educação propiciará experiências educacionais novas e ricas, ou pelo menos tornará muito mais eficiente o ensino efetivado nos moldes tradicionais.

Considerando os aspectos abordados acima apresenta-se a seguir uma relação das questões a serem observadas na avaliação de alguns tipos de software educacional.

Sob o **paradigma comportamentalista** serão enfocadas as categorias: Tutorial e Exercício e Prática:

Tutoriais - Sob quais aspectos justifica-se o uso dos recursos computacionais na proposta?

- Com relação a motivação para o aprendizado e a apresentação dos conteúdos: A estratégia motivadora utilizada é eficiente e adequada? O ritmo de trabalho é controlável? O conteúdo está desenvolvido corretamente? O diálogo é rico e bem formulado?

- Com respeito a aplicação dos conhecimentos e retro-alimentação: A seqüência de problemas propostos é adequada? Permite tratamento personalizado do erro com estratégias corretas de reforço? O resultado final alcançado pelos alunos é satisfatório?

- Outras questões: O software é de fácil operação? As instruções fornecidas são claras? Permite o registro eficiente da evolução de cada aluno?

Exercício e Prática - Excetuando-se a etapa da apresentação do conteúdo esta modalidade reproduz a anterior, devendo ser observados portanto os mesmos aspectos.

Outros tipos de usos - em geral sob o paradigma comportamentalista a qualidade da estratégia educacional deve ser medida a nível da eficácia em se provocar determinados comportamentos de maneira a não causar esforços e angústias desnecessárias.

Sob o **paradigma construtivista** percebe-se uma maior dificuldade quando se tenta delinear o processo de avaliação da utilização dos recursos computacionais na educação. Estas dificuldades eram de se esperar, uma vez que sob este paradigma os objetivos a serem alcançados no processo educacional, não se expressam através da obtenção de comportamentos que devam ser obrigatoriamente mensuráveis. Sob este paradigma serão enfocadas as seguintes categorias:

Simulações - A vivência concreta da experiência é inviável por questões financeiras, temporais , geográficas ou de periculosidade?

O software permite o enriquecimento cognitivo da experiência ampliando o leque das informações assimiláveis? - Este enriquecimento pode se dar via a introdução de interfaces que permitem a captação e o tratamento simultâneo de uma grande massa de dados. Pode dar-se também via a facilidade na repetição do experimento um grande número de vezes o que permitiria a criação de uma sensibilidade mais aguçada a respeito do relacionamento das variáveis envolvidas na experiência. Mas principalmente, por permitir maior flexibilidade em alguns casos e noutros por permitir o controle a nível ideal, das variáveis de entrada do modelo, o computador pode possibilitar a realização da experiência sob condições dificilmente obtidas na realidade.

Jogos Educativos - Os jogos educativos pretendem introduzir às simulações uma componente lúdica e prazerosa. Logo além dos aspectos relacionados com os simuladores cabe observar ainda:

A estratégia motivadora do prazer não apresenta nenhum tipo de efeito colateral? (Competição exacerbada, indução a preconceitos, angústia, etc.).

Outros tipos de uso - Além dos simuladores e jogos educativos foram mencionados muitos outros tipos de uso para os computadores. Sob o paradigma construtivista a avaliação deve se encaminhar no sentido de definir: o potencial cognitivo da proposta, o nível de satisfação e de interesse demonstrado pelos alunos, o nível de sociabilização fomentado entre os alunos, o nível de interação permitido entre o ambiente e o aprendiz.

3. Recomendações Ergonômicas para os Parâmetros de Interface

Para se a avaliar uma aplicação computacional interativa cabe observar se ela obedece ou não um conjunto de critérios ergonômicos desejáveis.

As recomendações sintetizadas a seguir foram propostas na sua maioria agrupadas e apresentadas por Barthet.

3.1 Sucessão de operações

O problema focado aqui é o de adequação entre a ordem das operações fixadas pela máquina e aquela necessária ao operador para realizar sua tarefa não importa o ambiente. O que é importante neste caso é que esta sucessão se dê de acordo com a lógica do usuário ou com uma lógica de utilização.

Trata-se neste caso:

- do tipo de encadeamento entre as operações;
- a possibilidade do encadeamento variar segundo o grau de experiência e do tipo do usuário;
- a possibilidade de acesso a todas as informações a todo momento;
- de poder sair, anular ou interromper (com retorno ao mesmo ponto) uma transação a todo momento;
- de fazer uma chamada a uma operação qualquer que seja a partir de uma outra e voltar a primeira;
- de transferir as informações de uma aplicação para outra.

A nível da interface tais recomendações concernem aos dispositivos de entrada e ordenação, ao vocabulário e sintaxe e às mensagens de erros e guias de utilização.

3.2 Linguagem de interação

A linguagem de interação é quem vai permitir que o usuário expresse, a partir de um vocabulário e de uma sintaxe, as operações que ele deseja que a máquina efetue; da mesma forma; que permite que o usuário interprete as respostas que lhe são fornecidas após a execução das operações solicitadas.

Quanto ao *vocabulário*, os ergônomos privilegiam de forma unânime o emprego, sempre que possível, da linguagem dos usuários. Esta recomendação não é facilmente seguida, principalmente há que se considerar que a diferenciação que a informatização introduz na realização da tarefa leva à criação de um novo vocabulário.

Nestes casos é necessário escolher novos vocábulos, e testar o seu uso com cuidado. Recomenda-se que esta escolha evite a utilização de códigos numéricos, pode-se utilizar códigos mnemônicos mas dá-se preferência à língua natural.

De modo geral os *pictogramas*, ou ícones, podem se revelar muito interessantes quando não apresentam ambiguidade; eles permitem exprimir um conceito a partir de um único item facilmente registrável na memória a curto tempo, a partir de analogias com um caso manual conhecido, levando a uma memorização de longo termo facilitada.

A *sintaxe* de uma linguagem de interação é formada pelo conjunto de regras que permitem expressar comandos mais complexos a partir de combinações entre os mesmos. As

recomendações básicas aqui referem-se a simplicidade, por razões de memorização e à homogeneidade para evitar riscos de erros e favorecer a formação de automatismos. A presença de exceções será fatalmente fonte de erros.

3.3 Os tempos de resposta

A questão que se põe aqui é a seguinte: qual é o tempo de resposta ideal? Os especialistas não entram em acordo totalmente sobre a resposta a esta questão. Alguns defendem uma resposta imediata, outros uma resposta compatível com a velocidade do processo cognitivo do ser humano.

Para tentar responder mais precisamente esta questão, devem ser relembrados os resultados da psicologia concernentes ao tempo de resposta que é da ordem de dois segundos numa conversação entre duas pessoas e à retenção dos dados na memória de curto termo (de dois a seis segundos no máximo).

Tais resultados levam a concluir:

- perto de dois segundos: tempo de resposta ideal;
- de 2 a 4 segundos: impressão de espera, pode ser gerado pela memória de curto tempo;
- mais de 4 segundos: muito longo se o dialogo necessitar uma memorização a curto tempo e se não existem mensagens fixadas no vídeo.

3.4 O tratamento de erros

Cabe neste caso distinguir dois tipos de erros: os *erros de execução*, facilmente detectáveis e retificáveis, provenientes do fato de se apertar inadvertidamente em outra tecla que não aquela desejada; os *erros de intenção* que correspondem a uma má interpretação do sentido dos comandos ou da significação dos procedimentos; estes erros podem não ser detectados e sua retificação pode exigir um esforço de aprendizagem de parte do usuário.

Para o assinalamento de erros, há unanimidade sobre os seguintes elementos sobre o fato de que os erros devem ser assinalados imediatamente ou o mais rapidamente possível por causa da volatilidade da memória de curto tempo;

Para a correção de erros, o usuário deve poder rever facilmente a operação ou a linha onde se situa o erro e deve poder anular eventualmente na totalidade ou em parte o trabalho que foi feito depois deste.

3.5 A ajuda na aprendizagem e na realização da tarefa

3.5.1 Ajuda ao Trabalho

- Transferência de Dados - é dada ao usuário a possibilidade de transferir um dado de uma aplicação a uma outra sem precisar reescrevê-lo. Esta função permite minimizar os embargos e evitar os erros de digitação.

- Interrupção - a interrupção permite ao usuário sair, não importa qual a operação esteja em curso de execução, para ir executar uma outra operação e retomar a primeira do ponto de interrupção. Pode-se prever vários níveis sucessivos de interrupções, apesar de que concretamente, os usuários não ultrapassem o terceiro ou quarto nível.

- Saída (*Quitter*) - o usuário pode precisar abandonar seu trabalho a qualquer momento ou ir para um outro item do menu. O software deve sinalizar as conseqüências desta atitude, conforme o lugar onde o usuário se encontre.

- Retardar - o usuário pode querer retardar não importa qual operação dentro do tempo. Retardar se distingue da saída porque dentro da ação de retardar existe a memorização da tarefa não terminada, e isto, não acontece na ação de saída.

- Anular - o usuário pode anular a última ação que foi realizada, esta possibilidade é particularmente útil para que se possa corrigir erros cometidos pelo usuário ou assinalados pelo software.

3.5.2 Ajuda na Aprendizagem

Aqui trata-se do auxílio à aprendizagem de manuseio do software interativo e não do trabalho em si. Barthet recomenda a existência de dois guias:

– Guia Funcional - um comando do tipo S.O.S. deve permitir ao usuário conhecer a todo o momento seja a lista de operações possíveis dentro do estágio atual do trabalho, seja uma explicação das funções e efeitos de um dado comando. Este tipo de possibilidade é clássica e possui grande popularidade.

– Guia de utilização - a todo momento, o usuário deve poder ter acesso a um modo guia, elaborado segundo a lógica de utilização, onde os encadeamentos de operações correspondentes às precedências permanentes ou indicativos, lhe serão propostos em função do objetivo que ele tenha fixado. Este tipo de guia pode permitir responder a questões do tipo "*Como fazer para*....?". A elaboração de tal guia é possível, graças a estruturação do sistema em objetivos, sub-objetivos, operações e precedências.

Sellen e Nicol (1990) abordam a questão dos guias de utilização de uma forma um pouco mais detalhada. Os autores afirmam que o tempo de busca nos *helps* poderia ser bastante reduzido se a forma do tópico do *help* correspondesse ao tipo de questão que o usuário faz. Acontece que o tipo de questão feita, freqüentemente determina o tipo de informação requerida. A tabela a seguir lista os tipos de questões citadas pelos autores:

Tipo da questão	Forma canônica da questão
Orientada a objetivos	Que tipo de coisas eu posso fazer com este Programa?
Descritiva	O que é isso? O que é que isso faz?
Procedural	Como é que eu faço isto?
Interpretativa	Como foi que isto aconteceu?
Relativa à navegação	Onde é que eu estou?

Tabela 1. Forma canônica para os tipos de questão mais comum feito pelos usuários.

Donde o tipo de informação enviada deveria ser diferente para cada tipo de questão formulada, ou a forma pela qual a informação é alcançada e apresentada deveria depender do tipo de questão feita. Definindo melhor:

O que eu posso fazer com este programa?

Deve introduzir informação facilmente acessível e visível para o usuário iniciante e deve prover informações sobre todo o espectro de capacidade do software, especialmente apontando usos e funções que o usuário não iria normalmente descobrir.

O que é isso? Para que é isso?

Elas ocorrem no modo operatório, quando o usuário aponta para um objeto na tela e pede uma descrição. O segundo ocorre no modo de execução de tarefa, e um termo específico precisa ser definido.

Como é que eu faço isto?

Este é o tipo mais freqüente de questão, pois é quando o usuário está diretamente envolvido em um tipo específico de tarefa. Ele já sabe o que quer, e as respostas a todas as outras questões, são na verdade fundamentais para que ele possa desenvolver a sua tarefa, ou seja esta questão é fundamental. Barthet já salientava este fato quando associava este tipo de questão à lógica da utilização do software.

Aqui o que se recomenda é um *help* procedural que seja expresso na "linguagem" e no vocabulário do usuário (linguagem orientada à tarefa) e que não desative o contexto de trabalho em que a questão foi feita.

Por que isto aconteceu?

Perguntas deste tipo ocorrem quando o sistema responde de forma inesperada, seja esta resposta uma mensagem de erro ou não. Nesse caso, para poder prover o usuário das ferramentas necessárias, o sistema precisaria poder descobrir qual era a intenção do usuário que foi frustrada. Ora, esta não é uma tarefa fácil, ela é dependente da capacidade de "inteligência" e de rastreamento embutidas no computador, pois:

mesmo que o computador seja capaz de rastrear a atividade do usuário e registrar os padrões de procedimentos de um usuário típico, muitas tarefas podem ser desenvolvidas de muitas maneiras diferentes;

- o usuário pode estar desenvolvendo múltiplas atividades que se sobrepõem;
- os usuários podem ter intenções muito diferentes do que as suas ações sugerem;
- ainda, os próprios usuários podem não estar totalmente conscientes das suas intenções.

O computador poderia oferecer para a crítica do usuário uma interpretação das suas intenções, isto ajudaria no diagnóstico do problema e o próprio usuário participaria deste diagnóstico se isso lhe fosse permitido e facilitado (com *helps* adicionais, por exemplo com acesso a história das suas ações passadas, ou, com a apresentação de uma lista com os erros mais comuns naquela situação).

Onde é que eu estou?

Este tipo de questão é dependente da aplicação - hipertextos - por exemplo, onde os usuários usam metáforas espaciais.

Sellen e Nicol (1990) ainda recomendam que:

- o sistema de *help* use adequadamente os recursos das mídias disponíveis - *helps* on line podem ser interativos, dependentes do contexto relativo à ação e ao nível de habilidade do usuário; podem ainda ser dinâmicos, com animação do seu uso, inclusive através de voz; podem também incorporar o uso de palavras chaves e cortar informações do texto do *help* para levá-lo à sessão de trabalho;
- a distinção entre a interface e o *help* não precisa ser explícita, na medida em que a interface facilita ou restringe a ação do usuário, pois todo o tipo de *feed-back* que possa ser dado ao usuário (*prompt* para a próxima ação numa série de ações ou uma mensagem de erro) pode ser considerado como um *help on-line*;
- boas interfaces talvez não necessitassem de *helps on-line*, pois as mesmas poderiam ser auto explicativas, a questão é: "não seriam os sistemas de *helps* desculpas para interfaces ruins?" Parece que por enquanto a resposta ainda é não, considerando-se o poder e a flexibilidade do software interativo e o estágio atual do desenvolvimento de interfaces inteligentes.

O *help* pode também incluir perguntas ao invés de respostas. Perguntas do tipo: você sabia que...? ajudariam o usuário a melhorar a sua performance de utilização (uso de teclas de atalhos e também uso de todo o espectro de oportunidades que a interface oferece), a partir de pequenos pedaços de informação que são oferecidas ao usuário.

Como direções a serem seguidas na área de interação homem-máquina Sellen e Nicol (1990) sugerem o desenvolvimento de interfaces adaptáveis e tolerantes, estas devem reduzir a necessidade de *helps on- line*. Quanto mais sensitivos ao contexto forem os *helps* desenvolvidos mais eles deixarão de ser obtusos. Mas um conhecimento superficial do contexto não é suficiente para tal, pois o sujeito poderá estar num contexto não relacionado com a ajuda que precisa, e um *help* sensível ao contexto nesta situação só trará mais confusão. *Helps* mais inteligentes não precisariam esperar o pedido de ajuda do usuário, pois isso exige que ele saiba

fazê-lo. Sistemas mais avançados poderiam completar comando ou *prompts* para a próxima ação, poderiam mesmo fazer a tarefa para o usuário ao invés de lhe explicar como fazê-lo.

É preciso contudo cuidar pois *helps* não solicitados podem ser irritantes e ainda, quando o computador exerce excessivo controle sobre a ação do usuário este pode sentir-se como um espectador. Isto exige que o computador conheça a intenção do usuário.

Uma outra dica muito importante de Sellen e Nicol é a seguinte: sistemas de ajuda não devem necessitar de ajuda para serem utilizados.

3.5.3 As Possibilidades de Evolução

A evolução que se quer aqui abordar não corresponde ao desenvolvimento da própria ferramenta, em termos de programação ou projeto, ela diz respeito ao tipo de evolução que é levada a efeito pelo próprio usuário, na sua forma de usar o software.

O que é passível de mudanças consiste em:

- A criação de novos procedimentos efetivos adaptados aos modos de trabalho dos diferentes usuários;

- A repartição da pilotagem entre homem-máquina, ou seja, as noções de desencadeamento, de operações obrigatórias ou facultativas, e de precedências permanentes ou indicativas. De outro modo, aqui se analisa a possibilidade de se programar o deslanchamento ou não de determinadas operações sob determinadas condições.

- O nível das operações intermediárias que não são inicialmente previstas mas que utilizam operações já conhecidas da máquina. Isto vai permitir ao usuário que crie suas próprias seqüências padrão ou macro-operações próprias.

Se forem necessários novos tratamentos que não podem ser compostos a partir das operações já disponíveis na máquina, será necessário a programação de novos módulos.

4. Referências Bibliográficas

BARTHET, Marie F. *Logiciels interactifs et ergonomie*. Dunod. Paris, 1988.

CAMPOS, G. H. B. - Construção e validação de ficha de avaliação de produtos educacionais para microcomputadores. Dissertação de Mestrado-UFRJ. Rio de Janeiro, out 1989.

CARRAHER, D. W. - O que esperamos do software educacional. Acesso-FDE-Revista de Educação e Informática, 2(3):32-36, jan/jun 1990.

COBURN, P. et alli - Informática na educação.Livros Técnicos e Científicos. Rio de Janeiro, 1988.

GALVIS, A. H. - Ambientes de enseñanza aprendizage enriquecidos con computador. Boletin de Informatica Educativa, 1(2):117-139.Bogotá, dez 1988.

MOREIRA, M. - A questão da produção e da avaliação do software educacional. In: Seminário o Computador e a Realidade Educacional Brasileira, 2. Belo Horizonte, UFMG/Centro Piloto de Informática na Educação, mai 1987.

MOREIRA, M. - O uso do computador na educação: pressupostos psico-pedagógicos. Educação em Revista 4: 13-17. Belo Horizonte, dez 1986.

MOREIRA, M. et alli - Produção e avaliação de software educativo. Educação em Revista 6: 41-44. Belo Horizonte, dez 1988.

REGGINI, H. C. - El Pasajero de la gondola: reflexiones en torno a la educaion y a LOGO. Boletin de Informatica Educativa 3 (1): 9-17. Bogotá, abr 1990.

STAHL, M. - Software educacional: características dos tipos básicos. In: Anais do Simpósio Brasileiro de Informática na Educação 1: 34-46. Rio de Janeiro, nov 1990.

SELLEN, A e NICOL, A. *Building user-centered on-line help*. In "The art for human-computer interface design" edited by Brenda Laurel. Addison-Wesley Publishing Company. Massachusetts, 1990.