

QUESTÃO DO SOFTWARE: parâmetros para o desenvolvimento de software educativo

José Armando Valente
UNICAMP-NIED

INTRODUÇÃO

A implantação da informática na educação consiste basicamente de três ingredientes: o computador, o software educativo, e o professor treinado para o uso do computador na sala de aula. O software educativo tem tanta importância quanto os outros ingredientes, pois sem ele o computador jamais poderá ser utilizado como ferramenta educacional.

O objetivo deste documento é apresentar algumas questões relativas ao desenvolvimento de software educativo, juntamente com as soluções que foram adotadas em outros países, as novas tendências na área e as possíveis soluções que podem ser adotadas no Brasil. Os tópicos a serem discutidos são: os diferentes tipos de software educativo, desenvolvimento, avaliação e distribuição de software, e utilização do software na sala de aula.

O ponto de vista que norteia a apresentação destes tópicos é que o software a ser utilizado nas escolas não deve substituir as atividades educacionais já existentes -- ele não deve ser simplesmente uma versão computadorizada dos atuais métodos de ensino. O computador deve ser uma ferramenta de complementação, de aperfeiçoamento e de possível mudança na qualidade do ensino. Isto se faz necessário pela própria mudança na nossa condição de vida e pelo fato que a natureza do conhecimento mudou. Hoje nós vivemos num mundo dominado pela informação e por processos que ocorrem de maneira muito rápida e imperceptível. Os fatos e alguns processos específicos que a escola ensina, rapidamente se tornam obsoletos e inúteis. Portanto, ao invés de memorizar informação, os estudantes devem ser ensinados a procurar e a usar a informação. Estas mudanças podem ser introduzidas com a presença do computador que deve propiciar as condições para os estudantes exercitarem a capacidade de procurar e selecionar informação, resolver problemas e aprender independentemente. Esta deve ser a tônica que deve guiar o desenvolvimento de software educativo.

TIPOS DE SOFTWARE EDUCATIVOS

Uma das questões fundamentais no desenvolvimento de software educativo é o aspecto pedagógico -- o que o software se propõe a ensinar e como isto é feito. Quanto ao conteúdo, o computador tem sido utilizado para ensinar informática -- ensino da informática ou "computer literacy", e ensinar praticamente qualquer assunto -- ensino pela informática. Quanto à maneira como o ensino pela informática ocorre, o software pode ser classificado em 3 grandes

categorias: instrução auxiliada por computador, aprendizagem por descoberta, e ferramentas educacionais tanto para o aluno como para o professor. A seguir apresentamos uma breve descrição de cada um dos diferentes tipos de software, suas vantagens e desvantagens, e as novas tendências no uso da informática na educação em vista da experiência e dos atuais avanços computacionais.

* Ensino da informática ou "computer literacy"

Este termo tem sido utilizado para caracterizar o ensino de conceitos computacionais, como programação, princípios de funcionamento do computador etc. Entretanto, "literacy" é um termo vago e não determina o grau de profundidade do conhecimento que o aluno deve ter -- até quanto o aluno deve conhecer sobre computadores e técnicas de programação. Isto tem contribuído para tornar esta modalidade de utilização do computador extremamente nebulosa e facilitado a sua utilização como chamarisco mercadológico. E como tal, as escolas oferecem cursos de computação (onde os alunos têm acesso ao computador somente 1 hora por semana) cujo objetivo pode ser caracterizado como "conscientização do estudante para a informática", ao invés de ensiná-lo a programar.

De acordo com um estudo feito em 1983 pelo "Center for Social Organization of Schools" da Universidade Johns Hopkins, virtualmente todas as escolas públicas nos E.U.A. oferecem cursos sobre história do computador, o funcionamento do hardware, programação elementar e implicações sociais do computador na sociedade. Estes cursos são oferecidos à cerca de 36% dos alunos da escola elementar (os 6 primeiros anos do 1º grau), e à 64% dos alunos da escola secundária (os 2 últimos anos do 1º grau). Cerca de 76% das escolas do 2º grau oferecem estes cursos a seus alunos. A maioria das escolas propiciam a alguns estudantes cerca de 30 horas ou mais de instrução em pelo menos uma linguagem de programação -- BASIC é utilizada em 98% das escolas do 2º grau, e Logo, FORTRAN, e Pascal são cada uma ensinada em 5% destas escolas.

A tendência atual é criticar este tipo de ensino. Primeiro, pelo fato dele estar servindo a propósitos bastante limitados: o estudante não adquire um conhecimento profundo de programação, portanto ele não tem habilidades para conseguir emprego de programador. Segundo, argumenta-se que não é necessário conhecer superficialidades sobre computadores para viver numa sociedade computadorizada -- vivemos numa sociedade permeada de motores elétricos e as escolas não oferecem cursos de "electric motor literacy". Portanto, a solução é aprofundar mais os cursos e oferecer uma possibilidade vocacional ou eliminá-los de uma vez.

Os que defendem a continuação destes cursos propõem a utilização de linguagens mais simples e poderosas -- "languages to think with". Na área de processamento numérico a linguagem proposta é AMP -- uma versão do APL, e na área de processamento simbólico a linguagem proposta é Logo (se a ênfase é procedimento) ou PROLOG

(se a ênfase é declaração).

No Brasil a grande maioria (se não for 100%) das escolas que possuem computadores, os cursos que elas oferecem ou a ênfase da utilização dos computadores, pode ser caracterizada como "computer literacy".

* Ensino pela informática

"The Educational Products Information Exchange (IPIE) Institute" uma organização do "Teachers College", Columbia, E.U.A., identificou, em 1983, mais de 7000 pacotes de software educacional no mercado, sendo que 125 eram adicionados a cada mês. Eles cobrem principalmente as áreas de matemática, ciências, leitura, artes e estudos sociais.

- instrução auxiliada pelo computador (CAI) -- uma versão computadorizada dos métodos de instrução programada tradicionais. As categorias mais comuns desta modalidade são programas de reforço ou treino-e-prática ("drill-and-practice") e programas tutoriais.

Tipicamente os programas de treino-e-prática são utilizados para revisar material visto em classe, principalmente material que envolve memorização e repetição, como aritmética e vocabulário. Segundo um estudo feito pelo EPIE cerca de 49% do software educativo no mercado americano é do tipo treino-e-prática. Estes programas requerem a resposta frequente do aluno, propiciam feedback imediato, exploram as características gráficas e sonoras do computador, e geralmente são apresentados na forma de jogos. Por exemplo, "Alien Intruder" é um programa para criança da escola elementar que exige a resolução de problemas de aritmética o mais rápido possível para eliminar um "alien" que compete com o usuário. Cerca de 40% do tempo que a criança da escola elementar passa no computador é consumido com programas do tipo treino-e-prática.

Os programas tutoriais constituem cerca de 19% dos programas educacionais avaliados por EPIE. A vantagem deste sistema de ensino é o fato da criança aprender de acordo com o seu próprio ritmo, já que estes programas são conhecidos pela sua paciência infinita. Por outro lado, o desenvolvimento de um bom tutorial é extremamente caro e difícil. As indústrias de software educativo preferem gastar no aspecto de entretenimento -- gráficos e som conquistadores -- ao invés de gastar no aspecto pedagógico, ou no teste e refinamento do programa. EPIE verificou que cerca de 80% dos 163 programas mais utilizados não passou pela fase de teste em campo.

A tendência dos bons programas CAI é utilizar técnicas de inteligência artificial para analisar padrões de erro, avaliar o estilo e a capacidade de aprendizagem do aluno, e oferecer instru-

ção especial sobre o conceito que o aluno está apresentando dificuldade. O exemplo de um programa com estas características é o programa para auxiliar a detecção de problemas num circuito elétrico. Ele identifica o estilo de resolução de problemas do usuário, identifica dificuldades conceituais que o usuário apresenta e, através de instrução detalhada, levá-lo à assimilar estes conceitos. O problema com este tipo de sistema é o tamanho e a capacidade do recurso computacional que eles requerem -- os computadores pessoais não são ainda tão poderosos para permitir que estes programas cheguem até às escolas.

- aprendizagem por descoberta ("discovery-learning") -- a pedagogia por trás desta filosofia é a exploração auto-dirigida ao invés da instrução explícita e direta. Os proponentes desta filosofia de ensino defendem a idéia que a criança aprende melhor quando ela é livre para descobrir relações por ela mesma, ao invés de ser ensinada. Exemplos de software nesta modalidade são Logo, simulação, jogos e software para controlar processos, como atividades em laboratório. De acordo com o estudo da Johns Hopkins, 24% do tempo das crianças da escola elementar é gasto com jogos e a criança na escola secundária gasta 6% com jogos e 12% com as demais atividades.

Logo geralmente é apresentado através da Tartaruga (mecânica ou de tela) que se move no espaço ou na tela como resposta aos comandos que a criança fornece através do computador. Neste ambiente de aprendizagem a criança pode explorar conceitos de matemática, resolução de problemas, planejamento e programação. A dificuldade com a utilização de Logo na escola é a preparação do professor, capacidade do computador para processar Logo e o fato de Logo não poder ser utilizado em todas as áreas do conhecimento.

Simulação envolve a criação de modelos dinâmicos e simplificados do mundo real. Estes modelos permitem a exploração de situações fictícias, de situações com risco, como manipulação de substância química ou objetos perigosos, e de situações impossíveis de serem obtidas, como desastres ecológicos. Por exemplo, "Odell Lake" é um programa que permite a criança aprender ecologia dos lagos americanos. A criança é colocada no papel de uma truta que procura alimento evitando predadores e outras fontes de perigo. A simulação oferece a possibilidade da criança desenvolver hipóteses, testá-las, analisar resultados e refinar os conceitos. As áreas de aplicação são potencialmente mais amplas. Entretanto, o que existe no mercado em geral é extremamente simples e trivial. Um bom programa de simulação geralmente é complicado para ser desenvolvido, requer grande poder computacional, gráfico e sonoro, e não garante que o aprendizado da criança possa ser aplicado à vida real. A visão que a criança forma a respeito do mundo pode ser distorcida e ela pode ser levada a pensar que o mundo real pode ser simplificado e controlado da mesma maneira que nos

programas de simulação.

Os jogos do ponto de vista da criança, constituem a maneira mais divertida de aprender. Talvez o melhor exemplo de um jogo educacional no mercado seja "Rocky's Boots" -- uma coleção de 39 jogos desenvolvido para ensinar crianças (a partir de 9 anos de idade) conceitos de lógica e de circuito de computadores. Usando componentes eletrônicos a criança monta o seu próprio circuito. O fato dele estar certo ou errado depende de como o circuito funciona e se ele ajuda a criança a atingir determinados objetivos estabelecidos pelos jogos. O grande problema com os jogos é que a competição pode desviar a atenção da criança do conceito envolvido no jogo. Além disto a maioria dos jogos explora conceitos extremamente triviais, e não tem a capacidade de diagnóstico das falhas do jogador.

Os programas de controle de processo oferecem uma ótima oportunidade para a criança entender processos e como controlá-los. Um dos melhores exemplos de programa nesta área é o "TERC Labnet", desenvolvido pela "Technical Education Research Centers". Uma coleção de programas que permitem a coleta de dados de experimentos, a análise destes dados, e a representação do fenômeno em diferentes modalidades, como gráfico, e sonoro. Outro exemplo é o projeto Lego-Logo que o "MIT Logo Laboratory" está desenvolvendo. Utilizando o brinquedo Lego a criança monta diversos objetos que ela aprende a controlar através do Logo. Bons programas nesta área requerem muita criatividade e imaginação e são poucos programas no mercado que têm estas qualidades.

* Ferramentas para alunos e professores

Programas de processamento de texto, planilhas, manipulação de arquivos, construção e transformação de gráficos, sistemas de autoria, calculadores numéricos, são ferramentas extremamente úteis tanto ao aluno como ao professor. Talvez estas ferramentas constituam uma das maiores fontes de mudança do ensino e do processo de manipular informação. As outras modalidades de software educativos descritos acima podem ser caracterizados como uma tentativa de computadorizar o ensino tradicional. Mais ou menos o que aconteceu nos primórdios do cinema -- cinema = teatro + câmera. Hoje o cinema tem sua indústria própria. Este mesmo fenômeno está acontecendo com o uso de computadores na educação. Com a criação destas ferramentas de manipulação da informação estamos vendo nascer uma nova indústria de software educativo que pode causar um grande impacto na maneira como ensinamos e como nos relacionamos com os fatos e com o conhecimento. Exemplo de ferramentas desenvolvidas especialmente com objetivos educacionais são os programas do "Bank Street", sendo o seu processador de texto o mais conhecido; a combinação de Logo e processamento de texto que a "Logo Computer System" está colocando no mercado (pode-se combinar texto e gráfico ao mesmo tempo); e alguns "sistemas especialistas" desenvolvidos para computadores de grande porte mas que podem ser adaptados para

alguns micros, como da linha PC.

O DESENVOLVIMENTO DE SOFTWARE EDUCATIVO NO BRASIL

No mercado brasileiro existe um pequeno número de software educativos que já foram desenvolvidos, graças à iniciativa do governo, através de projetos de pesquisa (como o EDUCOM) e a iniciativa privada. Entretanto, o número destes programas é muito pequeno.

A Biblioteca Brasileira de Software (BBS), empresa paulista que detem a maior variedade de software, tem catalogado cerca de 5.000 programas para quase todos os micros. A maioria destes programas são adaptados de programas estrangeiros. Isto significa que estes programas, provavelmente, tem muito pouco em comum com o nosso sistema educacional ou com nossa cultura. Recentemente uma série de empresas de desenvolvimento de software têm se preocupado com a produção de software educativo. Estes programas têm sido desenvolvidos em colaboração com professores que utilizam o computador e que desenvolvem algum software para ensinar algum tópico específico. Este programa, em geral, constitui a semente de um programa mais elaborado desenvolvido e comercializado por estas empresas. Entretanto, estes programas padecem de uma série de problemas: são desenvolvidos para situações educacionais muito particulares, o grau de sofisticação e a qualidade da documentação são pobres, e o baixo número de software comercializado faz com que o preço seja muito alto.

O MEC tem procurado incentivar o desenvolvimento de software através de dois projetos. Um é o Concurso Nacional de Software Educativo que avaliou mais de 150 software educativos em dois anos de realização do Concurso. Cerca de 30 destes programas foram catalogados e podem ser adquiridos para uso em escolas. Outra iniciativa tem sido os projetos de pesquisa sobre o uso da informática na educação. O projeto EDUCOM tem produzido diversos software educativos de boa qualidade e tem avaliado estes programas tanto sob o aspecto de efetividade pedagógica como o seu impacto no aluno e no sistema escolar.

Já para o desenvolvimento de ferramentas de caráter educativo a iniciativa tem sido isolada e ficado por conta das empresas -- como o desenvolvimento do Logo pela ITAUTEC -- ou os programas desenvolvidos para empresas que podem ser utilizados nas escolas, como as planilhas e os processadores de texto. Entretanto, com relação a estas últimas ferramentas, o aluno carece de algo mais simples do que está disponível no mercado. Algo que o permita concentrar nos conceitos envolvidos nestes programas ao invés de ter que dominar uma enorme quantidade de características que estão presentes nestes programas.

PADRÕES DESEJÁVEIS PARA SOFTWARE EDUCATIVO

O software educativo a ser desenvolvido deve satisfazer padrões que deverão ser definidos pela comunidade educacional. Neste tópico são apresentados alguns pontos que devem ser levados em consideração na elaboração destes padrões. A definição dos padrões é necessária para que a produção de software possa ser maximizada com um mínimo de recurso.

A definição dos padrões deve levar em conta alguns itens como:

a) engajamento do usuário com o sistema

- os programas devem ser interativos. O diálogo ou a interação com o computador é fundamental para o usuário engajar-se num verdadeiro processo de ensino-aprendizagem.
- a qualidade do diálogo. O diálogo do computador com o usuário deve ter algumas características como ser amigável, encorajador, apropriado, útil e fluído.
- o programa não deve julgar o usuário. Responder a um erro com "dicas" que não penalizam, que fazem a pessoa reconsiderar a resposta, e que levam o usuário a corrigir seu erro são características essenciais para manter o diálogo amigável e fluído.
- reações altamente positivas ou negativas são contraproduativas. A qualidade do diálogo perde muito com reações do tipo "fantástico", "fabuloso". As reações mais brincalhonas, não verbais, são consideradas mais apropriadas e mais encorajadoras, considerando-se que se trata de um diálogo com uma máquina.
- O programa deve oferecer ajuda. Com o objetivo de não quebrar o diálogo o programa deve oferecer ao usuário a oportunidade de solicitar ajuda em qualquer parte do programa.

b) controle do aprendizado

- o controle do aprendizado deve estar nas mãos do estudante. Os programas educativos devem ser desenvolvidos de tal maneira que o estudante sinta que suas ações não estão sendo limitadas pelo programa. Assim, o programa deve antecipar uma ampla variedade de respostas, desde o chute até as respostas obtidas estrategicamente.
- permitir mais que uma maneira de resolver o problema. O estudante deve ser capaz de resolver o problema de acordo com o seu estilo e capacidade cognitiva.

c) o valor do erro. Em geral, o processo de ensino tradicional não força o professor a ser explícito sobre como os erros devem ser tratados. No caso do computador este tratamento é obrigatório.

E o que é feito com o erro deve seguir um plano consistente e explicitamente determinado.

- o feedback deve ser neutro quanto à direção a ser seguida. Qualquer informação ou direção fornecida além da apropriada é vista como intervenção.
- a tentativa de fornecer um feedback "inteligente" pode ser perigoso. O usuário percebe a "inteligência do programa" e passa a se interessar mais pelo funcionamento do programa do que pelo conteúdo em questão.
- o programa deve reconhecer sua limitação e ser "humilde". Ao invés de tentar solucionar todas as situações o programa deve, em alguns casos, recomendar que o estudante solicite ajuda do professor.

d) programação sólida e efetiva

- importância do uso de diferentes tipos de representação da informação. Uma declaração verbal ou matemática pode ser visualmente melhorada através do uso de gráficos, cores etc. Por outro lado, estes efeitos devem ser utilizados em doses corretas para não se tornarem fontes de distração.
- intenção clara. O programa deve ser claro quanto aos seus objetivos, como eles serão atingidos e como o usuário deve fornecer suas respostas.
- uso do que já passou. O programa deve permitir ao usuário refletir sobre o que ele fez anteriormente. Ele deve ser capaz de rever o que foi feito ou o programa deve mencionar como o usuário se comportou em situações anteriores.
- um pouco de suspense ajuda a manter o interesse do estudante. O programa deve propor questões desafiantes cujo grau de dificuldade deve ser tal que o estudante possa resolvê-las.
- engajar mais de um estudante. A maioria dos programas são desenvolvidos para serem utilizados por um único usuário. Entretanto, uma situação de aprendizado bastante estimulante pode ser criada se duas ou mais pessoas interagem com o computador.

e) documentação

- os manuais do programa devem ser bem escritos. O nível de explicação de como o programa deve ser iniciado, mensagem de erro etc., deve ser tal que o professor e o aluno sejam capazes de entendê-la.

É importante mencionar que os itens propostos tem o carácter de identificar certos aspectos necessários para a produção de um bom software educativo. Eles não devem ser todos satisfeitos já que seria impossível desenvolver um software que tivesse todas estas características. Por outro lado, a existência destas características não é condição suficiente para produção de um bom software. A combinação dos diferentes aspectos aqui propostos é que fazem um software ser mais interessante e efetivo que outro.

PRODUÇÃO DE SOFTWARE

Outro assunto a ser considerado no desenvolvimento de software educativo é quem produz estes programas. As ferramentas, pela sua própria natureza, são programas que exigem um grau de sofisticação computacional e conhecimento interno da máquina, que talvez somente o fabricante do computador seja capaz de produzi-las. Já os outros tipos de software educativos devem ser produzidos pela iniciativa privada. O governo pode facilitar o financiamento através de órgãos de pesquisa como CAPES, CNPq, FINEP etc. Assim, uma pessoa interessada em desenvolver um determinado software deve submeter uma proposta à um destes órgãos. Esta proposta deve ser analisada segundo o padrão de software previamente estipulado e o controle do desenvolvimento do projeto, bem como o seu financiamento, são responsabilidades do órgão que aprovar o projeto.

Uma vez o produto pronto (fruto da iniciativa privada ou financiado por algum órgão governamental), se for do interesse do seu produtor ter este software utilizado pelas escolas públicas, ele deve submetê-lo ao MEC ou a Secretaria de Educação Estadual ou Municipal que deve avaliar e catalogar este programa. Uma vez o software disponível, ele pode ser requisitado pelas escolas mediante uma política de distribuição, descrita abaixo.

É importante afirmar que todos os tipos de software educativos devem ter a sua produção encorajada. Existem estilos de aprendizagem diferentes que devem ser respeitados. Os programas do tipo CAI adaptam-se melhor aos alunos que preferem situações de aprendizagem mais definidas. Outros preferem explorar e certamente rendem mais com programas que facilitam a aprendizagem por descoberta.

Outros fatores que devem ser levados em conta na produção de software são a participação do professor na elaboração do software, e o teste do software. Sem este envolvimento o software tenderá a ser mais um dos tantos materiais de ensino que serão produzidos em "laboratórios" sem levar em conta a experiência e conhecimento do professor. A falta de teste do software durante o seu desenvolvimento tem produzido certas aberrações educacionais que deveriam ser detectadas em situações de teste que sejam o mais próxima possível da situação de utilização. Sem estes testes será impossível avaliá-los corretamente e conhecer as suas funções educacionais.

AVALIAÇÃO DE SOFTWARE

O software produzido deve ser avaliado tanto do ponto de vista da presença ou ausência dos itens enumerados acima (componentes de um padrão), como do ponto de vista da sua efetividade como ferramenta educacional.

A avaliação mais fácil de ser feita é a relativa aos itens pertencentes ao padrão. Neste caso, de acordo com a performance do software pode ser atribuído pontos por itens. Uma vez estes pontos atribuídos, o software para ser considerado aprovado deve satisfazer um mínimo que pode ser: o resultado da média dos pontos, ou satisfazer um mínimo em todos os itens ou em itens que são considerados essenciais.

A avaliação da efetividade pedagógica do software é muito mais complicado de ser feito. Isto exige uma pesquisa que requer a formação de grupos experimentais de controle, e o monitoramento de certas variáveis, como o nível intelectual da população antes e após o uso do software. Os resultados destes estudos, em geral, são muito controvertidos, pois é muito fácil detectar falhas que invalidam o estudo da efetividade do software. Outros fatores que dificultam este tipo de avaliação são o alto custo de uma validação deste tipo, e o fato das empresas produtoras de software não disporem do conhecimento envolvido neste tipo de avaliação. Assim, a avaliação que tem sido apresentada se restringe à existência ou não de certas características previamente definidas como essenciais.

Como parte do processo de avaliação do software seria interessante que fosse elaborado um documento onde seria descrito a razão dos pontos atribuídos a cada item. Este documento deveria fazer parte da sua documentação -- ou um documento à parte. Esta é uma maneira de informar o professor sobre as vantagens e desvantagens daquele software, propiciando-lhes melhores condições de utilizá-los em sala de aula.

DISTRIBUIÇÃO DE SOFTWARE

Este é um problema bastante complicado e controvertido. Existem diversos modelos que podem ser seguidos.

O modelo mais interessante é a distribuição de software através do livre mercado. Atualmente, nos E.U.A., o problema com este tipo de distribuição é a falta de oportunidade que o professor tem em poder experimentar o software antes de comprá-lo.

Talvez o modelo de distribuição a ser adotado devesse ser o mais próximo possível deste. Primeiramente, não deve ser função do MEC ou Secretarias de Educação produzir, comercializar ou paternalizar a distribuição de software. A função destes órgãos deve ser a de facilitar o processo de experimentação do software por parte do professor, por exemplo colocando-os à disposição para experimen-

ção através das delegacias de ensino, bibliotecas, ou empréstimos, e deixar o controle de compra ser exercido pela escola -- a escola deve ter uma verba para compra de software e deve ser sua a responsabilidade da compra do software que melhor satisfaz às suas necessidades.

Uma outra alternativa é um modelo mais controlado pelo MEC ou pelas Secretarias de Educação. Por exemplo, à cada software catalogado seria atribuído um determinado "preço". Este preço reflete o tipo de software -- se for ferramenta custa mais caro, se for programa treino-e-prática tem um outro valor. À cada escola seria também fornecido uma cota por ano ou semestre que seria utilizado para a "compra" de software. A responsabilidade de qual e quantos software educativos a serem "comprados" deve ser responsabilidade da escola.

A vantagem do primeiro modelo é que o controle de "copyright", pirataria de software, lucro do produtor, ficaria totalmente a cargo do livre mercado. A desvantagem é a passagem de recursos para cada escola e o controle destes gastos.

Já no segundo modelo não existe o controle de verbas gastas por escola. Por outro lado, tem-se que montar todo um sistema de controle do software. Isto pode ser resolvido através da compra de todos os direitos do software pelo MEC ou pelas Secretarias. Assim todo software catalogado seria propriedade do sistema educacional público, eliminando grande parte dos problemas de distribuição do software. Mesmo assim, existiria o problema do controle de programas ferramentas, que como foi dito acima, são programas que atendem a uma larga faixa do mercado e os seus produtores têm interesse em explorá-los comercialmente.

UTILIZAÇÃO DE SOFTWARE NA ESCOLA

Além das questões levantadas acima, uma outra questão de grande importância é como este software é utilizado na sala de aula. Aqui cabe algumas considerações como:

- a) A utilização do software por si só é suficiente ou ele deve ser utilizado conjuntamente com outros materiais, com ou sem a intervenção do professor?
- b) O aspecto social do aprendizado -- ele ocorre individualmente, em pequenos grupos, ou em grupos maiores? É impossível pensar em um computador para cada aluno na escola. Como seria esta distribuição de tempo do aluno com a máquina?
- c) Qual é o relacionamento entre o tipo de software e a prática do ensino feito na sala de aula? O software pode incentivar a descoberta enquanto o professor pode desencorajá-la. Outro fator é o relacionamento entre o material computadorizado e o currículo em prática. Se este material for muito distante do currículo

lun e não permitir uma avaliação do desempenho de cada aluno, o computador terá muita dificuldade em entrar na escola.

- d) Qual deve ser o preparo do professor para enfrentar as mudanças que o computador ocasionará na escola? Simplesmente colocar nas mãos do professor uma nova ferramenta, não garante que ela será bem utilizada e que trará os benefícios que são esperados. O professor deve ser treinado e deve existir programas de reciclagem para que ele tenha contato com as novas tendências da área.

CONCLUSÕES

As questões aqui consideradas podem servir de base para a definição de uma política de desenvolvimento de software educativo. Esta política é de suma importância face a escassez de recursos na área educacional. Em outras condições econômicas talvez a melhor política seria a livre produção de software. A lei de mercado se encarregaria de prestigiar os melhores e eliminar os piores software, como acontece nos E.U.A. Entretanto, a estrutura centralizada do nosso sistema educacional e a falta de recurso desta estrutura faz com que haja a necessidade de uma política de desenvolvimento do software educativo.

Por outro lado, somente a existência da política não garante a existência do software. É preciso que haja também incentivos à sua produção e distribuição. Sem a integração destes esforços corremos o risco de ter o computador na escola, o professor treinado e estes recursos não utilizados pela falta de software educativo. Esta situação seria muito semelhante ao que aconteceu com a distribuição de televisores nas escolas públicas: muitas delas não dispunham de rede elétrica.