Integração da disciplina de Sistemas Digitais na plataforma PmatE

Sérgio Barbosa, Alexandre Moreira, Manuel Bernardo Cunha, Augusto Silva

***Resumo* – Este artigo pretende apresenta o processo de integração da Disciplina de Sistemas Digitais na Plataforma do Projecto Pmate, desenvolvida para apoiar o ensino assistido por computador.**

Começa por descrever como nasce e qual o âmbito do projecto, identifica os requisitos necessários antes de começar a produzir conteúdos, descreve procedimentos intrínsecos à organização, áreas de desenvolvimento, tecnologias utilizadas, inclusive alguns modelos conceptuais do sistema de informação.

Termina apresentando exemplos das ferramentas oferecidas, chamando atenção para as potencialidades que este tipo de ferramentas tem num contexto dum curso de Sistemas Digitais.

***Abstract* -.This article approaches the process of integration of a Digital Systems course in the Mathematical Education Project platform, developed to support computer aided education.**

The paper describes how the project appeared and its scope, identifies the needed requirements needed before actual content production, describes the intrinsic procedures of the organization, development areas, used technologies and also some of the conceptual models of the information system.

Several developed tools are presented and their main features are discussed in the framework of an undergraduate Digital Systems course

I. INTRODUÇÃO

O Projecto Matemática Ensino (P**mat**E) nasce no departamento de matemática da Universidade de Aveiro, através da iniciativa de um conjunto de professores que se reuniram em torno de uma ideia: modelo gerador de questões, motivados pela vontade de promover e estimular o gosto pela matemática. Este conceito pode ser definido como sendo uma estrutura, com suporte informático, capaz de gerar uma afirmação, designada de texto, e quatro proposições sorteadas aleatoriamente, às quais se deve responder "VERDADEIRO" ou "FALSO". Este tipo de exercício com características dinâmicas e designado de modelo, é a unidade que está na base da construção de todas as provas criadas no âmbito do P**mat**E. Provas de cariz competitivo associadas a todos os ciclos de ensino e provas integradas no modelo de avaliação de algumas disciplinas do ensino superior, leccionadas na Universidade de Aveiro.

A tecnologia inicial permitia apenas a realização de provas de matemática divulgadas a partir de CD's. Actualmente, com a expansão da utilização da Internet, foi possível desenvolver uma arquitectura baseada no modelo cliente-servidor, centralizando os conteúdos numa base de dados e potenciando o acesso às provas, possível com apenas uma ligação à Internet e a utilização de um simples browser, suportado por dois plug-in's.

É neste contexto que a disciplina de Sistemas Digitais é convidada a integrar a plataforma e a produzir conteúdos com formatos compatíveis com os já existentes e a desfrutar deste tipo de ferramenta que permite que os alunos possam aferir o seu conhecimento e os professores gerar provas de avaliação.

II. INTEGRAÇÃO

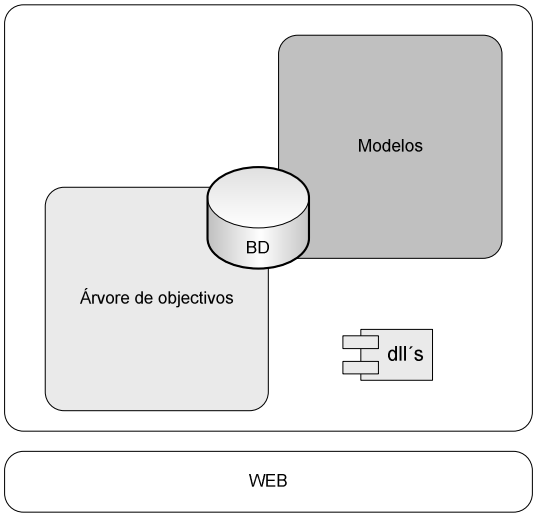

Figura 1 - Áreas de trabalho

Sistemas Digitais entrou para o projecto P**mat**E com o objectivo de estudar a integração de novos conteúdos, nomeadamente conteúdos das áreas de engenharia e criar um conjunto suficiente de modelos capaz de cobrir a área programática da disciplina de Sistemas Digitais. Conseguida esta meta inicial, os modelos criados são integrados em provas de treino e avaliação, com os formatos disponíveis na plataforma, e cujo objectivo será fornecer uma ferramenta alternativa e complementar ao estudo e avaliação da disciplina.

As áreas de intervenção na plataforma necessárias para criar provas encontram-se na Figura 1 e resultam nas seguintes acções:

- Pesquisa e inserção na base de dados de uma árvore de objectivos capaz de cientificamente indexar os conteúdos criados – uma árvore de objectivos é uma estrutura ramificada e hierarquizada, correspondente a uma determinada área científica, estruturada de objectivos mais gerais para objectivos mais específicos;
- Criar modelos e programar as diferentes componentes nas respectivas áreas da plataforma;
- Codificar modelos – processo de associação do modelo à designação que melhor descreva o objectivo que o modelo se propõem avaliar.

A. Árvore de objectivos

A árvore de objectivos resultou de uma pesquisa profunda na WEB, com o objectivo de encontrar uma especificação de uma área científica que incluísse os conteúdos programáticos da disciplina de Sistemas Digitais.

Foram estudados alguns sistemas de classificação, nomeadamente o CDU (Classificação Decimal Universal) e a ACM *Computing Classification System* [1] A solução encontrada para a árvore de objectivos resultou de uma estrutura base retirada da classificação da ISNPEC [2], à qual foram acrescentados os ramos necessários para especificar a totalidade dos conteúdos programáticos definidos para a disciplina. O resultado final é uma classificação híbrida, que pode a qualquer momento ser alterada, acrescentando ou retirando ramos de especificação.

A estrutura genérica da árvore de objectivos está descrita na Figura 2. Todas as áreas científicas com conteúdos na base de dados têm de ter uma árvore que obedeça a esta estrutura.

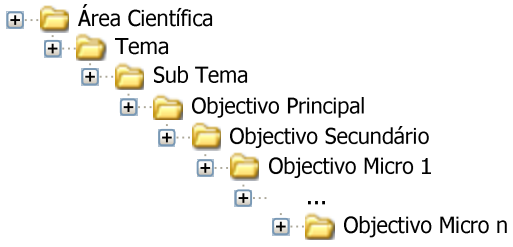

Figura 2 - Estrutura da árvore de objectivos

A Figura 3 representa um exemplo de um fragmento da árvore de objectivos. Neste exemplo é possível verificar toda a hierarquia, começando no objectivo de carácter mais genérico (Comunicações) até o objectivo mais específico (Objectivo Micro 1). Sempre que não for possível identificar o objectivo através do objectivo micro n, é sempre possível definir um outro de nível superior, objectivo micro (n+1).

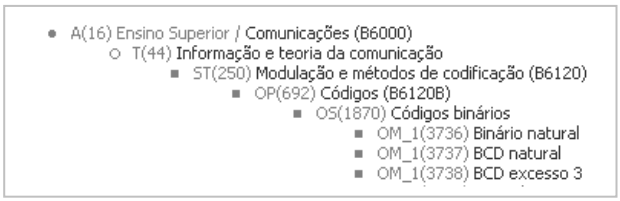

Figura 3 - Fragmento da árvore de objectivos

B. Elaboração de modelos

Na primeira fase de todo o processo elege-se um objectivo que os alunos se devem propor atingir. Segue-se uma outra fase que exige um conjunto de factores que suportem a elaboração de um modelo. Entre outros, será necessário ter uma ideia de um contexto que dê sentido a um conjunto de afirmações sobre um determinado tema. Este contexto pode ser original, pode resultar de um exercício proposto num livro, um exercício de um exame, etc. Não existem regras bem definidas, prevalece a criatividade.

Ultrapassada a etapa mais criativa, é necessário reunir toda a informação descritiva do modelo num documento específico, que obedece a uma norma de circulação interna do P**mat**E e que acompanha todas as fases de desenvolvimento: Programação, Codificação e Avaliação. Este documento deve conter toda a informação necessária para conduzir o modelo desde o momento da programação até ao instante em que é incluído numa prova.

C. Codificação de modelos

A codificação é o processo de associar o modelo a um objectivo secundário. Esta associação permite que os modelos se encontrem referenciados e agrupados por objectivos, favorecendo a organização do processo de criar provas. Quando se pretende avaliar um aluno num objectivo, determinam-se assim os modelos que se podem utilizar.

A árvore de objectivos tem ainda em vista soluções que estão por implementar. O objectivo é encontrar um mecanismo de condução do aluno entre objectivos, em função do seu desempenho. Este mecanismo deve ser capaz de propor exercícios de complexidade e nível cognitivo adequado, mediante o bom ou mau desempenho do aluno.

III. MODELO

Um modelo pode ser definido como uma estrutura de informação composta, principalmente, por 3 secções

distintas. *O domínio de parâmetros*, necessário para definir os conjuntos de valores a gerar e satisfazer o sorteio aleatório de parâmetros do modelo; *Texto*, proposição principal do modelo que se apresenta como enunciado introdutório do exercício, e por fim, *Complementos de proposição*, um mínimo de 4 orações que, juntamente com a proposição principal, complementam as afirmações resultantes da leitura continua de ambas as partes. Complementam ainda estas 3 secções todas as informações necessárias para codificar o modelo, informação bibliográfica, identificação do autor, anexos com algorítmos e qualquer outro tipo de informação pertinente, essencial na fase de programação.

Para clarificar a estrutura do documento criado por cada modelo, colocou-se na Tabela 1 uma forma aproximada de o representar.

Domínio de parâmetros		
a, b, c 5 {x, $\overline{X}$ }, com $\overline{X}$ a ser o complemento de x; a ≠ b b ≠ c		
Texto inicial da proposição		
O resultado de		
R	Complemento de proposição	Verdade se
1	a•(b+c) = a•b+a•c → c_{11}; a•b+a•a → c_{12}; b•b+a•c → c_{13}	c_{11}
2	a•b = 1 → c_{11}; 0 → c_{12}	c_{12}
3	a•b•c → c_{11}; a• c → c_{12} = 1 → C_{21}; 0 → C_{22}	(c_{11}^ c_{22}) v (c_{12}^ c_{21})
4	a • → c_{11}; + → c_{12} b = 0	c_{11}

Tabela 1 - Modelo

Domínio de parâmetros

Na zona de domínio de parâmetros devem ser especificados todos os parâmetros referenciados nos complementos de preposição e respectivo domínio a que pertencem. Devem ainda ser nesta zona definidas as relações de dependência entre parâmetros. Por exemplo, na Tabela 1 foi definido que o parâmetro ***a*** tem, obrigatoriamente, de ser diferente de ***b***.

Texto (Proposição)

Componente introdutória do exercício, comum a todas as afirmações resultantes da concretização/instanciação do modelo. No exemplo enunciado na Tabela 1, a expressão "O resultado de" representa o texto do modelo.

Complemento de proposição

Como o próprio nome indica, complementa a proposição principal dando corpo à afirmação. Segundo a estrutura adoptada, cada modelo exige um mínimo de 4 complementos, que se apresentam segundo uma ordem aleatória relativamente àquela indicada na Tabela 1.

De salientar que as respostas são do tipo Verdadeiro/Falso generalizado uma vez que a concretização pode conduzir a uma solução com todas as respostas falsas, todas verdadeiras e todas as outras combinações possíveis.

O número de formas diferentes que um complemento pode apresentar está sempre associado a uma de duas soluções possíveis. É muito importante que do universo de afirmações possíveis haja um equilíbrio entre o número de soluções verdadeiras e falsas, para que em momentos de avaliação não seja perceptível uma determinada tendência de resposta.

Do modelo representado pela Tabela 1, as concretizações possíveis para o complemento da proposição 4 (R4) são as seguintes:

- x • $\overline{X}$ =0 – solução: Verdadeiro;
- $\overline{X}$ • x =0 – solução: Verdadeiro;
- x + $\overline{X}$ =0 – solução: Falso;
- $\overline{X}$ + x =0 – solução: Falso.

Um exemplo de uma concretização do modelo é apresentado na Tabela 2.

Exercício	
O resultado de	
x•($\overline{X}$ +x) = x• $\overline{X}$ +x•x	(•) Verdadeiro ( ) Falso
x • $\overline{X}$ = 1	( ) Verdadeiro (•) Falso
x • $\overline{X}$ • x = 0	(•) Verdadeiro ( ) Falso
x + $\overline{X}$ = 0	( ) Verdadeiro (•) Falso

Tabela 2 - Concretização do modelo da tabela 1

IV. PROGRAMAR UM MODELO

A Programação de um modelo pode ser dividida em três fases: Programação de conteúdos, Programação do

domínio de parâmetros e validação, e por fim, avaliação do modelo.

Programação de conteúdos

Traduz-se na transcrição para a base de dados dos campos da Tabela 1 designados de *Texto inicial da proposição* e *Complementos de proposição*. Esta transcrição envolve a utilização de *LRM* (Linguagem de representação de modelos), que é uma linguagem própria, desenvolvida dentro do projecto, e que serve para representar toda a simbologia matemática que é posteriormente traduzida para *mathML*.

A programação de conteúdos é feita na base de dados, para uma estrutura tabelar bastante simples, de acordo com a imagem da Figura 4.

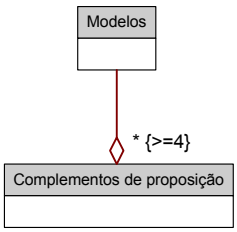

Figura 4 - Modelos na base de dados

Da leitura da imagem é possível concluir que um modelo deve ter pelo menos 4 complementos. Quanto maior o número de complementos, maior é a aleatoriedade aquando da instanciação do modelo.

Sempre que o número de complementos excede o número limite é muito importante que o grau de dificuldade, ou esforço necessário para chegar à resposta, seja comparativamente idêntico entre complementos, para evitar desigualdades em momentos de avaliação.

De salientar ainda que a LRM, não só é responsável pela descrição de todas as representações matemáticas, como descreve também a estrutura dos complementos – os complementos, quase sempre, contêm estruturas do tipo array, que representam diferentes alternativas no momento da instanciação do modelo (Tabela 3).

Falta apenas referir que os modelos têm parâmetros que são concretizados quando são instanciados

Exemplos de representações em LRM		
R Formal	LRM	Descrição
a c_{11} b c_{12}	\\{a//b}	Sortear aleatoriamente entre os parâmetros {a, b}
a é par.	\\a é par.	\\a representa em LRM um parâmetro que deve ser substituído por um valor pertencente ao seu domínio.
$\frac{a}{b}$	\\[a;b]	Razão entre a e b

Tabela 3 - Representações de LRM

Programação...

a) Programação do domínio de parâmetros – O domínio de parâmetros é programado utilizando a linguagem Visual Basic. A arquitectura de software é muito simples. Existe uma função que recebe o parâmetro encontrado, gera um valor entre o domínio definido para esse parâmetro e devolve-o, para ser integrado nas proposições (Texto inicial da proposição e complementos de proposição), de acordo com o esquema da Figura 5.

Esta característica confere ao modelo uma das suas componentes dinâmicas. Ou seja, cada instanciação resulta num conjunto de parâmetros diferentes.

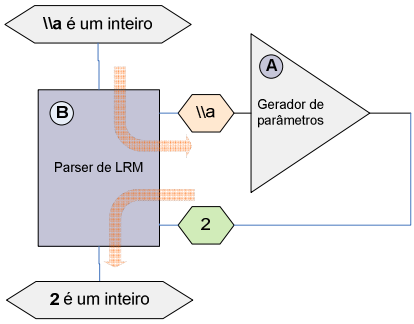

Figura 5 - Substituição de variáveis

b) Programação da validação – a instanciação do modelo resulta num conjunto de proposições cujas soluções variam em função do sorteio dos diferentes componentes dinâmicos. Paralelamente à programação do domínio de parâmetros, existe a programação da validação. A validação determina qual a solução lógica de cada afirmação, de acordo com a expressão lógica associada a cada complemento de proposição, conforme está descrito na Tabela 1, coluna da direita.

O processo de validar é assegurado por uma função que recebe como entradas os parâmetros e segmentos sorteados e testa a combinação apurada devolvendo apenas um de dois resultados possíveis: "*Verdadeiro*" ou "*Falso*".

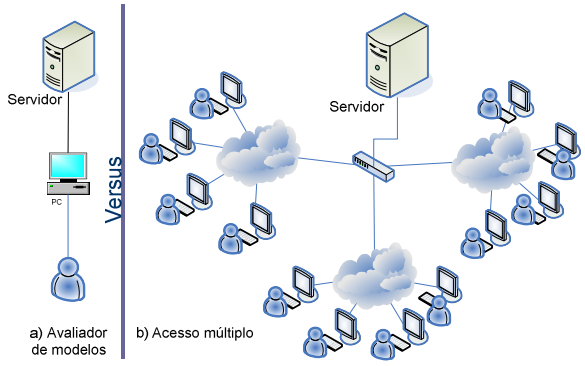

Figura 6 - Avaliação versus utilização

Avaliação de um modelo

Dependendo da complexidade do modelo, a tarefa mais penosa resulta da avaliação exaustiva de todas as

concretizações (quando possível). Adjectivou-se a tarefa de penosa pelo simples facto de na maioria dos casos, representar um número muito elevado de possibilidades.

A avaliação requer uma análise minuciosa, por parte de alguém cientificamente qualificado na área em que modelo foi codificado, tentando simular aquilo que é conseguido, de forma quase instantânea, quando o modelo é utilizado por grandes massas de alunos. Em termos ideais era desejado que o ambiente de validação tivesse o mesmo contexto que existe em provas , mas por motivos de limitação de recursos o paradigma é aquele descrito pela Figura 6. De acordo com este processo torna-se impossível definir um modelo como um produto acabado.

IV. TECNOLOGIAS DE SUPORTE PARA REPRESENTAÇÃO GRÁFICA E DE EXPRESSÕES MATEMÁTICAS

Suporte gráfico

A utilização da Internet como meio de distribuição dos conteúdos, exige que se tenha algum cuidado com as opções tomadas sobre as tecnologias de suporte informático para representação de imagem, expressões matemáticas e outras que possam surgir. Uma arquitectura que exige um fluxo de alguma intensidade na rede, ainda que esse fluxo seja irregular, deve dar preferência a formas de representação leves, de modo a ocupar reduzida largura de banda.

É exemplo disso o SVG (*Scalable Vector Graphics*) [3] que permite a inclusão de imagens sob a forma de objectos SVG em páginas HTML, e cuja interpretação fica a cargo de um plug-in disponibilizado pela Adobe (*SVG viewer*). O uso desta tecnologia, embora ainda sem grande suporte, traz grandes vantagens, uma vez que a informação é enviada sob a forma de texto, contribuindo para uma ocupação reduzida da largura de banda da rede e melhor desempenho do sistema ao nível dos tempos de espera. Esta qualidade de serviço é obtida sem descorar a qualidade gráfica das imagens.

A utilização do SVG como tecnologia de representação gráfica revela-se assim uma boa opção para um sistema que precisa de construir imagens dinâmicas em tempo real. Não necessita de criar nenhum ficheiro para suporte da imagem, segue apenas a mesma filosofia da linguagem XML, ou das linguagens de marcação, identificando numa *string* a imagem através de um marcador inicial e final que o plug-in consegue reconhecer.

O estado da arte relativamente aos browsers no mercado coloca o "Opera" na linha da frente [4]. Pelo menos a versão beta mais recente consegue interpretar correctamente tanto ficheiros SVG puros como SVG embutido em páginas HTML. Um outro browser, *Firefox*, da Mozilla [5] também se apresenta completamente independente dos *plug-ins*, embora com alguns comportamentos de instabilidade em sistemas operativos como o Windows 2003. O Netscape [6] pertence ao grupo dos browsers que não suportam nativamente o SVG embora as últimas notícias sobre as versões beta anunciem compatibilidade nativa. Por último, e apesar de mais divulgado, surge o IE(Internet Explorer) que apenas consegue interpretar SVG (puro ou embutido) com recurso a *plug-ins.*

As desvantagens que se podem apontar relativamente à utilização de *plug-ins*, estão relacionadas com limitações de desempenho. A incapacidade de interpretar algumas características da linguagem SVG e o não reconhecimento da imagem no momento do *streaming* para impressão são mesmo algumas das que são apontadas no site da Adobe.

Suporte para expressões matemáticas

A utilização do *mathml* para representação de expressões matemáticas justifica-se pelos mesmos motivos da utilização do SVG. Como o próprio nome deixa adivinhar, é uma linguagem XML para representação de expressões matemáticas. O plug-in utilizado não tem as mesmas limitações do plug-in gráfico (*SVG viewer*), nomeadamente a limitação da interpretação do código para impressão. Ou seja, quando é dada a ordem de impressão ao IE, as ilhas de código *mathml* embutido na página HTML são correctamente interpretadas e impressas.

V. PRODUTO FINAL

Interface

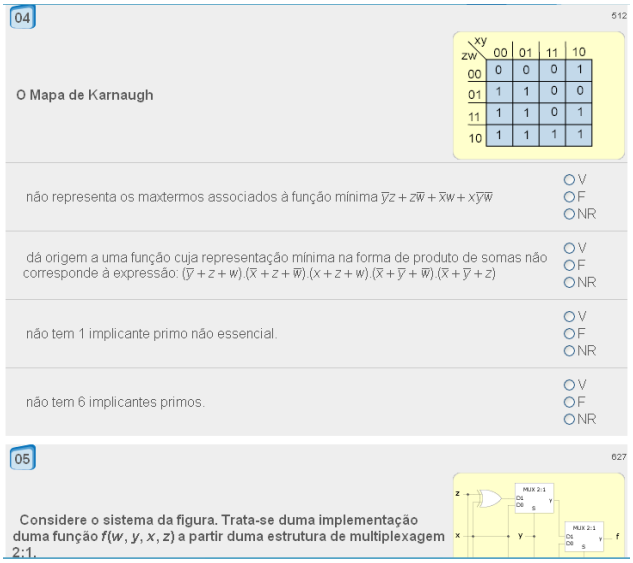

Figura 7 - Prova corrida

Os modelos, depois de validados como estando aptos a serem utilizados, podem ser integrados em provas com dois tipos de estrutura. Um primeiro estruturado por níveis, isto é, o aluno tem acesso a um só exercício (instanciação do modelo) por cada nível, ficando o acesso ao nível seguinte condicionado pela confirmação correcta de todas as respostas. A prova termina quando o aluno, pela segunda vez, no mesmo nível, tenta submeter as resposta sem conseguir acertar a totalidade das questões.

O segundo tipo está mais associado a um contexto de avaliação, em que os níveis anteriores são apresentados todos no mesmo plano, tendo agora o nome de questão – semelhante a uma prova clássica em papel. O aluno neste caso, apenas valida a prova depois de ter respondido a todas as questões. No exemplo da Figura 7 é possível ver duas das *n* questões da prova que obedece a este formato.

Resultados

A possibilidade de armazenar os resultados de todos os alunos, com uma associação directa aos objectivos que foram utilizados na avaliação, é uma mais-valia da plataforma. É um instrumento poderoso se pensarmos em avaliar grandes massas de alunos. Se por um lado se disponibiliza uma ferramenta para ajudar os alunos a consolidar conhecimento, não é menos verdade que o sistema de informação torna-se muito útil quando se pretende avaliar o panorama geral de uma turma, de uma escola, ou até mesmo de um país.

Por exemplo, suponha-se que se pretende avaliar a *Turma x* nos objectivos: Obj1, Obj2 e Obj3. É possível criar uma prova com 3 questões e seleccionar para cada questão um conjunto de modelos associados a um determinado objectivo (ver Figura 8) Quando a prova é gerada pelo sistema, é sorteado um modelo, entre vários associados ao mesmo objectivo, para cada questão.

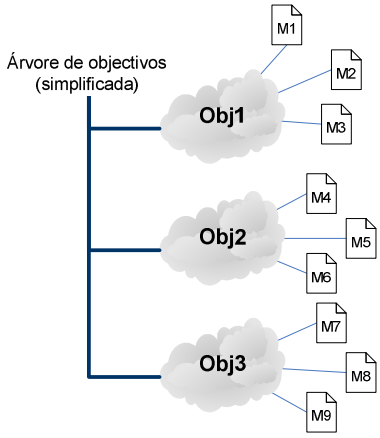

Figura 8 - Modelo simplificado da árvore de objectivos

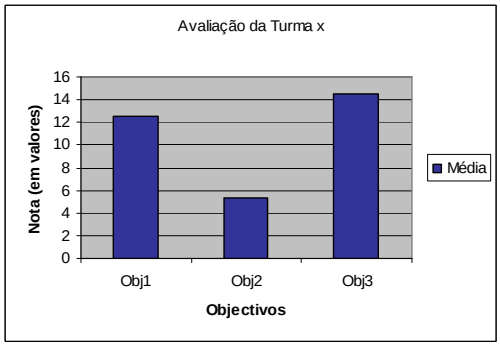

Gráfico 1 - Avaliação por objectivos

Terminado o período de avaliação, os resultados podem ser extraídos, como de uma radiografia se tratasse, e projectar o diagnóstico da amostra sujeita a avaliação. O Gráfico 1 é um exemplo de um possível resultado de avaliação da *Turma x*.

Estes tipos de resultados podem ser úteis como meros indicadores que podem ajudar o docente a traçar uma estratégia para responder ao défice em determinados objectivos.

VI. CONCLUSÕES

O P**mat**E vive em torno do conceito de modelo e dele depende totalmente. A equipa de Sistemas Ditais rapidamente assimilou este conceito e deu início ao desenvolvimento de modelos envolvendo conceitos mais teóricos, circuitos de pequena complexidade ou ilustrados por figuras de pequena dimensão.

A utilização deste tipo de ferramenta revelou poder ser uma forma complementar de estudo, embora ainda seja prematuro tirar conclusões definitivas sobre o impacto no aproveitamento académico.

Foi positiva a participação dos alunos, aproximou-os da disciplina, através do contacto com a equipa do projecto, quer directamente, quer por troca de correio electrónico, ou até mesmo em sessões de esclarecimento criados depois dos momentos de avaliação.

Desenvolvidos os primeiros modelos rapidamente foram encontradas algumas carências na interface que dá visibilidade aos conteúdos, nomeadamente um espaço gráfico reduzido que inviabiliza a criação de modelos mais complexos. Este é um dos objectivos da fase seguinte.

Em tom de conclusão podemos referir que Sistemas Digitais ganha com esta integração e o Projecto Matemática Ensino, passando a ser pluridisciplinar, também sai enriquecido.

REFERÊNCIAS

[1] ACM – Association for Computing Machinery. Data de consulta: Novembro de 2003. Disponível na WWW: <http://www.acm.org/>

[2] IEE – The Institution of Electrical Engineers. Data de consulta: 4 de Outubro de 2005. Disponível na WWW: <http://www.iee.org/Publish/INSPEC/ >

[3] W3C – World Wide Web Consortium. Data de consulta: 4 de Outubro de 2005. Disponível na WWW: <http://www.w3.org/Graphics/SVG/>

[4] Opera Software. Data de consulta: 24 de Outubro de 2005 Disponível na WWW: <http://www.opera.com/>

[5] Mozilla. Data de consulta: 24 de Outubro de 2005. Disponível na WWW: < http://www.mozilla.org/ >