risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido / Recibido: 20/10/2010
Aceitação / Aceptación: 03/12/2010

Metodologia Híbrida de Desenvolvimento Centrado no Utilizador aplicada ao Software Educativo

António Pedro Costa[1], Maria João Loureiro[1], Luís Paulo Reis[2]

apcosta@ua.pt, mjoao@ua.pt, lpreis@fe.up.pt

[1] CIDTFF - Centro de Investigação Didáctica e Tecnologia na Formação de Formadores, DE/UA Departamento de Educação, Universidade de Aveiro, Campus de Santiago, 3800-193, Aveiro, Portugal.

[2] LIACC - Laboratório de Inteligência Artificial e Ciência de Computadores da Universidade do Porto, DEI/FEUP - Departamento de Engenharia Informática, Faculdade de Engenharia da Universidade do Porto, Rua Dr. Roberto Frias s/n, 4200-465, Porto, Portugal.

Resumo: Este artigo descreve a Metodologia1 Híbrida de Desenvolvimento Centrado no Utilizador (MHDCU). Trata-se de um processo de desenvolvimento simples, iterativo e incremental que tem como "alicerces" princípios do Design Centrado no Utilizador (DCU), especificados na International Organization for Standardization - ISO 13407. Na sua base encontra-se a estrutura disciplinada de processos de desenvolvimento, bem como práticas e valores dos métodos ágeis de desenvolvimento de software. O processo é constituído por 4 fases principais: planeamento do guião didáctico, design do storyboard, implementação e manutenção/operação. A prototipagem e a avaliação são realizadas de modo transversal a todo o processo. A MHDCU está ser implementada numa Pequena e Média Empresa (PME) de desenvolvimento de recursos educacionais. O primeiro recurso que teve por base esta metodologia foi o Courseware Sere – O Ser Humano e os Recursos Naturais.

Palavras-chave: Metodologias de Desenvolvimento de *Software Educativo*; Engenharia de *Software*; Métodos Ágeis; Metodologia Híbrida de Desenvolvimento Centrado no Utilizador; Design Centrado no Utilizador.

Abstract: This paper describes the Hybrid User Centered Development Methodology (HUCDM). This methodology is a simple, iterative and incremental development process. The methodology is based on structured disciplined development processes, on principles of User Centered Design (UCD) processes, specified in the International Organization for Standardization - ISO 13407, as well as on practices and values of agile methods for software development. The process consists of 4 main phases: planning of educational guidelines, storyboard design, implementation and maintenance/operation. The prototyping and evaluation are carried out in order to cross the entire process. The HUCDM is being implemented in a Small and Medium Enterprise (SME) of educational

[1] Do grego *methodos* = 'caminho para chegar a um fim' e *logia* = 'estudo de'

resources development. The first resource that was based in this methodology was the Courseware Ser_e - The Human Being and the Natural Resources.

Keywords: Educational Software Development Methodologies; Software Engineering; Agile Methods; Hybrid User Centered Development Methodology; User Centered Design.

1. Introdução

O desenvolvimento de *software* é uma actividade de elevada complexidade, sendo que na maioria dos casos ocorre sem ser devidamente planeado, suportado por decisões de "curto prazo" (Fowler, 2005). Esta abordagem poderá funcionar para pequenos pacotes de *software*, mas à medida que o sistema cresce, cresce também a dificuldade de lhe adicionar novas funcionalidades. Complementarmente, Shneiderman & Plaisant (2005) afirmam que 60% dos projectos de desenvolvimento de *software* falham na definição dos objectivos. Este problema surge nomeadamente porque na maioria dos projectos existe falta de comunicação entre os elementos da equipa e, entre elementos da equipa e os utilizadores finais. Portanto, a escolha do método adequado para o desenvolvimento de um *software* terá vantagens em termos de qualidade, económicas e competitivas, mas caso seja seleccionada um método menos adequado, o mais provável será o projecto ultrapassar os limites temporais, existindo falhas e consequentemente problemas económicos (Toth, 2005).

Neste artigo é efectuado um breve enquadramento teórico sobre a evolução das metodologias de desenvolvimento de *software,* referindo alguns exemplos de métodos utilizados no desenvolvimento de pacotes de *software* educativo. Posteriormente, é efectuada uma breve descrição ao DCU, baseada na exposição dos pressupostos que o constituem e a sua importância no desenvolvimento de *software* educativo. Seguidamente, é apresentado o *Courseware* Ser_e - O Ser Humano e os Recursos Naturais e a descrição da MHDCU, caracterizando as fases, procedimentos e técnicas utilizadas. Para finalizar, é realizada uma breve reflexão em torno da MHDCU e respectivas conclusões.

2. Metodologias de Desenvolvimento de *Software* Educativo

Os primeiros métodos (designados na literatura como disciplinados, tradicionais ou clássicos) derivaram do processo mais comum do desenvolvimento de *software*. O método em cascata de água ou ciclo de vida de desenvolvimento de *software*, surgiu nos anos 70 e é-lhe atribuído o facto de servir de base teórica para outros métodos, sendo por vezes designado como um método genérico para o desenvolvimento de *software* (Sommerville, 2007). Porém, segundo Larman e Basili (2003), o método iterativo e o método incremental já remontam os anos 50, existindo exemplos concretos de projectos nos anos 70. Nos anos 80 surgiram, entre outros, os métodos em espiral e de prototipagem, e nos anos 90 os métodos ágeis, exemplos reais de integração das abordagens iterativas e incrementais (Abbas, 2006 ; Boehm, 2002 ; Boehm & Turner, 2003,s.d. ; Larman & Basili, 2003 ; Miguel, 2003 ; Paelke & Nebe, 2008 ; Sommerville, 2007). Os métodos ágeis, como o *Extreme Programming* (XP), o *Scrum e o Crystal Clear,* através do envolvimento dos utilizadores no processo de

desenvolvimento, procuram fornecer e dar prioridade aos novos requisitos do *software* e avaliar as iterações do mesmo. Dão enfoque ao papel das pessoas, sendo que as competências da equipa de desenvolvimento devem ser reconhecidas e exploradas. Os elementos da equipa são livres para utilizar os seus próprios métodos de trabalho sem serem prescritos processos (Beck, 2000 ; Bergin et al., 2004 ; Keith, 2002 ; Paelke & Nebe, 2008 ; Petersen, 2008 ; Sommerville, 2007).

Especificamente no concerne ao desenvolvimento de *software* educativo, os métodos descritos na literatura têm na sua génese premissas dos métodos supracitados. Exemplo disso, são os 3 métodos utilizados no desenvolvimento dos seguintes pacotes de *software* (ver tabela 2): o Univap Virtual (Bicudo et al., 2007), o Use Case (Castro & Aguiar, 1999) e Softvali (Benitti, Seara, & Schlindwein, 2005).

Tabela 2 – Fases fundamentais no desenvolvimento de *software*

Fases de desenvolvimento (Sommerville, 2007)	**Univap Virtual**	**Use Case**	**Softvali**
Especificação	Análise Planeamento	Preparação	Concepção
Concepção e implementação	Pré-produção	Prototipagem (análise, projecto, implementação, teste e avaliação)	Elaboração construção
Validação	Produção Pós-produção		Finalização
Evolução	------	Implantação	Viabilização

Apesar de surgirem com designações diferentes, a análise, o *design* e a implementação são fases que se podem identificar nestes processos e que advêm dos primeiros métodos de desenvolvimento. Na fase referente à análise é efectuado o levantamento dos requisitos do *software*, são definidos os objectivos educacionais e o público-alvo a que se destina o *software*. No processo do Univap Virtual, nesta fase é ainda efectuado o levantamento de informação científica sobre a temática do recurso e, no Use Case é realizado um estudo da área educacional do recurso a ser desenvolvido e consequentemente o levantamento dos requisitos necessários, para que este corresponda às necessidades dos utilizadores.

3. O Papel do Design Centrado no Utilizador

O DCU serve para descrever os processos de um projecto em que os utilizadores finais têm influência na forma como este é conduzido. Alguns métodos de DCU sondam os utilizadores sobre as necessidades que estes possuem em determinada área educacional, envolvendo-os em partes específicas do processo de desenvolvimento. Por outro lado, existem métodos em que os utilizadores têm uma maior presença, integrando a equipa, isto é, são envolvidos como elementos durante todo o processo (Abras, Maloney-Krichmar, & Preece, 2004).

O DCU é descrito na ISO 13407 (1999) - *Human Centered Design Process for Interactive Systems* e na ISO/TR 18529 (2000) – *Ergonomics of Human-System Interaction*. Estas duas normas descrevem uma situação ideal onde não existem quaisquer obstáculos à utilização dos pressupostos DCU, exceptuando a possível falta de competências por parte da equipa de desenvolvimento (Svanaes & Gulliksen, 2008).

Autores como Facer & Williamson (2004), entre outros, reforçam que o DCU é uma metodologia que combina, entre outros aspectos, a participação do utilizador e avaliação formativa de protótipos. De acordo com a norma ISO 13407 (1999), os projectos DCU são regidos por quatro princípios: i) constituição de uma equipa multidisciplinar; ii) a interacção entre o utilizador e o sistema; iii) o envolvimento activo dos utilizadores e; iv) a iteração de soluções de projecto.

Os três métodos de desenvolvimento de *software* educativo supracitados alicerçam-se em pressupostos DCU, entre os quais, a constituição de equipas multidisciplinares, organizadas por profissionais da área da educação (investigadores de psicologia e pedagogia), profissionais da área da informática, especificamente da área de engenharia de *software* e programadores, designers com conhecimentos de usabilidade e por fim os professores e os alunos.

Um outro método que incorpora pressupostos DCU, é o *Logical User-Centred Interactive Design* (LUCID), em que Kreitzberg (1996) identifica seis fases: visionamento, descoberta, fundamentação do projecto, detalhe do projecto, construção e lançamento. Tal como a maioria dos métodos que tem por base princípios DCU, LUCID emprega a prototipagem rápida e o teste de usabilidade iterativa.

Com base no que foi descrito nesta secção, concordamos com o relatório "*Quality Framework for UK Government Website Design: usability issues for government websites*", quando defende que o DCU é um complemento para os métodos de desenvolvimento de *software*, não sendo um substituto dos mesmos (e-Envoy, 2003).

4. Metodologia de desenvolvimento do *Courseware* Ser_e: uma abordagem possível para PMEs

O *Courseware* Ser_e – O Ser Humano e os Recursos Naturais é um recurso desenvolvido através de uma parceria entre a Universidade de Aveiro e a Ludomedia – Conteúdos Didácticos e Lúdicos, empresa de desenvolvimento de *software* educativo.

4.1. Apresentação do *Courseware* Ser_e

O *Courseware* Ser_e integra várias tipologias de *software* (simulações, inquérito, pesquisa,...) com actividades didácticas especificadas em guiões de exploração, tanto para o professor, como para os alunos. Como se depreende a partir dos seus propósitos (promover a compreensão do impacte que a actividade humana tem sobre os recursos naturais e sensibilizar de que o futuro da Humanidade passará pela adopção de atitudes e comportamentos mais conscientes e responsáveis, nomeadamente no que respeita às fontes de energia utilizadas, em particular o petróleo e a floresta), visa uma abordagem à relação entre a actividade humana e a exploração dos recursos naturais, bem como das consequências ambientais, sociais e económicas desta exploração (Sá et al., 2010 ; Sá et al., 2009).

O *courseware* foi pensado para a utilização, em sala de aula, por alunos do 1º e 2º Ciclos do Ensino Básico (preferencialmente a partir dos 8 anos), particularmente dos 3º aos 6º anos de escolaridade, com a orientação dos respectivos professores, embora a sua exploração possa ser adaptada a outros níveis de escolaridade, bem como a outros contextos.

Do conjunto de recursos do *Courseware* Ser$_e$ fazem parte: um *software* educativo (versão em CD-ROM e *online*, ver em: http://sere.ludomedia.pt), os Guiões de \Exploração Didáctica para o Professor, os Guiões de Registo para o Aluno/Utilizador e o Manual do Utilizador. No Manual do Utilizador encontram-se informações relacionadas com a navegação nos ecrãs e os ícones utilizados no *software*.

O *software* está dividido em duas fases principais: Fase 1 – Petróleo e Fase 2 – Florestas, não sendo obrigatoriamente sequenciais, isto é, o professor/aluno poderá optar por qual das fases e actividade pretende iniciar a exploração.

Os guiões (figura 1(B)) foram desenvolvidos para servir de base à exploração do *software*. No Guião de Exploração Didáctica - Professor são propostas actividades, estruturadas da seguinte forma: 1) Finalidades da Actividade; 2) Contexto de Exploração; 3) Metodologia de Exploração. Os guiões destinados aos/às alunos(as) são compostos fundamentalmente por folhas de registos.

(A) (B)

Figura 1 – (A) Exemplo de ecrã do *Courseware* Ser$_e$. (B) Guiões de exploração didáctica.

4.2. Metodologia de Desenvolvimento

Quanto à metodologia de desenvolvimento, a equipa tem procurado dar resposta a questões de investigação relacionadas com a implementação de metodologias de desenvolvimento de *software* educativo centradas no utilizador (Costa, Loureiro, & Reis, 2009b). Factores de qualidade, tais como, a usabilidade, o envolvimento dos utilizadores finais nas diversas fases de desenvolvimento e a constituição de equipas multidisciplinares, são alguns dos pressupostos do DCU em que se baseia a metodologia de desenvolvimento do *Courseware* Ser$_e$.

A equipa multidisciplinar foi constituída por elementos com diversas competências ao nível da Didáctica das Ciências (DC), da Tecnologia Educativa (TE), da Gestão de Projectos, do Design Gráfico, da Programação e da Usabilidade.

Tendo em vista reduzir o tempo e custo de desenvolvimento, duas das desvantagens do DCU (Abras, Maloney-Krichmar, & Preece, 2004), a equipa optou por envolver o utilizador final (professores e alunos) apenas nas tarefas de avaliação do recurso. O recurso (incluindo o *storyboard*) foi também submetido a avaliação por parte de peritos exteriores à equipa (Costa, Loureiro, & Reis, 2010a ; Costa et al., 2009a ; Guerra, 2007), o que se considera incontornável, independentemente da metodologia adoptada.

A figura 2 sintetiza o processo de desenvolvimento do *Courseware* Ser$_e$, o qual se descreve de seguida.

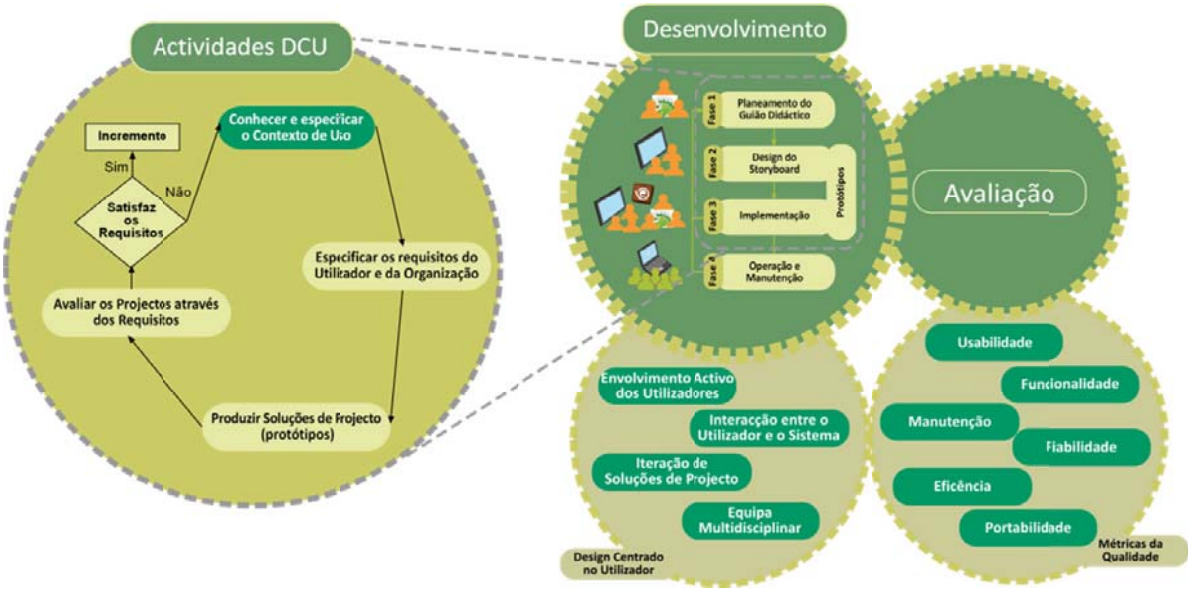

Figura 2 - Metodologia Híbrida de Desenvolvimento Centrado no Utilizador

4.2.1. Fases de Desenvolvimento da MHDCU

Fase 1, Planeamento do guião didáctico: compreendeu a realização de um documento por três peritos em DC e dois em TE com a definição do nível de ensino/público-alvo do recurso, da temática e dos propósitos didácticos, bem como aspectos relacionados com a arquitectura, a navegação e o desenho dos ecrãs do recurso, acima referidos. Esta fase compreendeu ainda o registo de marca e da patente, bem como, entre outros, acordos relativos aos direitos de autoria.

Fase 2, Design do *storyboard*: nesta fase harmonizaram-se as ideias preliminares das actividades didácticas e do conteúdo disciplinar, definidas na fase anterior, com os aspectos de interacção do *software*, particularmente a navegação e interface, com a colaboração de um designer e de um programador da empresa. Como Bassani, Passerino, Pasqualotti & Ritzel (2006) e Carvalho (2003), consideramos que o desenho dos cenários resultantes desta fase foi essencial para se compreender o contexto de utilização do recurso e para representar algumas das situações interactivas do *software*.

Fase 3, Implementação do recurso: esta fase foi dividida em duas subfases que decorreram em simultâneo:

- a parte educacional - requereu a especificação em detalhe de aspectos, para além dos já especificados no *storyboard*, como a animação inicial e os guiões do professor e do aluno.
- a parte técnica - correspondeu ao *design* e programação do *software* e do respectivo manual do utilizador.

Durante esta tarefa, a equipa multidisciplinar testou e ajustou o conteúdo dos guiões à exploração que se pretendia dos ecrãs do *software*. Esta tarefa envolveu a colaboração permanente de todos os elementos, realizada quer presencialmente quer *online*.

Protótipos: os protótipos foram desenvolvidos colaborativamente, entre todos os membros da equipa. Entre outros, a equipa identificou aspectos na interface que tiveram implicações na arquitectura do *software*, que, em alguns casos, levou a alterações nos guiões educacionais do recurso. A prototipagem do *software* foi também usada, no processo de desenvolvimento, de forma a explorar algumas soluções de *software* em particular.

Durante o desenvolvimento do recurso, a equipa recorreu a três tipos de protótipos, como se pode visualizar na figura 3:

- (A) protótipos em papel (*early paper prototypes*);
- (B) ecrãs chave (*key screens*);
- (C) protótipos programados (*running prototypes*).

(A) (B) (C)

Figura 3 - (A) Protótipo em papel de um cenário da fase 2 e de uma das personagens. (B) Protótipo do ecrã da escolha das personagens e de um ecrã de uma das actividades. (C) Protótipos programados de dois ecrãs: um da fase 1 - petróleo e um da fase 2 – floresta.

Avaliação: pretendendo-se avaliar tanto o recurso como o seu processo de desenvolvimento, esta fase é transversal a todas as fases acima indicadas. A primeira versão do recurso, também foi alvo de avaliação, por parte de: a) Professores - na avaliação feita por professores, o questionário para avaliação técnica e didáctica do *Courseware* Ser_e, foi respondido em *workshops* (sessões práticas com a duração máxima de 120 minutos, em que os professores em grupos de dois a três elementos, exploram duas actividades de uma das fases do *courseware*) por parte de um grupo heterogéneo de potenciais utilizadores do recurso (Costa et al., 2009a ; Guerra, 2007); b) Alunos - relativamente à avaliação efectuada pelos alunos, foi respondido um questionário de avaliação técnica e didáctica, após a utilização do recurso em contexto de sala de aula (em blocos de 90 minutos, os alunos em grupos de três a quatro elementos, exploram as actividades do *courseware*, devidamente planificadas pelo professor), tratando-se de uma avaliação controlada (Costa, Loureiro, & Reis, 2010b).

Fase 4, Operação e Manutenção: esta fase incluiu a correcção de erros, técnicos e educacionais, que não foram detectados nas fases iniciais do ciclo de vida do processo de desenvolvimento do *courseware*. Desta forma, é possível melhorar o *software* e implementar novas funcionalidades através de novos requisitos que são detectados durante este processo (Miguel, 2003 ; Sommerville, 2007). Foram tidos em consideração três tipos de manutenção: correctiva, perfectiva e preventiva.

A MHDCU também teve por base alguns princípios dos métodos ágeis, tais como: a) manter a simplicidade, foi desenvolvido o essencial de forma a responder aos requisitos actuais; b) a equipa (essencialmente os programadores) procurou corrigir e melhorar o código do *software* continuamente; c) a entrega foi incremental, sendo que cada ecrã do *software* era independente dos outros ecrãs. Enquanto soluções de projecto eram testadas/validadas/avaliadas, outras eram desenvolvidas com base nos requisitos.

4.2.2. Procedimentos e Técnicas da MHDCU

Para agilizar o processo de desenvolvimento e partindo do princípio que o trabalho colaborativo decorre simultaneamente em dois estados, presencial e *online*, a MHDCU incorpora o Procedimento de Verificação e Validação (PVV), representado no *workflow* da figura 4. Este procedimento surge integrado numa das actividades DCU, Produção de Soluções de Projecto (protótipos), antecedendo a fase de avaliação destas soluções, com o utilizador final e/ou peritos. De suporte a estas actividades é utilizado como *software* colaborativo (*groupware*) a plataforma *Learning Management System* (LMS) Moodle. Apesar, desta plataforma não estar orientada para a gestão de processos desenvolvimento de *software*, a mesma foi essencial para o agilizar da comunicação entre os elementos da equipa multidisciplinar, para disponibilizar documentos, debater ideias, entre outros.

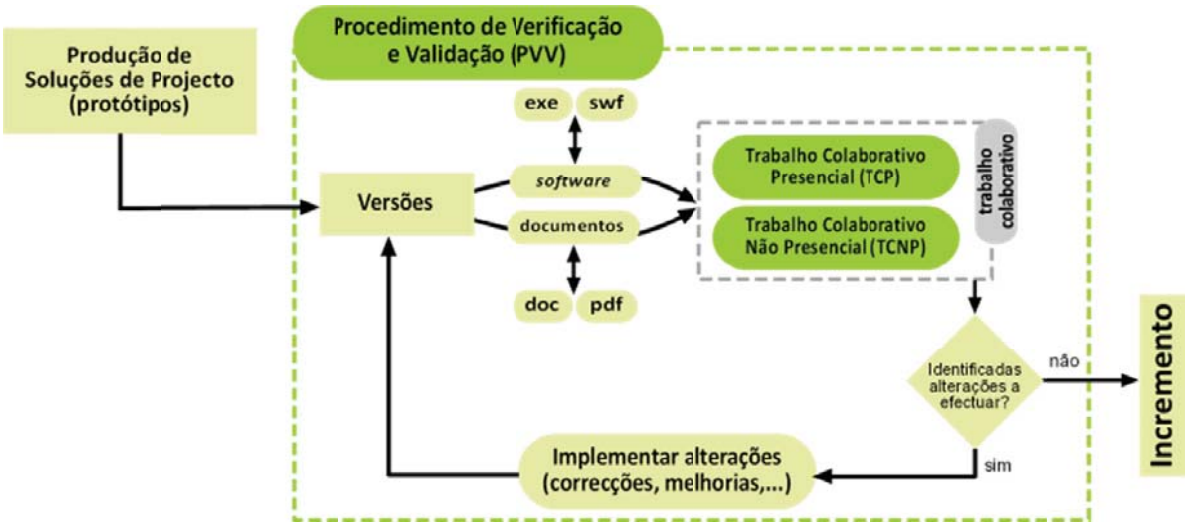

Figura 4 – Procedimento de Verificação e Validação

No PVV, compete aos elementos da equipa multidisciplinar efectuar a verificação e validação das versões do *software,* como das versões dos documentos (guiões do professor e do aluno, manual de utilizador, entre outros). Sendo identificadas alterações a efectuar, é disponibilizada no *moodle* uma nova versão, para verificação e validação. Estas iterações apenas terminam quando não se identificam alterações a efectuar.

No Trabalho Colaborativo Presencial (TCP), comummente, é o gestor de projecto que efectua um primeiro levantamento dos pontos a serem discutidos na reunião presencial. Estes pontos são ordenados por importância e/ou áreas de actuação, sendo enviados previamente para os elementos da equipa multidisciplinar. Para facilitar esta

tarefa é usada uma *mailling list* ou o *fórum* designado como "Notícias e Anúncios". O TCP é registado através de gravação áudio (ver figura 5).

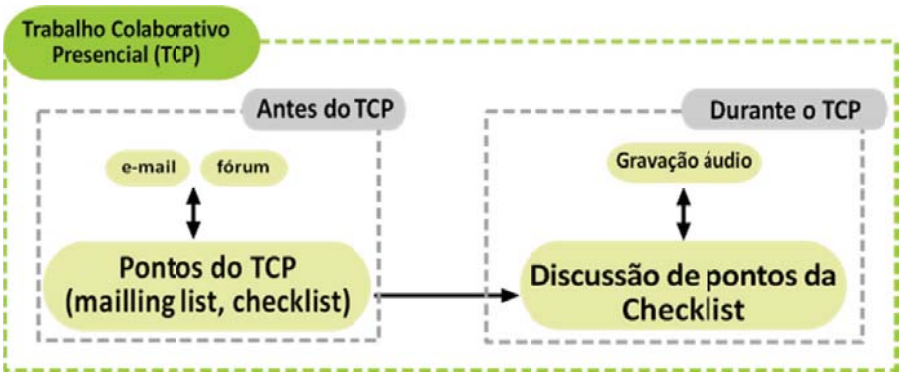

Figura 5 – Trabalho Colaborativo Presencial

Do TCP, ao serem identificadas alterações a efectuar, as mesmas são disponibilizadas na plataforma. Neste contexto, as ferramentas (recursos, módulos de actividades e blocos) utilizadas (ver figura 6) no Trabalho Colaborativo Não Presencial (TCNP), permitem promover e agilizar uma maior interacção entre os elementos da equipa multidisciplinar.

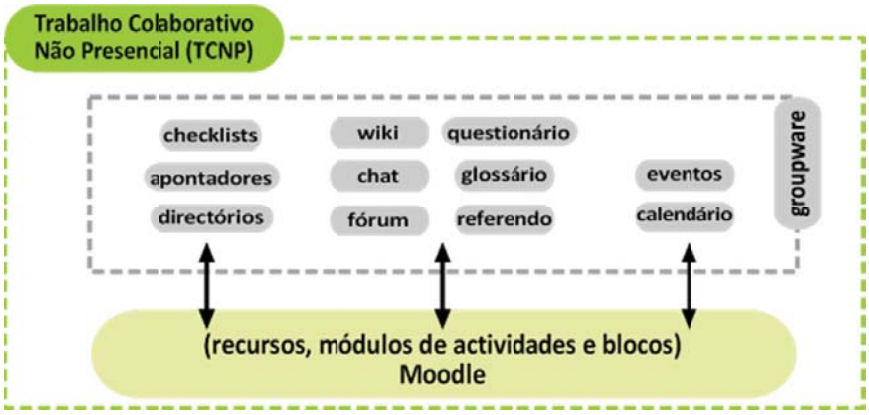

Figura 6 – Trabalho Colaborativo Não Presencial

Seguidamente, passamos a descrever alguns dos módulos de actividades, utilizados no TCNP e concretamente no desenvolvimento do *courseware*:

- referendos: a decisão sobre qual a versão do logótipo que se deveria optar, foi escolhida através de um referendo. Para a marcação de reuniões presenciais, também foi utilizado este módulo de actividades;
- fóruns: todas as versões do *software* e dos documentos (guiões de registo professor e aluno e manual de utilizador, entre outros), foram debatidos através de fóruns;
- glossários: definição de termos científicos e técnicos utilizados, com o intuito de facilitar a comunicação entre os elementos da equipa. O glossário, também serviu para disponibilizar as actas de reunião;

- *chats:* discussão síncrona de determinada questão do projecto, permitiu por vezes, que as tomadas de decisão fossem mais céleres;
- *wikis:* permitiu a construção colaborativa dos textos utilizados no *software*;
- inquéritos por questionário: este módulo foi utilizado durante os workshops de avaliação do recurso, em que os utilizadores no final do *workshop* respondiam a um inquérito por questionário de avaliação. Os resultados ficavam automaticamente disponibilizados para todos os elementos da equipa multidisciplinar.

As ferramentas apresentadas na figura 6, foram disponibilizadas evolutivamente, com o surgimento da necessidade de tornar o trabalho colaborativo mais fácil.

4.3. Reflexões em torno da metodologia explorada

Relativamente à MHDCU, foram detectadas algumas lacunas, que num próximo projecto necessitam de ser colmatadas. Será necessário na fase inicial definir o papel de cada elemento e dar formação na utilização das ferramentas de suporte à gestão de projecto. Além disso e tendo por base o estudo apresentado por Martin Maguire "*Methods to support human-centred design*" (2001), também concluímos que a fase do planeamento deve incorporar duas subfases: 1) contexto de uso e 2) requisitos. Na tabela 3 são identificados alguns métodos que podem ser aplicados em novos projectos.

Tabela 3 – Métodos DCU - Fase 2, Planeamento

Método	Descrição
Identificação e análise de stakeholders	Identificar todos os grupos de utilizadores (utilizadores finais, peritos, responsáveis pela instalação e manutenção) e outras partes interessadas (aqueles que podem ter influência ou são afectados pelo *software*) incluindo equipas de apoio, equipas comerciais e de marketing e diferentes clientes que poderão adquirir o recurso. Este levantamento pode ser complementado com uma análise de mercado. Nesta fase são atribuídos papéis e responsabilidades. Abordagem ao método: reunião realizada com o gestor de projecto e representantes dos utilizadores para discussão de papéis com maior detalhe.
Análise do contexto de uso(Bevan & Macleod, 1994 ; Kirakowski & Cierlik, 1999 ; Maguire, 1998 ; Thomas & Bevan, 1995)	Caracterização do grupo de utilizadores (competências e experiência, conhecimento da tarefa, formação, qualificações, competências linguísticas, capacidades físicas e cognitivas e atitudes e motivações), das tarefas (lista de tarefas, objectivos, output, passos, frequência, importância, duração e dependências) e do ambiente técnico, físico e organizacional (*hardware*, *software*, rede de internet, características das salas, objectivos organizacionais, políticas de utilização das TIC, apoio técnico). Abordagem ao método: Reunião com representantes de cada grupo de utilizadores e representantes da equipa de projecto.
Análise de tarefas e mapeamento de tarefas/funções	Estudo do que utilizador é obrigado a fazer em termos de acções e/ou processos cognitivos para realizar determinada tarefa. Clarificar quais as funcionalidades que são necessárias, de forma a excluir as menos importantes. Abordagem ao método: planear reuniões e eventualmente sessões de observação.
Focus groups	Grupo de discussão de potenciais utilizadores sobre o que pretendem que seja concebido no *software*. Para evidenciar aspectos dos utilizadores quando os mesmos não podem participar no projecto. Abordagem ao método: presencialmente ou através da plataforma LMS, os utilizadores discutem os requisitos.

5. Conclusões

Embora, a selecção do método dependa do ambiente em que se insere o projecto e de um conjunto de variáveis que, por vezes, são de difícil definição, os métodos existem para tentar auxiliar no desenvolvimento de *software*, minimizando a incerteza, de modo a permitir a obtenção do resultado esperado da forma mais eficiente possível. Tal como Toth (2005) e Sommerville (2007), consideramos que, a adopção do mesmo método para todos os projectos de desenvolvimento de *software*, dificilmente será uma boa escolha, se tivermos em conta a diversidade de utilizadores, objectivo da utilização do *software* e as alterações constantes da tecnologia.

Com base no trabalho desenvolvido, concordamos com Abbas, Gravell & Wills (2008), que designam o desenvolvimento de *software* como uma actividade imprevisível, sendo necessário um método adaptável para controlar esta imprevisibilidade. Quanto ao desenvolvimento de *software* educativo, os processos iterativos e incrementais associados a procedimentos de prototipagem, incluindo ferramentas de avaliação e monitorização nas diferentes fases, são uma forma eficiente de um processo se adaptar à mudança constante de requisitos e da tecnologia (Costa, Loureiro, & Reis, 2009b). Paralelamente, também revemos e estamos de acordo com Abras, Maloney-Krichmar &

Preece (2004), que duas das desvantagens do DCU, são a dos projectos necessitarem de mais tempo para serem desenvolvidos e tornarem-se mais dispendiosos. Porém e segundo Shneiderman & Plaisant, os métodos DCU permitem que o *software* gere menos problemas durante o desenvolvimento e que se reduza os custos na fase de manutenção (2005, p.118), fase esta, que é considerada a mais despendiosa do ciclo de vida de um *software* (Duim, Andersson, & Sinnema, 2007).

A MHDCU constitui-se como uma solução possível para o desenvolvimento de *software* educativo e foi utilizada para o desenvolvimento do *Courseware* Ser_e, cuja qualidade tem sido reconhecida, nomeadamente num concurso nacional de produtos multimédia, dado ter sido um dos produtos finalistas. Porém, como defendem as normas ISO, a melhoria de um processo de desenvolvimento de *software* deve ser contínua. Estando previsto o desenvolvimento da 3ª fase do *Courseware* Ser_e, relacionado com as energias alternativas, perspectivamos, que a introdução de novos métodos DCU, na fase inicial do desenvolvimento, permitam auxiliar a tomada de decisões e contribuam para a melhoria do processo.

Referências

Abbas, N. (2006). *Choosing the Appropriate Strategy for a Particular Software Development Project*. Unpublished MSc in Software Engineering, University of Southampton Shouthampton.

Abbas, N., Gravell, A. M., & Wills, G. B. (2008). Historical Roots of Agile Methods: Where did "Agile Thinking" Come from? . In A. p. a. e. p. i. S. Engineering (Ed.). Limerick, Irlanda.

Abras, C., Maloney-Krichmar, D., & Preece, J. (2004). User-Centered Design. In S. Publications (Ed.), *Encyclopedia of Human-Computer Interaction*: Thousand Oaks: Sage Publications.

Bassani, P. S., Passerino, L. M., Pasqualotti, P. R., & Ritzel, M. I. (2006). Em busca de uma proposta metodológica para o desenvolvimento de software educativo colaborativo. *Novas Tecnologias na Educação, 4(1)*, 1-10.

Beck, K. (2000). *Extreme Programming Explained: Embrace Change*: Addison-Wesley.

Benitti, F. B. V., Seara, E. F. R., & Schlindwein, L. M. (2005). Processo de Desenvolvimento de Software Educacional: proposta e experimentação *CINTED-UFRGS. Novas Tecnologias na Educação*.

Bergin, J., Caristi, J., Dubinsky, Y., Hazzan, O., & Williams, L. (2004). Teaching Software Development Methods:The Case of Extreme Programming. In ACM (Ed.), *SIGCSE '04*. Norfolk, Virginia.

Bevan, N., & Macleod, M. (1994). Usability measurement in context. *13*, 132-145.

Bicudo, S. F., Nogueira, T., Oliveira, G. S., Machuca, V. F., Romero, J. P. F., Montenegro, E., et al. (2007). Projecto e Desenvolvimento de Jogos Educativos em 3 Dimensões: a experiência da Univap Virtual.

Boehm, B. (2002). Get Ready for Agile Methods, with Care. *IEEE Computer*, 64-69.

Boehm, B., & Turner, R. (2003). *Observations on Balancing Discipline and Agility*: Addison Wesley.

Boehm, B., & Turner, R. (s.d.). Rebalancing Your Organization's Agility and Discipline.

Carvalho, C. V. (2003). *Conceitos básicos para o desenvolvimento de cursos multimédia - Manual do Formador (1.ª Edição)*. Porto: Sociedade Portuguesa de Inovação.

Castro, G. C. M. d., & Aguiar, T. C. d. (1999). Engenharia de Software no Desenvolvimento de Software Educacional Hipermídia, *XXV Conferencia Latinoamericana de Informática*. Asunción-Paraguay.

Costa, A. P., Loureiro, M. J., & Reis, L. P. (2009b). Development Methodologies for Educational Software: the practical case of Courseware Sere. In E. a. D. International Association of Technology (Ed.), *International Conference on Education and New Learning Technologies (EDULEARN09)* (pp. 5816-5825). Barcelona, Espanha: International Association of Technology, Education and Development (IATED).

Costa, A. P., Loureiro, M. J., & Reis, L. P. (2010a). Metodologia Híbrida de Desenvolvimento Centrado no Utilizador: o caso prático do Courseware Sere. In A. I. d. S. e. T. d. Informação (Ed.), *5ª Conferência Ibérica de Sistemas e Tecnologias de Informação (CISTI2010)* (pp. 192-197). Santiago de Compostela, Espanha.

Costa, A. P., Loureiro, M. J., & Reis, L. P. (2010b). Courseware Sere: Avaliação Técnica e Didáctica efectuada por Alunos. In A. I. d. S. e. T. d. Informação (Ed.), *5ª Conferência Ibérica de Sistemas e Tecnologias de Informação (CISTI2010)* (pp. 198-203). Santiago de Compostela, Espanha.

Costa, A. P., Loureiro, M. J., Reis, L. P., Guerra, C., Sá, P., & Vieira, R. (2009a). Courseware Sere: Technical and Didactic Evaluation. In A. e. a. e. R. In Méndez-Vilas, Reflections and Innovations in Integrating ICT in Education (Ed.), *V Conferência Internacional de Multimédia e TIC na Educação (m-ICTE2009)* (Vol. 1, pp. 502-506). Lisboa.

Duim, L. v. d., Andersson, J., & Sinnema, M. (2007). Good Practices for Educational Software Engineering Projects, *Proceedings of the 29th international conference on Software Engineering*: IEEE Computer Society.

e-Envoy, O. o. t. (2003). Quality Framework for UK Government Website Design: Usability issues for government websites

Facer, K., & Williamson, B. (2004). *Designing educational technologies with users - A handbook from Futurelab*, em http://www.futurelab.org.uk/resources/documents/handbooks/designing_with_users.pdf

Fowler, M. (2005). *The New Methodology* [online], em http://www.martinfowler.com/articles/newMethodology.html

Guerra, C. (2007). *Avaliação do Storyboard e da Metodologia de Desenvolvimento do Courseware Sere*. Universidade de Aveiro, Aveiro.

ISO13407. (1999). Human-centred design processes for interactive systems. Geneva: International Standards Organisation.

ISO/TR18529. (2000). Ergonomics of Human-System Interaction. Geneva: International Standards Organisation.

Keith, E. R. (2002). Agile Software Development Processes - A Different Aprroach to Software Design.

Kirakowski, J., & Cierlik, B. (1999). *Context of Use: Introductory Notes*. Acedido a 10 de Março, 2010, em http://hfrg.ucc.ie/baseline/filearchive.html#cou

Kreitzberg, C. (1996). Managing for usability, in Aber, Antone F. (Editor), Multimedia: A Management Perspective, Wadsworth, Belmont, CA, 65-88.

Larman, C., & Basili, V. R. (2003). Iterative and Incremental Development: A Brief History. *Computer, 36*(6), 47-56.

Maguire, M. (2001). Methods to support human-centred design. *Internacional Journal of Human-Computer Studies, 55.4*, 587-634.

Maguire, M. C. (1998). Respect User-Centred Requirements Handbook, *Telematics Applications Project TE 2010: Requirements Engineering and Specification in Telematics* (3.3 ed.): HUSAT Research Institute.

Miguel, A. (2003). *Gestão de Projectos de Software*: FCA - Editora de Informática.

Paelke, V., & Nebe, K. (2008). Integrating Agile Methods for Mixed Reality Design Space Exploration, *Proceedings of the 7th ACM Conference on Designing Interactive Systems* (pp. 240-249). Cape Town, South Africa: ACM.

Petersen, R. R. (2008). *ASAP: Agile Planning in Future Creative Room.*, University of Southern Denmark, Odense.

Sá, P., Guerra, C., Martins, I. P., Loureiro, M. J., Vieira, R., Costa, A. P., Reis, L. P. (2010, Fevereiro). Desenvolvimento de Recursos Didácticos Informatizados no Âmbito da Educação para o Desenvolvimento Sustentável. O Exemplo do Courseware Sere. *Revista Eureka sobre Enseñanza y Divulgación de las Ciencias, 7,* pp. 330-345.

Sá, P., Martins, I. P., Guerra, C., Loureiro, M. J., Vieira, R., Costa, A. P., Reis, L. P. (2009). Courseware Sere: Metodologia e finalidades de exploração, *XIII Encontro Nacional de Educação em Ciências - Educação e Formação: Ciência, Cultura e Cidadania (ENEC2009)* (pp. 899-908). Castelo Branco.

Shneiderman, B., & Plaisant, C. (2005). *Designing the User Interface- Strategies for Effective Human-Computer Interaction* (Fourth ed.): Pearson Education.

Sommerville. (2007). *Software Engineering* (Eighth Edition ed.): Addison Wesley.

Svanaes, D., & Gulliksen, J. (2008). *Understanding the Context of Design - Towards Tactical User Centered Design*. Paper presented at the Nordic Conference on Human-Computer Interaction (NordiCHI2008), Lund, Sweden.

Thomas, C., & Bevan, N. (1995). *Usability Context Analysis: A Practical Guide* (4.04 ed.). Teddington, Middlesex, TW11 0LW, UK: National Physical Laboratory.

Toth, K. (2005). Which is the Right Software Process for Your Problem? .