Requisitos não funcionais em jogos digitais com Scrum

Leandro Ramos da Silva

Faculdade de Tecnologia de São Caetano do Sul – São Caetano do Sul – SP - Brasil

Leandro.rsilva@fatec.sp.gov.br

Resumo

A criação de um jogo digital envolve tipicamente diferentes perfis de profissionais por conta das diversas abordagens com as quais podemos trabalhar a concepção de um jogo, como lógica, implementação, mecânica e imersão. Assim, se faz necessária a criação de métodos adequados para lidar com essa complexidade e garantir que o software a ser criado atenda a todos os aspectos envolvidos na criação. O foco deste estudo é apresentar o gerenciamento de requisitos não funcionais de um jogo digital em um ambiente empírico de desenvolvimento iterativo e incremental baseado em Scrum.

1. Introdução

Tipicamente, projetos de desenvolvimento de jogos digitais são complexos [BLOW 2004], razão pela qual se faz necessária a utilização de um método capaz de organizar as ideias e técnicas aplicadas na criação de um jogo digital [REIS 2002].

Diversos autores tratam os conceitos de "requisitos não funcionais" por vezes através do termo genérico "qualidade", que apesar de impreciso, ressalta a importância desse requisito no âmbito geral do desenvolvimento de software [PRESSMAN 2006; SOMMERVILE 2003]. Este estudo apresenta como esses requisitos não funcionais permeiam o desenvolvimento de um jogo digital.

2. Trabalhos Relacionados

Os problemas no desenvolvimento de games têm sido uma preocupação frequente para os funcionários e empresários do setor. Saber por que os projetos de desenvolvimento de um game são tão complexos e problemáticos [BLOW 2004] e, principalmente, encontrar formas de solucionar os diversos problemas que estão relacionados com essas questões, vêm sendo cada vez mais investigado por educadores e pesquisadores [NOVAK 2010].

Neste trabalho, a partir do estudo de casos realizados em projetos de desenvolvimento de jogos, são discutidos à luz da teoria atual os conceitos de requisitos não funcionais em um ambiente empírico de desenvolvimento amparado pelo Scrum, apresentando os métodos e ferramentas envolvidas neste processo e finalizando com a sugestão de um método para gestão dos requisitos não funcionais neste contexto [MYLLYAHO 2004].

2.1. Scrum e Jogos

Scrum é um framework de processo criado na década de 1990 que viu sua popularização no século XXI, na aplicação em produtos que demandam um processo de adaptação contínua e rápidas respostas às mudanças. Jeff Sutherland aplicou a primeira concepção de Scrum no início na década de 1990, e essa metodologia foi posteriormente aprimorada por Ken Schwaber.

Não se trata de um processo rígido, pois ele apresenta diretrizes, ou key features¸ que indicam características importantes a considerarmos ao utilizarmos os conceitos de produto funcionando¸ colaboração do cliente, resposta à mudança e iterações, propondo uma forma de trabalho flexível e tratamento às mudanças frequentes de requisitos, adaptações de cronograma ou orçamento, mudanças de equipe, entre outras características de um projeto de alta complexidade [KOSCIANSKI e SOARES 2007]. Uma das principais estratégias do Scrum para atacar as questões exemplificadas é a utilização de um controle descentralizado, no qual é possível lidar de forma mais eficiente, com contextos pouco previsíveis e de alta complexidade [ARAUJO 2006].

Os princípios do Scrum para subsidiar o conceito de projeto adaptável se baseiam na entrega constante de softwares, com times pequenos e trabalhos divididos, e com testes e documentações realizados durante todo o desenvolvimento [PRESSMAN 2006].

No Scrum, quando se é delimitado um período curto de tempo para cada sprint, as equipes devem realizar suas atividades em paralelo para conseguirem visualizar os resultados do projeto como um todo. A utilização de metodologias ágeis proporciona um alto nível de conhecimento do projeto, além de considerar esse conceito uma de suas prioridades.

Esses princípios são práticas para o processo de desenvolvimento do software que ficam organizadas em um processo chamado sprint. Sprints são iterações realizadas sucessivamente (uma após a outra). Quando um sprint é iniciado, nenhuma mudança pode ocorrer a fim de manter o ambiente de trabalho do time estável. A duração de cada sprint pode ser de duas a quatro semanas e, no fim desse período, entrega-se funcionalidades capazes de agregar valor ao negócio do cliente [PRESSMAN 2006].

As regras do sprint baseiam-se no time assumir e realizar as tarefas necessárias de forma que seus resultados sejam reportados em reuniões diárias – com duração máxima de 15 minutos – com o objetivo de entregar determinadas funcionalidades (backlog do sprint). Ao fim do sprint é realizada uma reunião de avaliação das funcionalidades apresentadas.

O Scrum possui alguns artefatos principais: backlog do produto, backlog do sprint e o gráfico de Burndown [PETRILLO 2008]. Destes, merece especial relevância para este estudo backlog do produto, uma lista de funcionalidades organizada por itens com prioridade e estimativa, tendo a prioridade de cada item definida de acordo com a necessidade do cliente, de forma que ela será alterada apenas pelo Product Owner. Enquanto que o backlog do sprint tem como objetivo transformar os itens do backlog do produto em produto pronto para ser entregue ao cliente.

Pela relevância que vem ganhando no cenário de projetos que demandam principalmente capacidade de rápida resposta à mudança, o Scrum foi adotado pela comunidade de desenvolvimento de jogos [KEITH 2010; LAUBISCH e CLUA 2010; PIMENTA e PETRILLO 2010].

2.2. Requisitos não funcionais em jogos

O entendimento de requisitos em software é tido como uma tarefa essencial, independente do momento em que ocorre ou das técnicas envolvidas, visto que dificuldades com requisitos de software implicam normalmente na dificuldade de participação dos clientes corretos [SHORE e WARDEN 2008].

Enquanto a escrita de funcionalidades em jogos é uma tarefa bem discutida em literaturas com diversos exemplos em Game Design, como os apresentados por Mike Cohn [COHN 2006], descrever requisitos não funcionais é tipicamente um desafio na indústria de software [COHN 2004], sendo vislumbrados normalmente como um atributo ou característica do sistema [PRESSMAN 2006; SOMMERVILE 2003]. Exemplos comuns de requisitos não funcionais são confiabilidade, disponibilidade, portabilidade, escalabilidade, usabilidade, manutenibilidade, segurança e performance, mas podem variar dentro deste contexto de acordo com a natureza do projeto.

Autores como Clinton Keith [2010] ressaltam a importância dos "requisitos não funcionais", colocando-os como elemento fundamental no cenário de jogos. Para Keith, tais requisitos são itens de uma lista que comporão o "jogo", o backlog do produto citado anteriormente – popularmente conhecido como Product Backlog.

Por outro lado, não é incomum identificarmos na indústria de desenvolvimento de jogos problemas que podem ser diretamente relacionados à gestão de requisitos não funcionais de software. Petrillo [2008] realizou uma série de estudos e levantamentos em post-mortem de jogos com a finalidade de identificar problemas de produção. Desses problemas, destacamos os que relacionamos diretamente aos requisitos não funcionais: problemas tecnológicos e problemas com ferramentas.

Os problemas tecnológicos identificados variam de falhas em plataformas, hardwares, entre outros. Dessa forma, cerca de 60% das análises dos relatórios de post-mortem apresentaram problemas tecnológicos [PETRILLO 2008].

Associados aos problemas tecnológicos, podemos identificar problemas com ferramentas. Embora sejam elementos fundamentais para o processo de construção de um jogo, foram identificados nos projetos analisados 35% de erros relacionados a elaboração de partes do game, controles e geração de versões entre outros obstáculos de ferramentas [PETRILLO 2008].

3. Discussão

A utilização de metodologias ágeis preza por iterações constantes e compostas de curtos intervalos de tempo – normalmente de duas a quatro semanas – durante as quais o jogo é desenvolvido [KEITH 2010]. Espera-se com isso tratar de forma preventiva os problemas de projeto, problemas tecnológicos, defeitos, ferramentas e orçamento extrapolado. Por meio das metodologias ágeis, as equipes de desenvolvimento trabalham com entregas de curta duração. Essa característica estimula um desenvolvimento mais iterativo e incremental.

Então, a utilização de metodologia ágil não evita planejamento, mas ele é estruturado de forma que mudanças e revisões do escopo sejam constantes. Assim, tipicamente em times que trabalham com Scrum, uma característica fundamental para auxiliar na questão de planejamento é a comunicação e modelagem da equipe em suas reuniões. No Scrum existem reuniões diárias de retrospectiva, planejamento e revisão do sprint que tornam o projeto mais

integrado e a comunicação entre as equipes inevitável. O game é revisado ao fim de cada iteração, tanto em termos de produto através da reunião de revisão, como através da reunião de retrospectiva, momento em que o time faz uma autorreflexão. Esse resultado influencia nos objetivos das futuras iterações, organizando os erros, gastos excessivos e evitando os problemas relacionados no início desta seção. Assim, é possível afirmar que os métodos ágeis proporcionam melhoria na interação entre os envolvidos, promovendo feedback constante e transparência durante o desenvolvimento do software [KEITH 2010; TELES 2004; VASCO 2006].

No entanto, apesar da intensa comunicação no Scrum, Petrillo [2008] ressalta que problemas de comunicação entre os integrantes da equipe ocorrem de forma considerável. Conforme conhecimento geral, as equipes técnicas e de arte acabam não se integrando corretamente. De acordo com os post-mortem analisados, 35% dos projetos passaram por essa situação. E, ainda segundo o autor, 40% dos post-mortem analisados apresentaram falta de documentação, apesar de muitos projetos relatarem êxito quanto as suas documentações.

No levantamento feito por Petrillo [2008], o principal incômodo quanto à falta de documentação estava relacionado à falta de conhecimento do projeto: *"Se tivéssemos desenvolvido um documento de projeto, nós só iríamos utilizá-lo com a compreensão de que poderia ser modificado a qualquer momento. Penso que existem bons motivos para se ter um guia central organizado com todas as ideias. Ter a possibilidade de sentar e olhar uma visão do projeto como um todo tem um grande valor"* [PETRILLO 2008]

Deste cenário, conclui-se a necessidade de documentar e comunicar determinados aspectos inerentes à adaptabilidade no desenvolvimento de jogos, principalmente o que tange ao gerenciamento de aspectos tecnológicos e ferramentais, um subgrupo dos requisitos não funcionais.

3.1. Requisitos não funcionais de jogos com Scrum

Segundo [LAUBISCH e CLUA 2010] informações técnicas e de usabilidade podem ser facilmente gerenciadas através de histórias. Dentre os métodos propostos fundamentados em extrair requisitos com base na experiência do usuário, ou usabilidade, está a priorização com base em User Stories Mapping, cunhado inicialmente por Jhon Patton. O método consiste basicamente em mapear um conjunto de requisitos de acordo com potenciais usuários, stakeholders ou, no contexto atual de desenvolvimento de jogos, os jogadores.

As histórias possuem em comum o fato de estarem focadas na funcionalidade (requisitos funcionais) do jogo como jogabilidade, gameplay ou mecânica. Essa estratégia tem por objetivo manter o foco do time Scrum no enredo do jogo e em histórias conhecidas como INVEST: independentes, negociáveis, com valor, estimáveis, pequenas e testáveis. Assim, aspectos técnicos são deixados para as cerimônias de Daily Scrum e Sprint Retrospective, sendo descritos no backlog da Sprint através das tasks.

Em termos de requisitos funcionais, no método proposto foi utilizada o Kunagi (http://kunagi.org) – ferramenta web open source disponível a todos os membros da equipe – implantada em um servidor dedicado aos membros integrantes do projeto. Na ferramenta são mapeadas todas as User Stories, ou requisitos funcionais, no formato apresentado na figura a seguir:

Figura 2: Product Backlog em jogo digital.

Já em relação ao requisitos não funcionais, foi construída uma tabela mapeando todos os requisitos não funcionais em critérios de importância para eventuais jogadores, conforme tabela 1 a seguir. Os perfis dos potenciais jogadores estão evidenciados na tabela 2.

Tabela 1: relevância dos RNFs para o jogo proposto.

Requisito não funcional	Mauricio	Ricardo	Antônio	Bruno	Milena	Fernanda	Joana	Pedro	Carlos	Carla
Tamanho de 40 Mb	•	•	•	•	•	•	•	•	•	•
Tela de loading com no máximo 10s	x - Insatisfeito	•	•	•	•	•	•	•	•	•
Rodar em versão IOS e Android antigas	x - Satisfeito	•	x - Satisfeito	•	x - Satisfeito	•	•	x - Satisfeito	•	x - Satisfeito
Rodar web	x - Satisfeito	x - Satisfeito	•	•	x - Satisfeito	•	•	•	•	•
Sem LAG em equipamento com requisitos baixo	x - Satisfeito	x - Satisfeito	•	•	•	•	•	•	x - Satisfeito	•
Jogo com alto grau de dificuldade	x - Insatisfeito	x - Satisfeito	x - Insatisfeito	x - Insatisfeito	x - Insatisfeito	x - Insatisfeito	x - Insatisfeito	x - Insatisfeito	•	x - Insatisfeito
Jogo gratuito	x - Satisfeito	x - Satisfeito	x - Satisfeito	•	•	•	•	•	x - Satisfeito	x - Satisfeito
Duração da tela com no máximo 5 minutos	•	x - Insatisfeito	x - Insatisfeito	•	•	•	•	•	•	•
Jogo com design retrô	x - Insatisfeito	x - Satisfeito	x - Satisfeito	•	•	•	•	•	x - Satisfeito	x - Satisfeito
Jogos de plataforma 2D	x - Insatisfeito	x - Satisfeito	x - Insatisfeito	x - Satisfeito	•	•	•	•	•	x - Satisfeito

Tabela 2: perfis baseados em personas.

Nome	Idade	Interesses	Objetivos	Frustrações
Mauricio	16	Jogo de futebol.	Vencer os amigos	Perder.
Ricardo	17	MMORPG	Se destacar no clã.	Não conseguir itens valiosos.
Antônio	22	FPS	Desestressar.	Lag.
Bruno	14	Jogos de luta	Vencer os amigos	Dificuldade em aprender novas técnicas.
Milena	13	Gosta de jogar xadrez online.	Adquirir conhecimentos	Falta de conteúdo educativo em jogos.
Fernanda	12	Candy Crush	Passar o tempo.	Pagar pra passar.
Joana	8	sta de puzzles tipo frozen para celul	Se divertir.	Não tem.
Pedro	7	Gosta muito do minecraft.	Montar seu acampamento.	Perder itens.
Carlos	19	FPS	Fugir da realidade	Lag.
Carla	18	Candy Crush	Passar o tempo.	Pagar pra passar.

Em seguida, os requisitos não funcionais são mapeados em cada uma das histórias (tabela 3) e se tornam atributos de qualidade para o jogo digital. Esses são mapeados para uma seção específica do projeto onde estão os requisitos não funcionais, denominada Quality Backlog (figura 3).

Tabela 3: mapeamento de requisitos funcionais e não funcionais.

R.F \ R.N.F	Tamanho de 40 Mb	Sem LAG em equipamento com requisitos baixo	Jogo com alto grau de dificuldade	Jogo com design retrô	Jogos de plataforma 2D
Controlar nave		Sim	Sim	Sim	Sim
atirar laser		Sim	Sim	Sim	Sim
desviar obstáculos		Sim	Sim	Sim	Sim
ataca pisando		Sim	Sim	Sim	Sim
sobe plataforma		Sim	Sim	Sim	Sim
morre depois de perder 3 corações		Sim	Sim		
Responder perguntas corretamente		Sim	Sim		
Errou perde vida		Sim			
NPC e jogador tem três vidas		Sim	Sim		
objetivo matar NPC		Sim	Sim		
atirar com metralhadora		Sim	Sim	Sim	Sim
pegar itens		Sim	Sim		
destruir pontes		Sim	Sim	Sim	Sim
Destruir inimigos		Sim	Sim	Sim	Sim
trilha sonora diferente para cada fase	Sim			Sim	

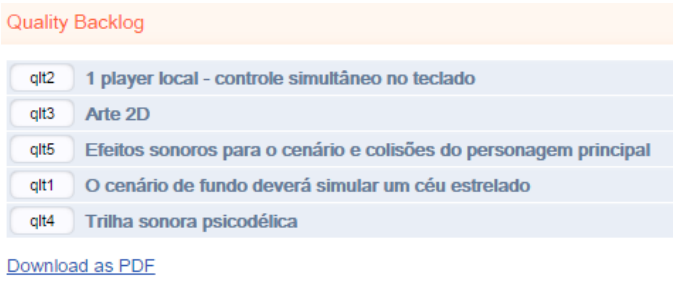

Figura 3: Quality Backlog em jogo digital.

Assim, cada requisito funcional pode ser mapeado para as respectivas histórias, mantidas assim em um repositório central a partir do backlog do produto.

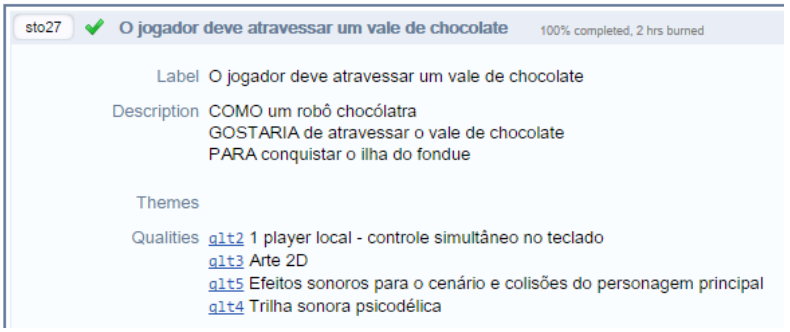

Figura 4: mapeamento RF vs RNF.

4. Conclusão

O presente estudo se propôs a analisar os requisitos não funcionais no contexto do desenvolvimento iterativo de jogos com o uso de Scrum.

Assim, foram evidenciadas as principais características para classificação de tais requisitos, que por vezes não recebem a devida atenção no processo de desenvolvimento, apesar da notável relevância para o resultado geral de um jogo digital.

No contexto do desenvolvimento de jogos, ressaltou-se, de maneira não exaustiva, eventuais impactos na falta de preocupação com essa categoria de requisitos. De tal forma, apesar de não esgotar o assunto em pauta, o presente estudo direciona a atenção para um aspecto relevante no desenvolvimento de jogos, capaz de interagir diretamente com outras áreas do game design, como enredo, mecânica e dinâmica, afetando tais aspectos de maneira mútua.

A continuidade deste trabalho pode se dar tanto pelo efetivo estudo da relação entre os requisitos não funcionais e cada aspecto de imersão em jogos, como estudos quantitativos sobre os resultados atingidos ao levar em consideração a gestão dos requisitos não funcionais.

Por fim, o trabalho baseia-se em teorias já consolidadas para trabalhar os requisitos não funcionais, e apresenta aspectos práticos para a efetiva gestão desses requisitos em jogos digitais desenvolvidos com Scrum.

Referências

ARAUJO, Allan R. S. Metodologia ágil para projetos de Advergames. Trabalho de graduação defendido em 2006 na Universidade Federal de Pernambuco.

BETHKE, E. Game Development and Production. Plano: Wordware Publishing. Texas, 2003.

BLOW, Jonathan. Game Development Harder Than You Think. New York, 2004.

COHN, Mike. Agile Estimating and Planning. Ed. Pearson Education, 2006.

COHN, Mike. User Stories Applied for Agile Software Development. Addison-Wesley, 2004.

FAGUNDES, Priscila. Framework para Comparação e Análise de Métodos Ágeis. Dissertação de Mestrado defendida em 2005 na Universidade Federal de Santa Catarina.

FERREIRA, Renata B.; LIMA. Metodologias Ágeis: Um Novo Paradigma de Desenvolvimento de Software. Belo Horizonte. Disponível em < http://www.cos.ufrj.br/~handrade/woses/woses2006/pdfs/10-Artigo10WOSES-2006.pdf >. Acessado em 15/05/2011.

JEFFRIES; Ronald. What is Extreme Programming? 2011. Disponível em <http://xprogramming.com/what-is-extreme-programming/>. Acessado em 02/03/2011.

KEITH, Clinton. Agile Game Development With Scrum. Boston. Ed. Addison-Wesley Signature Series, 2010.

KOSCIANSKI, André; SOARES, Michel. Qualidade de Software Aprenda as Metodologias e Técnicas mais modernas para o desenvolvimento de software. 2ª edição. São Paulo, 2007.

LAUBISCH, A., CLUA, E. 2010. Agile Game Development With Scrum. Disponível em: <http://www.sbgames.org/papers/sbgames10/artanddesign/Full_A&D_21.pdf>. Acessado em: 07/08/2016.

MEIR, JD. The Four Circles of XP (Extreme Programming). Disponível em <http://blogs.msdn.com/b/jmeier/archive/2010/04/04/the-four-circles-of-xp-extreme-programming.aspx>. Acessado em 27/03/2011.

MYLLYAHO, Myllyaho. et al. A review of small and large post-mortem analysis methods. In Proceedings of the ICSSEA. Paris, 2004.

NOVAK, Jeannine. Desenvolvimento de Games. Tradução da 2ª edição Note-Americana. São Paulo, 2010.

PIMENTA, Marcelo; PETRILLO, Fábio. Desenvolvimento de jogos é (quase) ágil, Porto Alegre, Brasil, 2010.

PETRILLO, Fábio. Práticas Ágeis no Processo de Desenvolvimento de Jogos Eletrônicos. Porto Alegre. Brasil, 2008.

PRESSMAN, Roger. Engenharia de Software. 6 Ed. São Paulo: Editora McGraw-Hill 2006.

REIS, ADEMAR S JR; BOGDAN, NASSU T.; JONACK, MARCO A. UM ESTUDO SOBRE OS PROCESSOS DE DESENVOLVIMENTO DE JOGOS ELETRÔNICOS (GAMES). CURITIBA. 2002.

SHORE, JAMES; WARDEN, SHANE. THE ART OF AGILE DEVELOPMENT. CALIFORNIA. ED. O'REILLY MEDIA. 2008.

SOMMERVILLE, IAN. ENGENHARIA DE SOFTWARE. 6 ED. SÃO PAULO: EDITORA ADDISON WESLEY, 2003.

TELES, VINÍCIUS M. EXTREME PROGRAMMING. SÃO PAULO: NOVATEC EDITORA, 2004

VASCO, CARLOS; VITHOFT, MARCELO; ESTANTE, PAULO. COMPARAÇÃO ENTRE METODOLOGIAS RUP E XP. TRABALHO DE PÓS-GRADUAÇÃO DEFENDIDO EM 2006 NA PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ.