Mestrado em Engenharia Informática
Estágio
Relatório Final

CrowdPlay
Crowdsourcing Gameplay Experiences

João Pedro da Silva Amsellem
amsellem@student.dei.uc.pt

Orientador:
Dr. Licínio Roque

2 de Setembro de 2014

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Resumo

CrowdPlay é uma plataforma de *crowdsourcing* capaz de proporcionar aos *game designers* uma opção viável para análise de protótipos de jogo de forma fácil e rápida de configurar. Possibilita a distribuição de cenários de teste de jogo, com base na parametrização das condições de ensaio, e a recolha, análise e visualização de dados de *gameplay* de videojogos, em grande escala e direcionada para o estudo da experiência de uso proposta pelo protótipo.

Mesmo não conhecendo a amostra que realiza a avaliação, estudos revelam que os resultados obtidos, comparando os métodos tradicional e através de uma plataforma colaborativa, são semelhantes. A diversidade de pessoas e a rapidez de conclusão do ciclo de teste são duas grandes vantagens para este tipo de abordagens.

Ao longo deste relatório são analisadas diversas abordagens e tecnologias para a implementação desta ferramenta. É ainda apresentado um esboço da arquitectura a implementar e um plano de atividades. Foram implementadas as duas interfaces (*gameCenter* e a interface para os *game designers)* e ambos os servidores definidos na arquitectura. Dada a sua dimensão, não foi possível implementar todos as tarefas presentes na lista de requisitos. O processo de desenvolvimento seguiu a hierarquia de prioridades da mesma lista.

Para validação da plataforma foram realizados testes de performance no webServer2 e testes de usabilidade em ambas as interfaces.

Palavras-Chave

User Experience, CrowdPlay, Crowdsourcing, parametrização de videojogos, avaliação, videojogos

Índice

Lista de Figuras

Capítulo 1
Introdução

Quando se pretende uma avaliação de videojogos, a principal abordagem utilizada é a experimentação. Associada à experimentação poderão ainda estar outros métodos como entrevistas ou questionários, ou mesmo observação *in loco*. A grande dificuldade das tradicionais experiências de *game testing* prendem-se com a necessidade de conseguir um grupo de pessoas, normalmente após períodos de seleção apertada, envolvendo uma grande quantidade de tempo e dinheiro, sem contabilizar o processo de tratamento e análise de dados que ocorre necessariamente após os testes. Todo este processo é realizado com o intuído de melhorar o jogo e com um isto aumentar o seu potencial de sucesso. Ao longo do desenvolvimento do jogo são realizadas um conjunto de experiências, sendo que cada uma delas é idealizada tendo por base definições específicas que definem um caso de teste.

CrowdPlay, a solução que apresento neste documento, pretende ser uma plataforma de *crowdsourcing* utilizada para avaliar as respostas comportamentais de vários jogadores a diferentes casos de teste para um mesmo protótipo. Por outras palavras, o que se pretende é verificar se existem diferentes reações do jogador perante mudanças no *gameplay* do protótipo. Será que velocidade da personagem principal influência a forma como o jogador interage com o jogo ? Será condição suficiente para que ele termine o jogo de forma mais rápida ? E o número de obstáculos ou a durabilidade deles, será que também promove alguma alteração ? São dúvidas como estas, específicas de cada protótipo, que poderão ter uma resposta mais conclusiva com a análise dos dados recolhidos. Esta solução, irá assim fornecer ao *game designer* as vantagens do método de teste tradicional e minimizar as desvantagens do mesmo, contribuindo para que seja possível conhecer um pouco melhor este tipo de relações.

Esta plataforma será provida de duas interfaces distintas para dois tipos de utilizadores: jogador e *game designer*. O jogador terá à sua disposição uma aplicação, *gameCenter*, disponível para qualquer dispositivo móvel com um sistema operativo *Android*, sendo que optimizado para *tablet's* com aproximadamente 10 polegadas de dimensão de ecrã. Aqui, o jogador, irá encontrar todos os protótipos disponíveis em cada momento para que se possa divertir. Por outro lado o *game designer* terá à sua disposição uma *interface web*, podendo aceder de qualquer dispositivo fixo ou móvel com acesso à internet, que lhe permite não só adicionar protótipos e definir diferentes casos de teste e em que condições estes poderão ser testados, mas também visualizar os resultados num *dashboard* definido à sua medida. De forma a interligar as partes anteriormente apresentadas é ainda disponibilizado ao *game designer* uma biblioteca com todos os métodos necessários para este possa utilizar múltiplos casos de teste para um mesmo protótipo sem que seja necessário fazer qualquer alteração no protótipo e ainda métodos para recolher todo o tipo de dados necessários.

À medida que os utilizadores do *gameCenter* forem jogando, vão sendo recolhidos dados relativos às suas opções de jogo. O sucesso de cada experiência dependerá essencialmente do próprio *game* designer, uma vez que será ele a definir os dados que pretende analisar e visualizar.

Os protótipos de jogos estarão disponíveis de forma gratuita a todos os utilizadores da aplicação móvel. A maior aliciante para os jogadores é poderem ter à distância de um simples clique um conjunto alargado de mini jogos. Uma vez que para cada jogo existem vários casos de teste, a diversidade aumenta significativamente. Desta forma é conseguida a aproximação voluntária de um grande conjunto de indivíduos às experiências programadas.

Mesmo não conhecendo a amostra que realiza a avaliação, estudos (Komarov S., Reinecke K., Gajos K. Z., 2013) revelam que os resultados obtidos, comparando os métodos tradicional e através de uma plataforma colaborativa, são semelhantes. A diversidade de pessoas e a rapidez de conclusão do ciclo de teste são assim, duas grandes vantagens para este tipo de abordagens.

Este projeto constitui uma contribuição no âmbito do trabalho de doutoramento do Eng. Rui Craveirinha, sendo que parte do trabalho já se encontrava realizado *a priori*. A minha participação enquadra-se maioritariamente na definição de arquitetura e posteriormente no desenvolvimento da plataforma.

Capítulo 2
Estado da Arte

Neste capítulo será apresentado um estudo sobre as áreas de conhecimento inseridas no âmbito deste projeto. Tendo por base esta ideia, o primeiro tema a ser abordado é *crowdsourcing*. Intrinsecamente ligado ao projeto, o seu conceito é definido e são identificados os motivos que levaram ao seu aparecimento. São ainda referidos quatro tipos fundamentais de aplicações que fazem uso deste conceito e um caso prático em que é adotada a framework desenhada por Feldman para a recolha de críticas. Como resultado da sua implementação é criada uma ferramenta colaborativa, nesta caso Mturk. A avaliação de *user interfaces* foi mais um caso prático encontrado que realçava vantagens no uso de ferramentas colaborativas, fazendo comparações com os métodos tradicionalmente mais usados.

Qualquer avaliação feita tem que definir os princípios pelos quais se rege. O próximo tema, acaba por ser um seguimento natural do anterior. O objetivo do estudo deste tema, *User Experience,* foi principalmente clarificar o seu conceito. UX é um termo amplamente abordado em diversas situações contribuindo, deste modo, para o seu conteúdo abstrato.

Os temas que se seguem, Métodos de Recolha de dados e Análise de Tecnologias prendem-se na necessidade de perceber como é que a solução poderia ser implementada, ainda que em âmbitos diferentes. O estudo do primeiro permitiu aumentar aos conhecimentos e verificar quais as possíveis opções para que fossem, como o próprio nome indica, recolhidos dados do comportamento dos jogadores. Já o segundo, Análise de Tecnologias, revelou-se essencial na escolhas feitas para o desenvolvimento da solução desejada. Foram estudadas características de algumas opções de desenvolvimento existentes no mercado e tendo por base alguns critérios de qualidade foi possível encontrar o que se esperam ser as melhores alternativas.

Cada um dos temas referidos revelaram-se de extrema importância para a definição do projeto. Não só permitiram desenvolver ideias e conceitos como ajudaram na definição de uma primeira versão da arquitetura. Introduzidos os tópicos, será então agora altura de revelar os resultados obtidos do estudo, o estado de arte, propriamente dito.

2.1. Crowdsourcing

Crowdsourcing é o ato de tornar uma tarefa tradicionalmente realizada por um agente numa resolução aberta a um conjunto grande, número indefinido, de pessoas. Por outras palavras, é utilizar os princípios que tão bem funcionam em projetos de *software open source* e aplicá-los em qualquer outra área. (Jeff Howe, 2008)

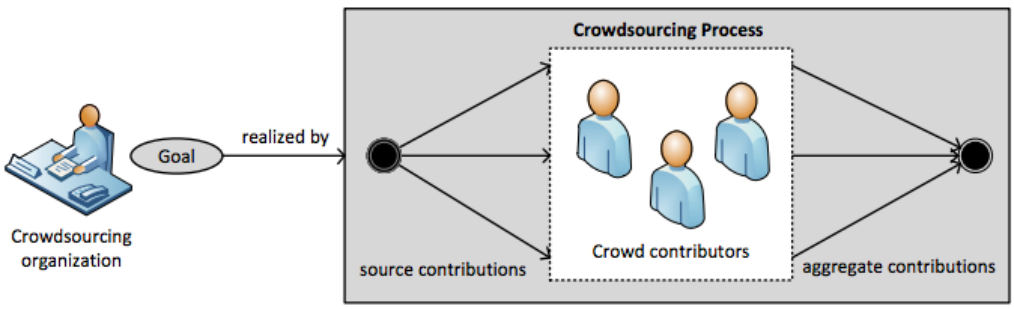

Figure 1 – Ilustração de um processo de *CrowdSourcing* (*Towards a Taxonomy of Crowdsourcing process*, 2011)

O termo *Crowdsourcing* foi pela primeira vez editado em 2006, tendo sido escrito por Jeff Howe e editado na revista *Wired Magazine*. Aqui é feita uma clara distinção entre *Crowdsourcing* e *Outsourcing*. Dando como exemplo duas empresas *NineSigma* (fundada em 2000) e *InnoCentive* (fundada em 2001) que ao contrário de contratarem pessoas externas para realizar tarefas específicas e bem definidas, aproveitam o conhecimento gerado num processo criativo por parte de vários indivíduos.

Em 2008, Jeff Howe, identificou quatro fatores fundamentais que, quando combinados, são responsáveis pelo sucesso do *Crowdsourcing*. Sendo eles:

- Amadorismo (e.g. o desenvolvimento de máquinas fotográficas digitais, a preços relativamente acessíveis, juntamente com a possibilidade de venda de fotos através da *iStockphoto* proporcionou a muitas pessoas a oportunidade de partilhar e/ou ainda conseguir algum retorno financeiro vendendo as suas obras)
- O crescimento do movimento de *open source software* (e.g. Sendo GNU o mais conhecido, fundado em 1983, tendo em 1991 Linus Torvalds anunciado a primeira versão oficial do sistema Linux)
- Equipamento a preços acessíveis (e.g. Áreas como publicidade, fotografia, cinema e até mesmo a música são cada vez mais abertas não só na de compra de equipamentos, que favorecem a produção, como na partilha graças a um conjunto grátis e alargado de meios de divulgação)
- Crescente número de comunidades que partilham os mesmos interesses (e.g. São exemplos disso mesmo *flicker*, *SoundClound*, V*ineScope*)

Numa publicação mais recente, Howe identificou quatro tipos base de aplicações de *Crowdsourcing*:

- *Crowd Wisdom* (i.e. inteligência coletiva)
- *Crowd Creation* (i.e geração de conteúdos por parte de um ou vários utilizadores)
- *Crowd Voting* (i.e. votação e avaliação de determinados produtos)
- *Crowd Funding* (i.e. forma de angariar financiamento para projetos através de doações de indivíduos ou empresas)

Claramente a ferramenta a ser desenvolvida entra na área de *crowd votitng*, uma vez que o seu objetivo é avaliar um protótipo, podendo a mesma ser realizada não só através da recolha de dados do comportamento do jogador ao longo da ação de jogo, mas também através de questionários automatizados a feitos em qualquer altura do jogo. Um outro indicador que pode ser definido por parte do *game designer* é o *rating*. Caso seja esse o interesse, basta que este, peça uma avaliação dentro de um limite (e.g. entre 1 a 5) e que defina no seu *dashboard* um gráfico do tipo *Single Value* onde apresente o seu resultado.

2.1.1. Adaptação da framework de Feldman e o seu uso no Mturk

Nesta secção será apresentada a abordagem sequencial desenvolvida por Feldman para a recolha de críticas e a sua adaptação a estúdios de design. Este estudo prende-se essencialmente com a necessidade de estudar casos práticos onde a organização contribuiu

positivamente para uma avaliação de qualidade. Uma avaliação bem estruturada torna possível uma evolução e consequente melhoria de qualquer produto.

Uma crítica é uma apreciação, uma análise, feita com maior ou menor profundidade, de uma qualquer produção intelectual. Deste modo, quando o objectivo do indivíduo é a recolha de críticas para melhorar a sua obra, é mais importante a qualidade do que a quantidade das mesmas.

Após décadas de investigação de métodos de críticas, foi perceptível que a estruturação do processo de crítica melhora a qualidade da mesma, tornando-a mais interessante para quem a procura.

De forma a garantir qualidade, Feldman (Feldman, 1972, 1994)definiu uma abordagem sequencial baseada num processo de quatro passos. Sendo eles descrição, análise, interpretação e julgamento ou decisão. A ideia principal da *framework* de Feldman é dividir o processo de crítica em várias subtarefas.

Mais tarde (Teo Y.H.,Chai C.S., 2009) tendo por base as ideias pré concebidas de Feldman, Teo e Chai decidiram aplicar a sua *framework* a estúdios de design.

Este novo método baseia-se na realização de quatro subactividades bem definidas:

- Descrição da intenção do design e o público alvo
- Avaliar os pontos fortes
- Avaliar os pontos fracos
- Resumir as declarações realizadas nas etapas anteriores

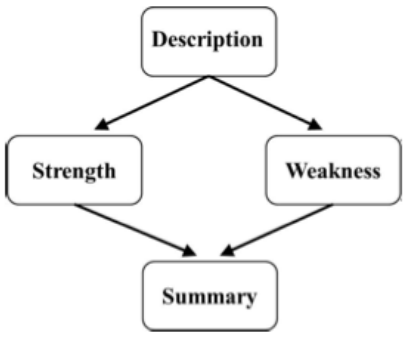

Figure 2 - Ilustração da framework de Teo e Chai (*A Crowdsourcing model for receiving design critique*, 2011)

Esta *framework* centra-se em mecanismos de agregação de esforço individual para garantir que as críticas são o mais completas possíveis. Garantindo a tão desejada qualidade.

De forma a avaliar a viabilidade da *framework* acima descrita em ambientes de *crowdsourcing* foi implementada, com recurso ao *Mechanical Turk* (MTurk da Amazon é uma plataforma online que faz uso de crowdsourcing para a realização de tarefas), um processo cujo objetivo era a avaliação do logótipo da conferência CHI 2011 (Xu A., Bailey B.P., 2011). Ao mesmo tempo foi adicionado o mesmo recurso numa comunidade de críticas online (Deviantart) de forma a ter um termo de comparação. Para além do logótipo foram adicionados outros casos de teste, como por exemplo posters, em ambas as ferramentas. Esta experiência tinha por

objetivo verificar as diferenças qualitativas entre o uso de um processo estruturado e um processo simples, sem utilizar qualquer método, na obtenção de críticas.

Quando comparadas, as críticas resultantes do Mturk, foram significativamente mais construtivas e resultaram numa apreciação com maior importância para os designers.

O objetivo da plataforma CrowdPlay não é de todo recolher críticas sob forma escrita. Antes, dados que quando processados possam dar informações relevantes aos game Designers. No entanto, este caso mostra mais uma vez, as potencialidades de aplicações colaborativas no processo de construção de um produto. Ainda mais importante se torna, quando identifica um caso específico onde a boa estruturação e definição de processos contribuem positivamente para o sucesso de uma avaliação qualitativa.

2.1.2. Avaliação de *User Interfaces* em ambiente de crowdsourcing

Com o desenvolvimento de ferramentas colaborativas do estilo do *Mturk*, *microTask* ou *cloudCrowd* é neste momento possível realizar tarefas como avaliação de designs, preenchimento de dados de forma repetitiva, correção de erros em textos ou até mesmo simples verificações de informação e a sua validação, de forma rápida, barata e não envolvendo um grupo de pessoas específicas, quer seja da empresa em questão quer seja um grupo de desconhecidos previamente escolhido.

A grande facilidade de recrutamento de indivíduos é a principal vantagem deste tipo de ferramentas. Acoplada a esta, estão a hipotética diversidade e quantidade de colaboradores, maior número de condições testadas e um ciclo de revisão mais rápido.

Este tipo de experiências tem um custo monetário associado (tipicamente 1 ou 2 cêntimos por clique), no entanto este é sempre controlado por quem propõe as tarefas. Naturalmente, acredita-se que a recompensa pela realização de uma tarefa condiciona linearmente o seu ciclo de conclusão.

Após testes realizados com três diferentes tarefas, comparando avaliações feitas com recurso ao *Mturk* e em laboratório (em ambiente controlado), em termos qualitativos as diferenças encontras não são significativas. Todos os resultados estatísticos obtidos foram equivalentes nas duas formas de teste realizadas.

Foram ainda realizadas as mesmas tarefas, no *Mturk*, em diferentes horas do mesmo dia. Como resultado, a área geográfica dos colaboradores que realizaram as tarefas alterou. Isto é explicado pelos diferentes fusos horários, provando assim a globalização do processo. No entanto, em termos qualitativos, mais uma vez, os resultados continuaram a ser equivalentes.

Apesar de esta comparação ter sido feita com um baixo número de casos de testes, foi evidente que não sendo possível controlar o grupo de colaboradores e o próprio ambiente a que estes estariam sujeitos, os resultados seriam igualmente bons. Este tipo de ferramentas oferecem uma maior escala de experimentação, acesso a um grupo de colaboradores com maior diversidade e ainda um ciclo de revisão mais rápido. Embora este tipo de metodologias ainda não tenha sido completamente estudado, os autores (Komarov S., Reinecke K., Gajos K. Z., 2013) acreditam que esta pode servir como complemento de outros métodos.

Seria bastante reconfortante, no final de concluído, colocar o *CrowdPlay* dentro do mesmo lote do *Mturk*, *microTask* ou *cloudCrowd*, não só em termos de notoriedade como de

reconhecimento e qualidade de serviço. Ainda que em áreas diferentes, o *CrowdPlay* pretende ser uma plataforma colaborativa, de uso grátis, e que permita testar as reacções ou as formas de jogar de uma população quando confrontadas com um protótipo com uma configuração específica. Com resultados tão animadores em anteriores experiências, é expectável que cada vez mais estúdios de pequenas / médias dimensões procurem este tipo de ferramentas para melhorar o seu produto.

2.2. User Experience e Gameplay Experience

User Experience é um termo adotado pela comunidade de interação humano-computador (HCI) muitas vezes criticado por ser vago, evasivo e efémero É por isso mesmo associado a um alargado conjunto de definições entre as quais usabilidade, beleza ou outros aspetos experienciais de uso de tecnologias ou ferramentas. (*User experience – a research agenda, 2006*)

Alben (1996) definiu *User Experience* como todos os aspectos que envolvem a interação com o produto: o sentimento que provoca quando o tocamos, a forma como rapidamente entendemos o seu funcionamento e o seu propósito, o sentimento que provoca em quem o experiencia e o enquadramento com todo o seu contexto de uso.

O ciclo normal de vida de qualquer design passa pelas quatro etapas na figura em baixo apresentada.

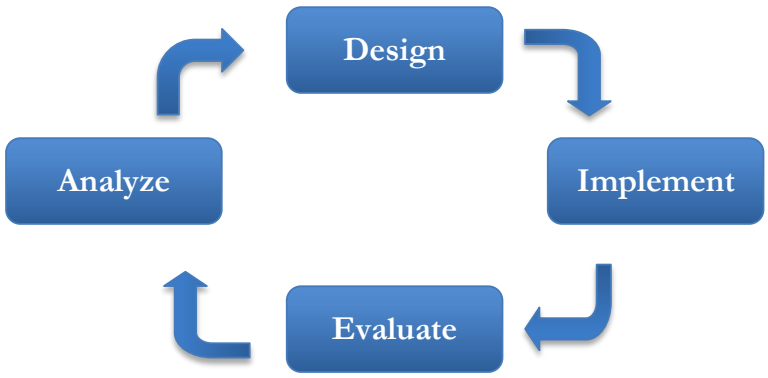

Figure 3 – Método abstrato de design (Hartson R., Pyla P.S, 2012)

De uma forma geral, UX centra-se em aspectos positivos da relação com produtos interativos.

A etapa de avaliação pode ser feita segundo vários fatores, tais como:

- impacto emocional
- cumprimento de objectivos de UX, previamente definidos

Apesar ser uma medida subjetiva, o impacto emocional que determinado design tem sobre um indivíduo pode ser considerado como uma forma de avaliação do mesmo. Esta avaliação depende de muitos contextos associados ao ambiente e ao próprio estado emocional em que o indivíduo se encontra. A forma mais fidedigna de recolha deste tipo de resultados é através de testes biométricos.

Os objetivos de UX de um determinado design podem ser definidos com base em duas perspectivas: de negócio e uso diário ou através de uma identificação do que realmente interessa para uma organização específica. Estes objetivos representam um funcionamento ou aspecto ideal e por isso mesmo podem ser considerados numa avaliação qualitativa do produto criado.

Game Experience é um tipo de *User Experience* mais abrangente e complexo do que outras experiências que os utilizadores possam ter com diferentes tipos de produtos. Sendo mesmo considerado, a par da usabilidade, um dos grandes fatores para sucesso de um jogo.

Os dados recolhidos pela plataforma *CrowdPlay* serão essencialmente numéricos. Como tal não serão avaliados, de forma direta, os sentimos dos jogadores. No entanto, os dados recolhidos poderão dar informações relevantes para o aperfeiçoamento da UX. Informações como dificuldade de jogo, interesse ou até mesmo vontade que o jogador mostra em jogar podem ser identificados com a definição de indicadores e métricas relevantes para o estes casos específicos. Como por exemplo, o tempo que o jogador demora a concluir um jogo, o número de *run sessions* abruptamente terminadas ou ainda o número de vezes que o jogador perde até que consegue terminar o jogo. Mais uma vez, as informações abstraídas serão tão completas como as variáveis definidas e recolhidas.

2.3. Métodos de Recolha de Dados para análise de Gameplay

Quando bem utilizada, um bom conjunto de informação pode se revelar bastante importante. A informação pode representar um ativo bastante valioso em várias áreas do conhecimento. Isto é um facto. A análise de informação recolhida durante experiências farmacológicas contribuem para o desenvolvimento de medicamentos que podem salvar muitas vidas. Outro exemplo prático é a melhoria da qualidade de software, tendo por base testes realizados por pessoas externas.

Recolha de dados é uma expressão que pode ser aplicada em variados contextos. Neste caso particular, são estudados diferentes métodos de recolha de informação sobre o comportamento de um indivíduo durante a ação de jogo. Estes dados, após uma cuidada análise, apoiam a decisão do *game designer*, promovendo deste forma, a percepção do comportamento de uma população perante casos de teste diferentes.

Dada a grande expansão do mercado de dispositivos móveis é possível, neste momento, recorrer a diversas pessoas, aproveitando o seu interesse e desejo de jogar, para consecutivamente melhorar e evoluir jogos.

Mais do que recolher um grande conjunto de dados, para que estes tenham um valor acrescentado, é imperativo que sejam tratados e apresentados de forma perceptível.

A apresentação gráfica dos dados obtidos proporciona uma das formas mais rápidas de interpretação entre outras vantagens que de seguida são enumeradas:

- compreensão de grandes conjuntos de dados

- percepção de propriedades emergentes
- capacidade de encontrar problemas dentro de conjuntos de dados
- capacidade para formar hipóteses

A grande vantagem de utilizar valores recolhidos de situações concretas faz com que uma análise seja menos ambígua.

Existem várias formas de recolher informação dos utilizadores / jogadores. Entres as quais recorrendo a :

- Logs
- Questionários
- Biométricas
- Observação Direta
- Entrevistas

Para que estes dados sejam mais fiáveis é normal que os indivíduos que experimentam o produto não estejam envolvidos no desenvolvimento do mesmo. Assim poderão ser captados comportamentos e resultados que os *game designer*s não estariam à espera.

2.3.1. Logs

Conjunto de dados recolhidos automaticamente pelo produto e que revelam o comportamento do jogador durante a experiência. A escolha dos indicadores e métricas corretos é assim uma das principais dificuldades deste método, uma vez que são pré programados. No entanto, podem ser redefinidos durante este processo, caso não afectem os resultados já recolhidos. Outra opção é reiniciar o processo, guardando os dados resultantes num ficheiro novo, garantindo assim a integridade.

Quanto melhor forem os indicadores escolhidos para análise, mais relevante serão as informações obtidas.

Este método pode ser tão complexo quanto o *game designer* desejar.

A grande vantagem é o facto de ser facilmente automatizável. Isto permite que não só a recolha como a análise e posterior apresentação dos dados possa ser realizada sem que o *game designer* tenha qualquer preocupação durante o fase experimental. É uma boa forma de aglomerar um grande conjunto de dados. Tradicionalmente, são guardados em ficheiro e só posteriormente analisados. Estes têm o formato que o *game designer* entender, normalmente são guardados em csv ou mesmo em ficheiro de texto.

Feita uma avaliação dos prós e contras, ficou claro, que dado os recursos e o objetivo central da plataforma, este é o método que melhor satisfaz as condições pretendidas. Por isso mesmo, será o método principal a ser implementado.

2.3.2 Questionários

Como o próprio nome indica, questionários, é um conjunto de perguntas, de forma escrita, direcionadas aos indivíduos que realizam a experiência. Dependendo do objetivo do estudo, podem ser realizadas em qualquer fase da experiência; antes, durante ou depois. As perguntas podem ser de resposta múltipla e / ou por extenso.

Mais uma vez, este método pode ser automatizado. Existem já plataformas, normalmente de avaliação, que utilizam este método.

Este método limita o público alvo da experiência pelo simples facto de a esta estar associada um idioma. Mesmo sendo realizado numa língua amplamente falada por todo mundo, como acontece com o Inglês, expressões características ou mesmo diferentes interpretações, podem ser um factor limitativo da experiência.

Outro facto que não se pode menosprezar, é a veracidade das respostas. Isto, mais facilmente, acontece nas perguntas com resposta no formato de escolha múltipla. Ao longo do processo de avaliação, por qualquer motivo, o utilizador pode perder o interesse no questionário e começar a responder de forma aleatória, sem que necessariamente as respostas façam sentido.

Existem vários métodos para despistar falsos questionários. Por exemplo a introdução de perguntas de teste, normalmente contraditórias, verificação humana ou ainda, dependendo da importância das experiências, a realização de entrevistas telefónicas.

Sempre que o *game designer* tiver necessidade, e por isso mesmo pretenda, pode integrar este método no protótipo a ser testado na *CrowdPlay*. Contudo irão ser impostas algumas restrições. Como as respostas serão guardadas nos ficheiros de log, os valores recolhidos terão que ser numéricos (podendo obviamente existir uma relação entre as respostas escritas e os valores guardados). Isto implica que não seja possível recolher qualquer tipo de texto, e as respostas terão que ser no formato de escolha múltipla.

2.3.3. Biométricas

Este método abrange o estudo de áreas fisiológicas e comportamentais de indivíduos durante as experiências. Pode envolver o estudo de cada uma delas ou mesmo de ambas em conjunto dependendo dos resultados que se pretendem recolher e dos recursos tecnológicos disponíveis para tal.

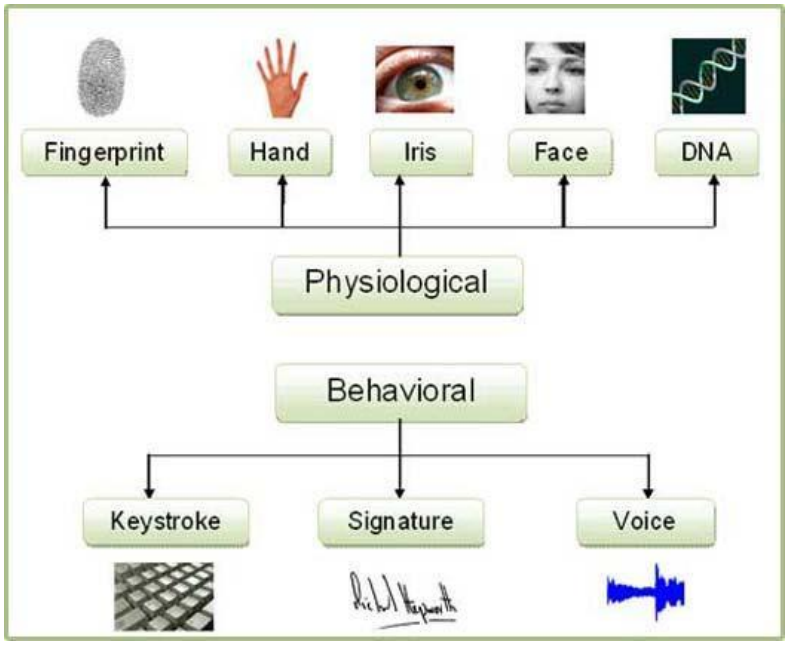

Figure 4 – Biométricas

A popularidade deste método tem vindo a aumentar, na área de vídeo jogos, por entre investigadores de UX na recolha de dados fisiológicos. Um exemplo prático para o uso deste método é a recolha de emoções como raiva, preocupação, tristeza, entre outros.

Apesar de todas as vantagens que este método tem, como por exemplo o facto de não ser subjetivo, de ser direto e fidedigno, também apresenta algumas desvantagens. Factores que muitas vezes inviabilizam o uso deste método como forma de recolha e avaliação de dados. É um método extremamente complexo e que é impossível de realizar em contextos não controlados pelos organizadores das experiências. À sua complexidade, junta-se o facto de ter que ser realizado por profissionais, com conhecimentos previamente adquiridos em análise e interpretação de sinais. De salientar é ainda o facto de envolver necessariamente um conjunto alargado de materiais específicos e de alto valor comercial.

Por todos os motivos anteriormente enunciados, este método não é uma opção viável para a plataforma a desenvolver e por isso mesmo foi uma hipótese descartada.

2.3.4. Observação direta

Método que prima pela observação do comportamento de indivíduos durante a experiência. A observação pode ser feita em tempo real, no exato momento de ação, ou após a experiência com a ajuda de gravações. De forma a minimizar uma possível influência no comportamento dos indivíduos durante a experiência, é normal esta ser gravada e a observação ocorrer a posteriori.

Apesar de, em certos casos, poder ser considerada como uma factor essencial, tem as suas vantagens e desvantagens. No âmbito de ferramentas colaborativas não é utilizado por duas principais razões. Não é um método que se possa considerar automatizável (por isso

mesmo, necessita de ser realizado em ambientes controlados) e entra em colisão com questões legais de privacidade.

Dadas as desvantagens e não tendo um forte motivo para que seja utilizado, este método não vai ser utilizado no produto a ser realizado.

2.3.5. Entrevistas

Tal como nos questionários, dependendo do objetivo do estudo, pode ser realizada em qualquer fase da experiência.

Este método consiste numa conversa entre duas ou mais pessoas. O entrevistador é a pessoa que guia a conversa, fazendo perguntas, retirando notas das respostas do entrevistado. Pode, caso necessário, ser gravado para que seja feita uma posterior análise e/ou revisão da experiência.

Podendo recorrer à mecanização das perguntas, é possível optimizar o processo de entrevista. Ainda que a tornando mais impessoal. De qualquer das formas, é sempre necessário que as respostas sejam gravadas e posteriormente analisadas. Apesar de poder ser parcialmente automatizável, não oferece grande benefício em termos de recursos necessários, dado que será sempre necessários que estas sejam revistas por humanos.

Como tal, também este método, não será utilizado.

2.4. Gameplay Metrics Systems

Gameplay Metric Systems são ferramentas que permitem a recolha e análise de forma gráfica de dados provenientes de jogos ou protótipos de jogos.

Os valores podem ser recolhidos através de qualquer uma das metodologias anteriormente abordadas. No entanto, a mais usada acaba por ser através de logs.

As primeiras ferramentas amplamente divulgadas foram as desenvolvidas para jogos na plataforma Flash. Mochibot, Nomoba e Playtomic são três exemplos de soluções existentes para monitorar o comportamento de indivíduos / jogadores. Mochibot foi, inclusive um dos responsáveis pelo serem adicionados *layers* de publicidade sobre jogos flash online. Isto porque esta ferramenta foi, e pode ainda ser, usada para controlar o número de carregamentos de um determinado jogo e assim justificar as receitas publicitárias angariadas pelos *game developers*.

Hoje com a evolução deste tipo de ferramentas é agora possível monitorar qualquer evento de *gameplay* de um jogo.

Fazendo a transição para plataformas PC e de consolas, SkyNet, fugindo um pouco ao conceito anterior, foi uma ferramenta mais centrada no processo de desenvolvimento do jogo. Era usado, por exemplo, para fazer *bug tracking*. Este é um exemplo da importância da análise de dados gráficos em outras áreas de desenvolvimento de jogos (e não só).

Um outro caso que exemplifica a potencialidade deste tipo de ferramentas é o Microsoft TRUE. Esta solução pode ser usada em dois tipos diferentes de análise: *user testing* e *beta*

testing de jogos. Dependendo do efeito para o qual é programada, TRUE recolhe informação através de diferentes metodologias (vídeo, logs, questionários).

Com o lançamento do *Dead Space 2*, em 2011, surgiu também uma nova ferramenta, DataCracker. Com o objectivo de melhorar *best pratices*, foi criado um *side project*, na *Electronic Arts (EA)*, com o intuito de desenvolver uma ferramenta capaz de analisar e apresentar de forma mais organizada e perceptível dados recolhidos durante os testes do jogo que estava a ser desenvolvido.

Ao verificar os resultados conseguidos, o que inicialmente estava previsto ser um produto de consumo interno, passou a ser de uso aberto a todo o público. Atualmente, basta fazer um registo na plataforma, e escolhendo um serviço grátis ou pago, após o *upload* dos ficheiros de *log* e de algumas configurações, visualizar os mesmos dados de forma mais inteligível.

Cada uma das anteriores ferramentas tem um *dashboard* que permite ao utilizador analisar e visualizar dados. Em anexo está um exemplo de um *dashboard* de uma ferramenta de monitorização de dados. Para além deste, foram ainda analisados outras exemplos que permitiram ter uma noção mais clara das necessidades existentes.

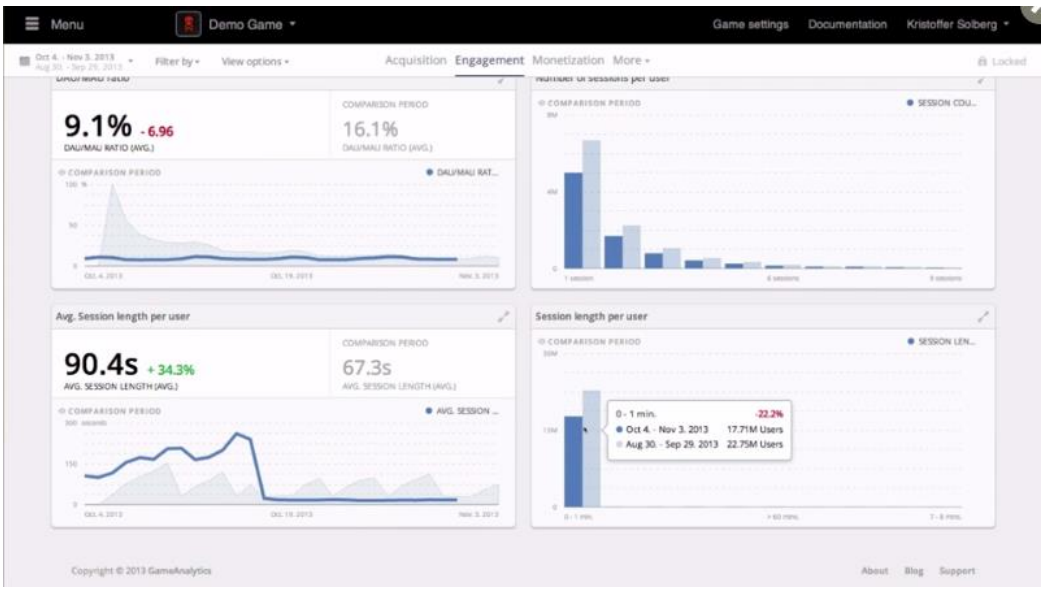

Figure 5 – Exemplo de dashboard

2.5. Análise de Tecnologias

Esta secção tem por objetivo ajudar na escolha das tecnologias a usar no desenvolvimento desta plataforma.

De forma a subdividir o processo de desenvolvimento em tarefas mais pequenas, foi pensada uma forma estruturada, que quando englobada forma a solução final.

Assim, de forma a tornar o processo de jogo para o utilizador o mais transparente e normal possível, esta interface será construída com recurso a uma *game engine*. A grande vantagem é que o cliente / jogador não será obrigado a instalar mais nada a não ser a aplicação principal.

A principal desvantagem é limitar os jogos apresentados à *game engine* escolhida. Esta desvantagem, poderá mais tarde ser anulada uma vez que todo o desenvolvimento, excepto a aplicação do utilizador, estará pensada para não ser limitativa a esse ponto.

De forma a agilizar o processo de desenvolvimento de um dos servidores e da interface do *game designer* será utilizada uma *framework* de desenvolvimento rápido. As várias opções e a escolha final é baseada em critérios bem definidos e são especificados no decorrer deste documento.

2.5.1. Game Engine

De todas a soluções existentes no mercado, as opções estudadas revelaram-se as mais viáveis quando feita uma comparação entre qualidade, complexidade e preço exigido pela licença. Como a *CrowdPlay* é para ser desenvolvida, inicialmente para numa versão android todas as ferramentas que não suportavam um *deploy* para essa plataforma foram logo excluídas.

Dada à carga que a aplicação do cliente / jogador poderá ter, não ter um *deployment* nativo iria baixar a performance e deste modo todas as *engines* estudadas conjugam esse requisito.

A tabela em baixo apresenta quatro opções que preenchem os requisitos inicialmente definidos e como tal, correspondem às *engines* que melhor foram estudadas.

	Corona	Unity3d	Marmelade	Cuttlefish
Linguagem	Lua	C#, JS, Boo	C++	Cuttlescript
Comunicação servidor	httpRequest	httpRequest	httpRequest	httpRequest
Formato deployment	Nativo	Nativo	Nativo	Nativo
Plataformas	Android, Ios	Android, Ios	Android, Ios	Android, Ios
	Kindle, nook	W. phone, blackberry	wind, blackberry	wind, blackberry
		Browsers,Linux, mac os, Windows pro		
Licença	$192ano	Free / $1500	$15mês	Free / $99ano

Será dada preferência a uma *engine* sem custos associados e que aliado a isso, consiga uma boa performance, tanto na a aplicação como na relação com o desenvolvimento de jogos.

Corona sdk só permite o desenvolvimento na linguagem Lua e apenas consegue fazer *deployment* para Android e iOS. Este facto é ainda partilhado com as outras duas *engines*, Marmelade e Cuttlefish. O que limita, as opções de escolha para uma expansão para outra plataformas no futuro.

Aliado a este facto, de todas as *engines* analisadas, só a Unity e a Cuttlefish é que têm uma versão grátis. Como este era um factor preferencial, a prioridade é dada a uma das últimas duas.

Após mais pesquisas, foram encontradas mais um dado que acabou por ajudar favoravelmente na decisão pela Unity. De acordo com o *Slant*, plataforma colaborativa de apoio à decisão, Unity é a segunda *engine* mais votada para o desenvolvimento de aplicações 2D. E ainda em finais de 2012, Christopher Reynolds escrevia na *mobyaffiliates* que os 10 aplicações mais compradas na App Store correspondiam a jogos e a Unity era uma das *game engines* mais usadas.

Por todos estes motivos a escolha recaiu sobre a Unity.

2.5.2. Framework de Desenvolvimento

De forma a agilizar o processo de desenvolvimento da interface do *game designer* e de um dos servidores, foi decido usar uma *framework* de desenvolvimento.

Como todas as *frameworks* não têm custo financeiros associados, esse apesar de ser um factor potencialmente condicionador, não interferiu com a escolha.

As hipóteses foram as escolhidas, por serem conhecidas graças de desenvolvimentos de outras aplicações de grande sucesso e atualmente bem cotadas no mercado. No quadro em baixo, estão então, as três opções escolhidos como objeto de estudo.

	Ruby on Rails	Django	Play! 2.2.x
Linguagem – servidor	Ruby	Python	Java
Linguagem – cliente	Html + css	Html + css	Html + css
Documentação	Bom	Bom	Bom
Concurrent HorsePower	Médio	Bom	Alto (AKKA)
Arquitetura	REST	REST	REST
Design Pattern	MVC	MVC	MVC

Ambas as opções apresentadas revelaram ser candidatas à escolha e a decisão acabou por ser tomada tendo em conta os pormenores. Tanto RoR como Django e a Play! são *frameworks* com uma arquitetura REST e estruturadas segundo a *design pattern Model View Controller* (MVC). Em termos de *front-end* as linguagens a ser usadas são semelhantes, sendo que apenas no *back-end* se encontram diferenças mais notórias. RoR usa Ruby, Django é em Python e a Play! é em Java.

A escolha final recaiu na Play! Framework 2.2.x principalmente por motivos de performance fazer uso da Akka, uma *toolkit* para construção de aplicações altamente concorrentes, distribuídas e com alta tolerância de erros. Outros factores se podem juntar ao anterior, como por exemplo o facto de tanto *Python* como *Ruby* serem linguagem dinâmicas, ou até mesmo, o facto de não existir necessidade de *redeploy* do servidor ser um dado interessante.

Capítulo 3
Metodologia

Design Science Research (DSR) é um método de orientação desenhado para realizar pesquisas na área de sistemas de informação. No entanto, dado o seu carácter genérico, pode ser utilizado em outras áreas de estudo não comprometendo o sucesso dos resultados obtidos.

A sua principal característica é promover a geração de conhecimento através da criação e posterior análise de uso e/ou performance. Como tal, e de forma a potenciar o crescimento continuo de conhecimento, pode ser usada de forma cíclica, como ilustrado na figura que se segue.

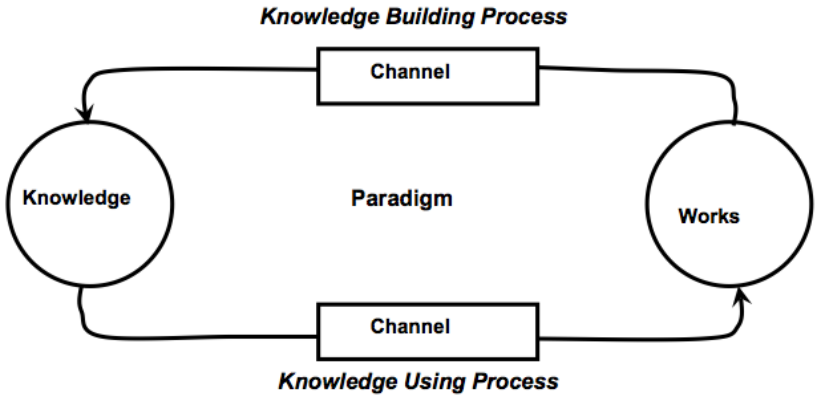

Figure 6 – Modelo de aquisição de conhecimento in *Design Science Research in Information Systems*, 2013

Novos conhecimentos são gerados tendo por base trabalho já realizado, e consecutivamente, o trabalho a realizar ganha com o resultado dessa evolução. O facto de ser um ciclo, idealmente, espera-se que os resultados obtidos vão sendo melhorados a cada iteração.

Analisando de perto a estrutura deste método é possível identificar cinco fases complementares. Em baixo, a figura, sintetiza a DSR numa diagrama.

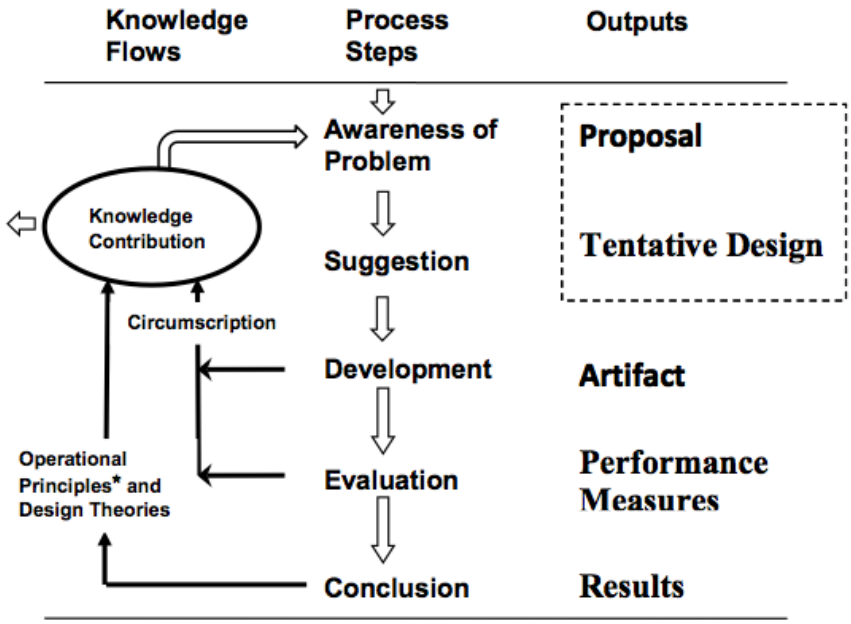

Figure 7 – Metodologia DSR (*Design Science Research in Information Systems*, 2013)

A metodologia DSR começa com o a percepção do problema. Esta noção por ser conseguida tendo por base várias fontes, como por exemplo pesquisas bibliográficas ou novos desenvolvimentos na industria e/ou numa área de estudo. Como resultado deste passo é gerada uma proposta de problema a ser pensado.

Segue-se então o processo de sugestão, intimamente ligado à proposta anteriormente definida, apela à criatividade como forma de resolução da proposta. O objectivo é encontrar uma forma inovadora, criando novos ou integrando artefactos já existentes, para solucionar o problema definido.

Tendo a proposta de resolução já sido feita, com a elaboração de documentos de requisitos, arquitetura e outras avaliações, chega-se a altura de prototipagem, fase de desenvolvimento. Nesta fase é implementada a solução anteriormente pensada. Como seria de esperar, nesta fase, as técnicas a utilizar dependem essencialmente do tipo de artefacto que se pretende criar.

A avaliação, processo que se segue, da solução gerada é realizada de acordo com critérios implícita ou explicitamente definidos aquando a proposta. Todas as diferenças encontradas entre a solução e a sugestão previamente definida, são reportadas. Após a analise cuidada é concluída a validação do protótipo que pode ou não ser coerente e responder a todos os requisitos pré-definidos.

A última fase desta metodologia pode ser considerada como o final do projeto ou o final de um ciclo. A conclusão acaba por ser o processo onde todo o conhecimento adquirido ao longo das fases anteriores é sintetizado. O término ou não do projeto é decidido tendo por base a validação feita anteriormente. Caso o artefacto gerado cumpra os requisitos e resolva de forma satisfatória a proposta, é dado por concluída a pesquisa. Por sua vez, caso as condições esperadas não sejam atingidas, e caso seja possível, é dado inicio a um novo ciclo da metodologia tendo já por base todo o conhecimento gerado na iteração anteriormente realizada.

Apesar de bem definida, os inputs e outputs em cada fase do processo podem ser diferentes. O conhecimento que vai sendo gerado durante diferentes fases e/ou iterações, condiciona positivamente cada etapa seguinte.

3.1. Objectivos

O resultado final esperado é uma arquitectura para sistema de distribuição e ensaio de protótipos de jogo, online. Espera-se a implementação de uma prova de conceito constituída por um servidor online que permite a configuração e distribuição de cenários de teste e uma aplicação para um tablet android projectado e construído para permitir a distribuição e execução de cenários de jogo.

Por outro lado, é ainda necessário outra aplicação *web* onde seja possível fazer *upload* de jogos, definir e visualizar dados resultantes do comportamento dos jogadores durante a ação de jogo, capturadas pela utilização da aplicação de jogo distribuída.

Este projecto tem assim quatro objectivos principais:

- Possibilitar a parametrização de jogos/protótipos
- Criar um serviço de registo de eventos de jogabilidade
- Criar uma aplicação avançada de visualização de indicadores de UX
- *Deployment* do sistema a ensaiar para múltiplos utilizadores online

CrowdPlay será uma ferramenta colaborativa que terá como principal objectivo, ajudar *game designer*s, que desenvolvem os seus jogos em Unity, ou outras engines, a melhorar o gameplay das suas obras. Para isso fará uso de uma aplicação para Android e otimizadas para tablets de 10', que permitirá aos seus utilizadores jogar todos os protótipos disponíveis. Durante cada jogo, serão guardados dados correspondentes ao comportamento do jogador face ao ambiente de jogo. Estes dados serão posteriormente enviados para o servidor e tratados de forma a que mais tarde possam ser apresentados ao *game designer* sob a forma de gráficos. Esta aplicação, poderá em situações futuras ser também distribuída noutras plataformas, como por exemplo iOS e Windows 8.

Para que o *game designer* adicione e faça uma gestão apropriada do seu projeto, será disponibilizada uma interface *web* para criar projecto, configurar ensaios e parametrizar cenários de teste. Será possível adicionar, editar e remover projectos. Um projecto será constituído por um protótipo e todas as variáveis e condições de teste.

Para que mais tarde, o *game designer* seja capaz de analisar os dados de forma mais eficiente será então, disponibilizado uma área, o *dashboard*, onde este terá a possibilidade de adicionar ou remover novos gráficos. Pretende-se que esta área seja personalizável e que forneça ao jogador uma panóplia alargada de opções de escolha, no que aos gráficos diz respeito.

Toda a ferramenta tem que ser escalável ao ponto de suportar muitos utilizadores. Este fator terá que ser tido em conta durante todo o processo de análise e desenvolvimento.

3.2. Abordagem de desenvolvimento de software

Ao entrar na fase de desenvolvimento do protótipo serão utilizadas algumas técnicas de gestão de projetos de software de uma metodologia ágil, tendo por objectivo tornar todo o processo iterativo e incremental.

SCRUM está pensado para a gestão de projetos cuja equipa de desenvolvimento é constituída por vários elementos. Neste caso particular, só existe uma pessoa a desenvolver a solução.

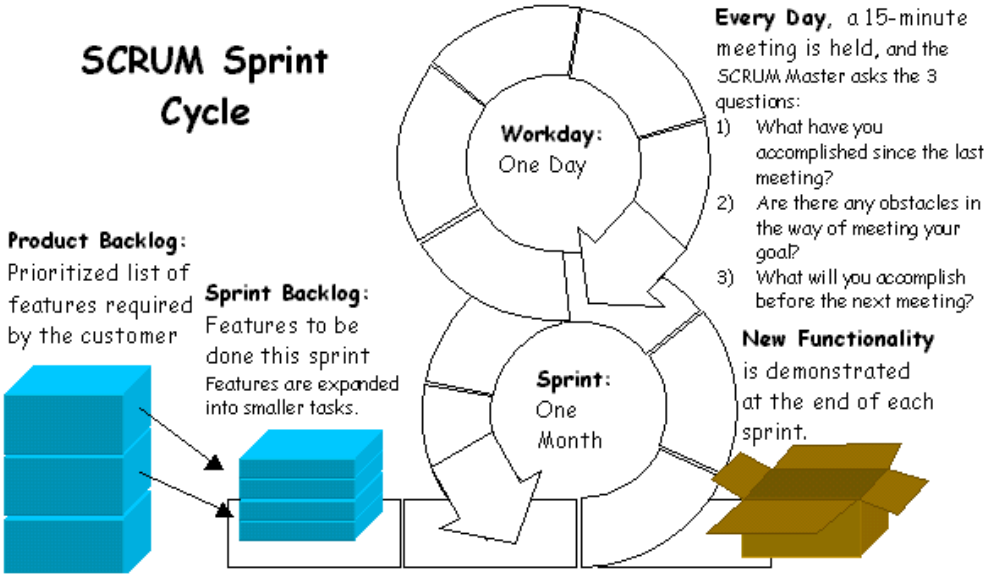

Figure 8 - Ciclos de Sprint da metodologia SCRUM (Marc Clinfton, J. Dunlop, 2003)

A escolha da metodologia SCRUM recaiu principalmente na sua adaptabilidade a projetos de grande complexidade e que por isso mesmo não é possível fazer um planeamento completamente estático.

Serão realizados sprints com períodos mensais e reuniões, no mínimo, semanais. As reuniões semanais terão como intervenientes, o programador e os *stakeholders*.

Ao longo do processo de desenvolvimento serão lançadas, no mínimo, duas versões do protótipo. Sendo as datas e os requisitos de cada *release* definidas nos subcapítulos seguintes.

3.3. Plano de Atividades

A fase de implementação irá começar logo após a data da defesa intermédia e terminará no mês de Maio. Durante ainda este mês será dado inicio a um período de avaliação e validação da plataforma.

Em baixo, a tabela apresenta as atividades que estão planeadas. Por questões de organização e visibilidade, a tabela está distribuída por meses, sendo que a cada um deles corresponde uma ou mais tarefas.

Este plano apenas explicita as tarefas e não as distribui segundo qualquer espaço temporal. Esse trabalho será realizado e apresentado no subcapítulo que se segue, *Segundo Semestre*, onde será apresentado um diagrama de Gantt para o efeito.

Mês	Atividades
Fevereiro	• Definição mais detalhada da arquitetura • Implementar design da app móvel • Configurar servidor2 • Implementar screens iniciais app móvel • Configurar servidor1
Março	• Implementar load de jogos na app móvel • Implementar save de logs no tablet • Implementar design de interface do *game designer* • Implementar design de criação e edição de indicadores e métricas • Implementar back-end para as screens feitas • Implementar listagem de jogos da app móvel
Abril	• Ciclo de avaliação da aplicação • Implementar sendLogs para servidor2 (run session / play session) • Implementar sendLogs para servidor1 • Implementar parsing de logs e save da base de dados
Maio	• Implementar dashboard • Fase de testes e avaliação
Junho	• Relatório final

A primeira tarefa a ser realizada será a definição mais detalhada da arquitetura. Antes de começar a fase de desenvolvimento é preciso validar e caso necessário reformular os conceitos até então pensados. O mês de Fevereiro terá como principal foco o desenvolvimento da aplicação. Será aqui implementado o *front-end* e o *back-end* das *screens* iniciais (*login* e registo, *screen* seguinte). Outra tarefa a realizar é a configuração dos dois

servidores. Estes serão criados e ficarão disponíveis para que seja possível em qualquer altura começar o desenvolvimento do *back-end*.

Em Março, o foco continuará a ser na aplicação móvel, no entanto irá também ser começada a interface do *game designer*. Na aplicação móvel será implementado o *load* de jogos externos, dentro da aplicação e os scripts necessários para integrar nos jogos de forma a que seja possível guardar os dados durante a execução do jogo. Será também implementado a listagem automática de todos os jogos disponíveis para tal. Na interface do *game designer* será implementado o *front-end* e *back-end* de *login*, registo, *upload* de jogos, definição de métricas e indicadores.

O mês seguinte, Abril, irá começar com uma avaliação do produto até então desenvolvido. Após a este ciclo serão implementadas as correções definidas. Terminado esta fase será dado relevo aos servidores. Será desenvolvido os métodos responsáveis pelas trocas de dados entre os dois servidores.

Em maio, será dado foco à apresentação dos dados. Será implementado o *front-end* e *back-end* relativo ao *dashboard*. O objetivo é que este seja o mais personalizável possível, que permita a criação de vários tipos de gráficos e que seja ainda possível edita-lo.

3.3.1. Segundo Semestre

Na figura seguinte está representado as tarefas e o tempo planeado para cada uma delas. Esta representação foi feita através de um diagrama de Gantt.

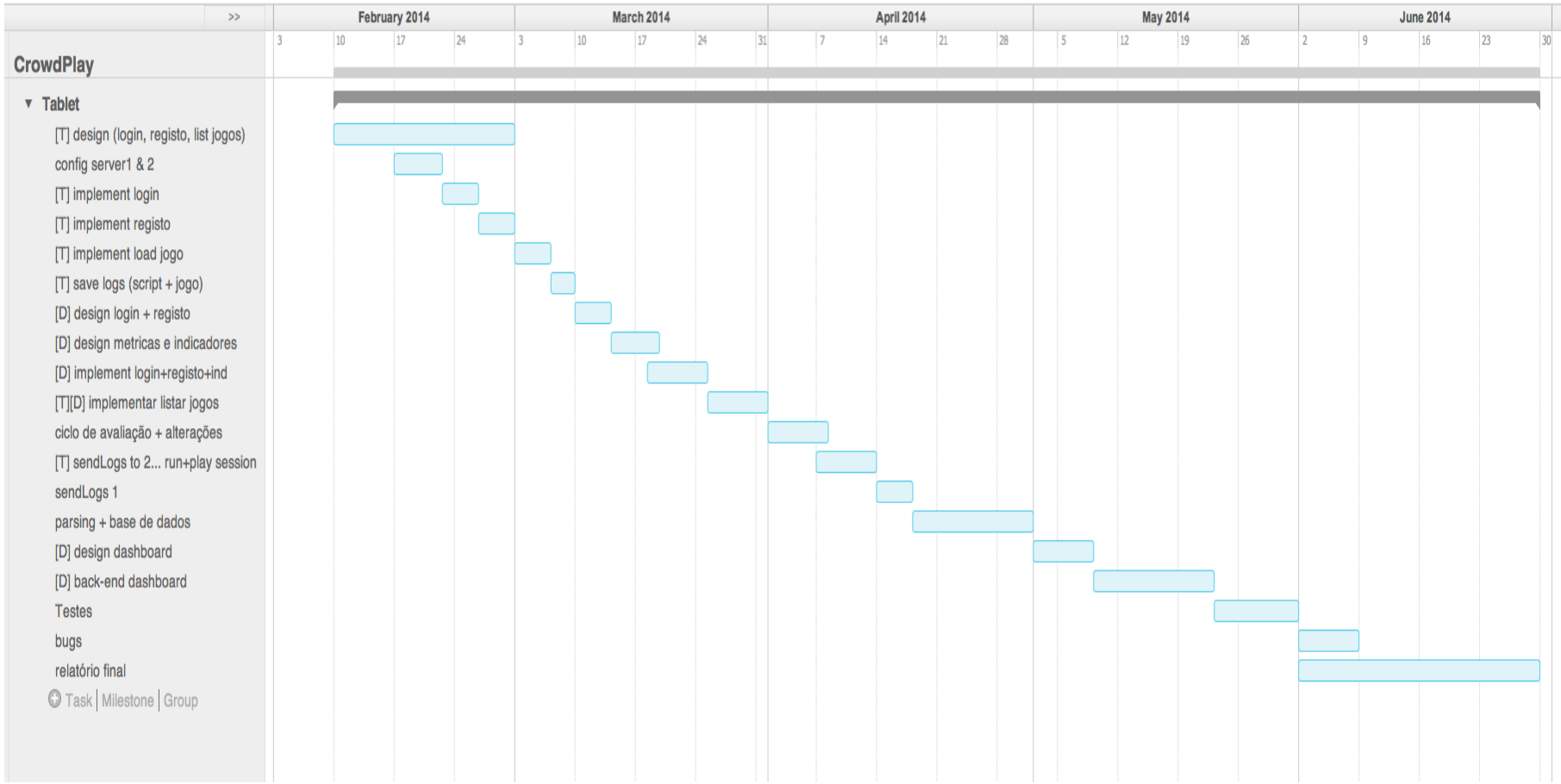

Figure 9 - planeamento 2º semestre

3.4. Milestones

As *milestones* estão implicitamente definidas no plano de atividades e no diagrama de Gantt definidos anteriormente.

Outras datas:

Data	Descrição
17 de Fevereiro	Arquitetura completamente definida
20 de Março	Design da aplicação móvel concluído
4 de Abril	Listagem e escolha de Jogos completamente funcional
19 de Maio	Funcionalidades base concluída (dashboard excluído)

A avaliação de sucesso em cada *milestone* é binária, isto significa que só é considerado como ultrapassada caso tenham sido realizadas todas as tarefas definidas até ao seu término.

3.5. Plano de actividades Real

Como é possível verificar na imagem que se segue, existiram algumas alterações relativamente ao planeamento inicial. O principal motivo para tal acontecer foi o facto de na altura em que o plano inicial foi definido (antes de Fevereiro) o documento de requisitos não estava concluído. No diagrama apresentado na imagem que se segue é possível ver o plano de actividades real para o segundo semestres.

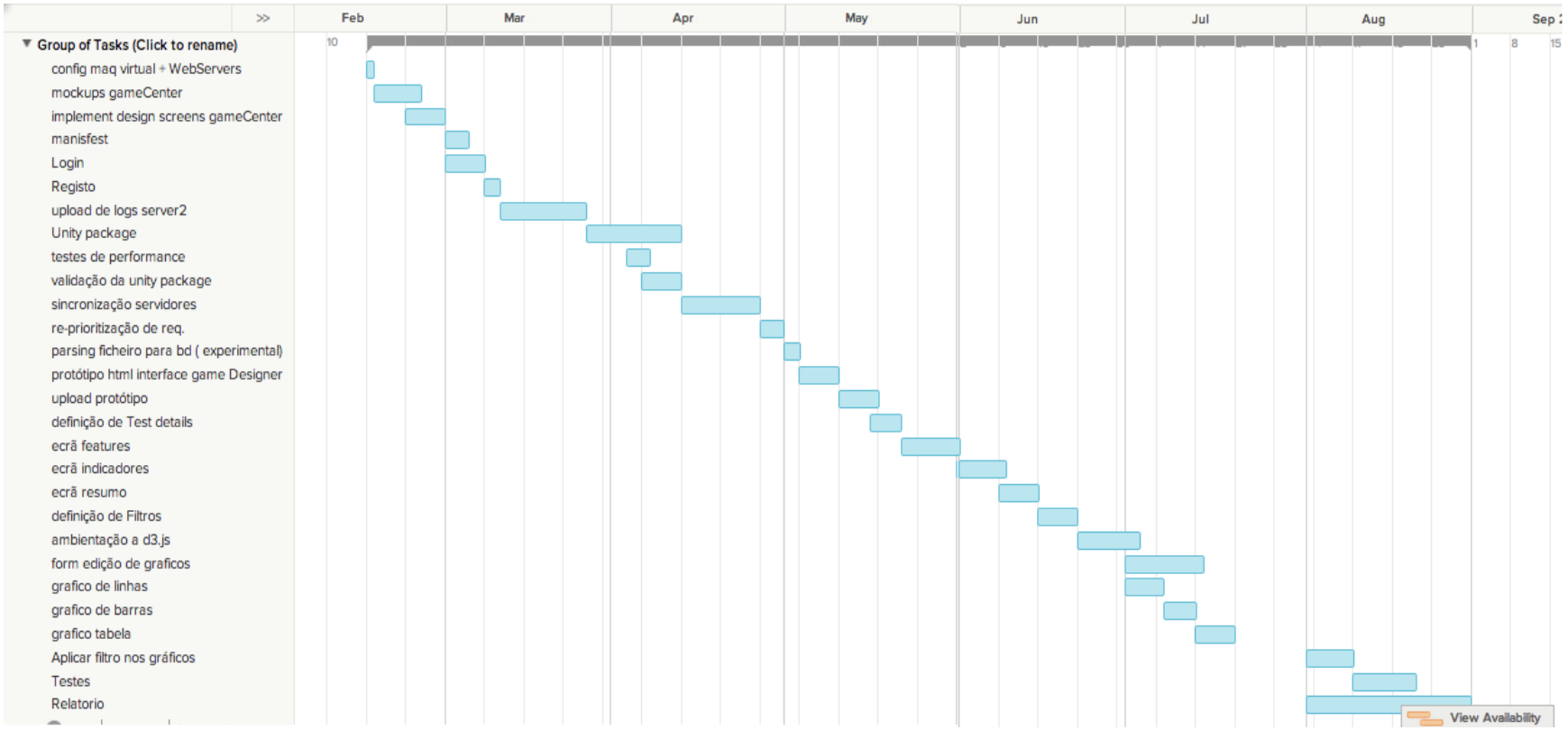

Figure 10 – plano de actividades real (2º semestre)

Capítulo 4
Design de Interfaces

Como referido anteriormente, existem dois grupos distintos de utilizadores sem os quais esta plataforma, nos moldes apresentados, não faria sentido. São eles *game designers,* responsáveis por inserir novos protótipos e definições associadas aos mesmos, e os utilizadores que se pretendem divertir jogando esses mesmos protótipos. Necessariamente terão de existir duas interfaces distintas, uma para cada um dos grupos de utilizadores da plataforma *Crowdplay*. Os seguintes subcapítulos pretendem explicar todo o trabalho de design que antecedeu o desenvolvimento de cada uma das interfaces.

4.1. Interface Android para o jogador (*GameCenter*)

Tratando-se de uma versão inicial, o design desta interface foi deixado a meu cargo. Após algumas reuniões com os *stakeholders* foi me apresentado alguns diagramas de casos de uso e de estado envolvendo esta interface (disponíveis em anexo). Com essa informação e após algumas experiências desenvolvi, recorrendo à ferramenta *Balsamiq,* os *mockups* dos ecrãs necessários. De forma a simular uma experiência o mais realista possível, com recurso à mesma, ferramenta defini as interacções possíveis em cada ecrã. Os *mockups* foram aprovados e como tal seguiu-se a fase de desenvolvimento.

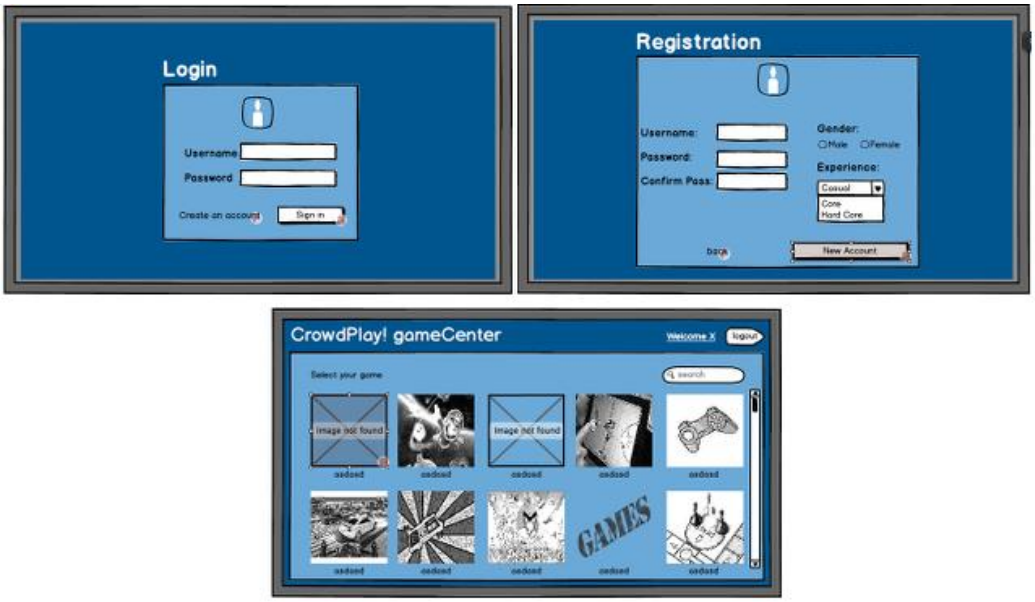

Figure 11 - Mockup gameCenter

4.2. Projecto de Interface Web para o Game Designer

Com o intuito de perceber as necessidades dos utilizadores desta interface, foram feitas três sessões de prototipagem colaborativa. Destas sessões resultaram um conjunto de ideias e desenhos para a interface a ser usada pelo *game designer* e grande parte dos fluxos de interação, etapas de criação e edição de projetos.

O conceito principal desta abordagem é envolver todos os *stakeholders* no processo de design de forma a tentar perceber as necessidades e assim construir um produto à imagem dos seus potenciais utilizadores.

Como tal, o grupo de participantes foi constituído por vários elementos que foram ou ainda são parte integrante do laboratório. Muitos dos quais desenvolvem jogos em *part-time* ou *full-time*. Dada a diversidade de participantes, foi conseguido o objetivo de simular as necessidades de estúdios de diferentes dimensões.

Todo o processo foi documentado através da gravação áudio das reuniões e ainda com a recolha de todos os *mockups* que foram sendo criados e / ou alterados.

O resultado final foi a junção de todas as opiniões e a respetiva construção de um *mockup* principal, e a definição de cada um dos seus elementos constituintes bem como das acções a suportar.

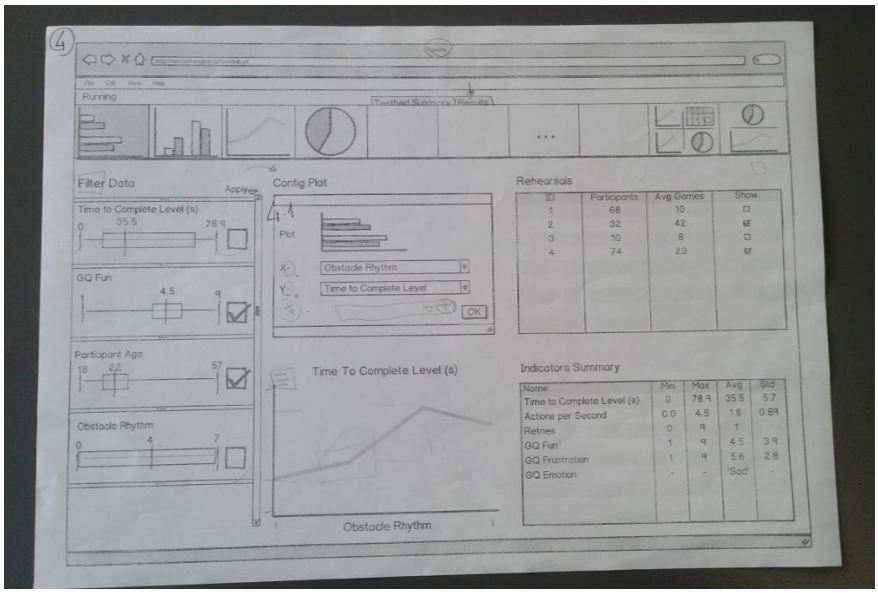

Figure 12 - Mockup de métricas e indicadores

Não tendo sido organizada por mim, o meu papel nesta tarefa foi o mesmo que qualquer um dos outros elementos convidados para participar no design. Apesar de fazer parte do projecto as sugestões dadas não foram condicionadas por qualquer questão prática de desenvolvimento. Desta forma o foco principal da minha participação foi tornar a interface o mais eficiente e *user-friendly* possível.

A proposta inicial foi desenvolvida por dois *game designers*. Foi-lhes pedido que desenhassem uma interface de acordo com as suas necessidades. De seguida foram realizadas 2 sessões de

prototipagem colaborativa, onde participaram 8 pessoas (dando origem a 2 grupos de 4 pessoas). Sendo que a primeira foi realizada antes do início do meu estágio. A interface resultante da primeira sessão foi melhorada durante segunda sessão. A interface actual é o resultado da compilação dos dados recolhidos na última sessão.

Capítulo 5
Definição de Arquitetura

Ao longo de todo este capítulo será apresentada a arquitetura da plataforma *Crowdplay*. Para tornar esta informação mais explícita, irá ser apresentada uma visão geral da plataforma e posteriormente irá ser apresentada uma visão mais detalhada de cada uma das partes.

Inicialmente a proposta era para ser limitada a um servidor (explorando o modelo de actores), sendo ele responsável por receber e distribuir toda a informação por entre as duas interfaces.

De forma a tornar a solução potencialmente mais escalável e tendo sido tomado em consideração os requisitos dos stakeholders, o peso do servidor único foi distribuído por dois. Um dos servidores (WebServer2) será responsável por comunicar com os clientes das interfaces para o *tablet* e o armazenamento em bruto dos dados recolhidos. O outro servidor (WebServer1) terá a seu cargo o *parsing* , armazenamento de dados processados e em bruto, armazenamento dos projetos e ainda será responsável por de toda a comunicação com os clientes da interface do *game designer*.

A troca de dados entre os dois servidores foi um requisito que baixou de prioridade dado o seu grau de complexidade e consequente exigência temporal. Apesar disso foi investido algum tempo nesta tarefa e por isso mesmo serão apresentadas duas possíveis soluções no respectivo subcapítulo.

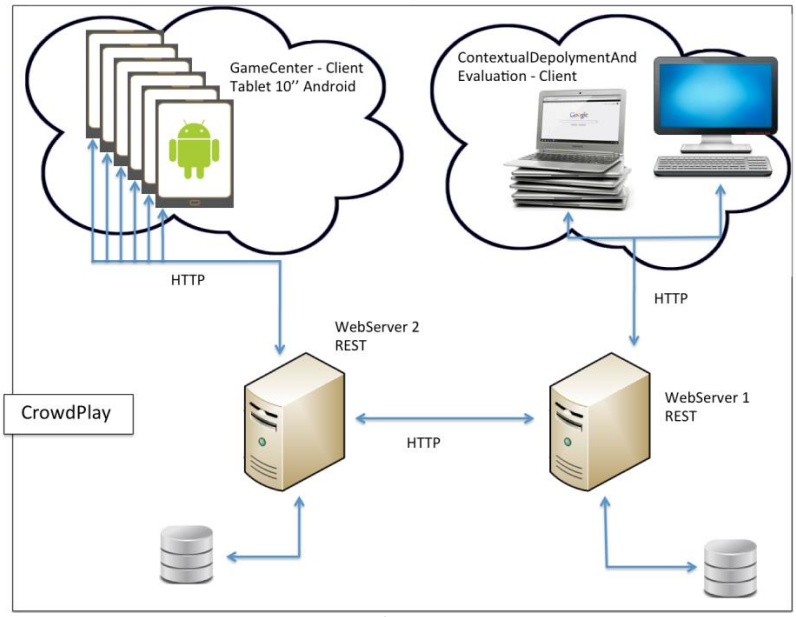

Figure 13 - Arquitetura nível 0

5.1. Unity package

Para uniformizar os processos que envolvem a comunicação com o servidor e a gestão de dados dos protótipos, foi criado um pacote unity (*crowdplay.unitypackage*) que contém um objecto (*prefab*) completamente definido e pronto a ser instanciado no jogo. Este objecto é denominado *CrowdplayPrefab* e faz uso da biblioteca *CrowdplayLib*.

Na imagem seguinte, a azul estão ilustradas 3 classes com os métodos genéricos. Na classe *FileHandler* estão todos os métodos de gestão de ficheiros, na classe *SessionHandler* estão os métodos que permitem fazer a gestão das sessões (*play/run session*) e na classe *FileUpload* estão, por sua vez, os métodos que permitem enviar os ficheiros de log para o servidor. A verde está ilustrada a classe principal que utiliza as classes anteriormente mencionadas para definir os métodos necessários na integração com o protótipo.

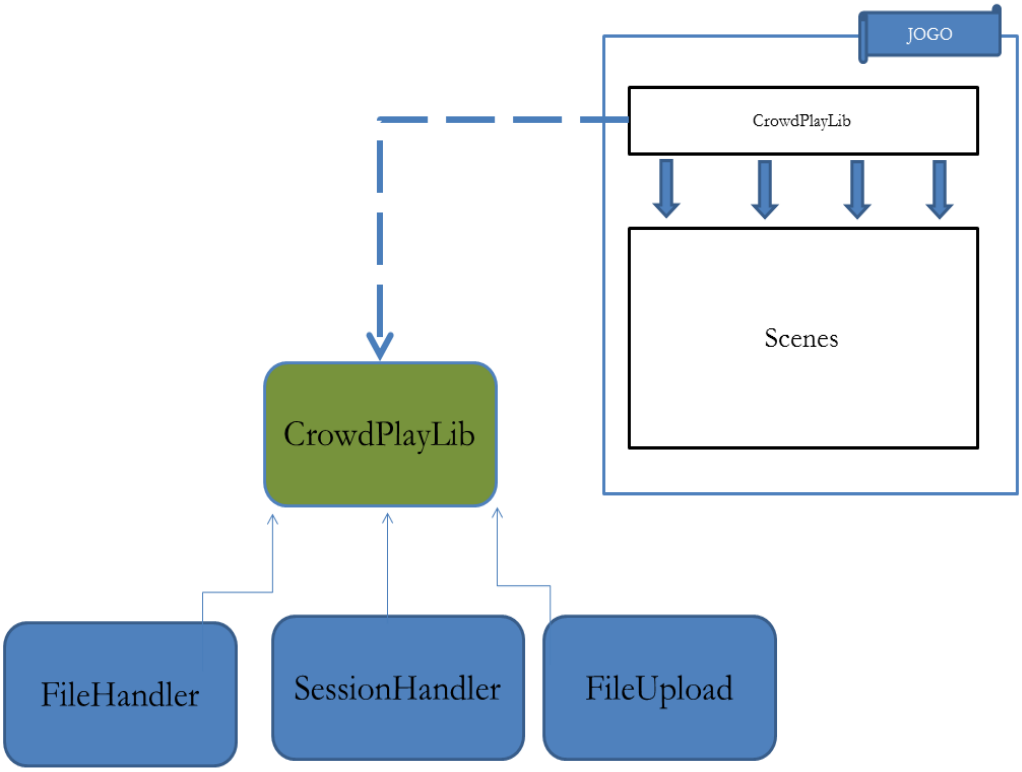

Figure 14 – Biblioteca Crowdplay

Após importar para o seu projecto o pacote fornecido e depois da instanciação do *CrowdplayPrefab*, o *game designer* terá acesso a um conjunto de métodos que terá de integrar no seu protótipo. Esta integração foi pensada para ser simples e intuitiva para qualquer pessoa. A sua construção foi um processo iterativo de desenvolvimento e avaliação até ao resultado final que é o que se apresenta neste relatório. Os métodos disponíveis são:

- Métodos de Registo de actividade de jogador (*upload* implícito)
    - startPlaySession()

    - endPlaySession()
    - endPlaySession()
    - startRunSession(),
    - endRunSession()
    - startLogFile()
    - saveLog()
- Métodos de definição de Casos de teste
    - getTest()
    - getTestVariable()

A integração dos métodos de gestão de actividade do jogador são obrigatórios, ou não faria sentido utilizar a plataforma *Crowdplay*. No caso do *game designer* apenas pretender um único caso de teste, poderá ser ele próprio a defini-lo no protótipo e os métodos de definição de casos de teste, anteriormente indicados, não serão necessários. Se esta condição não se verificar, para o bom funcionamento do protótipo, estes métodos deixam de ter carácter opcional. O método getTest() necessita que o ficheiro de *Feature Sets* tenha sido guardado no dispositivo móvel. Esta acção é realizada pelo jogador na mesma altura em que faz o *download* do protótipo.

Cada instância de um *log* corresponde a um conjunto de fixo de dados previamente definidos pelos *stakeholders*:

- time
- engine-time
- id_level
- id_gameState
- x,y,z
- id_event
- id_gameObject
- id_subject

É aconselhado que o jogo tenha um menu, pelo menos no final, que permita reiniciar uma jogada sem que para tal seja necessário sair do mesmo.

5.2. Base de dados

Este subcapítulo pretende apresentar o design da base de dados e explicar as razões de tais escolhas.

A base de dados definida no webServer2 (responsável pela comunicação com o gameCenter) é apresentada na imagem que se segue. Aqui são guardadas informações dos jogadores (tabela *playTesters*), a listagem dos jogos existentes (tabela *projects*) e para cada entrada nesta tabela existe uma correspondente na tabela de *test* onde estão guardadas as condições de cada teste ou experiência.

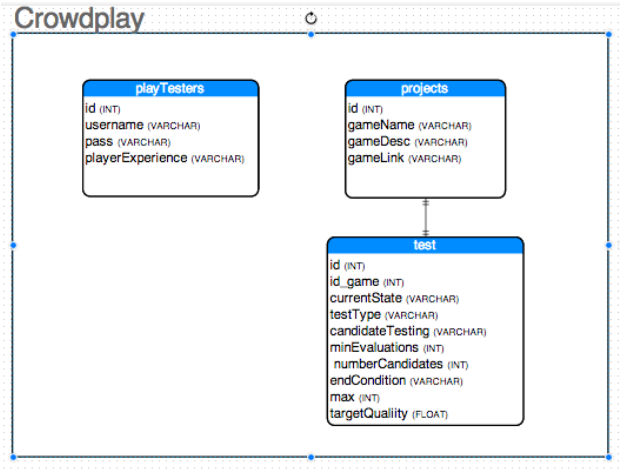

Figure 15 – Diagrama db webServer2

No webServer1 existem dois tipos de base dados. A primeira, apresentada na imagem imediatamente a seguir, contém duas tabelas. A tabela *gameDesigner* é utilizada para guardar toda a informação relevante dos *game designers* e a tabela *projects* guarda a informação dos projectos. Um *game designer* pode ter zero ou vários projectos associados.

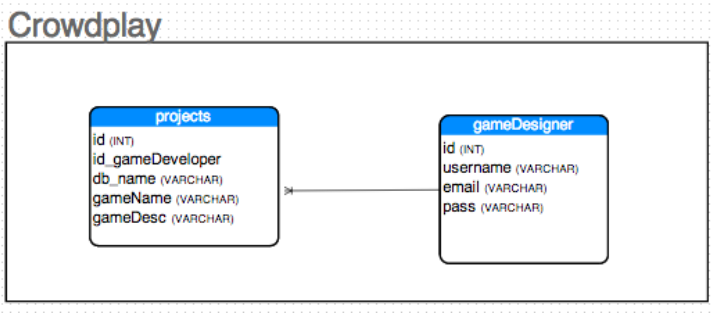

Figure 16 – Diagrama db principal do webServer1

A imagem que se segue mostra a estrutura da base de dados de cada um dos projectos. Assim que um projecto é iniciado, é adicionada uma nova base de dados. Esta estrutura foi pensada para isolar os dados relativos a cada projecto. Os principais objectivos para esta decisão foram potencialmente melhorar a performance de escrita e leitura de dados e evitar varrimentos recursivos para a remoção de dados de um projecto.

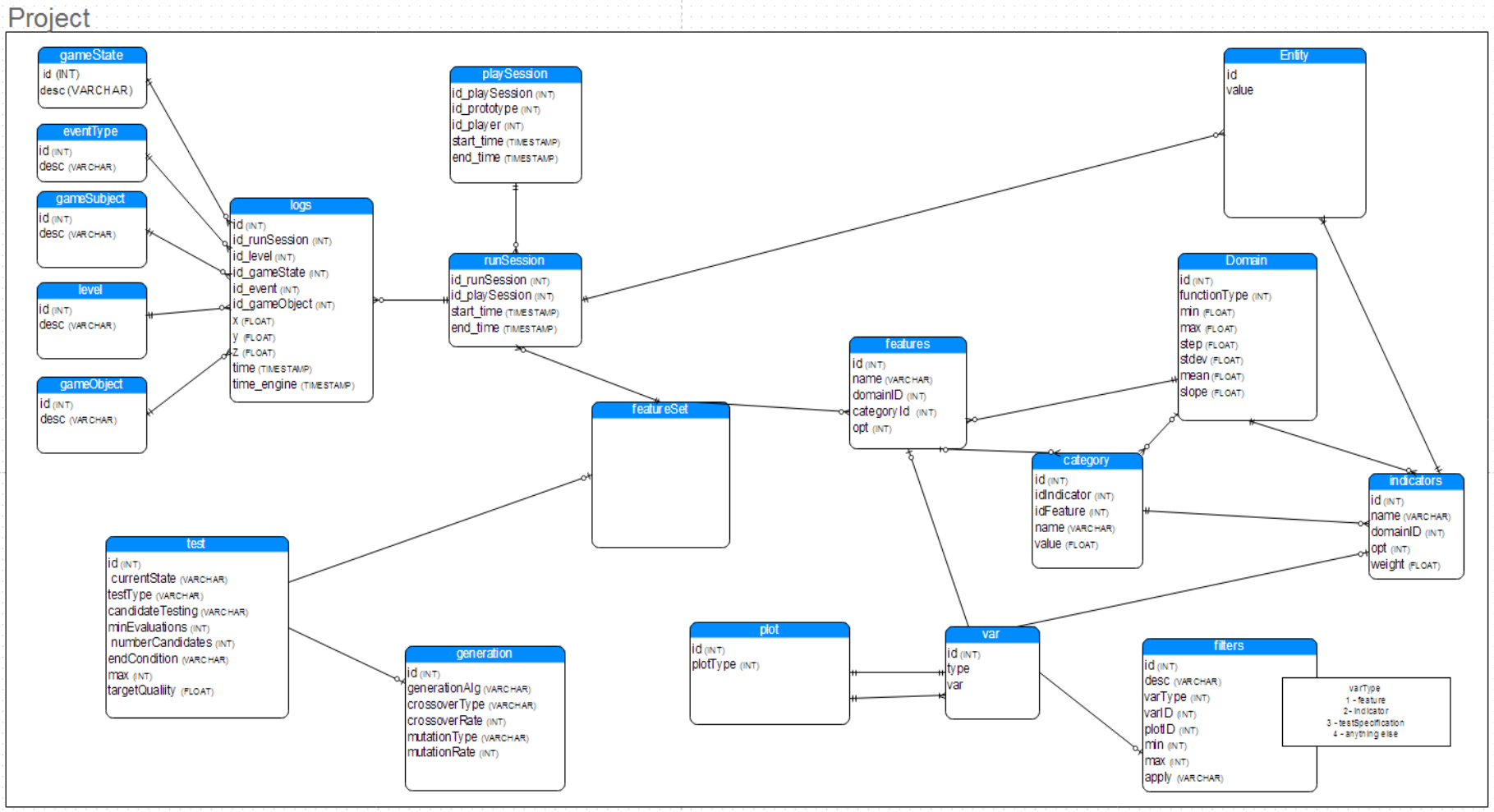

Figure 17 – Diagrama db de cada projecto

5.3. Interações entre Game Designer's Interface e webServer1

Isolando esta parte da plataforma é possível identificar uma arquitectura MVC (*Model – View – Controler*) tendo sido desenvolvido com recurso à *Play! Framework*.

Como é possível verificar na figura a baixo, todos os módulos disponíveis na interface têm um controlador que satisfaz as suas necessidades.

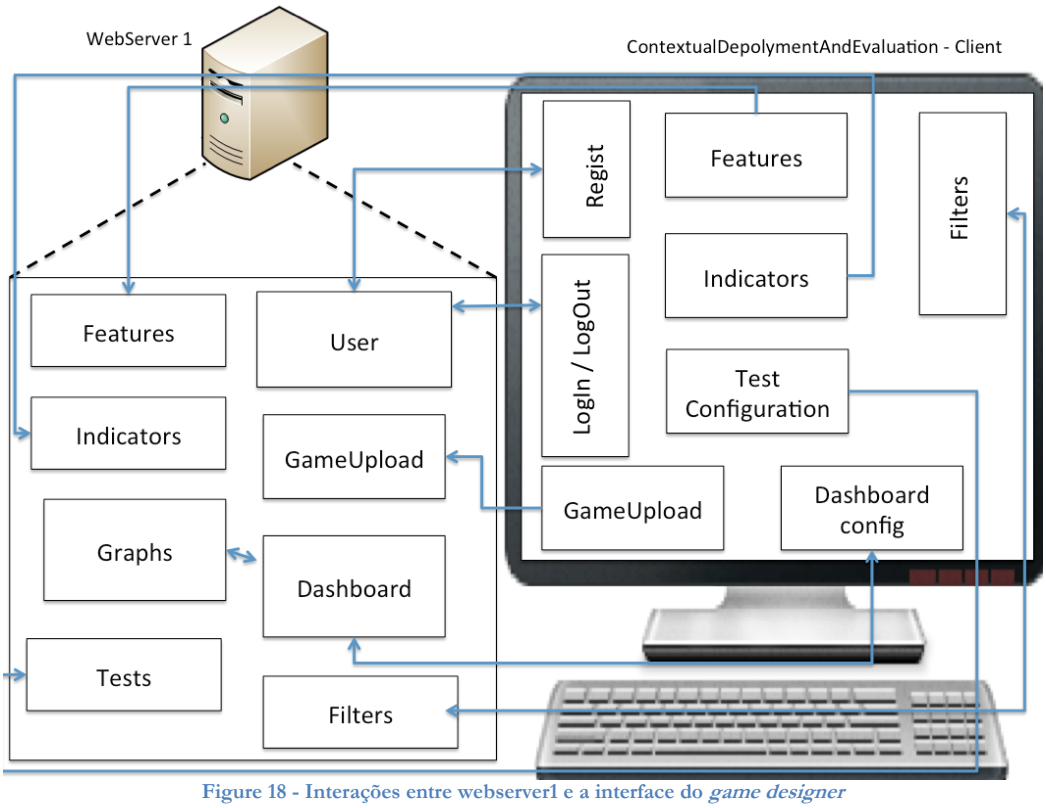

Figure 18 - Interações entre webserver1 e a interface do *game designer*

Estando os protótipos associados a um *game designer*, é de esperar que exista um módulo responsável pela identificação do utilizador (*login* e registo). Depois de aceder à sua conta, o *game designer* terá duas opções: iniciar um novo projecto ou carregar um previamente criado por si. A partir deste momento o utilizador terá à sua disposição as seguintes opções:

- Ver o resumo geral do projecto
- Adicionar / remover / editar *features*
- Adicionar / remover / editar *Indicators*
- Definir / editar *Feature set generation conditions*
- Definir / editar *Test Conditions*
- Definir / editar / remover *Filters*
- Definir / editar *graphs*

São de referir algumas dependências de conceitos da plataforma:

- A geração de *feature sets* só pode acontecer caso existam *features* definidas

- Só é possível iniciar um teste quando existe um protótipo associado ao projecto
- Para definir um gráfico é necessário existir pelo menos duas *features* ou *indicators* (ou um de cada)
- Para definir um filtro é necessário existir pelo menos uma *feature* ou *indicator*
- Não é possível remover uma *feature* ou um *indicator* caso estejam representados num dos gráficos definidos

5.4. Interações entre GameCenter e webServer2

A aplicação *gameCenter* foi desenvolvida em Unity e comunica com o webServer2 (que foi criado também com a *Play! Framework*) segundo o protocolo http.

Novamente na seguinte figura é possível verificar que todos os módulos disponíveis na interface têm um controlador correspondente.

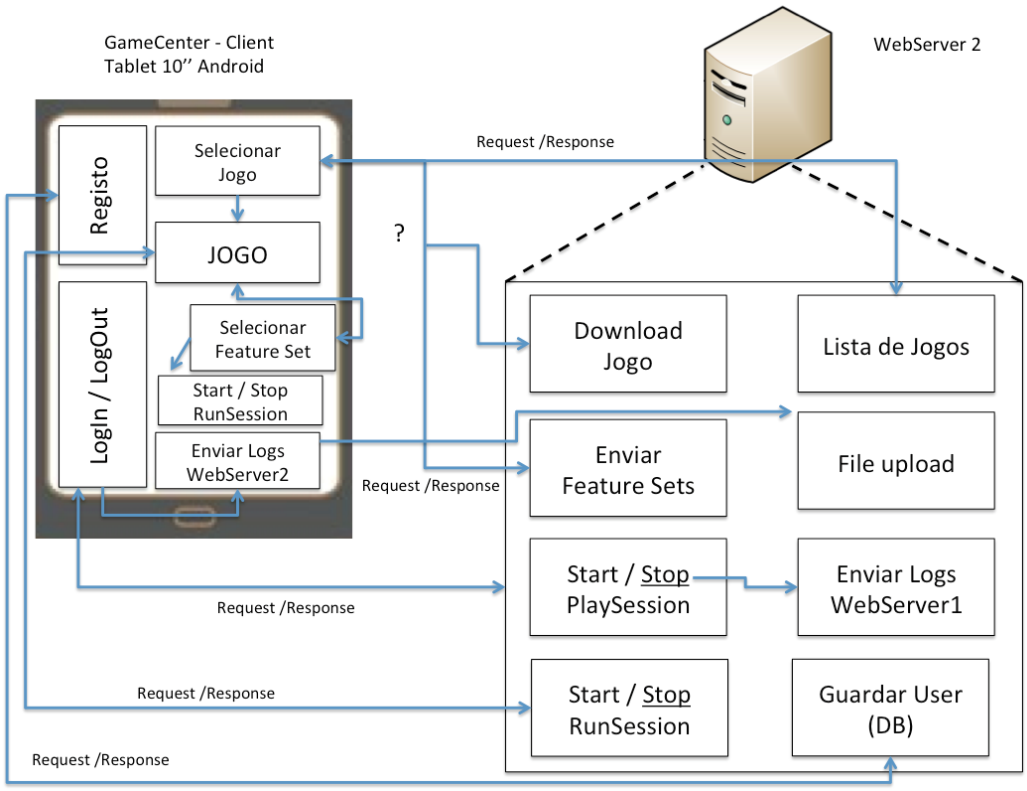

Figure 19 - Interações gamecenter e webserver2

Tal com a outra interface, o *gameCenter* tem um serviço de *login* para identificar o utilizador que em cada momento está a jogar um determinado protótipo. Para ter acesso aos protótipos, todos os jogadores terão que efectuar um registo. Assumindo que a listagem de jogos está actualizada, segundo a explicação feita no subcapítulo seguinte, após efectuar o *login* o utilizador irá encontrar uma lista com todas a opções disponíveis no momento. Ao seleccionar um jogo serão descarregados os ficheiros com os casos de teste e de instalação

do protótipo. Quando se inicia o jogo, o servidor é notificado e inicia uma *play session.* O caso de teste é escolhido automaticamente de entre as opções disponíveis no ficheiro de *feature sets,* entretanto já no dispositivo móvel. O caso de teste escolhido é em cada momento o menos utilizado dentro das opções existentes do ficheiro. Uma *run session* está associada uma *play session,* e é iniciada depois de o caso de teste ter sido escolhido (sendo o servidor novamente notificado) e terminada quando o protótipo chega ao fim. Ainda no menu de jogo o utilizador poderá reiniciar o jogo, sendo escolhido um novo caso de teste e iniciado uma nova *run session.*

À medida que o utilizador vai jogando o protótipo serão recolhidas várias instâncias de dados e guardadas em disco no dispositivo móvel usado. Os ficheiros de *log* só serão enviados para o servidor quando o utilizador fizer *logout*. Caso este não tenha uma conexão à internet ou termine abruptamente a aplicação *gameCenter* os dados serão enviados quando este voltar a fazer *login*.

5.5. Interações entre webServer1 e webServer2

Como foi referido anteriormente, a troca de dados entre os dois servidores foi um requisito que baixou de prioridade dado o seu grau de complexidade e consequente exigência temporal.

Foram encontradas duas soluções diferentes para resolver esta questão.

Uma solução seria utilizar um serviço de *cloud storage* (*zoolz*, *OpenDrive*, *justCloud,* entre outras), atribuindo assim todas as questões relacionadas com performance e segurança à entidade que forneceria o serviço. Com esta solução todos os dados relevantes neste caso (ficheiros de log, *feature sets*, ficheiros de instalação protótipos) seriam directamente guardados na *cloud* e ficariam acessíveis a qualquer um dos dispositivos.

Outra solução seria manter a ideia inicial, ou seja, recorrer a webSockets. Desta forma, o webServer2 não teria acesso aos ficheiros de instalação dos protótipos nem aos ficheiros para onde são exportados os Feature Sets, mas antes a uma listagem contendo uma descrição e um *link* público pertencente ao webServer1 (e.g. http://anaplay.dei.uc.pt/nomePrototipo). Por sua vez os ficheiros de *logs* guardados em disco no webServer2 continuariam a ser enviados para o webServer1 onde mais tarde seria feito o parsing e inserção na base de dados.

Capítulo 6
Trabalho de Desenvolvimento

Ao longo deste capítulo será apresentado, de forma resumida, todo o trabalho que até então fora realizado. Este período está compreendido entre Setembro de 2013, mais precisamente dia 16 (dia da primeira reunião no labc62) e o dia 2 de Setembro de 2014, data da entrega do relatório final.

Em cada um dos seguinte subcapítulos estará um breve resumo das tarefa realizadas.

6.1. Cronologia de actividades realizadas

Como é possível verificar existiram alguns atrasos e tomadas de decisões que influenciaram a duração do projecto. Inicialmente a previsão era para terminar a fase de desenvolvimento no final de Maio para conseguir resolver alguns bugs e escrever o relatório durante o mês de Junho permitindo assim a defesa no mês seguinte. Como no final desse período ainda não era possível identificar um fluxo completo de interacções da plataforma *Crowdplay*, foi decidido aumentar o período de desenvolvimento e adiar a defesa para o mês de Setembro.

6.1.1. Fevereiro – interação 1

Este período é iniciado após a defesa intermédia e termina no final do mês de Fevereiro. Durante este tempo o planeamento foi cumprido com sucesso e todas as actividades foram realizadas, não havendo atrasos. As tarefas realizadas foram:

- Fecho de requisitos
- Definição da arquitectura
- Criação e configuração dos servidores nas máquinas virtuais
- Criação de *mockups* do *gameCenter*
- Implementação das screens do *gameCenter* (front-end apenas)

6.1.2. Março – interação 2

Durante este período foram registados atrasos nas tarefas de desenvolvimento da aplicação móvel que envolviam a troca de dados com o servidor. Entre elas no *upload* dos ficheiros de *log* no webServer2 e pedidos de *login* e resgisto. Estas dificuldades deveram-se ao facto de todos as rotinas terem que ser construídas de raiz e mesmo havendo várias sugestões, por vezes eram contraditórias. A solução passou por utilizar a classe WWW do Unity para enviar e receber mensagens Json.

Para além desta questão havia outra relacionada com permissões de acesso à memória do telemóvel que foram resolvidas com o desenvolvimento de um *AndroidManifest.xml* (ficheiro com definições essenciais da aplicação utilizado pelo sistema operativo). Ao compilar a aplicação o Unity criava um *manifest* por defeito. Foi então necessário desenvolver um mais específico que permitisse tais operações.

Foi ainda iniciada a tarefa de criação do pacote de Unity (cap.5.1)

As tarefas realizadas foram:

- *Login* de utilizadores no gameCenter
- Registo de utilizadores no gameCenter
- Gravação e acesso de dados no dispositivo móvel por parte da aplicação
- Upload de dados no webServer2
- Início do *crowdplay.unitypackage*

6.1.3. Abril – interação 3

Como não foi possível iniciar as tarefas relativas à interface do game designer que estavam programadas para o mês anterior, foram acumuladas às programadas para este período.

O desenvolvimento do *crowdplay.unitypackage* foi efectuado num período de 2 dias, tendo sido, findo esse tempo, dado inicio a um ciclo de avaliação previsto no planeamento.

Enquanto se realizavam testes de performance do webServer2, o Rui Craveirinha integrou o *unity package* com um protótipo que havia sido feito no ano anterior por um elemento do laboratório ("*Run Ines, Run!*"). Desta integração surgiram algumas questões relacionadas com a utilização da biblioteca e ao tipo de dados recolhidos. Depois de todas as correcções feitas e da actualização do protótipo foi simulada uma experiência com vários colaboradores.

A tarefa de sincronização de dados foi iniciada e terminou ao fim de duas semanas (final do mês de Abril). Durante este tempo foram estudadas algumas formas de resolver esta situação. Quando apresentadas aos *stakeholders* e dado os atrasos que já se tinham verificado no planeamento foi decidido que este grupo de tarefas seria descartado, tendo a sua prioridade sido diminuída.

As tarefas realizadas foram:

- Terminar *crowdplay.unitypackage*
- Teste de performance do webServer2
- Validação do *crowdplay.unitypackage*
- Resolução de problemas encontrados
- Inicio da sincronização de dados

6.1.4. Maio – interação 4

Apesar de durante este mês o principal foco, no planeamento inicial, ser a implementação do *dashboard* estas tarefas foram adiadas. Os requisitos foram novamente avaliados e prioritizados.

Nos primeiros quatro dias do mês foi feito um protótipo rápido (em html) do que seria a interface do *game designer*. Após a sua validação junto dos *stakeholders* foi dado início às tarefas que se encontravam atrasadas.

As tarefas de desenvolvimento realizadas foram:

- Definição de uma classe de *parsing* dos ficheiros de log e inserção na base de dados (versão experimental)
- Protótipo rápido da interface para o *game designer*
- *Upload* de protótipo
- Tarefas associadas a *Test details*
- Tarefas associadas a *features*

6.1.5. Junho – interação 5

Durante este período foi dada continuidade às tarefas realizadas no mês anterior, tendo sido realizadas as seguintes tarefas:

- Tarefas associadas a *Indicators*
- Tarefas associadas ao resumo do teste/experiencia
- Definição simples de filtros (sem implicação directa com os gráficos, ainda n desenvolvidos)
- Estudo da biblioteca d3.js

6.1.6. Julho – interação 6

Este período de trabalho resume-se às três primeiras semanas do mês.

Foram realizadas as seguinte tarefas:

- Definição de form de edição de gráficos
- Criação de gráfico de linhas
- Criação de gráfico de barras
- Criação de tabelas

6.1.7. Agosto – interação 7

Durante o mês de Agosto foi realizada a ultima tarefa de desenvolvimento definida com prioridade máxima, aplicação de filtros globais no gráfico de linhas (tendo este sido estendido aos gráficos de barras).

Terminada esta tarefa foram realizados testes de usabilidade que permitiram avaliar a interface. Na parte final deste período foi dado enfâse ao relatório final.

6.2. Estado Final

Para permitir uma melhor visualização de todas as actividades desenvolvidas é apresentado em anexo a lista de requisitos da plataforma. Para além das prioridades são também identificados quais os requisitos concluídos.

Está ainda disponível em anexo um conjunto de imagens que apresentam as duas interfaces, *gameCenter* e interface para os *game designers*.

Capítulo 7
Testes

Ao longo deste capítulo serão apresentadas os testes realizados durante as duas fases de avaliação (Abril e Agosto). Será feita uma breve introdução aos métodos escolhidos e de seguida serão apresentados os resultados obtidos.

7.1. Planificação

Foram realizados dois tipos de teste: teste de performance e testes de usabilidade.

7.1.1. Testes no webServer2

Um dos requisitos do webServer2 era conseguir responder eficientemente a um mínimo de cem utilizadores por minuto. Para validar este requisito foram feitos teste de performance. Este tipo de testes é realizado com o intuito de avaliar a capacidade de resposta e a estabilidade de um determinado produto quando submetido a uma carga específica.

Para a elaboração deste teste foi tido em conta uma metodologia definida em 2007 para testes de performance. (Microsoft, 2007)

De uma forma geral para a execução de um teste de performance, segundo esta metodologia, são necessários 7 passos:

1. Identificar o ambiente de teste.
   Conhecer o ambiente físico e os recursos disponíveis durante o teste

2. Identificar os critérios de aceitação
   Conhecer os critérios que definem o sucesso ou o insucesso do projecto

3. Definir plano de teste
   Identificar os principais cenários e dados a ser recolhidos

4. Configurar ambiente de teste
   Preparar o ambiente de teste de forma a conseguir realizar todas as experiências definidas

5. Implementar o teste
   Desenvolver os testes de acordo o planeamento

6. Executar o teste
   Executar e acompanhar a execução do teste

7. Analisar os resultados obtidos e repetir o teste
   Recolher e analisar os dados, repetir

Os testes foram realizados dentro da rede DEI e máquina que contém o servidor tem 2GB de RAM, 25GB de disco e um processador com 2 núcleos. A máquina responsável por simular os pedidos tem 10Gb de RAM, 500Gb de disco e um processador com 4 núcleos.

Foram definidas duas situações práticas e para cada, 3 testes diferentes. Sendo assim temos:

- Login
  - 10 * 100 request sequenciais
  - 50 * 100 request sequenciais
  - 100 * 100 request sequenciais

- Upload de ficheiro (43kb - dimensão média dos ficheiros de log até então registados no jogo "Run Ines, Run!")
  - 10 * 100 request sequenciais
  - 50 * 100 request sequenciais
  - 100 * 100 request sequenciais

.

7.1.2. Testes no Interface para game designers

Durante o *design* da interface um dos principais objectivos sempre foi torná-la o mais *user-friendly* possível.

Para validar esta interface foram realizados testes de usabilidade com vários utilizadores. Um ponto comum entre todos eles era a sua formação curricular base, Engenharia Informática. Apesar da interface ter sido definida para *game designers* o grupo não era constituído estritamente por elementos desta área, ou o número de participantes no teste seria menor. No entanto, antes de começar o teste foi dada uma breve explicação de forma a ambientar todos os intervenientes.

O teste consistiu na realização de um cenário de uso da interface, previamente definido, num ambiente controlado e onde todas as actividades eram monitorizadas e registadas para posterior análise. Em baixo é mostrado o cenário apresentado:

O objectivo deste exercício é realizar um teste procedural do jogo "*Run Ines, Run!*".

Tarefas:

1. Criar um novo projeto
2. Fazer upload do protótipo "Run Ines, Run"
3. Definir um teste procedural com um numero mínimo de 20 avaliações para 10 candidatos
4. Adicionar uma feature denominada "Número de inimigos gerados por minutos", do tipo INT. Esta variável deve ser passível de ser optimizada e deve variar segundo uma curva gaussiana e precisão de 1
5. Adicionar uma feature denominada "Ritmo de aparecimento de Power-ups", do tipo FLOAT, inserindo uma descrição apropriada. A variável deve estar definida com precisão de 0.05. Deve ser passível de ser optimizada e deve variar segundo uma curva gaussiana
6. Adicionar um indicador denominado "Tempo de Gameplay", do tipo FLOAT. Este indicador deve ser optimizado segundo uma função Gaussiana e com um peso de 0.5.

7. Adicionar um indicador denominado "Número de power-ups colhidos", do tipo INT. Este indicador deve ser optimizado segundo uma função Gaussiana e com um peso de 0.5.
8. Definir o teste procedural como Algoritmo Genético, com selecção Torneio, Cross-over tipo 2 pontos a 90%, Mutação 1 gene a 5%
9. Estabelecer como condição de fim Manual End
10. Fazer deploy do teste
11. Ir ao Dashboard e definir uma tabela com os dois indicadores
12. Definir um gráfico de linhas com "Ritmo de aparecimento de Power-ups" no eixo dos XX e o "Número de power-ups colhidos" no eixo dos YY
13. Definir um gráfico de barras com os mesmos eixos enumerados anteriormente
14. Criar um filtro "Número de power-ups colhidos" com um valor minimos de 5 e máximo de 60 aplicando-o ao dashboard já definido
15. Eliminar o filtro "Número de power-ups colhidos"
16. Eliminar o indicador "Tempo de Gameplay"
17. Eliminar a feature"Número de inimigos gerados por minutos"
18. Fazer deploy do teste novamente

7.1.3. Testes no gameCenter

Tal como realizado na validação da interface anterior, para o *gameCenter* foi utilizado o mesmo método, ou seja, foram realizados testes de usabilidade. Neste caso, dado que o tipo de utilizadores não tem um perfil específico, não foram colocadas restrições na escolha da amostra.

Já com a aplicação aberta no dispositivo móvel, esta foi a lista de tarefas apresentadas:

1. Fazer o registo de um utilizador definindo um *username* e uma *password*, experiência de jogador e o seu género
2. Voltar ao ecrã de login
3. Fazer login com o utilizador anteriormente criado
4. Escolher o jogo "Run Ines, Run!"
5. Fazer download do jogo
6. Fazer download dos casos de teste
7. Instalar e iniciar o jogo

7.2. Dados Recolhidos

7.2.1. Testes no webServer2

Em ambos os casos foram recolhidos o tempo total de execução de um teste, tempo de execução de cada pedido (engloba todo o período desde envio da mensagem até ao momento da recepção da resposta do servidor) e ainda o tempo médio de cada pedido. O objectivo mínimo era superar os 100 pedidos por minuto.

De forma a evitar problemas alheios à plataforma foram feitos várias execuções de cada teste em diferentes horários do dia e descartados os testes cujos valores se dispersavam dos até então encontrados.

7.2.2. Interface para game designers

Como foi referido anteriormente, o cenário apresentado aos indivíduos que compunham a amostra estava previamente definido e as todas as suas actividades foram controladas.

Durante cada teste foi monitorizado o tempo de execução de cada tarefa, dúvidas / dificuldades apresentadas e ocorrências inesperadas (independentemente da causa ser mau funcionamento do utilizador ou da interface). No final foi pedido a todos os participantes uma lista de factores positivos e negativos experienciados.

Figure 20 – Teste de usabilidade

7.2.3. Testes no gameCenter

Uma vez que o conceito de teste é igual ao apresentado anteriormente, os dados recolhidos foram os mesmos.

7.3. Análise de resultados

7.3.1. Testes no webServer2

Nas tabelas em baixo apresentadas estão os valores recolhidos durante os testes realizados.

Login:

	10 * 100 pedidos	50 * 100 pedidos	100 *100 pedidos
Exec. Total (s)	3,43	13,28	27,75
Duração média (ms)	9,90	17,75	19,02
Pedidos analisados/s	291	376	360

Upload:

	10 * 100 pedidos	50 * 100 pedidos	100 *100 pedidos
Exec. Total (s)	11,98	49,44	124,06
Duração média (ms)	30,95	60,55	79,45
Pedidos analisados/s	83	101	80

Como é possível verificar nas tabelas em qualquer um dos casos foram analisados mais do que 100 pedidos por minuto, tendo sido então superado o requisito imposto no início do projecto. Em média, para o caso do *login*, foram analisados 342 pedidos por segundo o que perfaz um total de 20520 pedidos por minuto. Já nos testes de *upload* foi possível responder em média a 88 pedidos por segundo o que perfaz um total de 5280 pedidos por minuto.

Os cenários apresentados evidenciam as duas situações computacionais mais frequentes no servidor (i.e. acesso à base de dados e transferência de ficheiros). A situação de registo não foi testada, no entanto, apesar da escrita ser potencialmente mais demorada do que a leitura, os valores recolhidos sugerem claramente que o requisito foi cumprido. Os cenários de troca de dados entre dos dois servidores terão que ser testados assim que essas ligações estiverem concluídas.

Em anexo é podem ser consultados os gráficos que apresentam os valores de cada um dos testes anteriormente apresentados.

7.3.2. Testes no Interface para game designers

Na tabela apresentada em baixo é visível o tempo médio de execução, em segundos, de cada tarefa. O cronómetro foi iniciado depois de o utilizador ler a tarefa e terminado no final da sua realização.

Deste modo, em média, cada utilizador precisou de cerca de 4 minutos e 30 segundos para concluir o teste.

Tarefa	Tempo de Exec (s)
1	6,05
2	15,12
3	8,06
4	31,63
5	37,63
6	17,91
7	18,87
8	42,56
9	13,59
10	7,62
11	18,92
12	12,41
13	10,53
14	14,49
15	3,25
16	4,25
17	3,73
18	3,90

Ao longo de cada teste foram identificados alguns erros de funcionamento da interface. A tabela que se segue apresenta os erros identificados e a percentagem de pessoas que o detectaram. Esta informação é bastante relevante uma vez que permite corrigir problemas que até então não tinham sido detectados. As respostas que não se enquadravam na dentro da categoria de erros foram adicionadas às opiniões.

Bugs encontrados	Nº de ocorrência
Depois do upload não existe redireccionamento para home	5
Na definição do filtro, label "apply" duplicado	2
No ecrã "Features", menu situado na parte inferior da listagem não tem acções associadas	3

No ecrã "Test", select box "End Condition" não tem a opção "Manual End" (só é visivel no estado default, caso se escolha outra opção, esta não pode voltar a ser escolhida)	1

De modo a tentar aproximar a interface dos seus utilizadores foi pedido que cada um deles, no final do teste pudesse explorar a interface à vontade, sem limites temporais e posteriormente elaborasse uma lista com factores positivos e negativos. Na tabela que se segue estão registadas as opiniões apresentadas. Foram ainda contabilizadas as percentagens de ocorrências de cada uma.

Opinião	Positiva	Negativa	Nº de ocorrência
Layout Intuitivo	X		3
Layout responsivo	X		1
Boa organização	X		4
Adicionar barra de progresso no upload		X	4
Adicionar título na secção de geração de testes		X	1
Adicionar mensagens de alerta / validação		X	5
No gráfico "Table" adicionar efeito "zebra" (cores intercaladas e.g. cinzento, branco)		X	1

A apreciação geral de toda a amostra foi positiva. Dos dois factores negativos mais identificados, 1 é um requisito a implementar no futuro e outro uma sugestão interessante que será adicionada aos requisitidos.

Os testes realizados tiveram um impacto bastante positivo na validação da interface. Os resultados sugerem que os utilizadores ficaram satisfeitos com o produto apresentado, mesmo este não estando concluído.

7.3.3. Testes no gameCenter

Na tabela que se segue são apresentados o tempo médio de execução, em segundos, de cada tarefa. Tal como no teste anterior, o cronómetro só foi iniciado depois de o utilizador ler a tarefa e terminado no final da sua realização.

Deste modo, em média, cada utilizador precisou de cerca de 1 minuto e 36 segundos para concluir o teste.

Tarefa	Tempo de Exec (s)
1	20,37
2	3,86
3	10,05
4	2,47
5	22,75
6	5,32
7	31,07

Não foram encontrados erros de funcionamento por parte da amostra, no entanto as opiniões encontradas sugerem que este tempo poderia ter sido maior caso a amostra não estivesse familiarizada com a instalação de aplicações no sistema operativo Android.

Na tabela que se segue estão registadas as opiniões apresentadas. Foram ainda contabilizadas as percentagens de ocorrências de cada uma.

Apesar da maior parte da amostra ter ficado relativamente satisfeita com o uso da aplicação, foram anotadas alguns factores negativos. Todos estes factores já eram esperados antes de se iniciar os testes e já estavam definidos como trabalho futuro. De referir que a amostra terminou com sucesso todas as tarefas definidas para o teste.

Opinião	Positiva	Negativa	Nº de ocorrência
Layout Intuitivo	X		1
Estrutura adequada	X		3
Fácil de utilizar	X		4
Adicionar labels na experiência e género		X	5
Adicionar mensagens de erro / sucesso		X	5
Definir tutorial de instalação (tem imagem e espaço reservado, mas não está concluído)		X	3
Melhorar ecrã de listagem de jogos		X	5

Capítulo 8
Trabalho futuro

Dada a grande dimensão desta plataforma, parte do trabalho a realizar no futuro passa por terminar a lista de requisitos apresentados. Para além desta lista, serão dadas algumas sugestões que considero necessitarem de alguma atenção.

8.1. Dados resultantes dos testes

Relativamente à interface para os *game designers*, para além da resolução dos erros de funcionamento encontrados, foram ainda retiradas os seguintes melhoramentos:

- Adicionar barra de progresso no upload do protótipo
- Adicionar título na secção de geração de testes
- No gráfico "Table" adicionar efeito "zebra" (cores intercaladas e.g. cinzento, branco)

Outras sugestões foram apresentadas, no entanto como já fazem parte da lista de requisitos não são enunciadas neste subcapítulo.

Na interface *gameCenter* foram registadas mais hipóteses de melhoria. Mesmo estando já programadas, como não estão explicitamente definidas na lista de requisitos, podem ser consultadas na lista que se segue:

- Adicionar labels na experiência e género
- Adicionar mensagens de erro / sucesso
- Definir tutorial de instalação (tem imagem e espaço reservado, mas não está concluído)
- Melhorar ecrã de listagem de jogos

8.2. Sugestões

Para além das melhorias que foram apresentadas no subcapítulo anterior, aproveito este espaço para sugerir mais algumas ideias que foram surgindo ao longo da realização do projecto:

- **Alargar o leque de game Engines**
  Actualmente a plataforma apenas permite o uso do Unity. Apensar da aplicação *gameCenter* ter sido desenvolvida com esta ferramenta, suporta jogos realizados com outras *game engines*. No entanto, a biblioteca utilizada para gerir a comunicação com o webServer2 teria que ser alterada, uma vez que a sua distribuição está associada a um *game object* de Unity para facilitar a sua integração.
- **Pensar numa forma de tornar o gameCenter numa aplicação que respeite os termos de uso da Play Store/ outras lojas**
  Actualmente o gameCenter não respeita todas estas condições, a começar por garantir que todos as actualizações tenham que ser descarregadas pela loja. Neste caso e uma vez que a lista de jogos é actualizada com muita regularidade e os protótipos são também descarregados à parte, optou-se por uma solução diferente. Caso se pretenda adicionar esta aplicação à Play Store ou a outra loja, esta situação tem que ser repensada.

- **Exportar o gameCenter para outros sistemas operativos móveis**
  O *gameCenter* está nesta altura disponível para qualquer dispositivo android sendo que seria interessante exportar esta aplicação para outros sistemas operativos como iOS, Windows phone ou até BlackBerry. Um factor que se terá que ter em conta é a sua adaptação a dimensões de ecrã diferentes.

- **Suportar jogos online**
  Esta sugestão poderia aumentar a diversidade de protótipos. Apesar disso e tal como na primeira sugestão, o método de recolha de dados teria que ser alterado.

- **Aumentar a quantidade de testes realizados**
  Dada a janela temporal limitada que existe para a realização e apresentação deste projecto, não foi possível realizar uns conjuntos de testes mais alargado e que é essencial para uma validação mais acertada da plataforma. Sendo então esta mais uma sugestão de trabalho futuro.

- **Integração com redes sociais**
  Para finalizar e não menos importante é sugerida a integração desta plataforma com as redes sociais. As vantagens desta associação seriam incalculáveis (e.g. facilitar o registo de jogadores, potenciar o aumento do número de jogadores dada a publicidade gratuita inserida no mural de cada um, explorar o conceito de *gamification* para motivar a entrada de novos jogadores e ainda estimular os que já se encontram registados a continuar a jogar, entre muitas outras)

Capítulo 9
Conclusões

Existem várias soluções disponíveis que se assemelham a este projeto. No entanto, a maior parte, centra-se mais na análise dos dados, sendo o *game designer* o responsável pela definição de todo o processo de recolha. Sem dados não é possível construir gráficos. Como tal, esta é uma das lacunas mais encontradas neste tipo de ferramentas.

De todas as ferramentas analisadas nenhuma permite definir mais do que um caso de teste e condições de acesso.

O facto de a *CrowdPlay* fornecer ao *game designer* uma capacidade de personalizar todos os parâmetros de recolha, análise e visualização de dados fazem desta plataforma uma das mais completas soluções disponíveis para o público em geral.

Apesar de um dos objectivos principais desta plataforma ser permitir um ambiente experimentalista de avaliação de comportamentos dos jogadores, poderia ser adaptada e tornar-se numa das ferramentas mais completas, disponíveis no mercado, para efectuar *game testing*.

A nível pessoal, ao longo deste estágio, adquiri e melhorei um conjunto alargado de conhecimentos não só relacionados com os conceitos teóricos associados ao projecto, mas também práticos relativos a ferramentas e até métodos de trabalho. Foi portanto, um ano positivo e bastante interessante na minha formação.

Acrónimos

HTML linguagem de marcação de hipertexto (Hypertext Markup Language)

HTTP protocolo de transferência de hipertexto (Hypertext Transfer Protocol)

JSON função ou característica do protótipo

REST transferência de estado representativo (Representational State Transfer)

XML linguagem de marcação extensível (eXtensible Markup Language)

Glossário

Android sistema operativo baseado em Linux. Pode ser encontrado em vários dispositivos como relógios, telefones, tablets ou mesmo televisões

Dashboard ferramenta de monitorização de dados

Feature função ou característica do protótipo

Feature set caso de teste, conjunto de features

Feature set generation conditions condições de geração de casos de teste (e.g. algoritmo de geração, função de probabilidade associada a cada feature, entre outras)

Filter filtro, restrição imposta a uma feature ou indicator

Game Engine motor de jogo. Framework de desenvolvimento de vídeo jogos

Indicator indicador, variável que exprime um resultado

Parsing análise sintática é o processo de analisar uma sequência de entrada

Play session período compreendido entre o início e o fim de cada jogo. O seu registo está associado a um jogo e a um utilizador.

Run session período que compreendido entre o inicio e o fim de cada jogada. O seu registo está associado à *play session* a que pertence e ao caso de teste usado durante o decorrer da jogada.

Stakeholders pessoas envolvidas no projecto

Test Conditions condições de execução de uma experiência / teste (e.g. condições de paragem, número máximo/mínimo de participantes, entre outras)

View representação gráfica

Referências

Howe J., *Crowdsourcing: Why the power of the crowd is driving the future of the business*. Business Books, Great Britain, 2008

Howe J., *The Rise of Crowdsourcing*. Wired Magazine, Issue 14.06, June 2006

Geiger D., Seedorf S., Schulze T., Nickerson R., Schader M., *Managing the Crowd: Towards a Taxonomy of Crowdsourcing Processes*. Proceedings of the Seventeenth Americas Conference on Information Systems, Detroit, Michigan, 2011

Xu A., Bailey B.P., *A Crowdsourcing Model for Receiving Design Critique*. CHI 2011, Vancouver, Canada

Komarov S., Reinecke K., Gajos K. Z., *Crowdsourcing Performance Evaluations of User Interfaces*. CHI 2013, Paris, France

Hassenzahl M., Tranctinsky N., *User experience – a research agenda*. Behavior & Information Technology, March – April, 2006

Law E., Kort J., Roto V., Hassenzahi M., Vermeeren M., *Towards a Shared Definition of User Experience*. CHI 2008, Florence, Italy

Hartson R., Pardha P., *The UX Boook. Process and guidelines for ensuring a quality user experience*. ElseVier, 2012

Fierley R., Engl S., *User Experience Methods and Games: Lessons Learned*. BCS ' 10, September 2010

Garrett J., *The Elements of User Experience. User-Centered Design for the web*. New Riders , 2003

Vaishanavi V., Kuechler B., *Design Science Research in Information Systems*. http://desrist.org/design-research-in-information-systems/, 2013

Clinfton M., Dunlap J., *what is scrum?*. Code Project, http://www.codeproject.com/Articles/4798/What-is-SCRUM, 2003

Schwaber K., Sutherland J., *O Guia do Scrum*. https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20Portuguese_European.pdf#zoom=100, 2013

Basili V., Caldiera G., Rombach H., *The goal question metric approach*.

Pruett C., *Hot failure: Tuning Gameplay with Simple Player Metrics* Game Developer magazine, September 2010

Mirza-Babaei P., Long S., Foley E., McAllister., *Understanding the Contribution of Biometrics to Games User Research*. DiGRA Conference, 2011

Mirza-Babaei P., Nacke L., Gregory J., Collins N., Fritzpatrick G., *How does it Play Better? Exploring User Testing and Biometric Storybords in Games User Research*. CHI 2013, Paris, France

Basili V., *Data collection, validation and analysis*. Tutorial on Models and Metrics for software Management and Engineering , 1981

Medler B., John M., Lane J., *Data Cracker: Developing a Visual Analytic Tool for Analysing Online Gameplay*. CHI 2011, Vancouver, Canadá

Participatory design. http://en.wikipedia.org/wiki/Participatory_design, Wikipedia

Wakkary R., *A Participatory Design Understanding of Interaction Design*

Business Cloud Storage Services Reviews and Comparisons, http://business-cloud-storage-services.toptenreviews.com/, 10TopTenReviews,2014

Meier J.D., Farre Carlos, Bansode Prashant, Barber Scott, Rea Dennis, *Performance Testing Guidance for Web Applications*, Microsoft Developer Network, http://msdn.microsoft.com/en-us/library/bb924376.aspx , September 2007

Feldman, *Varieties of Visual Experience; Art as Image and Idea*. H.N.Abrams, New York, 1972

Feldman, Practical Art criticism. Englewood Clifss, NJ: Prentice Hall, 1994

Teo Y. H., Chai C.S., *Scaffolding Online Collaborative Critiquing for Educational Video Production. Knowledge Management & E-Learning*, An International Journal (KM&EL), 2009

Alben L., *Quality of Experience: Defining the Criteria for Effective Interaction Design*, interactions, 1996

Hartson R., Pyla P.S., *The UX Book: Process and Guidelines for Ensuring a Quality User Experience*, Elsevier, 2012