Trabalho Prático de Engenharia de *Software*

Diego Cintra, Jainor Souza, Mariana Caravanti, Nicolas Roque

22 de maio de 2014

Sumário

Resumo

Este documento refere-se ao trabalho prático da disciplina de Engenharia de *Software*, ministrada pela professora Débora Maria Barroso Paiva (FACOM − UFMS), que visa praticar os conceitos de Engenharia de Requisitos aprendidos ao longo do curso.

1 A Proposta

O Trabalho Prático de Engenharia de *Software* teve seu tema definido pela professora Débora, que é a criação de um projeto para o desenvolvimento ou a modificação de um jogo.

1.1 Exigências Básicas do Trabalho

1. Deve haver, no mínimo, um usuário final do *software*;
2. Executar todo o processo de Engenharia de Requisitos;
3. Um protótipo baseado em imagens necessita ser apresentado para a professora;
4. Casos de uso de formato expandido necessitam ser feitos para as principais ideias do projeto;
5. Pede-se estimar o tamanho e o preço do *software* utilizando a técnica de Pontos por Função.

2 Usuários Finais − *Stakeholders*

2.1 Escolha dos Usuários Finais

Usufruindo do privilégio de poder escolher uma pessoa conveniente para nos auxiliar no trabalho(não fazendo parte do grupo), escolhemos uma amiga de longa data que também cursa Ciência da Computação, mas que está, neste momento, estudando no Reino Unido, por meio do projeto de extensão Ciência Sem Fronteiras.

2.1.1 Aimée Sousa Calepso

Por ser amiga do grupo como um todo e ter tempo disponível para nos auxiliar no projeto, Aimée foi a escolhida para representar um Usuário Final do *software*. Decidimos que seu papel seria do *Stakeholder* responsável por contratar nosso grupo para desenvolver um novo jogo.

E-mail para contato: aimeesc@gmail.com

3 Conceitos Iniciais

Inicialmente, havíamos decidido, com os interesses do *Stakeholder*, que não modificaríamos um jogo existente, mas sim criaríamos um novo jogo. A ideia se deu ao fato de, em geral, julgarmos os jogos que nós(o grupo e o *Stakeholder*) gostamos bons, onde modificações não seriam necessárias.

3.1 Ideias

Iniciamos um *brainstorming* muito simplista entre os integrantes do grupo e o *Stakeholder*, de forma a definir qual seria o maior ponto de interesse do projeto, qual seria o público-alvo e qual plataforma teria de suportar o jogo. Como esta reunião foi muito rápida, algumas questões ficaram em aberto, como:

- Qual o molde que o *software* deve possuir?
- O *software* teria bases em algum jogo já existente?

- Há personagens fictícios no *software*?
- Qual seria a dificuldade do jogo?
- Qual o poder aquisitivo que os outros usuários finais tem de possuir para fazerem uso do *software*?

3.1.1 Alocação de Tarefas

Com a urgência de definir melhor o escopo e a aplicação dessas questões, uma reunião em grupo foi convocada. Primordialmente, procuramos sanar dúvidas de divisão de trabalho do grupo. Optamos por preferências pessoais, onde realizar um trabalho forçado seria fora de questão, para tornar o desenvolvimento do trabalho mais sadio, produtivo e divertido.

A tarefa de Engenharia de Requisitos foi alocada para o aluno Diego Cintra, que tem mais contato com a "dona de empresa" Aimée do que o resto do grupo, além de gostar da parte de Engenharia de Requisitos. A realização dos Casos de Uso ficou por conta dos alunos Mariana Caravanti e Nícolas Roque, que se voluntariaram a realizar os casos de uso por terem mais afinidade com o quesito do que os demais. A realização da documentação do trabalho, incluindo este relatório, ficou por conta de quem vos fala, o aluno Jainor, por gostar desta posição como um todo.

Ao final de cada atividade realizada, as produções eram compartilhadas entre o grupo, para então polirmos as ideias, dividir dificuldades, aplicar mudanças e basear uma próxima produção nas críticas de cada integrante do grupo. Com as atividades dos integrantes do grupo bem torneadas, pudemos então decidir quais seriam os melhores métodos de abordagem do *stakeholder*, de forma a suprir as necessidades de cada tópico do trabalho. Uma entrevista era iminentemente necessária.

4 Interação com o *Stakeholder*

Após decidirmos que o principal meio de contato com o *stakeholder* seria a entrevista, baseamo-nos em dúvidas pré-estabelecidas(como as da seção anterior) para formular uma lista de perguntas base competente o suficiente para proporcionar grande parte das ideias do desenvolvimento do *software*, bem como nos permitir descobrir peculiaridades sobre o contratante.

4.1 Perguntas

Sabendo que o coração de qualquer equipe de desenvolvimento de *software* é o documento de requisitos, necessitamos expandir as ideias para além das divisões de trabalho de cada membro do grupo, a fim de formular perguntas mais objetivas e precisas, para moldar os alicerces do documento de requisitos e começar o planejamento dos casos de uso. O quê perguntar e qual o momento da entrevista para desencadear perguntas foram fatores muito pensados, para que o cliente se sentisse mais à vontade e menos pressionando, acarretando num entendimento mais certo de suas necessidades e evitando possíveis problemas de comunicação, que são o terror de qualquer empresa no ramo de tecnologia da informação.

A lista de perguntas inicial consistiu de perguntas complexas, que deveriam ser proferidas de maneira gradativa e extrovertida, sem danificar a integridade da equipe(do grupo). Tendo em mente manter um constante fluxo de perguntas, o questionário base serviu apenas de molde para a entrevista que realizaríamos, já que o cliente não deveria seguir um algoritmo em especial para tirar nossas dúvidas. Fazê-lo sentir que estávamos(e, de fato, estávamos) interessados em serví-lo da melhor maneira possível era um ponto imprescindível na atmosfera do ambiente, pois, apesar de atuarmos num ramo considerado Ciências Exatas, não há produto de sucesso ou de qualidade que não siga as mais primordiais regras das ciências humanas: a comunicação social.

Ao final de muitos pensamentos, optamos por partir de algumas perguntas leves e discretas, porém poderosas.

4.1.1 Perguntas – Base

LISTA DE PERGUNTAS BASE

1. Qual o tipo de jogo desejado?
2. Qual é o público-alvo do jogo?
3. O jogo deve ser competitivo?
4. Existem limitações técnicas muito rústicas?
5. Qual a temática do jogo?
6. O jogo possuirá investimento muito pesado?
7. Existe suporte para mais de um jogador?

Por serem perguntas muito bem definidas, seria possível enraizar os pensamentos mesmo que o cliente as respondesse de forma algorítmica. O propósito de partir de um questionário base era exatamente explorar a capacidade de formar perguntas eficientes, objetivas e que não ocupassem muito tempo. A ligação entre estes sete fatores questionados é muito clara e informa as direções que o cliente desejaria tomar quanto ao produto final, reduzindo riscos de erro.

4.2 A Entrevista

Para realizar a entrevista, pedimos para que o *stakeholder* agisse como se não conhecesse o entrevistador, para produzir um conteúdo de conotação mais séria e com o intuito de praticar a comunicação. Apesar de baseada no questionário base anterior, a entrevista foi feita como uma conversa normal entre duas pessoas, para que houvesse maior entendimento das duas partes.

Por ter mais contato e estar mais ciente dos horários do *stakeholder*, o aluno Diego Cintra conduziu a entrevista. Aqui está a entrevista final, na íntegra:

4.2.1 Entrevista com o *Stakeholder*

ENTREVISTA COM O STAKEHOLDER

Entrevistador: Diego Silva Cintra
Entrevistado: Aimée Sousa Calepso
Perguntas(números) e Respostas(itens):

1. Você nos requisitou a criação de um jogo novo, que seja de qualidade e que agrade os seus clientes. Portanto, que tipo de jogo você deseja?

    - Primeiramente, o jogo deveria ser lucrativo, pois uma parte considerável dos nossos investimentos vão para a criação desse jogo. Em segundo lugar, contratamos uma equipe que fez uma pesquisa de mercado para saber o que nosso clientes querem, e até chegamos a um consenso em relação ao jogo; porém, ficamos na dúvida sobre o quão relevante seria essa pesquisa, visto que, caso o *software* demore um tempo considerável para ficar pronto, talvez a opinião dos consumidores já tenha mudado bruscamente, e aquele produto já não atenda às necessidades deles.

2. Compreendo. Dado essa dúvida, gostaria de saber então o que você espera em relação ao tempo de desenvolvimento desse *software*: um tempo muito longo traria complicações e perda de dinheiro para vocês? O jogo deve ser de alta qualidade, mesmo gastando tanto tempo? Ou deve-se criar um jogo mais simplório em detrimento do tempo de produção?

- Como somos uma empresa de pequeno porte, acredito que o mais eficiente a ser feito agora seria realmente produzir um jogo mais modesto, que não precise utilizar de todos os recursos computacionais de última geração para garantir uma exeperiência agradável ao jogador; ele deve possuir uma ideia muito bem solidificada, que mesmo com as limitações técnicas, ainda seja capaz de agradar aos jogadores. Novamente, como se vê na atualidade, alguns jogos como *Angry Birds* e *Flappy Bird* conseguiram gerar vastas quantidades de dinheiro devido ao simples fato de focar na diversão proporcionada ao jogador, e não em gráficos de última geração e história e jogabilidade impecáveis.

3. Então, deixe-me ver se eu entendi: o jogo que vocês desejam, por ser mais simples e mais rápido de ser produzido, deve focar em aspectos mais solidificados de um produto de entretenimento, como a diversão e o quanto de tempo o usuário gasta jogando, e deve abrangir uma grande massa de pessoas, com diferentes conhecimentos sobre jogos? Por exemplo, uma pessoa que nunca jogou antes deve se sentir atraída por esse produto? Quem acaba sendo o principal foco desse *software*?

    - Pois bem, o foco irá depender do gênero do jogo; se ele for do tipo ação e aventura, por exemplo, então o foco acabará sendo o público mais jovem. Agora se o jogo for de um gênero mais infantil, então o foco serão crianças. Esse aspecto depende principalmente do gênero a ser escolhido para o jogo. Como disse anteriormente, até chegamos a fazer uma pesquisa de mercado, porém os entrevistados talvez não tivessem características heterogêneas, e portanto o consenso obtido pode ser um pouco tendencioso.

4. Certo. Dado essa implicação, intuitivamente a pergunta que surge é: qual será o estilo do jogo a ser produzido?

    - Como já explicamos, o jogo é simples, portanto o aprendizado do jogador deve ser, no começo, rápido. Porém, não queremos um jogo que seja muito fácil, e portanto gostaríamos que ele propusesse um desafio maior para aqueles que realmente investirem seu tempo nele e desejarem se tornar o melhor. Logo, o jogo deve ser fácil de aprender, porém a dificuldade para os jogadores que quiserem ser imbatíveis deve ser bastante elevada, e acredito que o gênero que melhor se encaixa para tal situação é o *puzzle* (quebra-cabeça).

5. Então o escopo do consumidor pode ser mais bem especificado agora?

    - Sim, creio que qualquer pessoa que tenha acesso a uma mídia de entretenimento – como celurares ou *consoles* – pode ser um usuário em potencial, independente de sexo ou de idade. Também acredito que, se o aspecto da facilidade inicial for bem executado, o escopo será mais abrangente e incluirá pessoas que nunca jogaram na vida ou que não se satisfazem com outros jogos mais complexos, preferindo a simplicidade desse.

6. Ótimo, definimos um aspecto do jogo. Agora, outra característica importante a se considerar é: qual a plataforma de mídia de desenvolvimento que o jogo será suportado e qual será a forma de distribuição dele?

    - Creio que as plataformas seriam *mobile* (celulares e *tablet*) e computadores em geral, sendo a mídia disponível para o primeiro através de *download* gratuito e a do último como um jogo *online* acessível através de um navegador.

7. Mas a proposta da empresa não era gerar um produto lucrativo? Se o jogo é gratuito, como se daria o lucro?

    - Esse retorno financeiro será obtido através da popularidade que o jogo trará; com uma maior atenção por parte dos consumidores, expandiríamos a ideia do jogo para outras indústrias, como a de alimentos e de outros tipos de entretenimento, como a cinematográfica, de

bonecos e animais de pelúcia, através do licenciamento das características e personagens presentes no jogo. A ideia não deve ser limitada apenas ao conteúdo do jogo em si, porém ao *marketing* e a atenção que ele irá trazer com sua disseminação entre os consumidores e seus conhecidos – para o caso da indústria cinematográfica, por exemplo, pode-se licenciar os direitos autorais do jogo para a realização de propagandas ou participação em campanhas de diversos cunhos. Isso pode ser visto em outros jogos, como novamente *Angry Birds*.

8. Para a expansão a esses outros mercados, acredito que se deve levar em consideração um tipo de "mascote" que carrega a ideia do produto. No caso desse jogo, deveríamos considerar um personagem específico? Ou uma característica do jogo?

    - Talvez pudesse haver um personagem que servisse de mascote, entretanto acredito que alguma característica do jogo é mais adequada a ser explorada para outros âmbitos de mercado. Não há necessidade de forçar um personagem só para esse fim; se ele surgir naturalmente, então obviamente ele seria a escolha de mascote.

9. Perfeito. Até então, fomos capazes de especificar, em termos gerais, o que é o jogo, qual seu gênero, qual o escopo dele e a forma de lucro obtida por ele. Agora, vejamos alguns aspectos relativos ao funcionamento do produto. Qual será o modo de jogo dele? Ele pode ser jogado por apenas um jogador ou vários simultaneamente? Haverá alguma espécie de *ranking* ou classificação para comparar entre os diversos usuários?

    - O foco desse jogo deve ser no modo de um jogador, pois acho que, para o caso de *softwares* desenvolvidos para *mobile*, a ideia de multi jogador não é muito funcional, se compararmos com outras plataformas; dado que celulares e *tablets* são dispositivos com características bastante individuais, intuitivamente os aplicativos disponíveis para esses seguem a mesma filosofia. Entretanto, com relação a uma tabela de classificação e pontuação, pode existir, e diversos usuários podem comparar os seus pontos uns com os outros; entretanto, tal característica não é necessária para o desenvolvimento do produto.

10. Se o jogo deveria ter um padrão de dificuldade elevado para os jogadores mais assíduos, então por que essa última característica é opcional?

    - Repare que o jogo continua sendo desafiador, com ou sem a inclusão de uma tabela de classificação, pois o usuário pode comparar sua pontuação com a de outro colega de maneira lusitana – simplesmente mostrando a sua melhor pontuação. Esse aspecto não é necessário pois, como principal característica do jogo, deve-se haver uma competição entre o ser humano e a máquina; é essa particularidade que deve ser explícita ao jogador. De maneira geral, quando algum jogador ganha alguma conquista ou completa um jogo e outro não faz isso, conclui-se que o primeiro é um melhor jogador do que o segundo, pois a máquina possui uma dificuldade constante – ou seja, essa dificuldade é limitada a um certo ponto. Isso difere do fato de, caso um jogador seja derrotado por outro jogador em uma partida de um jogo *multiplayer*, isso não implica dizer que ele é pior que um terceiro jogador que perdeu de um quarto, nesse mesmo jogo, pois a dificuldade de se enfrentar um ser humano é sempre variável. Portanto, a filosofia que deve ser embasada no desenvolvimento desse software é a competição entre humano e máquina, e por isso se dá a necessidade de um jogo realmente desafiador.

11. Isso faz sentido. Portanto, dado que o principal objetivo é desafiar o usuário, como seria o funcionamento desse jogo? A definição "quebra-cabeça"é bastante abrangente. Quais seriam as principais características desse jogo?

    - Deve-se haver uma maneira de monitorar a progressão do jogador, através da inclusão de níveis de dificuldade que aumentam conforme o jogo vai evoluindo. A cada nível que o

jogador vai avançando, esporadicamente ele deve receber algum tipo de recompensa pelo seu esforço, que poderá ser utilizada durante o progresso do jogo. O jogo também deve ter uma interface bem fácil de entender, e a princípio, como previamente mencionado, ele deve ser fácil de interagir e jogar. Para a parte principal, o usuário é apresentado com diversas situações, como juntar blocos de diferentes cores ou arremessar uma pedra de um estilingue respeitando um tipo de precisão; em resumo, o jogo seria uma coleção de *minigames*, que vão evoluindo de dificuldade na medida certa, de maneira com que o jogador não enjoe pela facilidade logo de cara, porém não desista devido a dificuldade excessiva.

12. Essa coleção de diferentes *minigames* seria apresentada de que maneira?

    - Pode-se escolher qual o jogo que se quer jogar, ou essa escolha pode ser feita de maneira aleatória, através de menus; de maneira similar, a partir da evolução nos níveis de jogo, pode-se continuar em um único *minigame* ou aleatoriamente selecionar outros.

13. Ainda sobre eles, devemos estabelecer um limite para o número desses disponíveis?

    - Deve haver, no mínimo, três desses *minigames* disponíveis, e caso o jogador fosse capaz de alcançar uma pontuação realmente impressionante nos 3, um quarto mini-jogo ficaria disponível ao usuário para jogar. Ao longo dos 3 outros jogos, o software deveria implicitamente avisar ao jogador de que há uma recompensa caso ele cumpra os requisitos especificados, pois assim o esforço dele será maior e mais concentrado.

14. Agora, se possível, dê uma breve descrição de como seriam esses *minigames*.

    - O primeiro jogo seria um estilo clássico de quebra-cabeça, onde deve-se empilhar peças que descem formando pares de três ou mais elementos iguais, eliminando-os da pilha e dando lugar a novos elementos que descem; esses formatos podem ser variados, como até seis elementos na vertical ou horizontal, cinco elementos em formato de cruz, seis elementos concatenados como dois pares de três elementos, entre outros, desde que a propriedade de se haver três peças lado a lado ainda seja respeitada. Para o outro, uma mescla de estilo plataforma seria inclusa: dado um personagem e um conjunto de obstáculos, deve-se chegar do ponto A até o ponto B o mais rápido possível, passando ileso pelo caminho. Para o terceiro jogo, dado um conjunto de blocos na parte superior e um retângulo controlado pelo jogador na parte inferior, deve-se eliminar todos as peças de cima com o auxílio do bloco retangular e uma bola, que tem de sempre rebater no retângulo. Por último, o jogo final a ser desbloqueado seria um simples jogo de memória, onde dado uma sequência de elementos, deve-se memorizar a ordem em que eles aparecem e replicá-la logo a seguir; quanto maior o tempo, mais longa será a cadeia.

15. Ótimo. Porém, posso sugerir uma abordagem diferente a alguns dos jogos citados? Eles já são bastante conhecidos e não podemos garantir um impacto de peso desses sobre o público alvo. Além do mais, acredito que um deles não seria necessariamente um quebra-cabeça.

    - Tudo bem. Qual seria a ideia?

    Poderíamos fazer um *minigame* com foco em cálculos matemáticos, como por exemplo, dada uma expressão, deve-se armazenar o resultado dela e, ao aparecer a próxima, aplica-se a resposta da conta anterior, ainda mantendo a solução da expressão atual na memória. Pode-se fazer isso para 1 expressão ou mais. Seria uma boa recompensa como quinto jogo. Por isso, substituiríamos um dos 3 jogos pelo jogo da memória, para dar lugar a esse novo jogo.

    - Parece uma ótima ideia. Qual dos jogos seria substituído então?

Acredito que o quarto jogo possa ser descartado. Que tal?

- Sim, parece-me bom.

16. Acredito que estejamos no final da entrevista, e já possuo informações suficientes para a iniciação de desenvolvimento do *software*, portanto cabe-me uma última pergunta: Existe mais alguma colocação que deseja fazer?

    - Em relação a construção do jogo, acredito que tudo esteja esclarecido; porém, gostaria de saber a respeito do prazo para entrega e o retorno sobre investimento resultante.

    Sobre esses aspectos, faremos uma estimativa de custo, tamanho e tempo com base na técnica de pontos por função, sendo independente de linguagem de programação, linhas por código ou da tecnologia utilizada para implementação. Após a geração desses resultados, informaremos a vocês esses valores para garantirmos o desenvolvimento do *software*.

    - Perfeito. Espero o retorno de vocês.

4.3 Considerações Sobre A Entrevista

Percebe-se que as perguntas base foram grande parte do conteúdo da entrevista, tamanha a preocupação em manter os primórdios do projeto muito bem definidos. Vale a pena ressaltar que o cliente enfatizou bastante o fato de que o jogo necessita ser amigável a principiantes, então há a necessidade de realmente separar o público casual do público competitivo, de forma que haja suporte para os dois, mas sem tornar as finalidades de cada um distorcidas.

Vê-se também a importância de uma comunicação social impecável, visto que o cliente, às vezes, pode não perceber fatos sobre o produto que ele deseja que talvez não sejam tão bons. Como sugeriu nosso porta-voz nesta entrevista, trocar os mini-jogos foi realmente uma boa dica para o cliente, pois atrairá mais usuários. Estes pequenos fatos, às vezes, podem ganhar o projeto para uma equipe. Em nosso caso, não havia uma concorrência, mas aposto que o cliente já haveria criado tal empatia com nossa equipe que praticamente nos ganharia o "processo".

5 A Aplicação – *Jelly Brick Game*

Após a realização da entrevista, o escopo do jogo ficou muito bem definido, proporcionando uma boa noção para o desenvolvimento da Engenharia de Requisitos. Como o nome do jogo foi um tópico inicialmente em aberto, demos ao projeto o nome *Jelly Brick Game*, o qual fora repassado à cliente e aprovado. Tendo em mente a consistência dos dados, um documento de requisitos foi elaborado e discutido entre os membros do grupo, onde sofreu algumas alterações, que originou o documento de requisitos oficial do *Jelly Brick Game*. Segue o documento supracitado.

5.1 Documento de Requisitos do *Jelly Brick Game*

Documento de requisitos gerado a partir da entrevista de um *stakeholder*

A. **Visão Geral do Sistema**
O software "*Jelly Brick Game*" tem como foco servir como uma agradável experiência de entretenimento, tendo como usuários qualquer pessoa que se interessa ou não por jogos eletrônicos. Esse jogo deve permitir que o usuário seja capaz de registrar seu nome (ou *nickname*), escolha entre diversos tipos de *minigames* para jogar e veja suas melhores pontuações. Os *minigames* existentes são: *Puzzle Challenge, Balloon Breeze, Simon Fibonacci* e *Recursive calculations*. No primeiro, deve-se juntar combinações de peças, livrando-se delas da pilha; já para *Balloon Breeze*, o objetivo do jogador é fazer seu personagem chegar em terra firme, evitando diversos obstáculos pelo caminho. Para o terceiro, deve-se memorizar a sequência de elementos que aparece na tela, para

depois reproduzi-los. O *Recursive Calculations* consiste em um exercício de concentração, onde dado uma expressão, memoriza-se seu resultado e, ao aparecer a próxima expressão, deve-se escrever o resultado da anterior, e assim sucessivamente; esse último só poderá ser jogado assim que certos requisitos nos outros três forem cumpridos (que serão especificados mais adiante), sendo um jogo, a princípio, bloqueado.

B. **Requisitos Funcionais**

1. Cadastro de Jogadores
   (a) O software deve permitir o cadastro dos jogadores, através de uma única informação: o *nickname* (ou apelido) do jogador.
   (b) O jogo deve permitir que os jogadores possam alterar o seu nome dentro do jogo, sem perder os dados de pontuação obtidos até então.
2. Funcionamento do Jogo
   (a) O jogo deve permitir aos jogadores a seleção entre dois diferentes modos de jogo: ou um modo em que todos são misturados e cada um deles são divididos em fases, que aumentam de dificuldade ao longo do tempo (*jelly mode*), ou um modo em que pode-se escolher um dos jogos e prosseguir somente nele, com uma jogabilidade diferente da anterior (*free mode*).
   (b) O jogo deve, caso o usuário selecione o modo de jogo convencional, apresentá-lo a um próximo menu, onde ele pode escolher um dos 3 modos de jogo singularmente. Durante o jogo, caso a pessoa continue sem perder, com o passar do tempo, a dificuldade aumentará de maneira implícita, sem que o jogador perceba. Isso continuará até que o usuário perca, situação em que a sua pontuação é informada na tela, assim como opções de tentar novamente ou escolher outro modo de jogo.
   (c) O jogo deve, caso o usuário selecione o modo de jogo aleatório, permitir que ele jogue uma variante dos 3 jogos, onde os níveis são modificados para permanecerem mais curtos e o tempo para completar os desafios é limitado. Caso a pessoa consiga completar o que lhe é requisitado na tela, ao avançar nos níveis, esses limiares são reduzidos e a dificuldade aumenta, e a cada troca de nível, sua pontuação é incrementada em 10 pontos, e um breve período de tempo é reservado para a pessoa se preparar para o próximo jogo. Em caso negativo, o usuário perde o jogo e a sua pontuação é informada na tela, assim como opções de tentar novamente ou escolher outro modo de jogo.
   (d) O jogo deve disponibilizar um pequeno e compreensivo tutorial para todos os minigames, caso o usuário selecione essa opção a partir do menu principal, seja no *free mode* ou no *jelly mode*, especificando os controles e metas do jogo.
   (e) O jogo deve, para o *Puzzle Challenge*, trocar duas peças adjacentes da direita para a esquerda caso o jogador deslize o dedo sobre a tela da direita para a esquerda e vice-versa. Caso o usuário consiga casar uma combinação de elementos com características iguais, então as peças devem desaparecer da tela; uma combinação de elementos consiste em três peças ou mais de mesma cor, que estão uma ao lado da outra (seja verticalmente ou horizontalmente), e derivações dessas formas, cuja principal característica é a presença de três elementos adjacentes (como uma cruz ou um formato similar a letra "L"). Para o modo aleatório, o objetivo é fazer com que nenhuma peça reste na pilha, e deve-se eliminá-las com a quantidade de movimentos estabelecida pelo jogo, que é mostrada na parte superior da tela – o usuário pode desfazer seu último movimento ao tocar no botão de mesmo nome, ganhando um movimento a mais - e também respeitando o tempo especificado; já para o *free mode*, dado uma pilha, os elementos sobem indefinidamente, e deve-se fazer combinações de maneira a evitar com que o acúmulo de peças lote a tela. Quanto mais tempo o jogador aguentar, mais rápido a pilha cresce.
   (f) O jogo deve, para o *Balloon Breeze*, permitir que o jogador deslize o dedo sobre a tela, para que o personagem ganhe impulso e seja propulsionado para a frente. O objetivo é fazer

o personagem chegar à terra firme, evitando ao máximo obstáculos no caminho; caso, ao final da fase, o jogador pouse no topo do pódio, independente de qualquer modo de jogo, 10 pontos extras são adicionados ao seu placar. No *jelly mode*, caso o personagem consiga pôr os pés no chão, a sua pontuação é incrementada, e o usuário é introduzido à tela de seleção para outro jogo, escolhido aleatoriamente; já no modo convencional, cumprido esse objetivo, o personagem, caso o jogador queira, continua sua travessia, porém agora em outra fase, com a dificuldade aumentada. O usuário possui dois balões que suspendem seu personagem: no evento de encostar em espinhos ou ser atingido por um inimigo, um desses balões estoura. Caso ele voe muito baixo, um peixe pode submergir e engoli-lo, e ele perde o jogo.

(g) O jogo deve, para o *Simon Fibonacci*, iluminar uma sequência de botões coloridos, para então o jogador reproduzir essa sequência, respeitando a ordem proposta. Em um total de 7 elementos, para o modo aleatório, ao reproduzir corretamente uma sequência, o usuário é introduzido à tela de seleção para outro jogo, escolhido aleatoriamente; para o modo convencional, cumprido esse objetivo, o jogador pode escolher entre continuar a jogar ou mudar de modo de jogo, onde escolhendo a primeira opção, a sequência fica mais longa do que a anterior.

(h) O jogo deve permitir com que o usuário perceba que há um quarto jogo a ser liberado, da seguinte maneira: durante o modo aleatório, a cada 100 pontos alcançados, deve-se informar a frase "Continue assim!!! Uma surpresa espera por você ao chegar em 1000 pontos!". Esse quarto jogo deve ser desbloqueado através do cumprimento das seguintes tarefas: no *free mode*, para o *Puzzle Challenge*, deve-se aguentar um tempo mínimo de 50 minutos, e para os dois últimos, deve-se chegar até o nível 110; e no *jelly mode*, uma pontuação mínima de 1010 pontos deve ser alcançada. Conquistando esses dois desafios, o *Recursive Calculations* é liberado, sendo acessível somente para o free mode.

(i) O jogo deve, para o *Recursive Calculations*, permitir com que o usuário selecione, dentre um menu, qual será o tamanho das respostas a serem decoradas (*N-back*), e se serão somente expressões ou equações. De início, somente as opções "*1-back*" e "expressões" estão disponíveis; porém, caso o jogador consiga passar 5 níveis, um novo "*N-back*" é desbloqueado, sendo ele um incremento em um em relação ao anterior, cujo limiar vai até "*99-back*". Ao completar os 20 primeiros níveis, outra opção, "equações", também é liberada. Para o modo "equações", todos os "*N-back*" liberados ainda continuam valendo, não sendo eles reiniciados.

(j) O jogo deve, para o *Recursive Calculations*, dispor de um teclado virtual para o usuário, a fim de que ele escreva as respostas das equações (ou expressões, dependendo da escolha) e submeta-as. A cada acerto que ele realiza, sua pontuação é incrementada em um ponto, e caso ele erre, nenhuma pontuação é considerada. Em uma série de 20 equações, caso ele consiga uma pontuação maior que 12, então ele avança de nível; caso seu número de acertos esteja entre 12 e 8, então ele permanece nesse nível; e caso ele acerte menos de 8, então, salvo se ele estiver no primeiro nível, ele decresce um nível. Se ele estiver no primeiro nível e não conseguir uma pontuação acima de 12, ele irá permanecer nesse nível.

(k) O jogo deve, independente da escolha do usuário de modo de jogo, permitir, a qualquer momento, que o jogador interrompa o jogo e saia para a tela de seleção de modos de jogo; naquele momento, a pontuação atual associada ao usuário é gravada.

3. Emissão de Pontuações

   (a) O jogo deve mostrar na tela a pontuação recorde e o apelido (*nickname*) de todos os jogadores cadastrados no software, caso o usuário requisite-as.

C. Requisitos Não-Funcionais

1. Usabilidade

(a) O jogo deve ser fácil de aprender e usar e possuir uma interface bastante intuitiva e atraente ao usuário. Ele também deve considerar as sub-características de usabilidade apresentadas na ISO/IEC 9126.

2. Confiabilidade
   (a) O jogo deve ter capacidade de recuperar o último jogo e sua pontuação alcançada, em caso de falha.
3. Portabilidade
   (a) O jogo deve ser compatível com os sistemas operacionais de dispositivos móveis *iOS* e *Android*.
   (b) O jogo deve estar disponível como mídia digital para *download* nas plataformas de distribuição de aplicações *Google Play* e *Apple Store*.

5.2 Considerações sobre o *Jelly Brick Game*

Inicialmente, o grupo se manteve dividido sobre a real facilidade de se aprender a jogar o jogo. Com conversas a fim de explanar os passos e ambientes de cada tipo de mini-jogo, chegamos à conclusão de que não era a dificuldade do jogo em si, mas sim alguns detalhes do documento de requisitos. Para ilustrar melhor o que antes ficava somente na abstração textual, criamos algumas telas do jogo como o imaginamos, tanto em jogabilidade quanto em interface de interação humano-computador. Os modelos usados foram bem simplistas, apenas para identificar as passagens definidas no documento de requisitos e na entrevista e para facilitar o entedimento da proposta do projeto.

5.3 Ilustração do *Software*

As imagens foram feitas a partir das ideias dos membros do grupo e foram escolhidas as imagens que consideramos mais objetivas. Alguns detalhes, como a "mascote" não foram inclusos nestes protótipos de telas.

5.3.1 *Software* de Apoio – Ferramenta de Desenho

Para nos auxiliar no processo de desenho do *software*, contamos com o apoio da ferramenta ***Balsamiq Mockups***, que provê um meio bem prático de esquematizar aplicativos para algumas plataformas.

Como o foco está nos dispositivos *mobile*, as ilustrações foram feitas em protótipos do tipo *Smartphone*, pois a maioria desses dispositivos possui capacidade para executar um *software* do porte do que estamos esperando produzir.

5.3.2 Prototipação do Menu Principal

5.3.3 Prototipação do Cadastro de Novo Jogador

5.3.4 Prototipação dos Modos de Jogo

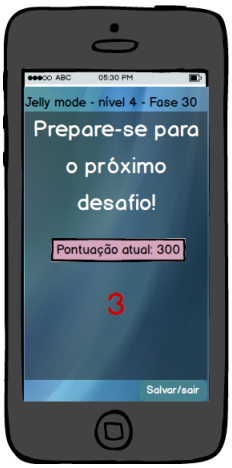
5.3.5 Prototipação do Mini-Jogo *Puzzle Challenge*

5.3.6 Prototipação do Mini-Jogo *Balloon Breeze*

5.3.7 Prototipação do Mini-Jogo *Simon Fibonacci*

5.3.8 Prototipação do Mini-Jogo *Recursive Calculations*

5.3.9 Prototipação do Intervalo Entre Jogos



5.3.10 Prototipação do *Ranking*

6 Casos de Uso do *Software*

Para acompanhar o documento de requisitos, que possui as definições formais do *software*, o documento de casos de uso foi produzido. Visando possuir o maior nível de detalhamento das principais funções do *software*, os casos de uso foram focados, principalmente, nos modos de jogo. Por ser um jogo, por mais que o *software* possua outras funcionalidades, elas acabam, invariavelmente, estando ligadas fortemente com a ação de jogar.

Essas funcionalidades acabam sendo aquelas de registro, alteração de registro ou o guia do jogo. Como previsto, estas funcionalidades agem em conjunto com os mini-jogos e estão disponíveis para o usuário assim que ele abre o jogo. Porém, quando o usuário é **forçado** a realizar tais ações, estas funcionalidades acabam constituindo um seleto grupo de exceções do *software* e é por isso que, apesar de conjuntos, são tratados de forma separada nos casos de uso. Veremos a seguir os quatro casos de uso mais básicos do *Jelly Brick Game*.

6.1 Caso de Uso: Jogar *Puzzle Challenge*

Caso de Uso: Jogar *Puzzle Challenge*

Escopo: *Jelly Brick Game*

Atores: Usuários do *Jelly Brick Game*

Interessados: Usuários de jogos de dispositivos móveis, solicitadores do desenvolvimento do jogo

Pré-condições: O usuário deve possuir *Jelly Brick Game* instalado em seu dispositivo móvel, que pode ser obtido por download através do *Google Play* ou *Apple Store*

Pós condições: O usuário utilizou o aplicativo e está satisfeito com sua pontuação.

Requisitos Correlacionados: B.1, B.2, B.3 e C (de acordo com o documento de requisitos gerado)

Variações teconológicas: O usuário pode controlar a direção da troca dos blocos para direita ou para a esquerda através do toque do dedo e seu movimento na respectiva direção, bem como selecionar o opção para salvar e sair.

Fluxo Principal:

1. [EV] O usuário carrega o jogo *Jelly Brick Game* em seu dispositivo móvel.
2. [EV] O usuário seleciona seu nome (*nickname*), no início do aplicativo.
3. [RS] O usuário é informado dos modos de jogo: *free mode* ou *jelly mode*.
4. [EV] O usuário opta por jogar o *free mode*.
5. [RS] O usuário é informado dos 3 diferentes tipos de jogos disponíveis no *free mode*.
6. [EV] O usuário seleciona *Puzzle Challenge* e inicia a partida.
7. [EV] O usuário joga:
   - Sucesso: Ver variante 7.1.
   - Derrota: Ver variante 7.2.
   - Pontuação: Ver variante 7.3.
   - Satisfação: Ver variante 7.4.
8. [RS] É informado ao usuário qual foi sua pontuação final.
9. [RS] É fornecido ao jogador a opção de tentar novamente ou escolher outro modo de jogo.
10. [EV] O usuário vai para a tela de menu principal.
11. [EV] O usuário fecha a aplicação.

Tratamento de exceções:

3a. O usuário é novato no aplicativo, e deseja aprender a utilizar o jogo.
3a.1 O usuário seleciona a opção tutorial, disponível na tela inicial, onde ele é introduzido e joga um pouco de todos os minigames em seus diferentes modos.
3a.2 Volte ao passo 3.

2b. O usuário teve a necessidade de alterar seu nickname dentro da aplicação.
2b.1 O usuário seleciona a opção de alteração do nickname e realiza a operação.
2b.2 Vá ao passo 3.

2c. O usuário não está cadastrado.
2c.1 O usuário realiza seu cadastro no *Jelly Brick Game*.
2c.2 Volte ao passo 2.

Variantes:
7.1: Sucesso:

7.1.1. O usuário aguenta tempo suficiente para passar para o próximo nível e a dificuldade aumenta. Continue no passo 7.

7.2: Derrota:

7.2.1. O usuário perde o jogo.

7.3: Pontuação:

7.3.1. O usuário conseguiu jogar por mais de 50 minutos sem perder, e anteriormente ele conseguiu chegar à pontuação 110 ou mais nos jogos Balloon breeze e *Simon Fibonacci*, e no *Jelly mode*, sua pontuação foi de 1010 pontos ou mais. É avisado que o jogo *Recursive Calculations* foi liberado.

7.4: Satisfação:

7.4.1. O usuário está satisfeito com sua pontuação, portanto ele aperta o botão de sair e finaliza o jogo.

6.2 Caso de Uso: Jogar *Balloon Breeze*

Caso de Uso: *Jogar Balloon Breeze*
Escopo: *Jelly Brick Game*

Atores: Usuários do *Jelly Brick Game*

Interessados: Usuários de jogos de dispositivos móveis, solicitadores do desenvolvimento do jogo

Pré-condições: O usuário deve possuir *Jelly Brick Game* instalado em seu dispositivo móvel, que pode ser obtido por download através do *Google Play* ou *Apple Store*

Pós condições: O usuário utilizou o aplicativo e está satisfeito com sua pontuação.

Requisitos Correlacionados: B.1, B.2, B.3 e C (de acordo com o documento de requisitos gerado)

Variações teconológicas: O usuário pode controlar a direção em que seu personagem irá planar através do movimento do dedo na mesma direção, bem como selecionar o opção para salvar e sair.

Fluxo Principal:

1. [EV] O usuário carrega o jogo *Jelly Brick Game* em seu dispositivo móvel.
2. [EV] O usuário seleciona seu nome (*nickname*), no início do aplicativo.
3. [RS] O usuário é informado dos modos de jogo: free mode ou *jelly mode*.
4. [EV] O usuário opta por jogar o *free mode*.
5. [RS] O usuário é informado dos 3 diferentes tipos de jogos disponíveis no *free mode*.
6. [EV] O usuário seleciona *Balloon Breeze* e inicia a partida.
7. [EV] O usuário joga:
   - Sucesso: Ver variante 7.1.
   - Derrota: Ver variante 7.2.
   - Pontuação: Ver variante 7.3.
   - Satisfação: Ver variante 7.4.
8. [RS] É informado ao usuário qual foi sua pontuação final.
9. [RS] É fornecido ao jogador a opção de tentar novamente ou escolher outro modo de jogo.
10. [EV] O usuário vai para a tela de menu principal.
11. [EV] O usuário fecha a aplicação.

Tratamento de exceções:

3a. O usuário é novato no aplicativo, e deseja aprender a utilizar o jogo.
3a.1 O usuário seleciona a opção tutorial, disponível na tela inicial, onde ele é introduzido e joga um pouco de todos os *minigames* em seus diferentes modos.
3a.2 Volte ao passo 3.

2b. O usuário teve a necessidade de alterar seu *nickname* dentro da aplicação.
2b.1 O usuário seleciona a opção de alteração do *nickname* e realiza a operação.
2b.2 Vá ao passo 3.

2c. O usuário não está cadastrado.
2c.1 O usuário realiza seu cadastro no *Jelly Brick Game*.
2c.2 Volte ao passo 2.

Variantes:
7.1: Sucesso:
7.1.1. O usuário consegue chegar em terra firme para passar para o próximo nível e a dificuldade aumenta. Continue no passo 7.
7.2: Derrota:
7.2.1. O usuário perde o jogo.
7.3: Pontuação:
7.3.1. O usuário conseguiu chegar à pontuação 110 ou mais, e anteriormente ele conseguiu chegar à pontuação 110 ou mais no jogo *Simon Fibonacci* e aguentou mais de 50 minutos no *Puzzle Challenge*, e no *jelly mode*, sua pontuação foi de 1010 pontos ou mais. É avisado que o jogo *Recursive Calculations* foi liberado.
7.4: Satisfação:
7.4.1. O usuário está satisfeito com sua pontuação, portanto ele aperta o botão de sair e finaliza o jogo.

6.3 Caso de Uso: Jogar *Recursive Calculations*

Caso de Uso: Jogar *Recursive Calculations*
Escopo: *Jelly Brick game*

Atores: Usuários do *Jelly Brick Game*

Interessados: Usuários de jogos de dispositivos móveis, solicitadores do desenvolvimento do jogo

Pré-condições: O usuário deve possuir *Jelly Brick Game* instalado em seu dispositivo móvel, que pode ser obtido por download através do *Google Play* ou *Apple Store* e o usuário cumpriu os seguintes requisitos: no *free mode*, ele durou mais de cinquenta minutos no *Puzzle Challenge* e conseguiu 110 pontos ou mais no *Simon Fibonacci* e *Balloon Breeze* e, no *jelly mode*, ele conseguiu uma pontuação total de 1010 pontos ou mais.

Pós condições: O usuário utilizou o aplicativo e está satisfeito com sua pontuação.

Requisitos Correlacionados: B.1, B.2, B.3 e C (de acordo com o documento de requisitos gerado)

Variações teconológicas: O usuário pode selecionar o resultado que ele considera correto através do

teclado virtual disponibilizado para esse jogo, bem como selecionar o opção para salvar e sair.

Fluxo Principal:

1. [EV] O usuário carrega o jogo *Jelly Brick Game* em seu dispositivo móvel.
2. [EV] O usuário seleciona seu nome (*nickname*), no início do aplicativo.
3. [RS] O usuário é informado dos modos de jogo: free mode ou jelly mode.
4. [EV] O usuário opta por jogar o *free mode*.
5. [RS] O usuário é informado dos 4 diferentes tipos de jogos disponíveis no *free mode*.
6. [EV] O usuário seleciona *Recursive Calculations* e inicia a partida.
7. [EV] O usuário joga:
   - Incremento: Ver variante 7.1.
   - Inativo: Ver variante 7.2.
   - Pontuação: Ver variante 7.3.
   - Expressões: Ver variante 7.4.
   - Satisfação: Ver variante 7.5.
8. [RS] É informado ao usuário qual foi sua pontuação final.
9. [RS] É fornecido ao jogador a opção de tentar novamente ou escolher outro modo de jogo.
10. [EV] O usuário vai para a tela de menu principal.
11. [EV] O usuário fecha a aplicação.

Tratamento de exceções:

3a. O usuário é novato no aplicativo, e deseja aprender a utilizar o jogo.
3a.1 O usuário seleciona a opção tutorial, disponível na tela inicial, onde ele é introduzido e joga um pouco de todos os *minigames* em seus diferentes modos.
3a.2 Volte ao passo 3.

2b. O usuário teve a necessidade de alterar seu *nickname* dentro da aplicação.
2b.1 O usuário seleciona a opção de alteração do *nickname* e realiza a operação.
2b.2 Vá ao passo 3.

2c. O usuário não está cadastrado.
2c.1 O usuário realiza seu cadastro no *Jelly Brick Game*.
2c.2 Volte ao passo 2.

Variantes:
7.1: Incremento:
7.1.1. O usuário consegue acertar 12 ou mais contas e passa para o próximo nível, aumentando a dificuldade. Continue no passo 7.
7.2: Inativo:
7.2.1. O usuário acerta de 8 a 12 contas e permanece no mesmo nível, e a dificuldade também permanece. Continue no passo 7.
7.3: Pontuação:
7.3.1. O usuário, somente se ele estiver abaixo do nível 495, consegue passar 5 níveis, portanto ele é avisado que um novo "N-back" foi liberado. Continue no passo 7.
7.4: Expressões:
7.4.1. O usuário completa os 20 primeiros níveis, portanto ele é avisado que a outra opção "equações" foi liberada. Continue no passo 7.
7.5: Satisfação:
7.5.1. O usuário está satisfeito com sua pontuação, portanto ele aperta o botão de sair e finaliza o

jogo.

6.4 Caso de Uso: Jogar *Jelly Mode*

Caso de Uso: *Jogar Jelly Mode*
Escopo: *Jelly Brick Game*

Atores: Usuários do *Jelly Brick Game*

Interessados: Usuários de jogos de dispositivos móveis, solicitadores do desenvolvimento do jogo

Pré-condições: O usuário deve possuir *Jelly Brick Game* instalado em seu dispositivo móvel, que pode ser obtido por *download* através do *Google Play* ou *Apple Store*.

Pós condições: O usuário utilizou o aplicativo e está satisfeito com sua pontuação.

Requisitos Correlacionados: B.1, B.2, B.3 e C (de acordo com o documento de requisitos gerado)

Variações teconológicas: O usuário pode controlar: a direção em que seu personagem irá planar através do movimento do dedo na mesma direção, para o *Balloon breeze*; a direção da troca dos blocos para direita ou para a esquerda através do toque do dedo e seu movimento na respectiva direção, no caso de jogar o *Puzzle Challenge*; selecionar a ordem que ele considera correta através do toque do dedo sobre os elementos do heptágono para o *Simon Fibonacci*; e selecionar o opção para salvar e sair.

Fluxo Principal:

1. [EV] O usuário carrega o jogo *Jelly Brick Game* em seu dispositivo móvel.
2. [EV] O usuário seleciona seu nome (*nickname*), no início do aplicativo.
3. [RS] O usuário é informado dos modos de jogo: *free mode* ou *jelly mode*.
4. [EV] O usuário opta por jogar o *jelly mode*.
5. [RS] O usuário é informado que o jogo irá começar.
6. [EV] O usuário joga:
   - Sucesso: Ver variante 7.1.
   - Derrota: Ver variante 7.2.
   - Pontuação: Ver variante 7.3.
   - Satisfação: Ver variante 7.4.
8. [RS] É informado ao usuário qual foi sua pontuação final.
9. [RS] É fornecido ao jogador a opção de tentar novamente ou escolher outro modo de jogo.
10. [EV] O usuário vai para a tela de menu principal.
11. [EV] O usuário fecha a aplicação.

Tratamento de exceções:

3a. O usuário é novato no aplicativo, e deseja aprender a utilizar o jogo.
3a.1 O usuário seleciona a opção tutorial, disponível na tela inicial, onde ele é introduzido e joga um pouco de todos os *minigames* em seus diferentes modos.
3a.2 Volte ao passo 3.

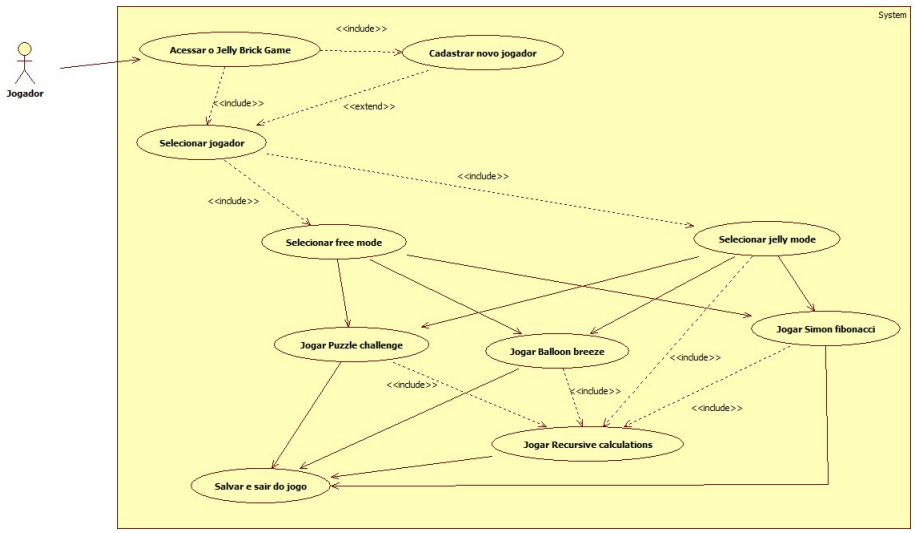
2b. O usuário teve a necessidade de alterar seu *nickname* dentro da aplicação.
2b.1 O usuário seleciona a opção de alteração do *nickname* e realiza a operação.
2b.2 Vá ao passo 3.

2c. O usuário não está cadastrado.
2c.1 O usuário realiza seu cadastro no *Jelly Brick Game*.
2c.2 Volte ao passo 2.

Variantes:
7.1: Sucesso:
7.1.1. O usuário consegue chegar em terra firme para passar para o próximo nível e a dificuldade aumenta. Volte ao passo 5.
7.2: Derrota:
7.2.1. O usuário perde o jogo.
7.3: Pontuação:
7.3.1. O usuário conseguiu 1010 pontos ou mais, e anteriormente ele conseguiu chegar à pontuação 110 ou mais nos jogos *Balloon Breeze* e *Simon Fibonacci* e aguentou mais de 50 minutos no *Puzzle Challenge*. É avisado que o jogo *Recursive Calculations* foi liberado.
7.4: Satisfação:
7.4.1. O usuário está satisfeito com sua pontuação, portanto ele aperta o botão de sair e finaliza o jogo.

6.5 Diagrama de Casos de Uso

A rotina de uso de uma aplicação é, normalmente, representada num diagrama de casos de uso, onde tem-se visão das funcionalidades do *software* num plano de agregações entre si ou com o usuário. Por exemplo, as ações de selecionar um modo de jogo (*free mode* ou *jelly mode*) estão intimamente ligadas com o mini-jogo que o usuário escolherá logo depois, pois se o usuário realmente não quisesse jogar, sequer teria escolhido um modo de jogo, quiçá realmente iniciado o jogo. Portanto, conclui-se que os casos de uso são ligados e procuramos explicitar essas ligações ao máximo neste diagrama de casos de uso:



7 Complexidade do Projeto

Após tantos detalhes minuciosos sobre os requisitos e os usos do *software*, falta abordar aspectos de complexidade deste projeto. Com o mercado de jogos para dispositivos móveis crescendo exponencialmente, o valor do projeto tem de ser extremamente fiel, pois, como estes *softwares* são relativamente

simples, a concorrência é acirradíssima(generalizando, pois, neste caso, o *software* é gratuito, vide orientação do *stakeholder*). Estimativas de tamanho, preço e tempo de trabalho são imprescindíveis para um bom projeto, e essa seção foi oriunda desta necessidade importante.

7.1 Estimativa dos Pontos de Função do *Jelly Brick Game*

7.1.1 *Software* de Apoio – Ferramenta de Contagem

A ferramenta de apoio escolhida foi o *APFPlus* – Análise de Pontos de Função. Esta ferramenta propicia um ambiente prático para realizar métricas baseadas na técnica de Pontos de Função, que consiste em dividir a complexidade dos arquivos e das funcionalidades do *software* em três níveis(alta, média e baixa), de acordo com seu tamanho e dificuldade.

7.1.2 Pontos de Função – Contagem Indicativa

1. **Pontos de Função – Arquivos Do *Software***

Fator	Pontos de Função
Arquivos Lógicos Internos	35
Arquivos de Interface Externa	0
Entradas Externas	–
Saídas Externas	–
Consultas Externas	–
Conversão de Dados(EE)	–
Conversão de Dados(AIE)	–

*Observação: não foram consideradas Entradas Externas, Saídas Externas, Consultas Externas e Conversão de Dados por não haver tamanho significante na aplicação.

2. **Pontos de Função – Funcionalidades Do *Software***

#	Funcionalidade	Pontos da Funcionalidade
1	Comunicação de Dados	4
2	Processamento Distribuído	0
3	Performance	0
4	Configuração Altamente Utilizada	0
5	Volume de Transações	0
6	Entrada de Dados On-Line	0
7	Eficiência do Usuário Final	2
8	Atualizações On-Line	0
9	Processamento Complexo	1
10	Reusabilidade	0
11	Facilidade de Instalação	0
12	Facilidade de Operação	0
13	Múltiplos Locais	0
14	Modificação Facilitada	1
–	**Total**	**8**

3. **Pontos de Função Ajustados**
   Com o conhecimento da fórmula de cálculo de Pontos de Função Ajustados, efetuamos os cálculos necessários.

Como $Pontos\_Função_{Ajustados} = Pontos\_Função_{brutos} * (0,65 + [0,01 * \sum_{i=1}^{14} Pontos\_Funcionalidade_i])$, ficamos com a seguinte tabela final de pontos de função:

Métrica	Pontos Totais
Pontos de Função Não Ajustados	35
Fator de Ajuste	0,73
Pontos de Função ajustados	25,55

7.1.3 Pontos de Função – Contagem Estimada

1. **Pontos de Função – Arquivos Do *Software***

Fator	Pontos de Função
Arquivos Lógicos Internos	14
Arquivos de Interface Externa	0
Entradas Externas	8
Saídas Externas	0
Consultas Externas	4
Conversão de Dados(EE)	0
Conversão de Dados(AIE)	0

2. **Pontos de Função – Funcionalidades Do *Software***

#	Funcionalidade	Pontos da Funcionalidade
1	Comunicação de Dados	4
2	Processamento Distribuído	0
3	Performance	0
4	Configuração Altamente Utilizada	0
5	Volume de Transações	0
6	Entrada de Dados On-Line	0
7	Eficiência do Usuário Final	2
8	Atualizações On-Line	0
9	Processamento Complexo	1
10	Reusabilidade	0
11	Facilidade de Instalação	0
12	Facilidade de Operação	0
13	Múltiplos Locais	0
14	Modificação Facilitada	1
–	**Total**	**8**

3. **Pontos de Função Ajustados**
Com o conhecimento da fórmula de cálculo de Pontos de Função Ajustados, efetuamos os cálculos necessários.
Como $Pontos\_Função_{Ajustados} = Pontos\_Função_{brutos} * (0,65 + [0,01 * \sum_{i=1}^{14} Pontos\_Funcionalidade_i])$, ficamos com a seguinte tabela final de pontos de função:

Métrica	Pontos Totais
Pontos de Função Não Ajustados	26
Fator de Ajuste	0,73
Pontos de Função ajustados	18,98

7.1.4 Pontos de Função – Contagem Detalhada

1. **Pontos de Função – Arquivos Do *Software***

Fator	Pontos de Função
Arquivos Lógicos Internos	7
Arquivos de Interface Externa	0
Entradas Externas	6
Saídas Externas	0
Consultas Externas	3
Conversão de Dados(EE)	0
Conversão de Dados(AIE)	0

2. **Pontos de Função – Funcionalidades Do *Software***

#	Funcionalidade	Pontos da Funcionalidade
1	Comunicação de Dados	4
2	Processamento Distribuído	0
3	Performance	0
4	Configuração Altamente Utilizada	0
5	Volume de Transações	0
6	Entrada de Dados On-Line	0
7	Eficiência do Usuário Final	2
8	Atualizações On-Line	0
9	Processamento Complexo	1
10	Reusabilidade	0
11	Facilidade de Instalação	0
12	Facilidade de Operação	0
13	Múltiplos Locais	0
14	Modificação Facilitada	1
–	**Total**	**8**

3. **Pontos de Função Ajustados**
   Com o conhecimento da fórmula de cálculo de Pontos de Função Ajustados, efetuamos os cálculos necessários.
   Como $Pontos\_Função_{Ajustados} = Pontos\_Função_{brutos} * (0,65 + [0,01 * \sum_{i=1}^{14} Pontos\_Funcionalidade_i])$, ficamos com a seguinte tabela final de pontos de função:

Métrica	Pontos Totais
Pontos de Função Não Ajustados	16
Fator de Ajuste	0,73
Pontos de Função ajustados	11,68

Seguem juntamente com este relatório as imagens referentes à ferramenta APFPlus que exibem estes resultados nitidamente no próprio *software*, para eliminar quaisquer dúvidas provenientes destas tabelas.

8 Conclusão

Por ser um projeto muito simplista e de cunho acadêmico, algumas considerações podem não ser totalmente fiéis a *softwares* do gênero, pois há muito mais elementos a serem explorados a fundo que não possuem um impacto muito grande neste projeto. Um jogo atrativo necessita muito tempo de estudo de mercado e de preferências gerais, a fim de gerar algo simples e contagioso.

Smash hits como *Angry Birds*, *Candy Crush* e o mais novo *Flappy Bird* são exemplos de jogos para as plataformas *mobile* de sucesso, que visam viciar o usuário a ponto de descontentamento próprio, o que os leva a jogar num ciclo praticamente infinito. A tabela de pontuações exerce sua função nesta hora, quando um mini-jogo termina e uma mensagem surge na tela, dizendo que a pontuação do usuário é menor que a de algum amigo que também joga o jogo, o que gera uma certa "discórdia" e faz com que o jogador se esforce mais, às vezes até o ponto de usar dinheiro para melhor suas estatísticas.

Desses fatos, tiramos que uma especificação minuciosa e completa deve ser muito bem preparada, para que a equipe consiga desenvolver casos de uso aplicáveis e ideias sólidas, para que o projeto não falhe ou caia em desuso muito rapidamente. A estrutura de cada uma das fases também há de ser extremamente competente − a documentação do projeto precisa ser concisa e informativa, caso contrário, um problema grande pode ocorrer em qualquer parte do projeto, proveniente de desentendimentos quanto aos requisitos do produto.

Além das práticas profissionais, há também o fator pessoal: uma pessoa introvertida provavelmente não tem vez neste mercado feroz de engenheiros de *software*, pois a necessidade de interação humana transcende as necessidades básicas e profissionais ao ponto de faça ou morra. Literalmente! Uma má forma de expressão, má conduta e má escrita são fatores de enorme peso e são passíveis de demissão. Nenhuma empresa precisa de um engenheiro de *software* que faça com que uma equipe inteira de desenvolvimento se perca, em vez de lhes trazer a luz. Sentir este trabalho crítico na pele nos fez atinar para questões que antes passavam despercebidas quando realizando trabalhos, como forma de escrita, forma de representação de dados e até mesmo veículos de informação usados quando procurando informações importantes.

Ainda assim, mesmo com uma realização "impecável" de todos os passos de engenharia de *software*, há algo que possa ser melhorado. Não há nada mais aplicável a computação e sociedade do que a engenharia de *software* e estas peculiaridades das modas da sociedade só vem à tona quando em contato frequente com ela. E, mesmo assim, ainda existirão pessoas que não gostarão do produto, normalmente por causa do tabu atribuído a profissionais na área de computação.

Por fim, conseguimos entender a dificuldade e o peso das relações humanas e de práticas modernas na construção de novas aplicações. Sem elas, não haveria um escopo bem definido sobre o que se pode fazer e como se deve fazer, levando a desastres comerciais e de produção. Diminuir a taxa de degradação do *software* e tornar a curva de falhas muito mais tênue são objetivos constantes e que não podem ser obtidos sem estas práticas, como comprovado no período da Crise de *Software*. Uma equipe competente e aplicada nunca deixará de ser a chave pro sucesso.