Chapter 1

Construção de Jogos Digitais Sérios para Processos de Serviços Públicos

Tadeu Moreira de Classe, Renata Mendes de Araujo, Geraldo Bonorino Xexéo

Abstract

The purpose of this course is to enable students to build a digital serious game based on a real public service process using the Construct 2 tool for game development. The course presents digital and serious games concepts, as well as business process concepts and process model elements, and the opportunities and challenges of designing these games. The game design process is based on a method of digital game design using public services process, which will be presented to the students.

Resumo

O objetivo deste minicurso é possibilitar que os alunos construam um jogo digital sério baseado em um processo de serviço público real, usando a ferramenta de criação de Jogos Construct 2. O curso apresenta os conceitos de jogos digitais e jogos sérios, as possibilidades e desafios do design destes jogos, bem como sucintamente os conceitos de processos de negócio e seus elementos constituintes. A construção destes tipos de jogos se baseia em um método de design de jogos digitais baseados em processos de serviços públicos, o qual também será apresentado ao público, a fim de contextualizar a etapa de codificação e desenvolvimento do jogo, foco deste minicurso.

1.1. Introdução

As tecnologias digitais estão em constante desenvolvimento, sendo cada vez mais presente na vida das pessoas e organizações. Seguindo tal tendência, as instituições, públicas ou não, precisam acompanhar esta evolução de modo a garantir a competitividade de seus negócios e/ou oferecer melhores serviços. Com o aumento do uso de tecnologias sociais, oportunidades inovadoras de diálogo entre organizações e seus clientes se apresentam, visando à melhoria de processos, ampliando a participação de pessoas tanto interna como externamente às organizações, na contribuição com tais melhorias [Pflanzl e Vossen 2014].

Neste sentido, há uma crescente necessidade de transparência e visibilidade dos processos organizacionais para uma maior interação entre as pessoas e as organizações. Para alcançar este objetivo, as pessoas antes de tudo precisam compreender e aprender sobre os processos organizacionais, sobretudo os de prestação de serviços. O uso de jogos digitais surge como uma alternativa em potencial a este ideal de aproximação, uma vez que jogos possuem a característica de engajamento de seus jogadores e têm sido utilizados como soluções para aprendizado em diversas áreas [Classe et al. 2016].

Em se tratando de contextos públicos, os jogos digitais podem representar uma nova maneira dos cidadãos compreenderem e aprenderem sobre os diversos processos de serviços prestados por entidades públicas, conscientizando-se sobre eles.

Este minicurso tem como foco os processos de prestação de serviços públicos e no design de jogos digitais baseados nestes processos. Neste curso os alunos irão construir um jogo digital sério, com uma única fase do gênero aventura. Este jogo irá se basear em um processo de prestação de serviço público real, o processo de Emissão do Cartão SUS[1], e terá como objetivo explicar os detalhes para que o cidadão possa solicitar e usufruir deste serviço. Para a construção do jogo será usada a ferramenta *Construct2*, sendo demonstrado o passo-a-passo para os alunos construírem o jogo digital proposto. Todos os materiais (assets, imagens, áudios, cenários etc.) necessários para a construção do game serão fornecidos. Ao final do minicurso os alunos saberão conceitos fundamentais sobre: i) prestação de serviços públicos; ii) elementos de modelos de processo, iii) conceitos e etapas para o design de jogos digitais, iv) atores envolvidos em design de jogos, v) principais *engines* de desenvolvimento de jogos digitais; e vi) uma visão do método de design de jogos digitais baseados em processos de serviços públicos, proposto por Classe, Araujo e Xexéo (2018).

As seções deste capítulo estão organizadas para a melhor compreensão das abordagens teóricas e práticas do curso. A Seção 1.2 aborda o referencial teórico do curso trazendo temas como democracia digital, modelagem de processos de negócio e jogos digitais. A seção 1.3 aborda o método de design de jogos baseados em processos de serviços públicos, apresentados as suas etapas. A seção 1.4, demonstra a criação de um jogo para o processo real de emissão do Cartão SUS. E finalmente, a seção 5 apresenta a conclusão do curso.

1.2. Referencial Teórico

Nesta seção são apresentadas as bases teóricas que motivaram este estudo, compreendendo as premissas e conceitos básicos envolvidos na proposta de design de jogos digitais baseados em processos de serviços públicos.

1.2.1. Democracia Digital

A democracia digital (também conhecida na literatura como democracia eletrônica ou ainda e-democracia) entende a internet como agente de transformação dos processos políticos e de prestação de serviços públicos tradicionais [Vedel 2006]. Silva [2005] define o conceito de democracia digital como "o conjunto de discursos, teorizações e experimentações que empregam Tecnologias de Informação e Comunicação (TICs) para mediar as relações políticas, tendo em vista as possibilidades de participação

[1] http://portalms.saude.gov.br/acoes-e-programas/cartao-nacional-de-saude

democrática nos sistemas políticos contemporâneos". Sendo esta última a definição que consideramos para este trabalho [Araujo e Magdaleno 2015].

É inegável que o uso de TICs ajuda na promoção do engajamento de cidadãos, na construção de políticas e na participação social, embora devamos reconhecer que a tecnologia é apenas um meio para viabilizar a solução, e não a solução por completo. Sendo assim, para a implementação da democracia digital, barreiras culturais, organizacionais, constitucionais, políticas e também tecnológicas devem ser transpostas [Kneuer 2016][Santos 2014].

Dentre um dos objetivos mais básicos dos governos se encontra a entrega e prestação de serviços públicos para a sociedade [Bertot et al. 2016]. Entretanto, isso é uma tarefa extremamente desafiadora devido às crescentes desigualdades sociais e econômicas, à diversidade social, ao quantitativo populacional, à carência de recursos, à dependência de cenários políticos, dentre outras [Bertot et al. 2016]. Devido a estes desafios, é necessário compreender que as agências prestadoras de serviços públicos possuem um papel complexo na medida em que precisam balancear a eficiência de seus processos com os interesses públicos [Daniels 2016].

As mudanças nas formas de prestação dos serviços ocasionadas pelos ideais de democracia digital, são um ponto crítico para o fornecimento e entrega de serviços públicos, pois os cidadãos demandam maior agilidade e transparência em sua prestação [Brown et al. 2017]. A falta de recursos, de gerenciamento adequado, e a forma como as informações sobre a utilização dos recursos públicos são passados para o cidadão, influencia diretamente na forma e na qualidade que os serviços são entregues à sociedade [Winters et al. 2014].

1.2.2. Prestação de Serviços Públicos

Grönlund [2010] defenda que a função do governo não é somente a prestação de serviços, e que este inclui outras características de relacionamento com o cidadão, como confiança, equidade e outros aspectos que não pertencem somente ao quadro de serviços. Os serviços públicos são produzidos por instituições nacionais, estaduais e locais, com o objetivo de serem entregues aos cidadãos, organizações e entidades em suas jurisdições [Bertot et al., 2016]. O fornecimento de serviços públicos pelos governos pode ser considerado como uma obrigação moral, prevista nos direitos humanos tanto quanto o direito à água, comida, energia, segurança e saúde [Bertot et al., 2016].

Porém, a prestação de serviços públicos para a sociedade é extremamente desafiadora [Tavares et al., 2016]. As crescentes desigualdades sociais e econômicas, a diversidade social, dentre outras, torna mais importante do que nunca a prestação contínua dos serviços públicos essenciais a todos os cidadãos, isto é, independentemente dos seus status na sociedade, o que contribui significativamente como complexidade de sua prestação com eficiência [Bertot et al., 2016][Tavares et al., 2016]. Devido a estes desafios é necessário compreender que as agências prestadoras de serviços públicos possuem um papel mais complexo do que as organizações privadas, pois necessitam balancear a eficiência de seus processos com os interesses públicos [Daniels, 2016]. A prestação de serviços públicos requer várias formas de inovação, sendo necessário que funcionem juntas as agências governamentais, não governamentais, particulares, universidades, cidadãos e outros atores envolvidos em todo o processo. Isso traz o

serviço para perto do seu usuário, de maneira que estes mesmo usuários possam aprender com os serviços [Tavares et al. 2016].

Embora atualmente haja este ideal de melhoria e transparência na prestação de serviços públicos, Winters et al. [2014] e Tavares et al. [2016] questionam, o porquê de os governos falham em prover os serviços básicos, com agilidade e qualidade para o cidadão. Os autores argumentam que a falta de recursos é um fator que contribui bastante para esta falha, e quando estes recursos existem, existem problemas na forma com que são geridos pelas instituições públicas. Esta falta de gerenciamento adequado, é associado também a falta ou à forma como as informações sobre a utilização dos recursos públicos são passados para o cidadão, devido a inexistência de meios eficazes para que o cidadão possa monitorar o governo. Essa falta de transparência e gestão de recursos públicos se associa à corrupção, que influencia diretamente na forma e na qualidade que os serviços são entregues à sociedade.

1.2.3. Gestão de Processos de Negócio

Independentemente do tipo da organização, todas elas possuem um conjunto de atividades que se relacionam no intuito de desenvolver um produto ou serviço para determinado cliente. Estas atividades interligadas compõem os processos, os quais ocorrem em diferentes níveis organizacionais - desde as atividades operacionais até as de níveis estratégicos do negócio. Sendo assim, não existe produto ou serviço oferecido por uma organização sem que para isso exista um processo relacionado a eles [Souza e Medeiros, 2008]. Neste sentido, Sharp e Mcdermott [2009] definem que um processo de negócio são tarefas de trabalhos inter-relacionadas, começadas a partir de uma resposta a algum evento, para que seja atingido um resultado específico ao cliente e outros envolvidos com o processo.

Os processos são executados pelas empresas visando entregar um produto ou serviço que agregue valor a seus clientes. A maneira com que os processos são definidos e executados influenciam diretamente na qualidade do produto ou serviço a ser entregue [Dumas et al. 2013]. Existem vários tipos de processos de negócios, esses tipos são vinculados ao ramo de cada organização, podendo ser tanto processos de negócios privados ou públicos. Os processos de negócio privados são aqueles internos às organizações, podendo fazer parte de processos estratégicos (processos onde são decididos os objetivos da organização, políticas, mudanças no planejamento e utilização de recursos), processos gerenciais (processos onde os gerentes garantem que recursos são obtidos e usados com eficiência e efetividade para atingir os objetivos organizacionais), ou processos operacionais (processos onde tarefas específicas são utilizadas com eficiência e eficácia no dia-a-dia organizacional). Já os processos de negócio públicos (também conhecidos como: processos de negócio colaborativos), são aqueles que envolvem organizações (ou pessoas) externas à empresa, seu relacionamento e a forma que interagem [Ko, 2009].

Business Process Management (BPM), ou em português, Gerenciamento de Processos de Negócio (GPN), é a arte e ciência de analisar como o trabalho dentro de uma organização é realizado para assegurar resultados positivos e oportunidades de melhorias nas suas atividades [Dumas et al. 2013]. Sendo assim, a GPN busca revisar e orientar a realização dos trabalhos executados em uma organização, definindo oportunidades de melhorias na execução das tarefas e no alcance aos objetivos, como redução de custos, agilidade de execução, ou melhoria na qualidade de produtos ou

serviços. Além disso, a área reúne princípios, métodos e ferramentas para projetar, analisar, executar e monitorar os processos de negócio [Dumas et al. 2013].

Uma característica constantemente associada à GPN é o seu ciclo de vida [Dumas et al. 2013], o qual descreve as fases que apoiam a gestão de processos de negócio. O ciclo de vida de GPN envolve as seguintes fases (Figura 1) de acordo com Dumas et al. [2013]: **Identificação de Processos; Descoberta de Processo; Análise do Processo; Redesenho de Processo; Implementação do Processo; e Monitoramento e Controle do Processo**.

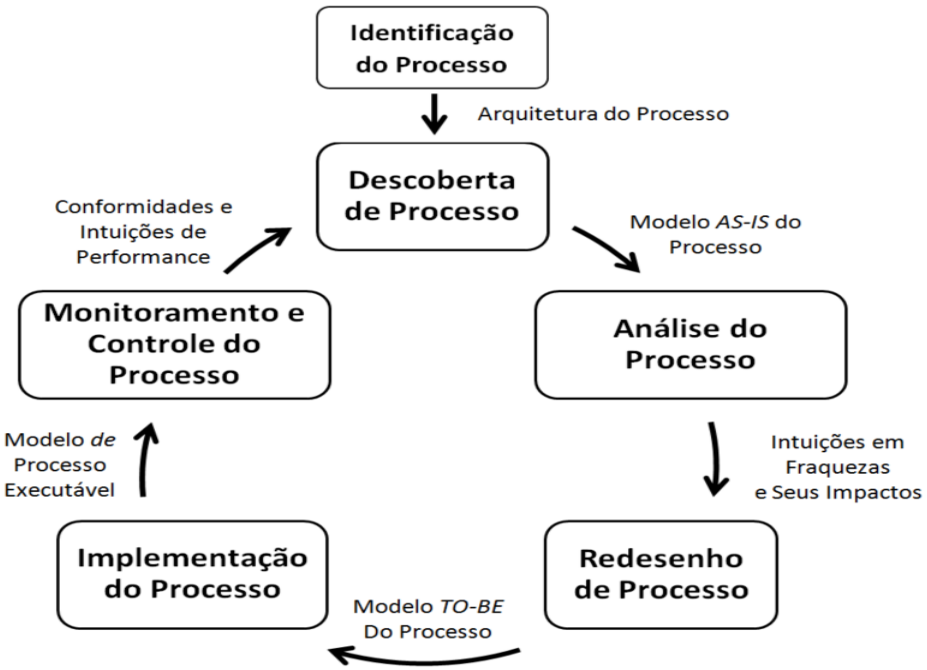


Figura 1 - Ciclo de Vida GPN [Traduzido de Dumas et al., 2013]

1.2.3.1 Modelagem de Processos de Negócio

A modelagem de processos de negócio está presente dentro das fases do ciclo de vida do BPM, no qual está presente a formalização de um modelo do processo de negócio, seus objetivos, métricas, fluxos, dados, integrações e relacionamentos entre áreas [Sobreira Neto, 2009]. Um modelo de processo de negócio é uma representação gráfica que simplifica o entendimento relacionado à execução sequencial das atividades que compõem o processo organizacional. Para representar estes modelos, existem linguagens de modelagem de processos. Dumas et al. [2013] discorrem que o propósito do modelo deve considerar quem deve utilizá-lo, a fim de abstrair certos elementos e objetos a serem modelados. Sendo assim, uma linguagem de modelagem de processos tem por finalidade a representação dos elementos destes processos.

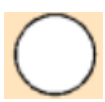
Os especialistas recorrem ao modelo de processo para fornecer uma compreensão de como o mesmo funcionam, como se relaciona com os componentes organizacionais, seus objetivos e regras, no intuito de que eles possam ser executados de forma mais simples e eficiente o possível, permitindo, assim sua análise e melhoramento [Aguilar-Saven, 2004]. O propósito do modelo é considerar quem deve compreendê-lo, a fim de abstrair certos elementos a serem modelados [Dumas et al, 2012].

Um modelo de processo é transcrito pode meio de alguma linguagem, está podendo ser uma linguagem natural falada ou escrita, ou um conjunto de elementos específicos que associados podem fornecer significados para quem conhece tais significados. Uma linguagem de modelagem de processos tem como foco a representação dos elementos de um processo e seu funcionamento. Essas linguagens focam nas mudanças contínuas dos objetos do processo, ou seja, em um modelo pode ser mostrado como as atividades se modificam no decorrer da execução e com as interações entres os atores do processo ocorrem [Reijers et al., 2013].

Dentre as várias linguagens de modelagem de processos de negócio a *Business Process Modeling and Notation* (BPMN) é uma das mais utilizadas, fornecendo um padrão entre o design de processos de negócios e a implantação destes processos. Este modelo representa práticas da comunidade de modelagem de processos de negócio, formalizando notações e semânticas sobre diagramações de processos. [OMG, 2013].

Atualmente em sua versão 2.0.2, o BPMN é reconhecido como um padrão internacional, compreendendo: a formalização de semântica para todos os seus elementos; a definição de mecanismos tanto para modelos de processos quanto para extensões gráficas; o uso de composição de eventos e suas relações; a redefinição das interações humanas; e a definição de modelos de coreografia. Tal padronização permite que o BPMN possa ser mapeado para mais de uma plataforma, fornecendo o padrão intercambiável para linguagens de modelagem de processos, com simplicidade de compreensão de seus elementos (Tabela 1) [OMG, 2013].

Tabela 1 – Principais Elementos BPMN

Título	Elemento	Descrição
Tarefa		Tarefa representa uma ação que pode ser executada por uma pessoa ou sistema dentro do fluxo do processo.
Sub-Processo		Sub-Processo representa uma abstração de um conjunto lógico (sequência) de atividades com um propósito específico.
Gateway Exclusivo		Gateway exclusivo fornece seguimento a uma condição exclusiva, no qual apenas um dos caminhos será seguido de acordo com uma condição testada.
Gateway Paralelo		Gateway paralelo divide o fluxo em dois ou mais, sendo executados todos os caminhos paralelamente.
Gateway Inclusivo		Gateway Inclusivo dá segmento ao fluxo por uma condição inclusiva, ou seja, pode haver uma combinação de um ou mais caminhos a serem seguidos dependendo da condição verificada.
Evento Inicial		Os eventos iniciais marcam pontos de início de um processo. Um processo pode apresentar diferentes pontos de início.

Evento Final		Eventos finais marcam onde o fluxo do processo terminar. Um processo pode apresentar diferentes pontos finais.
Evento Intermediário		Eventos intermediários indicam um ponto no fluxo do processo em que é planejado o acontecimento de algum evento, podendo mudar a sequência de ações.
Piscinas		Uma piscina é um contêiner de um processo de negócio. É permitida apenas uma piscina em um processo, no qual seu nome representa o mesmo. Uma piscina que não revela seu processo recebe o nome de "*pool black box*".
Raias		As raias são subdivisões da piscina, que podem ser usadas para representar um papel ou uma área organizacional responsável por uma tarefa no processo.
Objeto de Dados		Objeto de dados são informações cujas representações sejam necessárias e importantes para a compreensão do fluxo do processo.
Repositório de Dados		Repositório de dados representam informações de qualquer espécie, podendo ser bancos de dados, sistemas de informações e etc.
Fluxo Sequencial		Representa o fluxo, a sequência em que as atividades são executadas no processo.
Fluxo de Mensagem		Representa um fluxo de mensagens e é usado para mostrar a comunicação entre duas entidades no processo.
Associação		Associa artefatos aos elementos do fluxo do processo.

1.2.4. Jogos Digitais

Diversos autores definem o que são jogos. Crawford [2003] define que jogos devem ser interativos, possuir objetivos, permitir a competição e que um jogador possa atacar o outro de alguma maneira. Salen e Zimmerman [2003] possuem a definição que jogos são sistemas onde os jogadores estão engajados em um conflito artificial, regidos por regras que possam gerar um resultado quantificável. De acordo com Adams e Rollings [2007], jogos são atividades reais onde os jogadores tentam alcançar objetivos, guiando-se por regras pré-estabelecidas de maneira voluntária. Segundo Juul [2009], os jogos devem possuir regras fixas, resultados variados e valorizados, consequências negociáveis e ligações entre o jogador e os resultados.

Nesta linha, é proposta a seguinte definição a ser utilizada neste trabalho: "Jogos são atividades voluntárias com significância, altamente imersivas, onde os jogadores

estão engajados em conflitos em busca de seus objetivos, modificando interativamente de maneira quantificável um sistema artificial através de decisões e ações, sendo que todo este processo é regido por regras, o que normalmente resulta em diversão e entretenimento sobre adversários ou desafios" [Xexéo et al., 2017].

Em se tratando de jogos digitais, Schuytema [2008] define que jogos eletrônicos são atividades lúdicas formada por ações e decisões que geram uma condição final, sendo que as decisões são comandadas e limitadas por um conjunto de regras, as quais são regidas por um programa de computador. Complementarmente Marston [2015] dizem que jogos digitais são programas interativos que permitem um ou mais jogadores se engajarem em um propósito de entretenimento, através de dispositivos como computadores, videogames, aparelhos de TV e telefones, por exemplo [Ilomäki e Kankaanranta, 2009].

Desta maneira, para este trabalho, entendemos como definição de jogos digitais: "Jogos Digitais são atividade voluntárias com significâncias, altamente imersivas, onde os jogadores são engajados em conflitos em busca de seus objetivos, modificando interativamente de maneira quantificável um sistema artificial através de decisões e ações, sendo que todo este processo é regido por regras, controladas por programas de computador, executados por dispositivos digitais como computadores, videogames, aparelhos de TV e telefones, resultando normalmente em diversão e entretenimento".

Em geral, os jogos servem ao propósito de entretenimento, porém eles possuem um grande potencial para dar suporte à socialização, à educação e treinamentos [Michael e Chen, 2005]. Neste sentido, oportunidades que utilizam jogos e/ou seus elementos surgem, como a gamification (gamificação ou ludificação no Brasil), que consiste no uso de elementos e mecânicas de jogos em contextos que anteriormente não eram "jogáveis" [Deterding et al., 2011]; e os serious game (jogos sérios), jogos desenvolvidos para contextos sérios [Abt, 1970].

1.2.4.1 Jogos Sérios

Jogos sérios não é um tema relativamente novo, uma das suas primeiras menções foi por Abt [1970], o qual é considerado o primeiro autor a definir o termo. Segundo ele, jogos sérios são jogos em contextos sérios, no qual existe um compromisso com o caráter educacional e de treinamento, e não destinados simplesmente para o entretenimento. Neste sentido Michael e Chen [2005] em seu livro reafirmam o conceito de Abt ao escrever que um jogo sério tem como objetivo primário a educação (em seus vários formatos, sendo treinamento, conscientização, dentre outros), além do entretenimento. Ou seja, em sua definição mais simples, jogos sérios implicam em dizer que entretenimento e diversão não são o foco destes jogos, o que não quer dizer que estas características não possam estar presentes, onde nestes jogos é necessário conter elementos lúdicos, com o propósito de tratar problemas do mundo real Rocha e Araujo [2013].

Jogos sérios são jogos que usam seu meio artístico para transmitir mensagens, ensinar lições ou fornecer alguma experiência. Desta maneira, o termo "sério", não implica que o jogo seja maçante, mas sim o de refletir o propósito a que o jogo é criado [Michael e Chen, 2005]. Eles são ferramentas inovadoras amplamente conhecidas por seu potencial de apoiar a aprendizagem [Romero et al., 2015], podendo ser usados não

somente para fins puramente educacionais, onde podem contribuir para obter objetivos definidos [Susi et al., 2007].

Nos últimos anos, a literatura sobre jogos sérios apresenta crescimento significativo, pois seu estudo tem se demonstrado promissor para as técnicas de educação e treinamento, principalmente em se tratando das atuais gerações. Existem várias classificações sobre os diversos tipos de jogos sérios, de acordo com o propósito de cada um deles. De acordo com Connolly et al. [2012], é possível classificar os jogos sérios pelos seus propósitos, mídias, tecnologias utilizadas, gênero, área, impactos esperados, resultados habilidades necessárias ou conquistadas e pelo tipo de mudança comportamental. Entretanto, de acordo com [Alves, 2013], ainda não existe uma classificação formal amplamente aceita pela comunidade, embora ele destaque em seu trabalho algumas das classificações mais conhecidas (Tabela 2).

Tabela 2 – Classificação de Jogos Sérios

Tipo de Jogos Sérios	Descrição
Advergames	Jogos utilizados na promoção de alguma marca, produto, organização ou ponto de vista.
Edutainment	Jogos projetados com objetivo educacionais e também entretenimento.
Game-based Learning	Jogos com objetivos de aprendizagem, projetados de forma a equilibrar o componente lúdico e didático.
Newsgames	Jogos jornalísticos destinados a reportar acontecimentos recentes ou envio de comentários editoriais sobre acontecimentos.
Training and Simulation Games	Jogos de simulação que tentam abordar as atividades da vida real com o maior grau de exatidão possível. Normalmente utilizados para a aquisição ou exercício de habilidades.
Persuasive games	Jogos que persuadem os jogadores por meio da jogabilidade, normalmente projetados para mudar atitudes e comportamentos dos jogadores por meio de persuasão.
Organizational dynamic	Jogos projetados para promover o desenvolvimento pessoal e a formação do caráter.
Games for Health	Jogos utilizados para educação em saúde, simulações médicas, terapias e reabilitação de pacientes.
Art games	Jogos que expressam ideias artísticas ou arte produzidas por meio de jogos digitais.
Militainment	Jogos utilizados para fins militares por meio de simulações de operações com o alto grau de precisão.

1.2.4.2 Design de Jogos Digitais

Em se tratando do desenvolvimento de jogos digitais, é necessário pensar um pouco além de somente programação, pois o mesmo envolve também um grande trabalho criativo alinhados às complexas implementações de software. É necessário que exista um bom balanceamento entre arte e programação, do contrário o jogo pode não atingir os objetivos esperados. Por exemplo, um jogo com gráficos exuberantes que contenha

regras desconexas pode não oferecer uma boa experiência ao jogador, o qual irá abandoná-lo [Mastrocola, 2012].

Desta maneira, por serem programas de computador, é comum que desenvolvedores de jogos digitais usem métodos tradicionais de engenharia de software para sua construção. Contudo, as tradicionais metodologias de desenvolvimento de sistemas precisam ser adaptadas para o design jogos digitais, utilizando métodos que possam auxiliar a etapas de desenvolvimento do projeto do jogo, sua concepção até a sua finalização.

Pensando nisso, alguns autores como Zimmerman [2003], Adams e Rollings [2007] e outros, escreveram sobre suas experiências e seus processos e métodos que contribuíram para a concepção de jogos, seu desenvolvimento e meios de avaliação ao longo de todo o processo. Zimmerman [2003] propõe um ciclo interativo para o desenvolvimento de jogos, afirmando que "[...] o design interativo é uma metodologia baseada no processo cíclico de prototipação, análise e refino de um produto ao longo do processo. ". Desta maneira, a cada ciclo, o design anterior do jogo pode ser usado para sua melhoria e até mesmo para a concepção de novos jogos.

Na indústria de desenvolvimento de software o método de Schell [2009] é bem aceito. Ele subdivide os componentes de design em quatro grandes categorias (Figura 2), os quais devem se apresentar de forma balanceada no jogo. Estes quatro elementos básicos são: estética (*aesthetics*), mecânicas (*mechanics*), narrativa (*story*) e tecnologia (*technology*). Neste modelo, o autor descreve como **Estética** os elementos que dão aparência sensorial ao jogo (imagens, sons e etc.), fortemente relacionada à experiência do jogo, imergindo o jogador em seu ambiente. As **Mecânicas** são os procedimentos, as regras do jogo, os objetivos do jogo, o que um jogador pode ou não fazer e o que acontece quando ele atinge ou não um objetivo. As **Narrativas** são sequências de eventos, no qual se desdobram o jogo. É neste elemento em que podem ser apresentados os personagens e o contexto que o jogador interage com os elementos. Neste sentido, as mecânicas ajudam a contar a história e a estética ajuda a reforçar as ideias da história. Finalmente, a **tecnologia**, é o meio no qual a história é contada, implementando as mecânicas do jogo e criando a estética. A tecnologia pode ser criada, uma vez que a mesma não exista.

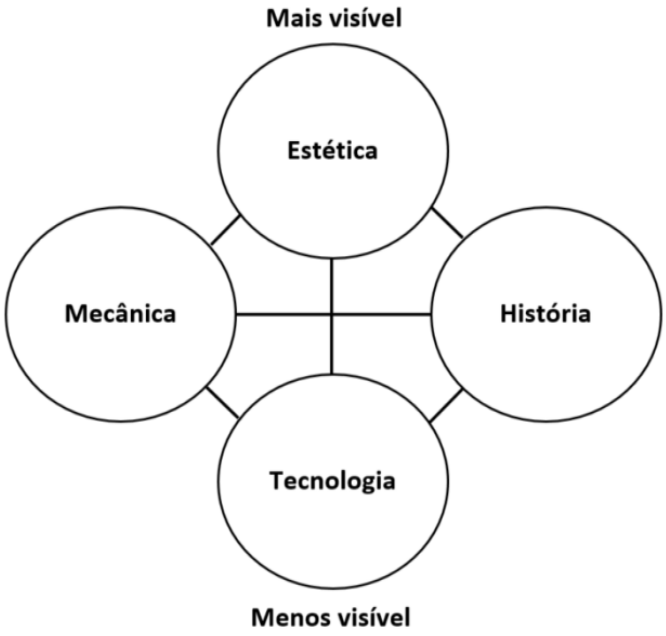


Figura 2 - Elementos de Design de Jogos [Schell, 2009]

1.3. Método de Design de Jogos Digitais Baseados em Processos de Serviços Públicos

Diferentemente dos métodos tradicionais de design de jogos digitas ou de jogos sérios, jogos baseados em processos de negócio (inclusive os processos de prestação de serviços públicos) precisam usar como fonte primária de inspiração os processos organizacionais e seus modelos. Portanto o método de design de jogos digitais baseados em modelos de processos de negócio (Figura 3), parte da existência de um modelo de processo de negócio para o design de jogos digitais.

O método é iterativo, ou seja, permite o retorno a etapas anteriores no caso de os designers dos jogos sentirem a necessidade de rever alguma informação de etapas já executadas. O método conta com as etapas de: i) estudo do contexto do processo; ii) mapeamento de elementos do modelo do processo para elementos de design do jogo; iii) projeto do jogo; iv) desenvolvimento e prototipação do jogo; v) validação com equipe de design; vi) validação com executores do processo; vii) validação com público alvo; e, viii) empacotamento e entrega do jogo. Estas etapas serão explicadas nas seções a seguir.

1.3.1 Estudo de Contexto

O método de design tem como entrada um modelo de um processo de negócio do processo que se deseja transformar em jogo. Este modelo de processo precisa ser compreendido pelos designers do jogo. Nesta etapa, o foco não é compreender todas as informações do processo, mas especificamente informações qualitativas como o contexto, valores, abordagens sociais, por exemplo, que não estão expressas em um modelo gráfico do processo. Além disso, esta etapa objetiva, também, discutir e organizar as informações sobre o contexto do processo de negócio, como ele é executado, qual são as percepções dos seus usuários ao solicitá-los, qual sua relevância, performance e qualidade. Estas informações são muito importantes para que os

designers consigam pensar em informações do jogo como ambiente, comportamento dos personagens, e outras informações relevantes.

Neste sentido, questões como: “O que é o processo?”, “Por que o processo existe?”, “Como o processo é executado?”, “Quem são os clientes do processo?”, “Como os clientes experenciam o processo?”, “Quais são as principais tarefas que precisam ser compreendidas no processo?”. São agrupadas em três categorias: **i) contexto organizacional**, a fim de entender critérios organizacionais (valores e missão, estratégia organizacional, clientes e etc.); **ii) contexto do processo**, a fim de compreender os detalhes do processo (motivação, objetivos, desafios, principais atividades, atores e etc.); e, **iii) contexto do usuário**, que investiga a experiência do usuário sobre o processo (sentimentos, frustrações, pensamentos e etc.).

Todas estas informações devem ser registradas em um documento (documento de contexto), o qual é útil para a contextualização do projeto do jogo, além de ajudar a compreensão do processo, permitindo que os designers do jogo entenderem particularidades do processo. Posteriormente, este documento também será útil para pensar em situações de fluxos de jogo, como histórias e fluxo temporal do jogo.

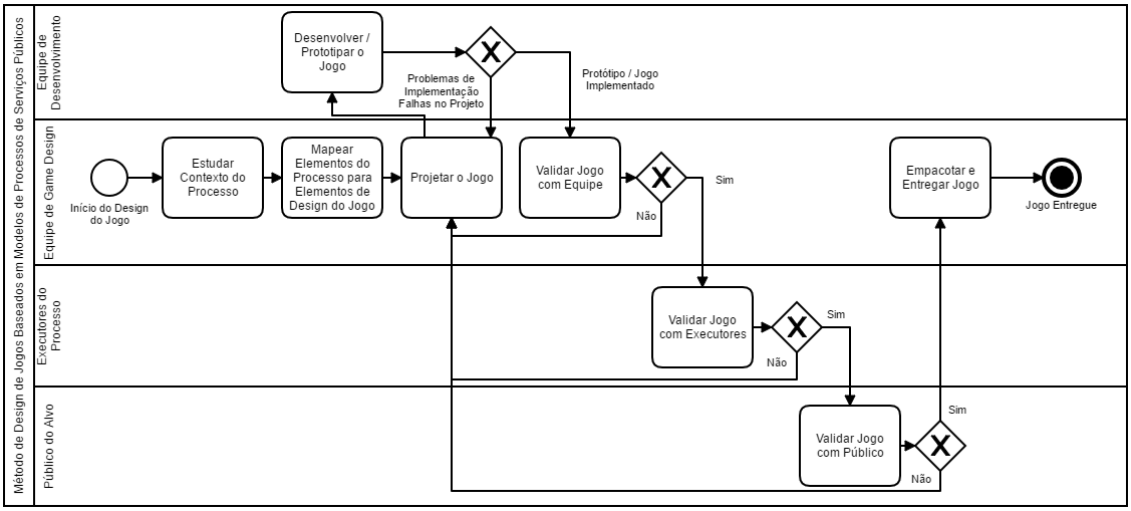


Figura 3 – Método de Design de Jogos Digitais Baseados em Modelos de Processos de Negócio

1.3.2 Mapeamento de Elementos

O mapeamento de elemento é a etapa do design do jogo que permite, a partir de um modelo de processo de negócio (considerando modelos gráficos e documentais), e um gênero de jogo digital, ser possível associar elementos (elementos do modelo do processo e elementos do gênero do jogo) a partir de seu significado, realizando o cruzamento de informações entre eles, que permite construir um documento de mapeamento de elementos (Anexo 1). A partir deste documento de mapeamento é gerado um documento de design de jogos (GDD – *game design document*), o qual é um documento de requisitos necessário para auxiliar a equipe de design do jogo na construção de jogos sérios de processos de negócio.

1.3.3 Projeto do Jogo

A etapa de projeto de jogo é inspirada na visão de design de jogos de Schell [2009]. Nesta etapa, o objetivo principal é melhorar a versão inicial do GDD gerado na etapa anterior com criatividade, no qual devem ser pensados elementos como: a temática do jogo, público, história e mecânicas, por exemplo. Portanto, apesar de o método propor a sistematização do design de jogos baseados em processos de negócio, é necessário utilizar a criatividade dos designers de jogos. Um jogo baseado em processo de negócio deve estar entre o determinismo de um processo de negócio, e a ludicidade de um jogo. Não deve ser somente uma simulação do processo, e nem somente um jogo de entretimento.

Para melhorar o documento de design a equipe de game design precisa refletir sobre: **i) definição do público alvo** (escolha do público do jogo); **ii) definição do gênero do jogo** (qual gênero de jogo mapeado pode representar melhor o processo a ser transformado em jogo); **iii) temática do jogo** (seleção do tema do jogo); **iv) personagens** (personagens relacionados aos atores do processo e suas características); **v) narrativa** (histórias, enredos, que deem suporte ao processo de negócio); **vi) mecânicas e dinâmicas** (ações e respostas às ações dentro do mundo do jogo); e **vii) tecnologia** (*engine* ou ambiente de criação do jogo).

1.3.4 Prototipação e Desenvolvimento

Esta etapa recebe como entrada o documento de design atualizado na etapa anterior. Aqui o objetivo é desenvolvimento, a codificação do jogo de acordo com os requisitos destacados do documento de design de jogos.

1.3.5 Avaliação com Equipe de Design

A avaliação com a equipe de design consiste em avaliar as versões dos jogos geradas pelo método de design do ponto de vista do balanceamento de elementos do jogo. Nesta etapa é necessário averiguar se todos os requisitos presentes do GDD foram devidamente implementados. Além disso, esta etapa é responsável por verificar o balanceamento entre os elementos do processo e elementos lúdicos no jogo.

1.3.6 Avaliação com Executores do Processo

Esta etapa é necessária pois os executores são as pessoas que têm o conhecimento sobre como o processo é executado. Portanto, eles devem ser capazes de reconhecer o seu contexto organizacional no mundo do jogo, por meio dos ambientes, regras, objetivos e aspectos organizacionais colocados no design.

1.3.7 Avaliação com Público Alvo

Esta é uma importante etapa no método de design pois é nela que se verifica se o jogo desenvolvido cumpre o propósito de fornecer a compreensão do processo pelos jogadores. Nesta etapa espera-se avaliar o jogo produzido pela ótica de qualidade do jogo, a qual investiga aspectos como clareza de objetivos, interação, desafios, imersão e etc.; além de verificar também se houve a compreensão dos valores, regras, etapas, dificuldades e desafios da execução do processo.

1.3.8 Empacotamento e Entrega

A última etapa do design do jogo compreende a publicação e entrega do jogo. Isso significa que o objetivo é juntar todos os recursos para execução do jogo em um único pacote (ajuda, jogo, sistemas de comunicação e etc.).

1.4. Construção de Jogo Digital Baseado em Serviço Público

Como já mencionado este minicurso tem como objetivo é a construção de um jogo digital para um processo de serviço público, portanto o ponto de partida será um GDD (*documento de game design*) previamente gerado na etapa de projeto de jogos (etapa 3 do método de design), ou seja, as etapas de estudo de contexto, mapeamento de elementos e projeto de jogos já foram executadas anteriormente. O foco desta seção é colocar em prática os requisitos descritos em documento de game design (disponível em: https://bit.ly/2QGClV1) para a construção do jogo digital que ele especifica (etapa 4 do método de design).

O processo de serviço público escolhido para a construção do jogo é o "Processo de Emissão do Cartão SUS", uma vez que é necessário que todo o cidadão brasileiro possa ter acesso à sua emissão, para a realização de procedimentos de saúde na rede do SUS. Além disso, este processo é pequeno o bastante para ser trabalhado no tempo do minicurso.

Como *engine* (ferramenta) de desenvolvimento de jogos será usada a plataforma de desenvolvimento de jogos *Sierra Construict 2*, a qual permite que todos os níveis de pessoas com pouco conhecimento em desenvolvimento de software consigam construir um jogo digital, uma vez que não apresenta sintaxes de linguagens de programação. As imagens e sons do jogo não serão construídos no minicurso, uma vez artes audiovisuais não está no foco deste curso, e, portanto, os mesmos estão disponíveis no link: https://bit.ly/2xk2wJ2.

1.4.1. Processo de Emissão do Cartão SUS

O SUS (Sistema Único de Saúde) foi instituído no Brasil por ocasião da promulgação da Constituição da República Federativa do Brasil em 1988 (Artigo 196) como forma de efetivar o mandamento constitucional do direito à saúde como um "direito de todos" e "dever do Estado", passando a oferecer a todo cidadão brasileiro acesso integral, universal e gratuito a serviços de saúde [Fiocruz, 2017]. O SUS traz benefícios a aproximadamente 180 milhões de brasileiros, realizando anualmente cerca de 2,8 milhões de atendimentos, desde procedimentos ambulatoriais simples a atendimentos de alta complexidade como transporte de órgãos, tendo como princípios básicos que o regem: universalidade do acesso à saúde, equidade dos serviços, integralidade, descentralização, regionalização, hierarquização e a participação social [Teixeira, 2011].

Devido à quantidade de brasileiros beneficiados pelo SUS, para que possam ter direito aos benefícios do sistema de saúde, todo cidadão tem direito ao Cartão Nacional de Saúde, ou como é conhecido, Cartão SUS [Portal Brasil, 2012]. Esta foi uma formalização instaurada para a identificação única do usuário e contribuir com a sua organização. O Cartão é um instrumento obrigatório do cidadão que possibilita a vinculação dos procedimentos executados no âmbito do SUS ao seu usuário, profissionais e a unidade de saúde onde foram realizados os procedimentos, sem este

documento, em várias instituições vinculadas ao sistema de saúde, os procedimentos não são realizados.

Existe um processo gratuito para solicitar a sua emissão (Figura 4). Atualmente o processo é descrito por etapas a serem seguidas. Devido à regionalização dos serviços de saúde, algumas etapas podem sofrer mudanças de um município para outro. O processo para realizar o seu cadastramento começa com a descoberta em sua localidade do local onde se deve comparecer para a confecção do cartão. Em seguida, é necessário se dirigir a esta unidade de atendimento de saúde com os documentos, carteira de identidade, CPF, comprovante de residência e, se possível, a carteira de vacinação. Estes documentos devem ser apresentados a um atendente, o qual incluirá seus dados no sistema, irá imprimi-lo, e poderá sair do local pronto para utilizá-la. Em algumas localidades, existe outro processo para a emissão do cartão, este realizado pelo Agente Comunitário de Saúde (ACS). Este servidor público é um agente intermediário entre o cidadão e o serviço de saúde, e devido a sua proximidade com ambas as partes, é possível solicitar ao mesmo que emita o cartão SUS. Outro meio de requisição do cartão é o cidadão realizar um pré-cadastro no Portal da Saúde do Cidadão, sendo necessário, o comparecimento à alguma unidade de saúde para a sua validação.

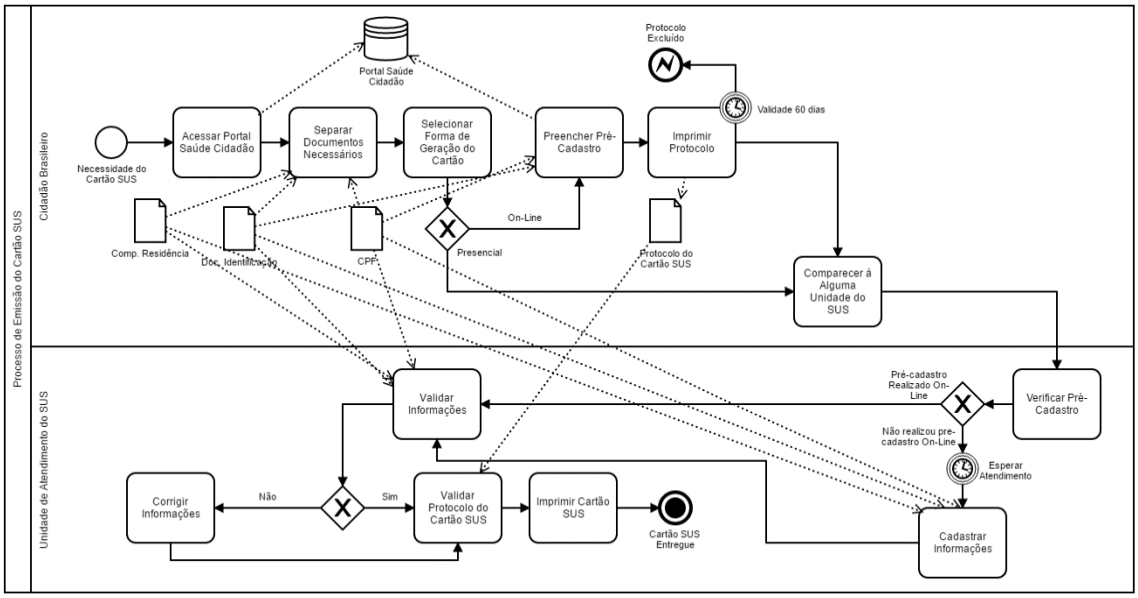


Figura 4 – Processo de Emissão do Cartão SUS

1.4.2. Plataforma Construct 2

Construct 2 é um programa que o deixa criar jogos de computador em HTML5 sem ter qualquer experiência em programação utilizando um intuitivo, ***'drag and drop'*** ambiente de desenvolvimento. A maioria das ferramentas dos programas pode ser usada a partir da interface gráfica sem ter que escrever uma única linha de código. A plataforma tem como foco a criação de jogos 2D, e vem com recursos que a torna fácil, incluindo um sistema físico que causa os itens no jogo serem governados pela lei da gravidade, como também bits gráficos e de som como os *sprites*, fundos e efeitos de som. Além disso, é tão simples como adicionar qualquer arquivo de mídia de uma aplicação exterior.

Basicamente a plataforma é organizada em 2 ambientes. O primeiro ambiente consiste no "desenho" do jogo, chamada ***Layout***, enquanto o segundo ambiente foca-se os eventos do jogo, ou seja, nos comportamentos de cada item inserido no layout. Esta segunda área recebe o nome de ***Event Sheet*** (Figura 5).

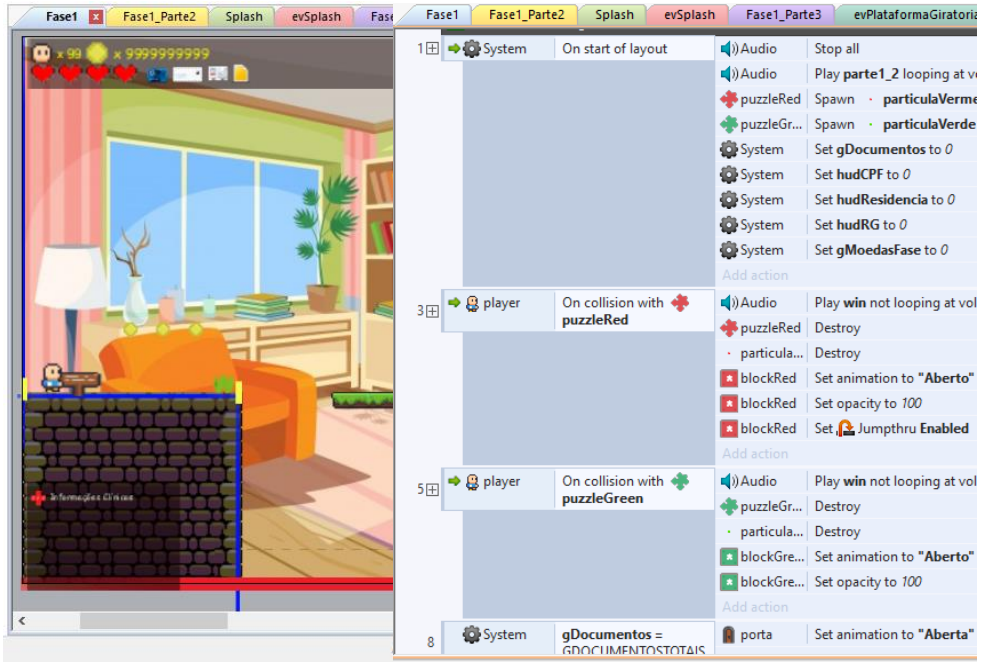


Figura 5 – Construct 2 (Layout e Event Sheet)

Outra característica importante da plataforma Construct 2 é que ela apresenta as áreas de: "***Projects***", responsável por exibir todos os projetos e sua organização de pasta; e "***Properties***", importante pois exibe as principais propriedades de algum item selecionada na área de *layout* ou na área de *projects* (Figura 6).

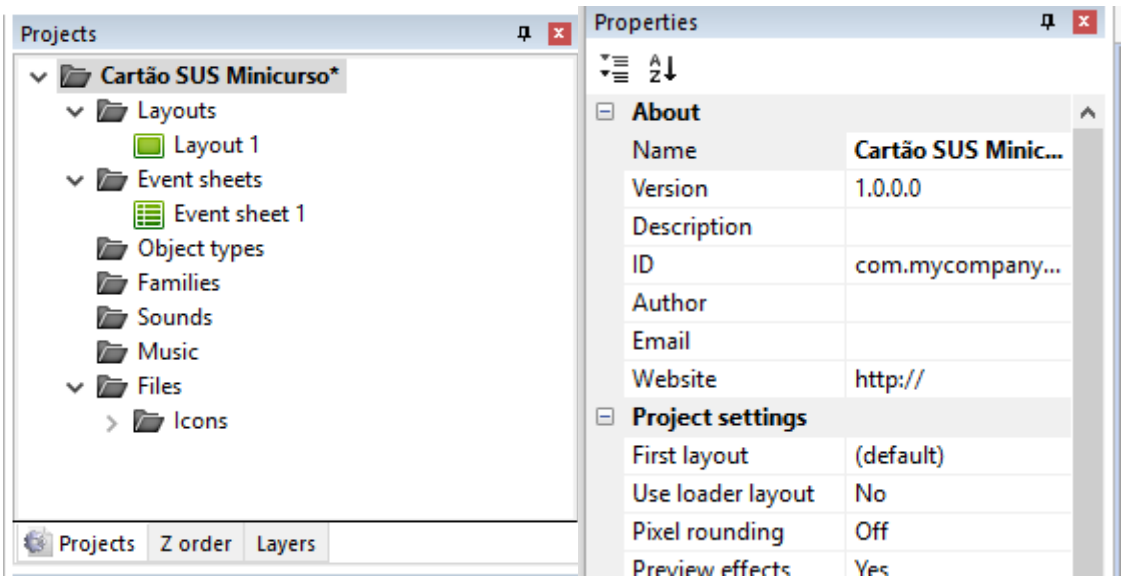


Figura 6 – Construct 2 (Project e Properties)

A filosofia de Construct 2 é uma da simplicidade e de um ambiente visual intuitivo. Quando arrasta os seus *sprites* (imagens) e os larga na sua posição do ecrã

(tela, fase etc.), outra parte do programa irá os fazer interagir com outros objetos de acordo com o tipo de objetivo. Ela é uma ferramenta interessante para qualquer um que quer começar a criar jogos digitais, mas que não tem qualquer habilidade em programação. As suas fáceis ferramentas e documentação extensiva tornam o numa escolha interessante.

1.4.3. Configurando o Projeto

Inicialmente deve-se criar um novo projeto no Construct 2, dê preferência para criar um "***New empty project***", o qual deverá criar uma área de *layout* em branco. Para atender ao mundo e à fase descrita no GDD (seção ***Scenarios (Places)***) é preciso realizar a configuração inicial do projeto de acordo com a Tabela 3 (A primeira coluna informa qual é o item selecionado; a segunda coluna indica o nome da propriedade; a terceira coluna indica qual é o valor esperado para à propriedade).

Tabela 3 – Configuração do Projeto

Item / Elemento / Sprite	Propriedade	Valor
New Project	Name	Cartão SUS Minicurso
	Windows Size	320, 300
	Loader style	Progress Bar & Logo
Layout 1	Name	Fase 1
	Layout Size	2400, 320
	Margins	160, 120
Event Sheet 1	Name	evFase1

1.4.4. Criando a Fase do Jogo

Segundo a seção ***Scenarios (Places)*** do GDD, a fase deste jogo será um ambiente em plataforma no qual o jogador deverá pular sobre obstáculos, coletar itens até conseguir chegar no posto de atendimento do SUS. Isso é descrito pelo processo de emissão do cartão SUS, portanto esta fase descreverá descrever os passos de coleta de documentação até sua entrega ao posto de atendimento para emissão do cartão SUS.

A Construct 2 permite trabalhar o layout como camadas, assim como os editores de imagens no mercado. Se selecionarmos o Layout "Fase 1", dentro da área de Projects existe uma aba chamada "***Layers***". Essas layers são as camadas de interação de um layout, ou seja, o que será exibido para o usuário em tempo de jogo. É comum ao construir jogos, fazê-lo em camadas de acordo com a sua funcionalidade.

Por exemplo, considerando o jogo que estamos construindo do cartão SUS, vamos trabalhar com a fase em 5 camadas: Fundo, Mapa, Principal, HUD, Mensagens (caso necessário). O fundo será apenas a imagem de fundo que a fase irá apresentar; o mapa é a imagem com as plataformas fixas do mundo; a principal será incluíra personagens, inimigos, itens e qualquer outro elemento de interação com o jogador; HUD são as informações de ***feedback*** dos jogadores como vidas, pontos e etc.; e. Mensagens, será telas de mensagem ou pausa do jogo. Portanto, crie estas layers para a Fase 1, lembrando de seguir a ordem tal como descrita aqui neste parágrafo, pois as camadas sobrepõem umas às outras (Figura 7).

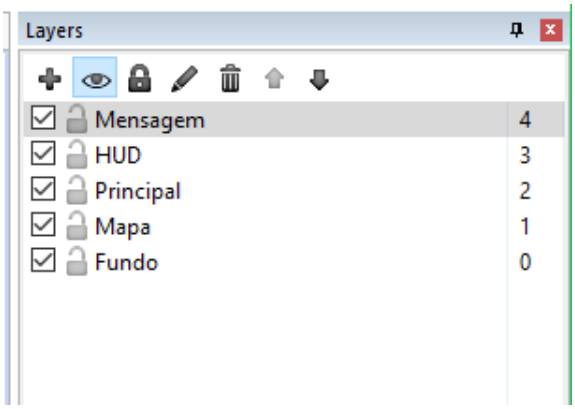


Figura 7 – Camadas da Fase 1

Já foi mencionado que o foco deste curso não é a criação artística de imagens, portanto, as imagens e sons do jogo já estão desenvolvidos de disponíveis para uso. Para inserir uma imagem como o **fundo da fase** ou o **mundo** do jogo e demais sprites (elementos gráficos) é necessário incluir um novo Sprite no level que deseja utilizar.

Para a criação do fundo da fase selecione a camada "Fundo". Na área de layout clique com o botão direito do mouse na parte de design e selecione a opção "***insert new object***". Irá aparecer uma tela de seleção com vários itens, nesta tela selecione "***Sprite***" e coloque em "***Name when inserted***" como "Fundo" (Figura 8). Atenção, quando um Sprite é inserido no projeto ele irá aparecer dentro de "***Object Types***" na área Projects. Quando você altera um objeto diretamente na área Projects você estará modificando todas as instâncias deles inclusas nas áreas de layout. Portanto, se quiser mudar propriedades de um objeto para um *layout* específico, é necessário selecionar este objeto dentro da área de layout, somente assim você irá alterar especificamente àquele objeto.

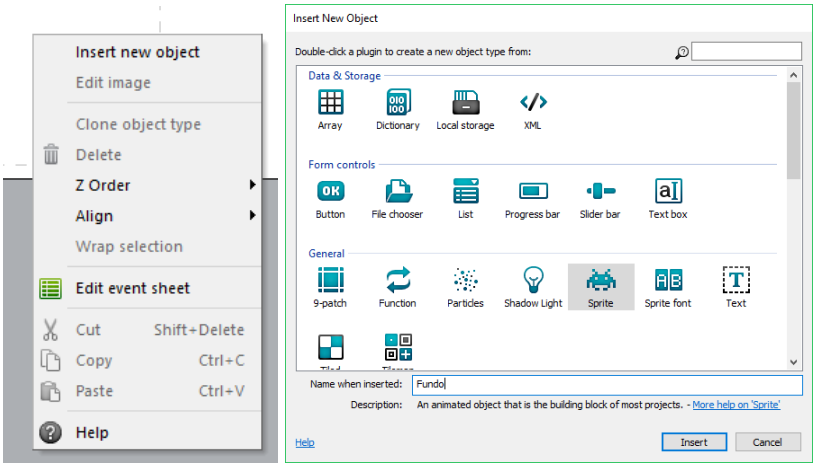


Figura 8 – Inclusão do Sprite "Fundo"

Após incluir esse Sprite a tela de edição de imagens será exibida (Figura 9). Esta tela apresenta um pequeno editor de imagens, mas ela também está preparada para trabalhar com animações (movimentos de objetos do jogo), ponto de colisão (pontos necessários de impacto entre objetos do jogo), pontos de origens (pontos sobre qual um objeto é construído ou criado) e outras funcionalidades.

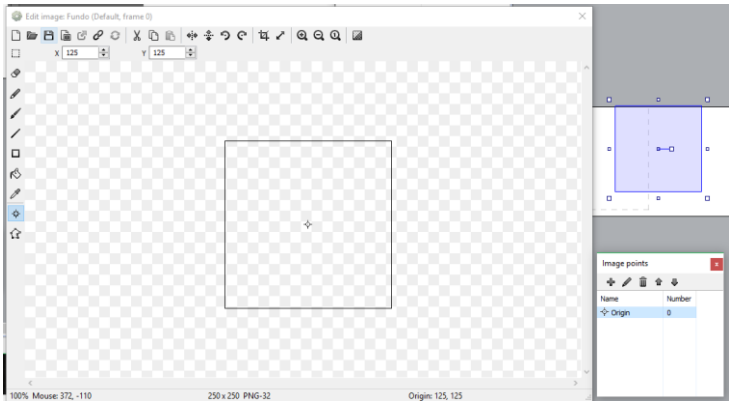


Figura 9 – Inclusão do Sprite "Fundo" – Edição de Sprite

O fundo se encontra dentro da pasta "***assets/fundos***" no link de recursos de utilização para o jogo. Para incluir o mundo abra nesta tela de edição de Sprite o arquivo "fundo1" para que a imagem seja carregada. Outra alteração importante neste Sprite é seu ponto de origem para que ele seja melhor posicionado na tela. Portanto na tela de "***imagem point***" clique com botão direito sobre *"**origin**", "**quick assign**", "**top-left**"*. Isso fará com que o ponto de origem desta imagem seja em cima e a esquerda. É possível fechar a tela de edição, e a imagem aparecerá no layout. Para que ela seja posicionada corretamente, selecione o Sprite "Fundo" e altere sua propriedade position para **0,0**. Para finalizar, ajuste somente o tamanho da imagem com a altura do layout. A largura não precisa, pois veremos mais à frente a opção de ***Parallax***.

Para incluir o mundo do jogo faremos o mesmo processo descrito acima para a inclusão do fundo, porém deve-se selecionar a camada de "Mapa" na Fase 1. Desta forma, insira um novo Sprite na Fase 1 com o nome de "**Mapa**". A imagem para o mapa está dentro de **"*assets/fundos***", selecione o arquivo "mapa". Selecione o ponto de origem como "***top-left***" e feche a tela de edição. Com o objeto de mapa selecionado, altere sua propriedade de "position" para **0,0**. Perceba que neste momento o mapa do jogo se sobrepôs ao fundo da fase. Neste momento, se rodarmos o jogo, a fase já tem uma aparência de jogo.

Uma propriedade muito importante ao cria um fundo de uma fase é o ***Parallax***. Esta propriedade permite dar velocidades diferentes de movimento para as camadas de um projeto.

Para finalizar esta seção, insira um objeto ***Keyboard*** no projeto. Este objeto é responsável pelo uso do teclado como "***joystick***" do jogo. Para inclui-lo é muito similar a um Sprite, porém, o que varia aqui é que deve ser selecionado o objeto "***keyboard***". Este objeto uma vez inserido, funciona para todas as telas e fases do jogo.

1.4.5. Criando o Chão do Jogo

Assim como o fundo e o mapa do jogo foram inseridos na Fase 1 o chão deve ser feito. Porém, para isso, serão usados blocos simples de Sprites, ou seja, sem imagens. Como assim sem imagens? Para criar chão de fases de jogos de plataformas não é necessário criar imagens, desde que já exista um mapa com o chão fixo desenhado. O que é o caso deste projeto.

Selecione a camada "Principal" da mesma forma que foram inclusos o fundo e o mapa, vamos criar um Spirte chamado "**chao**". Este Sprite não tem uma imagem, mas

para que ele não fique transparente, coloque uma cor sólida. Essa cor sólida é bem útil, até mesmo para o programador identificar onde estão os objetivos de chão da fase. No exemplo foi utilizada a cor azul para o chão. Neste momento, o projeto deve ser semelhante à Figura 10. Agora é necessário modelar chão da fase. Copie e cole o objeto chão no decorrer da fase, sempre os modelando (esticando, girar, diminuindo o objeto) de acordo com as plataformas do mapa (Figura 11).

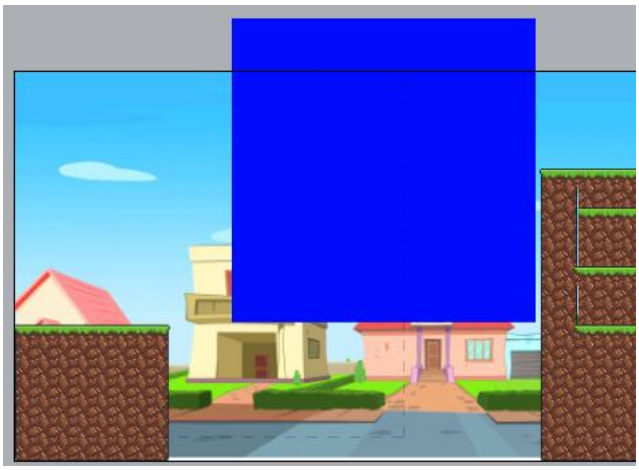


Figura 10 – Inclusão do Sprite de Chão

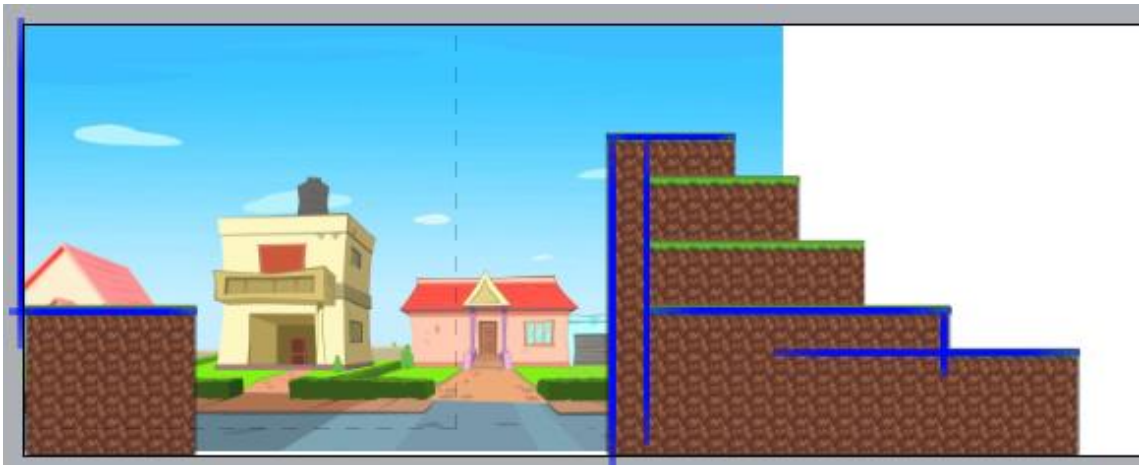


Figura 11 – Exemplo de Modelagem do Chão da Fase

Após a modelagem dos sprites de chão é preciso fazer com que eles tenham comportamento de chão. Para isso, selecione o objeto chão dentro de "***objetct types***". Em suas propriedades, identifique "**Behaviors**". Isso é uma facilidade da ferramenta Construct que assume que todo objeto tem comportamentos no jogo, e estes comportamento são adicionados aos objetos para que eles executem ações ou tenham determinadas características no jogo. Como este jogo é uma plataforma, e o chão deve possuir o comportamento de ser sólido, clique em behaviors. Será exibida uma tela para seleção de comportamento. Portanto, adicione um behavior chamado "***Solid***" (Figura 12).

Porém repare que não é bonito para o jogo que esses retângulos azuis fiquem sendo exibidor em tempo de jogo. Para escondê-los basta selecionar trocar sua propriedade "***Initial Visibility***" para "***Invisible***". O chão da fase está pronto.

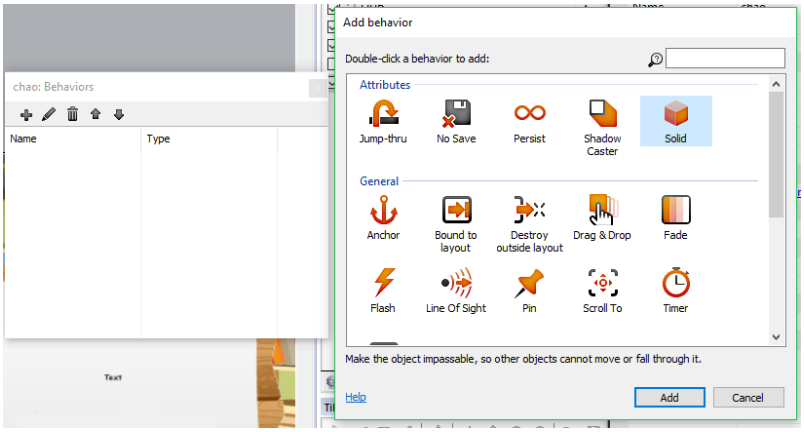


Figura 12 – Behavior Solid para o Chão.

1.4.6. Criando o Personagem Jogador

Um dos itens mais importantes de um jogo é o jogador. E neste caso, temos um personagem que deve andar pela fase, saltar e coletar itens para conseguir o seu cartão SUS. O personagem é uma seção específica do GDD, e neste exemplo suas definições podem ser observadas na seção ***Player(s)*** do documento de design. Como usual, da mesma forma que inserir sprites nas seções anteriores, vamos neste momento, inserir um novo Sprite na camada Principal, chamado "**jogador"**.

Neste primeiro momento apenas dê uma cor para ele. No exemplo optamos pelo jogador na cor vermelha (assim como coloquei a cor azul para o fundo). Coloque também a propriedade "***Size***" do jogador com o valor de **20,20**. E posicione o quadrado vermelho que representa o jogador no início da fase (Figura 13).

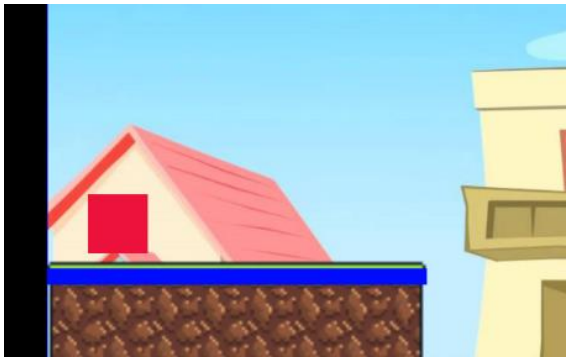


Figura 13 – Inclusão do Jogador na Fase.

Se rodarmos o jogo é possível perceber que são apenas imagens imóveis. Para que ele se mova como um personagem de jogo de plataforma, o que é necessário fazer é adicionar um "***behavior***" chamado "***Platform***" para o objeto jogador. Esse comportamento implementa automaticamente os comandos de um jogo de plataforma no objeto que o possui. Se rodar o jogo agora, vai perceber que o quadrado do jogador responde aos comandos de seta do teclado do computador. Como o comportamento de plataforma foi implementado automaticamente, algumas propriedades precisam ser modificadas para que o gameplay seja melhor aproveitado, como por exemplo: alterar força da gravidade, distância de salto e outros (Tabela 4).

Tabela 4 – Configurações do Comportamento do Jogador

Item / Elemento / Sprite	Propriedade	Valor
Jogador	Max Speed	100
	Jump Strength	400

Outro behavior que deve ser inserido para o jogador é o chamado "***ScrollTo***", pois do jeito que está no momento, a câmera do jogo não acompanhará o jogador pela fase. Após inserir este comportamento, a câmera do jogo irá acompanhar o jogador por ponde ele for na fase.

1.4.6.1. Definindo a Imagem do Personagem

Para os sprites anteriores foram inseridas imagens fixas ou alguma cor para sua identificação, mas em se tratando de personagens, é necessário movimento. A Construct 2 permite trabalhar com movimento, ou no termo correto, "Animações" de várias imagens separadas, ou de um arquivo único com várias imagens, os conhecidos "***tilesets***".

Portanto, para incluir imagem para o jogador é necessário abrir a tela de edição do Sprite do jogador. Dê um duplo clique no objeto do jogador para isso. Perceba que toda vez que você abre essa tela de edição do Sprite, surge duas telas menores: "***Animations***" e "***Animation Frames***". Estas telas são responsáveis pela animação de sprites. A primeira, é responsável por um delimitar animações, por exemplo, andar, saltar, parar e etc.; enquanto a segunda representa cada uma das imagens destas animações. No nosso exemplo criaremos as Animations: Andar, Parado, Pulando, Abaixar e Morte (Figura 14). Em cada uma delas iremos colocar a animação necessária.

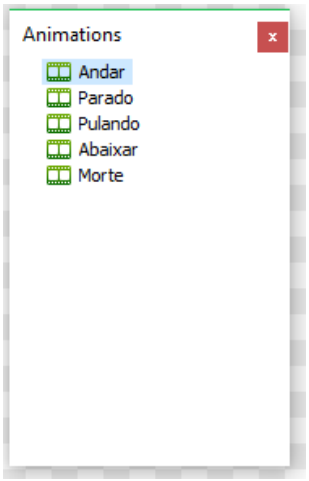


Figura 14 – Animações do Jogador.

Para criar a animação de andar, selecione a animação "Andar" e na tela de "***Animation Frame***", clique com botão direito, então em "***Import Frames***", "***From Sprite Strip...***". As imagens do jogador estão dentro de "***assets/jogador***", carregue o arquivo "***jogador***". A ferramenta perguntará quantas linhas e quantas colunas a imagem é composta, o que significa quantas imagens de movimento contém dentro deste único arquivo (Figura 15). Nesta imagem devemos colocar 6 verticais e 5 horizontais.

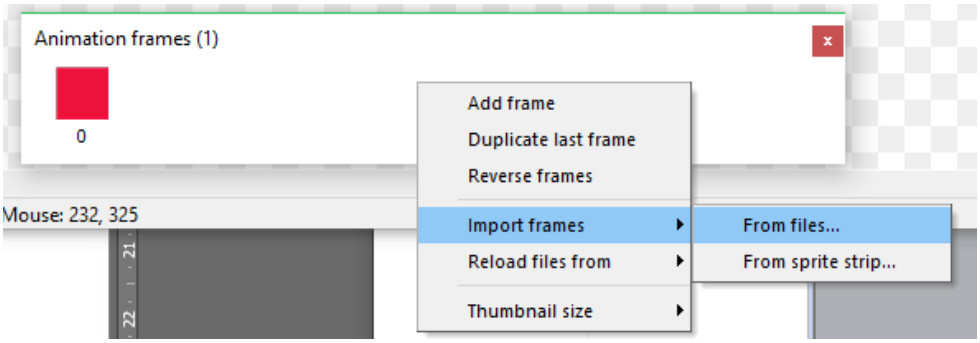


Figura 15 – Animações do Jogador Carregando TileSets.

Ao carregar o arquivo todas imagens do personagem irão aparecer. Como estamos interessados apenas no "Andar" do jogador. Devemos excluir o quadrado vermelho, e deixar apenas as imagens de 6 a 11, deletando as demais. Cada animação criada também possui propriedades. Portanto, altere a propriedade "***Loop***" de "Andar" para "***Yes***". Agora é só repetir estes passos para as demais animações. Com isso, ao executar o jogo, o jogador já aparece com um personagem.

Para finalizar em cada uma das animações, coloque a "***Origem Point***" de origem do jogador como "***Botton***".

1.4.6.2. Movimentando as Animações Corretamente

O jogador está se movimentando, já é possível ver animações do personagem, porém se formos detalhistas perceberemos que apesar das teclas de movimento de jogos de plataforma estarem funcionando, as animações do personagem não respeitaram isso. Por exemplo, ele continua virado para frente quando apertamos esquerda, ou mesmo quando está parado, parece que está em uma esteira.

Para que estas animações respeitem os movimentos do jogador é necessário programar estes eventos. Criaremos então uma nova ***Event Sheet*** com o nome de "***evJogador***". Para que este evento *sheet* seja usado na fase é necessário referencia-lo dentro da Event Sheet "***evFase1***". Clique com botão direito e depois em "***Include Event Sheet***", selecionando a "evJogador" que criamos.

Para programar os eventos do jogado, vamos a "***evJogador***". O primeiro evento que vamos programar os movimentos (Figura 16):

- "Pular": se o jogador estiver pulando OU se o jogador estiver caindo ele deverá estar com a animação de pulo;
- **SENÃO** "Parar": se o jogador não estiver pressionado direita OU se o jogador não estiver pressionado esquerda E se o jogador está sobre alguma plataforma (Sólida) ele ficará parado.
- **SENÃO** "Andando":
  - Se o jogador estiver pressionado a seta para direita ele está andando para direita.
  - **SENÃO**: Se o jogador estiver pressionado a seta para esquerda ele está andando para esquerda.

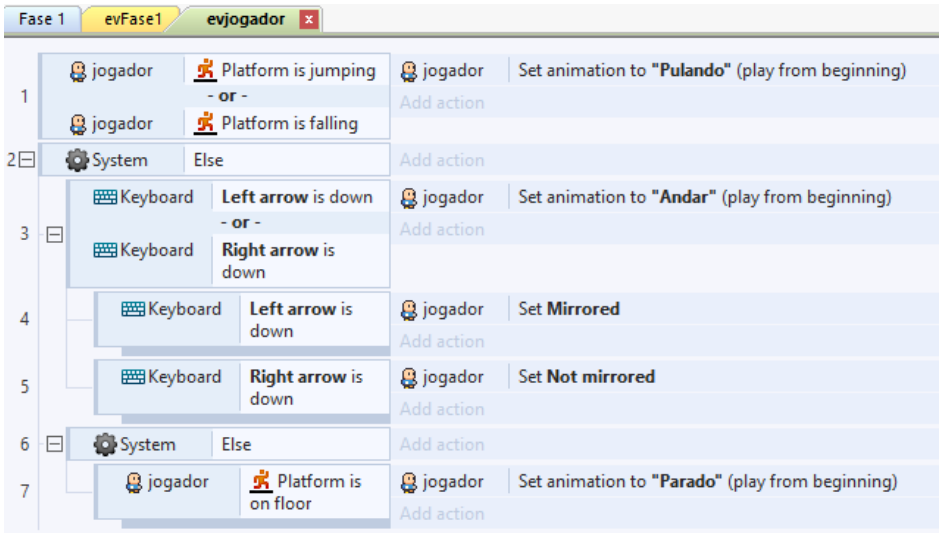


Figura 16 – Programação dos Movimentos e Animações do Jogador.

1.4.6.3. "Morte" do Jogador

Todo jogo deve ter critérios de sucesso e insucesso. De acordo com o GDD, na seção de ***Success – Game Overs e Fails – Game Overs*** são descritos os principais meios de ganhar ou perder o jogo. De acordo com o processo, a emissão do cartão SUS é encerrada quando o cidadão consegue emitir o seu cartão ou quando ele não consegue emitir o cartão. Desta maneira o jogo seguirá o critério de vitória de conseguir emitir o cartão SUS. Porém como derrota, os game designers pensaram nos jogos clássicos de plataforma que existem chances para realizar objetivos. Desta maneira, o jogador deverá ter critérios de derrota em alguns pontos.

O primeiro deles se baseia na quantidade de chances para reiniciar o objetivo do jogo, as famosas "vidas". Para criar vidas, partiremos de uma variável inicial chamada "***gVidas***". Para cria-la vá na "***evFase1***" clique com o botão direito do mouse e depois em "***Add global variáble***". Em suas configurações deixe-a como "***Numéric***" e adicione 5 como "***Initial Value***".

O outro critério se baseia em perder muita energia. A energia é configurar como uma variável do próprio jogador, no qual, ao atingir o valor de Zero, o mesmo perderá uma chance. Para criar uma variável de objeto, selecione o objeto de jogador e clique na sua propriedade "***Instance Variable***". Inclua a variável "***Energia***", marcando o tipo como "***Numéric***" e valor inicial igual a 3. Ou seja, o jogador terá 3 chances de sofrer danos antes de perder uma chance de jogo (Este critério será explicado na seção 1.4.10, ao criar inimigos).

1.4.6.3.1. Área de Morte

Para perder uma chance direta, foi definido que o jogador ao cair de uma plataforma para um vão "sem fundo" o mesmo irá perder uma chance. Para realizar isso, devemos colocar um Sprite conhecido como "área de morte" por toda a extensão da fase na borda inferior. Quando o jogador colidir com essa área ele perderá uma chance. Desta forma crie o Spirte "***AreaMorte***", ele não possui imagens, mas no exemplo optamos pela cor vermelha sólida para identifica-la com maior facilidade. Posicione-a na parte inferior da fase, esticando-a por toda extensão (Figura 17).

Figura 17 – Área de Morte (Estendida por toda a borda inferior da fase).

Posicionado este elemento, o que resta é apenas programar o evento de morte. Portanto no *event sheet* "***evFase1***", iremos definir a seguinte lógica: quando o jogador colidir com a área de morte, então perderá uma chance, destruirá o jogador, e reiniciará a fase. A lógica básica é essa, entretanto, para chamar a atenção a este evento, a animação também será alterada para "morte" e o jogador perderá o controle do personagem, até que a fase seja reiniciada. Além disso, é preciso verificar se não foi zerado o número de chances, que em caso positivo o jogo é encerrado por falta de chances. A Figura 18 mostra todos os eventos para esta etapa do jogo.

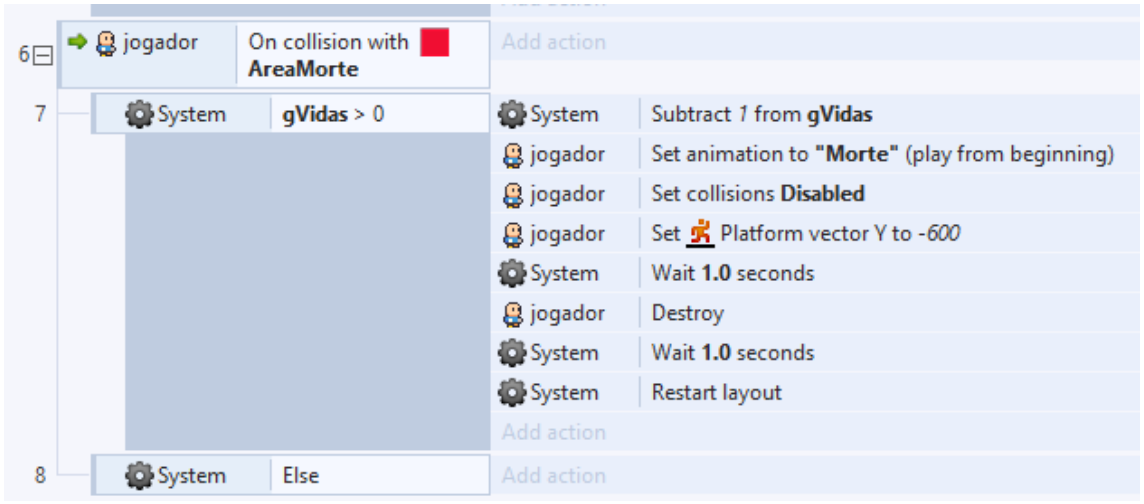


Figura 18 – Eventos da Área de Morte.

1.4.7. Criação de Plataformas Móveis

De acordo as informações da seção ***Scenarios (Places)*** no documento de design de jogo existem plataformas móveis a serem inclusas por toda a fase para possibilitar o jogador ultrapassar grandes distâncias, tanto em altura quanto em comprimento. A inclusão de plataformas móveis pode ser feita de uma maneira bem simples, e para isso, devemos criar um Sprite novo que terá o ***behavior*** "***Solid***", para que o personagem consiga subir neste objeto, mas também deverá ter o comportamento de "***Sine***". O behavior Sine permite movimentar elementos nas direções vertical ou horizontal, obedecendo a magnitude (distância) do movimento e também a período, que consiste no tempo que o objeto com o comportamento demora para percorrer toda a sua magnitude de movimento.

Desta forma, para criar as plataformas no jogo crie um novo Sprite com o nome de "***Plataforma***". Não é necessário ainda colocar uma imagem, no exemplo optamos por uma cor sólida amarela para diferenciar o Spirte. Inclua também os dois behaviors

mencionados "***Solid***" e "***Sine***". Para ilustrar no início da fase, foram colocadas duas plataformas (conforme diagrama da seção ***Scenários (Places)*** do GDD) (Figura 19).

Figura 19 – Configuração das Plataformas Móveis.

Para a plataforma móvel A, a propriedade "***Movement***" do behavior ***Sine*** deverá ser "***Horizontal***" fazendo com ela se movimente na horizontal. Enquanto a plataforma B, a propriedade "***Movement***" será "***Vertical***", fazendo a plataforma subir e descer. Neste momento ao executar o jogo, verá que a plataforma já segue os padrões de movimento especificados.

Para concluir esta seção, basta incluir as plataformas restantes no decorrer da fase, de acordo com a ilustração da Seção ***Scenários (Places)*** do documento de design do jogo.

Inserido todos as plataformas da fase, devemos então configurar sua aparência. Dê um duplo clique no objeto plataforma e abara a imagem "plataforma" dentro de "***assets/itens***" nos arquivos de construção do jogo. Ao realizar isso, suas plataformas terão a mesma aparência do cenário.

1.4.7.1. Movimentando o Cenário Junto com o Jogador

Neste ponto do jogo, já é possível perceber que a medida que o jogador caminha na fase, o cenário de fundo o acompanha, porém chega em determinado ponto que o jogador ultrapassa o cenário e o fundo fica totalmente branco. O que fazer? Aumentar a imagem de fundo? Isso pode ocasionar distorções no jogo.

Para solucionar isso, existe uma propriedade chamada ***Parallax***. Esta propriedade permite definir velocidades diferentes para as diversas camadas de um projeto de jogo no Construct 2. Por exemplo, se quisermos uma camada em que seus objetos sempre fiquem no mesmo local da câmera do jogo, devemos configurar um ***Parallax*** estático, ou seja, **0,0**. No nosso exemplo queremos que o fundo se movimente em uma velocidade menor que o movimento da camada do jogador. Como o jogador está situado na camada Principal, e a imagem de fundo do jogo na camada Fundo, devemos configurar um Parallax menor para ela. Com a camada "Fundo" selecionada, configure sua propriedade ***Parallax*** para **10,10**. Ao executar o jogo, é possível perceber que o fundo do jogo segue uma velocidade um pouco menor do que o jogador na fase, permitindo que todo o conteúdo da imagem seja preenchido no decorrer da fase.

1.4.8. Criação de Parede e Chão Atravessáveis

Em vários jogos do gênero aventura em plataforma, é comum se deparar com algumas plataformas que são passíveis de serem transportadas de baixo para cima apenas saltando. Estas são as plataformas atravessáveis e no Construct 2, para criarmos tal mecanismo basta configurar em um objeto o behavior "***Jump-Thru***". Esse comportamento permite que um chão seja atravessável por um pulo do personagem. Ele tem um comportamento de sólido, mas é atravessável.

Para criar este tipo de objeto, insira um novo Sprite no jogo e dê a ele o nome de "***PlataformaAtravessavel***", configurando também seu behavior com o comportamento de "***Jump-thru***". A Figura 20 mostra o ponto da fase onde este objeto será configurado (de acordo com o GDD). Repare que nestes pontos o jogador irá atravessar o "chão" saltando. Foi selecionado estes pontos, pois segundo o GDD do jogo, aqui acontecerá um puzzle, onde veremos nas próximas seções, que a área mais inferior irá conter um documento do cartão SUS e um inimigo que só poderá ser derrotado usando raciocínio e habilidades.

Configurada as plataformas atravessáveis, é possível executar o jogo e testá-las. É possível perceber que estes pontos ainda estão aparentes. Para que eles fiquem escondidos configure sua propriedade "***Initial Visibility***" como "***Invisible***".

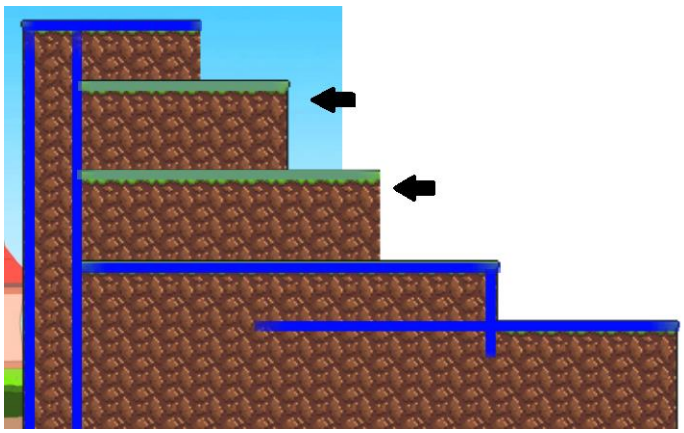


Figura 20 – Configurando Locais de Plataformas Atravessáveis.

1.4.9. Criação de Itens Coletáveis

Os itens de coleta do jogo estão descritos na seção ***Itens and Objects*** do GDD. No qual são detalhados as suas funcionalidades e localizações no decorrer do jogo. Assim como qualquer outro objeto do jogo, estes também são sprites que devem ser adicionados à fase. A Tabela 5 exibe a localização dos itens, seu nome, imagem e funcionalidade.

Tabela 5 – Itens da Fase

Item	Localização	Assets	Funcionalidade

 RG	Início da fase, dentro em baixo da primeira plataforma atravessável	“assets/itens/rg.png”	Item de desbloqueio de acesso ao posto de atendimento do SUS.
 CPF	Metade da fase, na plataforma superior.	“assets/itens/cpf.png”	Item de desbloqueio de acesso ao posto de atendimento do SUS.
 Comp. Residência	Final da fase na plataforma inferior.	“assets/itens/residencia.png”	Item de desbloqueio de acesso ao posto de atendimento do SUS.
 Cartão SUS	Meio da fase na plataforma superior.	“assets/itens/cartaosus.png”	Item de final de jogo. Só é obtido após acesso no posto de atendimento do SUS com toda a documentação.
 Computador	Início da Fase	“assets/itens/computador.png”	Informa ao jogador qual seu objetivo no jogo.

Para o objeto do Cartão SUS as propriedades de colisão e visibilidade serão desabilitadas (***Colisions* = *Disabled; Initial Visibility* = *Invisible***), pois o mesmo só poderá ser coletado pelo jogador após o mesmo visitar o posto de atendimento do SUS, de acordo com a tarefa do modelo de processo “Comparecer a Alguma Unidade do SUS” que tem como regra a coleta dos documentos para a validação e cadastro no SUS.

Outro item bastante comum em jogos de plataforma são as moedas, as quais contam como pontuação no jogo. Para incluir moedas crie um novo Sprite chamado “moeda”. Sua imagem é uma animação, que dará a impressão que ela está girando. Portanto, assim como foram feitas as animações do jogador, vamos criar a animação para a moeda. Com a animação default selecionado vá no “***Animation Frames***”, clique com botão direito do mouse, “***import frames/from strip...***”, marque 4 horizontais e 1 vertical. E para finalizar altere a propriedade Loop da animação para “***Yes***”. Agora cabe a você espalhar as moedas pela fase, onde desejar.

1.4.9.1. Contagem de Itens e Moedas

O sentido de ter itens espalhados pela fase, é permitir que sua coleta possa refletir em benefícios ao jogador. No caso dos itens de documentos, eles são necessários para permitir o acesso do jogador ao posto de atendimento do SUS para a emissão do seu

cartão. Para isso, o jogo deve permitir armazenar (ou minimamente) contar a quantidade destes itens coletados.

Realizar essa contagem, deve-se criar variáveis na ***Event Sheet*** que consigam contabilizar a coleta. Portanto, vá a ***Event Sheet "evFase1"***. Para criar variáveis clique com o botão direito do mouse e depois em "Add global variável". Inclua as variáveis chamadas "***gDocumentos***" e "***gMoedas***" todas do tipo "***Number***".

Para contabilizar as moedas e documentos devemos trabalhar com colisões de objetos, ou seja, quando o jogador encostar em um item, algo deverá ser executado. No contexto de documentos e moedas, a lógica executada será: jogador colidiu com algum destes itens, então, destrua o item, e contabilize mais um para o jogador de acordo como mostrado na Figura 21. A única diferença entre os itens será a do cartão SUS, que ao colidir com ele, o jogador finalizará o jogo.

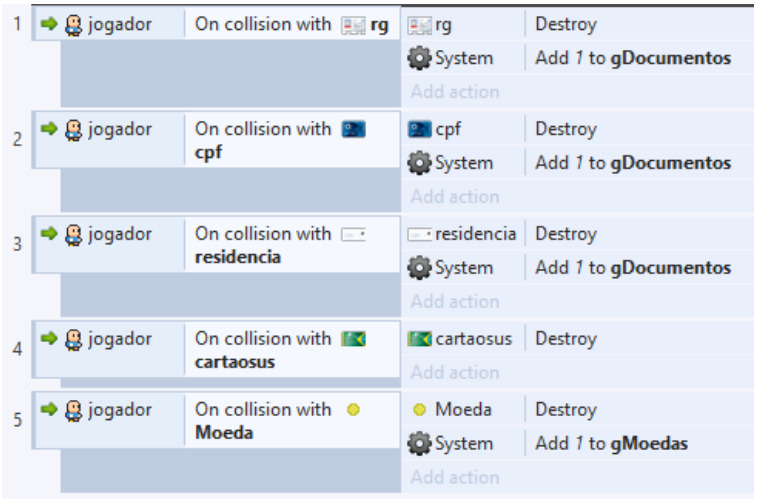


Figura 21– Eventos de Coleta de Itens para o Personagem.

Ao perder uma chance é reiniciada a fase do jogo, entretanto, do jeito que está no momento, as variáveis de moedas e de documentos não estão sendo reiniciadas. Por isso é necessário programar o evento de início do Layout para zerar tais variáveis (Figura 21).

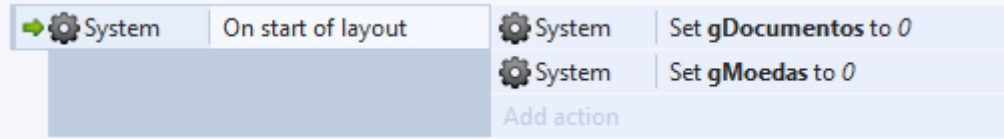


Figura 22– Eventos de Reinicio da Fase.

1.4.10. Criação dos Inimigos

Para este jogo, os designers definiram alguns inimigos que representam doenças, as quais, sem o jogador possuir o cartão SUS, o mesmo não poderá ter acesso a rede pública de saúde para trata-las. Para isso, serão criados como inimigos, as cobras (necessidade de soro antiofídico), os mosquitos (vacinas contra a dengue) e as abelhas (choque anafilático).

Para isso vamos incluir novos Sprites, um chamado "***cobra***", o "***mosquito***" e a "***abelha***". Estes sprites estão localizados dentro de "***assets/inimigos***". Assim como a

animações de personagem, que foi construída em 1.4.6.1, estes inimigos possuem imagens de movimento que devem ser inseridas na animação "***Default***".

O jogador precisa evitar a todo o custo encostar nestes inimigos, pois há modos de derrota-los, o mesmo sofrerá danos até perder as chances de concluir o objetivo. Os inimigos devem ser colocados na fase de acordo com o modelo da seção "***Scenarios (Places)***" do GDD

1.4.10.1. Cobras (Inteligência Artificial)

As cobras apenas ficam rastejando lentamente de um lado para o outro em determinado ponto da fase. Para programar tal comportamento para as cobras, são colocados Sprites limitadores, ou seja, serão colocados sprites que servirão de limitadores de movimentos paras as cobras. Quando uma cobra bater nestes limitadores, ela então dará meia-volta e rastejará para o outro lado. Estes limitadores receberão o nome de "***ColisorInimigo***", deverá ser invisível para os jogadores, funcionando apenas para colisão com as cobras. Para que a cobra se movimente, devemos configurar um behavior "***Platform***" nela, entretanto a propriedade "***Default Controls***" deverá esta como "***No***", "***Max Speed = 5***" e "***Acceleration = 50***". Além disso, para saber a direção do movimento da cobra, incluiremos uma variável de objeto nela chamada "***paraDireita***" (Figura 23).

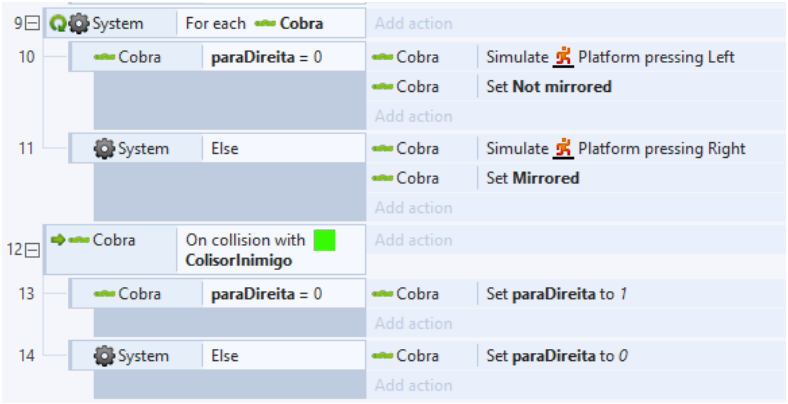


Figura 23– Eventos de Inteligência Artificial para Cobras.

1.4.10.2. Mosquitos (Inteligência Artificial)

Os mosquitos somente ficam voando em movimento aleatórios e em bando. Para simular este comportamento, devemos colocar nos mosquitos dois behaviors do tipo "***Sine***", bastando colocar diferentes propriedades de Magnitude e Amplitude de seus movimentos. Não é preciso programar nenhum evento uma vez que o comportamento da *Sine* já consegue fazer a simulação do voo do mosquito.

1.4.10.3. Abelhas (Inteligência Artificial)

As abelhas são os inimigos mais complicados, pois ao avistar o jogador ela irá prossegui-lo onde quer que ele esteja. Para este comportamento acontecer é necessário incluir na abelha o behavior "***Pathfinding***" o qual cria um caminho de um objeto para outro. Como estamos trabalhando com um jogo em que a maioria das imagens são em 16bits, é necessário ajustar a propriedade "***Cell size***" do behavior ***pathfinding*** para 16, no intuito de dar maior fluência ao movimento da abelha e "***Rotate Object = No***", para

não permitir que o objeto fique girando na tela. A programação da inteligência artificial da abelha pode ser observada na Figura 24.

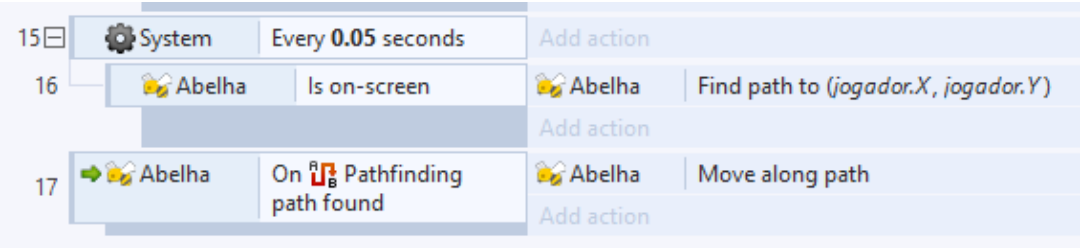


Figura 24– Eventos de Inteligência Artificial para Abelhas.

1.4.10.4. Colisão de Inimigos com o Jogador

Como mencionado anteriormente, quando o jogador encostar nos inimigos o mesmo sofrerá dano, e para isso foi criado uma variável de objeto para armazenar a energia do jogador. É comum em jogos de plataforma, quando o jogador sofre dano, o mesmo ficar alguns segundos invencível. Para cumprir este comportamento de invencibilidade momentânea, selecione o objeto do jogador e inclua nele uma nova variável de objeto chamada "invencível" do tipo numérico. Além disso, para mostrar a invencibilidade, ele ficará piscando. O comportamento de piscar precisa da inclusão do behavior "***Flash***".

Portanto, colidir com um inimigo o jogador irá sofrer um dano, mas ficará alguns segundos (3 segundos) invencível, sem poder sofrer danos novamente. Toda a lógica destes eventos pode ser observada na Figura 25.

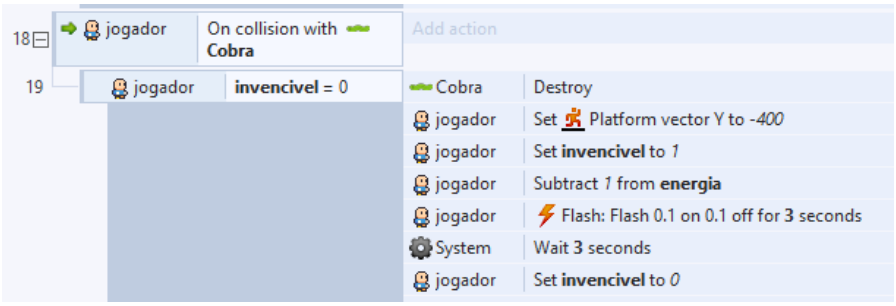


Figura 25– Eventos de Colisão com Inimigos (Todos inimigos devem ter os mesmos comandos).

1.4.11. Morte por Falta de Energia

Com os ataques dos inimigos o jogador deverá perder uma chance ao final de sua energia. Para isso basta verificar a energia do mesmo a cada ciclo de processamento do jogo. A Figura 26 exibe os comandos para realizar a funcionalidade de perda de chance por energia.

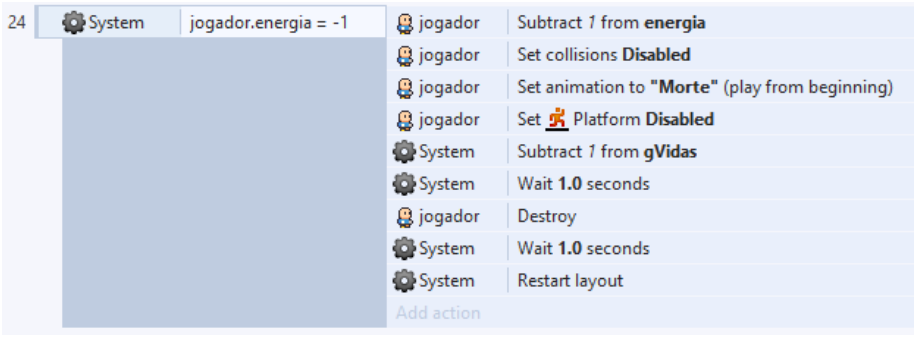


Figura 26– Eventos de Perda de Chance por Término de Energia.

1.4.12. Morte por Falta de Energia

Os HUD são itens de interface com o jogador que fornecem feedback para ele compreender o que está acontecendo com o jogo. No início do projeto foi criado uma camada somente para inclusão destes elementos. Portanto, para que os mesmos fiquem sempre expostos para o jogador na tela é necessário tomar cuidado com o ***Parallax*** desta camada, pois a mesma não pode se movimentar. Para fazer isso, coloque sua propriedade Parallax em 0,0, garantindo que o conteúdo fique estático. Além disso, é necessário tomar cuidado com a posição dos elementos na tela, pois se eles não fazerem parte da câmera do jogo eles irão desaparecer na medida que o jogador se movimenta.

Iremos então selecionar a camada HUD, e incluir os objetos de acordo com a Tabela 6.

Tabela 6 – Componentes de HUD

Objeto	Nome	Propriedade	Valor
Text	txtVidas	Text	Vidas: 5
Text	txtDocumento	Text	Documentos: 0/3
Text	txtMoedas	Text	Moedas: 0
		Horizontal Align	Right
Progress Bar	pgbEnergia	Maximum	3

Após incluir os componentes é necessário programar seu comportamento. Seus valores devem ser ajustados de acordo com cada ciclo do jogo, atrelados às variáveis de controle (Figura 27).

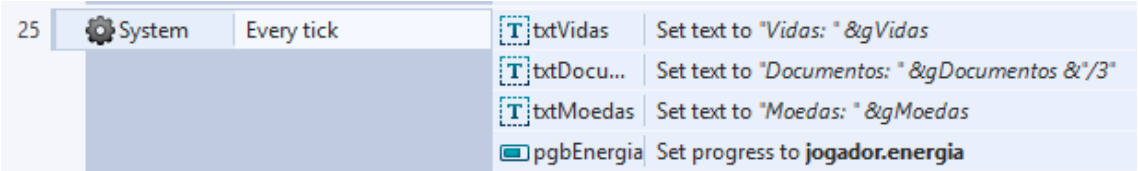


Figura 27 – Eventos de Configuração dos Objetos de HUD

1.4.13. Mensagens de Final de Jogo

Este jogo por se basear no processo de emissão do Cartão SUS brasileiro, algumas informações sobre o processo, curiosidades e contexto sobre o qual ele é prestado

precisam ser transmitidas para o jogador. Estas informações permitem o jogador a compreender como o processo é executado, suas dificuldades, motivos de sua criação, prestador e etc.

Neste jogo existem alguns itens durante a fase, os computadores, que dão estas informações aos jogadores, quando estes os acionam. Para exibir as mensagens neste jogo, no início do projeto foi criada a camada "Mensagem" exatamente para transmitir informações ao jogador. Portanto, para exibi-las, selecione tal camada e altere a suas propriedades ***Parallax*** para ***0,0***, pois queremos a mensagem estática para o jogador; e "***Initial Visibility = Invisible***" para que ela não esteja visível ao jogador em um primeiro momento.

Ainda nesta camada, incluiremos um novo Spirte com o nome de "***fundoMensagem***", no qual colocaremos nele uma cor branca sólida e alteraremos a propriedade "***Opacity = 70***". Expanda-o na tela de forma que fique parecendo uma caixa de mensagem de um software. Dentro do "***fundoMensagem***", coloque um objeto de Texto com o nome de "***txtMensagem***", e também o expanda de maneira que cubra o fundo branco da caixa. Neste momento você também pode alterar tamanho de fonte, para melhor comportar os conteúdos das mensagens.

Com isso pronto agora basta configurar os computadores para exibir as mensagens. Para isso, selecione um objeto computador e inclua uma variável de objeto com o nome de "mensagem" do tipo texto. Ao todo, de acordo com o GDD teremos 3 computadores espalhados pela fase no qual suas variáveis de mensagem devem conter os seguintes conteúdos:

1. "Prezado cidadão, para realizar qualquer atendimento na rede do SUS, você precisa do Cartão SUS. Sem ele nenhum hospital ou posto da rede pública poderá atendê-lo! A emissão do Cartão SUS é Gratuita. Você só precisa comparecer a qualquer posto do SUS com seu CPF, Documento de Identificação e Comprovante de Residência. Portanto tenha cuidado para não sofrer nenhum ataque de animais, quedas ou qualquer problema de saúde enquanto não emitir seu cartão. PORÉM CUIDADO: Existem animais pela fase que podem te machucar, como você não tem o Cartão SUS ainda, não poderá ser socorrido pela rede pública de saúde. NÃO ENCOSTE NELES.";
2. "Durante todo a fase estão espalhados itens e documentos que são úteis para a conclusão dos objetivos. Para isso você deve visitar as áreas da fase, ir ao final dela e então retornar ao início. Somente assim conseguirá concluir seus objetivos."
3. "Prezado cidadão, para emitir o cartão SUS você poderá realizar um pré-cadastro dos documentos no Portal "Saúde do Cidadão". Ao comparecer no posto de atendimento do SUS com o protocolo do pré cadastro, a emissão do seu Cartão SUS é mais rápida, pois é preciso somente validar os dados dos documentos. Não fazendo o pré-cadastro a emissão é mais demorada, pois é necessário, também, cadastrar todas as suas informações no sistema do SUS."

Para exibir tais mensagens, basta colidir com o computador. E para fechar as mensagens pressione a tecla "***Escape***" (ESC) no teclado (Figura 28).

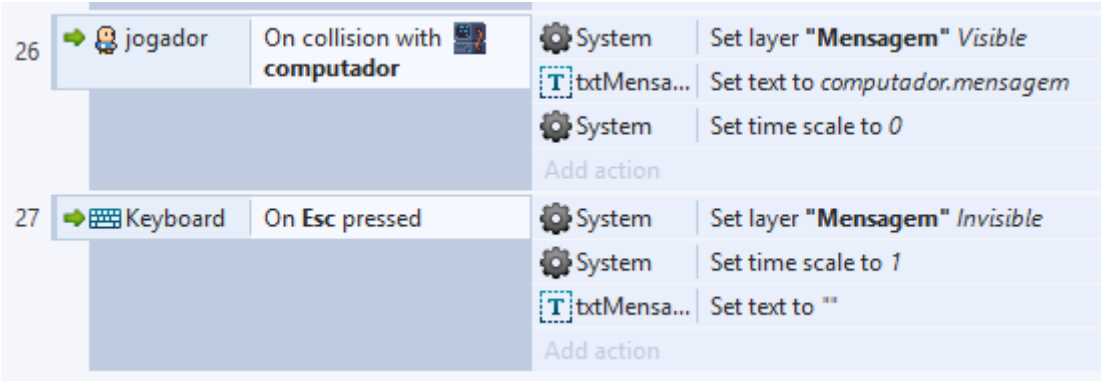


Figura 28 – Exibir e Fechar Mensagens de Informação.

1.4.14. Criação dos Finais da Fase

Assim como foi definido para a morte do jogador o final da fase irá ocorrer de acordo com as regras do processo de negócio, ou seja, o jogo termina com sucesso quando o jogador conseguir emitir o cartão SUS. O GDD define que deverá ter ao final da fase um Sprite que remete ao posto de atendimento do SUS onde o cartão será confeccionado. Desta forma, inclua um novo Sprite de nome "***PostoSUS***", com a imagem "hospital" que está em "***assets/itens***".

Haverá duas ações associadas a este Sprite. A primeira, caso todos os documentos não estejam coletados, será exibida uma mensagem para buscar os documentos pela fase e somente depois voltar ao posto de atendimento. A Segunda, habilitará a colisão e exibirá o objeto de cartão SUS, então uma mensagem sobre a localização do documento será mostrada para o jogador (Figura 29).

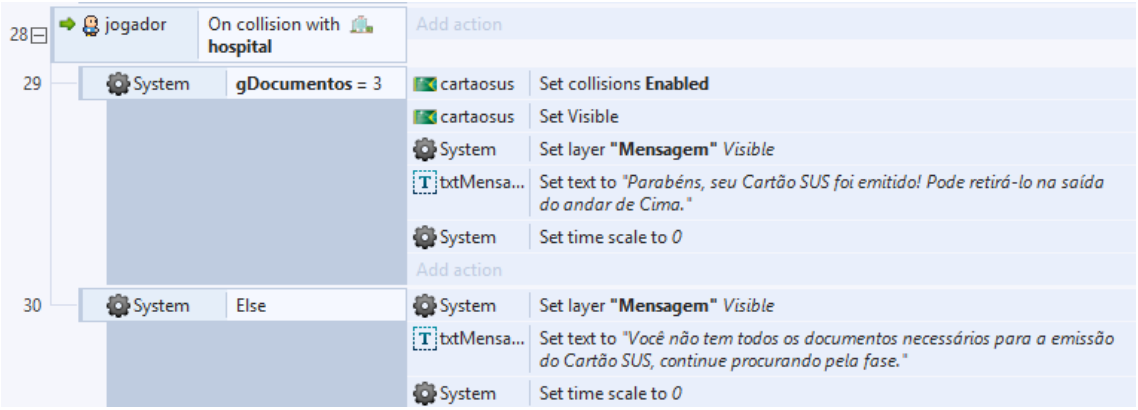


Figura 29 – Eventos do Hospital para Emissão do Cartão SUS

Para concluir o processo, basta o usuário coletar o cartão SUS no qual será exibida a mensagem de Sucesso do Final de Jogo. Ajuste o evento da colisão do cartão SUS seguindo o mostrado na Figura 30.

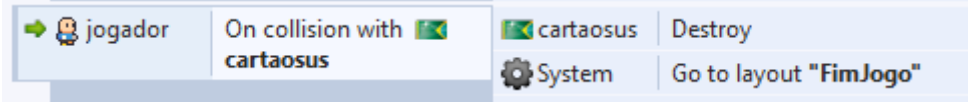


Figura 30 – Conclusão do Evento de Colisão do Cartão SUS

1.4.15. Finalização do Jogo

Para finalizarmos o jogo alguns detalhes precisam ser feitos, como: inclusão de uma tela para o game over e sons.

1.4.15.1. Tela de Game Over

Para incluir a tela de game over inclua um novo Layout e dê a ele o nome de ***"FimJogo"***. Ajuste seu tamanho para ficar do tamanho da câmera do jogo. Coloque um objeto de texto alinhado ao centro de nome "***txtGameOver***". Quando um layout novo é criado, também é gerado um novo event sheet, renomeie para "***evGameOver***".

A cada ciclo de processamento do jogo verifique se as vidas chegaram a zero, conforme a Figura 31.

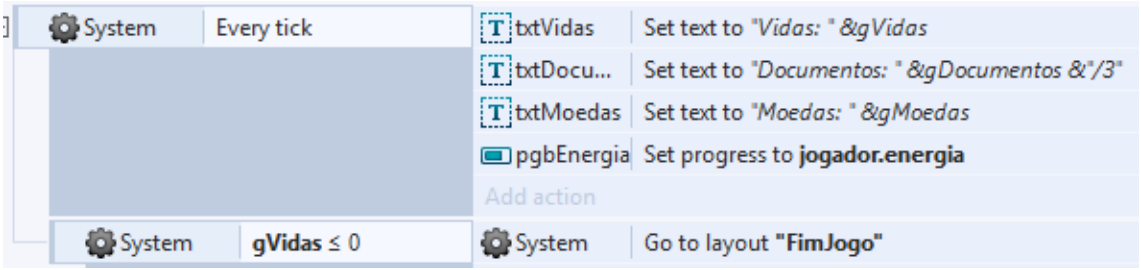


Figura 31 – Game Over por Falta de Chances

Dentro da ***Event Sheet*** "***evGameOver***" para registrar as mensagens de falhas e de sucesso corretas para o jogador coloque os eventos mostrados na Figura 32.

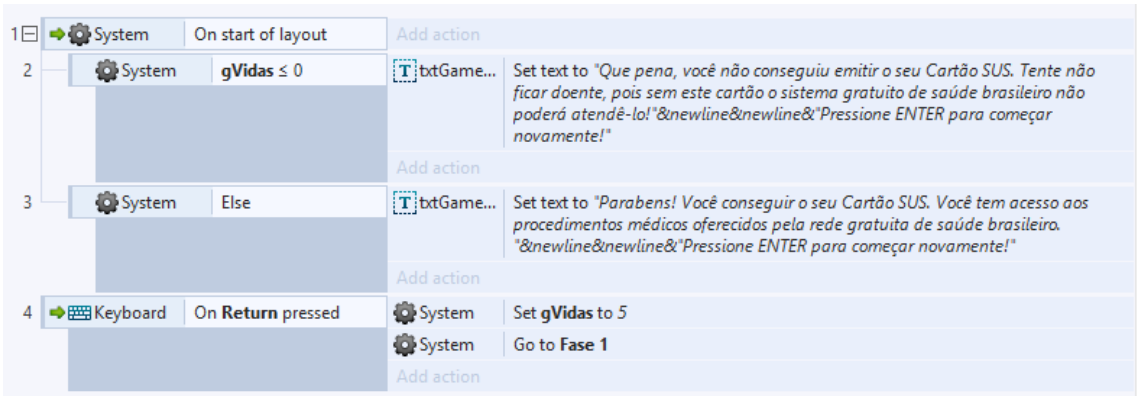


Figura 32 – Eventos da Tela de Game Over

1.4.15.2. Sons do Jogo

Um jogo de plataforma sempre remete a sons de 16 bits. Portanto devemos incluir sons no jogo. Para incluir sons no Construct é preciso incluir no projeto um objeto de áudio, e para isso, basta clicar com o botão direito do mouse em qualquer layout e inserir um novo objeto de "***Audio***". Este objeto uma vez incluso serve para todo o projeto.

Existe uma pasta em Projects chamada "***Music***" e se clicarmos nela com o botão direito, a opção "***import music***" irá surgir. Todos os áudios que vamos importar está dentro da pasta "***assets/sons***". Uma vez importados ao projeto, basta o mesmo ser utilizado nos seus eventos. Por exemplo: i) em todo evento de colisão ou morte chamaremos o evento de executar som "***dano2***" (Figura 33); ii) colisão com moedas,

"***coin***"; iii) saltar/pular, "***new_jump***"; iv) pegar o cartão sus, "***victory***"; e v) música de fundo que fica executando durante a fase, "***parte1_2***" (Figura 34).

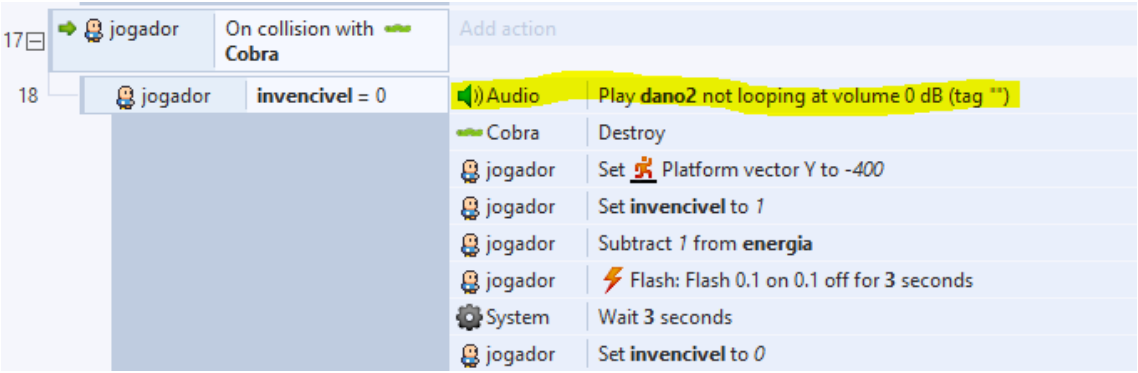


Figura 33 – Som de Tomada de Dano nas Colisões com Inimigo

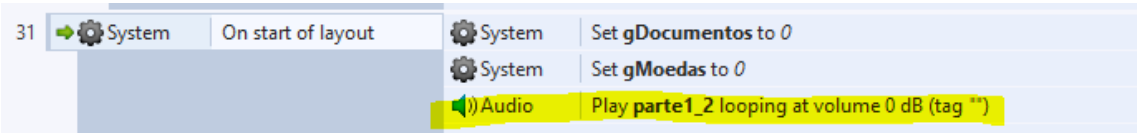


Figura 34 – Música da Fase

1.5. Conclusões

Este minicurso teve como objetivo principal a construção de um jogo digital baseado em um processo de serviço público especificamente a partir do processo de Emissão do Cartão SUS, o qual permite que todo brasileiro tenha acesso aos procedimentos médicos da rede pública de saúde.

As construções destes jogos visam auxiliar na compreensão dos cidadãos sobre a necessidade de determinados serviços públicos, não somente suas regras e etapas, mas também, transmitir aos jogadores, quem são os prestadores do serviço, o motivo da criação do processo, as dificuldades de sua entrega aos seus clientes, os valores embutidos neles e outros aspectos. Portanto, ao consumir estes jogos, o jogador é convidado a refletir sobre as instituições públicas e a forma com que elas realizam seu trabalho.

O método de design de jogos baseados em processo proposto por Classe et al. [2016] visa permitir que os designers de jogos possam construir seus jogos pensando nos contextos organizacionais, de prestação do serviço e na percepção do cidadão. Desta forma, considerando este método, este curso buscou demonstrar a execução da quarta etapa do método de design, que consistiu do desenvolvimento (programação) de um jogo digital para a compreensão do cidadão, a partir dos documentos originados pela equipe de design de jogo.

As avaliações acerca da compreensão do processo pelos jogadores, também é prevista pelo método de design, entretanto, não é o foco deste curso. Existem algumas escalas e métricas que podem ser utilizadas para realizar a avaliação do jogo. Estas abordagens focam em averiguar a percepção da qualidade do jogo sob a ótica do jogador, mas também, servem para verificar o grau de conhecimento adquirido por eles com o jogo. O que queríamos transmitir com este curso é que, por meio de um processo de negócio de prestação de serviços públicos, é possível obter elementos deste processo

através de seus modelos, construir uma especificação de requisitos de jogos e, com esta especifica, o jogo pode ser construído.

Espera-se que com este curso, os participantes consigam refletir sobre a necessidade de que a sociedade precisa compreender os serviços públicos, e que os jogos digitais podem ser uma maneira de transmitir tal compreensão. Pois acreditamos que ao compreenderem um processo, as pessoas possam refletir sobre suas necessidades, pensando inclusive em possíveis melhorias, as quais podem ser compartilhadas com as organizações.

Referências

Abt, C.C. (1970). “Serious Games”. Viking Press, New York.

Adams, E., Rollings, A. (2007). "Fundamentals of game design". Prentice Hall.

Aguilas-Saven, R.S. (2004). "Business process modelling: Review and framework". In: International Journal of production economics, v.90(2), pp. 129-149.

Alves, E. (2013). "Jogos Sérios para Ensino de Engenharia de Software". Dissertação de Mestrado, Faculdade de Engenharia da Universidade do Porto, Portugal. Disponível em: <https://repositorio-aberto.up.pt/bitstream/10216/68502/2/53127.pdf>. Acesso em: 22 de dezembro de 2016.

Araujo, R.M., Magdaleno, A.M. (2015). “Social BPM: Processos de Negócio, Colaboração e Tecnologia Social”, In Simpósio Brasileiro de Sistemas de Informação (SBSI).

Bertot, J; Estevez, E.; Janowski, T. (2016). “Universal and contextualized public services: Digital public service innovation framework”. In: Government Information Quarterly, v.33(2), pp. 211-222.

Brown, A.W.; Fisheden, J.; Thompson, M. (2014). “Revolutionising Digital Public Service Delivery: A UK Government Perspective”. Available at: <http://www.blogs.jbs.cam.ac.uk/markthompson/wp-content/uploads/2014/02/Digital-Public-Service-Delivery.pdf>. Access in: may 03 2017.

Classe, T., R. Araujo, R. (2016). “Jogos Digitais Para Participação Cidadã em Processos de Prestação de Serviços Públicos”, In Workshop de Teses e Dissertações SBSI (WTDSI).

Classe, T.; Araujo, R.M.; Xexéo, G.B. (2018). “Jogos Digitais Baseados em Processos de Prestação de Serviços Públicos: Um Estudo Exploratório”, International Journal of Game Studies (Ludica), v.2(2), pp. 26-56.

Connolly, T.M,; Boyle, E.A.; Macarthur, E.; Hainey, T.; Boyle, J.M. (2012). "A systematic literature review of empirical evidence on computer games and serious games". In: Computers & Education, v.59(2), pp. 661-686.

Crawford, C. (2003). "Chris Crawford on game design". New Riders.

Daniels, A. (2016). "Quality in Public Service Delivery". In: International Journal of Civil Service Reform and Practice, v.1(2).

Deterding, S.; Dixon, D.; Khaled, R.; Nacke, L. (2011). "From game design elements to gamefulness". In: Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11, pp. 9–11.

Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A. (2013). "Fundamentals of business process management", Springer, Berlin.

Fiocruz. (2017). "SUS: O que é?". Pense SUS. Disponível em: <http://pensesus.fiocruz.br/sus>. Acessado em: 08 de maio de 2017.

Grölund, A. (2010). "Ten years of E-Government: The "End of History" and New Beginning". In: 9TH IFIP WG 8.5 INTERNATIONAL CONFERENCE, EGOV 2010. Lausanne, Switzerland: Springer.

Ilomäki L.; Kankaanranta, M. (2009). "The information and communication technology (ICT) competence of the young". In: Handbook of research on new media literacy at the K-12 level: Issues and challenges, pp.101-118.

Juul, J. (2009). "A Casual Revolution: Reinventing Video Games and Their Players". The MIT Press.

Kneuer, M. (2016). "E-democracy: A new challenge for measuring democracy". In: International Political Science Review, v.37(5), pp. 666-678.

Ko, R.K. (2009). "A computer scientist's introductory guide to business process management (BPM)". In. ACM Crossroads, v.15(4), pp. 4.

Marston, H. R. (2015). "Gamification: Applications for Health Promotion". In: Handbook of Research on Holistic Perspectives in Gamification for Clinical Practice. p.78.

Mastrocola, V.M. (2012). "Ludificador: um guia de referências para o game designer brasileiro". São Paulo: Independente.

Michael, D., Chen, S. (2005). "Serious Games - Games that Educate, Train, and Inform.". Thomson Course Technology PTR, Boston.

OMG. (2013) "Business ProcessModelandNotation (BPMN)". OMG. Disponível em: <http://www.omg.org/spec/BPMN/2.0.2/PDF>. Acessoem: 10 de novembro de 2016.

Pflanzal, N.; Vossen, G. (2014). "Challenges of Social Business Process Management". In 47th Hawaii International Conference on System Science, pp. 3868-3877.

Portal Brasil. (2012). "Saiba Como Solicitar Seu Cartão SUS. Cidadania e Justiça. Disponível em: <http://www.brasil.gov.br/cidadania-e-justica/2012/03/saiba-como-solicitaro-seu-cartao-sus>. Acessado em: 08 de Maio de 2017.

Reijers, H.A., Slaats, T., Stahl, C. (2013). "Declarative modeling--An academic dream or the future for BPM?". Business Process Management, Springer, pp. 307-322.

Rocha, R.V., Araújo, R.B. (2013). "Metodologia de Design de Jogos Sérios para Treinamento: Ciclo de vida de criação, desenvolvimento e produção". In: XII Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames 2013), pp. 1-10.

Romero, M., Usart, M., Ott, M. (2015). "Can serious games contribute to developing and sustaining 21st century skills?". In: Games and Culture, v.10(2), pp. 148-177.

Salen, K., Zimmerman, E. (2003). "Rules of play: Game design fundamentals". Cambridge, Mass.: MIT Press.

Santos, P.M. (2014). “Framework de Apoio à Democracia Eletrônica em Portais de Governo Com Base nas Práticas de Gestão do Conhecimento”. Doctoral Thesis, Universidade Federal de Santa Catariza, Programa de Pós-Graduação em Engenharia e Gestão do Conhecimento.

Schell, J. (2009). “The Art of Game Design A Book of Lenses”. Burlington, USA: Morgan Kaufmann Publishers & Elsevier.

Schuytema , P. (2008). “Design de games: uma abordagem prática”. São Paulo: Cengage Learning, pp. 447.

Sharp, A., McDermott, P. (2008). "Workflow modeling: tools for process improvement and applications development". Artech House.

Silva, S. P. (2005). “Graus de participação democrática no uso da Internet pelos Governos das capitais brasileiras”. In Opinião Pública, v. XI(2), pp. 450-468.

Sobreira Neto, F. (2009). "Gerenci amento de Processos de Negócio – BPM segundo a Gestão Empresarial e a Tecnologia da Informação: uma Revisão Conceitual". In: XXXIII Encontro da ANPAD, São Paulo.

Souza, M.V., Medeiros, J.V. (2008). "Afinal, o que é business process management (BPM)? Um novo conceito para um novo contexto". In. Revista Eletrônica de Sistema de Informação, v.7.

Susi, T., Johannesson, M., Backlund, P. (2007). "Serious Games : An Overview". Sweden: InstitutionenFörKommunikationOch Information, pp. 28.

Tavares, A.; Soares, D.; Estevez, E. (2016). "Electronic Governance for Context-Specific Public Service Delivery: a Survey of the Literature". In: 9th International Conference on Theory and Practice of Electronic Governance, pp. 135-138.

Teixeira, C. (2011). "Os Princípios do Sistema Único de Saúde". Conferências Municipal e Estadual de Saúde, Salvador. Disponível em: <http://www.saude.ba.gov.br/pdf/OS_PRINCIPIOS_DO_SUS.pdf>. Acessado em: 08 de maio de 2017.

Vedel, T. (2006). “The idea of electronic democracy: Origins, visions and questions”. In: Parliamentary Affairs, v.59(2), pp. 226-235.

Winters, M.S.; Karim, A.G.; Martawardaya, B. (2014). “Public Service Provision under Conditions of Insufficient Citizen Demand: Insights from the Urban Sanitation Sector in Indonesia”. In: World development, v.60, pp. 31-42.

Xexéo, G.; Carmo, A.; Acioli, A.; Taucei, B.; Dipolitto, C.; Mangeli, E.; Kritz, J.; Costa, L.F.C.; Areas, M.; Monclar, R.; Garrot, R.; Classe, T.; Azevedo, V. (2017). "O Que São Jogos: Uma Introdução ao Objeto de Estudo do Ludes". Relatório Técnico do Programa de Engenharia de Sistemas e Computação, n. 5/2017 (ES-752/17), (COPPE/UFRJ). Disponível em: <http://www.cos.ufrj.br/index.php/ptBR/publicacoes-pesquisa/details/15/2766>. Acessoem 11 de maio de 2017.

Autores

Tadeu Moreira de Classe é Doutorando no Programa de Pós-Graduação em Informática da UNIRIO. Mestre em Ciência da Computação na Universidade Federal de Juiz de Fora (PGCC / UFJF). Graduado no Curso Bacharelado em Sistemas de Informações do Centro de Ensino Superior de Juiz de Fora (CES / JF). Professor universitário com mais de 4 anos de experiência na área de Sistemas de Informação e Analista de Sistemas com mais de 8 anos experiência. Membro do Grupo de Pesquisa e Inovação em CiberDemocracia (CIBERDEM / UNIRIO) e Laboratório de Ludologia, Engenharia e Simulação (COPPE / UFRJ). Tópicos de pesquisa são: Sistemas de Informação, Democracia Eletrônica e Jogos Digitais. Detalhes em http://lattes.cnpq.br/4540774422689570.

Renata Araujo é Doutora em Engenharia de Sistemas e Computação. Seus tópicos de pesquisa são: Sistemas de Informação, Democracia e Governança Digital, Gestão de Processos de Negócio e Gestão da Inovação. Atualmente ocupa a Diretoria de Educação da SBC (2018-2019). Renata é uma das editoras do livro Pesquisa e Inovação, da editora PUBL!T Soluções Editoriais. Renata é bolsista de Produtividade em Desenvolvimento Tecnológico e Extensão Inovadora do CNPq, Brasil processo no 305060/2016-3. Detalhes em http://lattes.cnpq.br/3589012014320121.

Geraldo Bonorino Xexéo tem Graduação em Engenharia Eletrônica pelo Instituto Militar de Engenharia (1988) e Doutorado em Engenharia de Sistemas e Computação pelo Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia (1994). Desde 1995 é professor da Universidade Federal do Rio de Janeiro. Tem experiência na área de Ciência da Computação, com ênfase em Banco de Dados e Engenharia de Software. Suas linhas de pesquisa atuais incluem Sistemas Peer-to-Peer, Busca, Recuperação e Extração da Informação, Qualidade de Dados e Lógica Fuzzy. Participou de projetos de consultoria em empresas públicas e privadas. Detalhes em http://lattes.cnpq.br/4783565791787812.