Certified Tester

Game Testing Syllabus

versão 1.0.1

International Software Testing Qualifications Board

Fornecido por:

Russian Software Testing Qualifications Board (RSTQB)

Direitos autorais

Histórico da Revisão

Versão	Data	Observações
1.0.1	21/10/2022	Lançamento no GA

Histórico da versão BSTQB

Versão	Data	Observações
1	14/06/2023	Padronização do layout com o ISTQB
2	28/06/2024	Adequação visual
3	06/01/2025	Adequação ao novo visual do ISTQB

Índice

Agradecimentos

Este documento foi produzido pela equipe central do RSTQB (Russian Software Testing Qualifications Board), composta por:

Andrey Konushin (President of the Board)
Margarita Trofimova (Vice-president of the Board)
Alexander Alexandrov (Editor-in-Chief)
Vadim Lukovaty
Maksim Nikolaev
Pavel Sharikov
Alexander Torgovkin
Svetlana Yushina

A equipe central agradece à equipe de revisão por suas sugestões e contribuições. O Russian Software Testing Qualifications Board gostaria de reconhecer e agradecer à LLC "Bytex" por sua contribuição para o desenvolvimento deste syllabus.

Além disso, a equipe central gostaria de reconhecer e agradecer aos líderes e membros dos Grupos de Trabalho por sua orientação inicial e contínua: Galit Zucker (Secretário Geral), Graham Bath (presidente, Grupo de Trabalho Especialista), Gary Mogyorodi (membro, Grupo de Trabalho Glossário) e Klaus Skafte (presidente, Grupo de Trabalho Exame). As seguintes pessoas participaram da revisão, comentário ou votação deste programa de estudos ou de seus antecessores:

Ivan Pchelkin
Matthias Hamburg
Tal Pe'er
Meile Posthuma
Wojciech Becla J
Nitzan Goldenberg
Georg Haupt
Bíró Ádám
Gergely Ágnecz
Natalia Sterkhova
Chunhui Li
Wim Decoutere
Nishan Portoyan
Ana Gierloff
Jurgen Beniermann
Gary Mogyorodi
Darvay Tamás Béla
Graham Bath
Irina Yakovleva
Blair Mo
Claude Zhang
Francisca Cano Ortiz
Paul Weymouth
Erwin Engelsma
Péter Sótér
Laura Albert
Mike Smith

Este syllabus foi oficialmente lançado em 21 de outubro de 2022.

O BSTQB® agradece aos voluntários do WGT BSTQB® pelo empenho e esforço na tradução e revisão deste material: Eduardo Medeiros Rodrigues, George Fialkovitz Jr., Irene Nagase, Osmar Higashi, Rogério Athaide Almeida, Stênio Viveiros, Yan Fialkovitz.

0 Introdução

0.1 Objetivo deste Syllabus

Este syllabus forma a base para o ISTQB® *Certified Tester Game Testing*. O ISTQB® fornece este syllabus da seguinte forma:

1. Aos **Conselhos Membros**, para traduzir para seu idioma local e para credenciar os provedores de treinamento. Os Conselhos Membros podem adaptar o syllabus às suas necessidades linguísticas particulares e modificar as referências para se adaptarem às suas publicações locais.
2. Aos **Provedores de Exame**, para derivar perguntas de exame em sua língua local adaptadas aos objetivos de aprendizagem deste syllabus.
3. Aos **Provedores de Treinamento**, para produzir material didático e determinar métodos de ensino apropriados.
4. Aos **Candidatos** à certificação, para preparar-se para o exame de certificação (seja como parte de um curso de treinamento ou independentemente).
5. À **Comunidade Internacional** de engenharia de software e sistemas, para promover a profissão de teste de software e sistemas, e como base para livros e artigos.

O ISTQB® pode permitir que outras entidades utilizem este programa para outros fins, desde que busquem e obtenham permissão prévia por escrito.

0.2 O Certified Tester Game Testing

A *Certified Tester Game Testing* é destinada a qualquer pessoa envolvida em testes de software que deseje ampliar seus conhecimentos sobre teste de jogos ou qualquer pessoa que deseje iniciar uma carreira de especialista em teste de jogos. A qualificação também é destinada a qualquer pessoa envolvida em engenharia de software de jogo que deseje obter uma melhor compreensão do teste de jogos.

O syllabus considera os seguintes aspectos principais do teste de jogos:

- Aspectos técnicos;
- Aspectos baseados em métodos;
- Aspectos organizacionais.

0.3 Carreira para os testadores

Construído para o Nível Fundamental, o Game Testing apoia a definição de caminhos de carreira para os testadores profissionais. Uma pessoa certificada com o Game Testing terá ampliado o entendimento dos testes adquiridos no Nível Básico para permitir que ela possa trabalhar efetivamente como um testador profissional em um projeto de jogo.

Aqueles que possuem um certificado de Game Testing podem usar a sigla CT-GaMe.

Por favor, visite [ISTQB-Web] para obter a mais recente visão geral da trajetória profissional do ISTQB®.

0.4 Resultados de Negócio

Esta seção lista os resultados de negócio esperados de um candidato que obteve a certificação de Game Testing.

GaMe-1: Descrever conceitos básicos de testes de videogames e software de jogo;
GaMe-2: Determinar riscos, objetivos e requisitos de software de jogo sob as necessidades e expectativas dos stakeholders;
GaMe-3: Conceitualmente projetar, implementar e executar testes básicos de software de jogo;
GaMe-4: Conhecer as abordagens aos testes de software de jogo e sua finalidade;
GaMe-5: Reconhecer as ferramentas de apoio aos testes de jogo;

GaMe-6: Determinar como as atividades de teste se alinham ao ciclo de vida de desenvolvimento de software e reduzir o custo de desenvolvimento e publicação de jogos;

0.5 Objetivos de aprendizagem e nível cognitivo de conhecimento

Os objetivos de aprendizagem apoiam a realização dos objetivos de negócio e são usados para projetar exames para a certificação em Game Testing. Cada objetivo de aprendizado refere-se a um processo cognitivo (nível K).

Um nível K, ou nível cognitivo, é usado para classificar os objetivos de aprendizagem de acordo com a taxonomia revista da Bloom [Anderson01]. O ISTQB® utiliza esta taxonomia para desenvolver seus syllabi (ver Anexo A para maiores detalhes).

Este syllabus considera três níveis diferentes de K (K1 a K3):

K	Palavras-chave	Descrição
K1	Lembrar	O candidato deve lembrar ou reconhecer um termo ou um conceito.
K2	Entender	O candidato deve selecionar uma explicação para uma declaração relacionada ao tema da pergunta.
K3	Aplicar	O candidato deve selecionar a aplicação correta de um conceito ou técnica e aplicá-la a um determinado contexto.

Em geral, todas as partes deste syllabus contêm objetivos de aprendizagem para o nível K1. Em outras palavras, um candidato a uma certificação deve reconhecer, lembrar e reproduzir um termo ou conceito. Os objetivos de aprendizagem para os níveis K2 e K3 são indicados no início de cada capítulo.

0.6 O exame de certificação Certified Tester Game Testing

O exame de certificação *Certified Tester Game Testing* será baseado neste syllabus. As respostas às perguntas do exame podem exigir o uso de material baseado em mais de uma seção deste syllabus. Todas as seções do syllabus são examináveis, exceto a Introdução e os Anexos. Normas e livros estão incluídos como referências, mas seu conteúdo não é examinável, além do que está resumido no próprio syllabus a partir de tais normas e livros.

Consulte [ISTQB_EXAM_S&R] para a Estrutura e Regras do Exame de Teste de jogos Certificado.

A certificação ISTQB® *Certified Tester Foundation Level* [ISTQB_FL_SYL] deve ser obtida antes de fazer o exame de certificação *Certified Tester Game Testing*. Entretanto, é fortemente recomendado que os candidatos também:

- tenham pelo menos uma formação mínima em desenvolvimento ou teste de software de jogos, como 1 ano de experiência como testador de aceite de usuários de software de jogos.
- fazer um treinamento que tenha sido credenciado nos padrões do ISTQB® (por um dos conselhos de membros reconhecidos pelo ISTQB).

0.7 Credenciamento

Um Conselho Membro do ISTQB® pode credenciar provedores de treinamento cujo material didático siga este syllabus. Os provedores de treinamento devem obter diretrizes de credenciamento do Conselho Membro ou do órgão que realiza o credenciamento. Um curso credenciado é reconhecido como estando em conformidade com este syllabus e é permitido ter um exame ISTQB® como parte do curso.

As diretrizes de credenciamento seguem as *General Accreditation Guidelines* publicadas pelo *Processes Management and Compliance Working Group*.

0.8 Manuseio de Normas

Existem normas referenciadas neste syllabus (p. ex., (IEEE, ISO etc.). O objetivo dessas referências é fornecer uma estrutura (como nas referências à ISO 25010 em relação às características de qualidade) ou fornecer uma fonte de informações adicionais, se desejado pelo leitor. Favor observar que o syllabus está usando estes documentos normativos como referência. Os documentos normativos não são destinados a exame. Consulte a seção 8. Referências para mais informações sobre Normas.

0.9 Atualizações

A indústria de software muda rapidamente. Para lidar com essas mudanças e fornecer aos interessados acesso a informações relevantes e atuais, os grupos de trabalho ISTQB® criaram links no [ISTQBWeb] que se referem a documentos de apoio, mudanças nas normas e novas ocorrências na indústria.

Estas informações não serão examinadas no âmbito deste syllabus.

0.10 Nível de detalhe

O nível de detalhes deste syllabus permite cursos e exames internacionalmente consistentes. Para atingir este objetivo, o syllabus consiste em:

- Objetivos gerais de instrução descrevendo a intenção do *Certified Tester Game Testing*;
- Uma lista de termos que os alunos devem ser capazes de lembrar;
- Objetivos de aprendizagem para cada área de conhecimento, descrevendo o resultado do aprendizado cognitivo a ser alcançado;
- Uma descrição dos conceitos-chave, incluindo referências a fontes como literatura ou padrões aceitos.

O conteúdo do syllabus não é uma descrição de toda a área de conhecimento do teste de jogos. Ele reflete o nível de detalhe a ser coberto nos cursos de treinamento de *Certified Tester Game Testing*. Ele se concentra nas áreas de testes e técnicas que podem ser aplicadas à maioria dos projetos de jogos.

0.11 Como este syllabus está organizado

O syllabus *Certified Tester Game Testing* contém sete capítulos que cobrem o conhecimento necessário para ser um especialista em teste de jogos. O título de cada capítulo especifica o tempo mínimo de estudo para o capítulo; o tempo de estudo não é fornecido abaixo deste nível. Para cursos de treinamento certificados, o syllabus exige um mínimo de 15 horas e 25 minutos de instrução, distribuídos pelos sete capítulos da seguinte forma:

Capítulo 1 - Características do teste de jogos – 75 minutos

- O testador aprende os princípios básicos relacionados aos testes de software de jogo, as razões pelas quais o teste de jogos é necessários, quais são os objetivos do teste e a diferença entre o teste de jogos e o jogo;
- O testador compreende o processo de teste de jogos, as principais atividades e produtos de trabalho.

Capítulo 2 - Testando a mecânica de jogo - 180 minutos

- O testador aprende a distinguir entre as diferentes mecânicas do jogo e suas abordagens de teste;
- O testador aprende os métodos dinâmicos e estáticos usados para testes de mecânica de jogo.

Capítulo 3 - Testes gráficos - 165 minutos

- O testador aprende sobre as características e tipos do conteúdo gráfico do produto do jogo e as melhores práticas de execução de testes gráficos, incluindo seu suporte de ferramentas;
- O testador aprende sobre as áreas de responsabilidade das pessoas envolvidas

Capítulo 4 - Testes de som - 190 minutos

- O testador aprende sobre tipos e tecnologias do conteúdo sonoro do produto do jogo, melhores práticas de execução de testes sonoros incluindo seu suporte de ferramenta;
- O testador aprende sobre as áreas de responsabilidade das pessoas envolvidas.

Capítulo 5 - Teste de nível de jogo - 65 minutos

- O testador aprende sobre a dependência do projeto de nível e abordagens de teste sobre o gênero do jogo e seu suporte de ferramenta;
- O testador aprende sobre as áreas de responsabilidade das pessoas envolvidas

Capítulo 6 - Teste de controladores de jogo - 95 minutos

- O testador aprende sobre os tipos de controladores de jogo, incluindo defeitos relacionados e suas possíveis causas;
- O testador aprende abordagens para testar controladores e seu suporte a ferramentas.

Capítulo 7 - Teste de regionalização - 155 minutos

- O testador aprende as bases e diferenças de regionalização e internacionalização, seus riscos, abordagens para testar e seu suporte a ferramentas;
- O testador aprende sobre as áreas de responsabilidade das pessoas envolvidas.

1 Características do teste de jogos - 75 min

Palavras-chave

Teste funcional, teste de jogos

Palavras-chave específicas

modelo 3D, estágio de conceito, desenvolvedor de jogos, multiplataforma, estágio de pós-produção, fase de pré-produção, fase de produção, jogos

Objetivos de aprendizagem

1.1 Noções básicas de teste de jogos

GaMe-1.1.1 (K1) Reconhecer os objetivos e especificidades dos testes de jogo.

GaMe-1.1.2 (K2) Dar exemplos de riscos do produto em software de jogo.

GaMe-1.1.3 (K2) Dar exemplos de defeitos específicos relacionados a testes de jogo.

GaMe-1.1.4 (K2) Resumir como os riscos do teste de jogos podem ser mitigados.

GaMe-1.1.5 (K2) Comparar as atividades de teste do jogo com as de jogo.

1.2 Papéis típicos da equipe de desenvolvimento de jogos

GaMe-1.2.1 (K1) Reconhecer funções e tarefas específicas na equipe de desenvolvimento do jogo.

1.3 Atividades de Teste durante todo o Ciclo de Vida do Desenvolvimento de Jogos

GaMe-1.3.1 (K1) Atividades de teste de recall durante todo o ciclo de vida de desenvolvimento de software de jogo.

1.1 Noções básicas de teste de jogos

1.1.1 Características do teste de jogos

Nos testes de software, é costume distinguir entre várias especializações. Alguns testadores estão empenhados em testar a performance de aplicações, outros se concentram em testar aplicações móveis, outros procuram vulnerabilidades em sistemas de segurança e alguns procuram defeitos em jogos de computador.

Os jogos podem variar em gênero, e serem destinados a um único ou a vários jogadores. Entretanto, as abordagens para testar jogos podem ser consideradas como independentes destas características. A seção 2.1 deste syllabus descreve em detalhes as abordagens de testes funcionais e não funcionais da mecânica dos jogos.

É importante notar que os termos "jogo", "jogo de computador" e "vídeo game" são tratados como sinônimos neste syllabus. Isto é para tornar o texto mais legível e dentro do contexto do syllabus. Um jogo só pode ser considerado como software e não pode ser confundido com jogos esportivos ou jogos de azar.

1.1.2 Riscos dos produtos do jogo

O desenvolvimento explosivo da indústria de jogos levou à sua distribuição em muitas plataformas diferentes. Os jogos se tornaram maiores, mais complexos e mais exigentes em termos de recursos técnicos. Ao mesmo tempo, o público de jogadores aumentou significativamente, tornando-se mais sofisticado e exigente em relação à qualidade dos jogos.

Projetos comuns de software e riscos de produtos se aplicam a videogames e desenvolvimento de jogos. Como resultado, os jogos, como outros softwares, estão sujeitos a testes. Alguns riscos são específicos à área de jogos de software e devem receber mais atenção. Tais riscos incluem, mas não estão limitados a:

- Ganhar vantagem injusta ao trapacear;
- Dependência do sucesso no mercado a partir da opinião subjetiva dos jogadores;
- Questões multiplataforma;
- Complexidade de prever a eficácia da mecânica de jogo (ver seção 2.1).

Os testadores podem encontrar defeitos nas seguintes áreas:

- Na arquitetura do software de jogos no início do ciclo de vida do desenvolvimento;
- No projeto do som;
- No texto de um vídeo game que já está disponível para lançamento.

1.1.3 Defeitos relacionados a teste de jogos

Há uma série de defeitos que são especialmente comuns nos videogames. Estes incluem:

- problemas com gráficos e animação;
- violação das leis físicas do mundo dos videogames;
- defeitos na inteligência artificial do videogame;
- inexatidões na construção do terreno dos videogames;
- funcionamento incorreto de alguns elementos da interface (p. ex., personagens de videogames surgindo do ar, não atacando inimigos e pairando sobre carros).

Embora tais defeitos possam ser fáceis de detectar, identificar os passos exatos para reproduzi-los é muitas vezes difícil. Esta complexidade torna o teste de jogos desafiadores. Um bom testador deve projetar testes de acordo com o cenário de jogo pretendido, verificar se ele atende à especificação, analisar possíveis desvios do cenário projetado e relatar as consequências desses desvios.

Testes negativos

Ao testar jogos, como qualquer software, é importante identificar defeitos óbvios e esforçar-se para encontrar defeitos implícitos que aparecem como resultado de ações não-padronizadas realizadas pelo usuário. Por exemplo, ao invés de lutar contra adversários e procurar uma chave para uma porta trancada, o usuário pode coletar caixas dispostas no todo o nível, fazer uma escada com elas, e subir e passar sobre a porta que limita o nível, evitando assim uma luta difícil, o que não era a intenção do jogo.

Outros defeitos podem ocorrer se a interação com outros objetos for definida incorretamente, tais como para um modelo 3D de um edifício usado no jogo. Este tipo de defeito pode permitir a passagem incorreta de um personagem controlado pelo jogador através de uma parede, o que pode estragar a experiência geral do usuário e dar aos jogadores vantagens ilegais no jogo. Isto é especialmente crítico em jogos multiplayer. Atrás de tal parede, o jogador estará escondido dos adversários, mas será capaz de atirar neles.

Riscos como estes aumentam a importância de realizar testes negativos para projetos de jogos. Pode haver muitos jogadores que não querem jogar como o desenvolvedor pretendia, mas tentam "quebrar" ou contornar o sistema a fim de ganhar mais facilmente e gastar menos esforço. Para conseguir isto, nem sempre é necessário usar software de terceiros ou comandos especiais. Pode ser o suficiente para encontrar o defeito "correto". Tais jogadores procuram deliberadamente por defeitos na lógica do jogo e os utilizam para vantagem pessoal, como por exemplo, obter acesso a recursos infinitos ou armas poderosas. Em jogos monetizados, isto também poderia levar a uma redução nos lucros do proprietário do jogo.

Dependência da opinião dos jogadores

O software de jogo, como nenhum outro, depende do julgamento subjetivo e muitas vezes emocional dos jogadores finais. Para softwares projetados para resolver problemas comerciais, sua funcionalidade é o fator mais importante. Para o software de jogos, o fator mais importante é o nível de interesse do usuário e suas impressões sobre a sessão de jogos. Os jogadores podem ser capazes de "fechar os olhos" em relação a alguns defeitos funcionais, mas se o jogo se revelar chato, monótono ou usar gráficos desatualizados, os usuários podem simplesmente parar de usá-lo. Uma parte significativa do público de videogames toma decisões sobre a compra ou uso de um produto com base principalmente no feedback de críticos de jogos, revisores e outros líderes de opinião.

Multiplataforma

Uma característica importante do software de jogos é que ele é frequentemente usado em múltiplas plataformas. Em um esforço para alcançar o maior público possível, os estúdios/editores de jogos o lançam em uma grande variedade de plataformas, incluindo vários computadores pessoais (PC), web, dispositivos móveis e consoles.

Cada atualização tem que ser testada para cada plataforma disponível. Isto dá origem a sérios riscos e aumenta significativamente o tempo de teste necessário.

O jogo pode funcionar bem em um computador de médio porte, mas tem uma grande variedade de defeitos em consoles de última geração. Defeitos também são possíveis devido à comunicação ineficiente entre várias tecnologias ou requisitos incompletos para portar um jogo da plataforma para a qual ele foi originalmente desenvolvido para outra plataforma pertencente a uma organização de terceiros (terceirizada).

Para mitigar estes riscos, recomenda-se dedicar mais tempo ao desenvolvimento e teste do software do jogo, testando a funcionalidade e a performance do jogo em muitas configurações diferentes de hardware e realizando testes baseados nas características específicas de cada plataforma.

O teste de jogos em consoles de jogos é um aspecto particular a ser considerado. Um console de jogos é um dispositivo projetado exclusivamente para jogos e não contém nenhum outro software, como dispositivos móveis ou computadores.

Há poucas diferenças entre plataformas quando se testa videogames usando técnicas de teste de caixas pretas. Entretanto, cada fabricante de console de jogos pode ter seus próprios requisitos específicos que um jogo deve cumprir antes de ser publicado. Estes requisitos são documentos proprietários fornecidos aos desenvolvedores e editores sob acordos de confidencialidade. Cada lista de verificação de requisitos consiste em várias categorias, e o jogo deve obedecer a elas para evitar ser rejeitado pelo fabricante do console [URL1], [URL2].

Portanto, os testadores de videogames em consoles devem testar a conformidade com esses requisitos, além dos métodos padrão de teste de software.

Pode ser necessário utilizar equipamentos especiais para detectar, identificar e compreender as razões para a manifestação de defeitos. Este equipamento é essencialmente o próprio console, mas fornece modos adicionais que ajudam no desenvolvimento e teste dos jogos. O console é registrado no site para desenvolvedores e testadores autorizados, após o que ele é ativado e o acesso aos modos especiais é concedido.

1.1.4 Como os testes mitigam os riscos dos produtos de jogos

A maioria dos jogos, especialmente os jogos de mundo aberto, são tão complexos que é impossível testar todas as combinações possíveis de objetos, eventos e fatores do jogo. Mesmo a realização de testes individuais de cada combinação durante todo o ciclo de desenvolvimento do software pode ser demorada, e os testes de confirmação após as mudanças na próxima versão do jogo requerem um esforço adicional.

O gerenciamento de risco é usado para alocar recursos de forma eficiente e avaliar os possíveis defeitos com base em sua probabilidade de ocorrência e impacto sobre o jogo.

Para cada caso de teste, são feitas duas perguntas para determinar a prioridade do teste:

- Qual a probabilidade dos jogadores encontrarem um defeito aqui?
- Se eles encontrarem um defeito, como ele afetará os jogadores e a empresa dona do jogo?

Por exemplo, se um jogo tem uma estrutura linear, e o usuário acessa seu conteúdo passo a passo, então os defeitos que aparecem durante a primeira sessão de jogo muito provavelmente terão uma prioridade maior (mas nem sempre uma gravidade maior) para serem corrigidos do que os defeitos que ocorrem na quinquagésima hora de jogo. Se um usuário fiel já passou muito tempo no jogo, é mais provável que ele leve o defeito encontrado com mais calma, e isso não terá muito efeito na impressão geral do usuário sobre o jogo. É outra questão se tal defeito afetar todos os novos jogadores durante o treinamento introdutório. Neste caso, muitos jogadores podem recusar o produto do jogo.

Para reduzir os riscos através de testes, é importante identificar grupos de defeitos em vez de procurar por defeitos individuais no jogo. Isto permite aos desenvolvedores resolver simultaneamente um problema no código ou na arquitetura do jogo e proporcionar uma melhoria na qualidade do produto final.

Os grupos de defeitos são agrupamentos de defeitos em áreas críticas. Há áreas nos produtos do jogo que podem ter defeitos, tais como gráficos, som, regionalização, interação cliente-servidor e hardware [Nystrom14].

É importante ganhar a lealdade do jogador ao jogo quando ele está aprendendo a jogá-lo ou tomando um tutorial. Nesta fase, muitas funções do jogo estão disponíveis para o jogador para que ele possa avaliar o produto. Como resultado, esta fase inicial é geralmente declarada como sendo de importância crítica.

1.1.5 A diferença entre teste e "diversão".

Pode-se ter a impressão de que a essência do trabalho dos testadores de jogos de computador é jogar on-line com os colegas o dia todo. Mas esta afirmação está longe de ser verdade.

O testador raramente consegue apenas brincar, exceto ao conhecer um novo projeto ou durante os testes de jogo. A maior parte de seu trabalho está relacionada à realização de vários tipos de testes, que em muitos aspectos são comuns com os tipos de testes utilizados para outros softwares.

O usuário médio começa o jogo para passar, vencer adversários ou se divertir, enquanto o testador verifica se o jogo atende aos requisitos escritos na especificação. Além disso, as preferências pessoais do testador podem não ser mapeadas para o público-alvo pretendido do jogo em si. Entretanto, o testador passará pelo mesmo nível uma e outra vez para testar, por exemplo, se os objetos do jogo são exibidos corretamente. A monotonia experimentada pelo testador na execução dessas ações repetidas é compensada pela satisfação que eles têm em encontrar defeitos que poderiam perturbar a jogabilidade concebida pelo desenvolvedor.

Relação com outros syllabus

As abordagens discutidas neste syllabus se aplicam exclusivamente aos testes de jogo. Elas se aplicam tanto a testes de software (p. ex., testes de mecânica de jogos) quanto a testes de hardware (p. ex., testes de controladores).

Alguns aspectos de testar um jogo não são tão diferentes de testar outro software, em particular ao executar tipos de teste como teste de regressão, teste de performance ou teste de usabilidade. Conceitos fundamentais de teste, técnicas de teste não funcionais e características de teste de software em plataformas móveis são cobertos em detalhes em outros syllabus do ISTQB®.

1.2 Papéis típicos da equipe de desenvolvimento de jogos

Na maioria dos casos, uma equipe de desenvolvimento de jogos de computador inclui (mas não está limitada a) funções que não são fundamentalmente diferentes daquelas encontradas em outras áreas de desenvolvimento de software. Estes incluem proprietários de empresas, gerentes de projeto, analistas, desenvolvedores, projetistas de jogos e testes. Em equipes pequenas, uma pessoa pode assumir várias funções ao mesmo tempo. A seguir, um resumo das principais funções específicas que podem ser desempenhadas por funções típicas em um projeto de software para jogos. As funções adicionais que podem ser incluídas em uma equipe para áreas especiais de desenvolvimento de jogos são descritas nos capítulos apropriados deste Syllabus.

Papel	Responsabilidades
Proprietário do negócio	• Desenvolve os negócios; • Define a política de marketing; • Inicia novos produtos; • Pode participar da formação de equipes.
Gerente de projetos	• Define e monitora a execução das tarefas; • Acompanha as etapas do ciclo de vida do desenvolvimento de software.
Analista	• Identifica os requisitos do produto; • Trabalha com as especificações originais; • Atualiza as especificações, mantém-nas atualizadas.
Desenvolvedor	• Desenvolve e depura o código; • Corrige defeitos; • Desenvolve testes de componentes.
Projetista de jogos	• Desenha a mecânica de jogo, o equilíbrio, a trama, os níveis, a parte financeira do jogo; • Cria a documentação do jogo; • Pensa sobre e desenvolve o conteúdo do jogo.
Testador	• Desenvolve um modelo de teste e uma estratégia de teste; • Modela e prepara dados de teste; • Revisa as especificações; • Realiza vários tipos de testes; • Encontra e analisa defeitos.

1.3 Atividades de Teste durante todo o Ciclo de Vida do Desenvolvimento de Software de Jogos

Etapa conceitual

O principal objetivo desta etapa é alcançar uma visão comum do futuro produto entre a equipe de desenvolvimento e o cliente. As ideias são elaboradas na forma de documentos que descrevem brevemente a visão geral do jogo, seu gênero, cenário, a essência da jogabilidade, a mecânica e suas principais características.

Os artefatos que podem ser revistos nesta fase incluem:

- Documento de visão;
- Documento conceitual;
- Restrições técnicas e de produto.

O *testware* produzido nesta fase inclui:

- Estratégia de teste;
- Esboço do plano de teste;
- Estimativa de esforço de teste de alto nível;
- Ambiente de teste.

Fase de pré-produção

O principal objetivo desta etapa é criar um protótipo de um futuro jogo que é uma versão inicial de trabalho do produto. O protótipo implementa a principal jogabilidade em qualidade de rascunho. Nesta fase, a mecânica de jogo principal e as hipóteses iniciais são testadas (ver Capítulo 2), e as capacidades técnicas da equipe são avaliadas. Os testadores, projetistas de jogos e outros membros da equipe de desenvolvimento podem participar dos testes do protótipo.

A equipe de desenvolvimento revisa os requisitos preliminares para estabelecer a possibilidade de implementação e a equipe de teste os revisa para identificar a interação da mecânica entre si e a funcionalidade do protótipo montado. Além disso, os testadores estão envolvidos na criação do plano de teste e dos documentos do cronograma de testes.

À medida que a documentação para o jogo é criada, o plano de teste é escrito e atualizado, e a documentação é revisada. Estas medidas mitigam os riscos técnicos do produto na fase de pré-produção.

Os artefatos que podem ser revistos nesta fase incluem:

- Especificações de requisitos;
- Protótipo de jogo;
- Artefatos de teste criados na etapa anterior.

O *testware* produzido e atualizado nesta fase inclui:

- Estratégia de teste;
- Plano de teste;
- Cronograma de teste;
- Ambiente de teste.

Fase de produção

Durante a fase de produção, o produto do jogo é criado. Este é o estágio mais longo e normalmente é dividido em partes separadas para as versões Alfa e Beta do jogo.

Nesta etapa, os testadores finalizam listas de verificação e conjuntos de condições de teste e executam os testes funcionais e não funcionais dos componentes do produto implementado. A integração dos componentes, testes de sistema e aceite são realizados (ver [ISTQB_FL_SYL]). Esta etapa é onde é identificado o maior número de defeitos, o que resulta em testadores realizando testes de confirmação e testes de regressão.

O *testware* produzido ou atualizado nesta fase inclui:

- Plano de teste;
- Cronograma de testes;
- Casos de teste, condições de teste, listas de verificação, conjuntos de teste, scripts;
- Relatórios de defeitos;
- Relatórios de teste.

Fase de pós-produção

A fase de pós-produção para testes de jogos inclui atividades de testes comuns durante o período de manutenção do software. Isso normalmente envolve testes de atualizações e testes de regressão, como descrito em [ISTQB_FL_SYL].

2 Testando a mecânica de jogo - 180 min

Palavras-chave

teste ad hoc, teste funcional

Palavras-chave específicas

mecânica do cliente, estágio de conceito, mecânica central, desenvolvedor de jogos, mecânica dos jogos, metamecânica, fase de pré-produção, fase de produção, mecânica do servidor, estado dos jogos

Objetivos de aprendizagem

2.1 Mecânica de jogo

GaMe-2.1.1 (K2) Classificar os tipos de mecânica de jogo.

GaMe-2.1.2 (K2) Diferenciar os testes de mecânica de jogo e mecânica de não-jogo.

GaMe-2.1.3 (K2) Diferenciar os testes de mecânica central e metamecânica.

GaMe-2.1.4 (K2) Diferenciar os testes de mecânica do cliente, do servidor e do cliente-servidor.

GaMe-2.1.5 (K2) Dar exemplos de defeitos na mecânica de jogo.

2.2 Abordagens para testar a mecânica de jogo

GaMe-2.2.1 (K2) Resumir as principais abordagens e testar objetos em diferentes estágios da criação de um produto de jogo.

GaMe-2.2.2 (K2) Distinguir a importância de testar a mecânica de jogo.

GaMe-2.2.3 (K2) Distinguir a importância de revisar a documentação que descreve a mecânica dos jogos.

GaMe-2.2.4 (K3) Aplicar as abordagens fundamentais de testar a mecânica dos jogos.

2.1 Mecânica dos jogos

2.1.1 Tipos de mecânica dos jogos

O capítulo 1 mencionou que para os jogos, uma das coisas mais importantes é criar interesse do usuário a fim de reter e aumentar a audiência. Os jogadores avaliam os gráficos do jogo, a jogabilidade em si e a mecânica que compõe a jogabilidade.

A mecânica dos jogos está no coração de qualquer jogo. A mecânica dos jogos é um método de interação entre o jogo e o usuário. Ela considera o impacto e o feedback entre o jogo e o usuário e as mudanças no estado do jogo dentro dos requisitos especificados. A mecânica pode ser dividida em diferentes tipos, dependendo de diferentes aspectos e objetivos do jogo. Dependendo do tipo de interação do jogador com a mecânica:

- A **mecânica de jogo** se refere à quando o usuário interage conscientemente com o sistema de jogo, baseando suas ações na disponibilidade de informações. Eles podem influenciar a mudança no estado do jogo, ao mesmo tempo em que veem claramente o resultado de suas ações;
- A **mecânica de não-jogo** se refere à quando o usuário não pode ou só pode influenciar parcialmente o estado do ambiente do jogo através de suas ações. Eles não reconhecem sua influência, porque parte da mecânica de jogo está escondida deles.

Dependendo do impacto sobre a jogabilidade principal:

- A **mecânica central** forma a base do jogo e define as ações que o usuário repete ao longo do jogo. Elas visam garantir que o usuário receba uma certa experiência que foi embutida no jogo por seus criadores;
- A **metamecânica** está fora da jogabilidade principal, mas pode influenciá-la. Ela visa fazer com que o jogador faça o que o projetista do jogo quer (p. ex., voltar ao jogo, continuar jogando, ou fazer compras). Esta mecânica pode ser combinada com a mecânica principal para tornar o jogo mais interessante.

Dependendo da arquitetura da estrutura do jogo:

- A **mecânica do cliente** relacionada ao processamento das ações do usuário ocorre exclusivamente no dispositivo do usuário;
- A **mecânica do servidor** está relacionada à mecânica que é processada somente no servidor do jogo;
- A **mecânica do cliente-servidor** está relacionada com a mecânica que é processada tanto no dispositivo do usuário quanto no servidor. Neste caso, há uma constante troca de dados entre o servidor e o cliente.

2.1.2 Diferença entre testar a mecânica de jogo e testar a mecânica de não-jogo

O usuário não interage diretamente com toda a mecânica implementada no jogo.

A **mecânica de jogo** pode ser descrita em uma especificação formal ou informal, mas a mecânica de não-jogo deve ser descrita para o usuário. Muitas vezes, elas afetam diretamente a realização do objetivo da jogabilidade. Por exemplo, a coleta de moedas por um personagem do jogo deve ser realizada dentro de um tempo limitado. A funcionalidade do movimento de apanhar as moedas e aumentar o número de moedas é um exemplo da mecânica de jogo.

A **mecânica de não-jogo** é geralmente escondida do usuário, embora possam ser acionadas pela ação do próprio usuário. Por exemplo, quando o usuário faz uma compra de qualquer produto dentro de um jogo, um fluxo sequencial de mecânica de não-jogo é iniciado. Por exemplo:

1. É feita uma solicitação ao banco de dados para obter o estado da conta do usuário no jogo;
2. É feita uma transação para retirar fundos;
3. A compra é registrada, ou seja, as informações sobre a ação executada são registradas em um arquivo de texto especial;
4. Um e-mail com os detalhes da compra é enviado para o usuário.

Além disso, a data da compra e o tipo de produto comprado podem ser registrados, de modo que um envio automático sobre um desconto para um produto similar será enviado. Mecanismos que não sejam de jogo devem ser especificados como requisitos formais, pois não são evidentes.

O teste da jogabilidade e da mecânica de não-jogo é um teste funcional, como descrito em [ISTQB_FL_SYL].

2.1.3 Diferença entre os testes da mecânica central e da metamecânica

A diferença nas abordagens para testar a mecânica central e a metamecânica depende de seu impacto.

A remoção de uma **mecânica central** do jogo muda completamente a jogabilidade. Por exemplo, se o movimento do carro for removido de um simulador de corrida, o jogo não pode mais ser classificado como "corrida".

Exemplos de **metamecânica** seriam escolher um modelo de carro com parâmetros específicos, dependendo do tipo de pista de corrida, ou comprar e equipar um personagem-jogador com uma nova armadura antes de entrar em uma batalha. A remoção de uma metamecânica de jogo raramente muda a essência do jogo inteiro.

Além dos requisitos funcionais, a mecânica central e a metamecânica podem ter outros requisitos não-funcionais que afetam diretamente como eles devem ser testados.

Por exemplo, para uma metamecânica mudar a aparência de um personagem jogável, o mais importante é que ela funcione corretamente. Mesmo que este mecanismo funcione com um leve atraso, de modo que a aparência mude alguns segundos após o jogador apertar o botão, nem o testador nem o usuário considerarão isto um defeito. Entretanto, para a mecânica central de um personagem saltando ou usando habilidade em batalha, tal atraso seria crítico, pois poderia impedir o jogador de ganhar. Como resultado, o teste de performance também é importante para a mecânica central.

A essência da mecânica central e da metamecânica também afeta quando eles são testados. Como regra, sua verificação geralmente começa em diferentes estágios de desenvolvimento do jogo. A mecânica central, como o movimento de um personagem de jogo, o recebimento de uma recompensa etc., já aparecem nas primeiras versões do produto. Ao mesmo tempo, começa a verificação da exatidão funcional e do nível de interesse do público-alvo.

A metamecânica é normalmente acrescentada ao jogo e começa a ser testada mais tarde, durante a fase de produção. A avaliação de seu impacto sobre a duração média de uma sessão de jogo, o número de amigos convidados, o tamanho do pagamento médio, os produtos mais populares do jogo etc., ocorre na fase de beta teste. Ao mesmo tempo, os testes A/B são frequentemente usados para determinar a mecânica mais eficaz.

2.1.4 Diferença entre os testes de mecânica do cliente, do servidor e do cliente-servidor

A necessidade de testar a mecânica do cliente, do servidor e do cliente-servidor depende da arquitetura do próprio jogo. Cada um desses tipos de mecânica requer uma abordagem diferente para os testes, dependendo das especificidades de sua implementação.

A **mecânica do cliente** é processada nos dispositivos dos próprios jogadores, para que o usuário possa acessar todos os dados armazenados no cliente. Alguns jogos podem ser totalmente executados no cliente e não utilizam o servidor, ou seja, todos os mecanismos são do lado do cliente. Estes jogos são normalmente de um único jogador e não requerem conexão à Internet.

O cliente do jogo contém o código que é executado no dispositivo do jogador e todos os parâmetros, fórmulas de cálculo de danos, níveis de jogo, modelos de personagens etc. Como resultado, o jogador, se desejar, pode alterar todos os dados no cliente do jogo. Modificando o cliente, o usuário pode, por exemplo, aumentar a velocidade de sua técnica, a quantidade de saúde do personagem, o valor da recompensa por completar a tarefa etc.

Tais mudanças de cliente podem dar ao jogador uma vantagem no jogo, mas em jogos de um jogador, isso geralmente não é considerado um defeito grave. Por exemplo, em um jogo de um jogador, a mecânica da luta de espadas com oponentes de computador pode ser implementada. Neste caso, os parâmetros da arma não serão validados no servidor. Portanto, se um jogador específico fizer uma mudança no cliente do jogo e aumentar o dano de sua espada, isto só mudará sua jogabilidade pessoal. Isto não afetará os interesses de outros jogadores independentes que jogam o mesmo jogo em seus dispositivos.

A mecânica do cliente também pode estar presente em jogos *multiplayer*, mas ainda assim não afetará outros jogadores. Por exemplo, a mecânica de visualizar um mapa de nível, abrir um inventário de personagens etc., são processados somente do lado do cliente e verificados por testes funcionais sem o envolvimento do servidor.

Os testes da mecânica do cliente são geralmente realizados utilizando testes caixa-preta. Ao fazer isto, o testador poderá verificar a interface do usuário da aplicação. Neste caso, a própria interface do usuário pode ser usada ao testar o jogo para obter resultados reais.

A **mecânica do servidor** é implementada exclusivamente no lado do servidor. Devido a esta característica, a lógica da mecânica é protegida da intervenção do usuário. Os jogadores não podem influenciar diretamente os processos que rodam no servidor remoto, portanto, os desenvolvedores transferem as partes mais importantes dos cálculos da lógica do jogo para lá.

Testar a mecânica do servidor é o mais importante para jogos *online multiplayer*. Por exemplo, quando um jogador inicia uma competição em equipe no modo jogador *versus* jogador (PvP), uma seleção de aliados e oponentes é conduzida no servidor com classificações de jogo e níveis de personagens aproximadamente semelhantes. Caso contrário, a competição será desinteressante e desconfortável para todos os participantes.

A abordagem para testar a funcionalidade implementada no servidor é diferente de testar a mecânica do cliente. Aqui, o testador geralmente não precisa de uma interface de usuário. Os testes são realizados em consoles de servidor ou usando ferramentas especiais. O testador gasta a maior parte de seu tempo analisando os logs ou interagindo com o banco de dados.

Para a mecânica do servidor, os tipos de testes mais importantes são testes funcionais, testes de performance e testes de segurança.

A **mecânica do cliente-servidor** envolve tanto o lado cliente quanto o lado servidor e combina as características dos dois tipos de mecânica mencionados anteriormente. Uma parte da lógica é processada nos dispositivos dos jogadores e a outra no servidor remoto. Partes de tal mecânica "comunicam-se" constantemente umas com as outras, enviando dados entre si. Em um jogo *multiplayer*, o servidor recebe dados do cliente, e então envia o resultado das ações para todos os jogadores do servidor que são afetados por estas ações.

Os dados que chegam ao servidor passam por uma validação obrigatória. Por exemplo, se um usuário quiser comprar algo na loja do jogo, então o servidor deve verificar se há uma quantidade suficiente de moedas disponíveis no jogo para uma compra, dependendo do progresso do jogador. Mesmo se o jogador tentar modificar a parte do cliente em seu dispositivo e enviar parâmetros falsos ao servidor a fim de obter uma vantagem no jogo, o servidor irá substituí-los por dados corretos.

Quando os testadores testam a mecânica do cliente-servidor, eles criam um ou mais casos de teste de modo a cobrir o número máximo de diferentes mecânicas.

Um exemplo de tal teste seria um cliente fazendo login em um servidor de jogos e matando um oponente controlado por computador. Apesar do pequeno número de ações necessárias que compõem este teste, quase toda a mecânica de interação cliente-servidor é usada aqui:

- Enviar uma solicitação do cliente ao servidor responsável pela autorização;
- Fazer uma solicitação do servidor de autorização para o banco de dados, executar a autorização e restaurar o estado anterior do jogo;
- Seleciona mapas já carregados no servidor ou criar uma mecânica de jogo no servidor;
- Conectar o cliente ao servidor de mecânica de jogos onde o mapa está localizado;
- Dar uma visão do personagem do jogo e objetos do jogo nas proximidades do personagem enquanto o servidor cria objetos e dá ao cliente o comando para exibi-los ao usuário;
- Mover o personagem pelo mapa enquanto exibe novos objetos de jogo;
- Encontrar um inimigo do computador e seu ataque e exibi-lo ao cliente;
- Receber o comando de "ataque" do servidor;
- Verificar se um ataque é possível, assegurando que o personagem e o objeto móvel (Mob) estejam a uma distância aceitável para um ataque, e que não haja objetos entre eles que impeçam que o ataque seja realizado;
- Quando um ataque é possível, enviar uma mensagem do servidor para o Mob sobre o recebimento de danos;
- Calcular a quantidade de danos recebidos pelo Mob no servidor, reduzindo o valor do parâmetro responsável pela saúde do personagem;
- Exibir os danos causados ao jogador pelo Mob.

2.1.5 Exemplos de defeitos na mecânica de jogos e possíveis causas de sua ocorrência

A mecânica de jogo envolve influências sobre objetos de jogo e o feedback sinalizando o resultado dessas influências. Em conjunto, isto cria o caráter único do jogo, com uma dinâmica de jogo adequada. Na maioria das vezes, os jogos utilizam uma ampla gama de influências e elementos de feedback.

Os seguintes tipos de defeitos estão associados à mecânica de jogo:

- Defeitos funcionais na mecânica;
- Defeitos de reconhecimento da adequação do jogo;
- Defeitos de eficácia da mecânica manifestados em um ambiente de jogo específico.

Os defeitos funcionais do jogo são mais mencionados quando os jogadores falam sobre defeitos na mecânica de jogo. Por exemplo, as armas não recarregam, o saque não é coletado, a imagem não se aproxima quando se usa binóculos. Tais defeitos geralmente surgem de erros cometidos pelo desenvolvedor no código do jogo. Eles são bastante perceptíveis e são bastante fáceis de detectar e corrigir.

Os defeitos de reconhecimento da adequação do jogo estão relacionados à obtenção de respostas do jogo enquanto se utiliza um mecanismo particular. A situação é incorreta quando o jogador não entende por que o jogo acabou. Portanto, a mecânica de jogo é frequentemente acompanhada por efeitos visuais e sonoros. Este pode ser o efeito visual de uma explosão, o som de um contador de tempo, ou mesmo de uma mensagem de texto. A própria presença de uma resposta quando se usa a mecânica é testada e a justificativa de sua presença ou ausência é avaliada. A operação e correção do efeito é testada por outros tipos de testes (testes gráficos ou sonoros).

O terceiro tipo de defeito está relacionado com a eficácia e o escopo da mecânica. A mecânica pode ser eficaz por si só, mas pode deixar de funcionar dentro da jogabilidade. Para evitar tal situação, a mecânica é testada em conjunto com outros mecanismos, objetos e outros conteúdos necessários no nível do jogo. Portanto, os testes de integração são realizados junto com testes de usabilidade para verificar a eficácia da mecânica na jogabilidade.

Por exemplo, um projetista de jogos pode adicionar dois oponentes ao jogo com comportamentos e características diferentes. O primeiro é rápido e saltitante, mas lida com uma pequena quantidade de danos, e o segundo é lento, desajeitado, mas ataca com precisão e força. Em termos de características, ambos os oponentes parecem iguais. Eles têm suas próprias vantagens e desvantagens. Cada um pode ter seu próprio algoritmo de comportamento e, em teoria, eles devem representar aproximadamente a mesma quantidade de perigo para o jogador. Entretanto, se uma grande parte do nível do jogo compreende várias colinas que o personagem do jogador pode escalar, a situação muda. Estando em uma altura inacessível a um inimigo lento, o jogador pode atacar com segurança o adversário. Neste caso, o usuário terá dificuldade para enfrentar outro inimigo, que também pode usar colinas, mas se move em alta velocidade e salta alto. A solução para este problema pode ser tanto uma mudança na própria mecânica, quanto uma mudança na regionalização dos objetos no nível.

A complexidade de tais testes está associada principalmente à dificuldade de prever tais situações e como a mecânica funcionará nelas. Portanto, a busca de problemas de interação entre mecânica e objetos do ambiente é frequentemente realizada como parte de testes ad hoc na fase de testes beta com um grande grupo de jogadores.

2.2 Abordagens para testar a mecânica de jogo

2.2.1 Procedimentos e abordagens para testar a mecânica de jogo no Ciclo de Vida de Desenvolvimento de Software de jogo

Em diferentes estágios do desenvolvimento de software, os testes de mecânica em jogos podem ser realizados utilizando várias abordagens de testes.

Na fase de criação de um protótipo de jogo, apenas a mecânica central é implementada. Aqui as principais tarefas do testador são testar a mecânica de jogo contra as seguintes características:

- Correção funcional;
- O nível de interesse para os jogadores;
- A atratividade para os jogadores.

Todos os mecanismos e suas regras de operação são revisados para garantir que sejam descritos de forma completa e inequívoca no documento de projeto do jogo. Todas as opções permitidas para a interação do usuário com a mecânica devem ser trabalhadas, e o impacto de cada mecânica sobre toda a jogabilidade deve ser determinado.

Durante a fase de produção, as mecânicas implementadas são submetidas a testes funcionais para verificar se atendem aos requisitos declarados. Como regra, os testes aqui acontecem utilizando conjuntos de condições de

teste e listas de verificação. A sequência de testes deve visar a cobertura máxima de todos os aspectos da mecânica.

Os testes exploratórios e os testes ad hoc são mais eficazes para testar a interação de diferentes mecânicas entre si. Isto é especialmente evidente em jogos multiplayer de larga escala, onde o número de formas de interação entre a mecânica e vários componentes do jogo se torna grande demais para ser refletido nas condições de teste.

Como parte dos testes não funcionais, o efeito da mecânica sobre a performance geral do jogo é testado. A utilização de recursos (p. ex., RAM, CPU etc.) e o resultado real (p. ex., ocorrência de congelamentos no cliente do jogo, diminuição significativa na taxa de quadros etc.) são testados. A compatibilidade de vários componentes e softwares deve ser testada, especialmente com programas antivírus.

Se o jogo exigir a presença de um servidor, a performance do servidor também é testada. O teste de performance é planejado e executado durante a fase de produção, quando a mecânica planejada que poderia afetar a performance está sendo implementada. Detalhes do planejamento e execução dos testes de performance estão descritos em [ISTQB_FL_PT].

É dada uma atenção considerável à metamecânica ao realizar testes beta com muitos jogadores. A conveniência e a clareza do jogo são avaliadas para os jogadores. Assim como nos testes de software que não fazem parte do jogo, o principal objetivo deste teste não é encontrar defeitos funcionais óbvios, que idealmente deveriam ser identificados nos estágios anteriores, mas obter feedback dos jogadores finais. Entretanto, os jogadores finais também podem participar e fornecer feedback em testes alfa anteriores, por exemplo, em grandes jogos multiplayer.

Após o jogo ser lançado no mercado, a tarefa dos testadores é lidar e verificar se há defeitos na mecânica relatados pelos jogadores e testar qualquer nova mecânica que seja adicionada ao jogo.

2.2.2 A importância de testar a mecânica de jogo

É a mecânica de jogo que é a espinha dorsal do jogo. Ela forma a base para a experiência do jogador, que é necessária para mantê-lo interessado. Como a mecânica de jogo é a essência de qualquer jogo, testar seu funcionamento correto é da maior prioridade para um testador, e os defeitos encontrados durante tais testes são geralmente os mais críticos.

Um defeito na mecânica central, como o salto inoperante de um personagem, pode levar à incapacidade de superar um certo obstáculo e, como resultado, de completar o jogo.

Mas mesmo que a mecânica seja raramente encontrada no jogo ou esteja associada a atividades desnecessárias para avançar no jogo, (p. ex., a capacidade do personagem de montar um cavalo), seu funcionamento incorreto afetará negativamente a percepção do jogo como um todo.

2.2.3 A Importância da Revisão da Mecânica dos Jogos

Uma etapa importante na criação de um jogo é o desenvolvimento da documentação do jogo, que, entre outras coisas, descreve a mecânica de jogo.

A revisão da especificação da mecânica de jogo no início da fase de desenvolvimento ajuda a evitar muitos problemas posteriormente. Estes problemas podem estar associados ao próprio processo de desenvolvimento, bem como à reação dos jogadores à mecânica de jogo implementada no produto final.

Durante a revisão, o testador deve prestar atenção às seguintes características de qualidade [ISO 25010]:

- Completude da descrição;
- Conformidade com a realidade;
- Estrutura e facilidade de navegação na documentação.

Problemas no processo de desenvolvimento do jogo que uma revisão pode evitar:

- Compreensão incorreta da essência da mecânica pelo desenvolvedor e avaliação incorreta do esforço necessário, devido a uma descrição insuficientemente clara, detalhada e sem ambiguidade da mecânica;
- A incompatibilidade de novos mecanismos com os existentes;
- A incapacidade de implementar a mecânica em um determinado projeto.

Problemas relacionados à percepção do jogador sobre a mecânica que uma revisão pode evitar:

- Inadequação geral da mecânica em um determinado jogo;
- Mecânica desinteressante;
- Mecânica desequilibrada;
- Uso muito raro da mecânica no jogo.

2.2.4 Testando o estado do jogo após o reinício da sessão e quando o usuário está inativo

Estado dos jogos

O estado do jogo é entendido como o valor de todos os parâmetros e variáveis que descrevem todos os objetos do jogo em um determinado momento no tempo.

Quanto mais complexo for o jogo e quanto mais possibilidades de ações ele proporcionar ao jogador, mais parâmetros cada objeto terá.

Todos os parâmetros podem ser condicionalmente divididos em visíveis e ocultos.

As informações sobre os valores dos parâmetros visíveis estão explicitamente disponíveis para o jogador. Isto pode incluir:

- O nível de experiência do usuário dentro do jogo para a tarefa concluída;
- As características da arma selecionada;
- O número de itens permitidos no inventário;
- O custo das mercadorias na loja dentro do jogo.

Parâmetros ocultos podem ser usados pelos desenvolvedores para influenciar o curso do jogo sem envolver a consciência do jogador. Por exemplo, além dos indicadores de Ataque e Defesa conhecidos pelo jogador, o personagem pode ter um parâmetro de precisão, que é escondido do jogador e determina a probabilidade de um tiro acertando o alvo. Muitas vezes o jogador desconhece a existência de tais parâmetros e o que exatamente eles afetam.

Os parâmetros ocultos podem ser responsáveis por diferentes aspectos do jogo:

- Taxa de queda, ou seja, a chance de obter um item aleatório como recompensa;
- A taxa de incêndio ou a velocidade de movimento do equipamento em vários tipos de superfícies.

A influência da escolha do jogador no desenvolvimento da trama do jogo. Um conjunto de parâmetros ocultos e visíveis definem de forma única o caráter do jogo controlado pelo jogador e outros objetos.

Objetivos de testar o estado dos jogos

Um ponto importante ao testar os parâmetros do jogo é testar se o jogo realmente usa os valores reais dos parâmetros.

No caso de parâmetros visíveis, os testes são realizados comparando visualmente o valor exibido com o valor efetivamente utilizado. Por exemplo, se um personagem de jogo com um parâmetro de ataque igual a 1 infligiu 10 unidades de dano ao inimigo, então depois que o jogador aumentou o parâmetro de ataque para 2, a quantidade de dano feito ao mesmo inimigo também deve aumentar e tornar-se, digamos, 20 unidades.

Tal teste é especialmente importante se a informação sobre a nova quantidade de dano estiver disponível para o jogador, na forma, por exemplo, de uma dica de jogo: como "*Cada unidade de um parâmetro de ataque acrescenta dez unidades de dano*". Neste caso, o jogador também será capaz de comparar o valor indicado na dica com o dano real.

Um jogador regular normalmente se concentra apenas na adequação e consistência da mudança do dano, sem considerar valores específicos dos parâmetros. Tal jogador pode não considerar como um defeito infligir o número errado de pontos de dano, mas se o dano não mudar ou diminuir, isto certamente será notado.

Um testador profissional é guiado pelas fórmulas e princípios para o cálculo da alteração dos parâmetros especificados na documentação. Se esta informação não puder ser obtida da documentação, o testador, como o jogador, a avalia relativamente, com base no senso comum.

Para testar parâmetros ocultos, o testador pode ocasionalmente acessar o banco de dados do jogo. Com sua ajuda, o testador pode comparar os valores reais dos parâmetros com os valores esperados.

Além disso, se for assumido que o usuário pode alterar os parâmetros visíveis de seu personagem durante o jogo, então a própria possibilidade de alterar os parâmetros e a correção de sua exibição é testada.

Testar o estado do jogo está intimamente relacionado a salvar e carregar um jogo.

Salvamento e Carregamento

Muitos jogos são impossíveis de serem concluídos em uma sessão, ou podem, em princípio, envolver uma jogabilidade sem fim. Os jogos, portanto, muitas vezes usam funções de estado "salvar e carregar" para que o jogador não tenha que começar sempre desde o início.

Outras razões para usar a função salvar são para encorajar o usuário a explorar o mundo do jogo, encontrar diferentes maneiras de resolver os problemas oferecidos pelo jogo e até mesmo gerenciar a complexidade do jogo. Se o jogador tiver a oportunidade de sobreviver diante de um obstáculo difícil, o custo de seu erro será menor do que se uma ação errada anular completamente o progresso.

Salvar refere-se às informações sobre o progresso do jogo que são armazenadas no disco rígido ou na nuvem. A gravação pode ser apresentada como um arquivo que contém informações sobre o estado atual do jogo. O tamanho de tal arquivo é várias vezes menor do que o próprio jogo e geralmente contém as seguintes informações:

- Uma lista de objetos para os quais as informações de estado precisam ser armazenadas;
- Uma lista de parâmetros de mudança para cada objeto;
- Um código específico para cada parâmetro, pelo qual o jogo pode determinar o que está acontecendo com ele no momento.

Quase qualquer variável pode atuar como um objeto. Seu estado é operado pelo sistema de jogo e precisa ser lembrado. Os objetos variáveis incluem, por exemplo, o nível de saúde do personagem do jogo, o número de habilidades aprendidas, e as coordenadas do personagem no mapa do jogo.

Tipos de salvamento e áreas de teste

O método de gravação, o formato de arquivo de gravação e a lista de parâmetros armazenados podem variar de jogo para jogo. Ao longo da história dos jogos, muitas abordagens têm sido inventadas para salvar o progresso. As mais comumente usadas são descritas abaixo.

Utilização de pontos de verificação (checkpoints) para salvar

O salvamento ocorre automaticamente nos momentos especificados pelos desenvolvedores. Atingir cada ponto de verificação seguinte sobrescreve as informações sobre o progresso atual do jogador. O estado do jogo salvo nos pontos de verificação é armazenado apenas para a sessão atual do jogo e é apagado quando o jogo termina.

Às vezes o jogo não mostra explicitamente que o usuário chegou ao ponto de verificação, e o progresso é salvo. A fim de testar a correção de salvar e carregar, o testador precisa de documentação que forneça uma lista da regionalização exata de todos os pontos de checagem existentes. Ao utilizar outros métodos de salvamento, esta informação na documentação pode estar ausente ou ser declarada com menos detalhes.

Usando salvamentos estáticos

Em certos lugares do jogo, os desenvolvedores colocam pontos especiais nos quais o jogador, se desejar, pode salvar manualmente seu progresso atual. Estes pontos são normalmente cercados por alguns poucos objetos de jogos, o que evita a criação de grandes arquivos de salvamento.

A área de verificação neste método é a capacidade de salvar em cada ponto, e a capacidade de salvar e carregar repetidamente cada estado salvo.

Salvamento automática

Esta é uma combinação de pontos de controle e salvamento estático. O progresso do jogador é salvo automaticamente durante toda a jogada em determinados pontos. Por exemplo, muitos jogos têm um recurso de auto salvamento na saída. Ao iniciar um jogo, o usuário pode iniciar um novo jogo ou carregar um jogo salvo anteriormente.

A tarefa do testador, neste caso, é verificar se houve salvamento e a possibilidade de carregar o estado alcançado do jogo em todos os pontos.

Salvamento manual ou gratuito

O usuário salva a qualquer momento durante o jogo usando um item especial no menu do jogo. O salvamento rápido pode ser implementado quando o salvamento e o carregamento ocorrerem ao se pressionar uma tecla ou através de um clique.

Como no caso do salvamento automático, o salvamento manual cria um arquivo especial que contém informações sobre o estado do jogo. O teste neste caso inclui a verificação da correção do carregamento das informações do arquivo, o nome correto do arquivo, a regionalização do arquivo na estrutura do sistema operacional, a operação correta na nova versão do jogo e até mesmo a possibilidade de usar o arquivo salvo, por exemplo, em outro computador.

É suficiente testar o carregamento correto do nível de jogo exigido para jogos que se encaixem na descrição a seguir:

- O jogo assume o único caminho e forma correta de passar (p. ex., apenas uma forma correta para sair de um labirinto);
- O personagem do jogo controlado pelo jogador não muda de nível para nível (p. ex., não aumenta a força de impacto, alcance do salto, velocidade de movimento através de um nível, e outras características que poderiam aumentar a velocidade de passar de nível).

Nos jogos que se encaixam na descrição a seguir, os salvamentos contêm informações mais exclusivas sobre a sessão de jogo, nível, personagem etc.:

- O personagem do jogo pode ter muitos parâmetros (p. ex., velocidade de movimento, alcance ou altura do salto, força de impacto, inventário, aparência do modelo 3D e outras características);
- Os parâmetros descritos acima podem mudar dependendo das ações e decisões do jogador dentro do jogo (p. ex., o jogador encontra o artefato em um labirinto e a velocidade do movimento é aumentada).

Para tais jogos, os testadores precisam testar a exatidão do carregamento do nível requerido do jogo e informações individuais, mas também as informações únicas de cada jogador e cada personagem do jogo. Ao testar o salvamento, o testador precisará testar o seguinte:

- Se o jogador está no nível de jogo em que terminou a sessão de jogo anterior;
- Se o personagem possui um artefato;
- Se o parâmetro de velocidade do personagem é aumentado devido à existência de um artefato.

3 Testes gráficos - 165 min

Palavras-chave

teste do jogo

Palavras-chave específicas

modelo 3D, animação, colisões, nível de jogo, caixa de batida, nível de detalhe (LoD), mapeamento, *rigging*, iluminação de cena, *skinning*, texturas, efeito visual (VFX)

Objetivos de aprendizagem

3.1 Princípios e conceitos de gráficos de jogos

GaMe-3.1.1 (K2) Explicar as características do conteúdo gráfico de um produto de jogo.

GaMe-3.1.2 (K2) Classificar os tipos de defeitos no conteúdo gráfico.

3.2 Abordagens para testar gráficos em jogos

GaMe-3.2.1 (K2) Resumir as principais abordagens dos testes artísticos.

GaMe-3.2.2 (K2) Resumir as principais abordagens aos testes técnicos.

GaMe-3.2.3 (K2) Resumir as principais abordagens para testes de jogabilidade.

3.3 Execução de testes gráficos

GaMe-3.3.1 (K3) Aplicar as abordagens fundamentais dos testes gráficos.

GaMe-3.3.2 (K2) Explicar a importância de testar gráficos para a validade histórica.

3.4 Suporte de ferramentas para testes gráficos

GaMe-3.4.1 (K2) Resumir o uso de ferramentas de teste gráfico

3.1 Princípios e conceitos de gráficos de jogos

3.1.1 Características do conteúdo gráfico do produto do jogo

Qualquer produto de jogo, seja um jogo para celular ou um grande jogo de alto orçamento, há muitos diferentes elementos gráficos que juntos permitem ao usuário visualizar o processo do jogo.

Diferentes especialistas desempenham certas funções no processo de criação de objetos gráficos para os jogos:

Papel	Responsabilidades
Artista:	Cria vários conteúdos gráficos para o jogo
Modelador 3D:	Cria modelos 3D dos objetos do jogo
Artista texturizado:	Cria texturas para objetos de jogo
Especialista em animação:	Trabalha na animação de objetos de jogo
Artista técnico:	Realiza testes técnicos
Testador:	Testa objetos gráficos no mecanismo do jogo.

Um testador de um produto de jogo, independentemente do orçamento do jogo e da quantidade de conteúdo gráfico no mesmo, precisa ter em mente, ao testar, as várias áreas que estão parcialmente, completamente ou separadamente presentes em cada jogo.

Considere as principais áreas do produto de jogo para testes.

Níveis (mapas de jogo)

Os níveis são uma área separada do mundo virtual do jogo, que geralmente representa um local específico, por exemplo, um edifício ou uma cidade. O termo veio de jogos de RPG, onde se referia aos níveis de uma masmorra (ou seja, o ambiente em que a maior parte do jogo acontecia). Os jogadores começaram nas profundezas da masmorra (nível 1) e tem que chegar à superfície passando por todos os demais níveis, o que se tornou cada vez mais difícil até chegar ao último (p. ex., nível 100), ganhando assim o jogo. Agora os níveis de uma forma ou de outra estão presentes em quase todos os jogos.

Modelos

Um modelo é qualquer objeto em computação gráfica. De acordo com os métodos de criação de imagens, os gráficos podem ser divididos nas seguintes categorias:

- Gráficos 2D;
- Gráficos 3D;
- Gráficos de imagens geradas por computador (CGI).

Gráficos 2D

Os gráficos de computador 2D são classificados de acordo com o tipo de apresentação das informações gráficas e algoritmos de processamento de imagem que se seguem. Normalmente, a computação gráfica 2D é dividida em vetores e rastros, embora exista também um tipo fractal de representação de imagem.

Gráficos 3D

Os gráficos 3D operam com objetos no espaço 3D. A computação gráfica 3D é amplamente utilizada em filmes e jogos de computador.

Os gráficos 3D podem ser:

- *Poligonal*;
- *Voxels*.

Os gráficos poligonais são uma coleção de vértices, bordas e faces que definem a forma de um objeto poliédrico em computação gráfica 3D e modelagem volumétrica. As faces são geralmente triângulos, quadriláteros ou outros polígonos convexos simples, pois isso simplifica a renderização.

Um *voxel* é um elemento virtual que corresponde a um conjunto de seis polígonos retangulares. Tudo no mundo virtual (ou seja, pixels, polígonos e *voxels*) deve ser projetado sobre os pixels da tela física. Os gráficos *voxel* são similares aos gráficos *raster*. O objeto consiste em um conjunto de formas 3D, na maioria das vezes em cubos.

Imagens gráficas geradas por computador (CGI)

CGI são imagens 3D obtidas por computador com base em cálculos e utilizadas em artes visuais, impressão, efeitos especiais cinematográficos, na televisão e em simuladores. As imagens em movimento são criadas por animação computadorizada, que é uma área mais estreita dos gráficos de CGI.

Qualquer imagem no monitor, devido à sua planicidade, torna-se um *raster*, uma vez que o monitor é uma matriz composta de colunas e linhas. Os gráficos 3D existem apenas em nossa imaginação, já que o que um humano vê no monitor é uma projeção de uma figura 3D, e o espectador cria o espaço por si só. Assim, a visualização dos gráficos só pode ser *raster* e vetorial, e o método de renderização é apenas um *raster* (ou seja, um conjunto de pixels). A maneira de definir a imagem depende do número desses pixels.

Texturas

A textura é um bitmap aplicado à superfície de um modelo poligonal para dar-lhe cor ou ilusão de relevo. Em geral, as texturas podem ser pensadas como um padrão na superfície de uma imagem escultural. O uso de texturas o faz possível reproduzir pequenos objetos de superfície que exigiriam recursos excessivos para serem criados com polígonos. Por exemplo, cicatrizes na pele, dobras na roupa, pequenas pedras e outros objetos na superfície das paredes e do solo.

A qualidade de uma superfície texturizada é determinada por *texels*, que representam o número de pixels por unidade de textura mínima.

O principal aspecto no teste de textura é verificar a presença de texturas em objetos visíveis, para uma exibição correta e uniforme. Se as texturas do jogo estiverem em alta resolução, não deve haver texturas de baixa qualidade.

Colisões

A colisão é um atributo responsável pela passagem de objetos colidindo uns com os outros.

Jogos de computador, especialmente jogos de console, devem distribuir muitas de suas tarefas entre recursos limitados de hardware e tempo de jogo limitado. Apesar destas limitações e do uso de algoritmos relativamente primitivos e imprecisos de detecção de colisão, os desenvolvedores de jogos têm sido capazes de criar formas visualmente críveis e relativamente realistas de exibir as interações entre os objetos do jogo.

Na maioria dos jogos de computador, os principais objetos que precisam evitar colisões e intrusões são a paisagem e o ambiente do nível. São as estruturas estáticas, não interativas e indestrutíveis, tais como montanhas, árvores, edifícios e cercas. Neste caso, o personagem é representado apenas por um ponto, e o método de divisão binária do espaço é utilizado (ou seja, o método de divisão recursiva do espaço euclidiano em conjuntos convexos e hiperplanos). Como resultado, os objetos são apresentados na forma de uma estrutura de dados utilizada para executar eficientemente operações em computação gráfica 3D, incluindo a classificação dos objetos visuais em ordem de distância do observador e detecção de colisão. Isto proporciona uma forma viável, simples e eficaz de testar se um ponto representando um personagem está ou não no ambiente (terreno). As colisões entre personagens e outros objetos dinâmicos são consideradas e tratadas separadamente. Mais informações sobre colisões podem ser encontradas na Seção 3.1.2.

Animações

Antes que o modelo seja animado, um "esqueleto" é criado nele. Este processo é chamado de *rigging*. Os modelos de seres vivos e todos os objetos que supostamente devem ser animados podem ter ossos virtuais. Por exemplo, o manto do protagonista. Os ossos do modelo dependem um do outro de tal forma que, por exemplo, quando a mão é deslocada, os ossos da palma também se moverão. Toda animação subsequente depende de quão bem são realizados o *rigging* e o *skinning*.

A animação é uma técnica para criar a ilusão de imagens em movimento, ou seja, movimento ou mudança da forma dos objetos. O *morphing* utiliza uma sequência de imagens estáticas (quadros) que se substituem entre si

em alta frequência (de 12 quadros por segundo para animação desenhada à mão até 30 quadros por segundo para animação por computador).

Efeitos

Os efeitos visuais podem ser divididos em dois tipos principais de tarefas:

- Efeitos de jogabilidade (efeitos de interação);
- Efeitos naturais (efeitos ambientais).

O princípio aplicado depende do projeto específico.

Entre outros gêneros, os efeitos de jogo são mais importantes em jogos de luta e RPGs. Por exemplo, quando seu personagem jogável empurra o personagem jogável de outro jogador, a animação de interação correta deve ser jogada para cada um deles. Existem outros gêneros de jogo, como os atiradores, e especialmente os realistas, como os efeitos naturais, que são tão importantes quanto a jogabilidade. Exemplos de efeitos naturais são cachoeiras, neblina e chuva.

Iluminação de cenas

A iluminação das cenas é necessária para que o jogador possa ver a cena. A luz afeta as emoções. A conexão entre a imagem e a resposta emocional fornece outra ferramenta poderosa que ajuda a trabalhar com o caráter, narrativa, som e mecânica de jogo. Neste caso, a interação da luz com a superfície permite que ela influencie o brilho, a cor, o contraste, as sombras e outros efeitos.

Devido às características estruturais, o olho humano reconhece objetos em três dimensões dentro de uma área de 110°, e totalmente coloridos em uma faixa ainda menor. Como a visão central (dentro da faixa mencionada acima) é a primeira coisa que um ser humano usa, ele deve obter os elementos críticos que o jogador deve definitivamente ver como pretendido pelo projetista. A visão periférica fornece contexto e reforça a visão central. Em um jogo, se os elementos que caem na visão periférica não fornecem o contexto necessário ou contradizem os elementos que estão na visão central, então a conexão entre o projetista e o jogador quebra.

A seguir, exemplos do uso correto da iluminação de cenas:

- A luz caindo no campo de visão central pode guiar o jogador;
- A iluminação pode mudar de estrutura. Por exemplo, o uso de uma lanterna faz dela a principal fonte de luz para o jogador. A perspectiva alterada capta a visão do jogador sobre a área iluminada e corta todo o resto devido ao forte contraste;
- A iluminação pode orientar e também criar uma atmosfera, como por exemplo, uma sensação de medo. O jogador é mantido em constante tensão quando, por exemplo, em algum lugar na escuridão, há um terrível inimigo escondido;
- A direção da luz pode facilitar a busca de elementos no nível do jogo ou torná-la mais complicada;
- A falta ou o excesso de iluminação pode levar o jogador a usar itens ou ações especiais.

A iluminação de cena é uma das áreas mais importantes de validação, pois a iluminação cria a estética do jogo e influencia a experiência do jogo. Assim, muitas das técnicas de iluminação de cena vistas em artes visuais, cinema e arquitetura são utilizadas em jogos de computador para complementar a estética do espaço virtual e melhorar a experiência do jogador. Entretanto, os jogos são muito diferentes do cinema ou do teatro; seus ambientes são dinâmicos e imprevisíveis. Além da iluminação estática da cena, são utilizadas fontes de luz dinâmicas. Elas adicionam interatividade e as emoções certas.

Precisão histórica

Qualquer item, tal como um elemento gráfico, animação ou efeito, deve corresponder ao estilo geral de jogo. Um texto ou descrição sonora de um objeto, modelo ou local também deve corresponder à descrição histórica do protótipo.

O conteúdo gráfico rico e complexo é uma das principais diferenças entre os jogos de computador e qualquer outro tipo de software. Isto muda significativamente a abordagem dos testes. Para testar gráficos em jogos, o testador deve ter conhecimentos adicionais de física e ótica, ter um entendimento das técnicas de renderização de cores e conhecimento de história. O teste de gráficos é uma das atividades de teste mais críticas, pois é aqui que se concentra a maior parte dos defeitos, que afetam diretamente a percepção do usuário sobre o jogo e sua experiência de jogo.

3.1.2 Tipos de defeitos de conteúdo gráfico

Uma categoria comum de defeitos nos testes gráficos são os defeitos visuais. Os defeitos gráficos incluem o rasgo da imagem na tela, a ausência de texturas, e o corte inesperado de certas áreas da imagem.

Ao criar gráficos e animação para jogos de celular, são usados os mesmos mecanismos que para um PC. A diferença é que as mesmas máquinas são adaptadas para plataformas específicas, o hardware que é usado nelas e suas capacidades técnicas [Gregory18]. Portanto, os defeitos que ocorrem em tais jogos são semelhantes.

Falta de textura

Entre os defeitos mais frequentemente encontrados ao testar as texturas estão

- Falta de texturas dos objetos gráficos;
- Perda de texturas durante o jogo;
- Texturas e entalhes temporários.

Como mostra a experiência, os dois primeiros grupos de defeitos estão na maioria das vezes relacionados à performance insuficiente dos processadores gráficos ou drivers desatualizados de placas gráficas. Em comparação, as texturas temporárias podem aparecer devido a um defeito no desenvolvimento.

Nível de Detalhe (LoD)

Em um jogo 3D moderno, muitos objetos localizados a diferentes distâncias da câmera (do ponto de vista do jogador) podem ser exibidos no cenário ao mesmo tempo. A fim de reduzir a carga no sistema, os desenvolvedores usam a tecnologia de nível de detalhe (LoD).

Quando os desenvolvedores criam um objeto de jogo (p. ex., uma árvore, um edifício ou um carro), eles adicionam variantes de modelos de objetos ao jogo. Estas incluem modelos com um número reduzido de polígonos e geometria simplificada (modelos de *Low-poly*) e modelos mais detalhados com muitos polígonos (modelos de *High-poly*). Dependendo da distância até a câmera, modelos com um número diferente de polígonos são usados para exibir o mesmo objeto no jogo. Nas proximidades da câmera, quando é necessário atingir a máxima qualidade de imagem, são usados modelos de muitos polígonos. Conforme a câmera se afasta, eles são substituídos por modelos menos detalhados com menos polígonos. A uma distância suficientemente longa, o modelo é exibido apenas como uma silhueta ou não é renderizado de forma alguma. Isto permite reduzir o número de polígonos processados e aumentar a performance do jogo.

Por exemplo, para uma floresta no horizonte, texturas de baixa resolução, podem ser usadas apenas com a cor exibida, sem relevo e reflexos. Se o jogador se aproximar, as árvores aparecerão em todos os detalhes.

A tecnologia LoD também é utilizada para animações, esqueletos e inteligência artificial de bots, onde um programa automatizado joga um determinado jogo em nome de um jogador humano. Por exemplo, em jogos de tiro com muitos inimigos no quadro, os bots terão diferentes níveis de detalhe. Os bots ao fundo serão representados de forma simplificada com baixos detalhes de textura, enquanto os oponentes próximos serão cuidadosamente desenhados com animações elaboradas e parecerão inteligentes o suficiente para lutar contra o jogador.

Os desenvolvedores também mudam o nível de detalhe dependendo da taxa de quadros atual, da velocidade de movimento do objeto na tela e do número total de objetos visíveis simultaneamente.

Os defeitos decorrentes do uso do LoD estão principalmente relacionados com a baixa performance do sistema. Os desenvolvedores tentam garantir que o aumento no detalhe do objeto ou a substituição do modelo ocorra de forma suave e imperceptível para o jogador, embora nem sempre seja este o caso. Em jogos que funcionam com hardware fraco, os modelos de *Low-poly* não podem ser substituídos por modelos de *High-poly* com rapidez suficiente. Como resultado, o jogador que se aproximar de um objeto verá primeiro um modelo "feio", e depois mudará para um modelo melhor.

Colisões

Colisão é a forma pela qual um objeto no jogo ganha tamanho e reage a colisões com outros objetos. Uma malha de colisão ou colisão é criada para que o modelo de objeto acabado interaja com seu ambiente e outros objetos. Esta é uma forma simplificada invisível de um objeto que é ligada a ele para calcular colisões com outros objetos. O colisor pode ser chamado de modelo físico do objeto. Ele permanece invisível para o jogador, mas qualquer colisão é tratada pelo motor do jogo [Buttfield19].

A forma do colisor deve corresponder aproximadamente à malha do modelo, mas muitas vezes uma aproximação bastante grosseira é suficiente. Para a maioria dos objetos estáticos no ambiente, como edifícios, rochas e carros batidos, os desenvolvedores usam colisores de baixa polimerização. Isto é indistinguível na jogabilidade, mas melhora a eficiência de processamento de todos os objetos visíveis.

Neste caso, o tamanho do colisor deve corresponder ao modelo visível, especialmente se o jogador puder interagir diretamente com este objeto.

Se o colisor do objeto for muito menor que seu modelo visual, então o personagem será capaz de "romper a textura" ou passar por ela. Estritamente falando, a frase "romper a textura" (p. ex., quando um personagem pode atravessar outro objeto) é incorreta, mas descreve perfeitamente o que o jogador vê quando seu personagem está parcial ou completamente imerso em outro objeto.

Tais falhas são mais críticas em jogos multiplayer, pois podem dar ao jogador uma vantagem de jogo ilegal. Por exemplo, um tanque acionado em uma pedra será praticamente invisível para outros jogadores. Ao mesmo tempo, tal pedra não o protegerá do fogo inimigo. É por isso que os testadores devem prestar atenção adicional a tais objetos nos mapas no modo jogador versus jogador (PvP).

No entanto, alguns objetos do jogo podem ser decorativos e não ter nenhuma colisão. Por exemplo, um tanque pode facilmente passar por um arbusto, e o jogador não precisa ser forçado a dar a volta ou pular sobre cada pequena pedra ou objeto.

As convenções de jogo são aplicadas aqui. Da mesma forma, pode-se enviar um personagem com uma longa lança para um estreito corredor baixo para defender-se de inimigos que atacam de diferentes direções. Ao contrário da vida real, o personagem será capaz de girar livremente em qualquer direção desejada, e nem mesmo notará que, ao girar, a lança passa através das paredes. Além disso, os longos cabelos do personagem podem atravessar as ombreiras maciças da armadura. No entanto, a presença de uma cerca baixa, que de repente impede o jogador de subir, irá irritar o jogador e reduzir seu prazer.

Se o tamanho do colisor do objeto for maior que o modelo visível, uma situação pode surgir quando o personagem descansa contra uma parede invisível ou fica no ar sobre um pequeno suporte. Tais falhas são frequentemente usadas para a passagem rápida não planejada de jogos. Os chamados *speedrunners* estão procurando especificamente lugares em locais onde os desenvolvedores eram preguiçosos ou esqueceram de redimensionar os colisores dos objetos ao redor. Às vezes isso permite que um jogador contorne uma parte significativa do local e reduza o tempo de passagem.

Hitbox

Se for assumido que um personagem ou objeto de jogo pode sofrer danos, por exemplo, de um projétil inimigo, então durante a batalha é necessário calcular a posição de cada ponto na superfície deste objeto e determinar se houve ou não um golpe.

Para objetos de forma complexa, isto é muito caro e nem sempre justificado.

Para simplificar tais cálculos, é utilizado o conceito de ***hitbox***. Por exemplo, em jogos de plataforma 2D ou de luta pode ser na forma de um ou mais retângulos. Sua posição em relação uma à outra é fácil de calcular.

Para um objeto 3D, a forma mais simples de uma caixa de impacto é uma esfera. Neste caso, é suficiente conhecer seu centro e raio para os cálculos. Desta forma, a qualquer momento, é fácil determinar se um projétil ou outro objeto está dentro desta esfera. No entanto, uma *hitbox tubular* tem uma forma mais adequada porque contém menos espaço vazio em comparação com uma esfera, e é suficiente conhecer o ponto central do objeto e três dimensões: comprimento, largura e altura para os cálculos.

Para uma maior autenticidade, o objeto pode ser descrito por várias *hitbox*, separadamente para cada parte do modelo: corpo, (p. ex., braços, pernas, cabeça, cauda, e até mesmo armas).

Entretanto, as partes do modelo que podem ser danificadas e que podem causar danos nem sempre coincidem. Portanto, em alguns jogos, o ***hitbox*** e o ***hurtbox*** (caixa de ferimento) são alocados separadamente para o mesmo objeto, de modo que áreas que infligem danos (p. ex., armas, punhos etc.) e áreas sobre as quais os danos são infligidos (p. ex., cabeça, membros) são definidas. Para determinar se um personagem deve sofrer danos, é testado se a *hitbox* da arma e a *hurtbox* do corpo do personagem se sobrepõem.

A principal tarefa do testador ao trabalhar com *hitbox* será testar se a área visível do impacto corresponde ao resultado real do ataque.

Mesmo em grandes jogos, há momentos em que o jogador claramente vê que sua arma tocou o inimigo, mas não causou danos. Pelo contrário, se a *hitbox* for muito grande, a espada do inimigo pode ferir o personagem mesmo que passe visualmente.

Outros defeitos

Quando se trabalha em um grande projeto, cada etapa da produção de modelos passa por muitas aprovações e confirmações. Ela é testada por artistas, modelistas e texturizadores, entre outros. Mesmo assim, os testadores ainda podem detectar defeitos visuais que apareceram durante o processo de produção.

Inconsistência com o protótipo histórico

Os jogadores podem aceitar armaduras irrealistas com ombreiras gigantes e uma aparência implausível de uma máquina voadora em jogos de fantasia ou de ficção científica. No entanto, se o jogo afirma ser realista, então certamente haverá jogadores meticulosos que indicarão que as hélices de certos modelos de aeronaves estavam localizadas de forma completamente diferente, e havia muito mais armas em certos navios de guerra.

Por isso é uma boa ideia que o testador se familiarize com pelo menos algumas fotografias do objeto testado, se possível, para imaginar como ele era na realidade.

Objetos pendurados no ar

Muitos especialistas podem trabalhar em um local ao mesmo tempo. Alguns adicionam ou removem objetos; outros mudam a geometria do terreno (mapa) ou fixam azulejos (texturas que são aplicadas ao terreno). Por causa disso, defeitos aparecem quando um objeto previamente existente é afogado em um terreno ou fica pendurado no ar, porque seu antigo "suporte" foi removido.

Visibilidade da união de texturas

Tais defeitos são mais frequentemente encontrados no terreno de grandes mapas e locais quando várias texturas são misturadas. Por exemplo, uma "costura" pode ser visível nas transições entre uma superfície de grama e areia ou pedra. Para obter mais detalhes, consulte o Capítulo 3.1.

Destruição de objetos

Normalmente, o modelo tem vários estados a fim de acrescentar realismo a diferentes situações, tais como original, danificado e destruído. Se o jogo assume a destruição realista de objetos, então um modelo de colisão especial é criado para eles, possivelmente mesmo com seu próprio modelo de colisão. Em um determinado momento, o modelo original é substituído por um modelo destruído.

Aqui é importante assegurar que o efeito da destruição corresponda ao próprio modelo. O caminhão destruído deve ser diferente do original e as peças de um prédio de tijolos quebrados não devem ser de madeira.

Defeitos de iluminação

Os defeitos de iluminação incluem:

- Defeitos na iluminação global;
- Defeitos de fontes pontuais de luz;
- Defeitos das fontes de luz de holofote;
- Defeitos de fontes com uma área;
- Defeitos na direção da fonte de luz;
- Defeitos na emissão da fonte de luz;
- Defeitos de difusão da fonte de luz.

Os modernos motores de jogos incluem capacidades muito avançadas para criar diferentes tipos de fontes de luz em total conformidade com as leis da ótica. Entretanto, ainda existem jogos onde a iluminação não é ideal, por exemplo, devido a configurações erradas de iluminação ou à falta de experiência do desenvolvedor. A consideração da composição visual (posição da luz, seus ângulos, cores, campo de visão e movimento) tem um grande impacto sobre como os jogadores percebem o ambiente do jogo, e cria a atmosfera emocionalmente certa para engajar o jogador. Os projetistas devem cuidar disso, criando integridade visual. [Lee16], [Tavakkoli18], [Romero19], [Tavakkoli18], [Romero19].

Defeitos de animação

Este defeito resulta de partir o esqueleto para o modelo, chamado de *skinning*. Quanto mais “ossos” o modelo tiver, mais realista a animação pode ser criada. Diferentes objetos podem ser animados, como, por exemplo, o cabelo do protagonista.

Os “ossos” de um modelo são dependentes um do outro. Portanto, quando uma mão é deslocada, os “ossos” da palma da mão também se movem. Toda animação subsequente depende de como tudo foi feito e montado de forma realista durante a manipulação e o *skinning*.

Se o projetista cometeu erros, isso poderia levar a braços/pernas de personagens alongados em várias animações, e partes de modelos que são "desprendidos" do resto dos elementos. Muitas vezes, estas falhas aparecem ao colidir com outros modelos ou em várias animações do modelo.

Falhas de efeitos visuais (VFX)

Os jogos frequentemente utilizam vários efeitos relacionados às ações de personagens, eventos e fenômenos naturais, tais como explosões, faíscas, fumaça, aparecimento e desaparecimento.

Quando se trabalha com animações, os efeitos visuais são geralmente presos aos “ossos”, aos *helpers* (ou seja, objetos utilitários que são usados para criar e animar modelos), e aos objetos de cena. A criação de efeitos visuais é um processo iterativo onde equipes de animação e artistas de efeitos visuais interagem. Estes últimos podem ser solicitados a fazer mudanças ou adições à animação, como por exemplo,

- adicionar mais quadros para uma trajetória mais suave de uma espada;
- girar o *helper* para usar sua rotação como direção para salpicos de sangue;
- mudar o ângulo da câmera para evitar um aspecto feio de malha granulada.

Inicialmente, é importante que tais efeitos sejam sincronizados com os eventos que os produzem. Por exemplo, o fogo e a fumaça do cano da arma devem estar sincronizados com o momento do disparo e o projétil que sai do cano. Caso contrário, o realismo do evento será violado. Tais ocorrências são tradicionalmente consideradas um fracasso.

Outro problema que causa falhas VFX é o descumprimento das condições técnicas associadas à manutenção da taxa de quadros e à otimização dos recursos de hardware utilizados para exibir corretamente o efeito. Não importa quão grande seja o efeito, é improvável que o jogador esteja feliz em ver nuvens de poeira tremendo ou explosões presas no tempo na tela. Como resultado, os efeitos especiais nos jogos são sobrecarregados com limites técnicos rigorosos, cujo objetivo é manter um certo número estável de quadros exibidos.

Ao testar o VFX, o testador precisa garantir que as capacidades de hardware do dispositivo de jogo estão sendo usadas de forma otimizada e que os efeitos estão em sincronia com os eventos que os acionam.

É desejável encontrar o maior número possível de defeitos na fase de produção do modelo. Por exemplo, exceder o número de polígonos do modelo ou defeitos de texturas, o que aumenta significativamente a carga sobre o hardware utilizado nos cálculos do modelo. Isto inevitavelmente leva a longos atrasos no jogo devido à sobrecarga do processador do adaptador de vídeo.

Para testar isto, o desenvolvedor deve ter recebido requisitos pré-determinados e fixos para aspectos como LoD e modelos de colisão.

Os testadores, como regra, encontram manifestações desses defeitos, incluindo as quebras e falhas causadas por eles. Se um objeto tiver, por exemplo, um modelo de colisão com muitos polígonos, o defeito não será detectado enquanto a taxa de quadros for alta durante a interação com este objeto.

3.2 Abordagens para testar gráficos em jogos

3.2.1 Testes artísticos

O processo de criação de um objeto gráfico passa por muitas etapas e revisões diferentes. Algumas pessoas criam o desenho e outras o colocam na aplicação. O resultado pode ser algo que o autor do desenho não pretendia. Um testador pode testar a conformidade do desenho com as regras, e notar defeitos, mas há defeitos que o artista notará mais facilmente. Por exemplo, em uma escalada, uma transparência insuficiente do fundo ou da aplicação não se comporta como pretendido.

Por outro lado, o artista não pode prever tudo e só notará o defeito na aplicação em execução, quando este já se tornou um fracasso. Portanto, é uma boa prática dar a aplicação ao projetista para revisão após a implementação para ajudar a eliminar defeitos críticos. Neste processo, o testador prepara o ambiente necessário, cria contas de teste, descreve os defeitos após os testes e, então será capaz de testá-los ou enviá-los ao artista para revisão.

Em projetos onde há uma grande quantidade de conteúdo, frequentemente há especialistas que estão envolvidos nos testes gráficos. Estes são principalmente artistas que entendem a fundo o processo de criação de modelos. Eles fazem uma confirmação de cada uma das seguintes etapas da criação do modelo:

- Criação da geometria;
- Criação de texturas;
- Criação de modelos de colisão.

O objetivo deste trabalho é melhorar a qualidade dos modelos antes de exportar para o motor de jogo. Por exemplo, eles testam aspectos como a poligonagem (ou seja, o número de polígonos) em geometria, a correção de sua regionalização, a correção do desdobramento do modelo, a presença de polígonos alongados, e as costuras no modelo.

Os testes artísticos são uma tarefa grande e complexa que pode ser realizada em vários estágios da produção, começando com a criação da geometria e texturas mais simples, e terminando com a exportação do modelo para o motor e a sua colocação no mapa. Para realizar testes artísticos, o testador muitas vezes não precisa de ferramentas especiais e editores de conteúdo. Ele pode ser executado diretamente durante a execução do jogo. Entretanto, a disponibilidade de ferramentas e editores de conteúdo permite que os defeitos sejam encontrados muito mais rapidamente e com muito mais eficiência.

Os testes artísticos são realizados por:

Papel	Responsabilidades
Artistas:	Revisar objetos. Visualizar e avaliar objetos
Modeladores 3D:	Revisar objetos
Supervisores:	Visualizar e avaliar objetos
Engenheiros de testes:	Após a exportação final dos modelos para o motor
Jogadores:	Participar de testes do jogo

3.2.2 Testes técnicos

Os testes técnicos envolvem um conjunto de tarefas relacionadas a parâmetros técnicos de gráficos, como por exemplo:

- O cumprimento dos limites do número de polígonos modelo, que foram mencionados anteriormente;
- Formatos de textura;
- Distâncias de comutação LoD, para as quais é suficiente comparar os recursos disponíveis com a especificação.

Os testes técnicos propriamente ditos são realizados tanto por testadores quanto por artistas técnicos. Ela inclui os procedimentos descritos abaixo:

Busca de arquivos não utilizados e temporários em recursos de conteúdo gráfico

Ao trabalhar com artistas, os recursos podem ser acidentalmente adicionados ao repositório que não serão utilizados posteriormente. Por exemplo, um desenvolvedor pode carregar para o repositório todos os recursos, inclusive os não utilizados. Neste caso, a quantidade de espaço ocupado no disco rígido pelo cliente do jogo aumenta, o que reduz a performance e eventualmente aumenta a quantidade total de patch entregue ao jogador (p. ex., atualizando arquivos de videogames individuais). Para evitar isto, é feita uma busca por recursos não utilizados.

Busca de todos os registros de conteúdo necessários nos recursos do cliente

Nos recursos do cliente do jogo, o conteúdo gráfico não existe independentemente de outros recursos. As referências a texturas são escritas em modelos, as referências a modelos são escritas em mapas. Mesmo os menores defeitos em tais registros podem levar a consequências inesperadas.

Formatos de teste de textura

Para cada tipo de textura, os artistas técnicos estabelecem seus próprios requisitos. Isto se deve à necessidade de encontrar um equilíbrio entre a performance do conteúdo e o componente visual. O formato da textura implica o tipo de compressão, e o seu tamanho.

Teste dos logs do cliente ao testar o conteúdo

Às vezes, o trabalho dos subsistemas do cliente não é muito perceptível. Por exemplo, os pós-efeitos são efeitos que se sobrepõem, imagem sobre imagem, como uma lente quando se olha para o sol. Uma textura dedicada é criada para cada pós-efeito, de acordo com a sobreposição na tela. Quando se olha para o sol com uma lente, por exemplo, pode-se ver a lente e os raios do sol, independentemente da textura ser designada para o efeito posterior ou não. Neste caso, o jogador simplesmente não verá o defeito, apesar de que ele exista. Para encontrar tais defeitos, é necessário examinar os logs do cliente, pois eles certamente mencionarão que o renderizador tentou utilizar uma textura que não existe nos recursos.

Verificação da conformidade com os limites do número de texturas, modelos

Cada tipo de conteúdo tem seus próprios limites, que geralmente são gerais para um grupo específico. Por exemplo, para edifícios indestrutíveis de vários andares, é permitido usar até 15 mil polígonos, mas para edifícios de um andar somente 10 mil polígonos. Devido ao fato de que as edições de conteúdo estão em andamento e o número de downloads para o repositório é grande, após cada nova exportação do modelo, é necessário testar o cumprimento dos limites e requisitos para evitar situações associadas à falta de recursos de hardware necessários para o cálculo dos modelos de processadores de vídeo.

Apesar da simplicidade dos testes, é vital compreender a importância dos testes técnicos. Os gráficos podem ter um enorme impacto na performance geral do sistema. Portanto, os testes técnicos incluem testes de performance. para garantir que a performance e o tamanho do jogo permaneçam no nível desejado. Ao aumentar o número de texturas e modelos, o tamanho da construção também aumenta. Este descasamento de formatos de textura piora o indicador de taxa de quadros, o que leva a um aumento na memória consumida pelo cliente. Os testes técnicos garantem que o conteúdo artístico será entregue ao jogador em toda a sua extensão e escopo.

3.2.3 Teste de jogabilidade

O teste de jogabilidade é uma abordagem de teste que visa aqueles fatores que afetam a jogabilidade. Isto pode ser tanto uma avaliação da conformidade do modelo de colisão com o modelo visual, quanto uma verificação de que as configurações dos objetos de jogabilidade estão de acordo com os requisitos do modo de jogo.

O teste de jogabilidade dos objetos inclui:

- Testar a conformidade dos Pontos de Saúde (PS). Este é um valor na dramatização (role-playing) e jogos de computador que determinam a quantidade máxima de dano que pode ser aplicada a um objeto para destruí-lo. Apesar dos PS serem atribuídos aos objetos automaticamente, levando em conta o material, tamanho e tipo do objeto, ainda podem ocorrer defeitos. Portanto, a finalidade da PS é testada manualmente quanto às características do objeto.
- Teste de que o modelo de colisão corresponde ao modelo visual. Como mencionado anteriormente, o modelo de colisão é significativamente simplificado em comparação com o modelo visual. Isto pode levar a defeitos quando o jogador está visualmente escondido atrás do modelo, mas na verdade pode ser danificado por um inimigo.

Os testes gráficos são um processo complexo e seus tipos se sobrepõem uns aos outros. Esta abordagem de teste descrita acima garante que o conteúdo gráfico do jogo seja minuciosamente testado em vários estágios para detectar defeitos gráficos.

3.3 Execução de testes gráficos

3.3.1 Execução de testes gráficos em diferentes etapas da produção de objetos

Criação de uma caixa cinza

Este é um modelo simplificado de um objeto gráfico, que na verdade é uma maquete usada para testar a jogabilidade. Nesta fase, os defeitos descobertos estão principalmente relacionados à conveniência com a qual os artistas podem trabalhar com este modelo.

Criação da geometria visível

A próxima etapa de modelagem e mapeamento inclui o mapeamento e a criação de uma projeção 3D sobre um objeto físico do ambiente. Isto leva em conta sua geometria e regionalização no espaço. A geometria visível e o desdobramento UV do modelo são criados e a correspondência entre as coordenadas na superfície do objeto 3D (X, Y, Z) e as coordenadas na textura (U, V) é verificada. Uma vez que as inspeções são frequentemente realizadas utilizando editores gráficos 3D, defeitos nesta etapa também podem ser detectados por artistas e modeladores 3D. Como regra, estes defeitos incluem:

- *Geometria duplicada*. A duplicação de polígonos degradará significativamente a performance.
- *Detalhes excessivos*. Por exemplo, um modelo que está fora do campo de visão do jogador tem *biséis*, o que significa que ao processar tais "decorações" serão alocados recursos adicionais de hardware que o jogador simplesmente não notará.
- *Detalhes insuficientes*. Este tipo de defeito é exatamente o oposto do detalhe excessivo. Se o modelo for claramente visível para o jogador, ele deve ser suficientemente detalhado para não parecer angular.
- *Partes da geometria que não são renderizadas*. Por exemplo, a falta de peças necessárias de um modelo pode torná-lo irreal.

Texturizando o modelo

Nesta fase, as texturas são aceitas pelo artista ou diretor de arte. Neste caso, a aplicação correta das texturas, a ausência de costuras e *pulling* (ou seja, topologia poligonal incorreta) pode não ser descoberta. O teste diz respeito apenas à paleta geral de cores e ao nível de contaminação (p. ex., presença de sujeira, inserções de cores desnecessárias, pixels aleatórios de diferentes cores no modelo). Os defeitos típicos encontrados nesta fase são casos em que o mapeamento automático e a subsequente texturização são realizados de tal forma que texturas de cores diferentes são aplicadas às partes do modelo que têm uma borda comum, o que resulta em costuras perceptíveis.

Revisão do modelo de LoD e colisão do objeto

Nesta fase, tanto os artistas quanto os projetistas de nível estão envolvidos na avaliação da qualidade dos modelos. Os projetistas de nível avaliam os modelos de colisão em termos de seu impacto sobre a jogabilidade. Por exemplo, se é necessário utilizar um modelo detalhado em um local específico ou se ele pode ser simplificado. Os defeitos típicos neste caso são modelos que ou são muito detalhados (p. ex., mais do que o número permitido de polígonos no modelo de colisão), ou não são suficientemente detalhados (p. ex., contêm um modelo de colisão número de polígonos insuficientes para o detalhamento).

Exportando o modelo para motor de jogo

Isto considera a exatidão das configurações para as distâncias em que se deve trocar os LoDs e o número de pontos de saúde do objeto.

Os testes são realizados pelos testadores diretamente no motor do jogo, tanto nos editores de conteúdo quanto diretamente na aplicação do jogo. Ao mesmo tempo, a maioria dos testes artísticos são reduzidos a uma avaliação visual do objeto, e uma série de testes duplicam os testes de artistas e modeladores. Os testes abordam o seguinte:

- Visibilidade da troca de LoDs de objetos, levando em conta a regionalização e a situação na qual o objeto será utilizado;
- Lacunas e costuras na geometria do objeto que são perceptíveis para o jogador;
- Visibilidade das inscrições em vários objetos gráficos;
- O efeito de destruição do modelo (p. ex., considerando cor e tamanho);
- Defeitos visuais na animação de objetos.

Nos testes artísticos, os jogadores podem participar diretamente que avaliam visualmente o conteúdo e podem encontrar outros defeitos gráficos.

Os testes gráficos são frequentemente feitos em conjunto com testes de mapas de jogo. Neste caso, há várias outras etapas.

Colocação de objetos no mapa

Esta é uma das etapas mais importantes para os testes. A colocação dos objetos é feita por projetistas de nível. Neste processo, um defeito típico que surge são os objetos "pendurados" e encastrados sob o mapa (terreno). Isto acontece frequentemente quando muitos especialistas trabalham com o mapa em paralelo que podem alterar a geometria do terreno, expor objetos e editar o próprio mapa.

Criação e colocação de efeitos

Nesta fase, os artistas de efeitos trabalham na criação de novos efeitos especiais. Observá-lo em várias situações lhes permite detectar defeitos artísticos e técnicos de renderização. Um exemplo de tal teste é um teste visual de um efeito em um motor. Os artistas de efeitos observam cada efeito no cliente do jogo para identificar parâmetros de efeitos incorretos (p. ex., tempo insuficiente de decaimento do efeito).

Os testes gráficos são frequentemente feitos usando testes de jogos. Isto se refere a qualquer teste no qual o mapa é testado em situações de jogo. Os objetivos deste teste são extensos e se aplicam a qualquer estágio da produção (p. ex., fazer uma avaliação da jogabilidade no mapa). Mais perto do final da produção, pode envolver uma avaliação artística e de jogabilidade do mapa do ponto de vista dos jogadores. Além de coletar estatísticas que os projetistas de jogos precisam ajustar o equilíbrio no mapa, tais testes revelam vários problemas artísticos e técnicos, desde objetos pendurados até defeitos de renderização gráfica (p. ex., congelamentos, defeitos e falhas). Em testes puramente artísticos, o teste de jogos revela problemas que são perceptíveis do ponto de vista do jogador.

Testando o mapa

Esta é a etapa final de testes onde o mapa está completamente pronto e nenhum trabalho mais está sendo feito nele (com raras exceções). Os testadores conduzem uma validação completa do mapa com base em uma lista de verificação de validação do mapa, desde a busca de objetos pendurados até o teste de animações. Os testadores se concentram em tais testes, o que lhes permite detectar muito mais defeitos (p. ex., objetos "pendurados") em comparação com os artistas. Durante os testes com artistas, os artistas inspecionam visualmente os mapas, enquanto os testadores usam um conjunto especializado de ferramentas para detectar problemas mais graves em objetos gráficos.

3.3.2 Testando gráficos para precisão histórica

Uma área separada de teste de conteúdo gráfico é o teste de precisão histórica. A exatidão histórica considera tanto a exatidão histórica quanto a autenticidade histórica.

A autenticidade histórica influencia a qualidade de um jogo, no qual um episódio específico da história é descrito usando imagens características. Estas não fazem referência a eventos específicos, personalidades etc.

A precisão histórica influencia a qualidade de um jogo, no qual é descrito um período específico da história. Todos os principais eventos são descritos em sua dinâmica, incluindo todas as personalidades relevantes e suas ações com imagens características do período. Eles excluem fundamentalmente imagens alheias ao período.

Como regra, nos jogos é impossível manter a autenticidade histórica, pois o foco é a jogabilidade viciante. É difícil envolver os jogadores em um jogo no qual o resultado é pré-determinado desde o início.

Entretanto, a precisão histórica é uma área de jogo que requer atenção especial, especialmente quando a trama do jogo está ligada a uma era histórica específica.

Aspectos típicos a serem considerados ao testar o conteúdo gráfico para a exatidão histórica são:

- A semelhança dos personagens com os protótipos históricos;
- Precisão na reprodução de objetos arquitetônicos;
- Precisão na reprodução de armas, equipamentos, veículos, roupas de uma determinada época e suas características;
- Precisão na recriação da vida cotidiana;
- Precisão de eventos históricos e datas.

Ao realizar testes de precisão histórica, o testador deve ter uma ampla gama de conhecimentos e horizontes. Muito frequentemente, os desenvolvedores envolvem consultores históricos nos testes de seus produtos para criar o jogo mais realista possível.

3.4 Ferramentas de suporte para testes gráficos

Ao testar o conteúdo, os testadores gráficos utilizam uma ampla gama de ferramentas, desde editores de conteúdo dentro do jogo até ferramentas automatizadas e scripts de teste. Como regra, os jogos são criados usando mecanismos de videogames que contêm várias ferramentas embutidas, incluindo aquelas para testar gráficos: um editor de objetos, um editor mundial e um editor de iluminação.

Todas as ferramentas proporcionam a capacidade de personalizar o conteúdo existente para que o jogador possa interagir com ele. Os editores de modelos permitem efeitos vinculativos a modelos como vários gatilhos, sons e outros eventos que acontecem ao interagir com o motor. O editor mundial permite criar um mapa, colocar objetos sobre ele e personalizar a mecânica de jogo do mapa. É nestes editores que os testadores fazem a maioria dos testes. Ao fazer isso, eles usam as mesmas ferramentas que os desenvolvedores. Estes editores também fornecem um extenso conjunto de ferramentas que permite configurar o conteúdo e testá-lo.

As ferramentas especializadas são normalmente desenvolvidas pelos fabricantes de placas gráficas. Estas ferramentas permitem a captura de quadros no jogo e análise detalhada de qualquer aplicação usando vários conjuntos de APIs usados para criar gráficos 2D e 3D. Elas também permitem as seguintes avaliações:

- Como a estrutura do jogo está sendo preparada (p. ex., geometria, texturas, chamadas de desenho);
- Onde surgem os problemas de performance;
- A forma de execução do percurso;
- Depuração para modeladores 3D, artistas gráficos e animadores.

Mais detalhes sobre como cada uma dessas ferramentas é utilizada estão descritos na especificação particular do software. As ferramentas de teste gráfico incluem scripts automatizados que proporcionam a passagem automática do personagem ao longo de um determinado percurso no jogo. Eles são particularmente usados em testes de performance e compatibilidade.

4 Teste de som - 190 min

Palavras-chave

nenhuma

Palavras-chave específicas

ambiente, efeito binaural, distorção, ruído de atraso (Foley), oclusão, reverberação, salto, descontinuidades sonoras, efeitos sonoros, zonas sonoras, volume

Objetivos de aprendizagem

4.1 Características do conteúdo sonoro do produto do jogo

GaME-4.1.1 (K1) Características de recall do conteúdo sonoro de um produto de jogo

4.2 Tipos de defeitos no conteúdo sonoro

GaME-4.2.1 (K1) Recordar os tipos de defeitos no conteúdo sonoro.

GaME-4.2.2 (K2) Classificar os defeitos no conteúdo de som

4.3 Abordagens para testar o conteúdo sonoro em produtos de jogos

GaME-4.3.1 (K2) Resumir as principais abordagens para os testes de conteúdo-auditoria.

GaME-4.3.2 (K2) Resumir as principais abordagens para testar a mistura de música e sons.

GaME-4.3.3 (K2) Resumir as principais abordagens para testar a composição musical.

4.4 Execução de testes de som

GaME-4.4.1 (K2) Explicar os níveis de teste de conteúdo áudio-musical

GaME-4.4.2 (K1) Relembrar as características de integração de sons no cliente

GaME-4.4.3 (K1) Relembrar as áreas de responsabilidade dos testes de som

GaME-4.4.4 (K3) Aplicar abordagens para testes de som

4.5 Suporte de ferramentas para testes de som

GaME-4.5.1 (K2) Resumir o uso de ferramentas de teste de som

4.1 Características do conteúdo sonoro do produto do jogo

O som é uma parte importante de qualquer jogo moderno. Sons e música criam o clima certo, alertam sobre o perigo, transmitem as emoções dos personagens, complementam e criam uma imagem completa do mundo do jogo.

Uma pessoa confrontada com um filme ou jogo que não tem som algum é provável que experimente uma leve dissonância. Isto acontece porque seu cérebro está tentando ativar sua audição, mas não pode, e em vez disso sinaliza um erro de percepção sobre o que está acontecendo diante de seus olhos.

Sons de baixa qualidade, irreais, inapropriados ou de mau funcionamento podem irritar o jogador e quebrar a imersão na jogabilidade.

O processo de criar uma trilha sonora em um jogo é bastante demorado, pois é necessário levar em conta a complexidade de organização do conteúdo do jogo:

- Sons ambientais;
- Vozes de personagens;
- Trilha sonora de fundo;
- Efeitos sonoros;
- Vários objetos sonoros;
- Arquivos de voz.

Isto complica a tarefa de testes

4.1.1 Tipos de sons

O design de som encontrado nos jogos pode variar de jogo para jogo e ser criado para diferentes fins. Considere que tipos de sons é costume distinguir, que funções eles desempenham e para que propósitos são criados.

Música

O tema musical principal do jogo, seu cartão de visita, pode aparecer mesmo no jogo mais simples. As músicas dos antigos jogos de 8 bits têm sido clássicas há muito tempo.

As principais funções da música:

- Enfatiza a sensação épica do momento;
- Aumenta a dinâmica do evento;
- Aumenta a atmosfera;
- Coloca o jogador com o humor certo e explica o estado emocional do personagem;
- Subconscientemente, forma uma certa resposta emocional no jogador devido à melodia repetitiva.

Efeitos sonoros

Se o mundo do jogo está cheio de objetos com os quais o herói pode interagir, cada um deles deve ter sons realistas. Isso faz com que o mundo do jogo ganhe vida e seja crível.

Abrir portas, golpes, atirar e recarregar armas, o uso do kit de primeiros socorros, a explosão de latas de combustível, o som de vidros quebrados - tudo isso é necessariamente expresso e tocado no momento certo.

Mesmo em pequenos jogos, os desenvolvedores adicionam interação de efeitos sonoros com elementos de interface (botões, interruptores, itens de menu). Isto dá ao jogador um feedback e um entendimento de que as ações do jogador são percebidas e analisadas.

Sons de personagens (Foley)

Os sons feitos pelos personagens do jogo incluem o barulho das roupas, respiração, exclamações, passos, gemidos etc. O termo tem o nome de Jack Foley, um dos pioneiros dos efeitos sonoros. Em alguns jogos, como os jogos de tiro, tais sons podem ser um indicador de que o personagem está ferido. Isto será ainda mais claro para o jogador do que os números de danos apresentados.

Discurso

Embora o discurso seja reproduzido pelos personagens, é ainda melhor destacá-lo separadamente. Nos jogos de hoje, cada personagem não jogador (NPC) tem uma voz única, gravada por um ator profissional. Os desenvolvedores colocam muito dinheiro e esforço na interpretação da voz, pois linhas gravadas com qualidade podem transmitir corretamente as emoções dos personagens e criar uma conexão emocional entre eles e o jogador.

Sons ambientes

Sons ambientes referem-se aos sons característicos de um determinado local. Como o tema musical, o ambiente enfatiza a atmosfera e cria o clima necessário para o músico. O ambiente não infere nenhum objeto de onde vem este som. É um som ambiente que simplesmente acompanha um determinado local, situação ou estágio do ciclo do jogo.

Os sons ambientes são geralmente tocados em segundo plano e não dependem das ações do personagem. Exemplos de tais sons são:

- O ruído das folhas e o canto dos pássaros na floresta;
- Ondas borbulhantes no mar;
- Conversa indistinta dos visitantes de um ambiente;
- Sirenes de polícia e o barulho da passagem de carros etc.

Tudo isso ajuda o jogador a navegar no espaço e a perceber melhor o mundo do jogo ao seu redor.

4.1.2 Efeitos sonoros e tecnologia

As trilhas sonoras são frequentemente processadas em editores de som para criar um efeito autêntico, levando em conta as capacidades dos equipamentos e tecnologia de áudio modernos.

A fim de imergir o usuário no mundo virtual do jogo, o uso apenas de sons realistas pode não ser suficiente. Para aumentar o realismo do que está acontecendo, os desenvolvedores muitas vezes hiperbolizam alguns dos sons. Esta técnica é frequentemente vista, por exemplo, em filmes com cenas de lutas, onde os golpes e até mesmo os movimentos dos lutadores soam muito mais alto do que na vida real.

Para alcançar este efeito, várias técnicas e tecnologias são utilizadas.

Oclusão

Uma das técnicas mais comuns é o efeito de oclusão quando o som passa por uma barreira (p. ex., uma parede do edifício). Para maior realismo, os especialistas em som em tais situações não apenas diminuem o volume, mas acrescentam um efeito especial.

O som principal e os sons refletidos são realizados de forma diferente quando há uma parede em branco entre a fonte sonora e seu ouvinte, quando os objetos estão em salas diferentes, mas em linha direta de visão (porta), e quando os objetos estão na mesma sala, mas não podem se ver (coluna entre eles).

Reverberação

O efeito de reverberação é usado para transmitir volume e profundidade do espaço (a partir da reverberação).

Neste caso, o som da fonte em um espaço fechado é refletido das paredes e causa numerosos ecos, que gradualmente se desvanecem. Este efeito funciona bem se o personagem estiver em uma caverna, em uma pequena sala ou em outra sala com superfícies sonoras bem refletidas.

Com motores de som avançados, pode-se facilmente criar ecos e acrescentar volume aos sons de tiros, explosões, vozes, passos etc.

Efeito binaural

O efeito binaural (do latim bini - dois e auris - ouvido) se baseia no fato de que uma pessoa escuta com os dois ouvidos simultaneamente, e quando uma pessoa gira a cabeça para o lado, o som chega a um ouvido antes do outro. Graças a isto, uma pessoa pode determinar em que direção a fonte do som está, e até mesmo a distância até ela.

Microfones especiais são usados para gravar som com este efeito, e para sua percepção é recomendado o uso de fones de ouvido. Isto é necessário para que os sons possam entrar em cada ouvido a partir de uma direção

específica. O ambiente sonoro torna a imersão mais profunda e pode ajudar o tocador. Devido ao áudio binaural, o tocador sabe de que lado o som está vindo, (p. ex., de onde o inimigo aparecerá em um momento).

4.1.3 Área de Sons

Em alguns projetos, as chamadas áreas sonoras são utilizadas para distribuir o ambiente sonoro em diferentes locais.

Em jogos, um exemplo seria quando um personagem em um jogo de role-playing entra em uma vila e os sons característicos aparecem, ou em um jogo de estratégia o jogador aproxima a câmera de sua base e ouve os sons dos recursos sendo reunidos.

As zonas sonoras devem ter transições. Perto da transição, o som de uma zona deve desaparecer ligeiramente, e o som da outra zona deve aparecer.

Na realidade, um exemplo do efeito do zoneamento acústico pode ser observado abrindo e fechando as janelas do seu quarto. Ao abrir a janela, os sons da rua são claramente audíveis, e quando a janela é fechada, ou não são ouvidos de forma alguma, ou são extremamente abafados.

4.2 Tipos de defeitos no conteúdo sonoro

Há três categorias de defeitos de conteúdo sonoro:

- Sem som;
- Som incorreto;
- Reprodução de som incorreta.

Efeito sonoro ausente

Nenhum som é tocado quando deveria ser, para uma determinada ação do personagem ou em um determinado momento.

Reproduzindo o som errado

O som do objeto ou do ambiente, por exemplo, um tiro de canhão soa como um tiro de pistola, um barco a motor soa como um avião, ou personagens (dentro e fora do jogo) dizem as falas de outra pessoa.

O efeito sonoro viola a autenticidade histórica, por exemplo, os personagens do jogo utilizam vocabulário moderno, citações de filmes e músicas, o tanque T-34 soa como um tanque T-72. Raro, mas pode ser significativo se realismo e autenticidade forem declarados como um valor chave do jogo em teste.

Reprodução incorreta do som correto

Queda de áudio

Quando tal defeito ocorre, alguns dos sons reproduzidos podem desaparecer, como quando você está ao telefone com uma conexão ruim. Devido a isso, o jogador pode, por exemplo, entender mal o significado de uma frase em um diálogo. Se este for o único som que está tocando atualmente, então os testadores encontrarão facilmente o problema. Mas se vários sons são tocados ao mesmo tempo, e um deles desaparece, então é muito mais difícil detectar tal defeito.

Salto

Saltar o som de uma música ou efeitos de áudio pode ser associado a mais do que apenas danos ao arquivo de som em si. O salto é frequentemente causado por problemas de performance. Neste caso, eles ocorrem quando a taxa de quadros falha.

Distorção

Devido a problemas de performance, algumas frases podem soar distorcidas e inaudíveis.

Atraso de reprodução

Nos jogos, há momentos em que o projeto de som fica descompassado da animação, objeto ou situação de jogo exibida.

Exemplos de defeitos no conteúdo sonoro e suas possíveis causas

1. Falta de som do objeto/ambiente.

   Exemplo: Ao mover-se sobre uma superfície metálica, não há som de passos, o personagem se move silenciosamente.

   Causa: O desenvolvedor esqueceu de montar ou conectar o som nesta superfície.

2. O som do objeto ou do ambiente é muito alto ou silencioso demais.

   Exemplo: Um determinado objeto no local está queimando, mas o ruído do fogo é muito alto, ou pelo contrário, muito silencioso, o que parece irreal.

   Causa: O volume do efeito é definido de forma incorreta.

3. O som do objeto ou do ambiente é ajustado incorretamente.

   Exemplo: Ao dirigir um barco a motor sobre a água, o jogo toca o som de um carro de corrida.

   Causa: Um arquivo de som incorreto para o objeto foi escrito no código.

4. Posicionamento incorreto do som, dependendo de sua fonte.

   Exemplo: O jogo tem um ferreiro que continuamente golpeia sua espada com um martelo, mas o som do golpe é jogado em outro canto da sala.

   Causa: No editor de som, o efeito sonoro é colocado na posição errada.

5. Área de cobertura sonora configurada de forma incorreta.

   Exemplo: O som de um tiro é ouvido apenas dentro de um raio de alguns metros. Se o jogador se afasta mais, então o som não é ouvido de forma alguma, o que é irreal.

   Causa: O raio de som é ajustado incorretamente.

6. Distorção sonora.

   Exemplo: Os jogadores experimentam o som que começa a estalar e sibilar.

   Causa: Um arquivo de áudio corrompido está sendo usado.

7. Repetição contínua do som.

   Exemplo: O som é feito em loop e tocado continuamente.

   Causa: Um ponteiro quando o loop de som deve parar não está ajustado corretamente.

8. Distorção sonora na forma de reprodução de "gagueira":

   Exemplo: Distorção do som em movimento ocorre durante o movimento de um dos personagens do jogo.

   Causa: Um problema pode ocorrer devido a problemas com a montagem do som ou devido à instabilidade geral do cliente do jogo.

9. Atraso no tempo de reprodução de áudio.

   Exemplo: Durante uma *cutscene*, um personagem começa a falar, mas sua linha é reproduzida um pouco mais tarde do que quando seus lábios começam a se mover.

   Causa: a animação dos lábios do personagem em movimento está fora de sincronia com o início do arquivo de som correspondente.

4.3 Abordagens para testar o conteúdo sonoro em produtos de jogos

O projeto do som do jogo pode ser representado na forma de efeitos, vozes de personagens ou música. O teste de conteúdo do áudio do jogo pode ser realizado formalmente ou informalmente. As abordagens formais para testes de som incluem testes de presença, correção, volume e integridade de todos os arquivos de áudio. O teste

de som informal é uma espécie de avaliação do realismo ou da manutenção da atmosfera do jogo. Por exemplo, acompanhamento musical "pesado" ao local nos jogos do gênero horror para acentuar a situação e manter periodicamente o jogador em estado de tensão.

O testador deriva casos de teste da especificação previamente revisada de um objeto sonoro específico. A partir desta especificação, o testador deve compreender sem ambiguidade em que locais, cenas de jogo e situações o som deve ser reproduzido. Em alguns casos, isto pode exigir uma consulta com um projetista de áudio para aprender detalhes específicos sobre o conteúdo sonoro criado. Em outros casos, o testador pode usar o editor de recursos de jogo para testar a sincronização dos arquivos de som com as ações do jogador ou com o ambiente do jogo.

O teste do projeto de som do jogo por um testador pode envolver várias etapas.

4.3.1 Teste de conteúdo acústico

O testador ouve os sons e testa os parâmetros de áudio do conteúdo criado usando o editor de som ou o editor de recursos de jogo. Todos os arquivos de som devem ser testados.

Esta é uma das etapas mais extensas. As seguintes características sonoras são verificadas em relação aos requisitos:

- Apropriação ao objeto e ao cenário. Por exemplo, se o personagem estiver usando armadura de ferro, então os sons do metal devem dominar os sons do tecido à medida que ele se move. A explosão levemente audível de um bastão de dinamite ou de uma pedra caída, o evidente pisar no chão ao se movimentar na areia, pode perturbar a imersão no mundo virtual.
- Vozes para os atores. As vozes dos atores, em frases bem vocalizadas, devem corresponder aos personagens que as falam e ao cenário do jogo. Normalmente, a voz deve corresponder ao sexo e à idade do ator. É feito para manter a atmosfera geral, por exemplo, o personagem sulista tem sotaque sulista, e não nordestino.
- Não incomoda. Amostras de vários sons juntos podem às vezes se tornar desagradáveis para o ouvido humano.
- Nível de volume. O volume para cada arquivo de som deve ser o mesmo ao usar vários arquivos dentro de um jogo.
- Efeitos sonoros. Os efeitos aplicados tanto aos sons como à locução também devem ser usados corretamente. Se o local ou a cena é preenchido com vários detalhes, então os detalhes são enfatizados pela atuação da voz. Em um escritório grande uma pessoa pode ouvir o som das unidades do sistema, o ruído das impressoras, o ruído dos aparelhos de ar-condicionado, a pressão dos teclados e as conversas dos trabalhadores.

A dublagem cuidadosa de todos os objetos contribui para a criação de uma imagem real. Por exemplo, quando o jogador se encontra na aldeia, ele não ouvirá necessariamente o mugido das vacas, mas provavelmente esperará os sons característicos de uma serraria se aparecer um na frente deles. Um pequeno riacho na floresta pode muito bem não ter desenho sonoro, mas se o herói estiver no local próximo a uma cachoeira, a falta de ruído de água só pode ser explicada por um defeito (ou som desligado nos ambientes).

4.3.2 Testando a mistura de sons de música e jogos

O tamanho do projeto e as habilidades profissionais da equipe de desenvolvimento variam de projeto para projeto, o que torna impossível separar claramente as funções desempenhadas por funções. Pode haver situações em que não é necessário um ator ter uma fala, o trabalho de um artista de *foley* (um especialista em criar ruído) é confiado a um projetista de áudio, ou mesmo um próprio especialista cria ou compra amostras de ruídos prontas, as vincula no cliente e testa o som resultante.

O papel de um projetista de áudio é criar sons e gravar arquivos de áudio. Estes arquivos de áudio devem ser testados para diferentes usos. O sistema de som de um fliperama é surpreendentemente diferente do de um telefone celular - e o áudio deve ser testado. Os jogos de console devem ser reproduzidos corretamente tanto em pequenos alto-falantes de TV, quanto em sistemas de *home-theater*.

Mesmo um som de alta qualidade e aparentemente apropriado pode causar confusão e irritação ao jogador, se a mistura final não for feita corretamente. Por exemplo, o som da ação dura muito mais do que a própria ação, ou seu volume está fora do lugar.

O testador testa a sonoridade finalizada e a exatidão do ajuste de novas alterações sonoras no cliente, usando o editor de recursos integrado ao motor do jogo, ou com base em sua própria experiência, ou na documentação fornecida.

Como durante o jogo o jogador ouve muitos sons vindos de diferentes objetos e música tocando ao fundo, é importante que todos os componentes de áudio estejam em harmonia uns com os outros. O testador deve avaliar o componente sonoro do projeto e identificar possíveis desvios, pontos irreais em relação ao volume, juntamente com a qualidade e naturalidade do som baseado no senso comum. Também é importante avaliar a precisão da regionalização da fonte sonora em relação aos objetos em si. Por exemplo, no jogo, quando um personagem abre uma porta, o jogador ouve o som da abertura da porta, não a partir da porta à sua frente, mas erroneamente por trás das costas do personagem. Também é importante que o testador possa avaliar corretamente a pontualidade dos sons em relação ao que está acontecendo na tela.

4.3.3 Testando a composição musical

A composição ou "*mixagem*" é o estágio de criação de uma gravação final a partir de faixas gravadas individualmente. Este é o próximo passo após a gravação de áudio que consiste em selecionar e editar (às vezes restaurando) as faixas originais gravadas, combinando-as em um único projeto e processando-as com efeitos. A edição é frequentemente uma etapa de trabalho autônoma.

A mistura em jogos que utilizam música eletrônica é o próximo passo após a criação do conteúdo de áudio. A etapa de gravação de som de um projeto eletrônico está na maioria das vezes ausente. A linha entre a criação e a *mixagem* de música eletrônica é indefinida; o projeto chega à *mixagem* já parcialmente misturada, porque muitos sintetizadores virtuais já possuem processamento de modelos de sons diferentes.

Como resultado, um projeto multicanal é produzido em um fonograma monofônico, estéreo ou multicanal, que geralmente recebe sua forma final em um processo chamado *masterização*.

Após a masterização estar completa, começa o teste da música masterizada. O testador deve testar o componente musical quanto a defeitos, descontinuidades sonoras e interferências e relatar quaisquer falhas encontradas.

4.4 Execução de testes de som

4.4.1 Níveis de teste de conteúdo sonoro durante o ciclo de vida de desenvolvimento de software de jogos

Frequentemente, os testes de som começam tarde no desenvolvimento. Enquanto um desenvolvedor cria conteúdo para um jogo (modelos de objetos/personagens, mapas, itens etc.), o componente sonoro é trabalhado em um estágio posterior. No estágio de desenvolvimento um objeto pode não ter som, mas um objeto em si já tem um modelo acabado com animação personalizada integrada ao cliente e pronto para testes.

As amostras de som devem ser testadas previamente pelos especialistas que criaram o som (ou seja, projetistas de áudio, artista de *foley*, ou especialistas em efeitos sonoros). Eles podem participar dos testes de sons individuais e depois enviá-los ao desenvolvimento para maior integração com o cliente.

4.4.2 Integrando sons no jogo

Após integrar o projeto de som no jogo, o testador avalia as seguintes características dos sons:

- Habilidade de ligar e desligar sons e/ou música;
- Capacidade de alterar o volume;
- Conformidade com um formato único de nomes de arquivos de áudio e sua correta distribuição em pastas no projeto;
- Regionalização das fontes sonoras, zonas de distância de propagação de som no mapa do jogo;
- Performance do sistema com sons integrados;
- Compatibilidade dos sons em diferentes sistemas de áudio: fones de ouvido, alto-falantes, alto-falantes de monitor.

4.4.3 Áreas de responsabilidade das pessoas envolvidas

O processo de criação de música e design de som no projeto envolve vários tipos diferentes de especialistas, desempenhando certas funções:

Papel	Responsabilidades
Artista de foley	Torna os sons que existem na realidade mais vivos e ricos. É responsável pela criação de efeitos sonoros que não existem na realidade. Ao combinar os sons de objetos comuns, eles gravam amostras e criam a partir delas os efeitos de, por exemplo, o som de uma espada laser, um rosnado de zumbi, ou uma batida de porta.
Ator de voz	Usa sua voz para dar vida a um personagem.
Compositor	Envolvido na criação de sons e músicas para o jogo.
Desenvolvedor	Integra o conteúdo de áudio criado e personalizado ao código do cliente do jogo.
Engenheiro de áudio	Envolvido na operação de gravação e amplificação de som, e equipamento de radiodifusão.
Engenheiro de som	Especialista que possui a máxima experiência sonora e determina a sonoridade final de cada objeto, de cena, e de todo o jogo.
Projetista de áudio	Cria conteúdo a partir dos arquivos de áudio originais com som gravado (samples) e o ajusta no editor de som.
Projetista de jogos	Envolvido na criação da documentação do jogo (p. ex., desenho de jogo). Este documento descreve as regras e características do jogo em linguagem simples. Assim, mesmo antes do motor ser desenvolvido, o projetista de jogo desenvolve uma visão holística dele. Durante o desenvolvimento, também participam parcialmente do papel de "consultor", aprovando a conformidade das propostas e mudanças da ideia principal do projeto. Também participam pessoalmente dos testes do jogo.
Projetista de nível	Distribui as fontes sonoras por nível e depois vincula os arquivos de som às fontes sonoras.
Projetista narrativo	Especialista que é responsável pela história e narrativa no desenvolvimento do jogo. Trabalha no som e na música tocada durante momentos relacionados ao desenvolvimento do enredo (*cutscenes*, introduções, diálogos, momentos significativos no jogo etc.).
Testador	Testar a sonoridade finalizada e a correção de novas mudanças de som no cliente.

4.4.4 Procedimentos e abordagens na condução das atividades de teste em testes de objetos sonoros

Ao testar objetos sonoros, o testador deve implementar os procedimentos e ações necessárias para obter informações completas sobre como o som é configurado corretamente em uma determinada versão do jogo. A lista de ações depende diretamente de quais objetos precisam ser testados. Por exemplo, para o teste final do projeto sonoro das armas adicionadas no cliente, o testador executa uma série de ações: adicionar armas caso elas não estejam inicialmente disponíveis no cliente, carregar novo conteúdo no mapa de teste e toda a lista de ações necessárias para testar ao máximo todos os sons ligados à arma (atirar, carregar, recarregar, clicar para mudar o modo de tiro etc.)

É importante entender que a lista de procedimentos e abordagens para cada objeto é diferente. Por exemplo, se um objeto do ambiente é testado e não o objeto com o qual o personagem interage, então o testador testa a inclusão do projeto sonoro em si - por exemplo, o murmúrio de um riacho ou rio quando o personagem se aproxima. O testador deve entender onde encontrar informações sobre todos os sons disponíveis para um determinado objeto. Ele também deve aproveitar o editor de recursos embutido e o editor de sons para testar a integridade do conteúdo sonoro e sincronizá-lo com as ações do personagem e o ambiente do jogo.

4.5 Ferramentas de suporte para testes de som

Editor de áudio do motor de jogo

O testador nem sempre tem acesso ao editor de áudio do motor de jogo. Portanto, enquanto se testa um conjunto padrão de software para validação de conteúdo é usado. Tal editor é usado por testadores que trabalham dentro da equipe de desenvolvimento (se houver), ou por especialistas que trabalham diretamente no editor de áudio (p. ex., compositores, projetistas de áudio etc.). Os projetistas de áudio têm que participar dos testes. Eles estão testando os ativos, incluindo os níveis de equilíbrio, ajustando os equalizadores e misturando/alinhando todos os ativos tanto juntos como individualmente. Assim que os recursos forem adicionados à versão de trabalho do jogo, os projetistas de áudio deverão ser capazes de obtê-los facilmente e editá-los, se necessário.

Para facilitar a manutenção do conteúdo sonoro, todas as amostras de som devem estar em trilhas sonoras diferentes, dispostas em pastas, nomeadas, com selo de data/hora e versão. Também é eficaz converter faixas de áudio em formato digital comprimido, de modo a não sobrecarregar os recursos de hardware do adaptador de som.

Editor de mapa/regionalização

Às vezes o testador tem acesso ao software de desenvolvimento, na maioria das vezes ao editor de mapas/locais. Isto permite ao testador expandir suas capacidades ao testar o conteúdo de áudio e todo o conteúdo que está no cliente. Os desenvolvedores de jogos adicionam funcionalidade aos editores de mapas que suportam a jogabilidade "simulada" diretamente no mapa de nível. Assim, o testador pode realizar os testes necessários sem mudar do editor de mapas para o próprio cliente do jogo, o que economiza significativamente o tempo.

A maioria dos efeitos sonoros e fontes sonoras disponíveis no mapa são dispostos e ajustados usando o editor de nível. No editor de mapas, o testador pode ver a regionalização das fontes sonoras nos locais do cliente, amarrado a diferentes objetos no local, ou caracteres. Assim, quando o especialista testa a fonte sonora através do editor, ele pode testar o som que está sendo reproduzido e a distância de propagação do som, seja exibida como uma esfera, ou como "zonas" onde o testador ou usuário final começa a ouvir o som/música que está sendo reproduzido, vindo de uma fonte sonora amarrada, ao cruzar seus limites.

Se a propagação do som for implementada na forma de "zonas", o testador testa a reprodução do som da fonte ao cruzar o limite da "zona". Também é relevante testar o aumento de volume ao se aproximar da fonte sonora para tornar o som amarrado mais realista. O teste da distância de propagação do som na forma de uma esfera difere apenas na forma da zona de propagação. O editor também permite testar a ligação, a regionalização e, respectivamente, a configuração da fonte sonora de cada cliente.

Com estas informações, o testador pode avaliar o realismo do conteúdo sonoro, a distância de propagação do som, a correção dos sons (sobre o local, objetos, caracteres) e indicar ao desenvolvedor defeitos em caso de detecção. Por exemplo, criar uma descrição de um defeito sobre uma distância definida incorretamente em uma determinada fonte sonora, ou o volume de um determinado som (referindo-se a uma determinada fonte sonora) é excessivamente alto contra o fundo das fontes "vizinhas". Na descrição dos defeitos, o testador, trabalhando através do editor, pode facilitar o trabalho dos desenvolvedores, tornando mais fácil encontrar áreas problemáticas ao apontar um problema em particular.

5 Teste de nível de jogo - 65 min

Palavras-chave

teste do jogo

Palavras-chave específicas

código de cores, nível de jogo, lugares inacessíveis, editor de nível, protótipo de nível, narrativa, geometria estrutural, gatilho

Objetivos de aprendizagem

5.1 Princípios e conceitos de projeto de nível de jogo

GaMe-5.1.1 (K1) Recordar os componentes do nível de jogo

GaMe-5.1.2 (K2) Classificar os defeitos típicos dos níveis de jogo

5.2 Etapas e execução de testes de nível de jogo

GaMe-5.2.1 (K2) Resumir os testes realizados em vários estágios da criação dos níveis de jogo

GaMe-5.2.2 (K2) Comparar as áreas de responsabilidade dos especialistas que participam nos testes de nível de jogo

5.3 Ferramentas de suporte para testes de nível de jogo

GaMe-5.3.1 (K2) Resumir o uso de ferramentas para testar os níveis de jogo

5.1 Princípios e conceitos de projeto de nível de jogo

5.1.1 O termo "nível" e sua especificidade, dependendo do gênero do projeto do jogo

O nível de jogo é uma área separada do mundo virtual do jogo, na qual o jogador deve completar uma determinada tarefa: encontrar um tesouro, derrotar todos os adversários ou simplesmente chegar à saída. Uma tarefa geral necessária para terminar um nível está intrinsicamente ligada à regionalização e muitas vezes consiste em diversas pequenas tarefas e missões necessárias para terminar um nível ou missões secundárias, que não são necessárias para terminar um nível. Em muitos motores de jogo, o nível é dividido pela tela de carregamento.

O projeto de níveis é o estágio de desenvolvimento do jogo associado à criação de mapas, locais, tarefas, missões e outros ambientes para os níveis do jogo. O projeto de níveis inclui a aparência dos objetos e mecânica de jogo, obstáculos no caminho do jogador, o enredo do jogo e outros elementos que, coletivamente, criam a experiência pretendida de jogo.

Para criar níveis, geralmente é usado um software especial - um editor de níveis.

Os níveis de jogo consistem em muitos componentes conectados uns aos outros em um único todo. Entre as partes principais estão as seguintes:

Geometria estrutural

A geometria estrutural, ou geometria de nível básico, é um dos elementos mais importantes e formativos de qualquer espaço de jogo. É a geometria estrutural que define o relevo do terreno e a superfície sobre a qual os personagens do jogo podem se mover.

A geometria estrutural também limita o espaço de nível disponível para que os caracteres se movimentem. Por exemplo, pode haver montanhas intransitáveis ou outras características do terreno nos limites de um nível.

Se um nível estiver claramente marcando pontos de entrada e saída, então a geometria estrutural pode fornecer orientação sobre onde o jogador deve ir para chegar ao final do nível.

Dependendo do local onde o jogo acontece, vários elementos podem atuar como objetos que formam o nível. Por exemplo, na cidade: são edifícios, ruas, pontes e passagens inferiores; fora dela, elementos da paisagem natural: campos, colinas, rochedos, vales e cavernas.

Ambiente do jogo

Os objetos do ambiente de jogo são usados pelo projetista de nível para criar uma imagem mais crível do espaço necessário de jogo.

Por exemplo, uma rua de aldeia é composta por casas, prédios e estradas, por isso o projetista a enche com objetos de diferentes tamanhos. Distinguem-se objetos grandes como - árvores, cercas, carrinhos, objetos de médio porte - poços, barris, bancos, assim como objetos pequenos, como plantas no jardim, pedras etc.

Iluminação

O uso de vários tipos de fontes de luz permite a atmosfera desejada no nível do jogo. Por exemplo, ela direciona o jogador na direção certa, atrai sua atenção ou o esconde dos adversários.

Acompanhamento de som

Trilha sonora de fundo ou efeitos sonoros reproduzidos para várias ações ou eventos. Leia mais na seção "4. Teste de som".

Funcionalidade do jogo

A funcionalidade do jogo inclui um sistema de configurações para organizar a jogabilidade em um nível específico. Os seguintes tipos de configurações são diferenciados:

Configurações relacionadas com a passagem do nível. Estas incluem:

- pontos de aparência dos personagens e seus oponentes;
- condições de vitória e derrota;
- pontos de salvamento automático do jogo;

- um sistema de gatilhos que lança certos eventos de jogo, ou roteiros. Por exemplo, a cada dois minutos um avião sobrevoa o campo de batalha, ou quando um personagem entra em um prédio, um vídeo da história é reproduzido, ou quando um soldado cruza a fronteira de uma área protegida, o alarme é disparado.

Configurações de objetos com os quais o jogador pode interagir, por exemplo, portas, *pickups*, objetos destrutíveis, armadilhas etc.

Configurações da concha física de nível que impedem que o personagem saia do nível ou fique preso em sua geometria.

Como regra, as configurações da funcionalidade do jogo são escondidas dos jogadores e são visíveis apenas no editor de níveis.

5.1.2 Entendendo os tipos de defeitos no projeto de nível

Há uma série de defeitos que são mais comuns na criação de níveis. O rastreamento de tais defeitos podem ser difícil, de modo que eles podem permanecer despercebidos no nível por algum tempo.

Algumas ferramentas dedicadas à criação de mapas têm a funcionalidade integrada para detectar tais defeitos nos mapas. Os projetistas de nível frequentemente utilizam essas ferramentas nos estágios finais de criação de um mapa ou nível. Mas, na maioria dos casos, a melhor maneira de encontrar defeitos em um mapa é testá-lo por jogadores experientes que são capazes de detectar o problema e relatá-lo.

Uma proporção significativa dos defeitos encontrados nos níveis está relacionada com a aparência e regionalização dos objetos de jogo, sua iluminação, modelo de colisão etc. Uma descrição detalhada de tais defeitos é dada na seção "3. Testes gráficos".

No entanto, os seguintes defeitos estão diretamente relacionados com o projeto dos níveis.

Geometria

Os defeitos de geometria são situações em que o personagem do jogador está parcialmente preso em algumas partes do nível ou mapa. Como regra, tais defeitos são encontrados nas bordas do mapa, saliências, declives. O jogador guia seu personagem em um objeto ou caminha perto de um objeto e fica preso dentro dele. Muitas vezes, com a ajuda de saltos, flexões e outros movimentos, o personagem consegue sair de tal situação, mas em alguns casos isso não ajuda. Nesses casos, o jogador tem que reiniciar o nível ou mapa, retornar ao último ponto de salvamento do progresso ou usar uma combinação especial de botões/chaves para se mover para o local mais próximo sem defeitos de colisão.

Por exemplo, um personagem de jogo pode ficar preso na geometria do mapa, incapaz de sair, ou em áreas que deveriam ter acesso negado. É especialmente importante identificar tais defeitos em jogos *multiplayer*, onde defeitos de nível podem colocar um jogador ou uma equipe em uma posição melhor.

Outra situação comum é a morte ao ficar preso, ou seja, um personagem despenca de uma superfície do nível / mapa. Por fim, o personagem ou morre e o nível tem que começar novamente, ou cai sem infinitamente, e o jogo tem que ser reiniciado.

Lugares inacessíveis

Nos jogos, especialmente em mapas em jogos multiplayer, pode haver locais intencionais nos quais o jogador obtém uma posição melhor do que os outros jogadores. Por exemplo, um atirador em uma colina terá um grande ângulo de tiro. Embora o jogador que for o primeiro a tomar tal posição ganhe alguma vantagem sobre os adversários, esta é uma intenção do desenvolvedor, um elemento de projeto do nível, e não um defeito.

Mas há situações em que tal lugar é criado por acidente, devido a um erro do projetista de nível. De acordo com o conceito do desenvolvedor, esta área deve ser inacessível aos jogadores. Entretanto, um certo personagem de jogo com a ajuda de suas características únicas (p. ex., saltos mais altos do que os demais) pode acabar nele. Desta forma, o jogador pode obter uma vantagem de jogo ilegal sobre seus rivais. Por exemplo, outros jogadores não serão capazes de detectá-lo, danificá-lo etc.

Jogabilidade complexa

Problemas associados à falta de indicadores de metas ou à não-obviedade das ações necessárias para completar o nível. Por exemplo, o jogador derrotou todos os adversários acessíveis no território, mas não consegue entender o que mais precisa ser feito para ir além.

Nível de equilíbrio do jogo

Muitos defeitos surgem na definição do equilíbrio de nível. Por exemplo, oponentes de ***bot*** que são muito complexos no estágio atual do jogo, e que excluem completamente a possibilidade de passar este nível. Em outros jogos, várias equipes tentam chegar à pontuação desejada no mapa antes dos oponentes. Aqui, o defeito no equilíbrio do nível pode ser condições de partida desiguais para as equipes.

Restrições e convenções inapropriadas

Estes problemas não afetam a jogabilidade e não proporcionam uma vantagem no jogo, mas degradam a percepção e a imersão no jogo.

Por exemplo, o jogador vê que o caminho do personagem é bloqueado por uma cerca na altura da cintura, mas é impossível ultrapassá-lo. Como planejado pelo projetista de nível, este lugar é uma área inacessível para o jogador, e é assim que deve ser. Mas do ponto de vista do jogador, esta situação parece ser um defeito que arruína a imersão no jogo. Da mesma forma, o jogador considerará como um defeito uma situação em que o personagem está trancado atrás das grades e não pode sair, embora ele veja que a distância entre as varas é suficientemente grande.

Narrativa

Defeitos de nível associados a uma violação do estilo geral de contar histórias, o enredo do jogo. O impacto de tais defeitos é mínimo na própria jogabilidade do jogo, entretanto, eles afetam negativamente a imersão do usuário e a percepção geral do jogo. Por exemplo, se um personagem se encontrar em uma cidade que, de acordo com a trama, foi submetida a um bombardeio nuclear, a aparência de edifícios intactos na mesma causará dissonância.

Alteração da codificação por cores dos objetos

Os objetos encontrados pelo usuário durante o jogo têm propriedades diferentes. Alguns deles são decorações, e com outros o jogador poderá interagir. Objetos codificados por cores com diferentes propriedades são uma parte importante do projeto de nível. Portanto, é desejável que uma vez adotado o código de cores não mude durante o jogo.

Por exemplo, um jogador descobriu um barril vermelho no nível, que explodiu de um tiro. Se um jogador se deparar novamente com tal barril, ele esperará o mesmo comportamento dele. Portanto, todos os barris que explodem com um tiro devem sempre ser da mesma cor (geralmente vermelho).

Caixas que o jogador pode quebrar, portas que o jogador pode abrir, pedras e bordas que o jogador pode agarrar e subir etc., devem ser diferentes em forma e/ou cor de outros objetos.

Caso contrário, o jogador será forçado a passar mais tempo entendendo o que fazer a seguir.

5.2 Etapas e execução de testes de nível de jogo

5.2.1 Etapas básicas de projeto e teste de nível de jogo

O nível de jogo é uma entidade que combina mecânica de jogo, objetos visuais, trilha sonora, inteligência artificial e outros componentes. As abordagens de teste para cada um destes componentes são discutidas em detalhes nas seções relevantes deste programa de estudos.

O teste de nível de jogo envolve testar todos esses componentes juntos e é semelhante ao teste de sistema, quando todos os componentes são considerados e testados como um todo.

O teste dos níveis de jogo começa a partir das primeiras etapas de sua criação. Os objetos dos testes, a natureza dos testes e o procedimento para sua implementação, assim como as áreas de responsabilidade de vários especialistas dependem do estágio atual de desenvolvimento do nível de jogo.

Considere os principais estágios de projeto e teste de níveis usando o exemplo de um jogo 3D.

Prototipagem de nível de jogo (projetista de block out/Caixa Cinza)

Nesta fase, baseado em um esboço previamente desenvolvido, um modelo 3D do nível futuro do jogo é criado a partir de objetos especiais. São utilizados apenas elementos que afetam diretamente a jogabilidade do jogo: cubos cinzas, esferas, cilindros e planos, que esquematicamente formam o layout do nível do jogo.

Tais formas geométricas monocromáticas e não chanfradas permitem que a estrutura do nível seja rapidamente alterada, como para remover ou adicionar pedaços enormes de regionalização e, se necessário, apagar tudo de forma limpa e recomeçar. Às vezes, nesta fase, nem toda a mecânica de jogo está pronta para uso, portanto, o projetista tem que implementá-la gradualmente, adaptando o nível às condições de desenvolvimento em constante mudança.

O layout é criado levando-se em conta todas as proporções e escalas. Para isso, o projetista de nível, juntamente com o projetista do jogo, forma um conjunto de métricas - os tamanhos e parâmetros que o personagem do jogo possui. Assim, o projetista de jogo fornece informações sobre a distância e altura do salto do personagem, sobre quais posições o personagem pode estar (p. ex., de pé em altura total, agachado, deitado) e como isto muda o tamanho da figura, o ângulo máximo de inclinação da superfície ao longo da qual o personagem pode se mover sem deslizar. Estas métricas afetam diretamente as dimensões dos objetos e elementos do ambiente do nível.

Após o projetista de nível ter decidido o tamanho do nível e definido a escala correta para os objetos, outras funcionalidades são configuradas para suportar a jogabilidade no nível. Isto inclui os pontos de aparência dos personagens, pontos de operação (p. ex., gatilhos) dos eventos do jogo etc.

O desenvolvimento do nível de jogo nesta fase, como regra, ocorre em várias iterações, durante as quais os testes ajudam a identificar e eliminar elementos problemáticos.

Teste de protótipo de nível de jogo

Devido aos requisitos especificados não serem finalizados na fase de protótipo, os testes exploratórios e outras técnicas de teste baseadas em experiência são considerados os mais eficazes.

No layout criado, os primeiros testes de jogos são realizados para testar a jogabilidade no mapa. Os participantes do teste que realizam os primeiros testes do jogo podem ser tanto testadores da equipe de desenvolvimento de software do jogo quanto testadores terceirizados. Recebendo feedback dos participantes do teste, o desenvolvedor pode tirar uma conclusão sobre soluções bem e malsucedidas e, se necessário, alterar, mover ou remover objetos no layout.

Além disso, através da realização de testes do jogo no protótipo, a mecânica de jogo é testada e como eles são combinados com a disposição e o tamanho dos objetos no modelo.

Por exemplo, que os abrigos localizados no nível são suficientemente grandes para que o personagem se esconda atrás deles dos inimigos, que a largura da fenda na superfície permite que o personagem pule sobre ela, ou que o personagem possa escalar todas as bordas necessárias e não bata com a cabeça em tetos muito baixos.

Individualmente, a complexidade desejada do jogo - criada pelo layout do nível - e a correta sensação de jogabilidade, em geral devem ser testadas. Por exemplo, se for assumido que em um determinado intervalo do jogo deve haver um tiroteio com muitos adversários, então o nível deve ter um número suficiente de abrigos. Se for suposto lutar contra um inimigo forte e tenaz (chefe), então o espaço do nível é limitado e livre de objetos desnecessários.

Prototipagem de Geometria (artista de block out/caixa-branca)

Os modelos 3D temporários criados pelo artista de ambiente substituem os objetos dos quais o projetista de nível fez o protótipo, e objetos adicionais são adicionados. A tarefa principal é trabalhar o componente visual do nível sem alterar a jogabilidade em si.

No entanto, tal substituição pode causar defeitos no nível que complica a jogabilidade. Por exemplo, um modelo temporário de árvore sobrepõe a saída do nível. Neste caso, o nível novamente vai para as mãos do projetista do nível. Agora é necessário testar novamente se a mecânica está funcionando corretamente, se o jogador pode ir a qualquer lugar e se ele vê tudo o que precisa.

Testando o protótipo de geometria

Os testes nesta fase são realizados para garantir que após o desenho artístico no nível não haja defeitos que impeçam a jogabilidade.

Exemplos de tais defeitos podem ser um modelo 3D temporário de uma pedra bloqueando a passagem de nível, ou uma copa de árvore bloqueando a visão do jogador.

Para identificar tais problemas, são novamente usados testes do jogo, onde a mecânica de jogo é testada para validar que o tamanho e a regionalização dos modelos de objetos correspondem às métricas exigidas e à exatidão da jogabilidade.

Paralelamente, um teste de aparência dos objetos, iluminação, ambiente etc. pode ser realizado (mais detalhes são dados no capítulo "3. Testes gráficos").

Fazendo a versão final

Nesta etapa, o projetista de nível integra o software de toda a equipe de desenvolvimento em um único todo. A partir de todo o conteúdo criado, ele coleta um único espaço de jogo. Nesta etapa, os detalhes decorativos são adicionados ao nível, os objetos se tornam similares aos reais na aparência, finalidade e regionalização. Por exemplo, papéis, papelaria, monitores de computador aparecem sobre a mesa, que o jogador pode esmagar e jogar fora da mesa.

Testando a versão final

Nesta fase, os testes de performance e compatibilidade são realizados. A performance do nível é testada em diferentes dispositivos, o número de quadros por segundo é medido, a qualidade de exibir modelos visuais de objetos a diferentes distâncias, a iluminação e as sombras são testadas.

Como parte dos testes, os testadores testam a presença de modelos de colisão para todos os objetos de jogo, a correspondência de modelos físicos e visuais de objetos etc.

Por exemplo, o teste de modelos de colisão procede da seguinte forma: o testador leva o personagem do jogo para um objeto do jogo, por exemplo, para uma pedra grande, e tenta "caminhar sobre a pedra" de diferentes lados. Com um modelo de colisão devidamente configurado com pedras, o corpo do personagem só tocará visualmente a pedra, mas não a atravessará ou entrará completamente dentro dela.

5.2.2 Áreas de responsabilidade das pessoas envolvidas

Vários especialistas trabalham na criação de um nível de jogo ou mapa, e muitas vezes estas tarefas são realizadas em paralelo, o que leva ao aparecimento de vários defeitos.

Considere as tarefas que são de responsabilidade desses especialistas do ponto de vista da obtenção de um produto de qualidade:

Papel	Responsabilidade
Projetista de nível	Responsável pela geometria e forma do nível, a regionalização dos objetos e pontos de tiro, o tamanho dos abrigos etc. Tudo isso deve apoiar a jogabilidade principal do jogo e promover o envolvimento do usuário no jogo
Artista	De um ponto de vista artístico, todos os objetos no mapa devem ter as texturas, a iluminação e outros efeitos visuais corretos. Isto se aplica tanto à aparência (colocação correta de áreas sombreadas e iluminadas) quanto ao realismo da imagem. Por exemplo, uma cabana na floresta feita de blocos de concreto teria um aspecto estranho e fora do lugar.
Testador	Os testadores, como regra, trabalham com as versões finais de níveis e mapas, quando todos os objetos já foram colocados e configurados. A tarefa principal de um testador é testar se um jogador pode jogar em um determinado mapa sem encontrar defeitos técnicos e artísticos. A interação com todos os objetos do mapa é testada, a saber: • Os objetos têm um modelo físico (modelo de colisão); • Ausência de obstáculo invisível; • Impossibilidade de sair do mapa.

5.3 Ferramentas de suporte para testes de nível de jogo

Várias ferramentas são usadas para facilitar os testes de níveis ou mapas, incluindo editores 3D, editores de nível e motores de jogo.

No entanto, os testadores utilizam as mesmas ferramentas que os desenvolvedores que criaram o nível.

O mecanismo de jogo geralmente tem funcionalidade que permite a personalização do mapa e dos objetos nele contidos para vários testes e busca de defeitos. O motor desenvolvido internamente pode ser alterado a pedido da equipe de testes, a fim de aumentar a eficácia dos testes.

Por exemplo, como resultado de mudanças na geometria da superfície plana, alguns objetos podem estar suspensos no ar ou encrustados na superfície. Tais defeitos podem ser descobertos pela inspeção visual de todos os objetos, ou, se o editor de mapas permitir, um algoritmo especial pode simplificar o teste.

Além disso, ao desativar a exibição de texturas de vários objetos no editor, o testador pode encontrar lugares intransitáveis no mapa, ou vice-versa, um caminho viável não intencional pelo projetista do mapa. Isto é especialmente importante para jogos multiplayer, porque em alguns casos, isto pode dar a alguns jogadores uma vantagem indesejada sobre outros.

6 Teste de controladores de jogo - 95 min

Palavras-chave

conformidade, testes ergonômicos, testes funcionais

Palavras-chave específicas

acelerômetro, controlador de jogo, gamepad, giroscópio, volante e pedal de corrida, touchscreen, trackball

Objetivos de aprendizagem

6.1 Princípios e conceitos dos controladores de jogo

GaMe-6.1.1 (K2) Classificar os típicos e os especializados dispositivos de entrada

GaMe-6.1.2 (K2) Dar exemplos de diferentes dispositivos de entrada em termos de sua aplicação

GaMe-6.1.3 (K1) Chamar diferentes tipos de controladores de jogo

GaMe-6.1.4 (K2) Classificar os defeitos em um produto de jogo relacionados com as especificidades dos controladores de jogo, e possíveis causas de sua ocorrência

6.2 Abordagens para testar controladores de jogos

GaMe-6.2.1 (K2) Dar exemplos de condições de teste a serem cobertas ao se testar controladores de jogo

GaMe-6.2.2 (K2) Classificar as tarefas para especialistas em UX, testadores e projetistas de jogos durante o teste de jogos

6.3 Suporte a ferramentas para testes de controladores de jogo

GaMe-6.3.1(K2) Resumir o uso de ferramentas para testar o comportamento dos controladores de jogo

6.1 Princípios e conceitos dos controladores de jogo

6.1.1 Tipos de controladores de jogo

Um controlador de jogo é um dispositivo de entrada que é conectado a um console de jogos ou a um computador pessoal. Usando o controlador de jogo, o jogador controla o movimento e as ações dos elementos do jogo. Neste caso, os tipos de elementos dependem do próprio jogo, mas em sua maioria será um dos personagens do jogo.

Dispositivos de entrada típicos

Os dispositivos para jogos, tais como telefones, PCs, consoles e caça-níqueis são garantidos para permitir o uso de um dos seguintes dispositivos.

Dispositivo de entrada	Descrição
Gamepad	Este é o principal dispositivo de entrada nos consoles de jogo
Teclado e Mouse	Como estes dispositivos se tornaram dispositivos de entrada comuns para computadores pessoais, eles também são comumente usados para jogos de computador. Alguns consoles de jogos também permitem que um mouse e um teclado sejam conectados e controlados nos jogos.
Volante/Pedal	É um controlador que possui um elemento de rotação e um ou mais pedais de ação
Trackball	Parece uma bola meio protuberante da base; a bola é acionada (girada) passando-se a palma da mão sobre ela
Tela sensível ao toque	Usados em telefones, PDAs, consoles portáteis e máquinas caça-níqueis modernas. Alguns jogos têm elementos de jogabilidade especificamente voltados para a tela sensível ao toque

Dispositivos especializados são dispositivos focados em certos tipos de jogos.

Dispositivos específicos	Descrição
Joystick	Este era originalmente um dispositivo universal de jogo. Um teclado, mouse ou *gamepad* foi considerado preferível em jogos de ritmo rápido, e o joystick se tornou um dispositivo especializado para jogos no gênero simulador de voo. Entretanto, por hábito, os *gamepads* são frequentemente chamados de "joysticks".
Gamepad de corrida	Usado para simplificar o jogo de corridas no console. Na verdade, este é um *gamepad* normal, que tem um eixo de direção adicional incorporado nele (e às vezes botões de aceleração e freio analógicos). Tal dispositivo é muito mais barato do que um volante/pedal de corrida.
Volante de voo	Para simuladores de voo civis (os simuladores de voo militares usam um *joystick*).
Pedais	Para simuladores de carro e de voo que têm projetos fundamentalmente diferentes.
Acelerador de voo	Também conhecido como alavanca de controle de potência. Utilizada para simuladores de voo.
Alavanca de comando	Para simuladores de direção.
Pistola de luz	Para atirar objetos na tela.
Graphics tablet	Usado para controlar o cursor em vez do mouse (touchpad).
Controladores de jogos musicais	Usado em jogos de música para simular instrumentos musicais como guitarras, bateria, ou um console de DJ.

Dispositivos específicos	Descrição
Plataforma de dança	Uma pista de dança (como nas *slot-machines*). É uma plataforma com vários botões que podem ser pisados. A jogabilidade de tais jogos é pisar na sequência necessária de botões para se assemelhar a uma dança.
Teclado de jogos	Este é um teclado especializado com botões que estão localizados de acordo com as especificidades de um jogo em particular, e botões adicionais para a criação de macros.
Controlador de pesca Microfone	Para jogos de simulação de pesca. Um microfone ou *headset* é usado como um dispositivo de entrada adicional, com o qual são dados comandos ao jogo, de modo que os personagens e os jogadores possam se comunicar.
Controlador ferroviário	Um simulador de um painel de controle para ferrovias, onde é controlado os materiais circulantes de tração ferroviária/elétrica.

Tecnologias de captura de movimento

- Desde o início dos anos 2000, sistemas de controle de cabeça têm sido usados para jogar simuladores de voo e para pessoas com mobilidade reduzida;
- Dispositivos de controle remoto que rastreiam sua posição no espaço usando sensores IR e acelerômetros;
- Câmeras que rastreiam as imagens do controlador no espaço 3D e reconhecem seus movimentos;
- Dispositivos que permitem o uso de comandos verbais, posturas corporais e de objetos ou imagens exibidas.

6.1.2 Defeitos relacionados com as especificidades dos controladores de jogo

As causas dos defeitos associados aos controladores podem ser diferentes. A aparência dos defeitos pode ser causada pelo próprio software, pelo casamento dos componentes do controlador e até mesmo pela falha do desenvolvedor em cumprir com as instruções de uso do controlador do fabricante:

- Controlador desatualizado;
- Incompatibilidade do modelo do controlador com a aplicação;
- Defeito de um dispositivo individual ou do lote inteiro;
- Inconsistência da jogabilidade com as instruções.

O defeito mais comum é a falta de substituição ou a ausência completa de uma ferramenta ao trocar os controladores durante o jogo. As chaves de ligação podem variar de jogo para jogo. Elas também podem ser reatribuídas pelo jogador a seu próprio critério.

Uma versão desatualizada dos controladores ou sua ausência pode levar ao fato de que o controlador não funciona como esperado. Se os defeitos de software puderem ser eliminados por meio de atualização, então os defeitos técnicos e falhas dos controladores podem ser corrigidos apenas pela liberação de sua nova revisão.

Entretanto, uma imprecisão na leitura do movimento de um volante de corrida ou qualquer outro controlador pode surgir de um defeito de hardware ou de um defeito nos cálculos de software. Para jogos onde a precisão da leitura dos sinais de controle do controlador do jogo é crítica, o documento de projeto do jogo deve conter os valores requeridos em graus de inclinação.

Além disso, ao lançar um jogo em uma plataforma popular, o proprietário da plataforma pode fornecer requisitos para as imagens dentro do jogo de seu controlador. As imagens dentro do jogo são as imagens do controlador utilizadas no jogo. Por exemplo, pode ser uma representação esquemática de um controle de jogo no menu de configurações de chaves de ligação ou em dicas para a jogabilidade. As imagens não podem estar apenas dentro do jogo; os controladores também podem ser exibidos em uma embalagem de software ou em uma imagem digital. Este requisito geralmente se aplica a editores bem conhecidos e simultaneamente a fabricantes de consoles e controladores. As empresas, em sua documentação de teste para desenvolvedores de jogos, podem

fornecer requisitos sobre como os controladores devem ser apresentados em uma aplicação, incluindo tanto os contornos quanto as marcas registradas.

Além disso, para videogames onde o acelerômetro e o giroscópio do controlador são usados, o proprietário da plataforma impõe requisitos de segurança. O exemplo mais simples seria a necessidade de fazer movimentos específicos com os controladores para implementar a jogabilidade. Os sensores de movimento dentro dele permitem que ele seja usado como controle no espaço 3D. Nesses casos, é necessário indicar antes de iniciar o jogo, que o usuário precisa colocar as correias de retenção que estão presas aos controladores. Caso contrário, ao fazer movimentos bruscos com os controladores, eles podem escorregar das mãos do usuário e danificar o equipamento ao redor ou, pior ainda, tornar-se um perigo para a saúde.

6.2 Abordagens para testar controladores de jogos

Os testes usando controladores de jogo geralmente começam quando a funcionalidade do jogo está pronta para usar algum controlador padrão: para PC é um teclado / mouse, para consoles é um controle de jogo.

Este tipo de teste pode incluir o seguinte:

- conectar / desconectar o controlador do PC / console;
- baixo nível de bateria do controlador;
- suporte ao jogo para certos fabricantes de controladores;
- suporte para certas APIs que permitem que uma aplicação receba dados de um controlador;
- o uso de um e/ou vários controladores;
- uso não tradicional do controlador (testes negativos);
- vibração (presença e grau de sua intensidade).

Testes funcionais

Se o software exigir o uso de um controlador como dispositivo de entrada, então toda a funcionalidade da aplicação deve interagir corretamente com o dispositivo conectado. O usuário deve ser capaz de controlar tanto os elementos da interface do usuário quanto diretamente a jogabilidade sem alterar os controladores. Os elementos de controle atribuídos: botões, bastões, voltas do volante de corrida, comandos de voz ou movimentos no espaço devem corresponder às ações do personagem do jogo. Um testador também precisa certificar-se de que quando o controlador é desconectado do dispositivo, a aplicação faz uma pausa, se possível. Se a implementação da jogabilidade for realizada em tempo real e não puder ser interrompida, então a desconexão do controle de jogo não deve levar à desconexão do jogador do servidor, e também deve ser acompanhada por uma mensagem de informação.

Teste de segurança

Um controlador pode conter vulnerabilidade tecnológica e acesso aberto para invadir o console. Os testes de segurança devem ser realizados para mitigar os riscos de segurança: obter acesso não autorizado ao modo desenvolvedor, evitar o bloqueio do dispositivo através da rede usando o modo de voo etc. [ISTQB_AL_SEC]. O teste de segurança dos controladores de jogo mistura abordagens de descoberta de falhas de hardware que influenciam a funcionalidade do software.

Testes ergonômicos

Os botões dos controladores das marcas mais populares têm um propósito bem estabelecido. Os botões X e A, e os botões ☐ e ×, são muito utilizados em aplicações como botões para "aceitar", "aceitar seleção" ou "interagir" com um item interativo, e os botões B e ○ como "cancelar", "recusar" ou "devolver".

As combinações de botões são configuradas e controladas pelo software. Ao testar as combinações de botões, o testador precisa prestar atenção se elas são anatomicamente convenientes para diferentes grupos de jogadores (dependendo do gênero, idade, deficiência etc.): se o jogador tem a capacidade de alcançar vários botões simultaneamente ou sequencialmente.

Teste de conformidade dos controladores de jogo

As funções de especialista UI / UX e projetista de jogos são obrigatórias no desenvolvimento de jogos. Sua tarefa é desenvolver uma interface que seja agradável ao olho, que seja funcionalmente conveniente e que cumpra as convenções geralmente aceitas, ou normas do gênero. O objetivo dos testes será tanto testar a operacionalidade

da interação das entradas enviadas pelo controlador e as ações realizadas na interface, como também testar o UX geral (experiência do usuário).

O testador precisa ter certeza de que o desenvolvedor está usando layouts tradicionais para controladores populares, por exemplo, como mencionado anteriormente, os botões X e A, ou ☐ e × são usados como botões de "aceite", e os botões B e ○ são usados como botões de "cancelamento". Para o teclado, as teclas W, A, S e D são usadas como botões direcionais, Ctrl ou C para se agachar ou se arrastar, e a barra de espaço para pular. A função tradicional do botão esquerdo do mouse seria um tiro ou ataque para jogos de tiro em primeira pessoa, ou uma escolha para estratégias. O botão direito do mouse é normalmente reservado para apontar em atiradores em primeira pessoa ou para mover tropas em estratégias.

Os testadores também precisam certificar-se de que o uso de qualquer controlador não dê ao jogador uma vantagem significativa sobre os outros. Como os controles de jogo são significativamente inferiores ao par teclado/mouse em termos de velocidade e precisão de apontar para um alvo, eles são muitas vezes adicionados ao alvo hostil e à perseguição subsequente. A mira simplesmente "fixa" no adversário. Neste caso, é responsabilidade do testador acompanhar até onde a mira começa a seguir o oponente e se ele agarra sua cabeça, que muitas vezes é uma área vulnerável, infligindo danos adicionais quando atingido.

6.3 Ferramentas de suporte para teste em controladores de jogo

Várias ferramentas de software podem ser usadas para dar suporte aos testes dos controladores de jogos por gravação de vídeo ou representação esquemática dos controladores.

As ferramentas de captura/reprodução permitem a gravação de vídeo a partir de uma tela de PC. Elas são usadas para relatório de defeitos para exibir a natureza do defeito e as etapas para reprodução.

Outro tipo de ferramenta é usado para exibir uma representação esquemática do controlador de jogos e comandos em vídeo. Estas ferramentas simplificam a compreensão das atividades do controlador para descobrir e analisar o defeito.

Há serviços que suportam testes da entrada do controlador e do grau de deflexão dos pinos de controle, bem como a performance da vibração do controlador (ver [URL3]). Tais serviços têm todas as ferramentas de teste necessárias para o controle do jogo, ajudando a avaliar a integridade do controlador.

Ferramentas especiais para exibir teclas no teclado também estão disponíveis e podem ser úteis durante a execução do teste.

7 Teste de regionalização - 155 min

Palavras-chave

conformidade, internacionalização, regionalização

Palavras-chave específicas

adaptação cultural, precisão histórica, local

Objetivos de aprendizagem

7.1 Princípios e conceitos de teste de regionalização

GaME-7.1.1 (K1) Reconhecer as etapas do teste de regionalização

GaME-7.1.2 (K1) Recordar os principais objetivos da internacionalização e regionalização

GaME-7.1.3 (K2) Comparar capacidades de internacionalização e regionalização

7.2 Tipos de defeitos de regionalização e suas causas

GaME-7.2.1 (K2) Classificar os defeitos de regionalização e suas causas

7.3 Abordagens e execução de testes de regionalização

GaME-7.3.1 (K1) Reconhecer os testes de regionalização total e parcial

GaME-7.3.2 (K3) Classificar tipos de testes de regionalização

GaME-7.3.3 (K2) Resumir tarefas de teste para um escritor, editor, tradutor e testador de regionalização

7.4 Ferramentas de apoio para testes de regionalização

GaME-7.4.1 (K2) Resumir o uso de ferramentas para testar a regionalização de jogos

7.1 Princípios e conceitos de teste de regionalização

7.1.1 Regionalização e internacionalização

O processo de regionalização de qualquer produto, seja ele um reprodutor de vídeo, um produto de jogo ou um sistema operacional, tem etapas, abordagens, técnicas e difere nos procedimentos internos únicos adotados na empresa de desenvolvimento de software [Chandler11].

O desenvolvimento de jogos é uma ideia que é implementada através da escrita de código, complementada com o conteúdo necessário e liberada aos jogadores. Um importante estágio inicial no desenvolvimento é determinar o mercado para o qual o jogo será lançado:

- Esta etapa nem sempre é levada em conta, o que pode afetar negativamente as vendas do jogo no futuro;
- A presença de uma versão local do jogo é obrigatória para sua promoção bem-sucedida no mercado nacional;
- A maioria dos jogos é desenvolvida levando em conta a necessidade de adaptar o produto ao mercado alvo;
- A adaptação do jogo ao mercado-alvo pode aumentar suas vendas várias vezes.

Como o lançamento do jogo para outros mercados pode ocorrer após o fim do desenvolvimento da versão local do jogo (p. ex., a empresa tem dinheiro para lançar o jogo em outro mercado ou apareceram parceiros que estão prontos para promover o produto em certos países), então, ao desenvolver o jogo, a equipe deve levar em conta que o jogo pode ser adaptado a alguns outros mercados. Caso contrário, as adaptações subsequentes podem exigir grandes custos financeiros e de recursos.

Internacionalização

Para evitar riscos e dificuldades de adaptação para uma região específica, é utilizado um processo chamado de internacionalização. A internacionalização é a adaptação de um produto para uso potencial em quase qualquer lugar, enquanto a regionalização é uma mudança para uso em uma região específica.

Ao contrário da regionalização, a internacionalização de software é um conjunto de atividades realizadas durante a fase de desenvolvimento para facilitar a tradução e subsequente regionalização do software.

A internacionalização é realizada nos estágios iniciais de desenvolvimento e na maioria dos casos é realizada sem o envolvimento de tradutores e é de responsabilidade do desenvolvedor do software. A regionalização envolve tradutores especializados (em alguns casos nativos) com uma grande quantidade de conhecimentos adicionais.

A internacionalização inclui:

- Criação e desenvolvimento de software de tal forma que não haja barreiras à regionalização e ao uso internacional. Possivelmente utilizando *unicode* ou fornecendo uma abordagem de codificação de caracteres (se necessário).
- Criar oportunidades para utilizar elementos que não podem ser utilizados antes do processo de regionalização. Por exemplo, adicionar uma estrutura para texto bidirecional à linguagem de marcação da DTD (Definição do Tipo de Documento). Ou adicionando ao CSS (*Cascade Style Sheet*) uma base para texto vertical ou caracteres tipográficos não latinos.
- Capacidade de apoiar as referências regionais, linguísticas ou culturais. Isto geralmente inclui a introdução de dados localizados predefinidos ou uma tradução criada anteriormente e armazenada em software especializado. Exemplos incluem: formatos de data e hora, calendários locais, formatos de números e sistemas numéricos (números romanos e arábicos), seleção e apresentação de listas, e uso de nomes pessoais e formulários de contato.
- Extrair elementos no código ou conteúdo para que as versões regionalizadas possam ser baixadas posteriormente ou selecionadas com base na preferência do usuário. Como regra, isto é implementado na forma de pacotes de idiomas carregáveis.

Esta listagem não inclui necessariamente a regionalização do conteúdo, programa ou produto em outro idioma. Estas são as técnicas e abordagens ao desenvolvimento de software que permitem uma fácil migração para uma futura regionalização.

Regionalização

A regionalização é o processo de adaptação do software à cultura de um país. Um caso especial de regionalização é a tradução da interface do usuário, documentação e arquivos de software relacionados [Retsker81].

A regionalização de jogos inclui:

- Tradução de textos de diálogos e legendas de jogos, dicas de ferramentas, descrições e mensagens, nomes de personagens, nomes de objetos;
- Tradução de protetores de tela, itens de interface e menu;
- Redesenho de texturas e gráficos;
- Seleção de atores, dublagem e gravação de arquivos de som;
- Integração de materiais localizados no jogo;
- Tradução e adaptação do website do projeto e impressão;
- Tradução de materiais publicitários (notícias, comunicados de imprensa e materiais de marketing);
- Suporte para o jogo após o lançamento da versão localizada (atualizações, notícias e correções).

Assim, testar a regionalização é um teste de quão bem um produto de jogo é adaptado para um público-alvo específico de acordo com suas características e referências culturais, linguísticas, religiosas, políticas e outras. Normalmente, os aspectos culturais e linguísticos são considerados aqui, em particular a tradução da interface do usuário, documentação e arquivos para outro idioma, bem como os formatos de moedas, números, horários, números de telefone etc.

Os testes de regionalização incluem testar o conteúdo de uma aplicação de jogo contra requisitos linguísticos e culturais, bem como as especificidades de um determinado país ou região. Este tipo de teste ajuda a encontrar defeitos ou falhas de tradução em uma versão regionalizada antes que o produto final chegue ao usuário. O objetivo dos testes de regionalização é encontrar e corrigir defeitos em diferentes versões regionalizadas de um produto para diferentes mercados e cenários regionais.

É importante observar que a regionalização não é apenas a tradução para vários idiomas, e os testes de regionalização e testes linguísticos não são a mesma coisa. Os testes linguísticos consistem principalmente em testes para defeitos ortográficos, gramaticais e estilísticos.

7.1.2 Diferença entre a regionalização de um produto de jogo e o software aplicativo

A diferença entre testar a regionalização de produtos de jogos e software aplicativo baseia-se no entendimento de que estes dois tipos de produtos diferem significativamente um do outro em vários aspectos levados em consideração durante a regionalização.

As principais diferenças são:

- Adaptação do conteúdo gráfico ao público-alvo;
- Regionalização do conteúdo de áudio;
- Regionalização e adaptação do conteúdo do texto;
- Conformidade com o gênero e características literárias.

Como regra, uma grande quantidade de conteúdo gráfico pode exigir adaptação para o público-alvo. Este tipo de gráficos inclui sinais, personagens, objetos de jogo, mapas de nível, símbolos e acessórios, itens de jogo, protetores de tela publicitários etc.

O conteúdo de áudio também pode exigir uma regionalização cuidadosa. Muitas vezes, o conteúdo de áudio de um produto de jogo deve ser duplicado pelos atores no idioma do público-alvo, e o conteúdo em si deve ser adaptado.

Estilos mistos de conteúdo de texto requerem tradução e adaptação adequadas. os seguintes estilos podem ser usados em um jogo:

- Um estilo científico (descrição dos mecanismos, instruções);
- Estilo jornalístico (jornais, revistas, artigos);
- Estilo artístico (diários pessoais, livros);
- Negócios oficiais (dossiê, vários documentos);
- Colóquio diário (diálogos de personagens).

Por exemplo, no gênero de aventura (busca), haverá todos os dias um trabalho científico, jornalístico, coloquial, pois o jogador precisa consertar ou montar algum dispositivo de acordo com as instruções, encontrar as informações necessárias nos jornais e interrogar as pessoas.

Há um requisito de adaptação obrigatório do conteúdo de um produto de jogo, levando em conta fatores como: exatidão histórica, características religiosas, culturais, políticas e ideológicas do público-alvo etc. Em alguns casos, a violação deste requisito pode levar a uma percepção ambígua do produto pelo público-alvo e proibir seu uso no território de um determinado país ou região.

Há uma necessidade de respeitar as peculiaridades de gênero e literárias ao localizar um produto de jogo. Por exemplo, no RPG multiplayer, ao traduzir, é necessário levar em conta os nomes das raças dos jogadores, nomes dos objetos etc. Às vezes é necessário adaptar o conteúdo usando conceitos estilisticamente próximos e relacionados ao gênero.

São necessários conhecimentos e informações adicionais para a implementação de uma tradução de alta qualidade de um produto de jogo. Por exemplo, várias referências a outros produtos de jogos, trabalhos de mídia ou eventos da realidade que possam estar contidos no jogo.

Abaixo estão algumas das abordagens para a regionalização de produtos de jogos de azar subjacentes aos testes de regionalização.

7.1.3 Etapas de teste de regionalização

Os testes de regionalização incluem testar a correção do conteúdo traduzido, vários elementos de interface, defeitos e mensagens do sistema e testar as seções de Perguntas Frequentes e Ajuda.

Testes de tradução

O objetivo do teste de tradução é testar a interface multilíngue do jogo quanto a defeitos de tradução, correção de endereços postais, primeiro e último nomes, moedas, formatos de data e hora etc.

Às vezes, durante o desenvolvimento, é necessário adaptar a aparência do número e do valor numérico às normas regionais (p. ex., unidades de medida). Ao testar, o testador deve sempre lembrar qual sistema é adotado em um determinado país a fim de informar claramente ao usuário sobre velocidade, comprimento, peso, temperatura etc.

A mensagem "*Você está se movendo a 62 milhas por hora*" pode ser difícil para um jogador de um país que usa o sistema métrico internacional. Neste caso, não basta apenas alterar o valor numérico da velocidade, mas também é necessário adaptar a unidade de medida.

O acima exposto também se aplica à moeda utilizada nos jogos para fazer compras. O exemplo mais simples de regionalização de preços é a tradução automática para a moeda da região onde o produto foi ativado. Além da conversão à taxa de câmbio, o símbolo da moeda correspondente também deve estar presente. Entretanto, ao desenvolver um aplicativo, o testador também deve levar em conta o fato de que os serviços de distribuição de software podem estabelecer preços regionais.

Em vários videogames, os preços das lojas são convertidos em moeda e variam de acordo com a região.

Ao testar jogos, a atenção deve ser dada aos gráficos. Eles devem corresponder às realidades do país para o qual o jogo está sendo publicado. Por exemplo, a sinalização rodoviária pode parecer diferente de país para país. Além disso, imagens e cenas de feriados locais são adicionadas. Nos países muçulmanos, os gráficos são radicalmente revisados - todas as imagens de pessoas e animais são removidas e arabescos (ornamentos medievais orientais complexos compostos de elementos geométricos e vegetais) são acrescentados.

A comemoração do Ano Novo Chinês é especialmente popular. Em muitos jogos, a interface é decorada com símbolos nacionais: lanternas, dragões e fogos de artifício. Além disso, o jogo pode ser reabastecido com conteúdo limitado (disponível por um tempo limitado): heróis mitológicos chineses estilizados, *skins* para personagens, animações para lançamento de fogos de artifício e foguetes, cavalgar em porcos e dragões.

Nos jogos, as piadas muitas vezes têm que ser adaptadas, e ocasionalmente até mesmo o enredo tem que ser ajustado para se alinhar com a mentalidade do país em que o jogo será vendido.

A regionalização não se limita ao trabalho com texto - ela também inclui a conciliação de características culturais e aspectos morais e éticos.

Tradicionalmente, há alguns tópicos e instruções para testar a regionalização que os testadores devem prestar a maior atenção:

- Religião, incluindo o ocultismo e o satanismo;
- Sexo, personagens vestidos de maneira reveladora e linguagem vulgar;
- Preconceitos e estereótipos associados à cultura e ao próprio povo;
- Guerras, conflitos militares, terrorismo;
- A política, incluindo a visão específica dos diferentes países sobre a história.

Por exemplo, se um jogo tiver a citação verbal de uma passagem do Alcorão, a maioria dos países islâmicos provavelmente proibirá este jogo.

Nas condições de controle rigoroso do politicamente correto no mundo moderno, sua observância deve ser levada a sério.

Além das dificuldades causadas pelo componente técnico do jogo ou preparação insuficiente para a regionalização do jogo, um tradutor de jogo também pode enfrentar uma série de dificuldades associadas não ao jogo que está traduzindo, mas com a necessidade de ter conhecimentos adicionais para realizar uma tradução de alta qualidade.

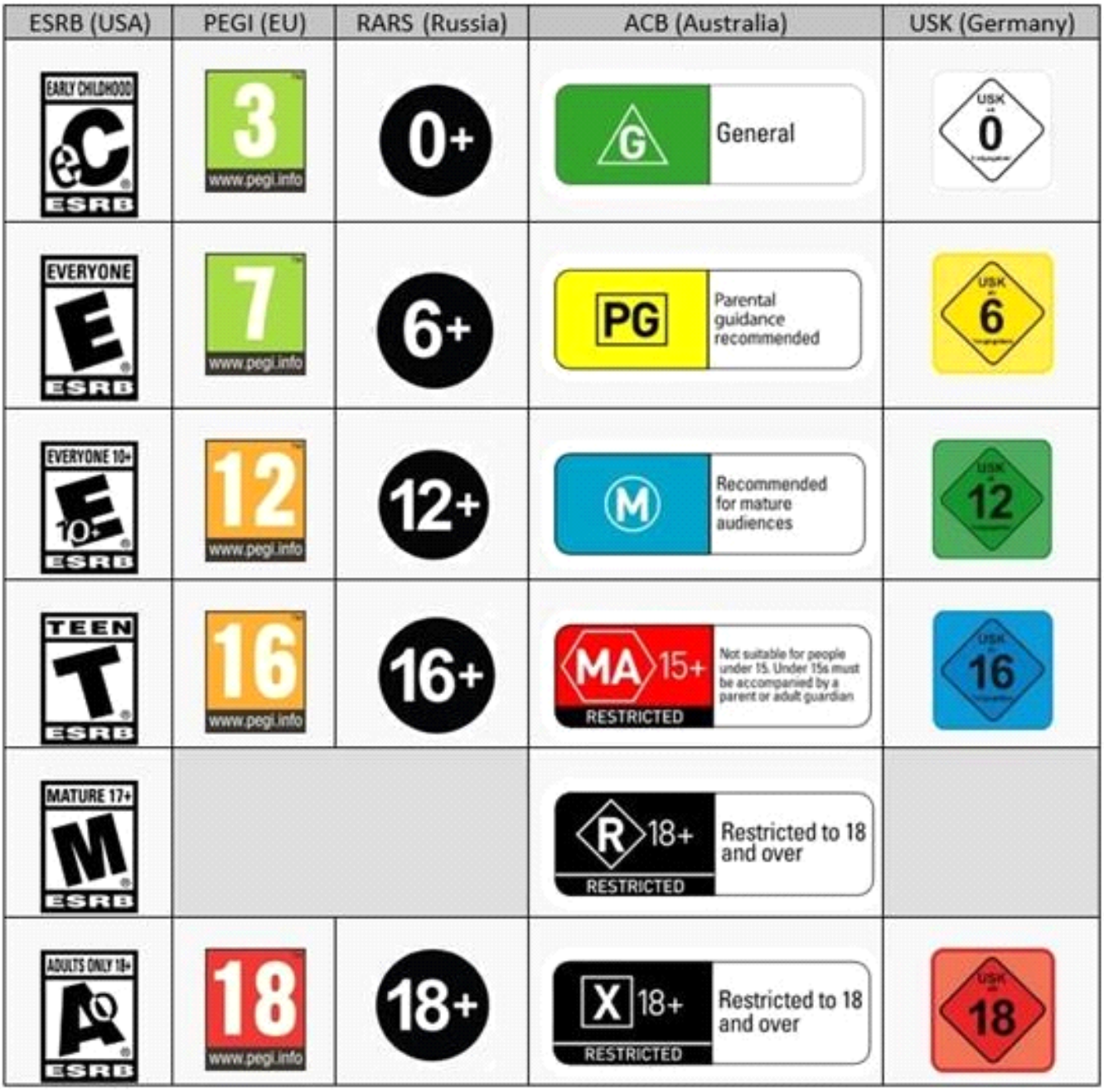
Um conhecimento adicional referidos em outros produtos de jogo, trabalhos de mídia ou eventos reais que podem estar contida no jogo, devem ser traduzidas adequadamente de acordo com a forma como foram traduzidas anteriormente. Portanto, o tradutor deve estar ciente dos últimos eventos que acontecem no mundo, assim como de vários filmes e jogos populares.

Teste de conformidade

O sistema de restrições de idade para jogos e aplicações leva em conta as peculiaridades da legislação e da cultura de cada país. Isto permite que os desenvolvedores definam com mais precisão as restrições de conteúdo e distribuam aplicações para o público ao qual elas se destinam.

Ao testar a regionalização, os requisitos legais que são suportados em diferentes regiões devem ser levados em conta. Por exemplo, na Rússia a idade da maioridade chega aos 18 anos, nos Estados Unidos, dependendo do estado, esta idade varia de 18 a 21 anos, no Japão os jovens são considerados adultos a partir dos 20 anos de idade, e no Brasil a maioridade é atingida aos 21 anos. Jogos com conteúdo diferente na maioria dos casos devem seguir o sistema de restrição de idade em vigor em diferentes países.

O mesmo jogo pode receber classificações etárias diferentes em países diferentes.



Classificação do jogo em diferentes países

https://www.kaspersky.com/blog/gaming-idade-classificações/11647/

Ao testar a regionalização para cumprimento da legislação e requisitos legais, é necessário levar em conta muitas circunstâncias, às vezes associadas não tanto ao idioma alvo, mas às características legais e culturais do país para o mercado no qual o produto do jogo é oferecido.

Além disso, também vale a pena prestar atenção ao suporte no jogo da moeda e às operações com ela para cada país. Por exemplo, se em um determinado país as transações de câmbio são proibidas para todos, exceto o Banco daquele país. Lançar um jogo naquele mercado certamente levará os editores de jogos a sérios problemas.

Indiretamente, as transações monetárias incluem a compra de itens - literalmente "pacotes de itens", que, como regra, contêm várias *skins* de personagens, itens consumíveis, upgrades ou acessórios.

Em dezembro de 2016, o Ministério da Cultura da República Popular da China anunciou a promulgação de uma lei que exige que uma editora de jogos on-line publique a probabilidade de aquisição de todos os itens e serviços virtuais a partir de maio de 2017.

Na Austrália, os jogos de pacotes de itens estão sujeitos a restrições de jogo se eles puderem ser jogados por dinheiro ou qualquer coisa de valor. As perguntas permanecem mesmo quando o valor de um objeto que existe somente no jogo pode ser determinado somente em conexão com o prestígio deste objeto.

Cores e símbolos

Cores e símbolos específicos, que podem ter significados diferentes dependendo do país, também devem ser considerados ao testar a regionalização.

Por exemplo, o vermelho para os habitantes da China é um símbolo de resistência e fé, na Índia simboliza a pureza. A Europa, por outro lado, vê o pecado e o sacrifício nesta cor. Para o povo da África do Sul, é a cor do luto. Nos Estados Unidos e no Japão, o vermelho simboliza o perigo e a ameaça terrorista, enquanto os egípcios o associam ao luto. Portanto, ao testar um projeto de jogo, o testador precisa prestar atenção aos esquemas de cor, pois eles podem ser importantes.

Além disso, ao testar, o testador deve prestar atenção ao suporte para layouts de teclado e *hotkeys*.

No caso da aplicação utilizar integrações com recursos de terceiros (p. ex., armazenamento em nuvem ou rede social), é necessário levar em conta sua disponibilidade para regiões.

Assim, os processos de teste para regionalização e internacionalização serão diferentes:

Internacionalização	Regionalização
• Codificações UTF • Formatos de dados • Direção do texto	• Tradução • Requisitos legais • Transações em moeda e divisas • Cores e símbolos • Layout e teclas de atalho do teclado • Integração com recursos de terceiros

7.2 Tipos de defeitos de regionalização e suas causas

7.2.1 Possíveis causas de defeitos na regionalização de jogo

Mesmo as grandes empresas não estão imunes a falhas "culturais". Ao liberar qualquer aplicação, o testador deve se familiarizar com os precedentes e regulamentos existentes. Em alguns casos, uma frase ou brincadeira pode levar a uma proibição total da venda do jogo em certos países. As seguintes causas de possíveis defeitos de regionalização (p. ex., detalhes ou conteúdos proibidos em uma determinada região) devem ser abordadas:

- Imagens;
- Som e trilhas sonoras;
- Cenas realistas ou históricas;
- Frases ou citações.

7.2.2 Defeitos e riscos de regionalização

A falta de conhecimento do idioma de destino ou do próprio produto pode significativamente complicar os testes de regionalização, especialmente se o produto for um representante do gênero de simulação: médico, técnico ou esportivo [URL4].

Também deve ser levado em conta que o teste de regionalização pode ser um processo bastante longo, já que leva tempo para estudar as características de diferentes regiões.

Os principais defeitos de regionalização podem ser agrupados da seguinte forma:

Aspectos técnicos

- *Mojibake* (文字化け). O texto distorcido aparece como resultado da decodificação de texto usando uma codificação de caracteres incorretamente selecionada. O resultado é a substituição sistemática de caracteres com caracteres completamente não relacionados, muitas vezes de um sistema de escrita diferente. Como regra, isto leva a textos ilegíveis.
- Uma determinada *string* não cabe nos limites estabelecidos pela interface (cortado, deslocado). Um termo traduzido de um idioma para outro pode usar um número diferente de caracteres.
- Compensação. Após a regionalização de uma aplicação, os layouts dos elementos de interface do usuário podem exigir reconfiguração para manter o alinhamento original.
- Nenhuma tradução de linha para o idioma localizado. Os testadores devem prestar atenção aos textos nas caixas de diálogo, imagens ou capturas de tela nos documentos ou na interface do usuário. Todo este conteúdo precisa ser localizado para atender às expectativas do usuário.
- Sobreposição. Isto acontece quando certos controles em uma janela ou diálogo se sobrepõem a outros.
- Textos em falta. Durante a tradução ou criação de aplicações, os textos ocasionalmente podem ser perdidos.
- Fonte / tamanho errado. Diferentes países utilizam fontes e tamanhos padrão diferentes. Por exemplo, os países asiáticos geralmente usam um tamanho de fonte 9 por padrão, enquanto os americanos usam um tamanho de fonte 8. Esta questão geralmente não interfere em nenhuma funcionalidade, mas afeta muito a experiência do usuário, tornando o texto difícil de ler.
- Teclas de atalho incorretas. Em alguns casos, a tecla de atalho não está disponível devido à ausência da letra no idioma ou localização no teclado.
- Variáveis em modelos de texto. Diferentes tipos de idiomas (analítico, sintético) usam diferentes formas de declinação, casos e plurais, podem mudar ou não as terminações e sons, assim, as variáveis podem afetar as formas das palavras. Um dos erros mais comuns que os desenvolvedores cometem é a divisão das frases contendo variáveis em partes. Uma estrutura gramatical do idioma de destino é quase a principal pedra angular da regionalização. De fato, ao adicionar variáveis, os desenvolvedores muitas vezes não levam em conta que em vários idiomas seu significado pode mudar a construção de uma frase.
- Fora de sincronia entre a sequência sonora original e a da tradução. Ao traduzir, as frases dos caracteres podem ser mais longas ou, inversamente, mais curtas do que no original, devido ao número diferente de sílabas. Os vídeos nos jogos são frequentemente pré-gravados, o que obriga os localizadores a adotarem uma abordagem mais cuidadosa na tradução das sílabas: é importante manter um equilíbrio entre a precisão da tradução e o número de sílabas, levando também em conta a sincronização do som com os movimentos dos lábios dos personagens (*lip sync* – sincronização labial).

Defeitos de tradução

- Tradução ou transliteração incorreta de nomes próprios, datas, valores numéricos, nomes de eventos históricos, feriados, realidades, jargão, profanação, vocabulário coloquial, abreviações, nomes e apelidos "evocativos" etc.
- Termos contraditórios. O mesmo termo deve ter uma tradução consistente ao longo de toda a aplicação. Acontece com frequência que as traduções são incompatíveis entre si.

Adaptação cultural do conteúdo

- Humor. A atitude em relação a objetos de brincadeira, humor e sátira, a consideração do humor e sua permissibilidade em relação a objetos individuais etc.
- Religião. A atitude do público-alvo em relação às questões religiosas, incluindo locais religiosos, rituais e cerimônias. Atitude do público-alvo em relação aos cultos periféricos (satanismo, paganismo).
- Precisão histórica e percepção dos eventos. A interpretação de eventos históricos, conflitos militares, descobertas no campo da ciência, a aparência e características de objetos específicos existentes, as peculiaridades do modo de vida das pessoas etc.
- Peculiaridades da cultura regional e da visão do mundo (incluindo atitudes em relação às crianças, sexo, violência, outras culturas etc.). Estereótipos nacionais, culinária regional, roupas, estilo de vida das pessoas. atitude em relação às crianças, atitudes em relação às minorias sexuais, violência contra animais etc.

- Restrições legais. Levando em conta classificações etárias, legislação no campo da proteção à criança, crenças religiosas, propaganda de violência, drogas, relações sexuais, raça, terrorismo, discursos contra o governo e o sistema estatal, proibição do uso de símbolos específicos etc.
- Regionalização excessiva. Nem tudo precisa ser regionalizado e alguns elementos devem manter sua aparência original sem tradução, por exemplo: marcas registradas, logotipo, abreviações, nomes de produtos.

7.3 Abordagens e execução de testes de regionalização

7.3.1 Diferença entre os testes de regionalização total e parcial

Teste de regionalização total

Isto implica um teste minucioso da regionalização de defeitos. Este tipo só é eficaz quando se desenvolve uma nova regionalização para um cliente, desde que todas as *strings* tenham sido traduzidas e não tenham sido testadas antes. É o mais caro, pois envolve todos os tipos de testes.

Testes de regionalização parcial

Isto é usado quando o texto muda dentro de regionalizações que foram previamente testadas. A validação identifica linhas que foram alteradas como parte de uma grande atualização que afetou a história principal e sua regionalização de referência. Somente o texto modificado é testado, o qual deve corresponder ao texto previamente apresentado no jogo. Esta abordagem de teste é considerada ótima em termos de obtenção de eficiência, ao mesmo tempo em que minimiza os custos.

7.3.2 Procedimentos e abordagens para a regionalização de testes durante ciclo de vida de desenvolvimento de software de jogo

Os testes de regionalização verificam a tradução, os arquivos de suporte, a justificativa correta e a adaptação dos elementos da interface, bem como as regras de redação do texto.

O objetivo do teste de regionalização é garantir que o jogo suporte interface e funcionalidades multilíngue, e que não haja problemas de regionalização (tradução para outro idioma, formato de data e número, endereços de correspondência, ordem de nome e sobrenome, moeda etc.).

O processo de teste de regionalização inclui:

- Determinação e estudo da lista de idiomas suportados;
- Teste da correção da tradução, incluindo elementos da interface do usuário, mensagens do sistema e defeitos;
- Verificação da tradução da seção "Ajuda" e da documentação de acompanhamento (se houver).

Os testes de regionalização envolvem comparar as *strings* traduzidas pela equipe de regionalização com as *strings* da regionalização de referência sobre:

- Gramática, pontuação, defeitos de sintaxe;
- Violações dos requisitos regionais da regionalização testada (formato de hora, data, medidas, conformidade legal etc.);
- Falta de dados técnicos necessários (variáveis, seções utilizadas para formatar o texto etc.);
- Violações do estilo artístico e do contexto de regionalização de referência;
- Defeitos na exibição de texto nas interfaces do jogo (linhas muito longas, violação do formato de exibição das fontes etc.).

Todos os testes acima devem ser realizados pela equipe de testes. Entretanto, nem sempre é possível testar a gramática, pontuação, sintaxe ou estilo e contexto da regionalização de referência, já que é bastante difícil encontrar testadores com conhecimento de cada um dos idiomas para os quais o jogo é traduzido. A execução destes testes fica do lado dos regionalizadores. Os testadores verificam as violações dos requisitos locais da regionalização testada, verificam a falta dos dados técnicos necessários e identificam defeitos na exibição do texto nas interfaces do jogo. (Para mais detalhes, consulte a seção "7.3.3. Tipos de testes de regionalização").

O estágio preliminar dos testes de regionalização

Esta etapa inclui

- Fornecer aos testadores toda a documentação necessária do produto;
- Criação de um glossário e memória de tradução para ajudar os testadores a interpretar corretamente os termos utilizados;
- Fornece uma versão anterior do produto, caso já tenha sido localizada anteriormente, para fins de avaliação;
- Seleção e configuração de ferramentas de gerenciamento de defeitos - um documento ou uma plataforma onde todos os defeitos encontrados durante os testes de regionalização serão registrados.

Teste das características regionais e culturais

Este é um dos passos mais importantes nos testes de regionalização. Serão usadas imagens de tela ou uma construção localizada do jogo. Os seguintes itens precisam ser testados:

- Formato de data e hora para a região selecionada;
- Formatos de números de telefone e endereços;
- Esquemas de cores;
- Conformidade dos nomes dos produtos com as normas regionais;
- Formato da moeda;
- Unidades.

Teste linguístico

As características linguísticas são testadas. O testador precisa ter certeza disso:

- A mesma terminologia é utilizada;
- Não há defeitos gramaticais;
- Não há defeitos ortográficos;
- As regras de pontuação são seguidas;
- É utilizada a direção correta do texto (da direita para a esquerda ou da esquerda para a direita);
- Os nomes corretos de marcas, cidades, lugares, posições etc. são definidos.

Interface do usuário (ou aparência)

É importante ter certeza do seguinte:

- Todas as legendas das fotos estão regionalizadas;
- O layout da versão regionalizada é o mesmo que o original;
- As quebras de linha nas páginas / telas estão colocadas de acordo com as regras do idioma de destino;
- As conversas, pop-ups e notificações são exibidas corretamente;
- O comprimento das linhas não excede os limites existentes, e o texto é exibido corretamente (às vezes o texto traduzido é mais longo do que o original e não cabe nos botões).

Funcionalidade

É necessário testar se a aplicação regionalizada está funcionando corretamente, por isso, chama-se a atenção para isso:

- Funcionalidade de um produto regionalizado;
- Funções para entrada de informações;
- Suporte para caracteres especiais para diferentes locais e idiomas;
- Suporte para atalhos de teclado;
- Suporte para várias fontes;
- Suporte para vários separadores de formatos.

Os testes em vídeo e em escala incluem:

- Correspondência da extensão da escala e do elemento do jogo ao qual pertence;
- Correspondência da escala dos personagens por gênero;
- A pureza do som (ou seja, a ausência de interferência, assim como a sincronização da escala do original e da tradução e sua sonoridade em relação um ao outro);

- A confiabilidade da trilha sonora, incluindo o aspecto histórico;
- Características estilísticas e culturais do som (sotaques, características da fala).

7.3.3 Tipos de testes de regionalização

Regionalização da "embalagem"

Se um jogo é lançado e vendido em mídia física, o que está escrito na embalagem é regionalizado. Se não for vendido em uma mídia física, mas em uma plataforma de jogos, então a página da loja é traduzida: descrição e imagens. A regionalização da caixa é limitada a isto.

Regionalização da interface

A descrição e a embalagem serão traduzidas no jogo, a interface, a página de ajuda, os *labels* dos botões etc. Um tipo incomum de regionalização surge em uma situação em que a inscrição no botão "Jogar" é feita em um idioma enquanto a trama está completamente em outro idioma.

Regionalização de texto

Normalmente, todos os textos do jogo podem ser traduzidos em legendas. Isto significa que um usuário pode ouvir e tentar entender, por exemplo, a gíria afro-americana no jogo, mas ainda ver as legendas em outro idioma.

Regionalização com atuação vocal

O discurso e os diálogos são traduzidos e expressados pelos atores. Se a regionalização com atuação da voz é feita a um bom nível, não é percebida como algo estranho.

Regionalização gráfica

Qualquer jogo contém algum tipo de motor, desenho, objetos gráficos, texturas - tudo que não é texto, por exemplo, uma inscrição em uma cerca em um jogo. A regionalização gráfica implica que todas as inscrições dentro do jogo devem ser traduzidas. Estas podem incluir, por exemplo, jornais, letreiros de lojas e algumas notas.

Muitas vezes as ações do jogo acontecem em determinados locais. Ao mesmo tempo, a abordagem da regionalização pode ser diferente. Se algum objeto (p. ex., notas de jornal) ajudar na trama, ele deve ser traduzido, caso contrário, um ponto importante será perdido. Ao mesmo tempo, as inscrições nas paredes no idioma do local onde o jogo acontece não precisam ser traduzidas se forem apenas um arranjo.

Regionalização profunda - adaptação cultural

Esta é uma adaptação cultural, quando o jogo é completamente refeito. Tudo o que resta é o código fonte e a mecânica. Os desenvolvedores podem refazer texturas, tramas, diálogos e modelos de personagens e fazer um jogo completamente diferente no mecanismo do jogo. Isto é feito muito raramente, mas este método de regionalização ainda ocorre. Isto é feito nos casos em que um jogo sem tal adaptação não é capaz de ser usado pelo público e vendido em um determinado mercado.

7.3.4 Áreas de responsabilidade das pessoas envolvidas

Papel	Responsabilidades
Escritor / Projetista Narrativo	• Desenvolvimento de uma estratégia e de um plano de regionalização; • Definição de terminologia, tradução de conceitos específicos do gênero de jogo e termos difíceis de traduzir relacionados ao projeto; • Explicando o contexto aos editores e tradutores.
Tradutor	• Tradução de textos
Editor	• Revisão do material traduzido; • Teste de conformidade do texto com as características culturais juntamente com os aspectos morais e éticos do idioma em que a regionalização é realizada; • Controle da uniformidade do estilo de tradução.

Papel	Responsabilidades
Testador de regionalização	• Testar a correção da tradução (contexto e significado das observações, coerência do texto, conformidade com o estilo de jogo); • Testar a conformidade técnica da tradução (aparência, interfaces, funcionalidade das fontes, separadores, caracteres especiais etc.); • Controle da integração técnica da tradução (com aplicações internas e externas).

7.4 Ferramentas de suporte para testes de regionalização

As ferramentas de teste de regionalização podem ser usadas em diferentes estágios de teste e para tratar de diferentes questões. Estas ferramentas incluem:

- Ferramentas de comparação visual de *strings*. Estas ferramentas ajudam a comparar cadeias de caracteres de referência e alvo da regionalização;
- Ferramentas de comparação automática de *strings*. As ferramentas deste tipo incluem quaisquer scripts auxiliares, programas, utilitários, como resultado dos quais é possível obter dados sobre a presença de defeitos de qualquer tipo na regionalização testada;
- Ferramentas para testar a estrutura do arquivo de regionalização para arquivos ausentes ou redundantes;
- Exemplos de ferramentas automatizadas para testar a regionalização incluem:
    - ferramentas que comparam *screenshots* para duas localizações (referência e verificadas) e geram falhas se houver uma discrepância significativa;
    - ferramentas que determinam a presença de mudanças na regionalização do alvo, com base na presença de mudanças na referência;
    - ferramentas que comparam variáveis e valores numéricos em cada par de referência de cordas - e regionalização de destino;

Referências

Normas

[ISO25000] ISO/IEC 25000:2014, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE

[ISO25010] ISO/IEC 25010:2011, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models

Documentos ISTQB

[ISTQB_AL_SEC] ISTQB Advanced Level Security Testing Syllabus, Version 2016

[ISTQB_ALTA_SYL] ISTQB Advanced Level Test Analyst Syllabus, Version 3.1.2

[ISTQB_ALTTA_SYL] ISTQB Advanced Level Technical Test Analyst Syllabus, Version 4.0

[ISTQB_ALTM_SYL] ISTQB Advanced Level Test Manager Syllabus, Version 2012

[ISTQB_CTFL_MAT] ISTQB Mobile Application Tester, Version 2019

[ISTQB_EXAM_S&R] ISTQB Exam Structure and Rules, Game Testing, Version 1.0

[ISTQB_FL_AT] ISTQB Foundation Level Agile Tester Syllabus, Version 2014

[ISTQB_FL_PT] ISTQB Foundation Level Performance Testing Syllabus, Version 2018

[ISTQB_FL_SYL] ISTQB Foundation Level (Core) Syllabus, Version 2018

[ISTQB_GLOSSARY] ISTQB Glossary of Terms used in Software Testing, https://glossary.istqb.org/

[ISTQB_UT_SYL] ISTQB Foundation Level Usability Testing Syllabus, Version 2018

Literatura

[Nystrom14] Nystrom, R. (2014). Game programming patterns. Genever Benning., ISBN: 978-0990582908. URL: https://www.gameprogrammingpatterns.com/

[Gregory18] Gregory, J. (2018). Game engine architecture. AK Peters/CRC Press., ISBN: 978- 1466560017. URL: https://www.gameenginebook.com/

[Buttfield19] Buttfield-Addison, P., Manning, J., & Nugent, T. (2019). Unity game development cookbook: essentials for every game. O'Reilly Media., ISBN: 9781491999158

[Lee16] Lee, J. (2016). Learning unreal engine game development. Packt Publishing Ltd., ISBN: 9781784398156

[Tavakkoli18] Tavakkoli, A. (2018). Game Development and Simulation with Unreal Technology. CRC Press., ISBN-13: 978-1498706247

[Romero19] Romero, M., & Sewell, B. (2019). Blueprints Visual Scripting for Unreal Engine: The faster way to build games using UE4 Blueprints. Packt Publishing Ltd., ISBN: 9781789347067 Certified Tester Game Testing (CT-GaMe)

[Chandler11] Chandler, H. (2011). The Game Localization Handbook 2nd Edition, Jones & Bartlett Learning, ISBN: 0763795933

[Retsker81] Retsker, Ya. I. (1981). Textbook for translation from English into Russian. M.: Prosveschenie. (In Russian).

Links

Nota: Todas as referências são de 28 de abril de 2022

[ISTQB-Web]

https://www.istqb.org

https://bstqb.org.br

[URL1] https://research.ncl.ac.uk/game/mastersdegree/workshops/technicalrequirementschecklists/Technical%20Requirements%20Checklist%20Workshop.pdf

[URL2]

https://docs.microsoft.com/en-us/gaming/gdk/_content/gc/live/get-started/live-xbloverview

[URL3]

https://gamepad-tester.com

[URL4]

https://igda-website.s3.us-east-2.amazonaws.com/wpcontent/uploads/2021/04/09142137/Best-Practices-for-Game-Localization-v22.pdf

Apêndice A - Objetivos de aprendizagem e nível cognitivo de conhecimento

Os seguintes objetivos de aprendizagem são definidos como aplicáveis a este syllabus. Cada tópico será examinado de acordo com objetivo de aprendizagem para ele. Os objetivos de aprendizagem começam com um verbo de ação correspondente ao seu nível de conhecimento cognitivo, conforme listado abaixo.

Nível 1: Lembrar (K1)

O candidato lembrará, reconhecerá e recordará um termo ou conceito.

Verbos de ação:

Relembrar, reconhecer

Exemplos:

- Recordar os conceitos da pirâmide de teste.
- Reconhecer os objetivos típicos dos testes

Nível 2: Compreender (K2)

O candidato pode selecionar as razões ou explicações para as declarações relacionadas ao tópico, e pode resumir, comparar, classificar e dar exemplos para o conceito de teste.

Verbos de ação:

Classificar, comparar, diferenciar, distinguir, explicar, dar exemplos, interpretar, resumir.

Exemplos:

- Classificar as ferramentas de teste de acordo com sua finalidade e as atividades de teste que suportam.
- Comparar os diferentes níveis de teste (pode ser usado para procurar semelhanças, diferenças ou ambos).
- Diferenciar o teste da depuração (procurar diferenças entre conceitos).
- Distinguir entre riscos do projeto e riscos do produto (permitir que dois (ou mais) conceitos sejam classificados separadamente).
- Explicar o impacto do contexto no processo de teste.
- Dar exemplos da necessidade de testar.
- Inferir a causa raiz dos defeitos a partir de um determinado perfil de falhas.
- Resumir as atividades do processo de revisão do produto de trabalho.

Nível 3: Aplicar (K3)

O candidato pode realizar um procedimento quando confrontado com uma tarefa familiar, ou selecionar o procedimento correto e aplicá-lo a um determinado contexto.

Verbos de ação:

Aplicar, implementar, preparar, utilizar

Exemplos:

- Aplicar análise de valor limite para derivar casos de teste de determinados requisitos (deve se referir a um procedimento / técnica / processo etc.).
- Implementar métodos de coleta de métricas para apoiar os requisitos técnicos e de gerenciamento.
- Preparar testes de instalabilidade para aplicações móveis.
- Usar a rastreabilidade para monitorar o progresso do teste para que ele seja completo e consistente com os objetivos, estratégia e plano de teste (pode ser usado em um LO que deseja que o candidato seja capaz de usar uma técnica ou procedimento - semelhante a "aplicar").

Apêndice B - Matriz de rastreabilidade de resultados de negócio com os objetivos de aprendizagem

Esta seção lista a rastreabilidade entre os resultados de negócio do *Certified Tester Game Testing* e os objetivos de aprendizagem do *Certified Tester Game Testing*.

Resultados de negócio	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
Entender os conceitos básicos dos testes de software de jogos.	17					
Determinar riscos, objetivos e requisitos de software de jogo de acordo com as necessidades e expectativas dos stakeholders.		12				
Conceitualmente projetar, implementar e executar testes básicos de software de jogo.			6			
Conhecer as abordagens aos testes de software de jogo e seu propósito.				9		
Reconhecer as ferramentas de apoio aos testes de jogo.					7	
Determinar como as atividades de teste se alinham com o ciclo de vida do software e reduzir o custo de desenvolvimento e publicação de jogos.						6

Especificidade do teste do jogo

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-1.1.1	K1	Reconhecer os objetivos e especificidades do teste de jogos.	X					
GaMe-1.1.2	K2	Dar exemplos de riscos do produto em software de jogo.	X	X				
GaMe-1.1.3	K2	Dar exemplos de defeitos específicos relacionados a teste de jogos.	X					
GaMe-1.1.4	K2	Resumir como os riscos do teste de jogos podem ser mitigados.		X				
GaMe-1.1.5	K2	Comparar as atividades de teste de jogos com as de jogo.	X					
GaMe-1.2.1	K1	Reconhecer funções e tarefas específicas na equipe de desenvolvimento do jogo.	X					
GaMe-1.3.1	K1	Atividades de testes de recall durante todo o ciclo de vida do software do jogo.					X	

Testando a mecânica de jogo

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-2.1.1	K2	Classificar os tipos de mecânica de jogo.	X					
GaMe-2.1.2	K2	Diferenciar os testes de mecânica de jogo e de jogabilidade fora da mecânica de jogo.			X			
GaMe-2.1.3	K2	Diferenciar os testes de mecânica central e metamecânica.			X			
GaMe-2.1.4	K2	Diferenciar os testes de mecânica cliente, servidor e cliente-servidor.			X			
GaMe-2.1.5	K2	Dar exemplos de defeitos na mecânica dos jogos.		X				
GaMe-2.2.1	K2	Resumir as principais abordagens e testar objetos em diferentes estágios da criação de um produto de jogo.				X	X	
GaMe-2.2.2	K2	Distinguir a importância de testar a mecânica de jogo.	X					
GaMe-2.2.3	K2	Distinguir a importância de revisar a documentação que descreve a mecânica de jogo.	X					
GaMe-2.2.4	K3	Aplicar as abordagens fundamentais de testar a mecânica de jogo				X		

Testes gráficos

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-3.1.1	K2	Explicar as características do conteúdo gráfico de um produto de jogo.	X					
GaMe-3.1.2	K2	Classificar os tipos de defeitos no conteúdo gráfico.		X				
GaMe-3.2.1	K2	Resumir as principais abordagens dos testes artísticos.				X		
GaMe-3.2.2	K2	Resumir as principais abordagens aos testes técnicos.				X		
GaMe-3.2.3	K2	Resumir as principais abordagens para testes de jogabilidade.				X		
GaMe-3.3.1	K3	Aplicar as abordagens fundamentais dos testes gráficos.			X			
GaMe-3.3.2	K2	Explicar a importância dos gráficos de teste para a validade histórica.		X				
GaMe-3.4.1	K2	Resumir o uso de ferramentas de teste gráfico.					X	

Teste de som

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-4.1.1	K1	Características do conteúdo de som de um produto de jogo.	X					
GaMe-4.2.1	K1	Recordar os tipos de defeitos no conteúdo sonoro.		X				
GaMe-4.2.2	K2	Classificar os defeitos no conteúdo de som		X				

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-4.3.1	K2	Resumir as principais abordagens para os testes de conteúdo-auditável.				X		
GaMe-4.3.2	K2	Resumir as principais abordagens para testar a mixagem de música e sons.				X		
GaMe-4.3.3	K2	Resumir as principais abordagens para testar a composição musical.				X		
GaMe-4.4.1	K2	Explicar os níveis de teste de conteúdo áudio musical.	X					
GaMe-4.4.2	K1	Relembrar as características de integração de sons no cliente.	X					
GaMe-4.4.3	K1	Relembrar áreas de responsabilidade de testes sólidos.						X
GaMe-4.4.4	K3	Aplicar abordagens para testes de som.			X			
GaMe-4.5.1	K2	Resumir o uso de ferramentas de teste de som.					X	

Teste de nível de jogo

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-5.1.1	K1	Recordar os componentes do nível de jogo.	X					
GaMe-5.1.2	K2	Classificar os defeitos típicos dos níveis de jogo.		X				
GaMe-5.2.1	K2	Resumir os testes realizados em vários estágios da criação dos níveis de jogo.				X		X
GaMe-5.2.2	K2	Comparar as áreas de responsabilidade dos especialistas que participam nos testes de nível de jogo.						X
GaMe-5.3.1	K2	Resumir o uso de ferramentas para testar os níveis de jogo.					X	

Teste de controladores de jogo

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-6.1.1	K2	Classificar os dispositivos típicos de entrada e os específicos.	X					
GaMe-6.1.2	K2	Dar exemplos de diferentes dispositivos de entrada em termos de sua aplicação.		X				
GaMe-6.1.3	K1	Recordar os diferentes tipos de controladores de jogo.	X					
GaMe-6.1.4	K2	Classificar os defeitos em um produto de jogo relacionados com as especificidades dos controladores de jogo, e as possíveis causas de sua ocorrência.		X				
GaMe-6.2.1	K2	Dar exemplos de condições de teste a serem cobertas ao testar controladores de jogo.		X				
GaMe-6.2.2	K2	Classificar tarefas para especialistas em UX, testadores e projetistas de jogos durante os testes de jogo.						X
GaMe-6.3.1	K2	Resumir o uso de ferramentas para testar o comportamento dos controladores de jogo.					X	

Teste de regionalização

Seção	K	Descrição	GaMe 1	GaMe 2	GaMe 3	GaMe 4	GaMe 5	GaMe 6
GaMe-7.1.1	K1	Reconhecer os principais objetivos da internacionalização e da regionalização.	X					
GaMe-7.1.2	K1	Relembrar as etapas do teste de regionalização.			X			
GaMe-7.1.3	K2	Comparar as capacidades de internacionalização e regionalização.		X				
GaMe-7.2.1	K2	Classificar os defeitos de regionalização e suas causas.		X				
GaMe-7.3.1	K1	Reconhecer o total e parcial testes de regionalização.				X		
GaMe-7.3.2	K3	Classificar os tipos de testes de regionalização.						X
GaMe-7.3.3	K2	Resumir tarefas de teste para um escritor, editor, tradutor e testador de regionalização.						X
GaMe-7.4.1	K2	Resumir o uso de ferramentas para testar a regionalização de jogos.					X	

Apêndice C – Notas da Versão

Este é o primeiro lançamento do syllabus de Teste de Jogos. O desenvolvimento deste módulo começou em 2020 como a reação no crescente mercado do software de entretenimento iterativo, onde os testes são um campo altamente técnico, mas não havia um padrão para teste de jogos.

A equipe de desenvolvimento do Certified Tester Game Testing avalia o número de especialistas na área de desenvolvimento de jogos em mais de 400.000.

O módulo Certified Tester Game Testing dá a oportunidade de adquirir uma certificação reconhecida internacionalmente para ser reconhecida entre colegas, empregadores e clientes.

Apêndice D - Teste de jogos e outros termos específicos

Termo	Nome do Termo	Definição
accelerometer	acelerômetro	Um sensor que permite ao software determinar e transmitir informações sobre como o controlador de jogo está localizado no espaço em um dado momento no tempo.
cultural adaptation	adaptação cultural	O processo de adaptação das condições do ambiente dos jogos para criar as características de uma determinada cultura.
help	ajuda	Documentação que explica as características de um programa e ajuda o usuário a entender suas capacidades.
ambient	ambiente	O ambiente que acompanha um determinado local, situação, ou estágio de um jogo.
video game environment	ambiente de jogo	O conteúdo do jogo: jogabilidade, nível do jogo, objetos do jogo e, em alguns casos, a história do jogo.
animation	animação	Uma técnica para criar a ilusão de imagens em movimento usando uma sequência de imagens estáticas e substituindo umas às outras em alta frequência.
first person shooter	atirador em primeira pessoa	Um subgênero de jogos de tiro centrado em armas e outros combates baseados em armas na perspectiva da primeira pessoa, com o jogador experimentando a ação através dos olhos do protagonista.
grey box	caixa cinza	Um estágio no desenvolvimento de nível de jogo no qual um "rascunho" de layout 3D é criado a partir de formas não definidas de uma cor para testar o processo do jogo
loot box	loot box	Um prêmio na forma de objeto(s) virtual(ais) para completar uma partida, ganhar um nível de experiência, ou outra realização dentro do jogo. [Pós Wikipédia]
setting	cenário	O ambiente de jogo no qual a ação se desenvolve.
boss	chefe	Um oponente controlado por computador que é muito mais difícil de derrotar do que um inimigo normal em um jogo.
color coding	codificação por cores	O processo de marcação de objetos com cores diferentes como meio de identificação.
collision	colisão	Uma tecnologia responsável pela interseção de objetos de jogo.
video game console	console de jogo	Um dispositivo eletrônico projetado para jogos
video game controller	controlador de jogo	Um dispositivo usado que fornece a entrada para um jogo.
sound discontinuity	descontinuidade sonora	Uma inesperada ausência prolongada de som ou um corte brusco de som.
video game design	projeto de jogo	O processo de criação da forma e do conteúdo da jogabilidade do jogo que está sendo desenvolvido.
video game designer	desenvolvedor de jogos	A pessoa responsável pelo desenvolvimento das regras e do conteúdo do jogo. Ver também: o projeto do jogo.
distortion	distorção	Mudança na forma de onda sonora durante o processamento. [Pós The Oxford Companion para a língua inglesa]
video game design document	documento de projeto de jogo	Documentação contendo uma descrição detalhada do jogo que está sendo desenvolvido.
level editor	editor de nível	Software que é usado para criar e editar locais e ambientes dentro de um jogo.
publisher	editora	Uma empresa que financia, distribui e comercializa um jogo. [Pós Wikipédia]
binaural effect	efeito binaural	Uma ilusão auditiva que é observada quando os estímulos oscilatórios são entregues em duas frequências adjacentes a cada orelha ao mesmo tempo.

Termo	Nome do Termo	Definição
sound effect	efeito sonoro	Um som diferente da fala ou música feita artificialmente para um objeto de jogo.
visual effect (VFX)	efeito visual (VFX)	A criação ou manipulação de qualquer imagem na tela que não exista fisicamente na vida real.
video game state	estado do jogo	O valor de todos os parâmetros e variáveis que descrevem todos os objetos de um jogo em um determinado momento no tempo.
concept stage	estágio de conceito	Um estágio inicial de planejamento de um projeto de desenvolvimento de jogo, que se concentra na criação de conceitos centrais e na redação de documentos de projeto inicial que descrevem o futuro jogo.
historical accuracy	exatidão histórica	Conformidade do conteúdo dentro do jogo com seu conteúdo correspondente na vida real.
post-production stage	fase de pós-produção	O processo de manutenção e distribuição de um jogo, e o marketing após seu lançamento.
pre-production stage	fase de pré-produção	O processo de planejamento e documentação dos elementos de um jogo.
production stage	fase de produção	O processo de desenvolvimento de um jogo.
gamepad	gamepad	Um tipo de controlador de jogo bimanual usado com consoles de jogo. Ver também: controlador de jogo.
trigger	gatilho	Uma ou mais ações que iniciam um evento.
structural geometry	geometria estrutural	A tecnologia que proporciona o relevo do terreno e a superfície sobre a qual os personagens do jogo podem se mover.
gyroscope	giroscópio	Um sensor que mede a orientação e a velocidade angular de um corpo em sua estrutura de repouso
hacking	hacking	Uma maneira desonesta de ganhar vantagem em um jogo.
hit box	hit box	Um modelo simplificado de objeto 3D para detecção e manipulação de colisão de objetos dentro de um jogo. Sinônimo: área afetada, caixa de resposta
scene lighting	iluminação de cena	A quantidade, tamanho, cor e aspereza da luz que envolve um personagem para combinar com a cena de um jogo.
computer-generated imagery (CGI)	imagens geradas por computador (CGI)	A aplicação de computação gráfica 3D para criar efeitos especiais.
gameplay	jogabilidade	As regras que descrevem a interação entre o mundo dos jogos e o jogador.
player versus player (PvP)	jogador contra jogador (PvP)	Um modo de jogo no qual um jogador interage com um ou mais jogadores.
quest	jogos de procurar	Um gênero de videogames que inclui atividades baseadas em objetivos, seja para uma história interativa ou para o avanço do nível do personagem. [Pós Wikipédia]
lag	lag	Um atraso na performance dos jogos devido a uma conexão lenta à Internet ou vazamento de memória ou uso severo de CPU/RAM/HDD (por exemplo, Windows/antivírus atualizados em segundo plano). Sinônimo: defasagem
locale	Regionalização	Um conjunto de opções que definem o idioma, a região e qualquer preferência especial de variante na interface do usuário de um jogo. Sinônimo: regionalização.
store	loja	Um serviço que fornece distribuição digital de videogames e outros conteúdos de jogos.
inaccessible place	lugar inacessível	Um local no mundo dos jogos que é colocado incorretamente à disposição de um jogador.
fighting	luta	Um gênero de jogo que simula o combate corpo a corpo de um pequeno número de personagens dentro de um espaço e tempo limitados.

Termo	Nome do Termo	Definição
mapping	mapeamento	Uma disciplina no desenvolvimento de jogos para criar um nível em um jogo. Ver também: nível de jogo.
core mechanics	mecânica central	Mecânica de jogo que define a experiência de jogo pretendida para o jogador. Ver também: a mecânica dos jogos.
video game mechanics	mecânica de jogos	As ações dentro de um jogo definidas por certas regras.
client mechanics	mecânica do cliente	Mecânica dos jogos que funcionam no lado do cliente de jogo. Ver também: a mecânica dos jogos.
server mechanics	mecânica do servidor	Mecânica de jogo que funciona no servidor de jogo. Ver também: mecânica dos jogos.
meta mechanics	meta mecânica	Mecânica de jogo que define como um jogador deve usar um jogo. Ver também: a mecânica dos jogos.
3D model	modelo 3D	Um modelo 3D de um objeto utilizando formas matemáticas visuais.
morphing	morphing	Um efeito visual que cria a impressão de uma transformação perfeita de um objeto em outro.
video game engine	motor de jogo	O software no qual todos os outros componentes de um jogo são construídos.
multiplatform	multiplataforma	A capacidade de usar software em diferentes plataformas.
narrative	narrativa	Uma história veiculada através da mecânica dos jogos. Ver também: design de jogo.
level of detail (LoD)	nível de detalhe (LoD)	Uma tecnologia na qual uma cópia simplificada de um objeto é criada para exibir o mesmo objeto visto a uma distância diferente.
level prototype	nível de protótipo	Uma amostra antecipada de um nível de jogo que é criado como uma prova de conceito. Ver também: nível de jogo.
video game level	nível de jogo	Um local separado dentro do mundo virtual de um jogo.
video game object	objeto de jogo	Qualquer objeto com o qual um jogador possa interagir em um jogo.
mobile object (Mob)	objeto móvel (Mob)	Um inimigo que percorre uma área específica de um jogo.
occlusion	oclusão	O processo que imita a forma como os sons são percebidos em ambientes "fechados".
game character	personagem de jogo	Uma pessoa ou qualquer outra entidade atuando em um jogo.
player character	personagem do jogo	Um personagem em jogos, que é controlado por um jogador, tipicamente um protagonista da trama do jogo. Ver também: personagem do jogo.
non-player character (NPC)	personagem não-jogador (NPC)	Um caráter controlado por computador. Ver também: Mob.
platformer	plataforma	Um gênero de jogo que envolve saltar em plataformas, subir escadas e coletar itens necessários para derrotar inimigos ou completar um nível.
polygon	polígono	Um conjunto de vértices, bordas e faces que definem a forma de um objeto poliédrico em computação gráfica 3D e modelagem de volume.
level design	projeto de nível	Um estágio no desenvolvimento de jogos que envolve a criação de níveis e missões de jogo.
skipping	pulando	Um fundo de áudio defeituoso que soa como partes da trilha sonora é ignorado.
raster	raster	Um tipo de gráfico que representa uma grade de pixels visualizável em uma tela.
video game resource	recurso de jogo	Uma entidade ou característica importante para um jogador de jogo para atingir os objetivos dos jogos.
reverb	reverberar	Um efeito de eco produzido eletronicamente. [Pós Merriam-Webster]
rigging	rigging	O processo de construção de um esqueleto modelo em um jogo para animá-lo depois.

Termo	Nome do Termo	Definição
role-playing game (RPG)	role-playing game (RPG)	Um gênero de jogo no qual um jogador assume os papéis de personagens em um cenário fictício.
skinning	skinning	O processo de ligar um personagem 3D a um esqueleto de modelo dentro de um jogo. Ver também: rigging.
background sound	som de fundo	Música de fundo dentro do jogo para melhorar a atmosfera de um jogo.
speedrunner	speedrunner	Uma tentativa de completar um jogo o mais rápido possível. [Pós Wikipédia]
stick	stick	Um controlador de jogo cuja mobilidade é limitada a dois graus de liberdade. Ver também: controlador de jogo.
frame rate	taxa de quadros	O número de quadros trocados por unidade de tempo.
touchscreen	tela sensível ao toque	Um controlador de jogo que se baseia no toque físico de uma tela.
terrain	terreno	Uma coleção de elementos relacionados à superfície de jogo na qual um personagem de um jogo se move.
texture	textura	Uma imagem raster sobreposta na superfície de um modelo poligonal para dar-lhe cor ou imitar o relevo.
tile	tile	Um pequeno objeto ou fragmento a partir do qual um nível ou outra imagem grande é construída em um jogo.
trackball	trackball	Um controlador de jogo no qual uma bola que gira livremente é usada como dispositivo de entrada para um jogo. Ver também: controlador de jogo.
video game	jogo	Um jogo eletrônico no qual um jogador controla as imagens em uma tela de vídeo. [Merriam-Webster]
racing wheel	volante/pedal	Um tipo de controlador de videogames com duas mãos usado para imitar um volante, pedais e alavanca de câmbio.
volume	volume	O grau de ruído ou a intensidade de um som. [Merriam-Webster]
sound zone	zona sonora	Uma tecnologia que permite a criação de áreas dentro do jogo com propriedades sonoras reais.