Open Dynamic Audio

Um middleware para áudio dinâmico em jogos digitais

Wilson Kazuo Mizutani

Exame de Qualificação
Orientador: Prof. Dr. Fabio Kon

17 de agosto de 2015

Introdução

Conceitos e Ferramentas

Trabalhos relacionados

Proposta

Plano de trabalho

Introdução

Contexto: jogos digitais

Não são programas convencionais

- Objetivo não é obter resultados
- É a experiência da interação
- Imersividade
- Papel da trilha sonora

Contexto: jogos digitais

Não são programas convencionais

- Objetivo não é obter resultados
- É a experiência da interação
- Imersividade
- Papel da trilha sonora

Contexto: jogos digitais

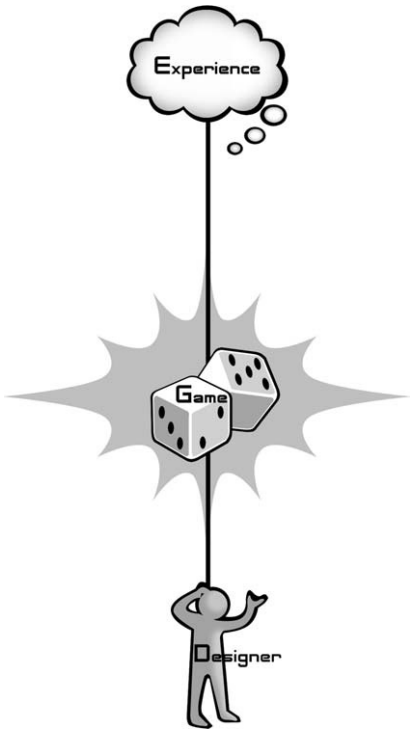
Não são programas convencionais

- Objetivo não é obter resultados
- É a experiência da interação
- Imersividade
- Papel da trilha sonora

Contexto: jogos digitais

Não são programas convencionais

- Objetivo não é obter resultados
- É a experiência da interação
- Imersividade
- Papel da trilha sonora

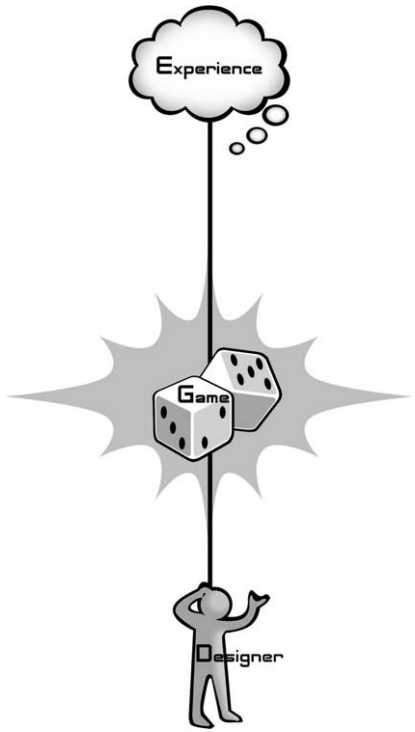


Fonte: SCHELL, Jesse. **The Art of Game Design: A Book of Lenses**. Morgan Kaufmann Publishers Inc., 2008.

Contexto: jogos digitais

Não são programas convencionais

- Objetivo não é obter resultados
- É a experiência da interação
- Imersividade
- Papel da trilha sonora

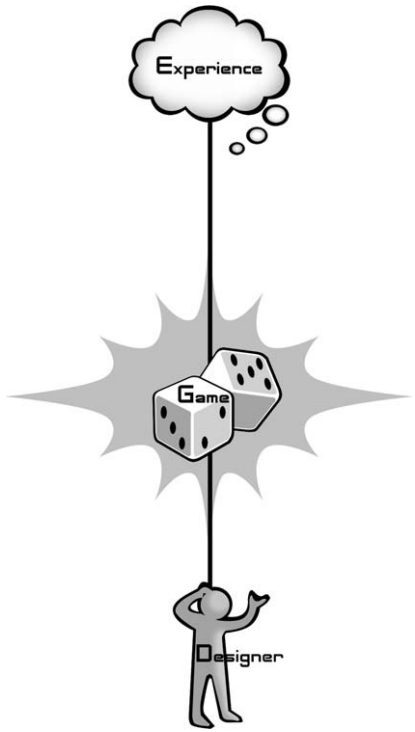


Fonte: SCHELL, Jesse. **The Art of Game Design: A Book of Lenses**. Morgan Kaufmann Publishers Inc., 2008.

Contexto: jogos digitais

Não são programas convencionais

- Objetivo não é obter resultados
- É a experiência da interação
- Imersividade
- Papel da trilha sonora



Fonte: SCHELL, Jesse. **The Art of Game Design: A Book of Lenses**. Morgan Kaufmann Publishers Inc., 2008.

Motivação

Desenvolvimento de um jogo

- Processo multidisciplinar
- Atividades em paralelo
- Como juntar os materiais?

Problemas na entrega de áudio

- Tempo de *feedback*
- Limitações tecnológicas
- Domínios de conhecimento

Solução na forma de middleware para áudio dinâmico

Motivação

Desenvolvimento de um jogo

- Processo multidisciplinar
- Atividades em paralelo
- Como juntar os materiais?

Problemas na entrega de áudio

- Tempo de *feedback*
- Limitações tecnológicas
- Domínios de conhecimento

Solução na forma de middleware para áudio dinâmico

Motivação

Desenvolvimento de um jogo

- Processo multidisciplinar
- Atividades em paralelo
- Como juntar os materiais?

Problemas na entrega de áudio

- Tempo de *feedback*
- Limitações tecnológicas
- Domínios de conhecimento

Solução na forma de middleware para áudio dinâmico

Motivação

Desenvolvimento de um jogo

- Processo multidisciplinar
- Atividades em paralelo
- Como juntar os materiais?

Problemas na entrega de áudio

- Tempo de *feedback*
- Limitações tecnológicas
- Domínios de conhecimento

Solução na forma de middleware para áudio dinâmico

Motivação

Desenvolvimento de um jogo

- Processo multidisciplinar
- Atividades em paralelo
- Como juntar os materiais?

Problemas na entrega de áudio

- Tempo de *feedback*
- Limitações tecnológicas
- Domínios de conhecimento

Solução na forma de middleware para áudio dinâmico

Motivação

Desenvolvimento de um jogo

- Processo multidisciplinar
- Atividades em paralelo
- Como juntar os materiais?

Problemas na entrega de áudio

- Tempo de *feedback*
- Limitações tecnológicas
- Domínios de conhecimento

Solução na forma de middleware para áudio dinâmico

Objetivos

1. Investigar soluções de áudio dinâmico já existentes
2. Desenvolver um middleware de áudio dinâmico
3. Avaliar qualitativamente o uso do middleware em jogos
4. Analisar a praticidade do middleware através de entrevistas

Objetivos

1. Investigar soluções de áudio dinâmico já existentes
2. Desenvolver um middleware de áudio dinâmico
3. Avaliar qualitativamente o uso do middleware em jogos
4. Analisar a praticidade do middleware através de entrevistas

Objetivos

1. Investigar soluções de áudio dinâmico já existentes
2. Desenvolver um middleware de áudio dinâmico
3. Avaliar qualitativamente o uso do middleware em jogos
4. Analisar a praticidade do middleware através de entrevistas

Objetivos

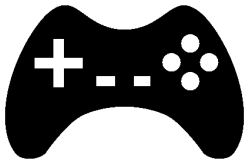
1. Investigar soluções de áudio dinâmico já existentes
2. Desenvolver um middleware de áudio dinâmico
3. Avaliar qualitativamente o uso do middleware em jogos
4. Analisar a praticidade do middleware através de entrevistas

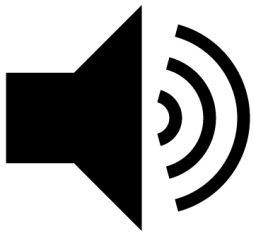
Objetivos

1. Investigar soluções de áudio dinâmico já existentes
2. Desenvolver um middleware de áudio dinâmico
3. Avaliar qualitativamente o uso do middleware em jogos
4. Analisar a praticidade do middleware através de entrevistas

Conceitos e Ferramentas

Os domínios



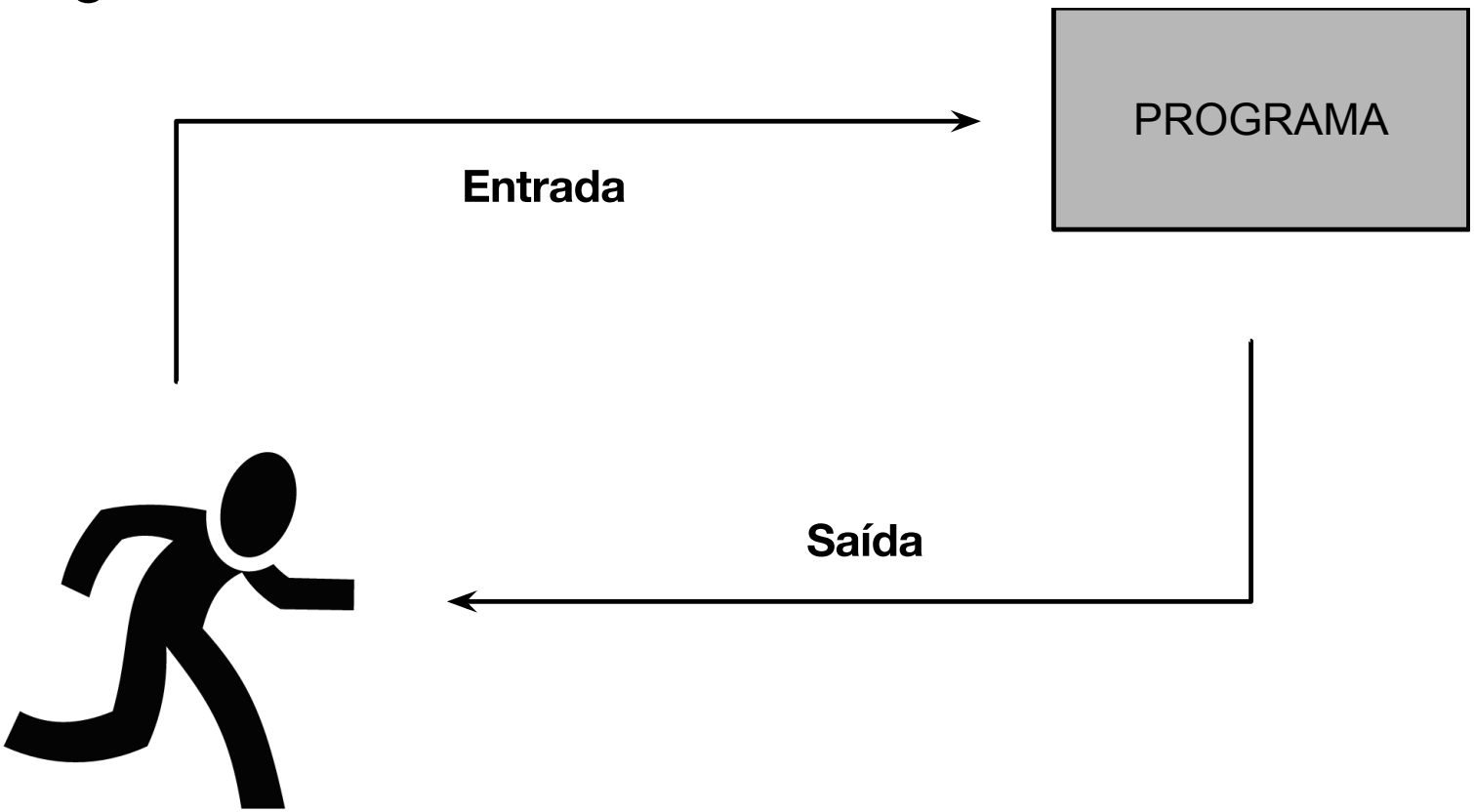


Jogos digitais

Programas **interativos**:

Jogos digitais

Programas **interativos**:



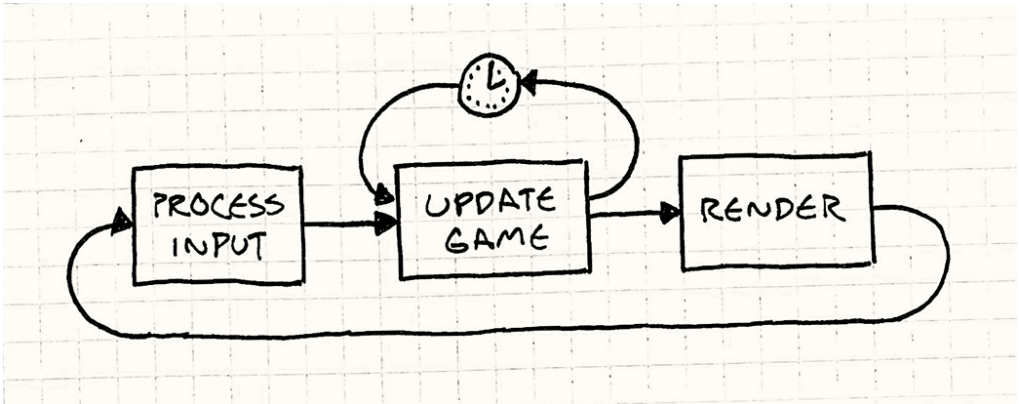
Jogos digitais: padrão arquitetural

Todo jogo possui um **Game Loop**

- Taxa de atualização deve ser alta o suficiente ($30Hz$ - $60Hz$)
- A renderização do áudio também deve seguir esse padrão

Jogos digitais: padrão arquitetural

Todo jogo possui um **Game Loop**

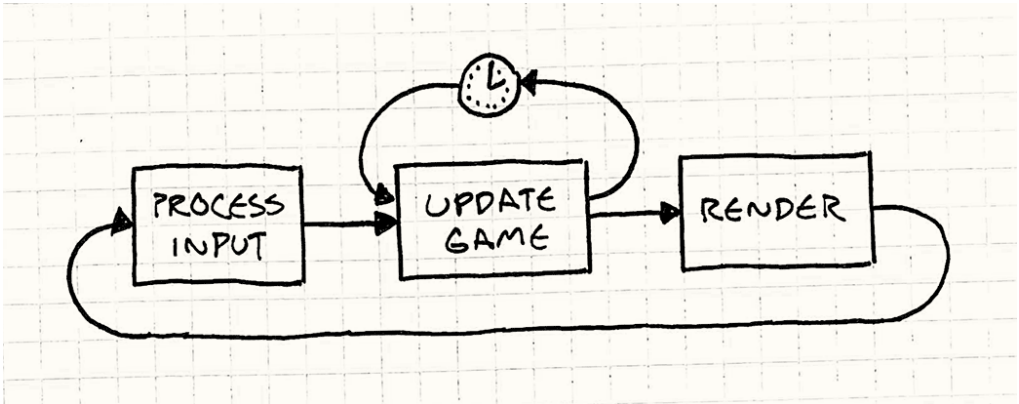


- Taxa de atualização deve ser alta o suficiente ($30Hz$ - $60Hz$)
- A renderização do áudio também deve seguir esse padrão

Fonte: Robert Nystrom. **Game Programming Patterns**. Genever Benning, 2014.

Jogos digitais: padrão arquitetural

Todo jogo possui um **Game Loop**

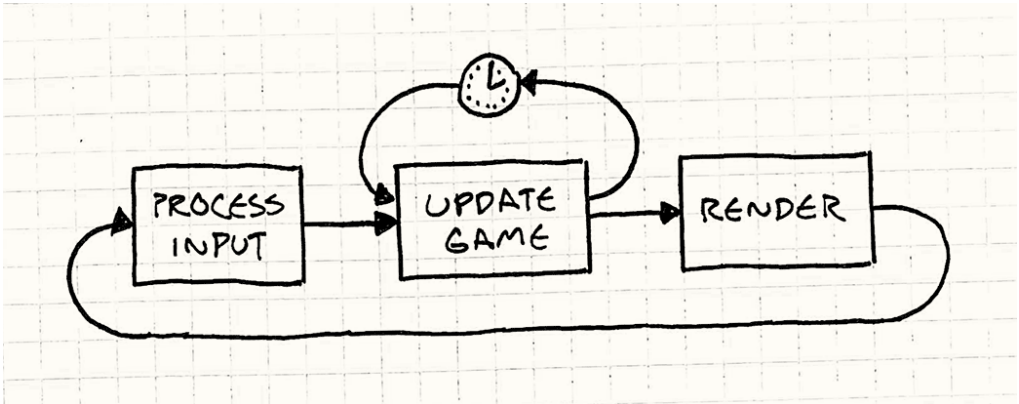


- Taxa de atualização deve ser alta o suficiente ($30Hz$ - $60Hz$)
- A renderização do áudio também deve seguir esse padrão

Fonte: Robert Nystrom. **Game Programming Patterns**. Genever Benning, 2014.

Jogos digitais: padrão arquitetural

Todo jogo possui um **Game Loop**



- Taxa de atualização deve ser alta o suficiente ($30Hz$ - $60Hz$)
- A renderização do áudio também deve seguir esse padrão

Fonte: Robert Nystrom. **Game Programming Patterns**. Genever Benning, 2014.

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: C++

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: C++

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: C++

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: C++

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: C++

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: C++

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: `C++`

Jogos digitais: ferramentas de desenvolvimento

1. **Bibliotecas de programação**
   - Coleções de rotinas sobre um domínio específico
   - Reutilizáveis
   - Exemplos: OpenGL, OpenAL, libpng, zlib
2. **Arcabouços**
   - Colaboração de ferramentas sob uma interface unificada
   - Mais generalista
   - Exemplos: LÖVE, Corona SDK
3. **Motores**
   - Sistemas que assumem controle sobre parte da aplicação
   - Podem ser generalistas ou específicos
   - Exemplos: Unity3D, OGRE

Linguagem de programação principal: `C++`

Áudio digital

Representação digital do som

- Sequência de números
  - Variam ao longo do tempo
  - Variação da pressão do ar
  - Chamados de **amostras**
- Resolução da sequência
  - Valores por unidade de tempo
  - Chamada **taxa de amostragem**
  - Medida em **Hz**
- Precisão dos valores
  - Representação binária usada
  - Inteiros vs. ponto flutuante
  - Quantidade de bytes por valor

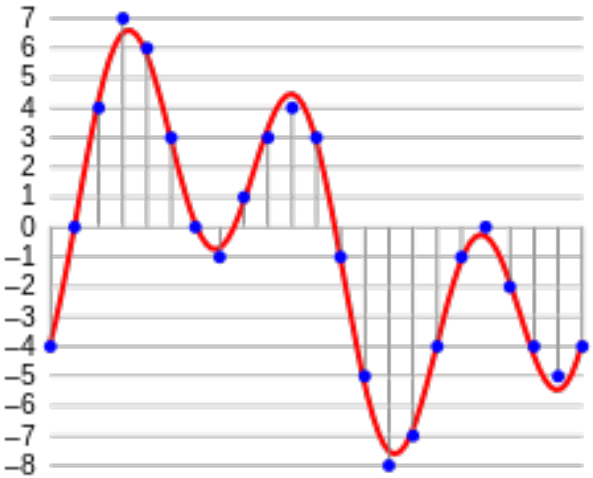
Áudio digital

Representação digital do som

- Sequência de números
  - Variam ao longo do tempo
  - Variação da pressão do ar
  - Chamados de **amostras**
- Resolução da sequência
  - Valores por unidade de tempo
  - Chamada **taxa de amostragem**
  - Medida em **Hz**
- Precisão dos valores
  - Representação binária usada
  - Inteiros vs. ponto flutuante
  - Quantidade de bytes por valor

Áudio digital

Representação digital do som

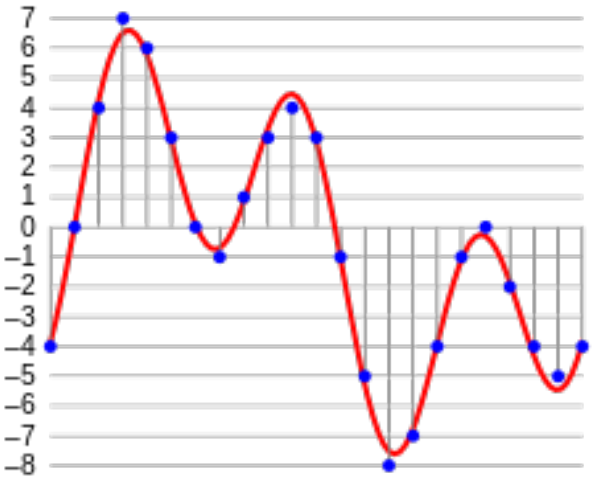


- Sequência de números
  - Variam ao longo do tempo
  - Variação da pressão do ar
  - Chamados de **amostras**
- Resolução da sequência
  - Valores por unidade de tempo
  - Chamada **taxa de amostragem**
  - Medida em **Hz**
- Precisão dos valores
  - Representação binária usada
  - Inteiros vs. ponto flutuante
  - Quantidade de bytes por valor

Fonte: Aquegg. 4-bit-linear-pcm.svg. Licensa CC BY-SA 3.0.

Áudio digital

Representação digital do som

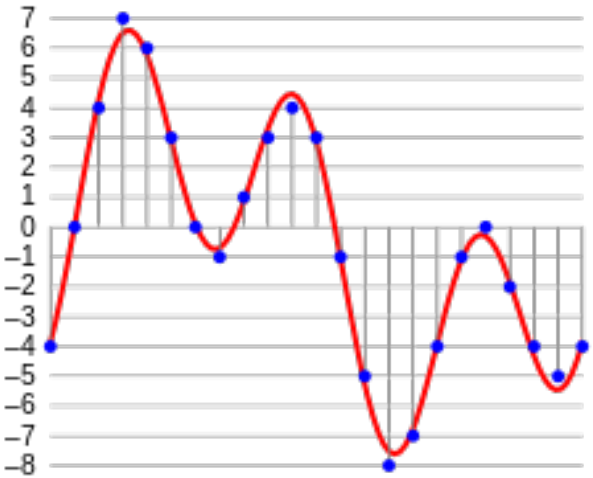


- Sequência de números
  - Variam ao longo do tempo
  - Variação da pressão do ar
  - Chamados de **amostras**
- Resolução da sequência
  - Valores por unidade de tempo
  - Chamada **taxa de amostragem**
  - Medida em **Hz**
- Precisão dos valores
  - Representação binária usada
  - Inteiros vs. ponto flutuante
  - Quantidade de bytes por valor

Fonte: Aquegg. 4-bit-linear-pcm.svg. Licensa CC BY-SA 3.0.

Áudio digital

Representação digital do som



- Sequência de números
  - Variam ao longo do tempo
  - Variação da pressão do ar
  - Chamados de **amostras**
- Resolução da sequência
  - Valores por unidade de tempo
  - Chamada **taxa de amostragem**
  - Medida em **Hz**
- Precisão dos valores
  - Representação binária usada
  - Inteiros vs. ponto flutuante
  - Quantidade de bytes por valor

Fonte: Aquegg. 4-bit-linear-pcm.svg. Licensa CC BY-SA 3.0.

Áudio digital: implementação

Como um programa pode produzir som?

- Placa de som
  1. O programa envia a sequência de amostras
  2. A placa usa o formato especificado
  3. O sinal digital é convertido para analógico
- Ferramenta: OpenAL
  - Especificação de API para esse procedimento
  - Implementação usada: OpenAL Soft
  - Multiplataforma
  - Efeitos de espacialidade em 3D

Áudio digital: implementação

Como um programa pode produzir som?

- Placa de som
  1. O programa envia a sequência de amostras
  2. A placa usa o formato especificado
  3. O sinal digital é convertido para analógico
- Ferramenta: OpenAL
  - Especificação de API para esse procedimento
  - Implementação usada: OpenAL Soft
  - Multiplataforma
  - Efeitos de espacialidade em 3D

Áudio digital: implementação

Como um programa pode produzir som?

- Placa de som
  1. O programa envia a sequência de amostras
  2. A placa usa o formato especificado
  3. O sinal digital é convertido para analógico
- Ferramenta: OpenAL
  - Especificação de API para esse procedimento
  - Implementação usada: OpenAL Soft
  - Multiplataforma
  - Efeitos de espacialidade em 3D

Áudio digital: implementação

Como um programa pode produzir som?

- Placa de som
  1. O programa envia a sequência de amostras
  2. A placa usa o formato especificado
  3. O sinal digital é convertido para analógico
- Ferramenta: OpenAL
  - Especificação de API para esse procedimento
  - Implementação usada: OpenAL Soft
  - Multiplataforma
  - Efeitos de espacialidade em 3D

Áudio digital: implementação

Como um programa pode produzir som?

- Placa de som
  1. O programa envia a sequência de amostras
  2. A placa usa o formato especificado
  3. O sinal digital é convertido para analógico
- Ferramenta: OpenAL
  - Especificação de API para esse procedimento
  - Implementação usada: OpenAL Soft
  - Multiplataforma
  - Efeitos de espacialidade em 3D

Áudio digital: implementação

Como um programa pode produzir som?

- Placa de som
  1. O programa envia a sequência de amostras
  2. A placa usa o formato especificado
  3. O sinal digital é convertido para analógico
- Ferramenta: OpenAL
  - Especificação de API para esse procedimento
  - Implementação usada: OpenAL Soft
  - Multiplataforma
  - Efeitos de espacialidade em 3D

Fontes:

OpenAL - http://openal.org/

OpenAL Soft - http://kcat.strangesoft.net/openal.html

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Áudio digital: origem do sinal

Arquivos de áudio

- Armazenam o sinal em disco
- **Áudio estático**

Alternativa: processamento digital de sinais (DSP)

- **Sintetiza ou modifica o sinal em tempo real**
- Linguagens de programação visuais:
  - Max/MSP (proprietário)
  - *PureData* (livre)
- Distribuição em *patches*
- Incorporação em outros programas (libpd)

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras

Mídias lineares vs. não lineares

- Entender as diferenças e semelhanças
- Evolução histórica

O que nos interessa:

- ~~Qualidade da trilha sonora~~
- Solução computacional que torne isso possível
- Características comuns

Fontes:

Eugênio Matos. **A Arte de Compor Música para o Cinema**. Senac, Brasília, DF, Brasil, 2014.

Karen Collins. **Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design**. The MIT Press, 2008.

Trilhas sonoras: características

Composta por

- Diálogos
- Efeitos sonoros
- Música

Diegese

- Diegética
- Extra-diegética
- Trans-diegética

Com respeito ao jogo, pode ser

- Interativa
- Adaptativa

Fonte: Lucas Correia Meneguette e Pontifícia Universidade Católica De São Paulo. **Áudio Dinâmico Para Games: Conceitos Fundamentais E Procedimentos De Composição Adaptativa**. Simpório Brasileiro de Games, páginas 1–10, 2011.

Trilhas sonoras: características

Composta por

- Diálogos
- Efeitos sonoros
- Música

Diegese

- Diegética
- Extra-diegética
- Trans-diegética

Com respeito ao jogo, pode ser

- Interativa
- Adaptativa

Fonte: Lucas Correia Meneguette e Pontifícia Universidade Católica De São Paulo. **Áudio Dinâmico Para Games: Conceitos Fundamentais E Procedimentos De Composição Adaptativa**. Simpório Brasileiro de Games, páginas 1–10, 2011.

Trilhas sonoras: características

Composta por

- Diálogos
- Efeitos sonoros
- Música

Diegese

- Diegética
- Extra-diegética
- Trans-diegética

Com respeito ao jogo, pode ser

- Interativa
- Adaptativa

Fonte: Lucas Correia Meneguette e Pontifícia Universidade Católica De São Paulo. **Áudio Dinâmico Para Games: Conceitos Fundamentais E Procedimentos De Composição Adaptativa**. Simpório Brasileiro de Games, páginas 1–10, 2011.

Trilhas sonoras: características

Composta por

- Diálogos
- Efeitos sonoros
- Música

Diegese

- Diegética
- Extra-diegética
- Trans-diegética

Com respeito ao jogo, pode ser

- Interativa
- Adaptativa

Fonte: Lucas Correia Meneguette e Pontifícia Universidade Católica De São Paulo. **Áudio Dinâmico Para Games: Conceitos Fundamentais E Procedimentos De Composição Adaptativa**. Simpório Brasileiro de Games, páginas 1–10, 2011.

Trilhas sonoras: características

Composta por

- Diálogos
- Efeitos sonoros
- Música

Diegese

- Diegética
- Extra-diegética
- Trans-diegética

Com respeito ao jogo, pode ser

- Interativa
- Adaptativa

Fonte: Lucas Correia Meneguette e Pontifícia Universidade Católica De São Paulo. **Áudio Dinâmico Para Games: Conceitos Fundamentais E Procedimentos De Composição Adaptativa**. Simpório Brasileiro de Games, páginas 1–10, 2011.

Trabalhos relacionados

Trabalhos acadêmicos

Scott fala das dificuldades da área:

- Compor músicas para mídias visuais não é fácil
- Principalmente quando não é linear
- Tecnologias ajudam, mas trazem outros custos

Meneguette ressalva a importância da pesquisa:

> *[...] há um desafio interessante aos compositores e aos pesquisadores em áudio dinâmico: como descrever as potenciais situações que envolvam a sonoridade no jogo, levando-se em consideração questões estéticas e tecnológicas [...]*

Fontes:

Nathan Scott. **Music to middleware: The growing challenges of the game music composer**. Em Proceedings of the 2014 Conference on Interactive Entertainment, IE2014, páginas 34:1–34:3, New York, NY, USA, 2014. ACM.

Lucas Correia Meneguette. **Situações Sonoras e Jogos Digitais**. Simpório Brasileiro de Games, páginas 30–33, 2013.

Trabalhos acadêmicos

Scott fala das dificuldades da área:

- Compor músicas para mídias visuais não é fácil
- Principalmente quando não é linear
- Tecnologias ajudam, mas trazem outros custos

Meneguette ressalva a importância da pesquisa:

> *[...] há um desafio interessante aos compositores e aos pesquisadores em áudio dinâmico: como descrever as potenciais situações que envolvam a sonoridade no jogo, levando-se em consideração questões estéticas e tecnológicas [...]*

Fontes:

Nathan Scott. **Music to middleware: The growing challenges of the game music composer**. Em Proceedings of the 2014 Conference on Interactive Entertainment, IE2014, páginas 34:1–34:3, New York, NY, USA, 2014. ACM.

Lucas Correia Meneguette. **Situações Sonoras e Jogos Digitais**. Simpório Brasileiro de Games, páginas 30–33, 2013.

Trabalhos acadêmicos

Scott fala das dificuldades da área:

- Compor músicas para mídias visuais não é fácil
- Principalmente quando não é linear
- Tecnologias ajudam, mas trazem outros custos

Meneguette ressalva a importância da pesquisa:

> *[...] há um desafio interessante aos compositores e aos pesquisadores em áudio dinâmico: como descrever as potenciais situações que envolvam a sonoridade no jogo, levando-se em consideração questões estéticas e tecnológicas [...]*

Fontes:

Nathan Scott. **Music to middleware: The growing challenges of the game music composer**. Em Proceedings of the 2014 Conference on Interactive Entertainment, IE2014, páginas 34:1–34:3, New York, NY, USA, 2014. ACM.

Lucas Correia Meneguette. **Situações Sonoras e Jogos Digitais**. Simpório Brasileiro de Games, páginas 30–33, 2013.

Trabalhos acadêmicos

Scott fala das dificuldades da área:

- Compor músicas para mídias visuais não é fácil
- Principalmente quando não é linear
- Tecnologias ajudam, mas trazem outros custos

Meneguette ressalva a importância da pesquisa:

> *[...] há um desafio interessante aos compositores e aos pesquisadores em áudio dinâmico: como descrever as potenciais situações que envolvam a sonoridade no jogo, levando-se em consideração questões estéticas e tecnológicas [...]*

Fontes:

Nathan Scott. **Music to middleware: The growing challenges of the game music composer**. Em Proceedings of the 2014 Conference on Interactive Entertainment, IE2014, páginas 34:1–34:3, New York, NY, USA, 2014. ACM.

Lucas Correia Meneguette. **Situações Sonoras e Jogos Digitais**. Simpório Brasileiro de Games, páginas 30–33, 2013.

Trabalhos acadêmicos: música adaptativa

Mapeamento computacional de sentimentos em manipulações sonoro-musicais

- Livingstone et al.: *Computacional Music Emotion Rule System*
  - Modelo positivo-negativo × ativo-passivo
  - Usa MIDI
- Eladhari et al.: *Mind Module*
  - Modelo humor interno × humor externo
  - Usa áudio pré-renderizado

Fontes:

Steven R. Livingstone, Ralf Muhlberger, Andrew R. Brown e William F. Thompson. **Changing musical emotion: A computational rule system for modifying score and performance**. Computer Music Journal, 34(1):41–64, Março 2010.

Mirjam Eladhari, Rik Nieuwdorp e Mikael Fridenfalk. **The soundtrack of your mind: Mind music - adaptive audio for game characters**. Em Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE '07, New York, NY, USA, 2006. ACM.

Trabalhos acadêmicos: música adaptativa

Mapeamento computacional de sentimentos em manipulações sonoro-musicais

- Livingstone et al.: *Computacional Music Emotion Rule System*
  - Modelo positivo-negativo × ativo-passivo
  - Usa MIDI
- Eladhari et al.: *Mind Module*
  - Modelo humor interno × humor externo
  - Usa áudio pré-renderizado

Fontes:

Steven R. Livingstone, Ralf Muhlberger, Andrew R. Brown e William F. Thompson. **Changing musical emotion: A computational rule system for modifying score and performance**. Computer Music Journal, 34(1):41–64, Março 2010.

Mirjam Eladhari, Rik Nieuwdorp e Mikael Fridenfalk. **The soundtrack of your mind: Mind music - adaptive audio for game characters**. Em Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE '07, New York, NY, USA, 2006. ACM.

Trabalhos acadêmicos: música adaptativa

Mapeamento computacional de sentimentos em manipulações sonoro-musicais

- Livingstone et al.: *Computacional Music Emotion Rule System*
  - Modelo positivo-negativo × ativo-passivo
  - Usa MIDI
- Eladhari et al.: *Mind Module*
  - Modelo humor interno × humor externo
  - Usa áudio pré-renderizado

Fontes:

Steven R. Livingstone, Ralf Muhlberger, Andrew R. Brown e William F. Thompson. **Changing musical emotion: A computational rule system for modifying score and performance**. Computer Music Journal, 34(1):41–64, Março 2010.

Mirjam Eladhari, Rik Nieuwdorp e Mikael Fridenfalk. **The soundtrack of your mind: Mind music - adaptive audio for game characters**. Em Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE '07, New York, NY, USA, 2006. ACM.

Trabalhos acadêmicos: música adaptativa

Mapeamento computacional de sentimentos em manipulações sonoro-musicais

- Livingstone et al.: *Computacional Music Emotion Rule System*
  - Modelo positivo-negativo $\times$ ativo-passivo
  - Usa MIDI
- Eladhari et al.: *Mind Module*
  - Modelo humor interno $\times$ humor externo
  - Usa áudio pré-renderizado

Fontes:

Steven R. Livingstone, Ralf Muhlberger, Andrew R. Brown e William F. Thompson. **Changing musical emotion: A computational rule system for modifying score and performance**. Computer Music Journal, 34(1):41–64, Março 2010.

Mirjam Eladhari, Rik Nieuwdorp e Mikael Fridenfalk. **The soundtrack of your mind: Mind music - adaptive audio for game characters**. Em Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE '07, New York, NY, USA, 2006. ACM.

Trabalhos acadêmicos: música adaptativa

Mapeamento computacional de sentimentos em manipulações sonoro-musicais

- Livingstone et al.: *Computacional Music Emotion Rule System*
  - Modelo positivo-negativo × ativo-passivo
  - Usa MIDI
- Eladhari et al.: *Mind Module*
  - Modelo humor interno × humor externo
  - Usa áudio pré-renderizado

Fontes:

Steven R. Livingstone, Ralf Muhlberger, Andrew R. Brown e William F. Thompson. **Changing musical emotion: A computational rule system for modifying score and performance**. Computer Music Journal, 34(1):41–64, Março 2010.

Mirjam Eladhari, Rik Nieuwdorp e Mikael Fridenfalk. **The soundtrack of your mind: Mind music - adaptive audio for game characters**. Em Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, ACE '07, New York, NY, USA, 2006. ACM.

Trabalhos acadêmicos: música adaptativa

O que aprendemos com esses autores:

- ~~A relação entre emoções e música~~
- Manipulações típicas sobre o áudio
  - Volume
  - Altura
  - Timbre
  - Andamento
  - Ataque
  - Melodia
- MIDI $\times$ áudio digital

Trabalhos acadêmicos: música adaptativa

O que aprendemos com esses autores:

- ~~A relação entre emoções e música~~
- Manipulações típicas sobre o áudio
  - Volume
  - Altura
  - Timbre
  - Andamento
  - Ataque
  - Melodia
- MIDI $\times$ áudio digital

Trabalhos acadêmicos: música adaptativa

O que aprendemos com esses autores:

- ~~A relação entre emoções e música~~
- Manipulações típicas sobre o áudio
  - Volume
  - Altura
  - Timbre
  - Andamento
  - Ataque
  - Melodia
- MIDI $\times$ áudio digital

Trabalhos acadêmicos: música adaptativa

O que aprendemos com esses autores:

- ~~A relação entre emoções e música~~
- Manipulações típicas sobre o áudio
  - Volume
  - Altura
  - Timbre
  - Andamento
  - Ataque
  - Melodia
- MIDI $\times$ áudio digital

Trabalhos acadêmicos: música adaptativa

O que aprendemos com esses autores:

- ~~A relação entre emoções e música~~
- Manipulações típicas sobre o áudio
  - Volume
  - Altura
  - Timbre
  - Andamento
  - Ataque
  - Melodia
- MIDI $\times$ áudio digital

Trabalhos acadêmicos: aplicações de áudio dinâmico

Algumas outras pesquisas interessantes

- *Audio Games*
- Música que controla fenômenos no jogo
- Música composta pelas ações do jogo
- Aplicação de técnicas de *DJing*
- Síntese procedimental de efeitos sonoros físicos
- Propagação do som em ambientes dinâmicos

Como podemos contemplar essas ténicas em nossa proposta?

Trabalhos acadêmicos: aplicações de áudio dinâmico

Algumas outras pesquisas interessantes

- *Audio Games*
- Música que controla fenômenos no jogo
- Música composta pelas ações do jogo
- Aplicação de técnicas de *DJing*
- Síntese procedimental de efeitos sonoros físicos
- Propagação do som em ambientes dinâmicos

Como podemos contemplar essas ténicas em nossa proposta?

Trabalhos acadêmicos: aplicações de áudio dinâmico

Algumas outras pesquisas interessantes

- *Audio Games*
- Música que controla fenômenos no jogo
- Música composta pelas ações do jogo
- Aplicação de técnicas de *DJing*
- Síntese procedimental de efeitos sonoros físicos
- Propagação do som em ambientes dinâmicos

Como podemos contemplar essas ténicas em nossa proposta?

Trabalhos acadêmicos: aplicações de áudio dinâmico

Algumas outras pesquisas interessantes

- *Audio Games*
- Música que controla fenômenos no jogo
- Música composta pelas ações do jogo
- Aplicação de técnicas de *DJing*
- Síntese procedimental de efeitos sonoros físicos
- Propagação do som em ambientes dinâmicos

Como podemos contemplar essas ténicas em nossa proposta?

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

“como um jeito do sound designer não precisar do programador”

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

“como um jeito do sound designer não precisar do programador”

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

"como um jeito do sound designer não precisar do programador"

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

"como um jeito do sound designer não precisar do programador"

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

"como um jeito do sound designer não precisar do programador"

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

"como um jeito do sound designer não precisar do programador"

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

“como um jeito do sound designer não precisar do programador”

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais

Arcabouços e motores

- Conseguem reproduzir arquivos de áudio estático
- Compatíveis com formatos populares
- Podem ter um efeito especial ou outro
- Para mais funcionalidades, precisa programar

Solução: usar um middleware para áudio dinâmico

“como um jeito do sound designer não precisar do programador”

Opções:

- Desenvolvido pela própria equipe
- Implementação já existente
- Ad hoc

Trabalhos comerciais: soluções já existentes

Aspectos gerais

- Plataformas compatíveis
- Plataforma-alvo $\times$ palaforma de desenvolvimento
- Interface autoral e API

Wwise
Audiokinect

FMOD Studio
Firelight Technologies

Trabalhos comerciais: soluções já existentes

Aspectos gerais

- Plataformas compatíveis
- Plataforma-alvo $\times$ palaforma de desenvolvimento
- Interface autoral e API

Wwise
Audiokinect

FMOD Studio
Firelight Technologies

Trabalhos comerciais: soluções já existentes

Aspectos gerais

- Plataformas compatíveis
- Plataforma-alvo $\times$ palaforma de desenvolvimento
- Interface autoral e API

Wwise
Audiokinect

FMOD Studio
Firelight Technologies

Trabalhos comerciais: soluções já existentes

Aspectos gerais

- Plataformas compatíveis
- Plataforma-alvo $\times$ palaforma de desenvolvimento
- Interface autoral e API

Wwise
Audiokinect

FMOD Studio
Firelight Technologies

Trabalhos comerciais: soluções já existentes

Aspectos gerais

- Plataformas compatíveis
- Plataforma-alvo $\times$ palaforma de desenvolvimento
- Interface autoral e API

Wwise
Audiokinect

FMOD Studio
Firelight Technologies

Trabalhos comerciais: soluções já existentes

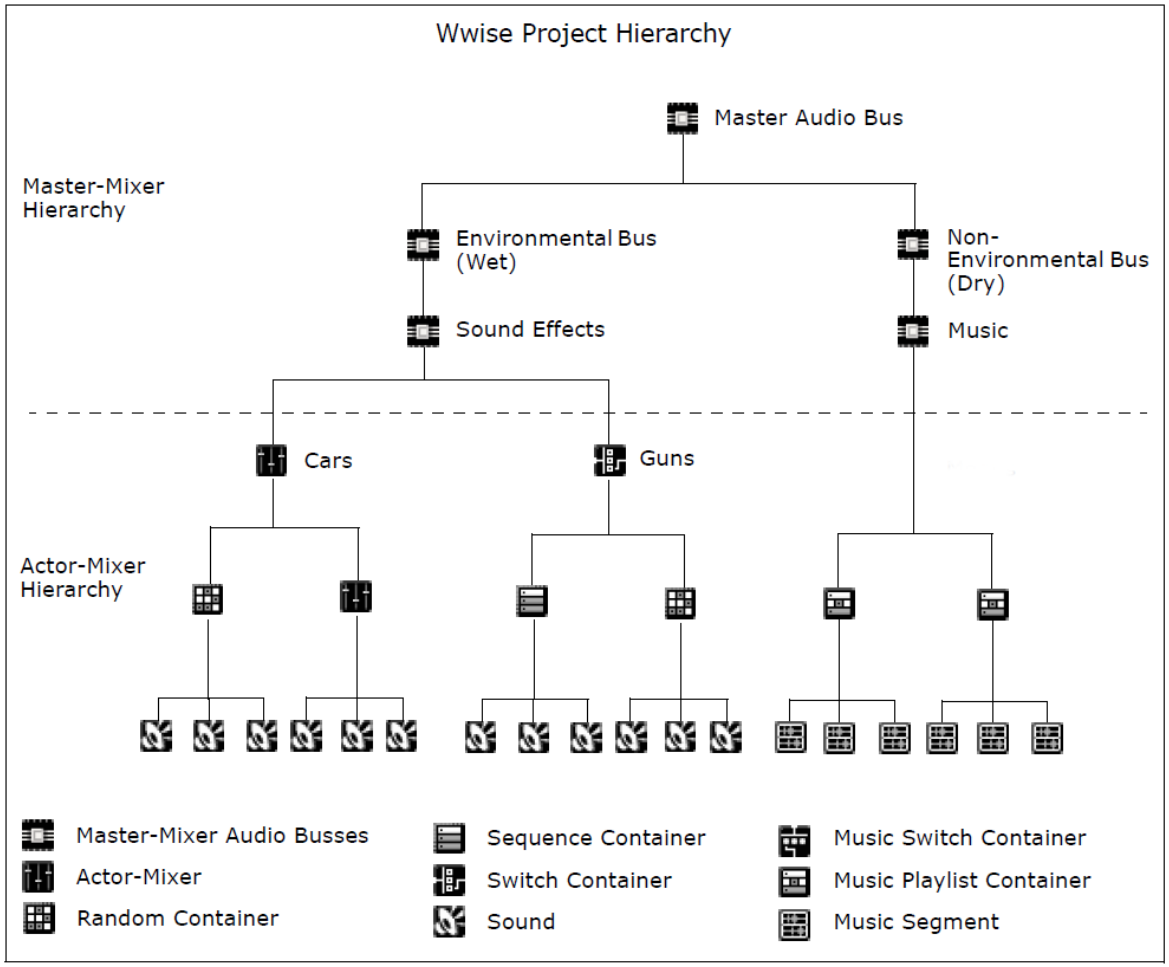
Aspectos gerais

- Plataformas compatíveis
- Plataforma-alvo $\times$ palaforma de desenvolvimento
- Interface autoral e API

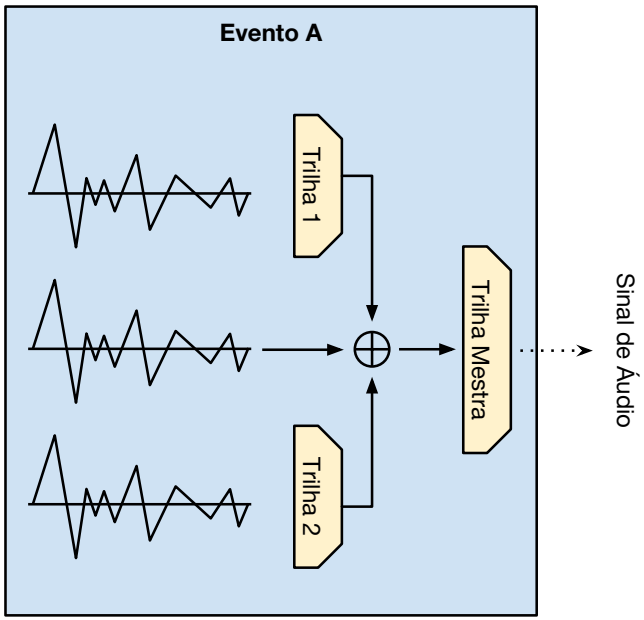
Wwise
Audiokinect

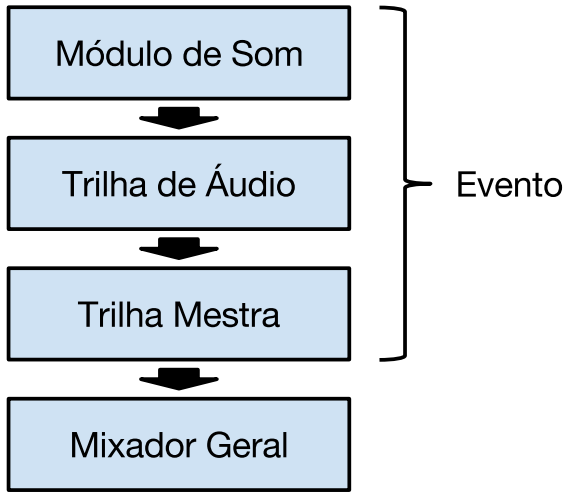
FMOD Studio
Firelight Technologies

Trabalhos comerciais: Wwise



Trabalhos comerciais: FMOD Studio





Trabalhos comerciais: soluções ad hoc

Motivos

- Não existia esse tipo de middleware na época
- Plataforma incompatível
- Preço

Jogos analisados

1. Super Mario Bros.
2. The Legend of Zelda: Twilight Princess
3. Faster Than Light

Trabalhos comerciais: soluções ad hoc

Motivos

- Não existia esse tipo de middleware na época
- Plataforma incompatível
- Preço

Jogos analisados

1. Super Mario Bros.
2. The Legend of Zelda: Twilight Princess
3. Faster Than Light

Trabalhos comerciais: soluções ad hoc

Motivos

- Não existia esse tipo de middleware na época
- Plataforma incompatível
- Preço

Jogos analisados

1. Super Mario Bros.
2. The Legend of Zelda: Twilight Princess
3. Faster Than Light

Trabalhos comerciais: soluções ad hoc

Motivos

- Não existia esse tipo de middleware na época
- Plataforma incompatível
- Preço

Jogos analisados

1. Super Mario Bros.
2. The Legend of Zelda: Twilight Princess
3. Faster Than Light

Trabalhos comerciais: soluções ad hoc

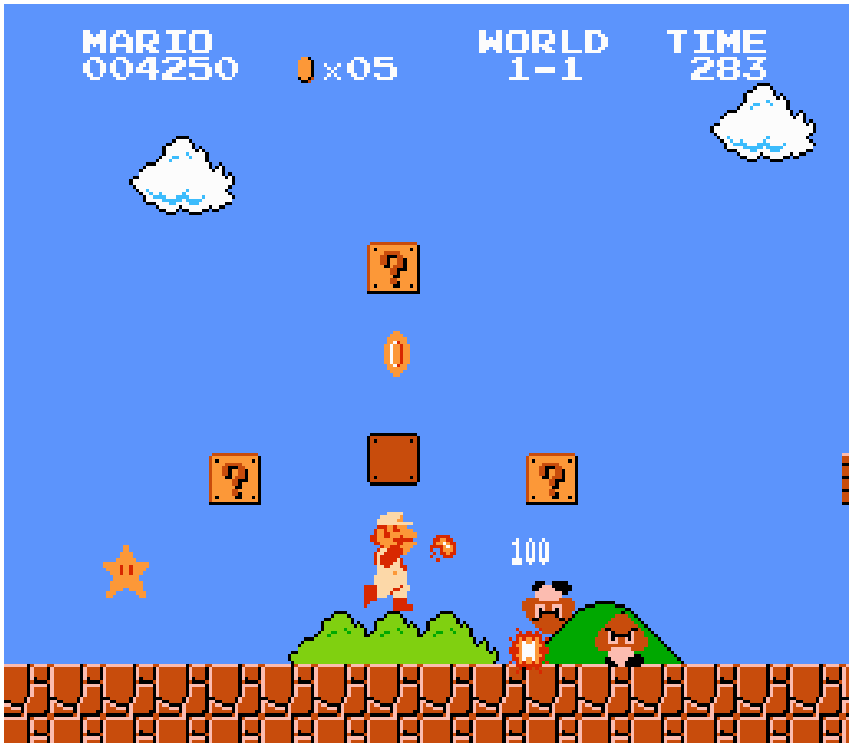
Motivos

- Não existia esse tipo de middleware na época
- Plataforma incompatível
- Preço

Jogos analisados

1. Super Mario Bros.
2. The Legend of Zelda: Twilight Princess
3. Faster Than Light

Trabalhos comerciais: Super Mario Bros. (1985)



- Música de tempo acabando
- Música de invencibilidade

Trabalhos comerciais: Zelda - Twilight Princess (2006)

- Música de inimigos por perto
- Transições em lutas com chefes

Trabalhos comerciais: Faster Than Light (2012)

- Toda música tem versão “exploração” e “combate”
- Sons de incêndio, vazamentos e confrontos

Proposta

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Open Dynamic Audio

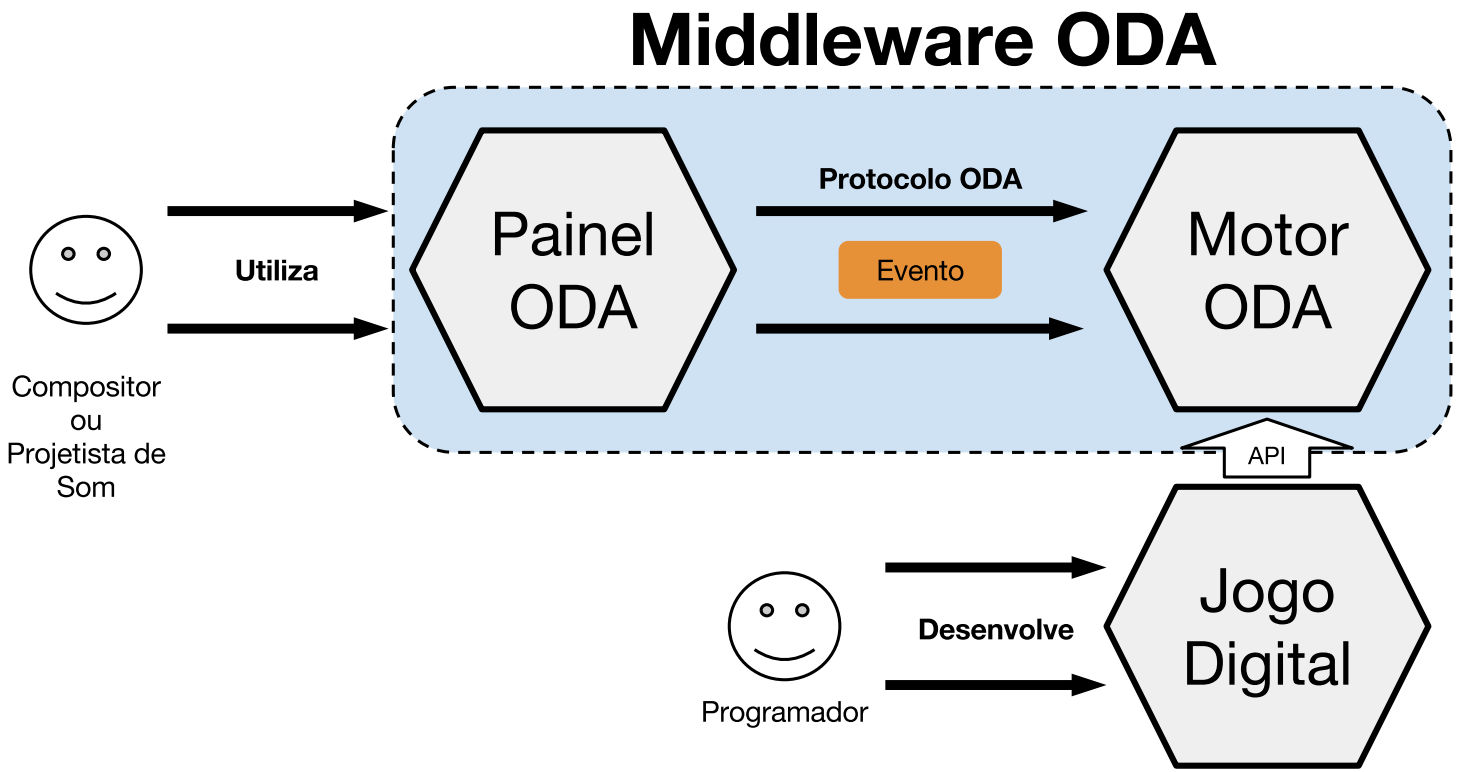
Um middleware de áudio dinâmico precisa de:

- Interface autoral
- Motor de áudio dinâmico
- Protocolo entre eles

Diferenciais

- Desenvolvimento em Linux
- Software Livre
- Extensível

Arquitetura geral do sistema



Implementação

Ferramentas usadas:

- Linguagem: C++11
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação

Ferramentas usadas:

- Linguagem: `C++11`
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação

Ferramentas usadas:

- Linguagem: `C++11`
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação

Ferramentas usadas:

- Linguagem: `C++11`
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação

Ferramentas usadas:

- Linguagem: `C++11`
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação

Ferramentas usadas:

- Linguagem: `C++11`
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação

Ferramentas usadas:

- Linguagem: `C++11`
- Reprodução de som: OpenAL
- DSP: *PureData*
- Incorporação de *patches*: libpd

Outras informações:

- Licença MIT (por enquanto)
- Repositório: https://github.com/open-dynamic-audio/liboda
- Exemplos: https://github.com/open-dynamic-audio/examples
- Coorientação com aluno de graduação (TCC)

Implementação: API

Classes principais:

- Engine controla o motor e gerencia eventos
- Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

- Engine controla o motor e gerencia eventos
- Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

- Engine controla o motor e gerencia eventos
- Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

Engine controla o motor e gerencia eventos

Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

Engine controla o motor e gerencia eventos

Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

Engine controla o motor e gerencia eventos

Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

Engine controla o motor e gerencia eventos

Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

Engine controla o motor e gerencia eventos

Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: API

Classes principais:

Engine controla o motor e gerencia eventos

Event reproduz e manipula eventos

Caso de uso típico:

1. Obter instância do motor
2. Usar esse objeto para instanciar eventos
3. Usar objetos de evento para reproduzir áudio
4. Manipular parâmetros dos eventos conforme convier

Implementação: patches de *PureData*

O ideal é que eventos sejam patches

- Motivação: interface autoral "pronta"
- Dificuldades:
  - Ambiente global
  - Usabilidade questionável
  - Sequenciamento cronológico de áudio
- Vantagens:
  - Software Livre
  - Multiplataforma
  - Comunidade ativa
  - Compatível com MIDI e OSC
  - Permite expansões pelos usuários!!!

Implementação: patches de *PureData*

O ideal é que eventos sejam patches

- Motivação: interface autoral "pronta"
- Dificuldades:
  - Ambiente global
  - Usabilidade questionável
  - Sequenciamento cronológico de áudio
- Vantagens:
  - Software Livre
  - Multiplataforma
  - Comunidade ativa
  - Compatível com MIDI e OSC
  - Permite expansões pelos usuários!!!

Implementação: patches de *PureData*

O ideal é que eventos sejam patches

- Motivação: interface autoral "pronta"
- Dificuldades:
  - Ambiente global
  - Usabilidade questionável
  - Sequenciamento cronológico de áudio
- Vantagens:
  - Software Livre
  - Multiplataforma
  - Comunidade ativa
  - Compatível com MIDI e OSC
  - Permite expansões pelos usuários!!!

Implementação: patches de *PureData*

O ideal é que eventos sejam patches

- Motivação: interface autoral "pronta"
- Dificuldades:
  - Ambiente global
  - Usabilidade questionável
  - Sequenciamento cronológico de áudio
- Vantagens:
  - Software Livre
  - Multiplataforma
  - Comunidade ativa
  - Compatível com MIDI e OSC
  - Permite expansões pelos usuários!!!

Implementação: patches de *PureData*

O ideal é que eventos sejam patches

- Motivação: interface autoral "pronta"
- Dificuldades:
  - Ambiente global
  - Usabilidade questionável
  - Sequenciamento cronológico de áudio
- Vantagens:
  - Software Livre
  - Multiplataforma
  - Comunidade ativa
  - Compatível com MIDI e OSC
  - Permite expansões pelos usuários!!!

Validação

Do motor

- Usuário: programador
- Usar dois jogos abertos:
  - Mari0 (ação *platformer*)
  - Battle for Wesnoth (estratégia por turnos)

Da interface autoral

- Usuário: compositor ou projetista de som
- Pelo menos dois
- Validação por entrevistas

Validação

Do motor

- Usuário: programador
- Usar dois jogos abertos:
  - Mari0 (ação *platformer*)
  - Battle for Wesnoth (estratégia por turnos)

Da interface autoral

- Usuário: compositor ou projetista de som
- Pelo menos dois
- Validação por entrevistas

Validação

Do motor

- Usuário: programador
- Usar dois jogos abertos:
  - Mari0 (ação *platformer*)
  - Battle for Wesnoth (estratégia por turnos)

Da interface autoral

- Usuário: compositor ou projetista de som
- Pelo menos dois
- Validação por entrevistas

Plano de trabalho

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Resultados parciais

Protótipo:

- Usa o jogo aberto Mari0
- Áudio com *PureData*
- Conexão via UDP com protocolo OSC
- Percussão intensificada pelos inimigos
- Modulação quando coleta uma *Fire Flower*

Motor ODA:

- Som através da OpenAL
- Classe `Engine`
- Início da integração com *PureData*
- TCC

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Expectativas

Negativas:

- Qualidade final
- Incompatível com jogos Web

Positivas:

- Diferenciais da proposta
- Compatível com *mobile*
- Aplicações mais gerais

Atividades futuras

Implementação

1. Investigar possibilidades da libpd
2. Projetar Painel ODA e Protocolo ODA
3. Implementar

Validação

1. Mari0
2. Battle for Wesnoth
3. Parceria com músicos

Dissertação

1. Artigo
2. Escrever a cada resultado

Atividades futuras

Implementação

1. Investigar possibilidades da libpd
2. Projetar Painel ODA e Protocolo ODA
3. Implementar

Validação

1. Mari0
2. Battle for Wesnoth
3. Parceria com músicos

Dissertação

1. Artigo
2. Escrever a cada resultado

Atividades futuras

Implementação

1. Investigar possibilidades da libpd
2. Projetar Painel ODA e Protocolo ODA
3. Implementar

Validação

1. Mari0
2. Battle for Wesnoth
3. Parceria com músicos

Dissertação

1. Artigo
2. Escrever a cada resultado

Atividades futuras

Implementação

1. Investigar possibilidades da libpd
2. Projetar Painel ODA e Protocolo ODA
3. Implementar

Validação

1. Mari0
2. Battle for Wesnoth
3. Parceria com músicos

Dissertação

1. Artigo
2. Escrever a cada resultado

Cronograma

Atividades	8/15	9/15	10/15	11/15	12/15	1/16	2/16	3/16	4/16	5/16
Motor básico	×	×								
Experimentos		×	×	×						
Interface autoral			×	×	×					
Validação 1				×	×					
Artigo					×	×	×			
Validação 2						×	×	×	×	
Dissertação		×		×	×		×		×	×

Obrigado!