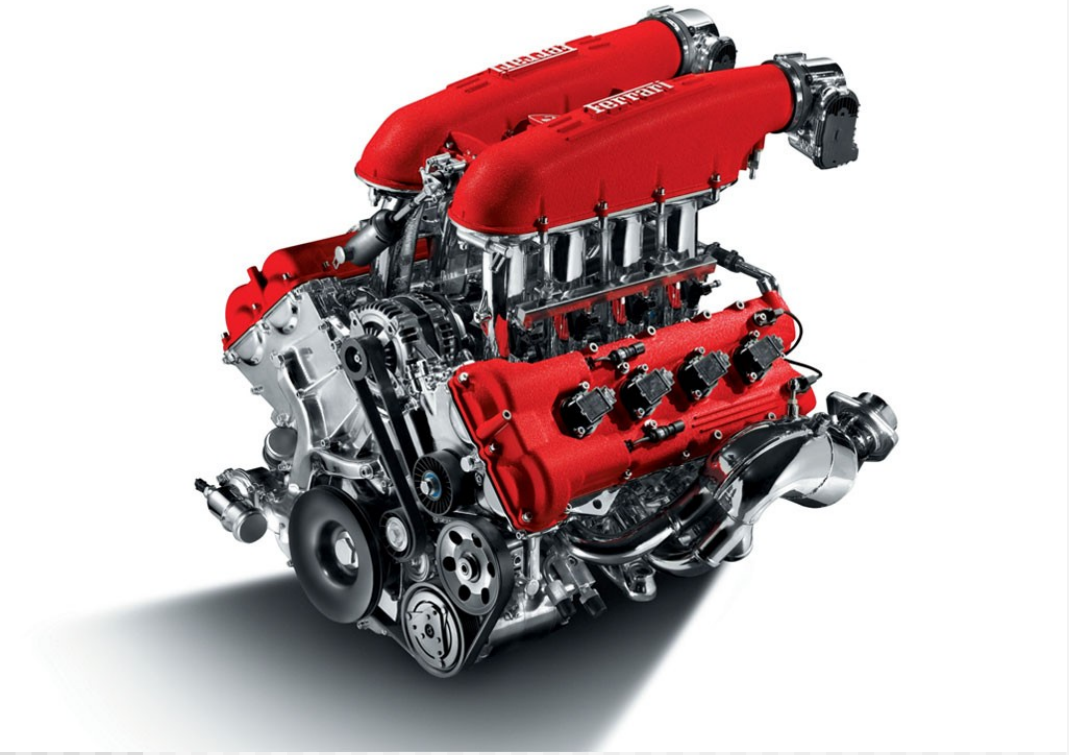
Introdução aos Motores de Jogos

Prof. MSc. João Ricardo Bittencourt

joaorb@unisinos.br
www.inf.unisinos.br/~jrbitt
www.ludensartis.com.br

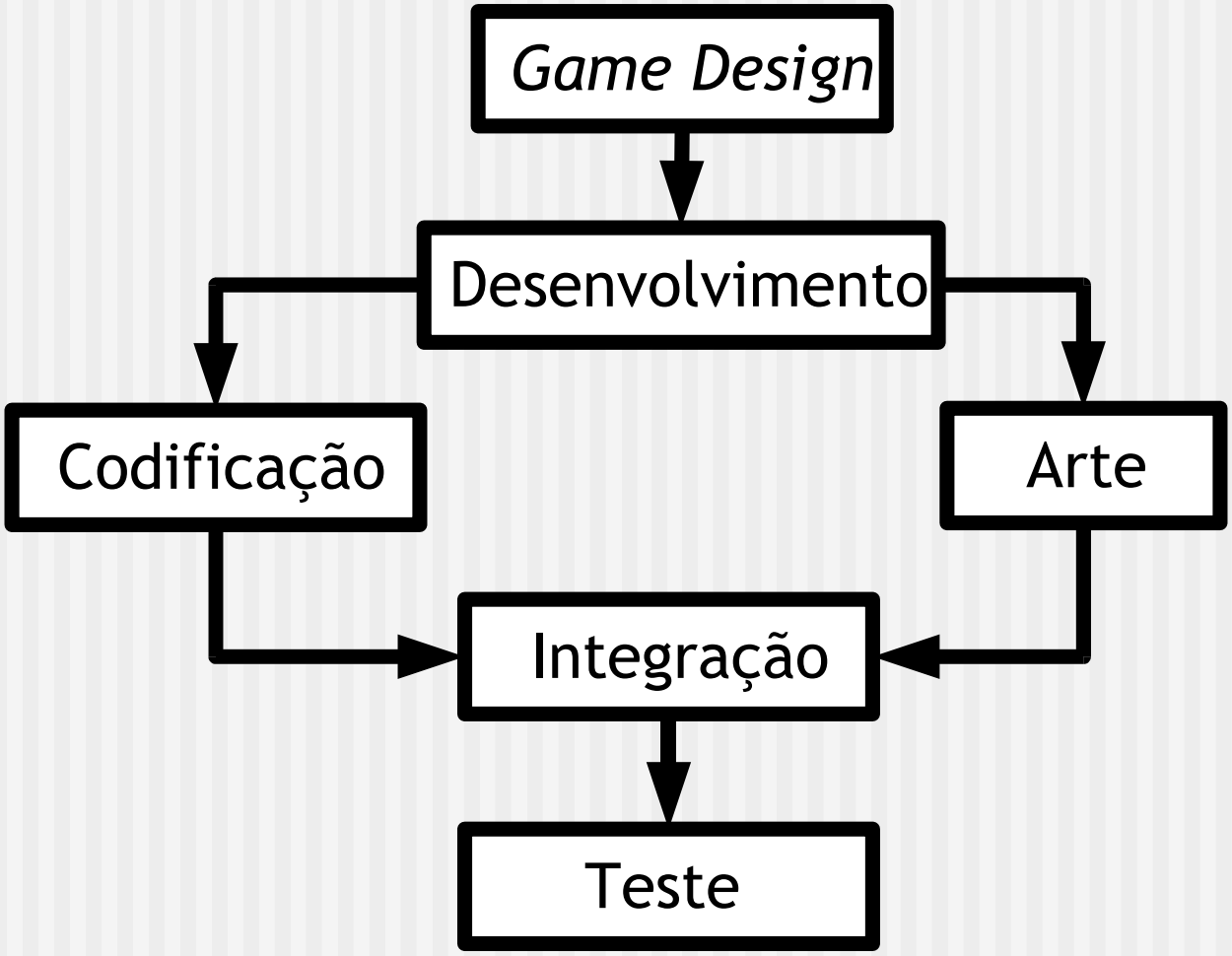


UNISINOS

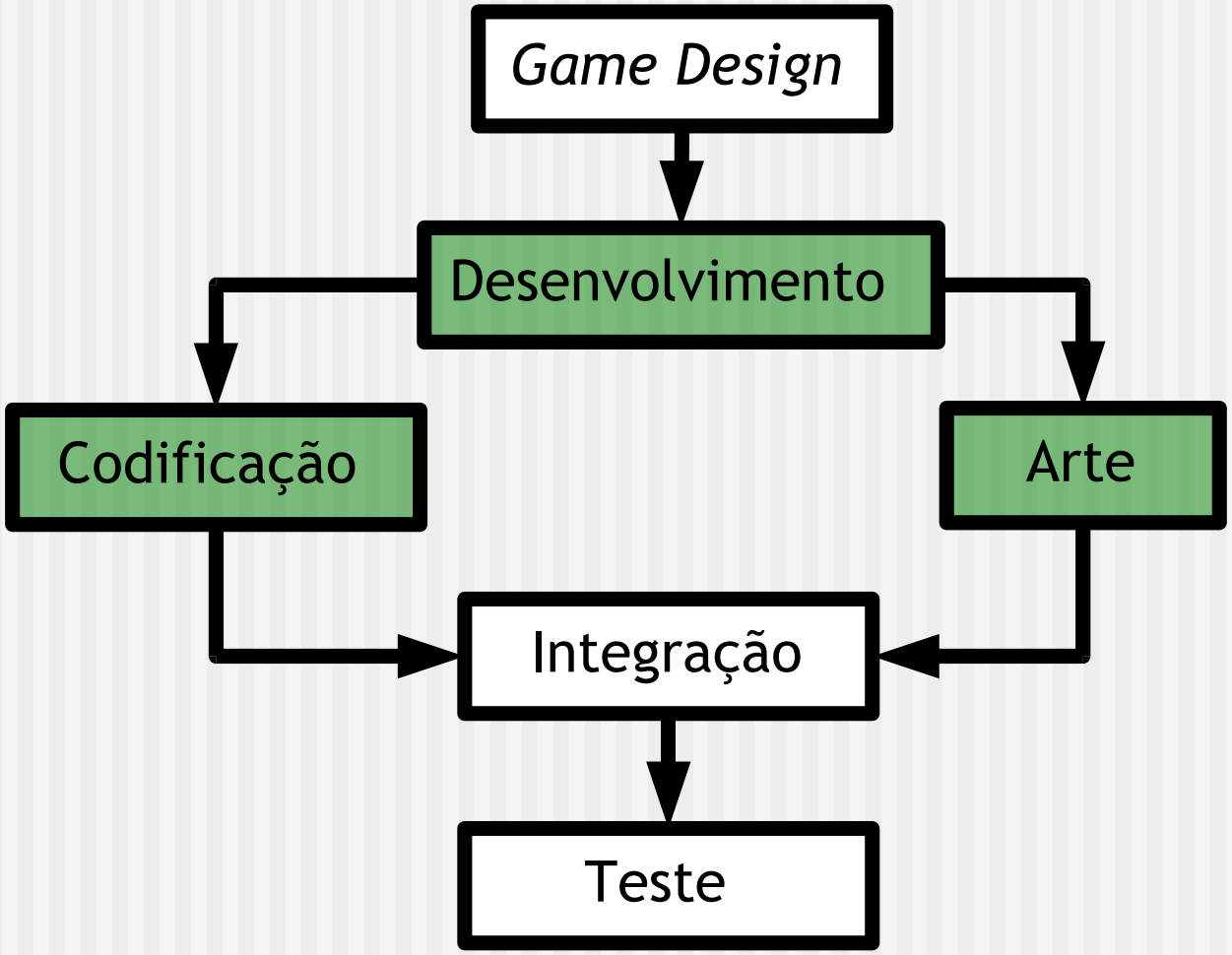
Sumário

1. Contexto
2. Principais conceitos de motores
3. Arquiteturas de motores
4. Ogre3D
5. Referências

Contexto


Game Design
Desenvolvimento
Codificação
Arte
Integração
Teste

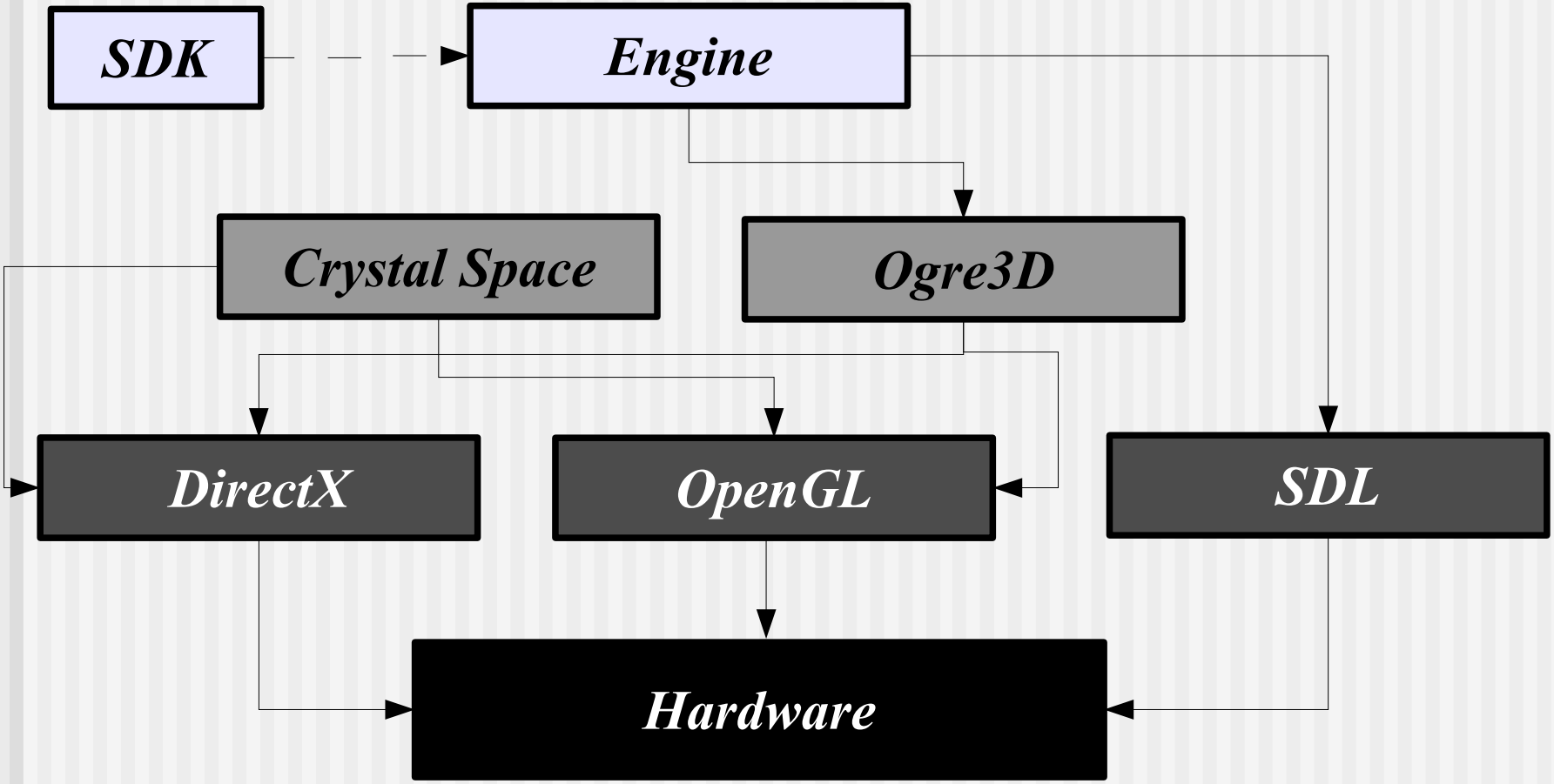
Contexto


Game Design
Desenvolvimento
Codificação
Arte
Integração
Teste

Contexto

- Existem muitas alternativas para o desenvolvimento de jogos digitais
  - Código de máquina – *Assembly*
  - Bibliotecas – OpenGL/DirectX/SDL
  - Toolkits – Crystal Space
  - SDKs – Dark Game SDK
- Mas queremos projetar/desenvolver **motores de jogos**!

Contexto


SDK
Engine
Crystal Space
Ogre3D
DirectX
OpenGL
SDL
Hardware

Contexto

- **Biblioteca** – conjunto de **funções** para desempenhar alguma tarefa
  - SDL, OpenGL, DirectX
- **Toolkits** – coleção de **classes** que oferecem um conjunto de serviços
  - CrystalSpace, Ogre3D
- **SDK** – conjunto de softwares e demais artefatos usados para programar outro software

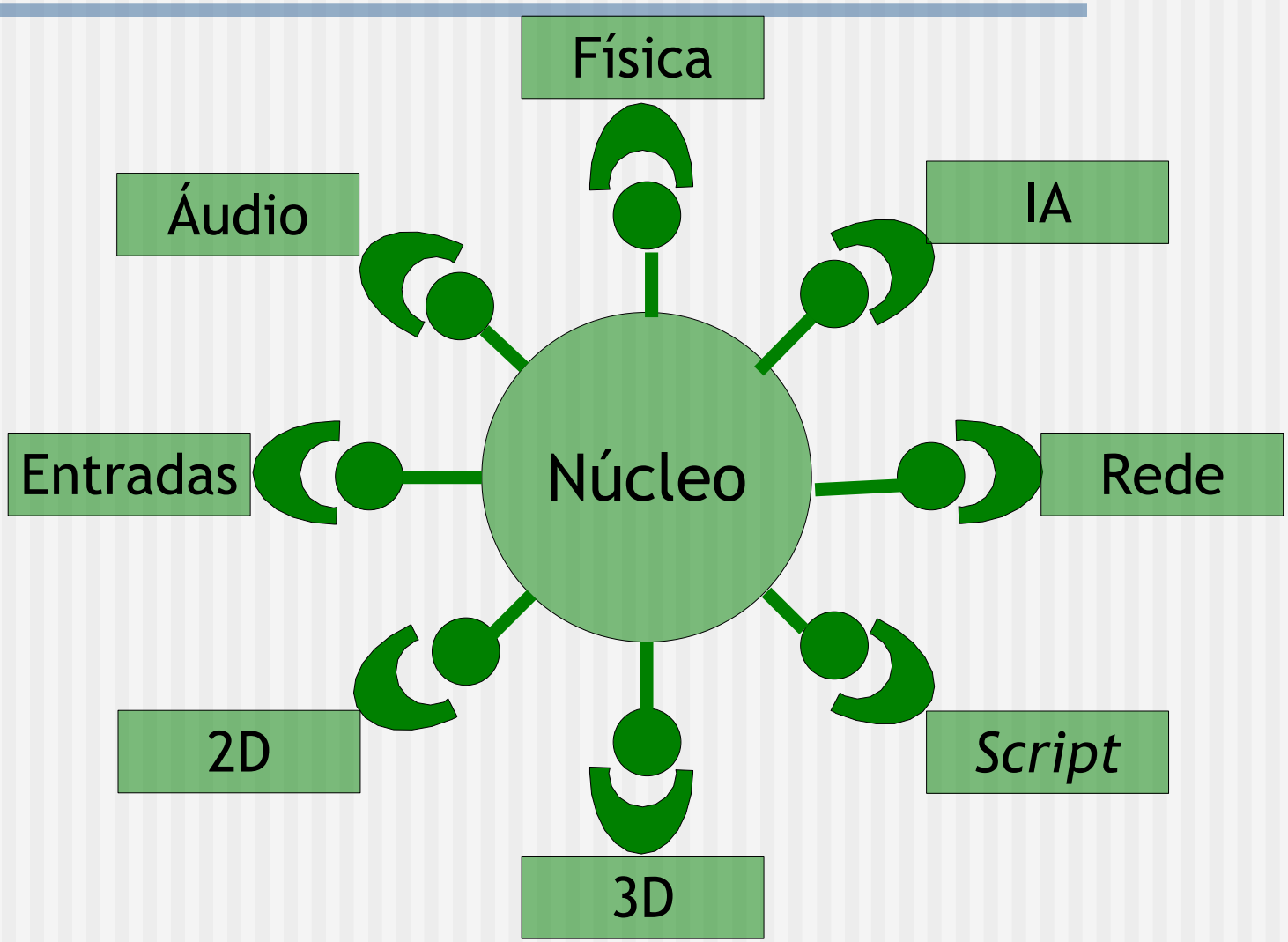
Conceitos

- **Relação Generalização x Desempenho**
  - Criação de um *game design*
    - Eixo norteador
  - Contextualização de mercado
    - Serão produzidos outros títulos?
  - Definição da plataforma de execução do jogo (recursos)
  - Qual o nível de desempenho requerido pelo jogo? O que é crítico?
    - Nem todo o jogo tem desempenho como requisito crítico
  - Qual o grau de generalização posso adotar?

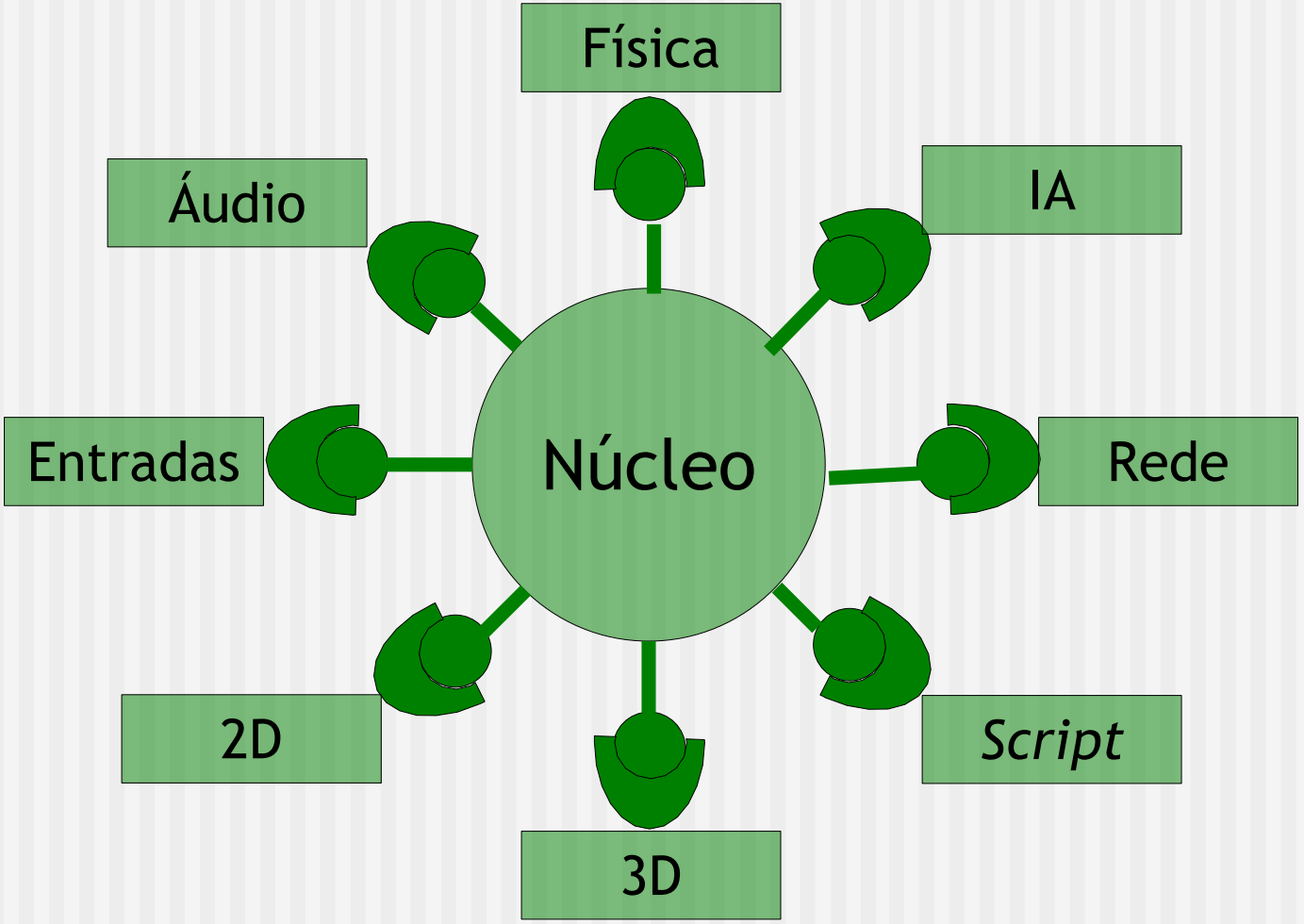
Conceitos

- Para Lewis & Jacobson, motores de jogos tratam-se de uma **coleção de módulos** de **simulação** que **não especifica diretamente** o comportamento ou ambiente do jogo
- Inclui módulos para:
  - Capturar eventos de entrada
  - Gerar saída gráfica e de áudio
  - Gerenciar a dinâmica do mundo de jogo

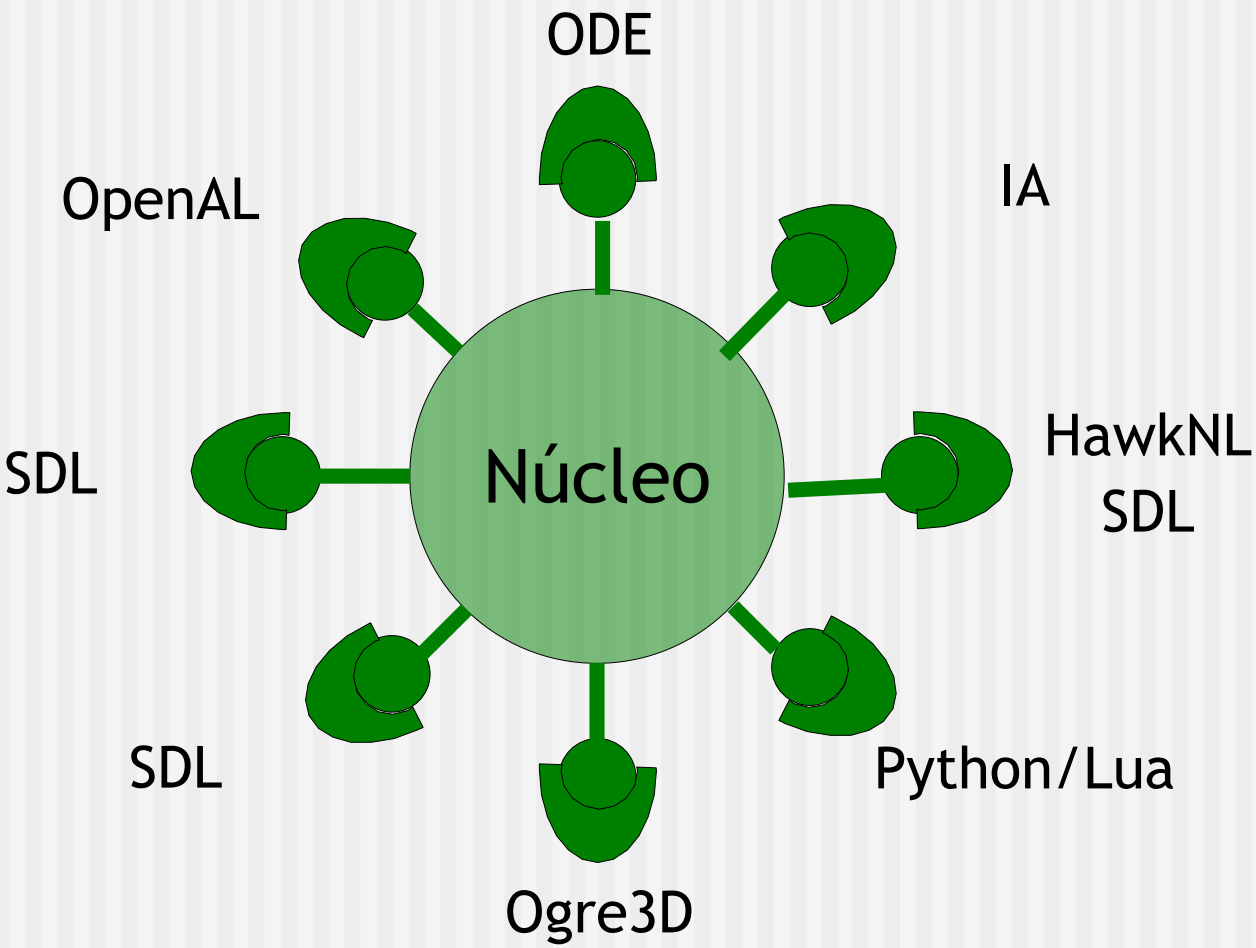
Conceitos



Conceitos

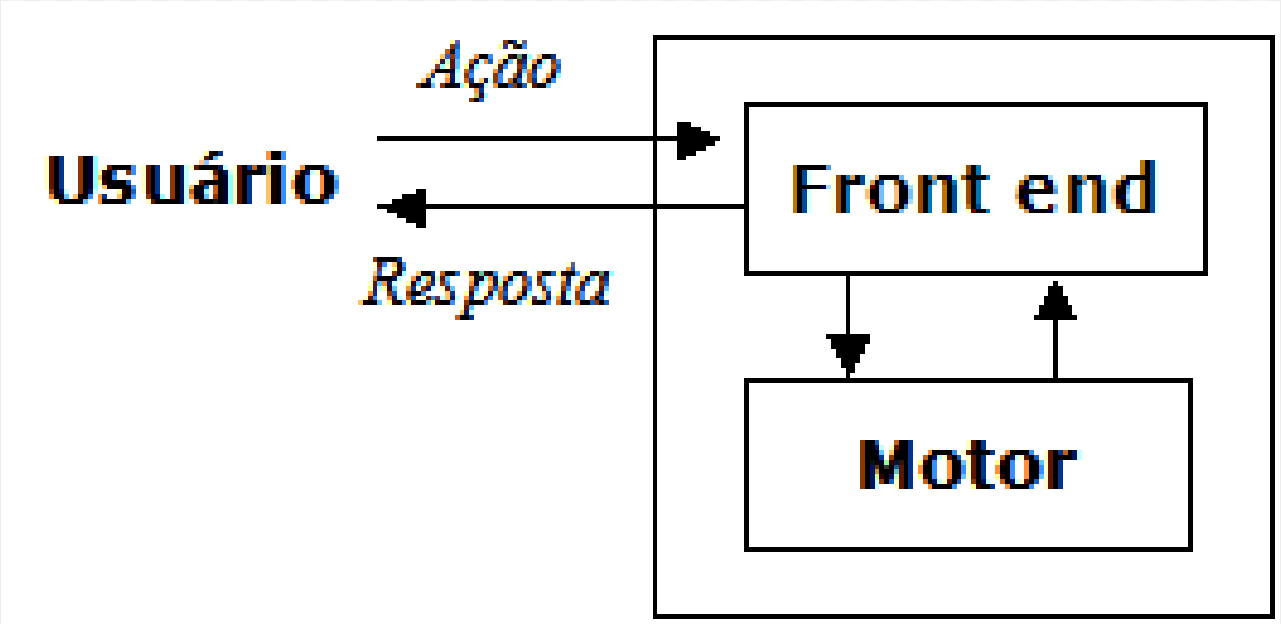


Conceitos



Conceitos

- **Esquema genérico de um motor de jogo**



Conceitos

- Para Watt & Policarpo, os principais fatores envolvidos na criação de um motor de jogo
  - Qual o nível de facilidade que será oferecido pelo motor? Oferecer facilidades conforme o gênero.
  - Oferecer diferentes otimizações típicas para cada gênero
  - O que pode ser delegado para o hardware?
  - Evitar despender grande parte do projeto desenvolvendo ferramentas com baixa reusabilidade
  - Complexidade de um jogo digital está embarcada no motor de jogo

Exemplos

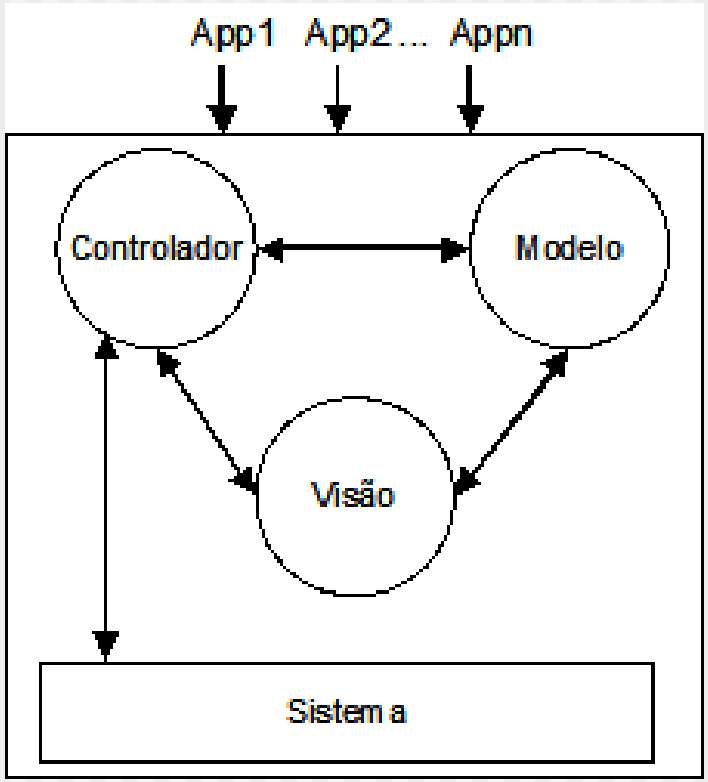
- SCUMM
- *RenderWare*
- *Quake, Quake II e Quake III*
- *Half Life*
- *Unreal*
- *Torque*
- *Yake* (baseada no Ogre3D)
- Entre outras centenas ...
  - http://cg.cs.tu-berlin.de/~ki/engines.html

Arquiteturas

- **Hodorowicz**
  - Dados – imagens, sons, modelos 3D, scripts,...
  - Sistema – comunicação com o hardware
    - Gráficos, entradas, som, *timer* e configuração do motor
  - Console – usada no modo *debug*
  - Suporte – matemática, gerenciador de memória, carregar arquivos, estruturas de dados
  - Renderer – produção das imagens sintéticas
  - Interface do jogo – abstração do gênero
  - Jogo – lógica e os dados usados pelo motor

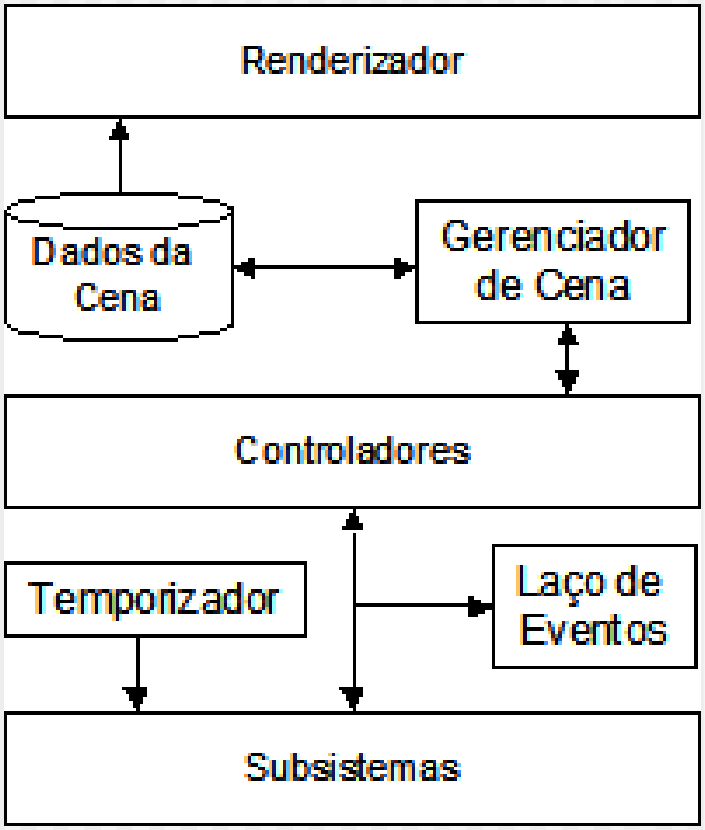
Arquiteturas

- **Forge V8 3D (Domingues)**
  - Baseada no padrão MVC (*Modelo-Visão-Controle*)



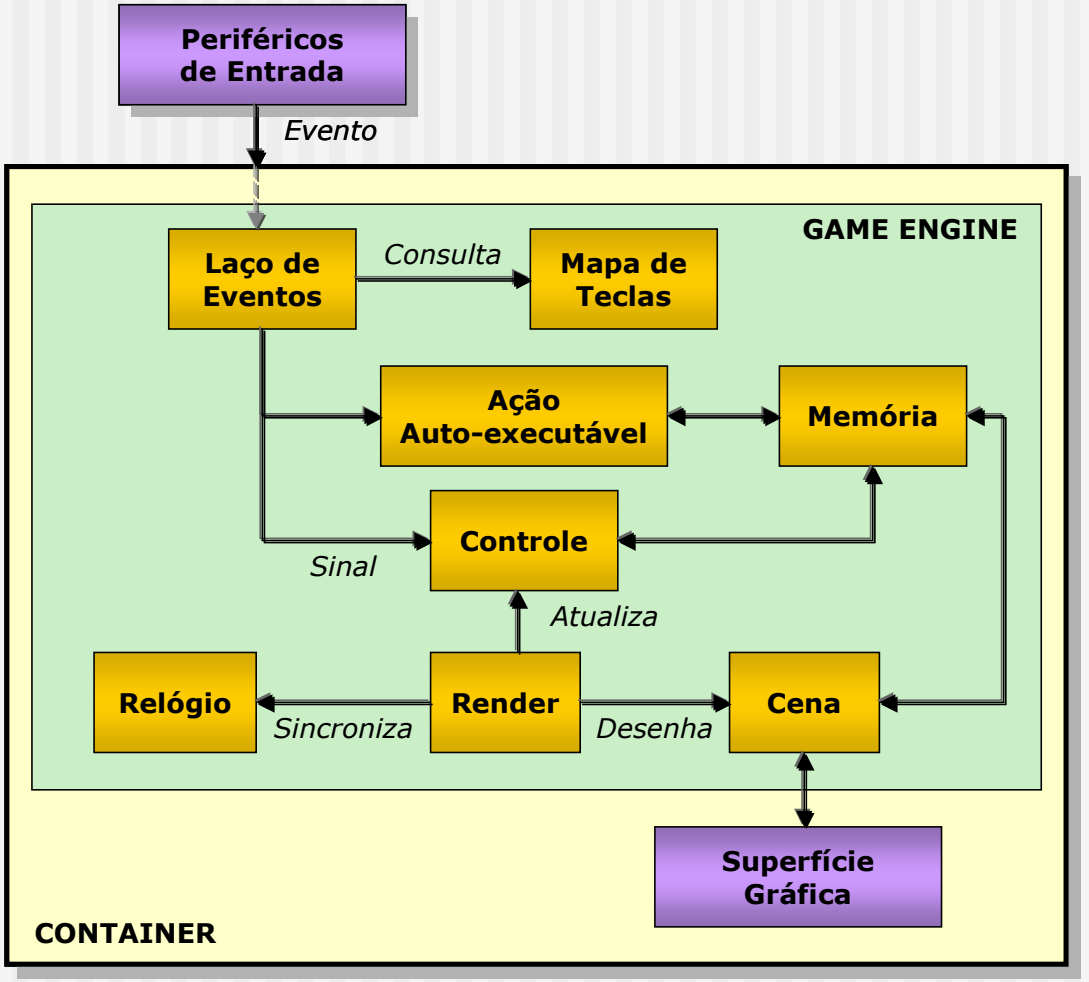
Arquiteturas

- **Gelatti**
  - Proposta de generalização da arquitetura do motor



Arquiteturas

- **Bittencourt**
  - Usada no JFRoGE - multiplataforma



Ogre3D

- O OGRE (*Object-oriented Graphics Rendering*) foi criado no final de 2001 pelo inglês Steve Streeting, conhecido como Sinbad. É o proprietário da ***Torus Knot Software Ltd*** que presta serviços e desenvolve produtos baseado no OGRE.
- Desenvolvido em C++ executa em diferentes plataformas (Win/Linux/MacOS)
- **Importante!** Trata-se de um *engine* gráfico **NÃO** é um *game engine*.
- Possui uma série de *plug-ins*, ferramentas e *add-ons* para criar diferentes aplicações gráficas.

Ogre3D

- Suporta diferentes configurações de *hardware* e é compatível com *DirectX* e *OpenGL*. Inclusive permite **detectar o hardware** e utilizar ***shaders***.
- Possui *bindings* para Java, .Net e Python.
- **Vantagem**: não precisar reinventar a roda, reiniciar o processo do zero.
- E o melhor de tudo – é um projeto *Software Livre! (GNU LGPL)*
- **Demonstrações**

Referências

- BATES, Bob. **Game Design – The Art & Business of Creating Games**.Roseville: Prima Tech, 2001, 300 p.
- BATTAIOLA, André L. Jogos por Computador – Histórico, Relevância Tecnológica e Mercadológica, Tendências e Técnicas de Implementação In: **XIX Jornada de Atualização em Informática**. Proceedings. Curitiba:SBC, Julho/2000, v. 2. pp. 83-122.
- DOMINGUES, Rodrigo G. **Projeto de um framework para auxilio no desenvolvimento de aplicações com gráficos 3D e animação.** São Carlos, UFSCAR, 2003, 196 p. (Dissertação de Mestrado)

Referências

- FURTADO, André; SANTOS, André. FunGEn – Um Motor para Jogos em Haskell. **In: I Workshop Brasileiro de Jogos e Entretenimento Digital.** Proceedings. SBC: Fortaleza, 2002, [Meio digital].
- GELATTI, Giorgenes P. A Framework for Building Engines for Games and Simulations **In: I Workshop Brasileiro de Jogos e Entretenimento Digital.** Proceedings. SBC: Fortaleza, 2002. [Meio digital]
- HODOROWICZ, Luke. **Elements of a Game Engine.** 2001. Disponível em: <http://www.flipcode.com/tutorials/tut_el_engine.shtml> Acesso em 16 ago. 2005. (Whitepaper)

Referências

- MADEIRA, André G. **Forge V8: Um Framework para o Desenvolvimento de Jogos de Computador e Aplicações Multimídia.** Recife, UFPE, 2001, 99p. (Dissertação de Mestrado)
- PESSOA, Carlos A.C. **wGEM: Um Motor para o Desenvolvimento de Jogos para Dispositivos Móveis**. Recife: CIN/UFPE, 2001, 58p. (Trabalho de Conclusão)
- LEWIS, Michael; JACOBSON, Jeffrey. Game Engines in Scientific Research **In: Communication of the ACM.** Jan. 2002, vol. 45, n. 1, 27-31p.

Referências

- ROLLINGS, Andrew. MORRIS, Dave. **Game Architecture and Design**. Arizona: Coriolis, 2000, 742 p.
- WATT, Alan; POLICARPO, Fabio. **3D Games: Real-Time Rendering and Software Tecnology**. Addison-Wesley, 2000, 800 p.