Universidade Regional de Blumenau
Centro de Ciências Exatas e Naturais
Departamento de Sistemas e Computação

FURB

Protótipo de um ambiente virtual distribuído multiusuário

Acadêmico: Vandeir Eduardo
Prof. orientador: Dalton Solano dos Reis

Apresentação para defesa em banca do Trabalho de Conclusão de Curso da primeira fase do ano de 2001, para obtenção do título de Bacharel em Ciências da Computação

Roteiro da apresentação

- **Introdução**
  - Contextualização / objetivos
- **Fundamentação teórica**
  - Ambientes virtuais distribuídos (características, técnicas e protocolos associados)
  - Java3D (características e principais classes da API)
  - DIS-Java-VRML (características e principais classes da API
- **Desenvolvimento do trabalho**
  - Ferramentas utilizadas
  - Grafo de cena e fluxogramas dos principais processos
- **Considerações finais**
  - Conclusões e extensões

Introdução - Contextualização

- Imagens em 3D como uma forma de interface mais natural
- Surgimento dos ambientes virtuais (imersivos, não imersivos, mono ou multiusuário)
- Implementação de AVD's exigem especial atenção de aspectos de rede (utilização de largura de banda, latência, confiabilidade)
- Necessidade de apoio de outras ferramentas (API's do Java3D e DIS-Java-VRML)

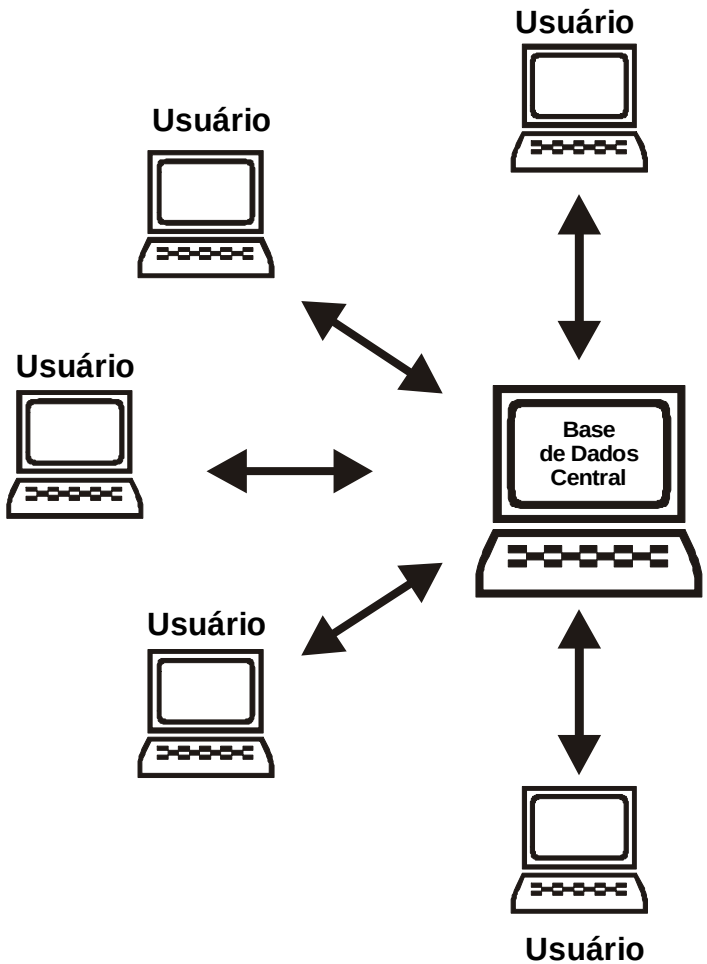
Introdução - Objetivos

- **Implementar um protótipo de ambiente virtual distribuído sobre uma rede local, com suporte a multiusuários e com uma interface não imersiva**
- **Conseqüentemente:**
  - Pesquisar principais características de AVD's
  - Avaliar funcionamento das API's do Java3D e DIS-Java-VRML

Fundamentação – AVD's

- **Primeiras experiências estão relacionadas a aplicações para simulações militares**
- **Projetos que servem como referência: SIMNET e NPSNET**
- **Resultado das pesquisas desses projetos servem como referência nos seguintes aspectos de construção de um AVD:**
  - modelo geral de comunicação
  - utilização da largura de banda, confiabilidade e latência
  - técnicas (*heartbeats*, e protolocos (*broadcast*, DIS) mais indicados

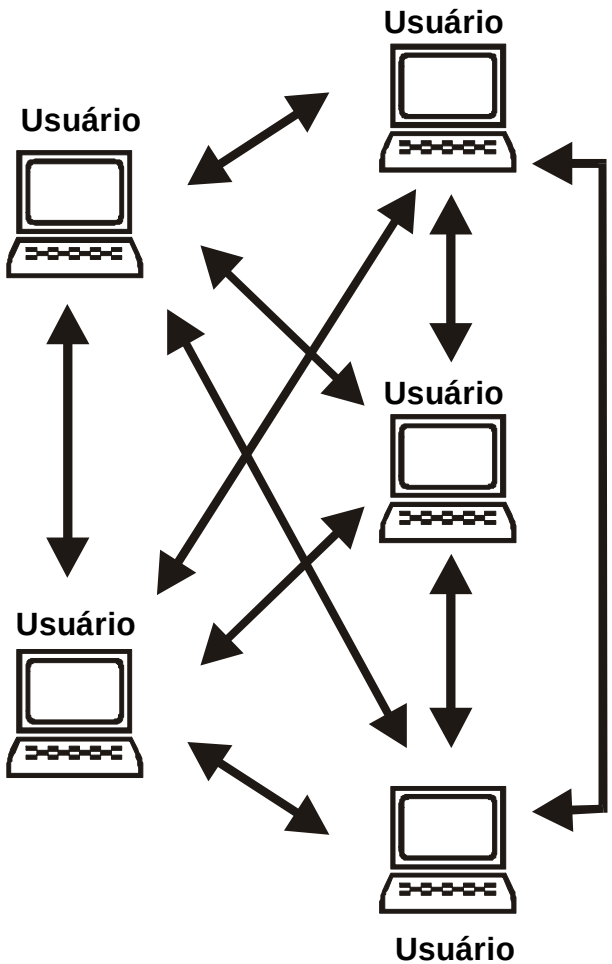
Modelos de Comunicação



Modelo Centralizado:

- Processamento e representação do ambiente virtual centralizados
- Simplifica o mecanismo de controle da comunicação
- Mais suscetível a falhas
- Problema de escalabilidade
- Sobrecarga de mensagens para o computador central

Modelos de Comunicação



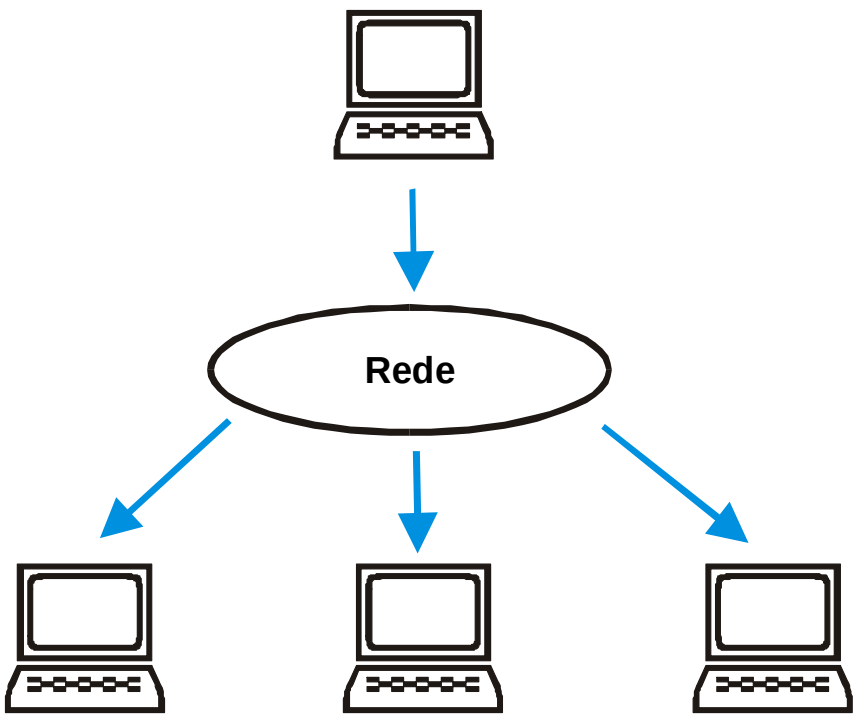
Modelo Distribuído:

- Cada usuário mantém a sua própria representação do AV
- Cada usuário usa seus próprios recursos computacionais
- Qualquer alteração no AV precisa ser comunicada a todos os usuários
- Mais tolerante a falhas, evita congestionamento de mensagens sobre um único computador
- Problema de geração excessiva de mensagens

Aspectos de rede

- **Largura de banca (*Bandwidth*)**
  - Maior detalhamento + nº usuários = maior utilização
  - LAN's (10 Mbps), nº usuários limitado (OK)
  - WAN's (1,5 Mbps), nº usuários ilimitado (não OK)
- **Latência (*Latency*)**
  - Controlar a latência é importante para uma boa dinâmica e interação no AV
  - Latência baixa = maior freqüência de quadros por segundo = maior ilusão de realidade
  - Novamente: LAN's (OK), WAN's (não OK)
- **Confiabilidade (*Reability*)**
  - Associada a utilização de POC's
  - POC's causam atrasos inaceitáveis, PNOC's agilizam a comunicação
  - Uso de cada um depende da situação

Protocolos de comunicação



Broacast (apenas uma mensagem enviada, todos recebem)

Broadcast:

- Não é necessário uma conexão para cada usuário
- Uma mensagem enviada, todos recebem
- Útil principalmente sobre LAN's
- Não aplicável a WAN's: UDP *broadcast* não roteável

Protocolos de comunicação

- ***Distributed Interaction Simulation* (DIS)**
  - Sua origem está ligada ao SIMNET
  - Visa determinar uma arquitetura de comunicação comum para AVD's
  - Consiste de PDU's (originalmente 27 tipos) para comunicar eventos e estados do AV
  - Originalmente definido no padrão IEEE 1278-1993

Técnicas Associadas

- ***Heartbeats***
  - Idéia de enviar PDU's de atualização de estado obrigatoriamente num intervalo de tempo determinado
  - Importante para objetos com baixa ou nenhuma freqüência de atualização no AV (estáticos)
  - Essencial para manter novos usuários que entram no AV atualizados
  - Causa um aumento na utilização da largura de banda

Java3D

- **Características**
  - Parte da família de API's "*Java Media Family*" voltadas para a utilização de recursos multimídia, vídeo e gráficos em 3D
  - Utilizada para desenvolver softwares que tenham por objetivo exibir e interagir com cenários e gráficos em 3D
  - Conjunto de mais de 100 classes reunidas nos pacotes `javax.media.j3d`, `com.sun.j3d.utils` e `javax.vecmath`

Java3D

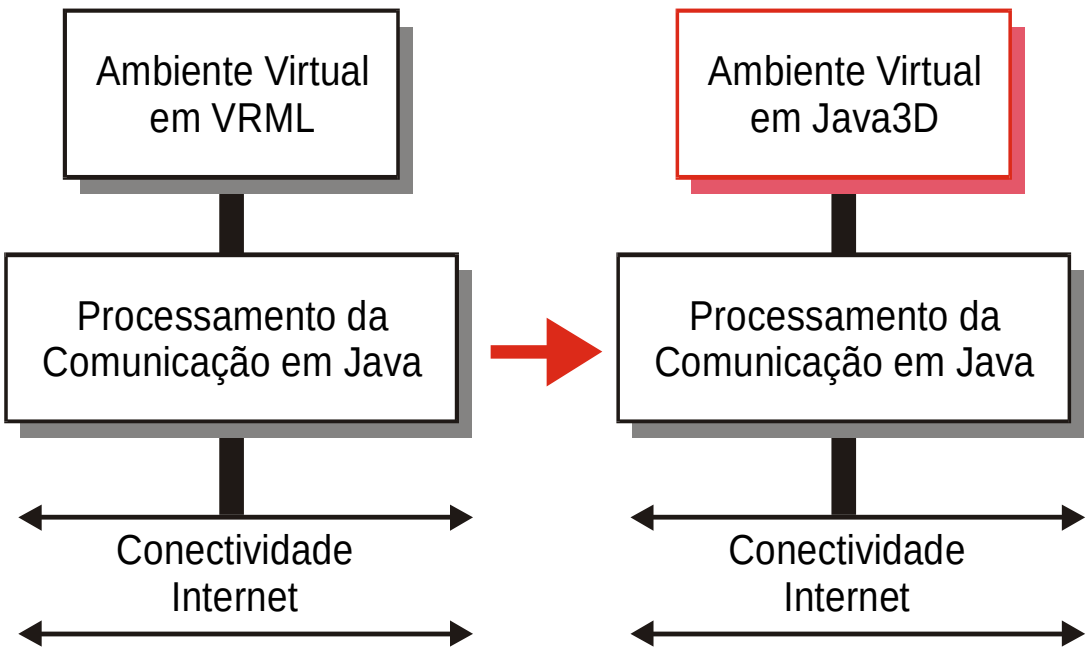
- **Algumas classes**
  - **`SimpleUniverse`:** universo simples, **`BranchGroup`** de visão criado automaticamente
  - **`Cone`**, **`Box`**, **`Cylinder`**, **`Sphere`:** figuras geométricas básicas
  - **`Transform3D`** e **`TransformGroup`:** transformações sobre os objetos visuais
  - **`Behavior`** e subclasses **`KeyNavigatorBehavior`**, **`PickRotateBehvaior`**, **`PickTranslateBehavior`**, **`PickZoomBehavior`:** interação com o AV

DIS-Java-VRML

- **Características**
  - Resultado de um dos grupos de trabalho do consórcio Web3D
  - API visa facilitar a construção de AVD's portáveis, oferecendo uma interface entre a linguagem Java e o ambiente virtual feito em VRML
  - Nelas estão contidas várias classes que implementam a maioria dos PDU's do protocolo DIS, bem como classes que controlam o processo de envio e recebimento dos mesmos

VRML - Java3D

Substituição VRML – Java3D



- Classes que interfaceiam Java e VRML foram desconsideradas
- Uso somente das classes que implementam os PDU's e mecanismos de controle de envio e recebimento dos mesmos

DIS-Java-VRML

FURB

DIS-Java-VRML: algumas classes

- **`BehaviorStreamBufferUDP` e `BehaviorStreamBufferTCP`:** controle de envio e recebimento de PDU's
- **Subclasses da classe `ProtocolDataUnit`**
  - **`EntityStatePDU, CollisionPDU, FirePDU`**
- **Subclasses da classe `SimulationManagementFamily`**
  - **`DataQueryPDU`, `DataPDU`, `Create/RemoveEntityPDU`, `StartResumePDU`**

Desenvolvimento do protótipo

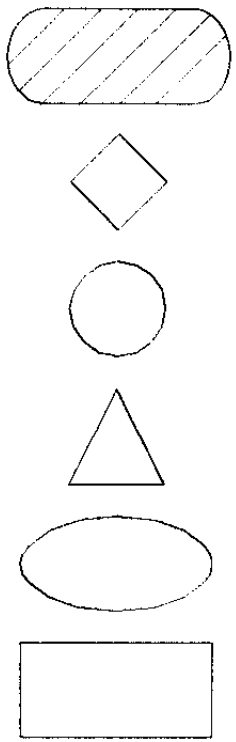
- **Problema:**
  - Ambiente virtual distribuído sobre uma rede local, com suporte a multiusuários e com uma interface não imersiva.
- **Requisitos identificados:**
  - Modelo de comunicação distribuído
  - Envio de mensagens através de *broadcast* UDP
  - Protocolo de comunicação baseado no DIS
  - Utilização da técnica de *heartbeats*

Desenvolvimento do protótipo

- **Técnicas de especificação:**
  - Para a estrutura gráfica do AV: **grafo de cena**
  - Para lógica dos processos: **fluxogramas**
- **Linguagem de programação:**
  - Java
- **Ferramentas:**
  - API J2SDK 1.3
  - API Java3D 1.2.1 Beta 2
  - API DIS-Java-VRML

Grafo de cena

Simbologia utilizada no grafo de cena



VirtualUniverse

Locale

Group

Leaf

NodeComponent

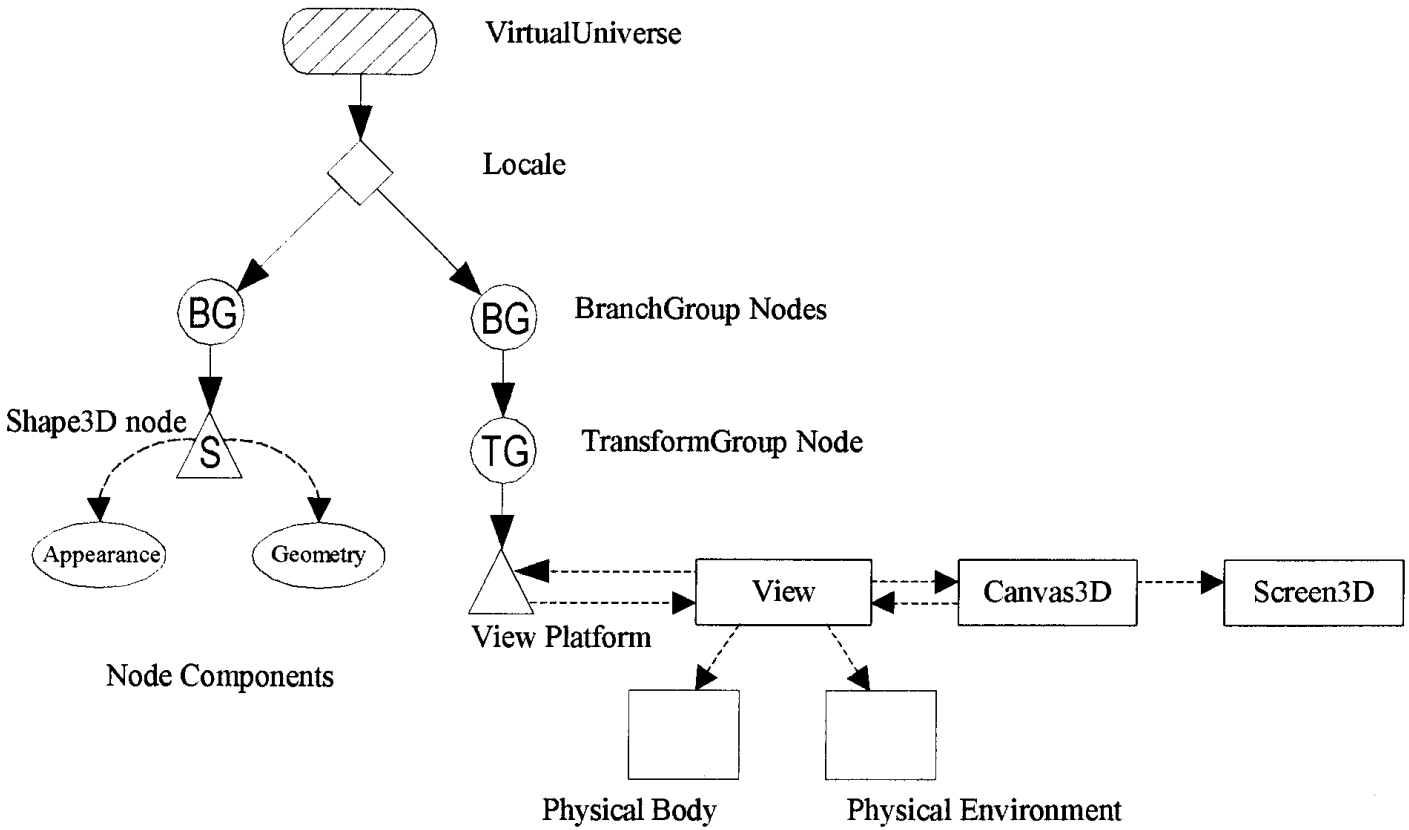
Outros objetos

ligação pai-filho

referência

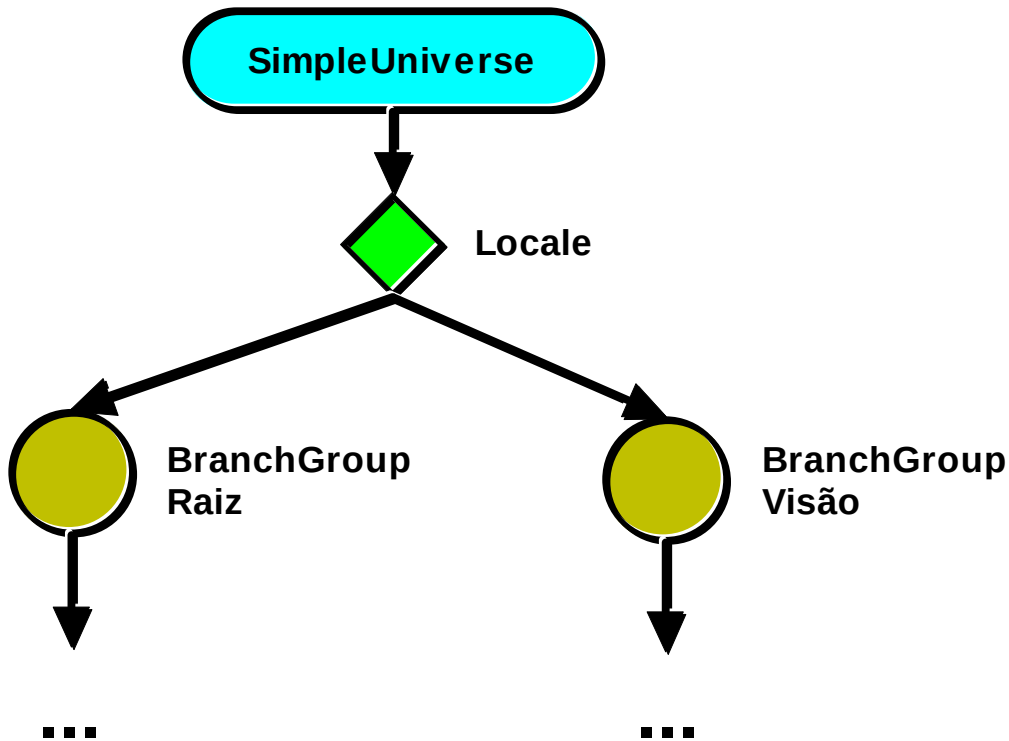
Grafo de cena

Grafo de cena exemplo



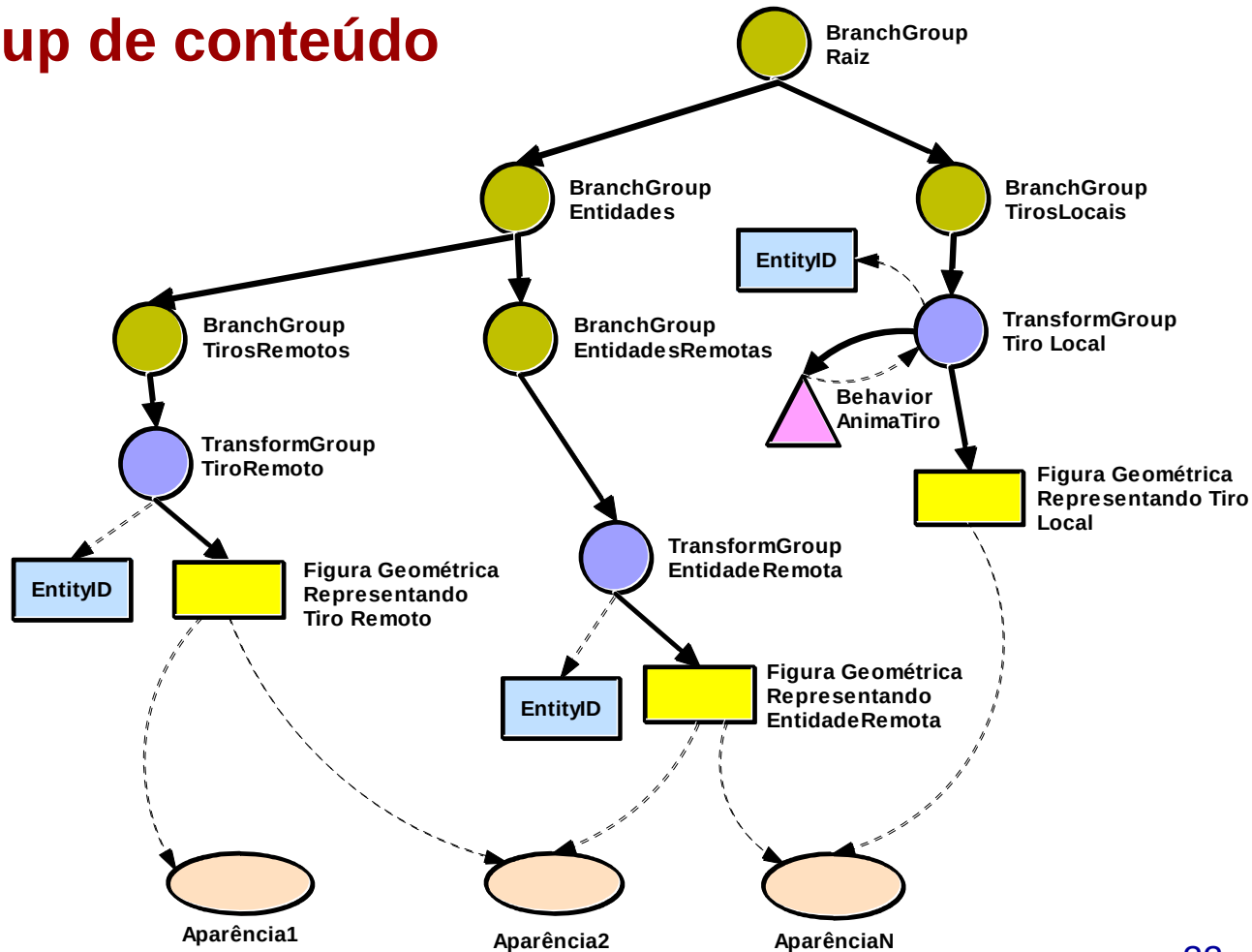
Especificação - Grafo de cena

BranchGroup's principais: conteúdo e visão



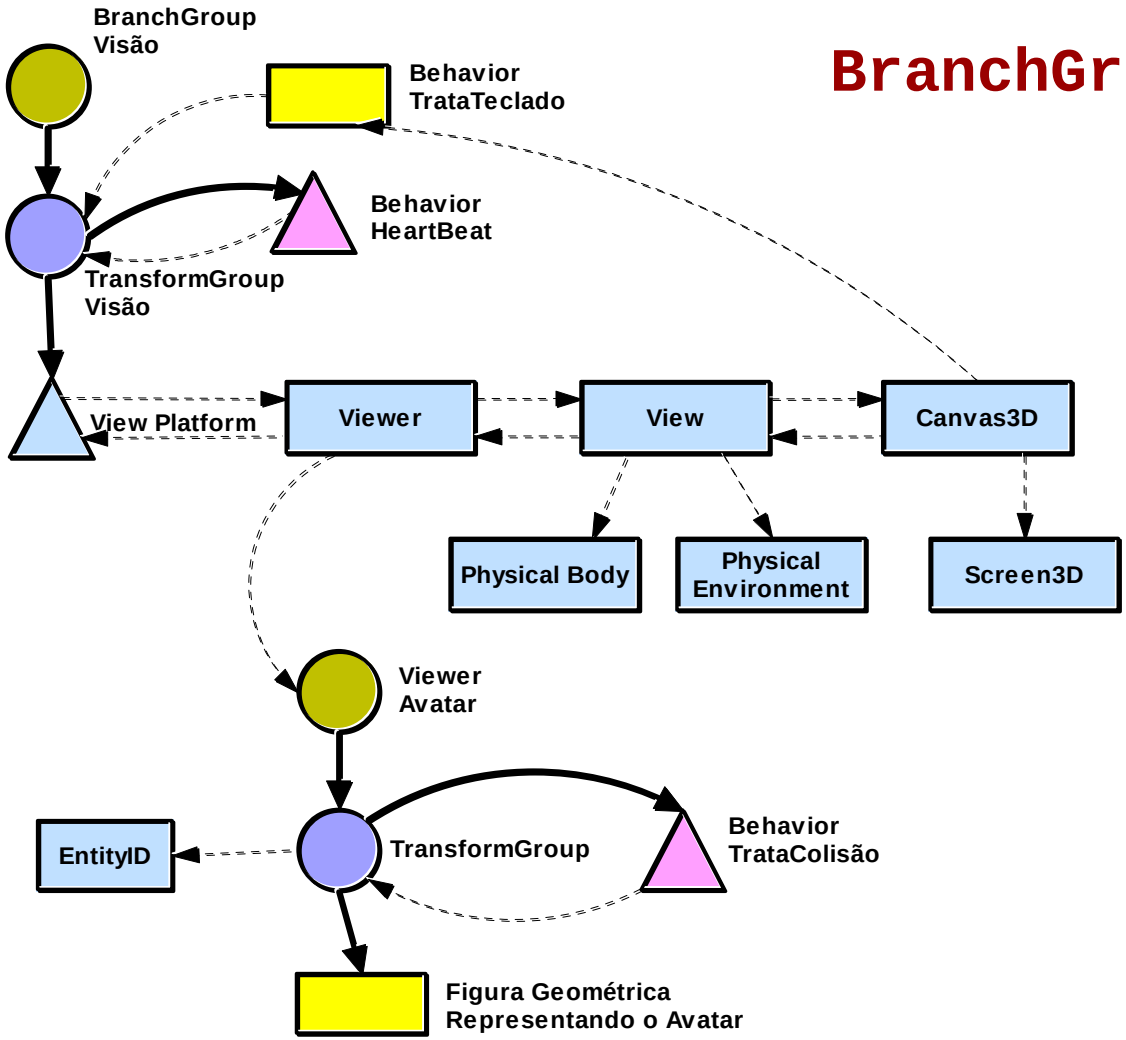
Especificação - Grafo de cena

BranchGroup de conteúdo



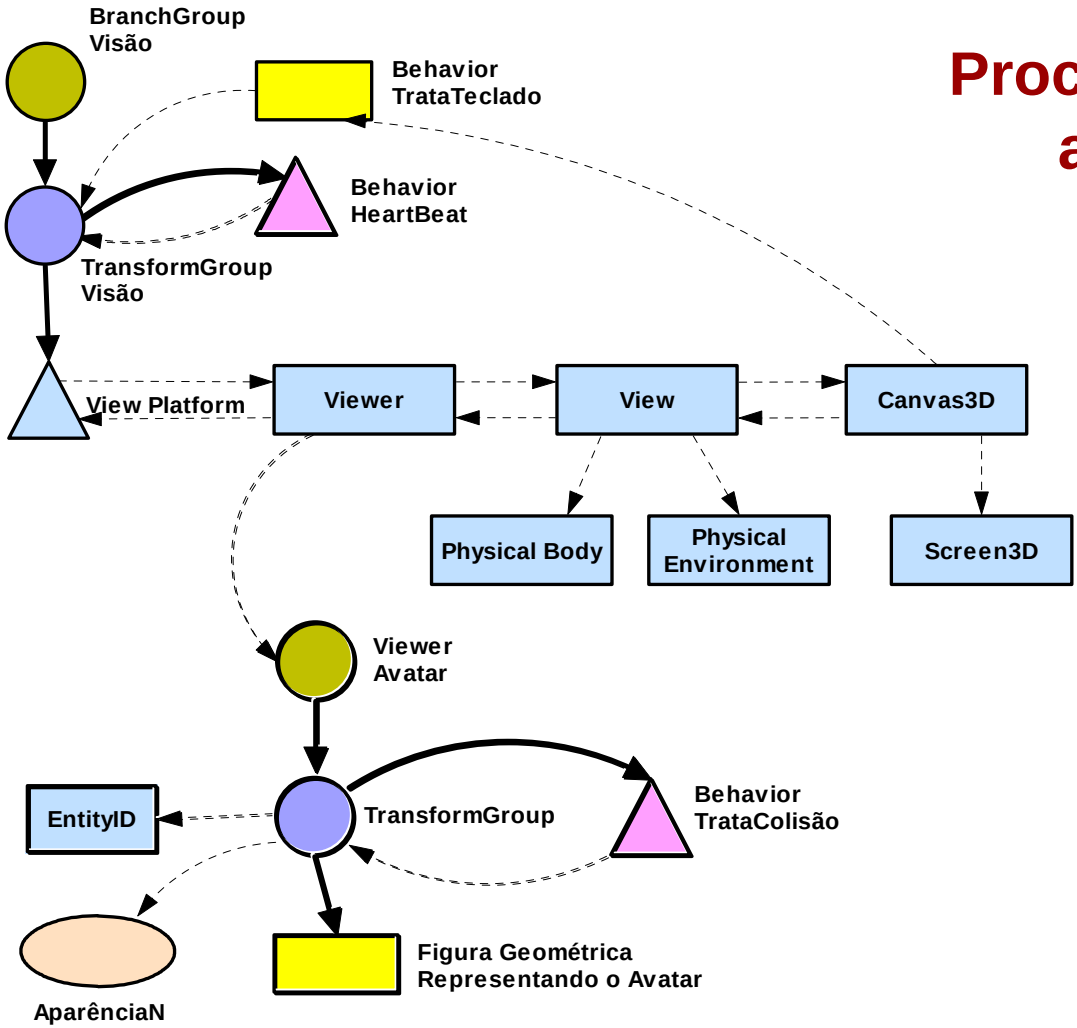
Especificação - Grafo de cena

BranchGroup de visão



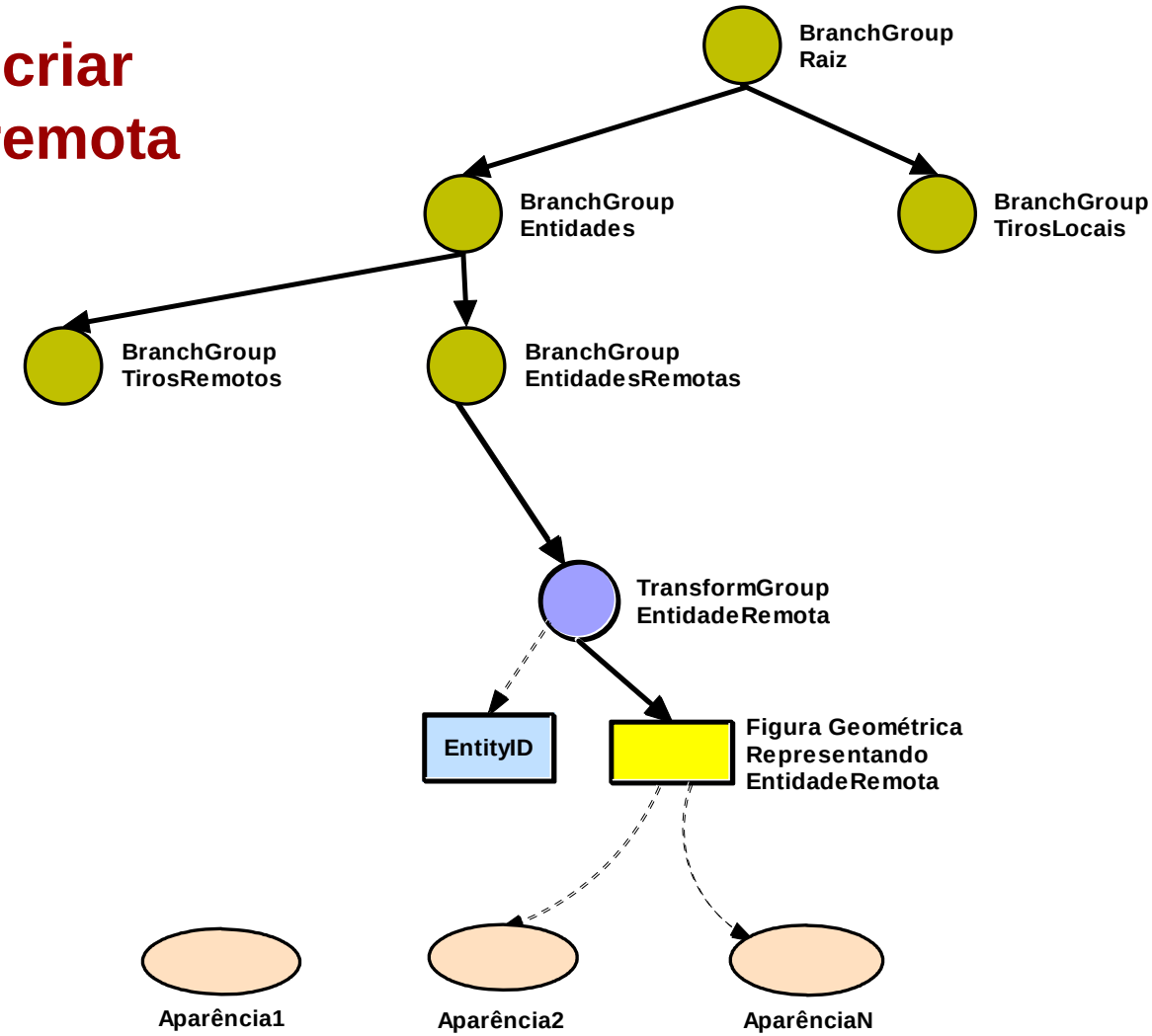
Principais processos

Processo: entrar no ambiente virtual



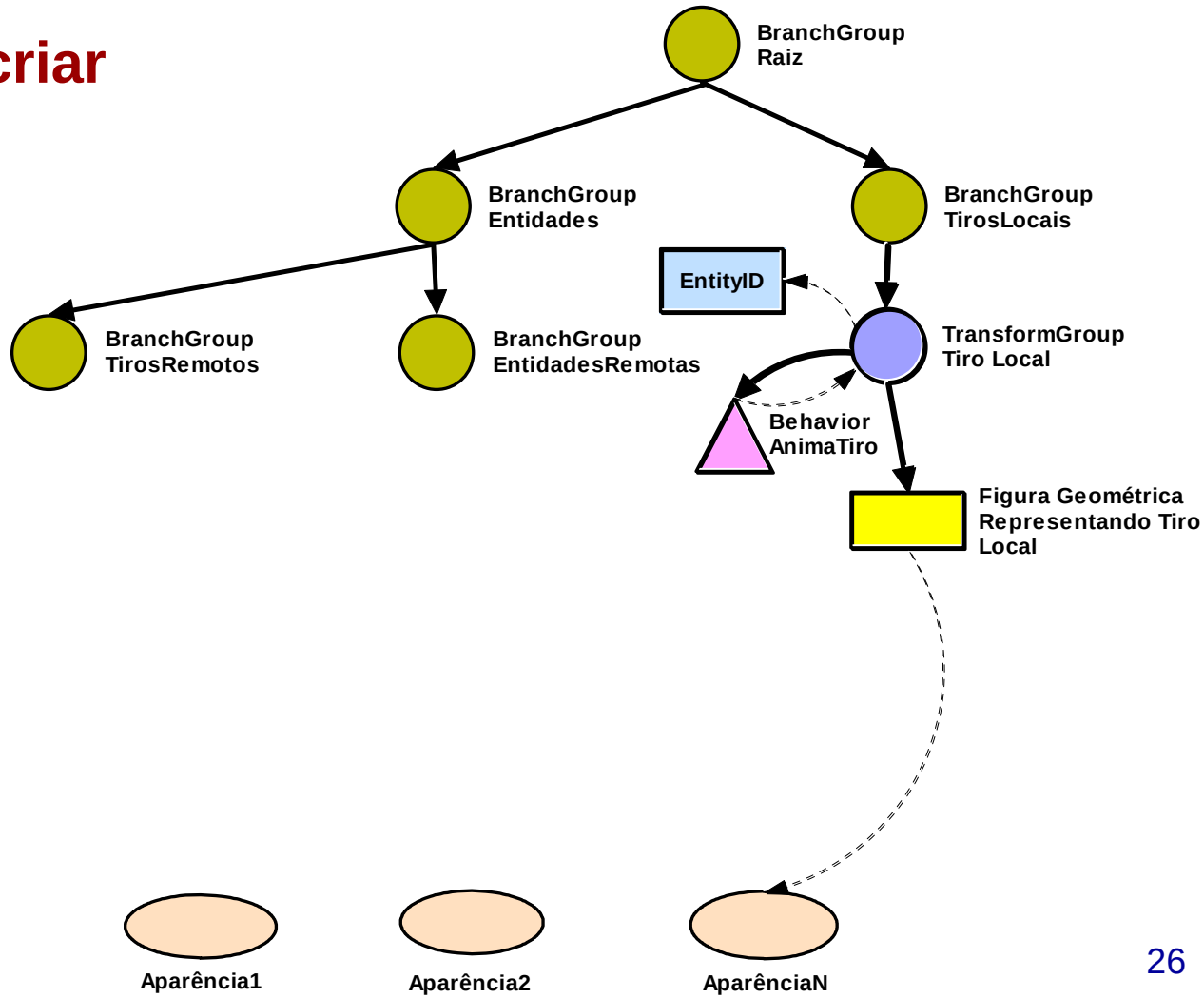
Principais processos

Processo criar entidade remota



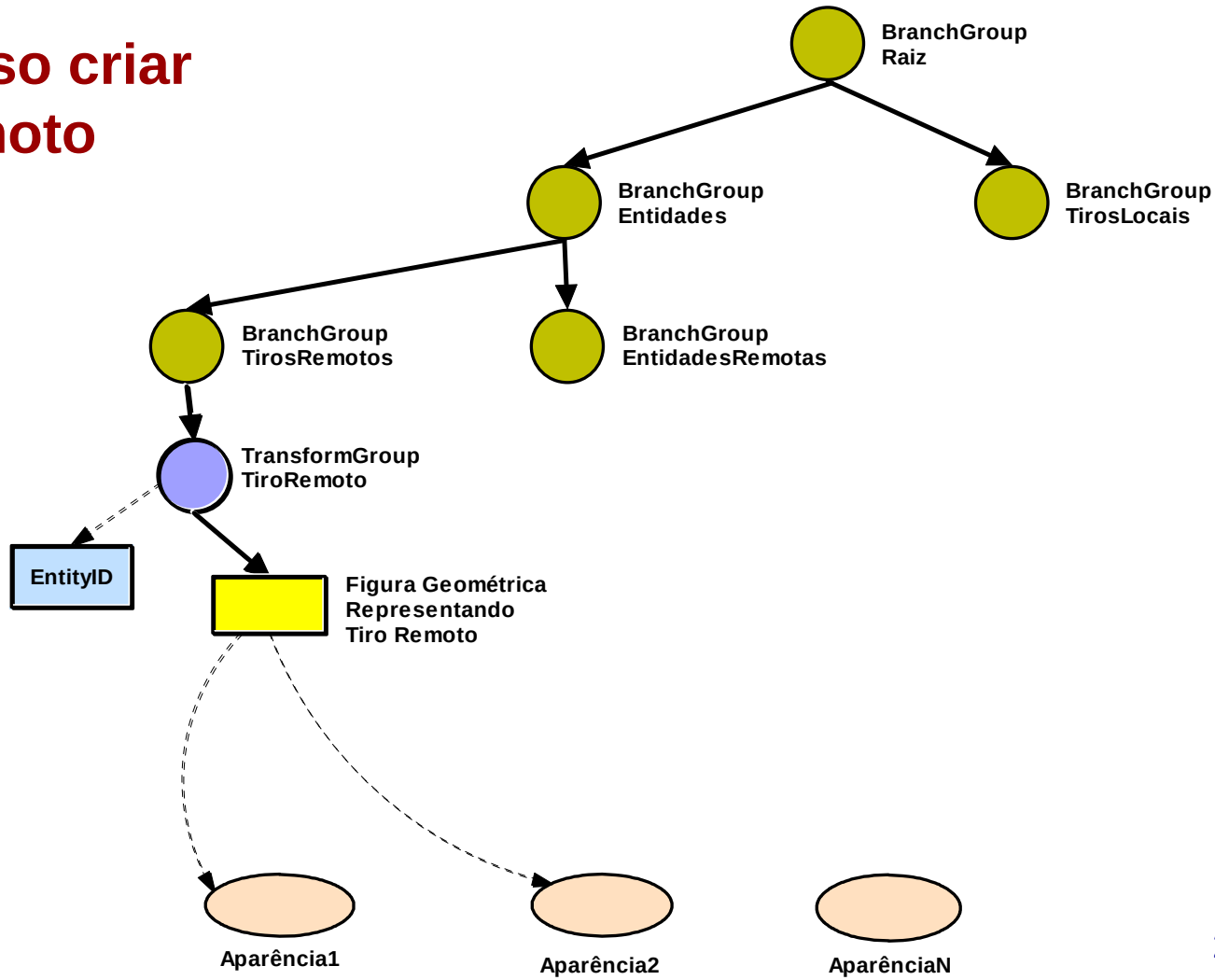
Principais processos

Processo criar tiro local



Principais processos

Processo criar tiro remoto



Resultados e discussão

Resultados alcançados e restrições encontradas:

- Apresenta a funcionalidade desejada com um grau de sincronia aceitável entre os usuários
- Foi implementado, adicionalmente, um controle rudimentar de colisão, aplicado durante a navegação no AV
- Mesmo em modo aramado e com um cenário gráfico muito simples, a renderização do AV é pesada em computadores AMD K6 475 Mhz e 128Mb de memória
- Isso faz com que a navegação no ambiente não seja suave (meio travada)

Resultados e discussão

Resultados alcançados e restrições encontradas:

- A utilização de computadores de capacidade de processamento diferentes gera animações sem suavidade nos demais usuários
- Por se utilizar de UDP *broadcast*, seu funcionamento está limitado a uma rede local
- O protocolo UDP é utilizado em todas as situações, o que pode levar a alguns problemas sérios caso algum pacote importante (Create/RemoveEntityPDU) seja perdido

Considerações Finais

Conclusões:

- Muito importante conhecer a fundo as características relacionadas a aspectos de rede, e conseqüentemente, suas limitações
- Importante conhecer as várias técnicas já desenvolvidas para AVD's que procuram, justamente, contornar as limitações impostas pela rede, utilizando de forma mais otimizada seus recursos
- A API do Java3D se apresentou como uma boa solução para a construção de AV pela sua simplicidade de uso, aliada a robustez de recursos e ótimo material de referência (documentação, tutoriais)

Considerações Finais

Conclusões:

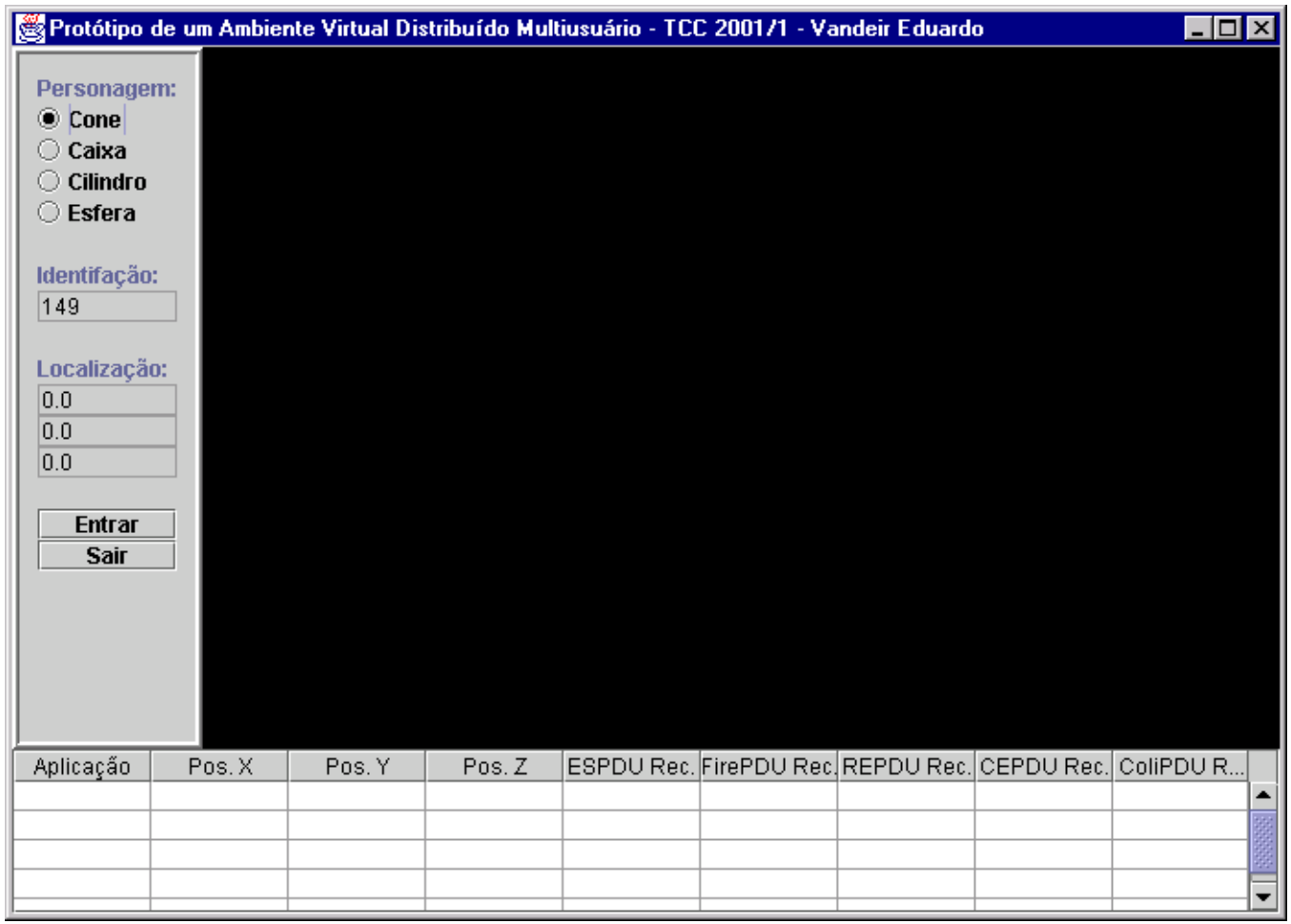
- A API do DIS-Java-VRML, apesar de ter sido desenvolvida para ser utilizada em AV construídos em VRML, foi muito útil por apresentar uma série de classes que puderam ser reutilizadas
- Com isso, pode-se deduzir que também a API do Java3D se apresenta como uma boa opção para a construção de AVD's
- Esse trabalho também pode contribuir consideravelmente para a conclusão de uma das tarefas do grupo de trabalho DIS-Java-VRML, que ainda está em aberto. Essa tarefa é justamente portar a API do DIS-Java-VRML para que a mesma também funcione com o Java3D

Considerações Finais

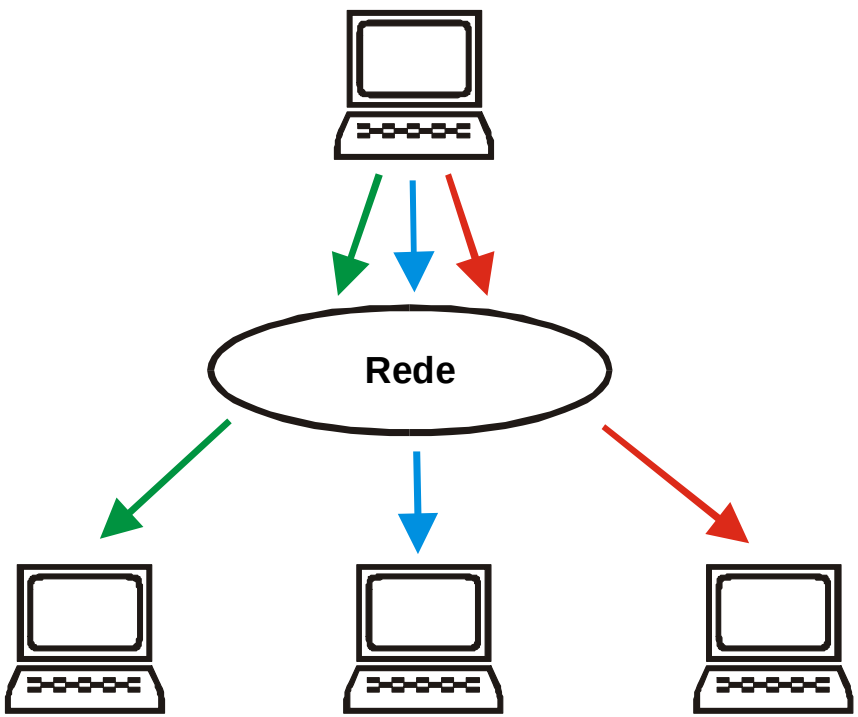
Extensões:

- Enriquecer o detalhamento do AV, utilizando classes adicionais (mais avançadas) do Java3D
- Permitir interação dos usuários com os objetos do AV
- Melhorar o processo de comunicação do AVD, fazendo uso de mais tipos de PDU's do protocolo DIS
- Implantar técnicas melhores de colisão e algoritmos de *dead reckoning*
- Criar uma API genérica para interfacear a API do DIS-Java-VRML e os elementos gráficos do AV criados com o Java3D (API DIS-Java-Java3D)

Demonstração do protótipo



Anexo A

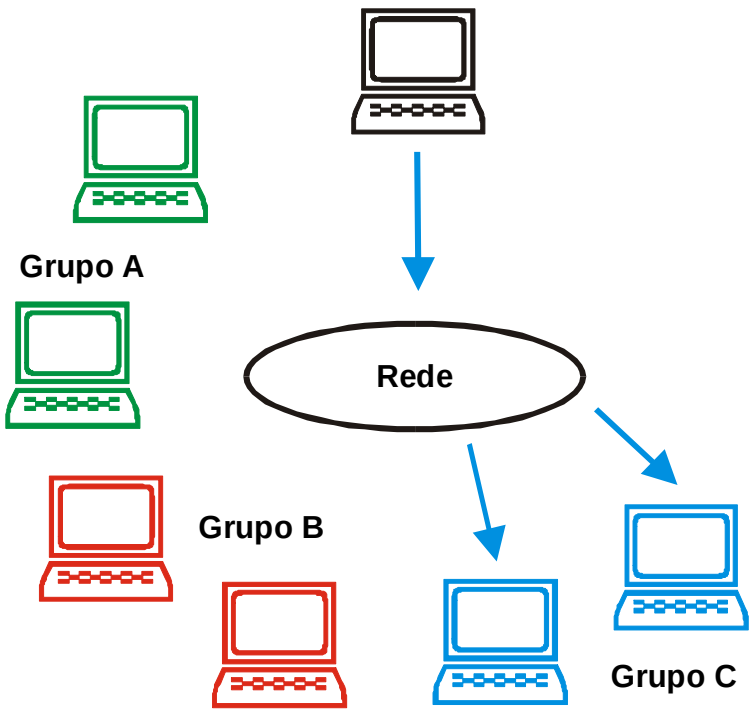


Unicast (3 mensagens enviadas, uma para cada usuário)

Unicast:

- Sincronia entre os usuários é elevada
- Nº elevado de conexões, dificuldade de gerenciamento das mesmas
- Nº elevado de mensagens geradas

Anexo B



Multicast (uma mensagem enviada para Grupo C, somente usuários desse grupo recebem)

Multicast:

- Endereços IP's classe D (224.0.0.0 a 235.255.255.255) são utilizados para definir canais de comunicação
- Host's só recebem mensagens dos canais nos quais tem interesse
- Host's precisam estar conectados à Internet através de roteadores compatíveis com multicast
- Indicado para AVD's sobre WAN's
- Pode ser utilizado para criar áreas de interesse dentro do AV

Anexo C

- **Algoritmos de *Dead Reckoning***