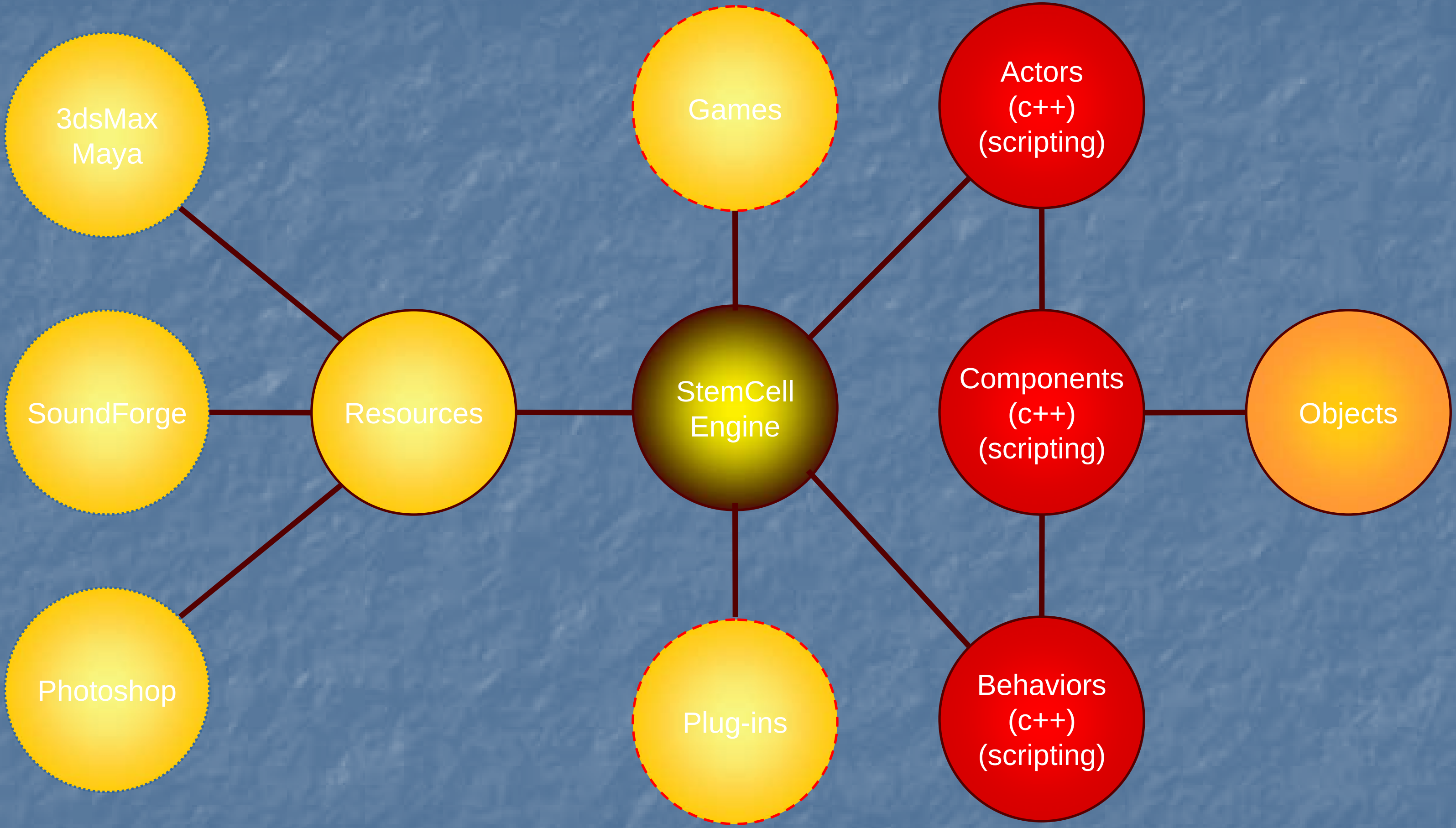
StemCell
Game Engine

Architecture overview

StemCell Game Engine ver1.0 - Architecture Overview



Architecture

1. **Resources** (models, animations, sounds, particles, textures, materials, shaders)
2. **Objects** (lights, mesh instances, animations, physics primitives)
3. **Components** (engine objects with scripting and eventing support)
4. **Actors** (component based data structures)
5. **Behaviors** (component based logics)
6. **Plug-ins** (custom resources, components, behaviors, actors)
7. **Games** (all of the above)

Resources

- **3d models** brought from 3dsMax or Maya (by use of content plug-ins)
- **Animations** brought from 3dsMax or Maya (by use of content plug-ins)
- **Sounds** – .wav, .ogg format
- **CellBuilder made** – particle systems, billboard based effects, materials, shaders

Objects

Objects created by the engine which we can manipulate from the API classes and methods

- **Lights**
- **Particle systems**
- **Mesh instances**
- **Physics primitives**
- **Sound sources**
- **Scripting objects**

Components

Engine components add eventing and scripting support to engine objects. They can also be used to construct data and logical structures called actors

- **Graphics** components
- **GUI** components
- **Physics** components
- **Sound** components
- Data components (**floats, strings, vectors**, etc)

Actors

Actors allow runtime creation of structured data sets and logical relations. We can attach to actors components, behaviors or even other actors, broadcast and propagate events and execute scripted code

- A **trigger** is an actor
  - Has a shape component (to determine the collision)
  - Has a behavior attached for what happens when the trigger is set
- A **pickup object** is an actor
  - It can contain components for its intersection shape, its type, its properties
  - Has a behavior attached for what should happen when the player picks it up
- The **main game character** can be actor
  - Containing all sorts of child actors for the statistics, inventory
  - Components for its speed, health, physical shape
- The **game** can be an actor
  - It can contain the main character actor
  - Enemy actors
  - Game statistics actor
  - HUD actor

Behaviors

Logical code (c++ or scripted) which can be attached or can influence directly other components or actors

- **Path finding** behavior
  - Can be applied to all the enemies into the scene
- **Main character idle** behavior
- **Flickering light** behavior
- **Camera animation** behavior
- **Scripted sequence** behavior

Plug-ins

Plug-ins are engine modules developed by users who want to add their components, actors, behaviors or just new code, objects and logic and integrate this directly with the engine core architecture

- Each of the engine module (**graphics, physics, scripting, logic**, etc) are plug-ins libraries developed by DevCell Software
- The **Samples example library** is a plug-in library
- Any new already developed libraries can be integrated with the engine (**NVidia PhysX, xAItment, Morpheme, SpeedTree**, etc)

Why use the component, behavior, actor constructions?

- They are designed to be **extensible**
- They are designed to work either with **scripting or c++**
- They support **eventing**
- They offer the possibility to construct your **data sets without writing code** (meaning we can use actors to describe our data without using class inheritance – which can only be done by code)
- They organize the logic of the application from the start into well defined elements

Why use the StemCell Engine?

1. Is created with the most basic development rules in mind
   1. **Extensibility**
   2. **Flexibility**
   3. **Tool driven**
2. Is developed by people who love creating and playing games

Questions

STEM Cell
Intelligent Technology
Copyright © 2008 DevCell Software

Thank you!