Stavros Vassos, DIS Room B214, vassos@dis.uniroma1.it
Stefano Cianciulli, stefano.cianciulli@gmail.com

VISUALIZING SERVICE BEHAVIOR COMPOSITION

Examples in video-game scenarios!

Behavior Composition with JaCO

- AI for non-player characters (NPCs) in video games
- Unity game engine
- Angry Bots Patrolling Domain for Behavior Composition
  - Demo / Getting started guide
  - Behaviors in TGF (Trivial Graph Format)
- JaCO server
  - API and usage
  - Behaviors in XML
- A possible application in interactive storytelling

Non-player Characters in Video Games

- Game engine, a typical example:
  - C++
  - Creates game-world objects with (x,y,z) coordinates and calculates what happens to them on every frame
  - E.g., a crate is up in the air on frame1. On frame 2 the game engine will calculate the new position, etc

Non-player Characters in Video Games

- Game engine, a typical example:
  - C++
  - Creates game-world objects with (x,y,z) coordinates and calculates what happens to them on every frame
  - E.g., a crate is up in the air on frame1. On frame 2 the game engine will calculate the new position, etc
  - Same for **non-player characters** (NPCs)!

Finite State Machines (FSMs)

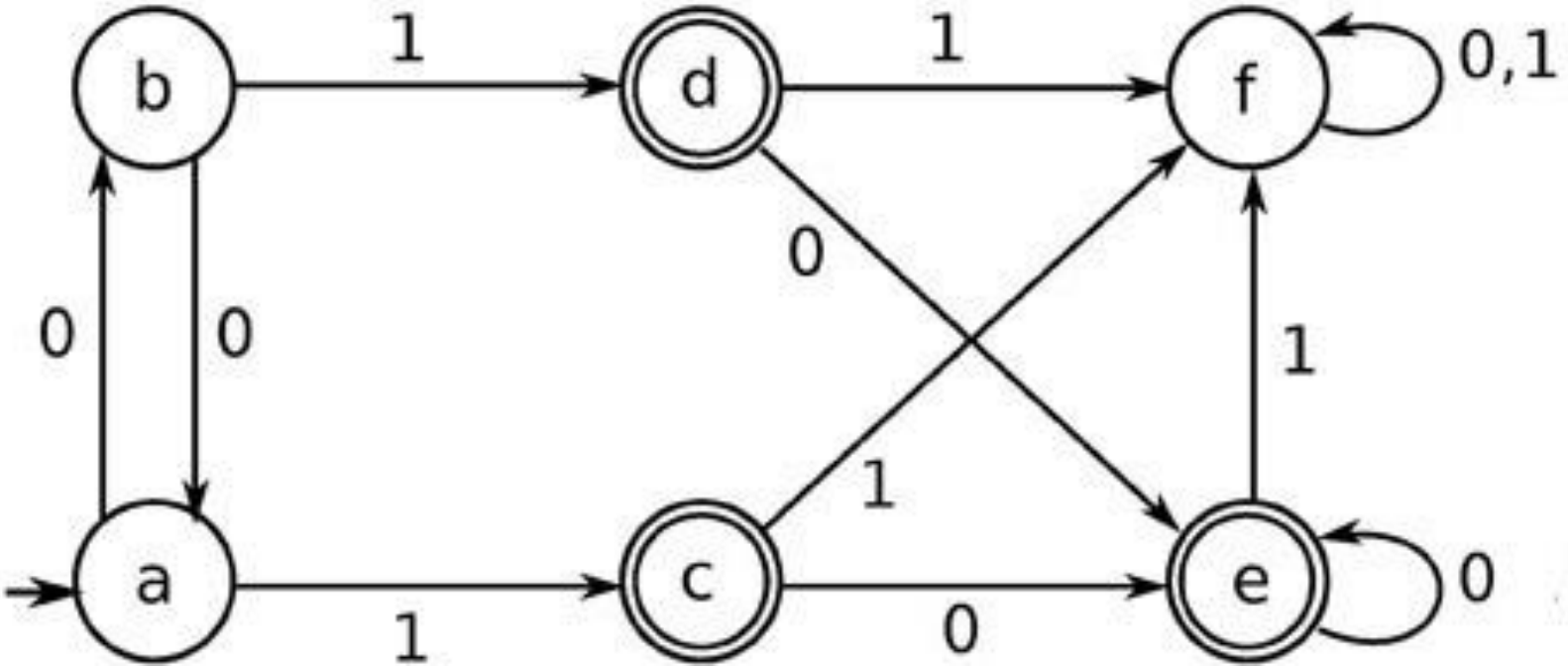
- Video Games:
  - Finite State Machines
  - Decision Diagrams
  - Behavior Trees
  - Goal Oriented Action Planning
- Academic AI on agents:
  - Knowledge representation, First-order logic, Classical planning, Planning with preferences, …
  - Belief-Desire-Intention architecture, Agent-based programming, …
  - Probabilistic reasoning, Bayesian networks, Utility theory, Markov Decision Processes, …

Finite State Machines (FSMs)

- Video Games:
  - Finite State Machines
  - Decision Diagrams
  - Behavior Trees
  - Goal Oriented Action Planning
- Academic AI on agents:
  - Knowledge representation, First-order logic, Classical planning, Planning with preferences, …
  - Belief-Desire-Intention architecture, Agent-based programming, …
  - Probabilistic reasoning, Bayesian networks, Utility theory, Markov Decision Processes, …

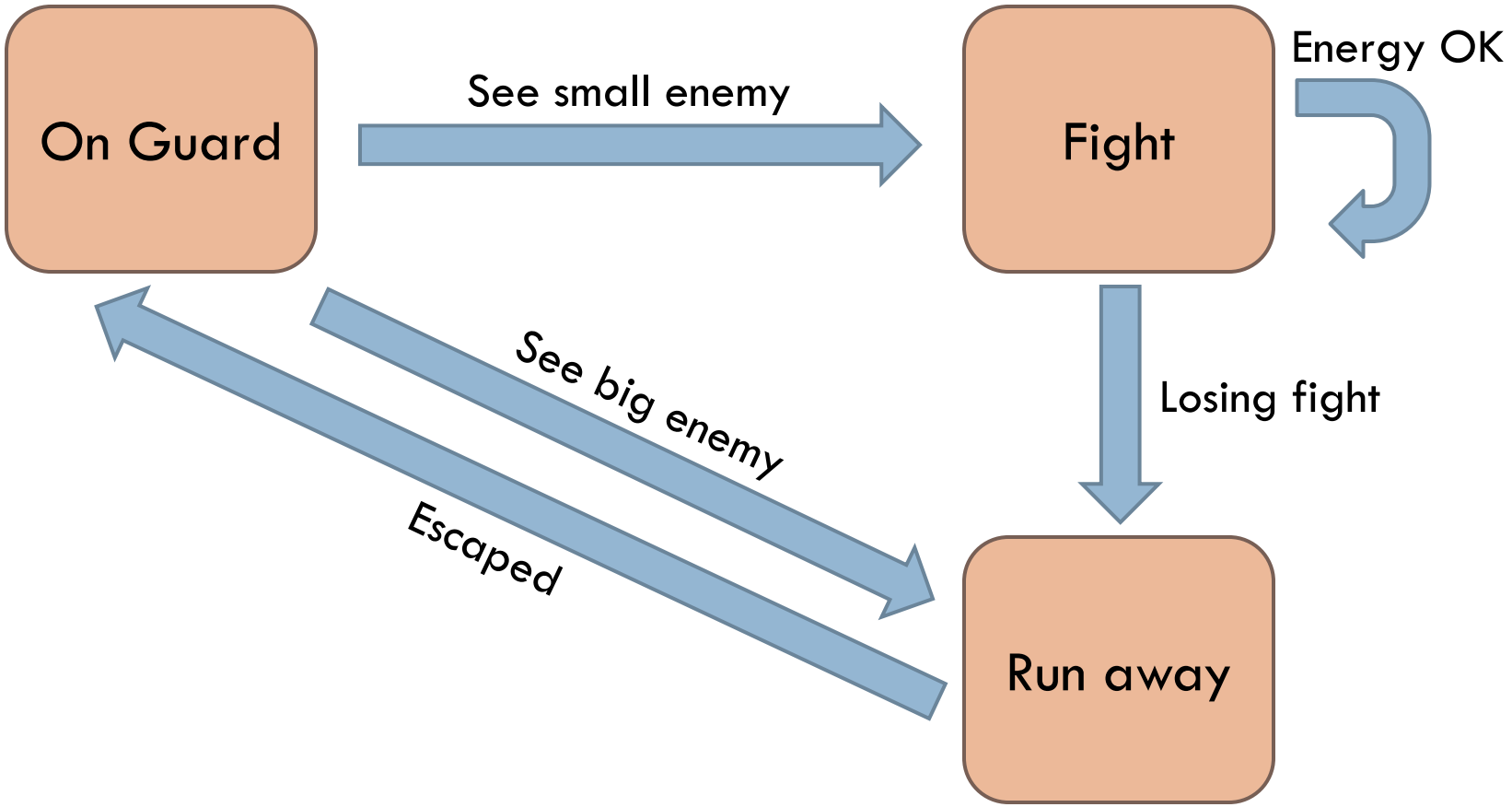
Finite State Machines (FSMs)

- Simple transition systems



Finite State Machines (FSMs)

- NPC behavior based on high-level states



Finite State Machines (FSMs)

- Traditionally one of the first techniques for NPC behavior
- Very simple to understand
- Very simple to implement
  - E.g., directly using if-then-else statements

Finite State Machines (FSMs)

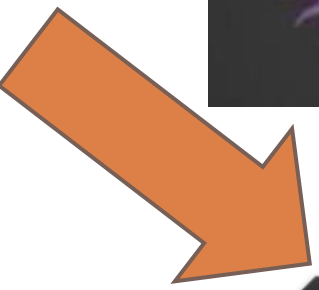

```
int NPC::think(){
  if (state==ONGUARD && seeSmallEnemy()){
     state=FIGHT;
     makeScarySound();
  }
  else if (state==FIGHT && energy>30){
     ...
  }
  else if ...
}
```

Behavior Composition in videogames?

- Transition systems are already used in videogames
  - Each NPC expresses an available behavior
  - A target behavior can be used to express a “virtual” intended behavior
  - A controller can be used to orchestrate the NPCs
- Two examples
  - Angry Bots Patrolling Domain
  - A possible application in interactive storytelling

Game development with Unity

- Amazing tools available for (indie) game developers!



ShiVa3D

UDK

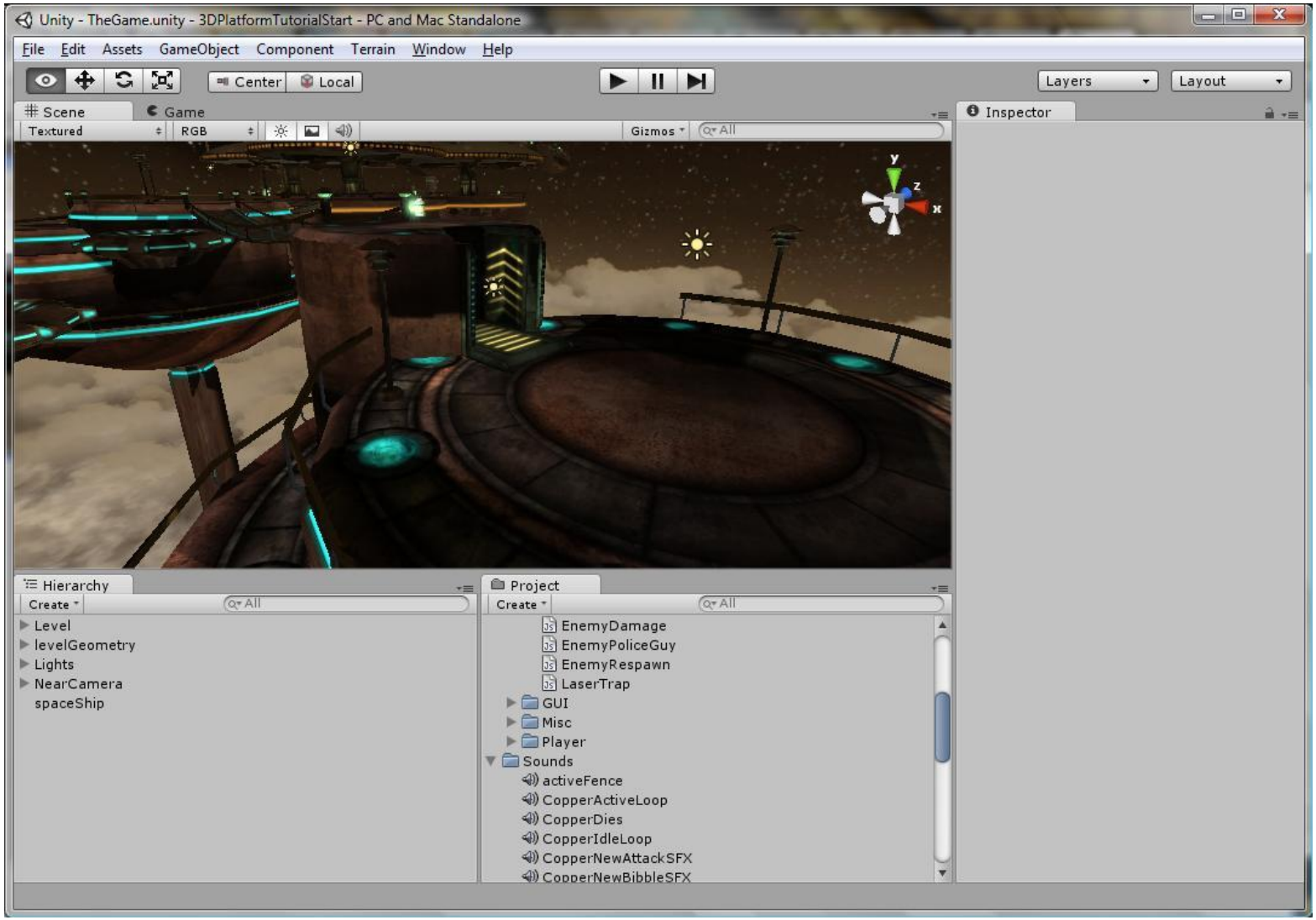
unity

VALVE

Game development with Unity

- Integrated Game Development Environment
- C#, Javascript, Boo programming languages
- Asset-centric instead of code-centric, adopting a look and feel like 3D CAD software

Game development with Unity



Game development with Unity

- Terminology
  - Project
  - Scene
  - GameObject and Component
  - Asset and Prefab
  - Script

Game development with Unity

- 3D platform game tutorial available online by Unity3D
  - http://unity3d.com/gallery/demos/demo-projects
  - http://u3d.as/content/unity-technologies/3d-platformer-tutorial/3yF

Game development with Unity

- Sections 1,2 of the tutorial
  - Start with an empty platform level
  - Add our player: Lerpz
  - Add a camera that follows him
  - Add a 3rd person controller to control Lerpz
  - Tweak his movement
- Section 5
  - Add NPCs!

Game development with Unity

- Quick demo using
  - Lerpz
  - SpringFollowCamera
  - ThirdPersonController
  - CharacterController
  - ThirdPersonPlayerAnimation

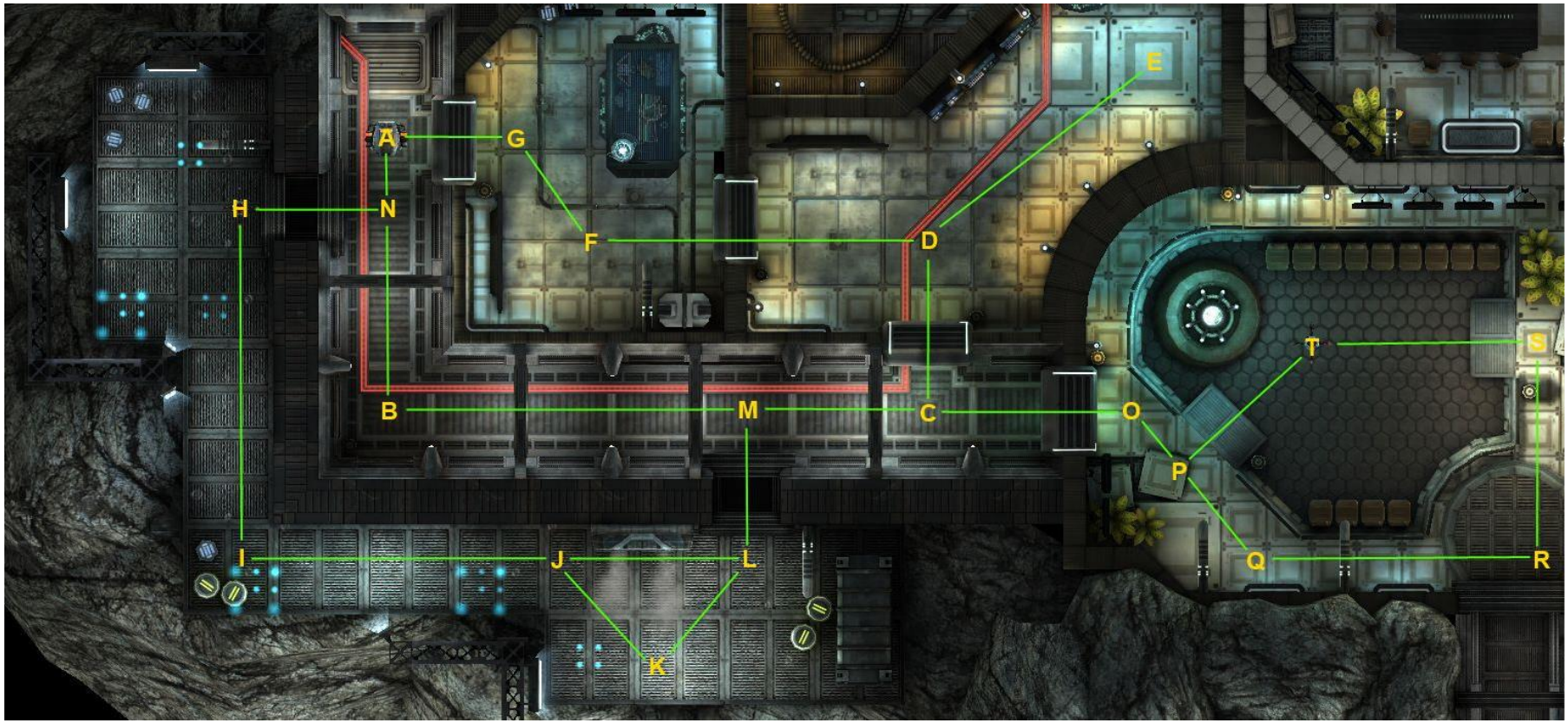
Game development with Unity

Game development with Unity

- Angry Bots Patrolling Domain
- Getting started
  - http://jaco.dis.uniroma1.it
  - Example1: download jaco.jar and patrolling-win.zip
  - Initialize the standalone JaCO server (Java 1.7 required). Run it from command line:
    java –jar jaco.jar
  - Unzip and run the patrolling demo

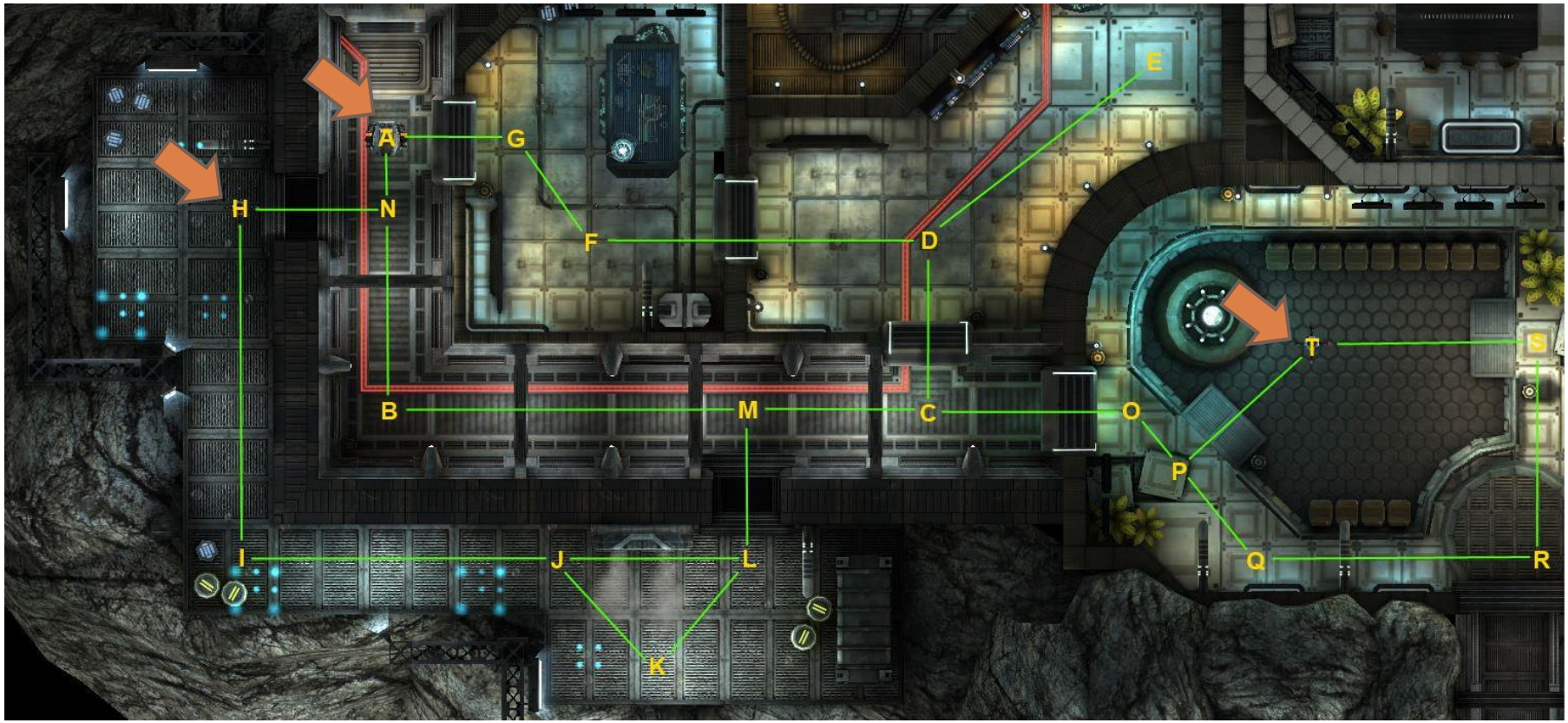
Angry Bots Patrolling Domain

- 20 Points of interest
- 3 Robots



Angry Bots Patrolling Domain

- 20 Points of interest
- 3 Robots



Angry Bots Patrolling Domain

- Stefano Cianciulli Non-Player Character Behavior Composition in Unity Game Engine, M.Sc. Thesis, March 2013
  - Angry Bots Patrolling Domain project: http://github.com/CianciuStyles/angrybots-jaco
  - JaCO behavior composition server project: http://github.com/CianciuStyles/jaco-web-service
  - JaCO API/examples/server/executable demo http://jaco.dis.uniroma1.it

- Next: more details on these on Jaco-patrolling.pdf

Angry Bots Patrolling Domain

Further development

- Add a shared environment
  - This is the environment in which all services act
  - Possible MSc thesis: extend the system to include a shared environment
- More advanced forms of synthesis
  - E.g., describing the target behavior in terms of sequencing of goals to be achieved
  - Possible MSc thesis: extend the system to more advanced forms of service synthesis and composition
- Exploit this idea for more flexible/interactive storytelling in video games (also for MSc thesis)

Interactive storytelling: a possible application

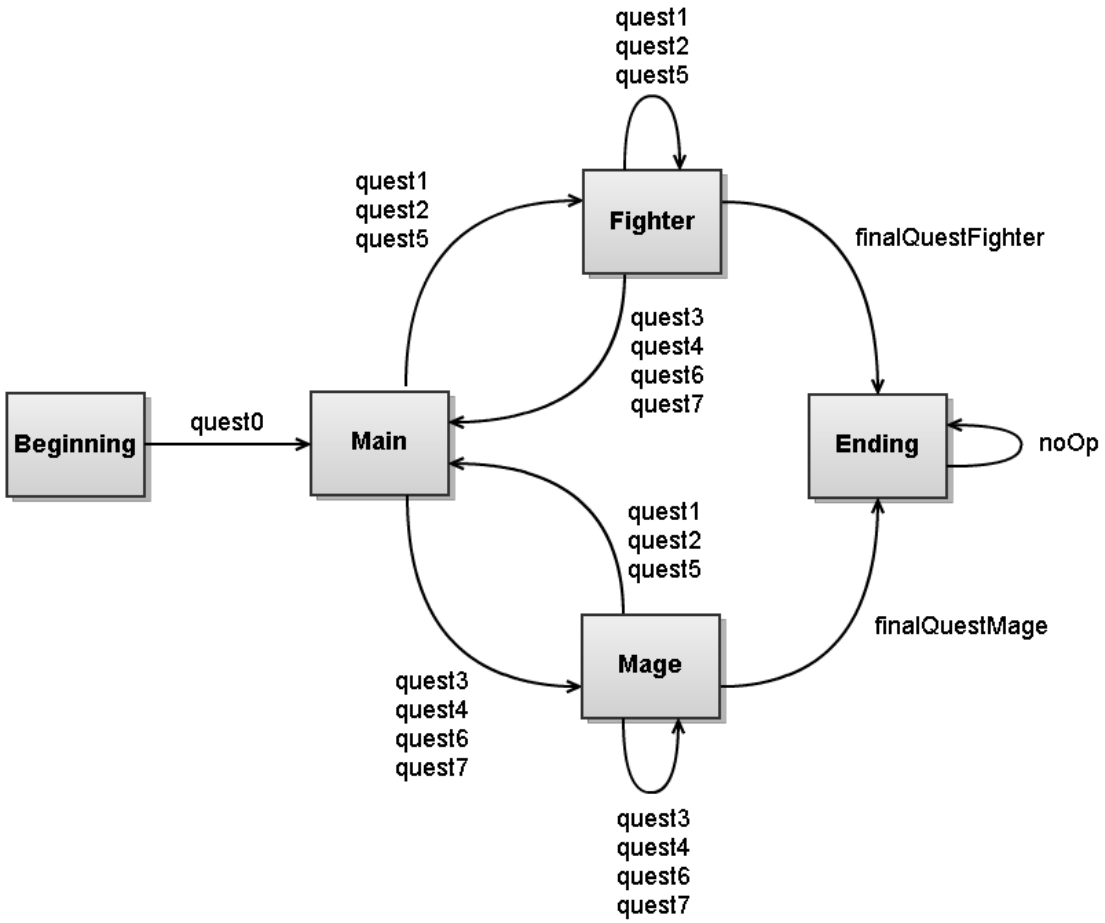
- A simple example of a nonlinear story

Interactive storytelling: a possible application

- A simple example of a nonlinear story
  - The player embarks to a journey to become a powerful fighter or magician
  - The story evolves by means of self-contained quests

Interactive storytelling: a possible application

- A simple example of a nonlinear story



Interactive storytelling: a possible application

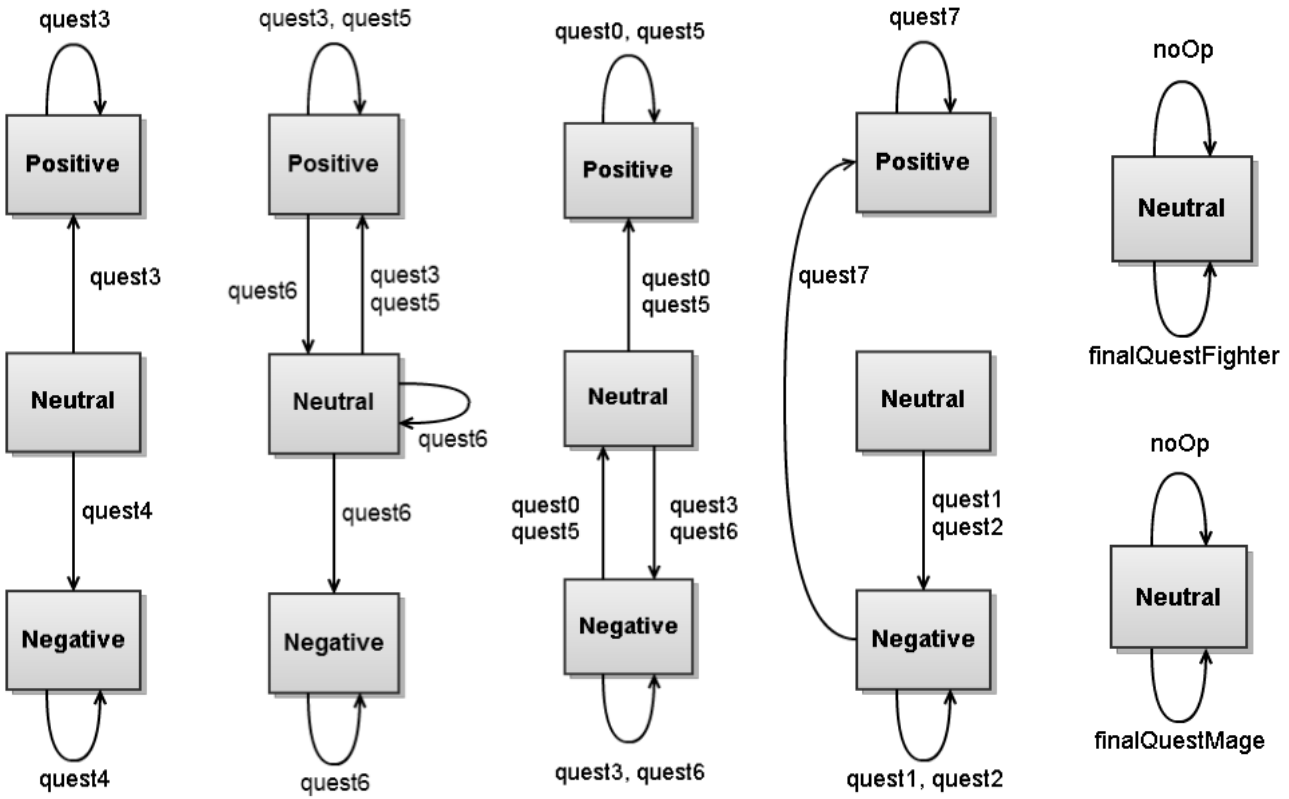
- A simple example of a nonlinear story
  - The player embarks to a journey to become a powerful fighter or magician
  - The story evolves by means of self-contained quests
  - Each node in this transition system is a **decision point**
  - The story is led by an **"AI Director"** who may chose the next action based on different parameters, e.g., player's satisfaction, time of play, etc.

Interactive storytelling: a possible application

- A simple example of a nonlinear story
  - The quests can be initiated/facilitated/handled by some key NPCs that participate in the story, e.g., the mayor of a small town, an evil wizard, a village
  - Each NPC can be involved in more than one quests
  - Each quest can be handled by more than one NPC
  - Each quest affects the mood of the NPC

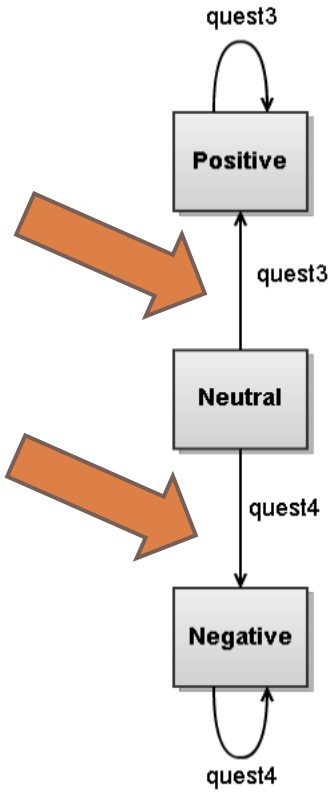
Interactive storytelling: a possible application

- A simple example of a nonlinear story



Interactive storytelling: a possible application

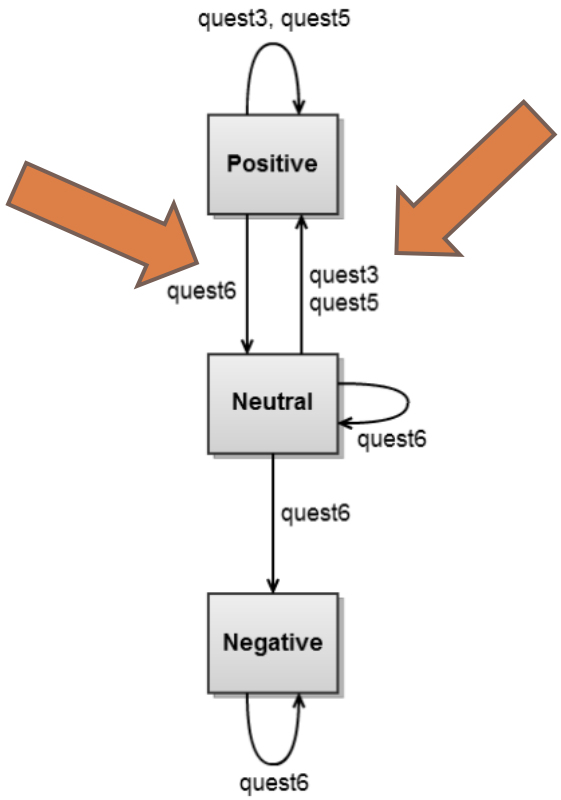
- A simple example of a nonlinear story



- NPC1
  - Can be used only for quest3 or quest4
  - Either of them fixes the NPC into a positive or negative state

Interactive storytelling: a possible application

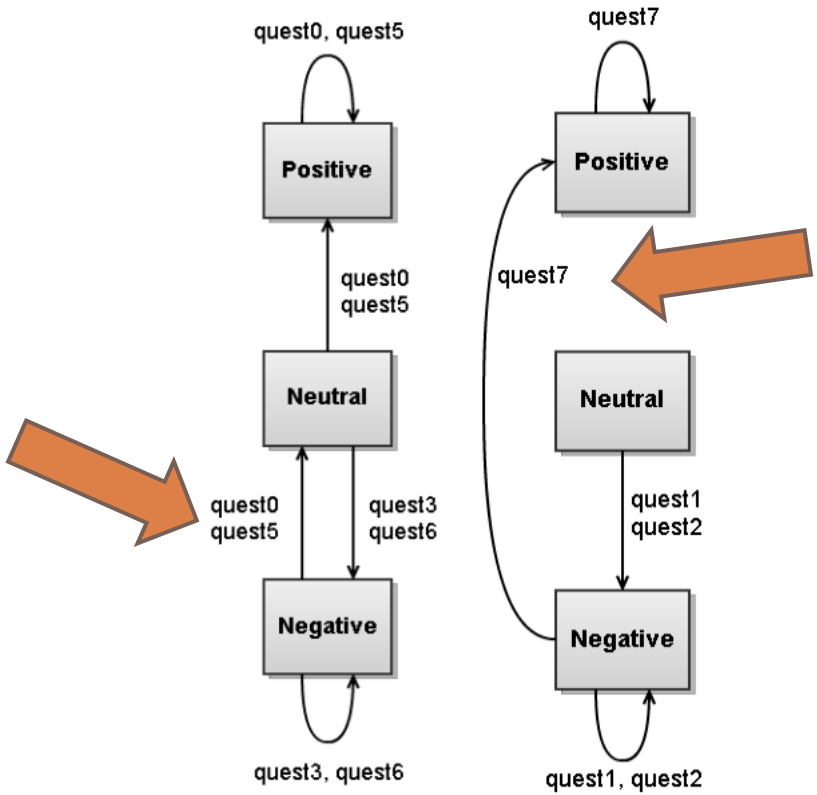
- A simple example of a nonlinear story



- NPC2
  - A positive mood is hurt by quest6
  - Note the nondeterminism
  - [Note that for simplicity we assume that quests may be performed more than once]

Interactive storytelling: a possible application

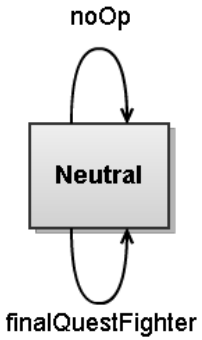
- A simple example of a nonlinear story

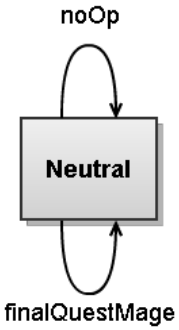


Interactive storytelling: a possible application

- A simple example of a nonlinear story

  - NPC5, NPC6
    - Special ending of the game depending on the fighter or magician track
    - [Note that the runs of the transition system are infinite]





Interactive storytelling: a possible application

- A simple example of a nonlinear story
  - The quests can be initiated/facilitated/handled by some key NPCs that participate in the story, e.g., the mayor of a small town, an evil wizard, a village
  - Each NPC can be involved in more than one quests
  - Each quest can be handled by more than one NPC
  - Each quest affects the mood of the NPC
  - **Deadlocks** arise due to overlap! How can this be handled in an automated way?

Interactive storytelling: a possible application

- A simple example of a nonlinear story
- Behavior composition
  - The transition systems representing the mood of NPCs are the available behaviors
  - The transition system representing the nonlinear story is the target behavior
  - A controller can be extracted from the composition, showing how to orchestrate the available behaviors in order to avoid deadlocks
- We can connect to JaCO from the command line

Interactive storytelling: a possible application

- XML files for the example
  - http://jaco.dis.uniroma1.it/#example2
  - http://jaco.dis.uniroma1.it/storytelling.zip
- cURL
  - http://curl.haxx.se/
- Rest client
  - http://restclient.net/
  - http://code.google.com/p/rest-client/

Interactive storytelling: a possible application

- Accessing jaco.dis.uniroma1.it
  - Inside DIS: interaction-dis.bat
  - Outside DIS: interaction-jaco.bat
  - Accessing the local server: interaction-local.bat