OPERAČNÍ PROGRAM PRAHA ADAPTABILITA

EVROPSKÁ UNIE

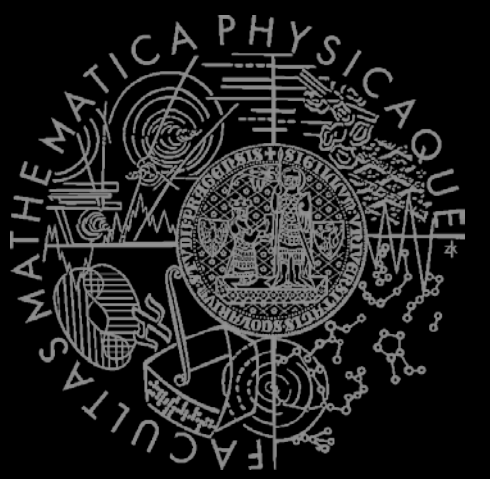
EVROPSKÝ SOCIÁLNÍ FOND

Pogamut 3
Lekce 1 - Úvod

PRAHA & EU

INVESTUJEME DO VAŠÍ BUDOUCNOSTI

Faculty of mathematics and physics
Charles University at Prague
24th February 2012



UT2004 & UE2 bots made easy!

Pogamut 3

Lecture 1 – Gentle introduction

Virtual worlds

familiar

simplified reality

gravity

solid walls

simulated

communication

real-time!

Virtual humans

Intelligent virtual agents (IVAs)

needs

goals

motivations

rules

emotions

communication

real-time!

Our scope – UT2004, UE2

needs

goals

motivations

rules

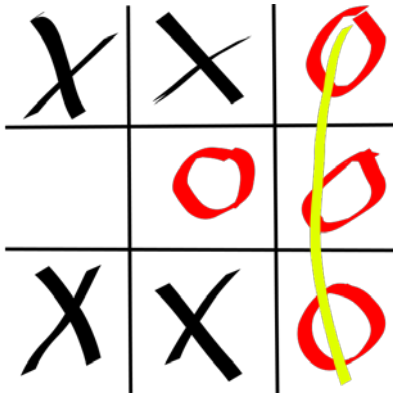
emotions

communication

real-time!

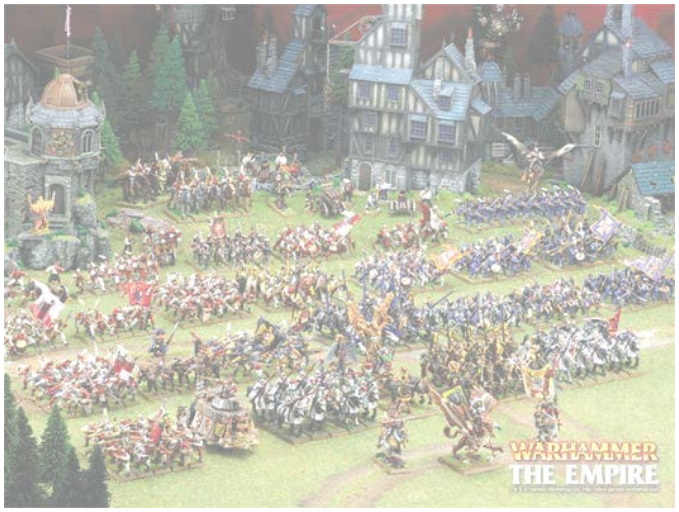
World properties

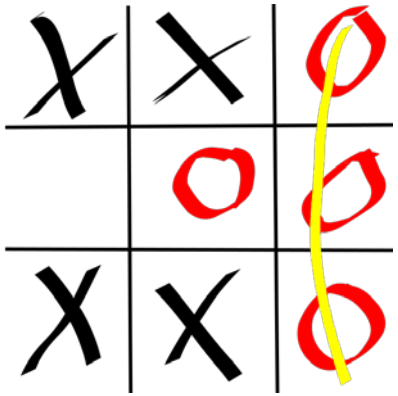
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy



World of TicTacToe?

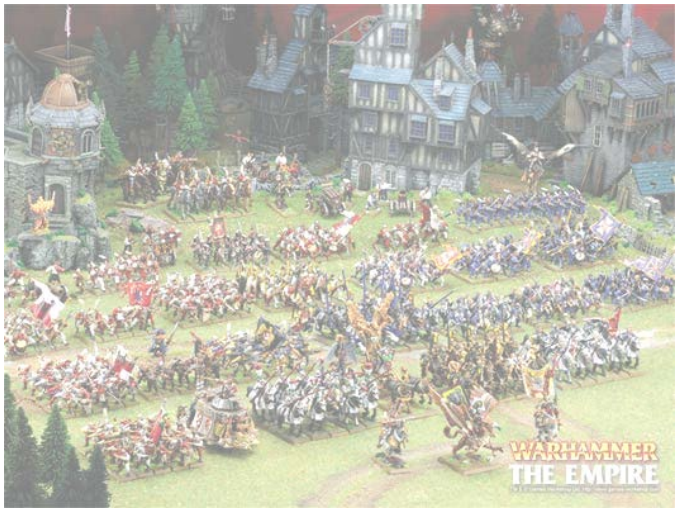
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

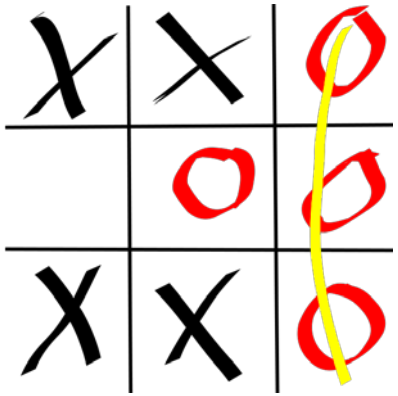




World of TicTacToe?

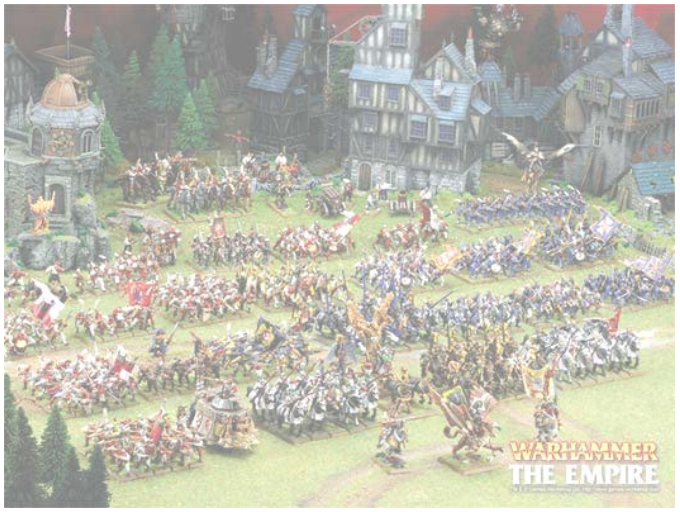
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

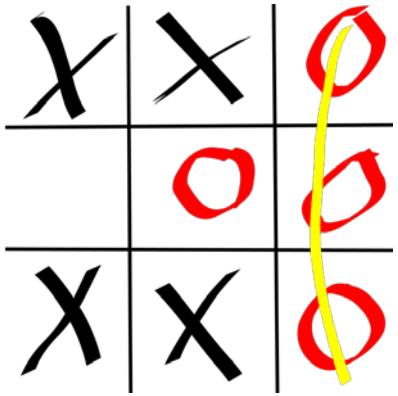




World of TicTacToe?

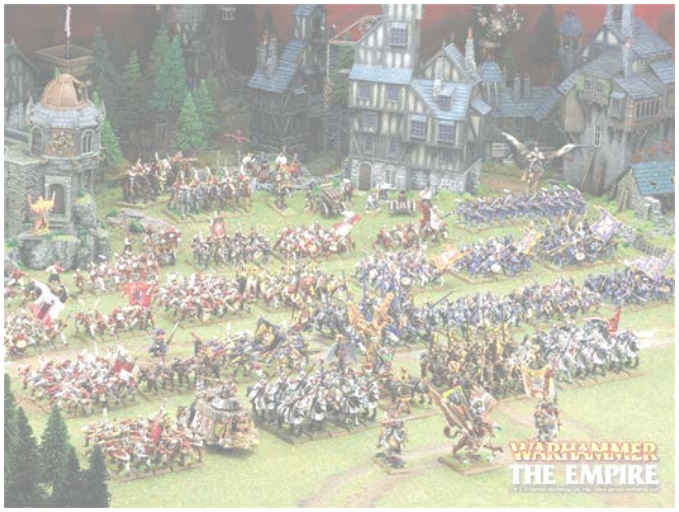
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

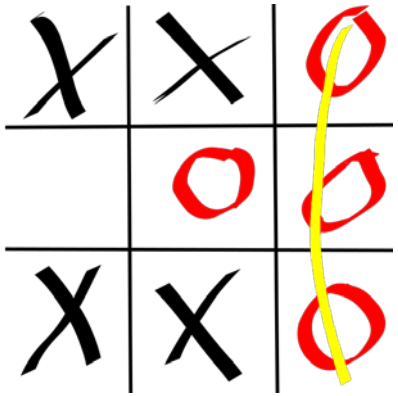




World of TicTacToe?

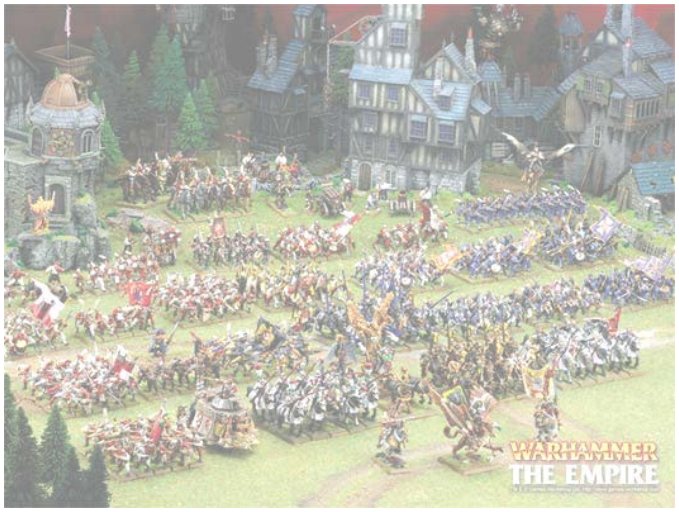
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

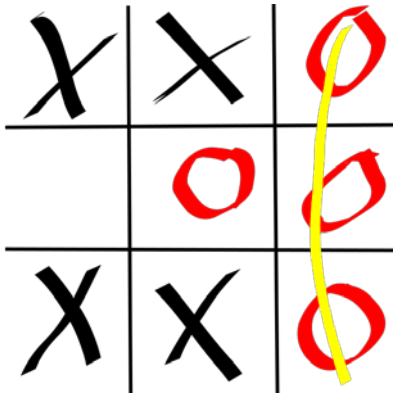




World of TicTacToe?

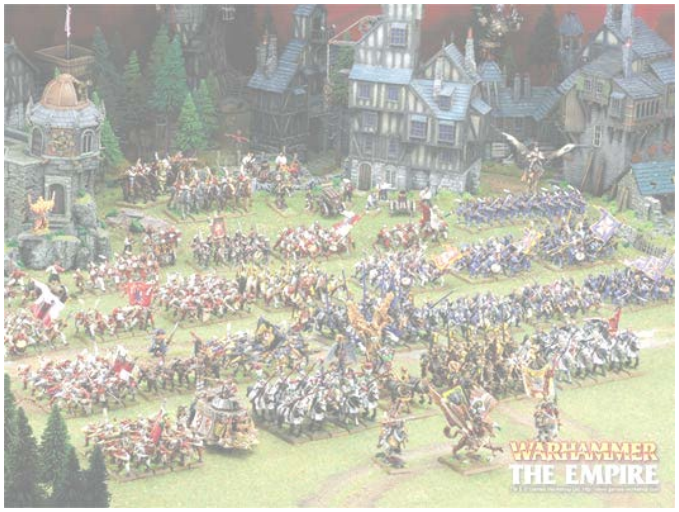
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

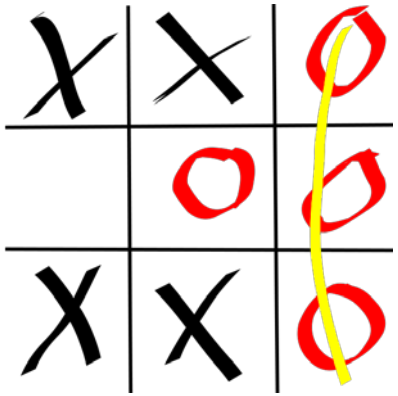




World of TicTacToe?

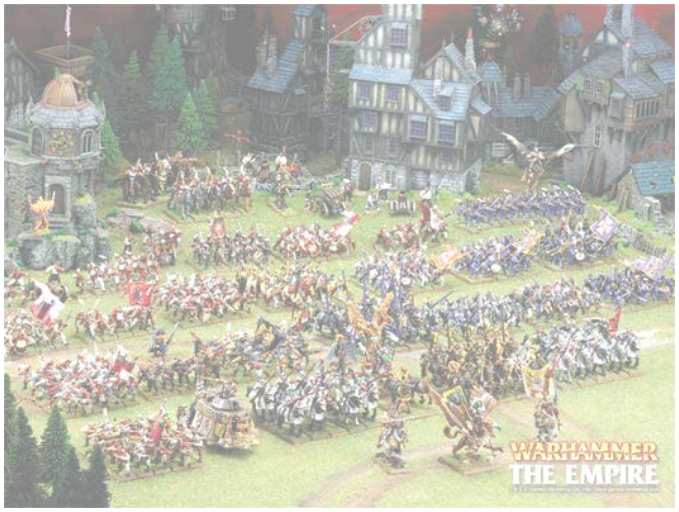
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

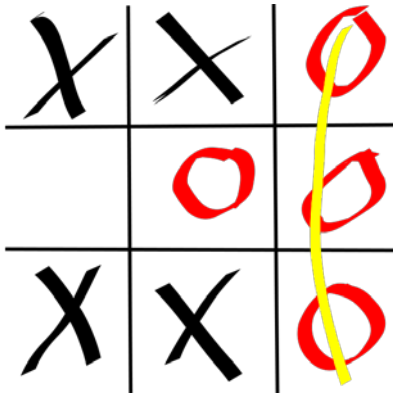




World of TicTacToe?

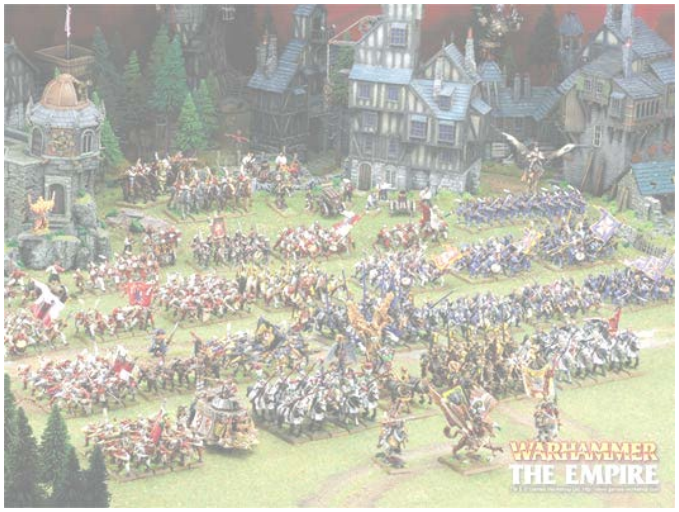
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

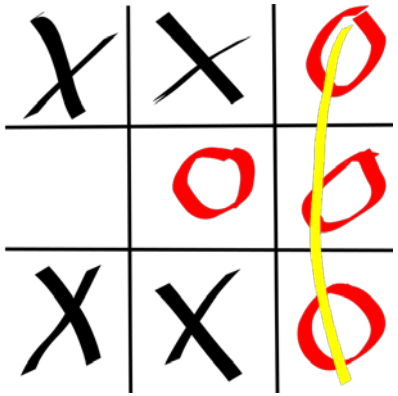




World of TicTacToe?

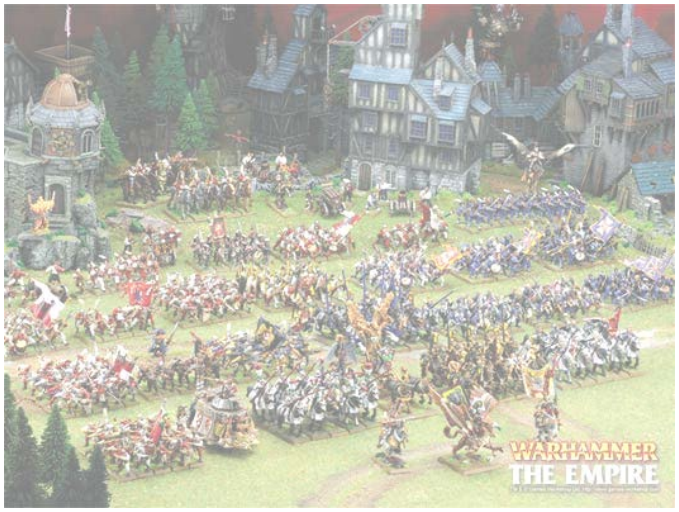
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

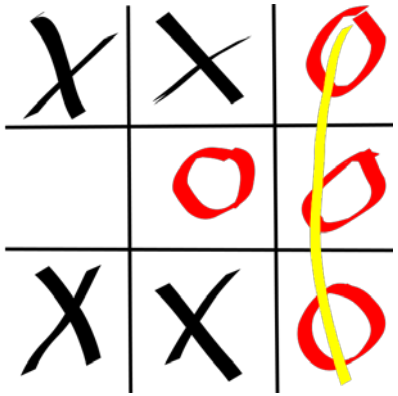




World of TicTacToe?

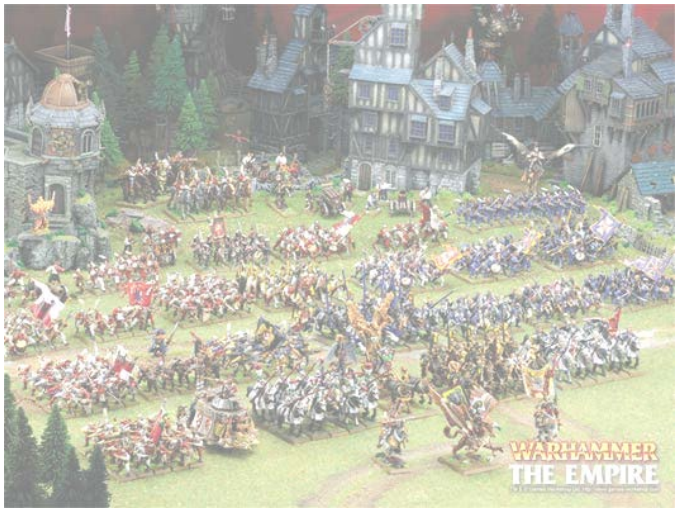
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

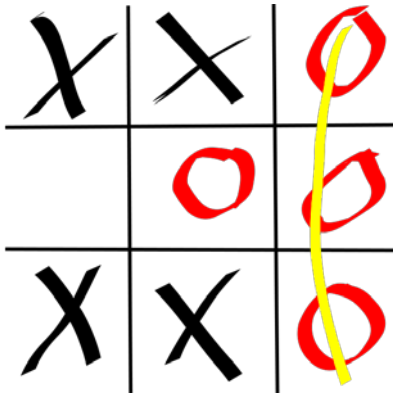




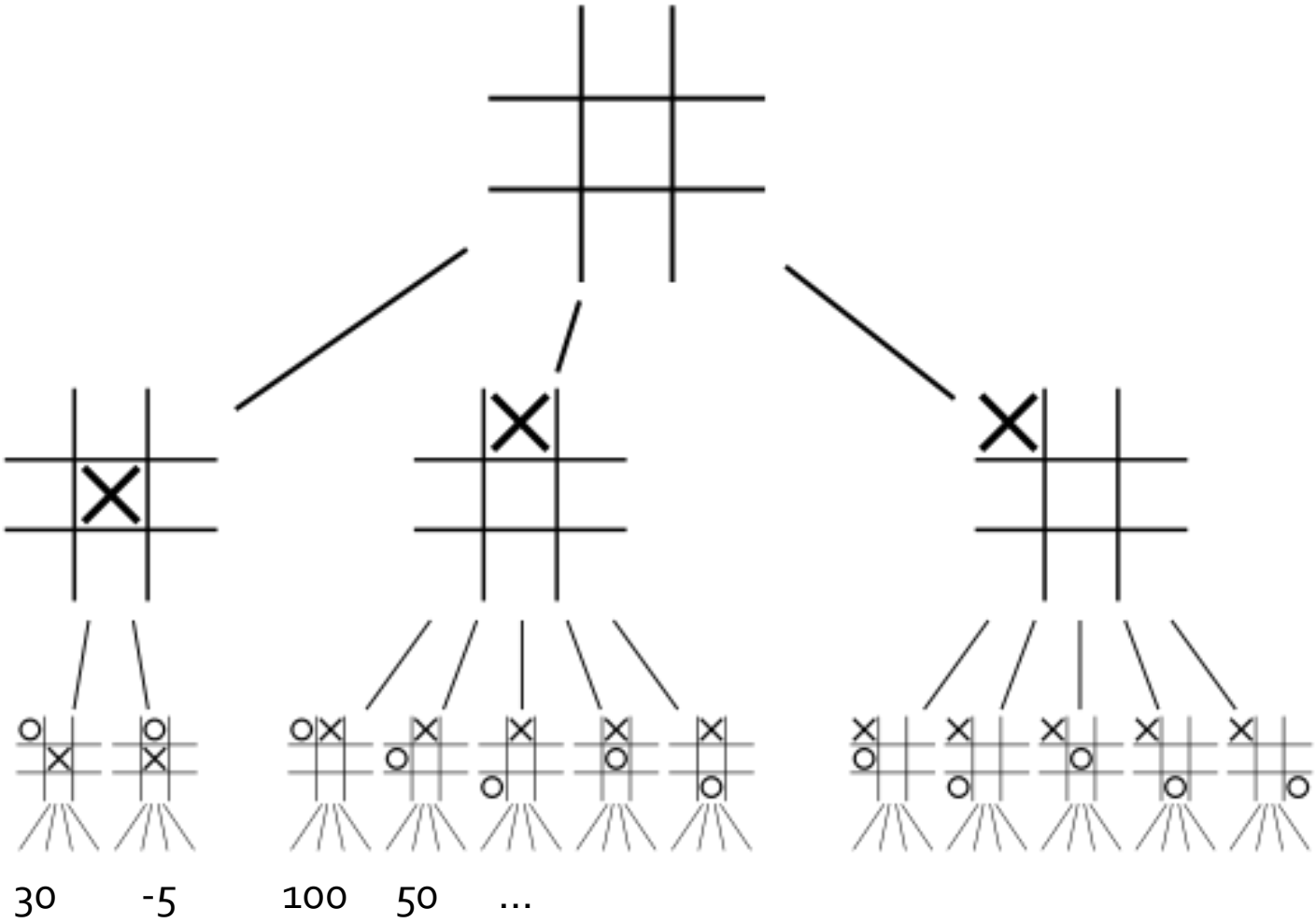
World of TicTacToe!

- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy



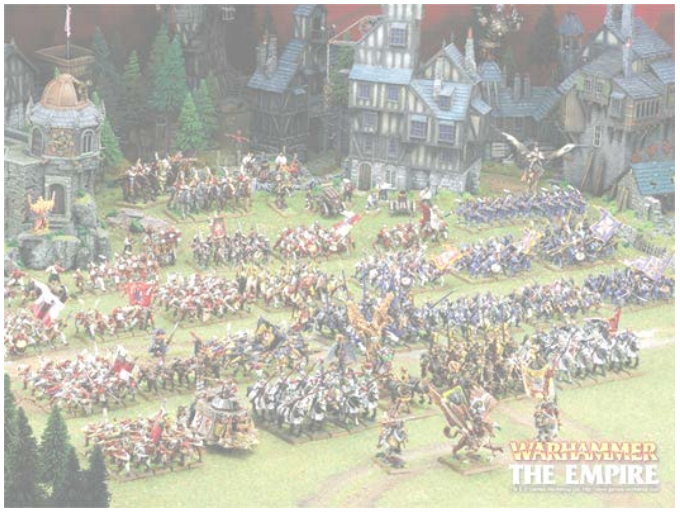


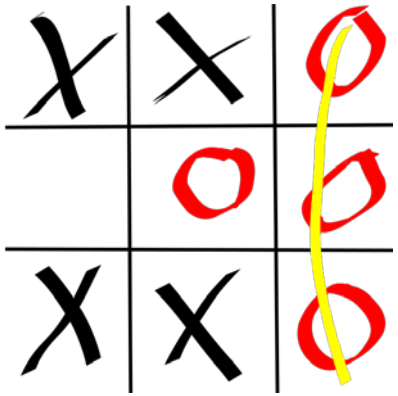
We can search for solution!



World of TicTacToe!
World of Poker?

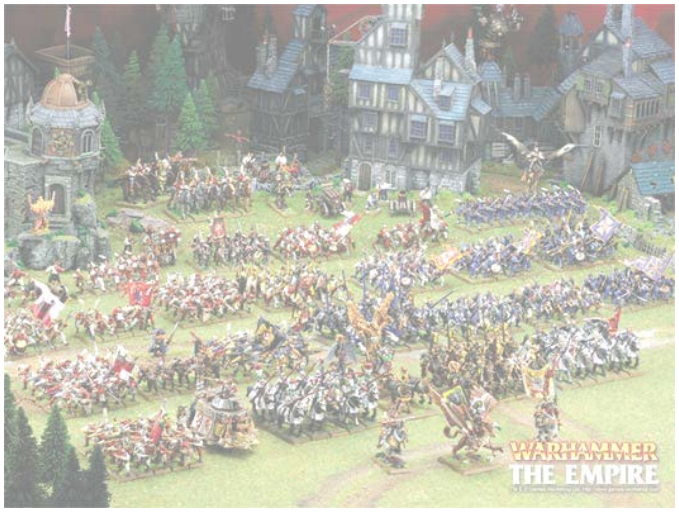
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy





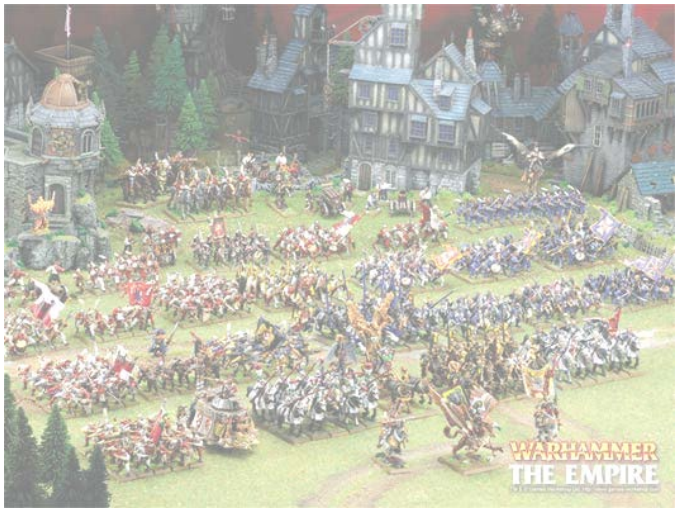
World of Poker!

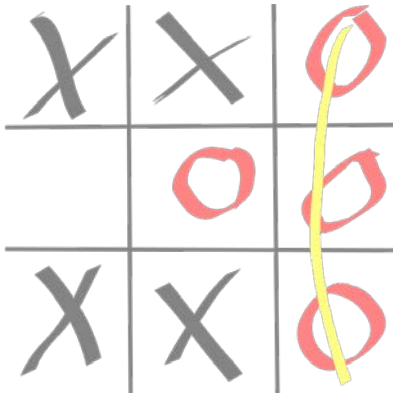
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy



World of UT2004?

- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic
- Discrete *vs.* Continuous
- Known *vs.* Unknown
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy

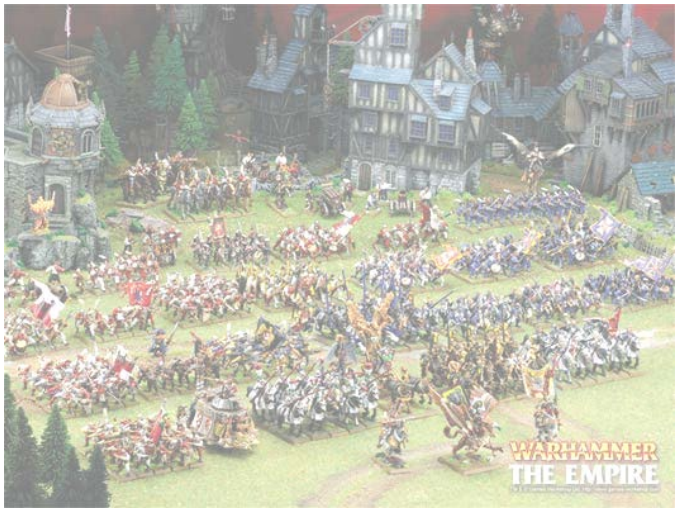


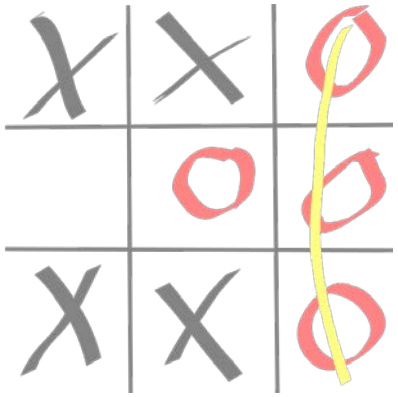


World of UT2004

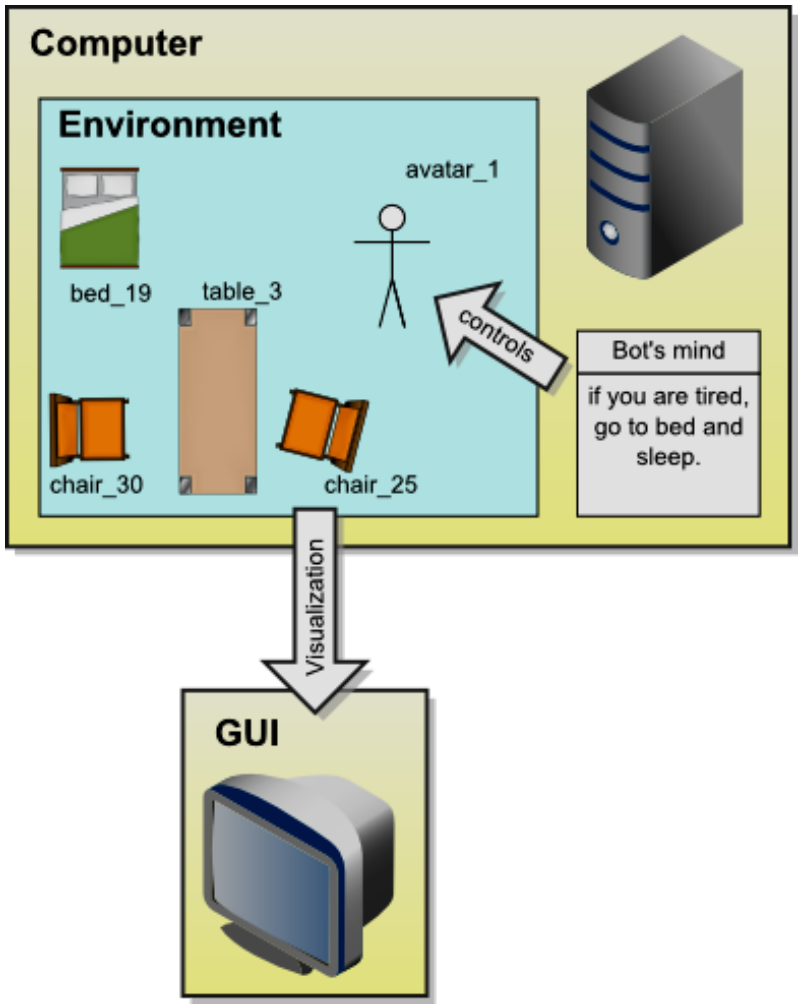
The (almost) worst case imaginable!

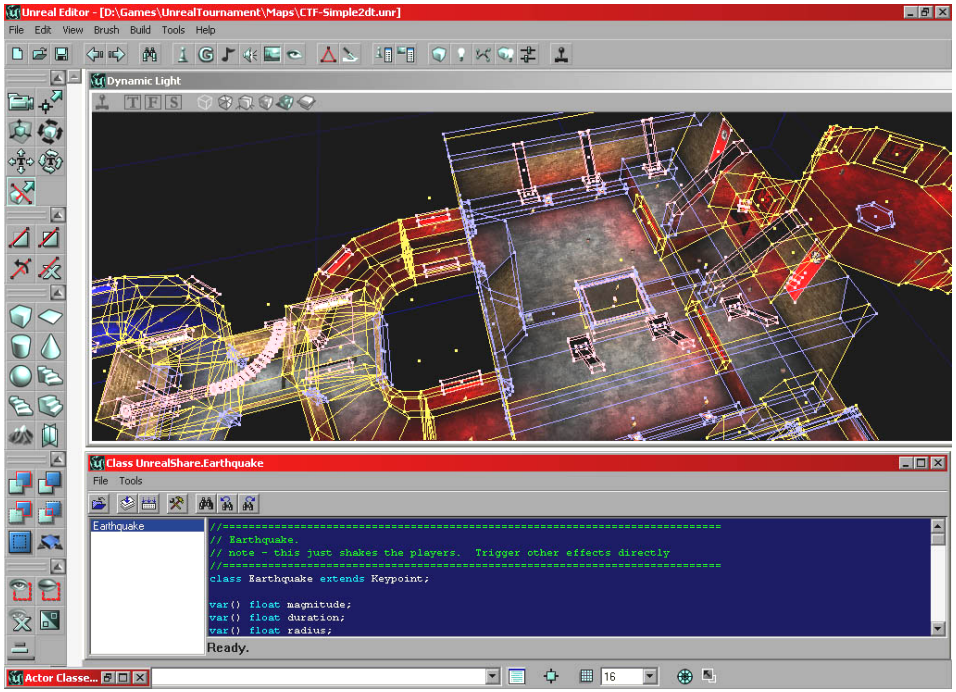
- Fully *vs.* Partially observable
- Episodic *vs.* Sequential
- Static *vs.* Dynamic
- Single *vs.* Multi agent
- Deterministic *vs.* Stochastic (weakly)
- Discrete *vs.* Continuous
- Known *vs.* Unknown (weakly)
- Turn-based *vs.* Real-time
- Noiseless *vs.* Noisy





Virtual worlds





IVAs and Virtual worlds

Environment state (E)

Perception (P)

Action (A)

Memory (S)

1. Part of environment state E is exported to the agent p(E) = P

2. Agent performs action-selection: f(P,S) -> AxS

3. Actions are carried out in the environment: a(A^n,E) -> E

IVAs and Virtual worlds

Environment state (E)

Dynamic world

Non-complete information

Perception (P)

Action (A)

Actions mail fail!

Memory (S)

Inaccurate

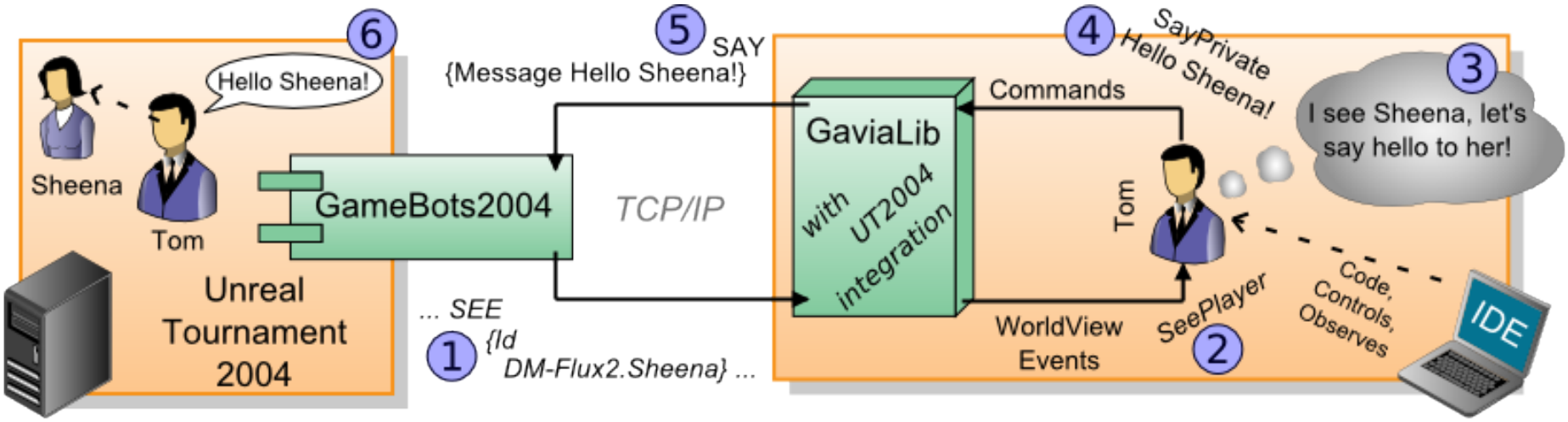
1. Part of environment state E is exported to the agent p(E) = P

2. Agent performs action-selection: f(P,S) -> AxS

3. Actions are carried out in the environment: $a(A^n,E) \rightarrow E$

Pogamut 3 platform

UT2004 and IVAs



UT2004 is providing action execution function *a*.

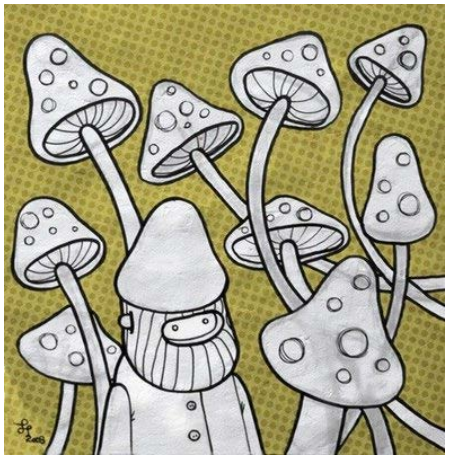
GameBots2004 mediates decisions to UT2004 and implements partial observability function *p*.

Pogamut 3 provides observe function *o*.

You have to supply reason function *r*, decide function *d* and possibly extra memory states S.

Decision Making Systems

- Reactive DMS
- Mushroompicker Cyril



Initial state: not_at_home AND picking_mushrooms

1. **IF** in_front_of_obstacle **THEN** change_rotation
2. **IF** full_basket **AND** picking **THEN** stop_picking
3. **IF** see_mushroom **AND** picking **THEN** put_it_to_basket
4. **IF** noon **AND** picking **THEN** stop_picking
5. **IF** at_home **THEN** end
6. **IF** picking **THEN** random_walk
7. **IF** not_picking **THEN** go_home

Pogamut web

Main web

- http://pogamut.cuni.cz/

JavaDoc (IMPORTANT!)

- http://pogamut.cuni.cz/pogamut_files/latest/_doc/javadoc/

Lecture web

- http://pogamut.cuni.cz/pogamut-devel/doku.php?id=lectures

Tutorials

- http://pogamut.cuni.cz/pogamut_files/latest/ doc/tutorials/

Pogamut manual installation Win32

- http://pogamut.cuni.cz/main/tiki-download_file.php?fileId=22

Pogamut on Linux (external)

- http://cicolink.blogspot.com/2011/11/unreal-tournament-2004-create-bot-with.html

Installation of Pogamut

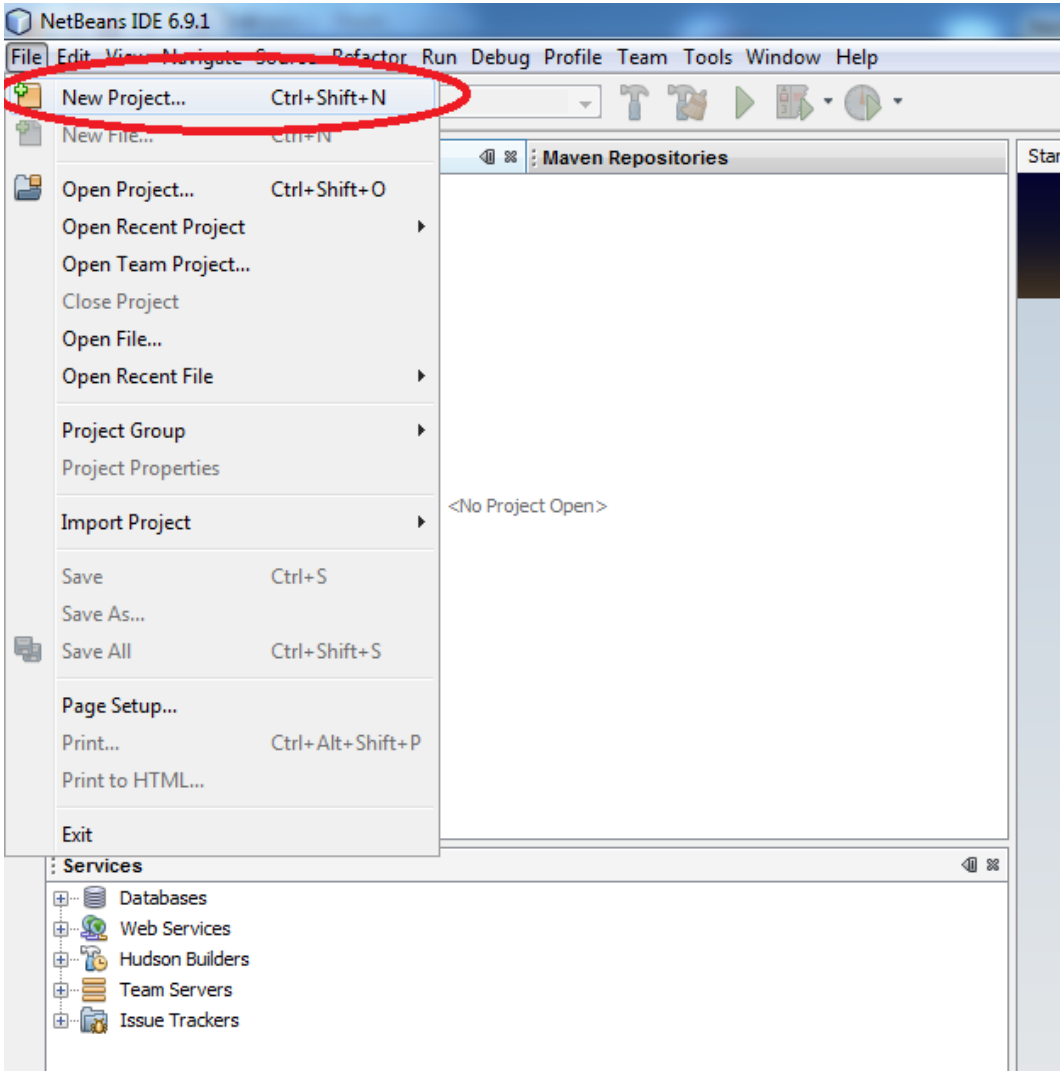
Step 1: Install Pogamut

- Run Pogamut installer found in Download section at
  - **http://pogamut.cuni.cz**

Import bot project

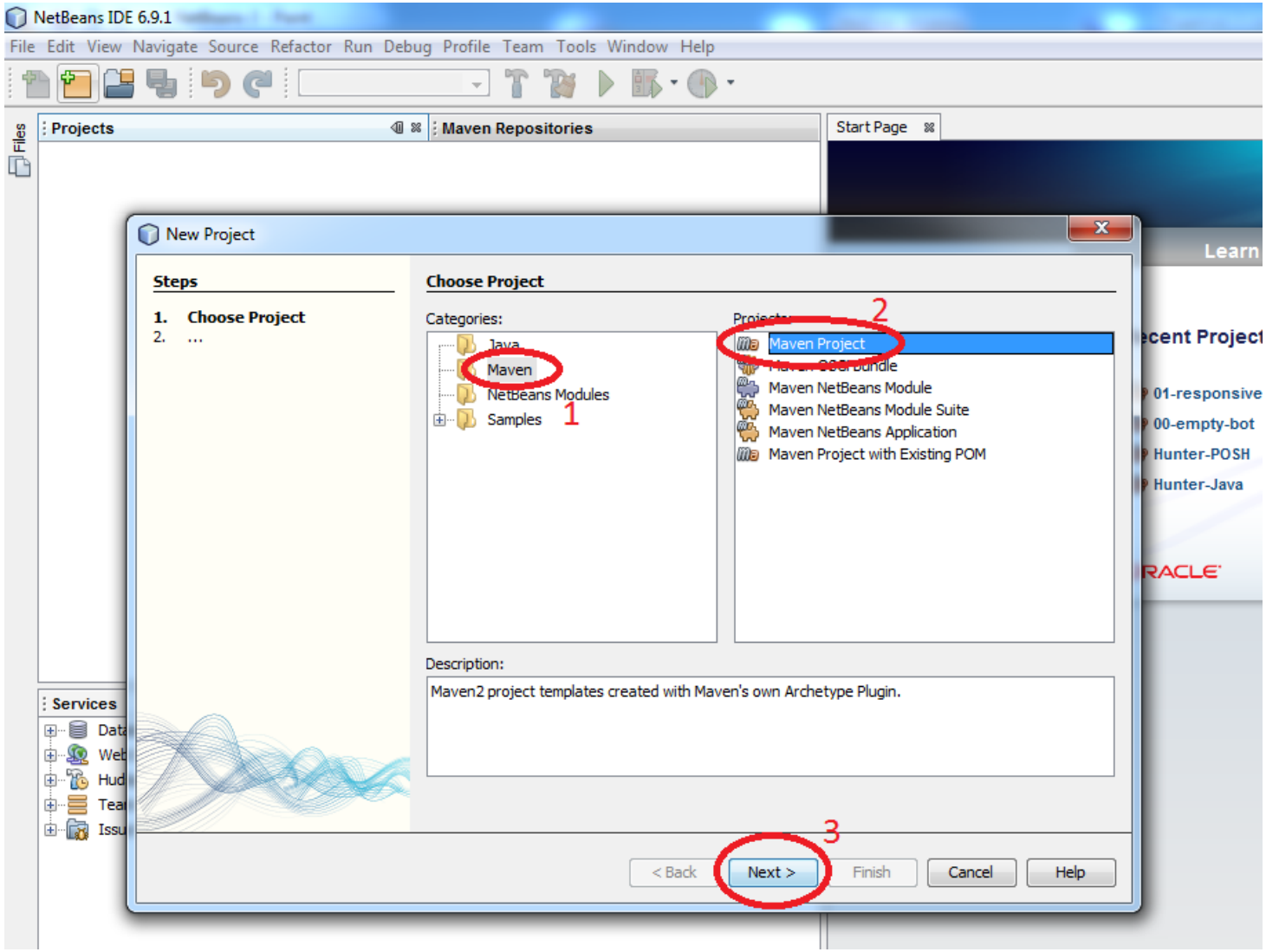
Step 2: Create new bot project

- Create new Maven project…



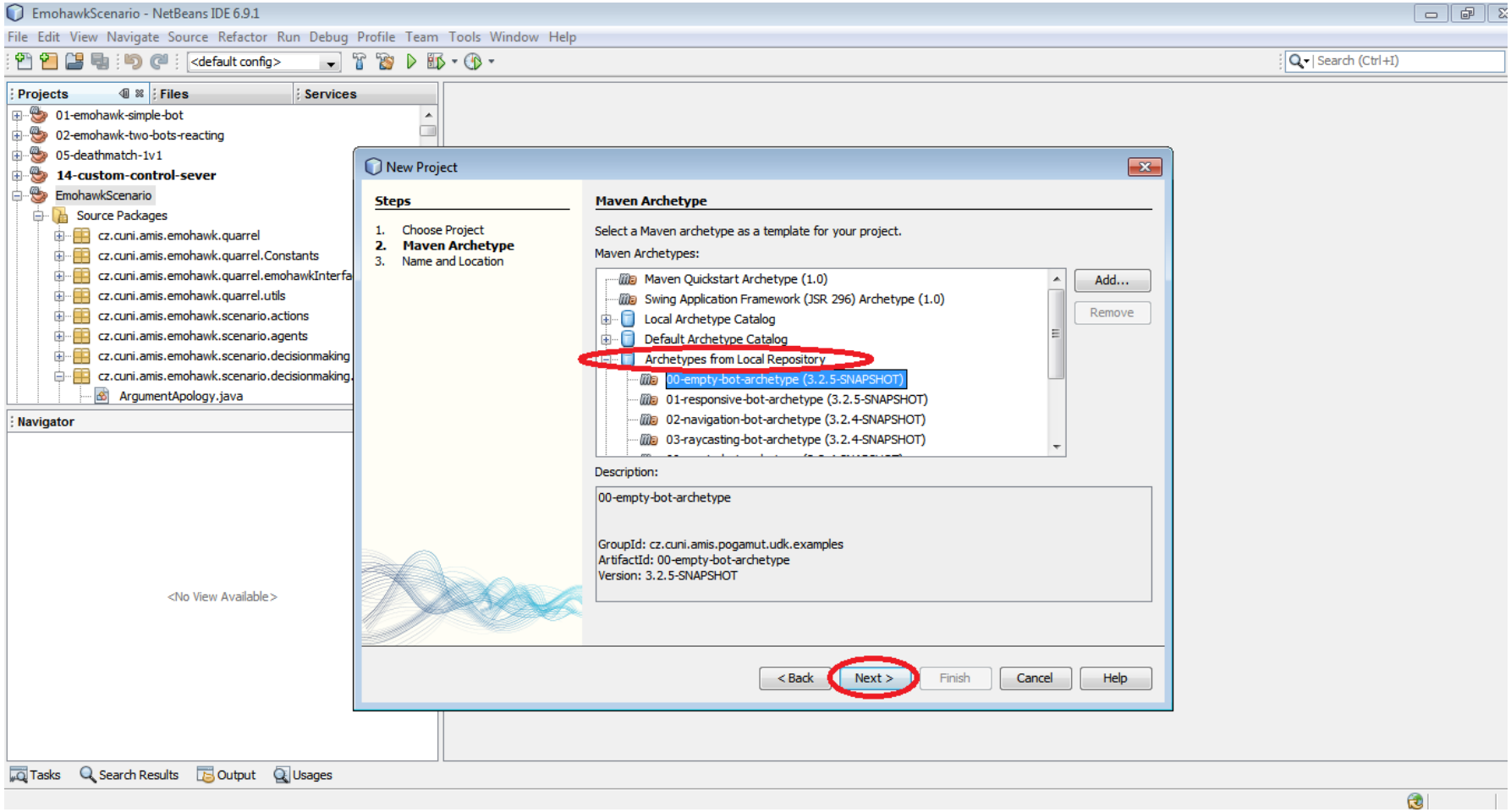
Import bot project

Step 2: Create new bot project



Import bot project

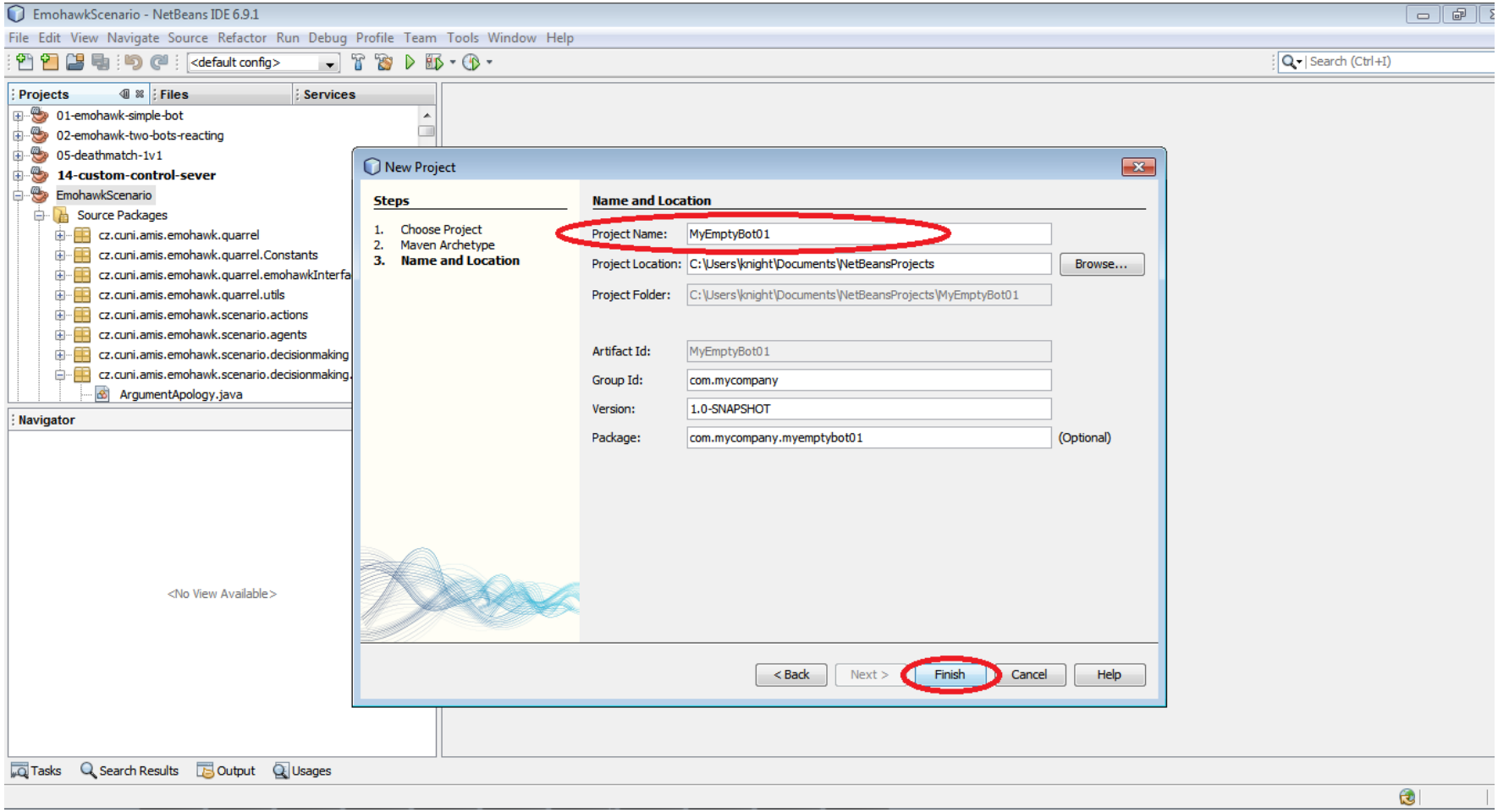
Step 2: Create new bot project



Click Archetypes from Local Repository and select one of the Pogamut archetypes

Import bot project

Step 2: Setup new bot project

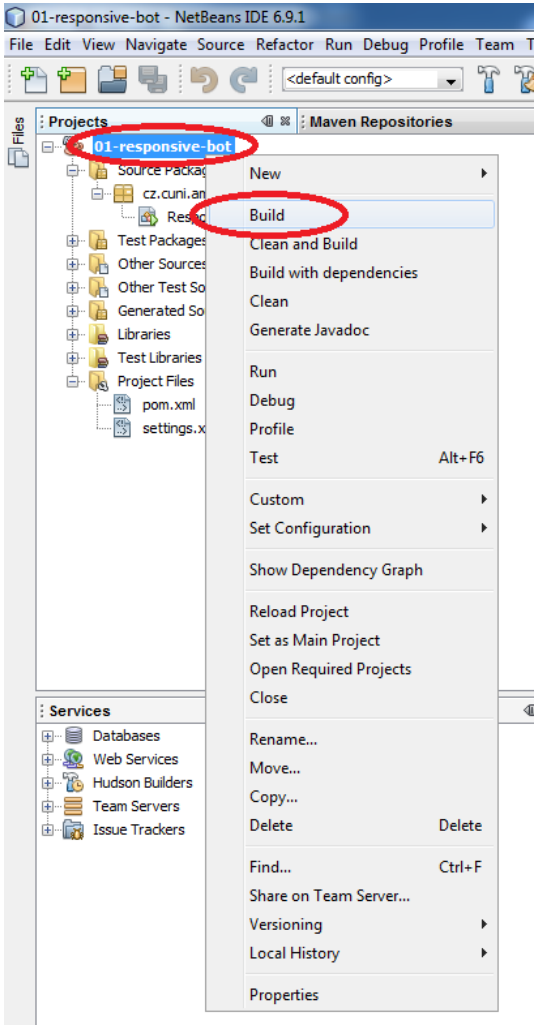


Fill up the name of your project and click Finish

Import bot project

Step 2: Open it!

- And Build it… it should have end with "BUILD SUCCESS"
- Run UnrealEngine2 server
  - UE2/System/startUnrealServer.bat
- And run it!
- Run UnrealEngine2 client and observe bot presence in UT



Tutorial 1 – Empty bot

- Now we have imported first Pogamut example bot!
- See the tutorial:
  - http://pogamut.cuni.cz/pogamut_files/latest/doc/tutorials/EmptyBotTutorial.html

…

- Let's fool around ☺

Assignment 1

Done during the lecture

- Extend EmptyBot to FollowBot:
  - Check whether you can see player
  - If so, run to it directly
  - What if you cannot see anybody?
    - Start turning around to scan your surroundings

Tutorial 2 – Responsive bot

- Second example bot!
- See the tutorial:
  - http://pogamut.cuni.cz/pogamut_files/latest/doc/tutorials/ResponsiveBotTutorial.html

…

- Let's fool around again!

Assignment 2

5 / 100 points

- Extend EmptyBot to:
  - Remember last position of the player and if the player is lost, run to that location
  - If player didn't show up, start turning around to scan your surroundings

Send your assignment to

- Completely zip-up your project(s) folder
- Send it to:
  - Jakub Gemrot (Friday practice lessons)
    - jakub.gemrot@gmail.com
  - Michal Bída (Wednesday practice lessons)
    - michal.bida@gmail.com
- Write us how much time you have spent on setting up the Pogamut platfrom and the assignment respectively!

OPERAČNÍ PROGRAM PRAHA ADAPTABILITA

EVROPSKÁ UNIE

DĚKUJI ZA POZORNOST