Games Engine Design

Course SS 2015

Rüdiger Westermann
Lehrstuhl für Computer Graphik und Visualisierung

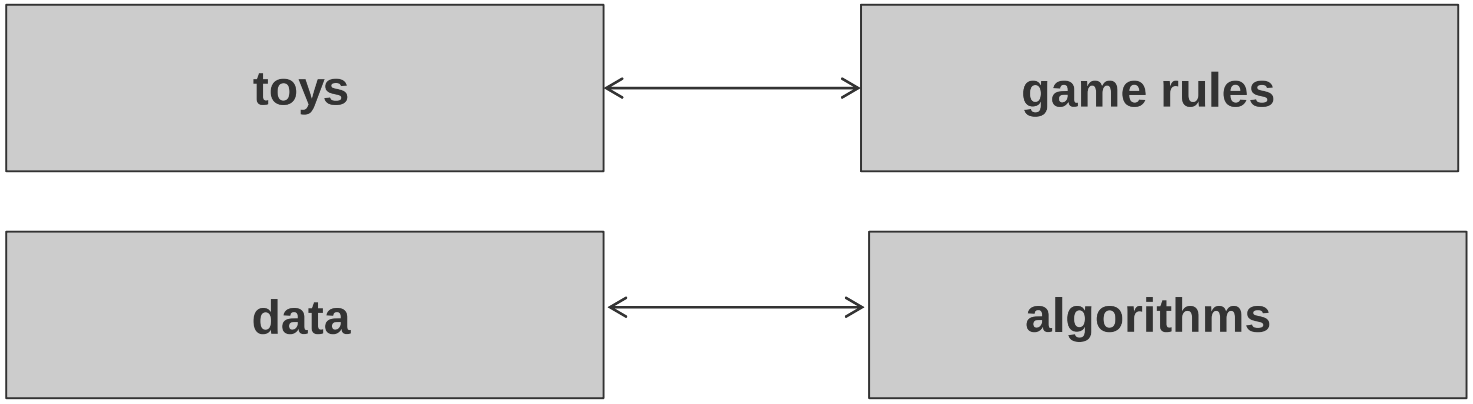
Games development

Differentiation by field of activity

- Game art – aesthetics, graphical design, animation etc.
  - Problem oriented, creative
- Game programming – language, algorithms, engines
  - Problem oriented, systematic
- Game design – specification of the game rules and content
  - Communication oriented, creative
- Game production – management, production cycle, game plan
  - Communication oriented, systematic

Games engine design

- What is a game?
  - Crawford: Closed formal system, rule-based, goal-oriented, interaction, conflict, security; addresses Stone-Age abilities
  - A game is a mixture of game rules and toys you are playing with



Games engine design

- Distinguish between game logic, data and engine
  - Game logic: algorithms which realize the game rules
  - Data: different kinds of data used by the engine
  - Game engine: the other algorithms, „independent“ of the game logic and data

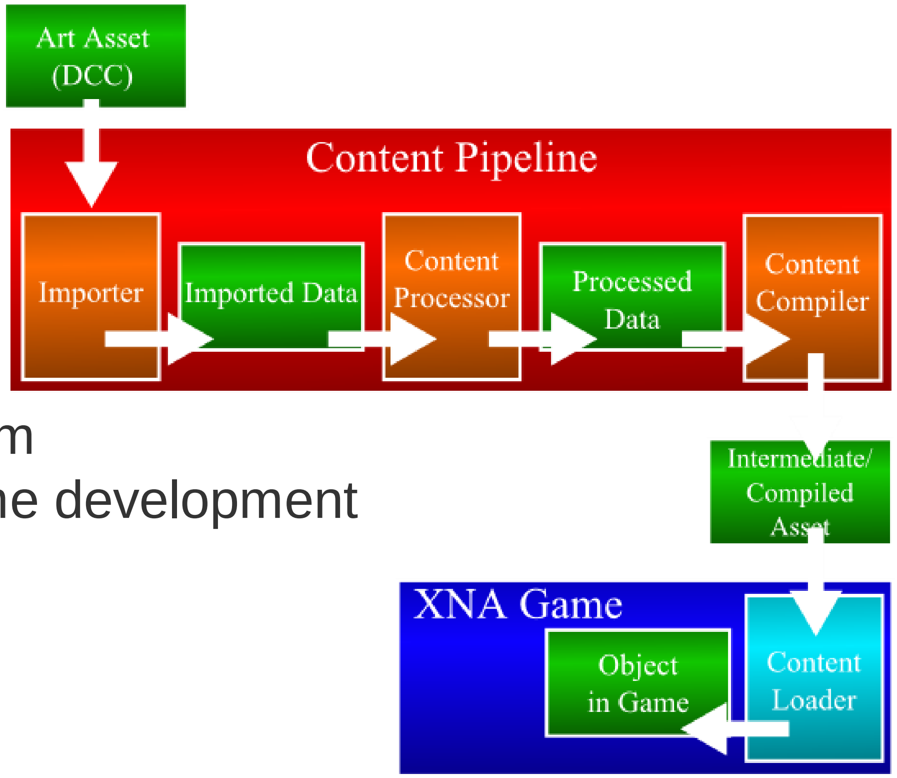
- Game engine should be re-usable, should allow fast prototyping, should make it easy to change the game later on;
  should allow for different perspectives, i.e., player vs. designer

Games engine design

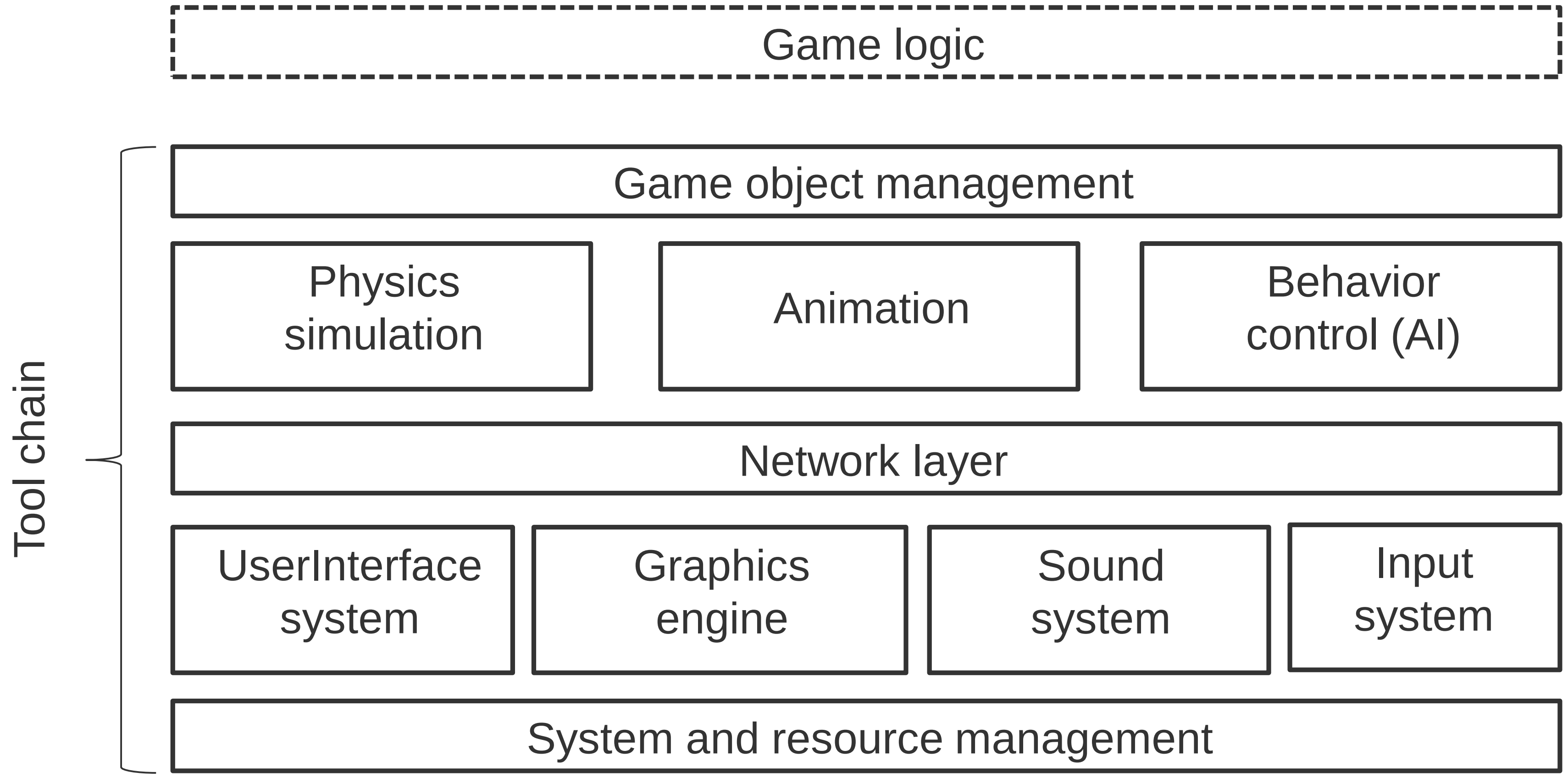
- What are the data an engine is confronted with
  - Media assets: geometric models, textures, animations, sounds
  - Level data: which things are where in a level, how can they move in a level
  - Object configuration data: physical properties and behavior control data
  - User interface configuration data: which input devices; their mapping to the game
  - Engine configuration data: which drivers, which resolution
- Always try to separate data from the code!
- Always try to avoid duplication – use inheritance!

Games engine design

- Asset editing - Photoshop, StudioMax, XNA-Studio etc. (Digital Content Creation)
- Asset compilation
  - From raw data to a proprietary binary format
  - Example XNA game studio: software development system and tool box to facilitate game development



Engine components



Engine components

- Game objects: interface, static vs. dynamic, components, properties, instantiation, life time
- Network layer: not really separable; tasks like synchronization, replication, consistency checks
- System/resources: math utilities, memory management, multi-threading, streaming, generic data structures, iterators