Lecture Notes

Managing and Mining Multiplayer Online Games

for the Summer Semester 2019

Chapter 1: Computer Games

http://www.dbs.ifi.lmu.de/cms/VO_Managing_Massive_Multiplayer_Online_Games

Organisation

- ***Dates***
  - **Lecture**: Tue, 13:00 – 16:00 , room M 105 (Main Building)
  - **Tutorials**:
    - Group 1: Wed, 14:00 – 16:00, Room D Z003 (Main Building)
    - Group 2: Wed, 16:00 – 18:00, Room A 120 (Main Building)

- ***Homepage***:

  http://www.dbs.ifi.lmu.de/cms/VO_Managing_Massive_Multiplayer_Online_Games

- ***Tutorials***: Sebastian Schmoll
- ***Exams***: Dates will be announced on the homepage and in UNIWORX. (Please register to the lecture and the exam)

  Exam material is presented at the lecture and during tutorials. (The lecture notes are merely a learning aid)

What are Computer Games?

Definition:

„An interactive experience that provides the players with an increasingly challenging sequence of patterns which he or she learns and eventually masters." [Ralph Koster: A Theory of Fun for Game Design, Phoenix, AZ, Paraglyph 2004]

⇒ *Interactive*: Interaction with the computer or other players

⇒ *Experience*: Perceiving and experiencing content are important

⇒ *Learning and Mastering*:

A good game poses a challenge but allows a learning curve leading to mastery.

Why study Games?

- Important branch of the media industry
- Computer games develop into popular sports (tournaments, live streams, streaming platforms..)
- Second level products, add ons, analytics platforms, ingame goods,..
- Sandbox of research in many areas
  - social analytics
  - economics
  - simulation
  - artificial intelligence

Related Areas

- *Real-Time Simulations*: Modeling a real physical environment and depicting its dynamic development. Traffic simulations or simulating swarm behavior.

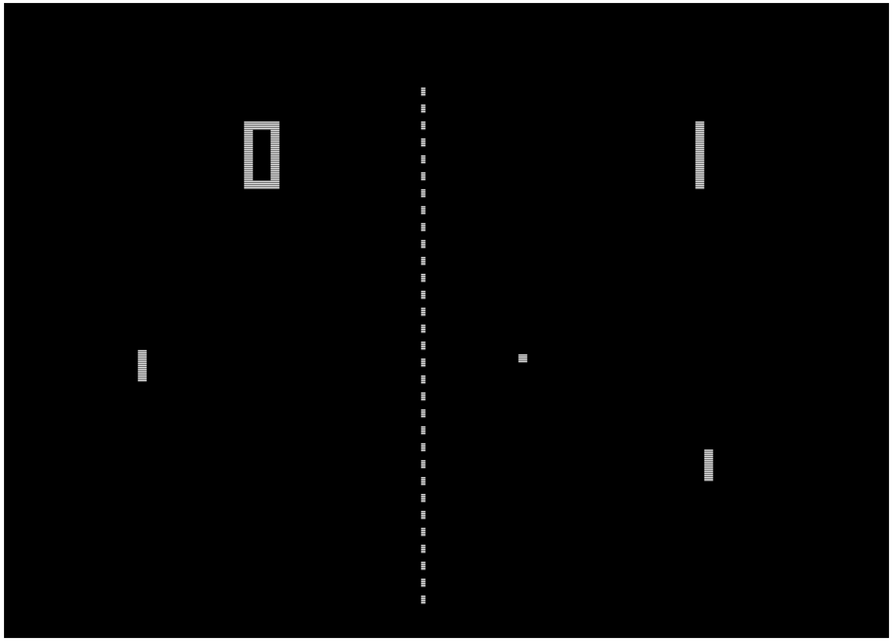
- *Geographic Informations Systems*: Managing spatial information (2D/3D-Maps) and simulating moving objects, e.g. cars.

- *Agent Based Model*: Simulating the actions and interactions of autonomous agents with a view to assessing their effects on the system as a whole.

- *Virtual Reality*: Rendering sound and visuals of a 3D enviroment.

The obligatory PONG slide

PONG (Atari 1972) was the first commercially successful computer game



- PONG is a simple tennis game (game of skill)
- The controllers were custom made
- Has been copied and modified numerous times over the following years

Social Aspects of Computer Games

- 58% of Americans play Video games
- The average Gamer is 30 years old and has been playing for 13 years.
- The average Buyer is 35 years old.
- 45% of gamers are women. Woman age 18+ are more numerous among gamers than boys younger than 18. (*31% vs. 19% respectively.*)
- In 2013 38% of Americans age 36+ played Video games.
- A conclusive Connection between Video Games and propensity towards violence has not been shown yet.
- Several studies exist, that show a negative correlation between distribution of Video Games and the number of violent crimes.

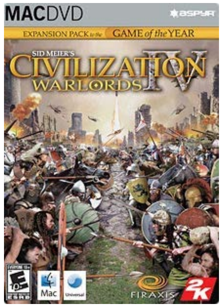
Source : http://www.theesa.com/facts/pdfs/esa_ef_2013.pdf

Business Models

Boxed Games

- the customer purchase the game with a single payment
- sold via shops or online
- suitable for offline games
- high initial risk for the customer/ no or limited possibility for a refund if the customer is disappointed
- danger of pirate copies
- marketing and advertisements are crucial for success

examples: Warcraft, Call of Duty, Resident Evil, Gran Tourismo,
Grand Theft Auto, Civilization, …

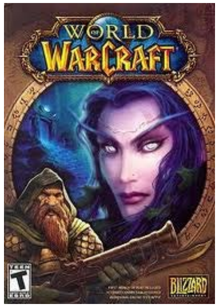


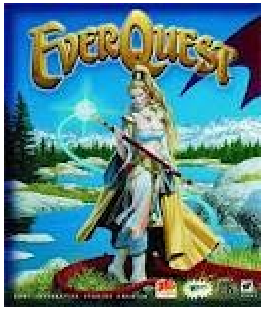
Business Models

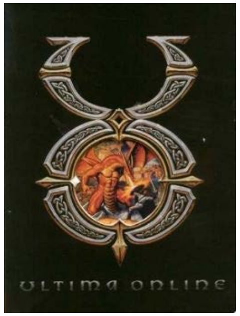
Subscription Games

- the customer has to pay a regular fee for the account
- suitable only for online games
- requires account management
- new forms of criminal abuse (e.g. account-napping)
- guarantees regular income for the game provider
- content updates and customer satisfaction are important factors

examples: Ultima Online, World of Warcraft, Everquest, …





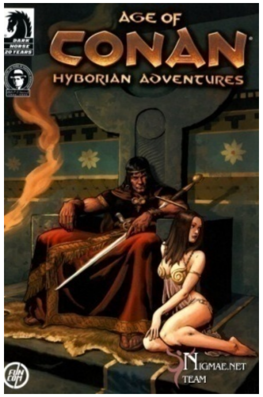


Business Models

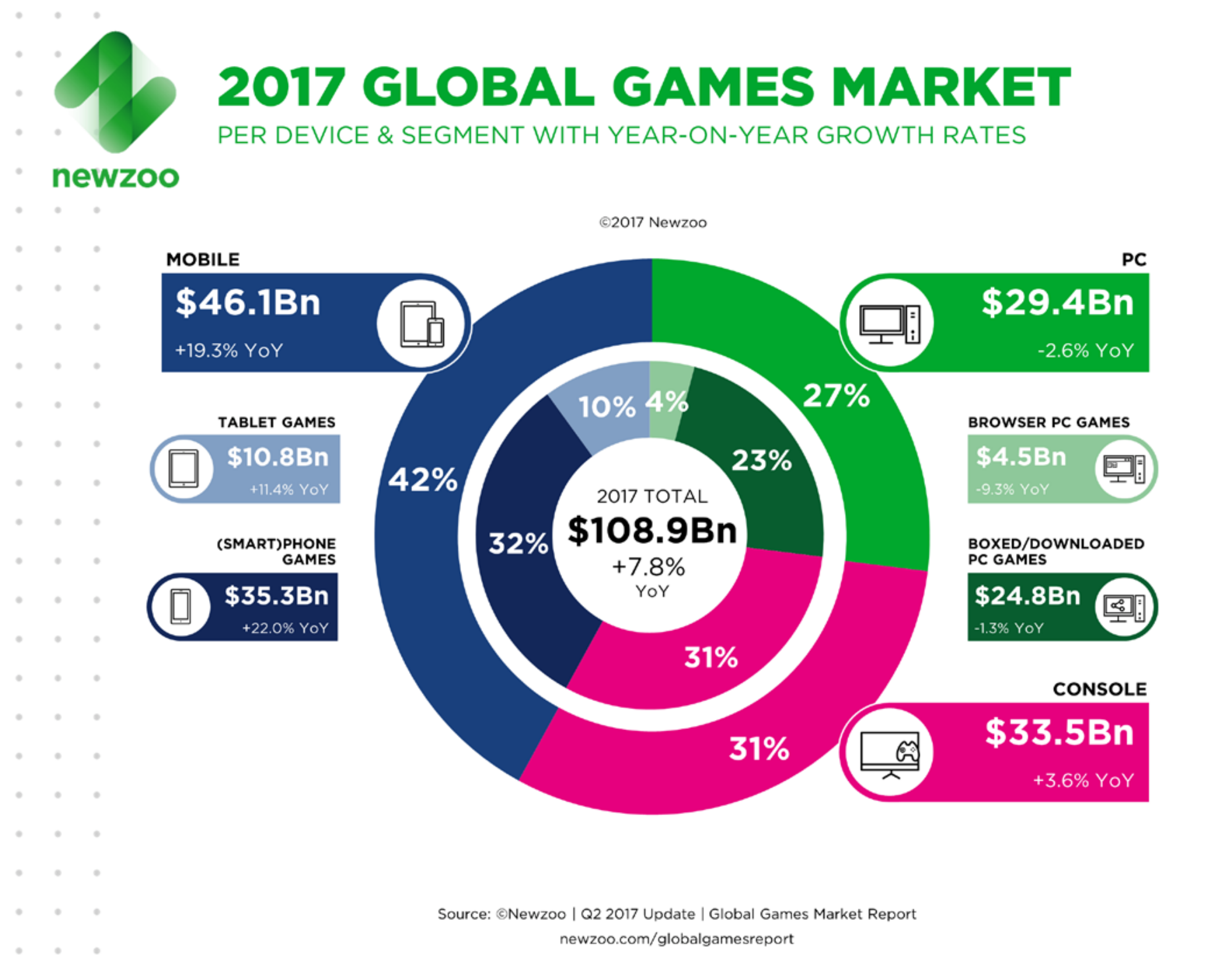
Micro-Transaction Games

- sale of virtual goods : currency, items or abilities
- suitable for free-to-play games
- biggest risk for the developer but very high margin
- lowest entry threshold for customers and low initial risk (players only invest money if they like the game.)
- contains the most threats for fraud: resale of online goods, ingame currency can be exchanges for cash, botting..

Examples: Age of Conan, Herr der Ringe online, Farmville, Travianer, ..



Global Game Market 2017



2017 GLOBAL GAMES MARKET

PER DEVICE & SEGMENT WITH YEAR-ON-YEAR GROWTH RATES

Source: ©Newzoo | Q2 2017 Update | Global Games Market Report
newzoo.com/globalgamesreport

In 2017, mobile games will generate

$46.1Bn

or **42%** of the global market.

newzoo

https://newzoo.com/insights/articles/the-global-games-market-will-reach-108-9-billion-in-2017-with-mobile-taking-42/

E-Sports

Computer games as competitive sports

- 71 MM Viewers in 2014
- 36 MM Viewers for the League of Legends (LoL) World Championship 2015 (NBA finals: 23,3 MM)
- 27 MM active players every day
- 18 MM USD Price Money for "The International" 2015, (world biggest DOTA2 tournament)
- ca. 6th most import sport in professional betting

Success Factors of Computer Games

- appealing experience, challenging but with potential for continuous improvement (good game design)
- quality of service: availability, customer services,…
- sufficient numbers of players: the more players a game has, the more appealing it is
- time spent: The more time players spend with a game,
  the more successful it is:
  - social prestige of a game raises continually to a certain point
  - more sales, longer subscriptions, more transactions
- there is another side to success:
  - addictive: a tremendous blow to social prestige
  - big community => it takes less time for guides to get published
  - organized fraud concentrates on successful titles

=> long term control over game design and game behavior is very important for game providers

Distinction of Games

- *number of controlled game entities*: Unit, Group, Faction
- *perspective*: First-Person, Third Person, Top-Down
- *temporal organistation*: Real-Time, Game-Time, turn based
- control complexity: simulation vs. arcade, manual aim vs. auto-target, …
- *graphics and gaming world*: 2D, 3D, grids vs. free Movement
- *number of players*: single player, multiplayer (1-100), massive multiplayer (100-100.000)
- *avatar development*: virtual abilities, player abilities (RPG vs. Chess)
- *influence of randomization*: Game of Dice vs. Chess

Classic Game Genres

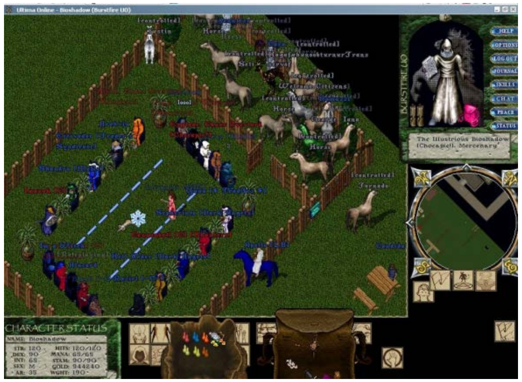
Real-Time Strategy Games (RTS)

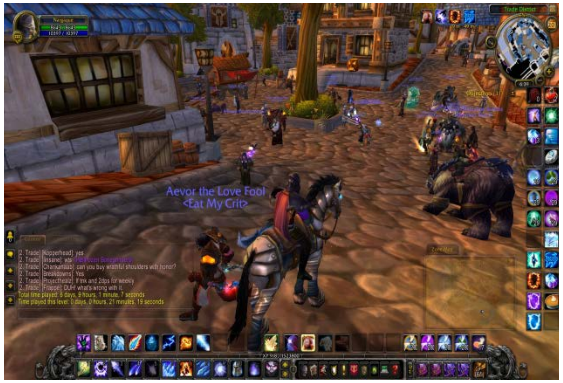
- games that focus on creating an army and defeating enemies
- control many units, top down perspective, low granularity of control, real-time, 2D-/3D world, number of Players: 1-10
- *examples* : Dune II, Starcraft, Warcraft, Command & Conquer, Age of Empires, ...

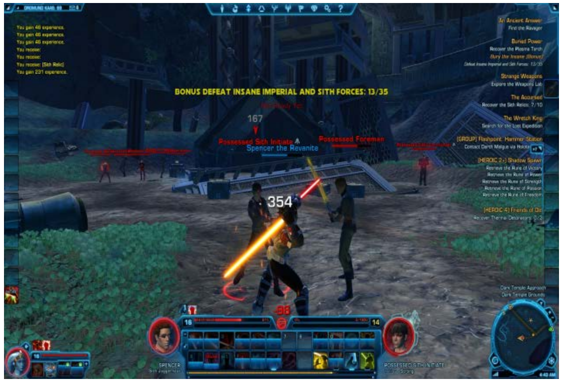
Classic Game Genres

Massive Multiplayer Online Role Playing Games (MMORPGs)

- development and story of one game character
- one unit, 3rd person, mediocre granularity of Control, real-time, 3D-world, large numbers of players (several thousand), core element is character development
- *examples*: Ultima Online, Everquest, World of Warcraft, Star Wars: The Old Republic, Age of Conan, Lorf of the Rings Online, …







Classic Game Genres

Multiplayer Online Battle Arena Games(MOBAs)

- team PVP Game (often 5 vs. 5)
- mixture of RTS and MMORPG/ popular E-Sport
- one unit per player, 3rd person perspective, medium granularity of control, real-time, unit development, team play,..
- *goal*: mostly destroy the enemy base
- *examples*: League of Legends(LoL), Dota, Dota2, Heroes of the Storm,..

Classic Game Genres

First-Person-Shooter (FPS)

- simulates fire fights
- one unit, first person perspective, real-time, high control granularity, 3D world, usually 1-20 Players
- *examples* : Wolfenstein 3D, Doom, Unreal, Half-Life, Team Fortress, Halo, Counter Strike, …

Classic Game Genres

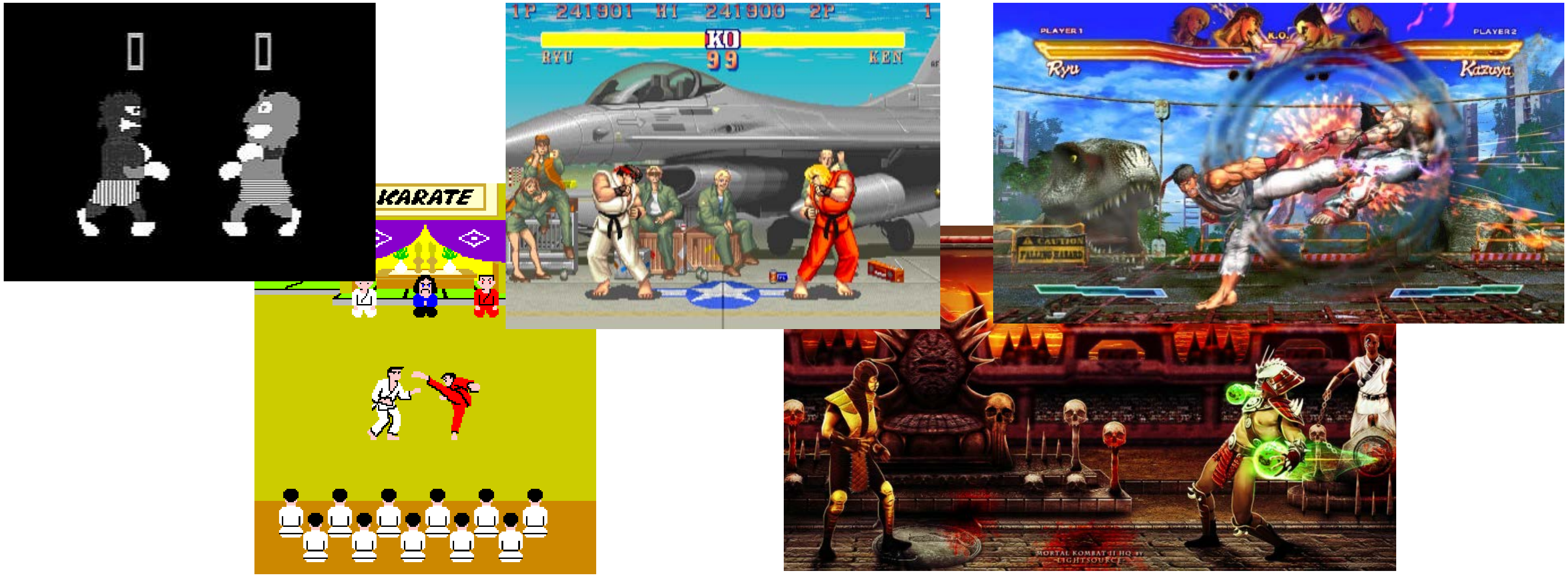
Racing Games

- controlling a vehicles
- one unit, 1st or 3rd person perspective, real-time, between medium and very high control granularity, 2D-/3D world, often 1-2 players but multiplayer is also possible
- *examples*: Atari's Space Race (1970), Test Drive, Need for Speed, Outrun, Gran Tourismo, …

Classic Game Genres

Fighting Games

- game content is about (melee fighting)
- one unit, 3rd person perspective, medium control granularity, real-time, 2D/3D world, multiplayer game
- *examples*: Heavyweight Champ (1973), Karate Champ (1984), Street Fighter, Tekken, Mortal Combat, …



Classic Game Genres

Economic Simulations or Turn based Strategy games

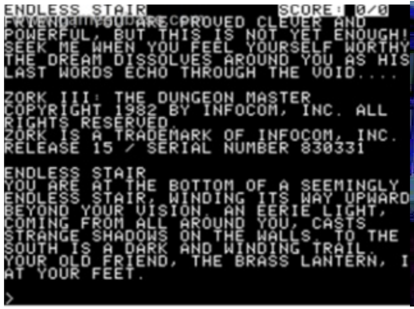
- game content focuses on managing a city, a nation, a corporation or other group
- may change to strategy game
- many units, top-down, real-time or turn-based, low control granularity, 2D world, single up to massive multiplayer
- *examples*: Intopia (1963), M.U.L.E (1983), Civilization, The Settlers, Sim City, Travianer, ...

THE SHIP WILL BE BACK IN 12 MONTHS.

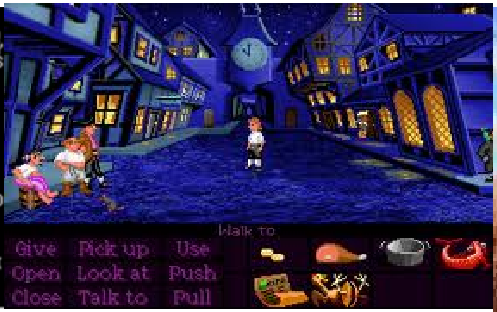
Classic Game Genres

Adventure Games and Interactive Movies

- games that include an interactive story
- interaction: solving puzzles and collecting items
- 1st and 3rd person, variable control granularity
- often transition to other genres
- *examples*: Zorc, Monkey Island, Leisure Suit Larry, Deponia, Firewatch, Telltale Game Series, Until Dawn

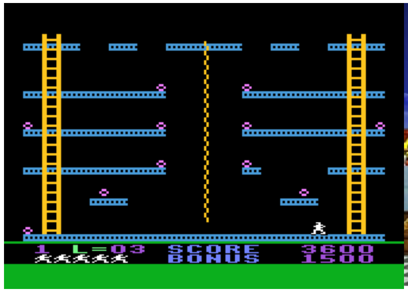


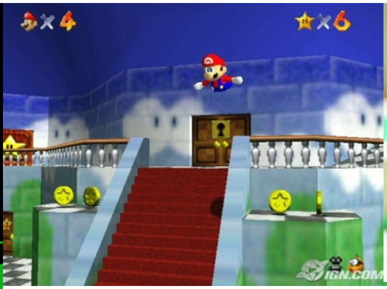
```
ENDLESS STAIR                 SCORE: 0/0
FRIEND, YOU ARE PROVED CLEVER AND
POWERFUL, BUT THIS IS NOT YET ENOUGH!
SEEK ME WHEN YOU FEEL YOURSELF WORTHY
THE DREAM DISSOLVES AROUND YOU AS HIS
LAST WORDS ECHO THROUGH THE VOID....

ZORK III: THE DUNGEON MASTER
COPYRIGHT 1982 BY INFOCOM, INC. ALL
RIGHTS RESERVED.
ZORK IS A TRADEMARK OF INFOCOM, INC.
RELEASE 15 / SERIAL NUMBER 830331

ENDLESS STAIR
YOU ARE AT THE BOTTOM OF A SEEMINGLY
ENDLESS STAIR, WINDING ITS WAY UPWARD
BEYOND YOUR VISION. AN EERIE LIGHT,
COMING FROM ALL AROUND YOU, CASTS
STRANGE SHADOWS ON THE WALLS. TO THE
SOUTH IS A DARK AND WINDING TRAIL.
YOUR OLD FRIEND, THE BRASS LANTERN, I
AT YOUR FEET.

>
```

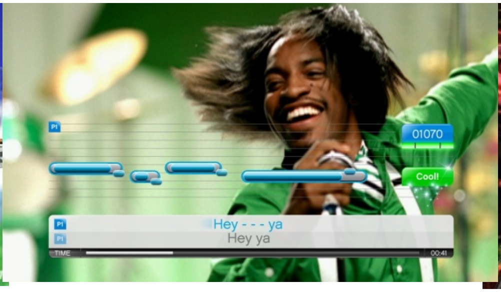


More Genres

- ***Jump 'n' Run***: 3rd person control of a character in a 2D/3D environment (e.g. Super Mario, Jumpman, ...)
- ***Singing, Music, Rhythm Games***: input matching a music title are evaluated with a computer (e.g. Sing Star, Rock-Band, Just Dance series
- ...







There is a wide variety of other games, that cannot be categorized into a single category,

- combinations of elements from several genres
- gaming experience outside of any genre

Structure of Computer Games

- Modern computer games are often multi-layered complex software systems, developed by a big team.
- Computer games vary in complexity and scope depending on their structure and attributes.
- Simple game principles are experiencing a renaissance, thanks to new platforms (Android, iPhone/iPad, Browser and Social Networks).
- Even simple game principles cause more effort when extended to an MMOG, due to scaling and the necessity to analyze player behavior.

Roles in the Development Team

- ***Software Engineers***: development of game engine/game core and of tools for artists and game designers
- ***Artists***: responsible for content: concept, 3D models, sound, animation, textures, lighting, animation and motion capture, actors, ...
- ***Game Designers***: responsible for gameplay (interaction) and the gaming experience: story, level design, game goals, enemy placement, equipment, virtual abilities, ...
- ***Producers***: senior game designer, project management
- ***Other***: marketing, team assistance, maintenance, ...
- ***Publishers and Studios***: marketing, fabrication, and distribution of the game

Structure of Computer Games

Factors speaking against a standardized architecture:

- focus is on content
- varying requirements for different games
- performance requirements
  (less required resources mean more potential players)
- hardware-resources are subject to constant change

Consequently: Game Architectures are more likely to be integrated and specific than modular and reusable.

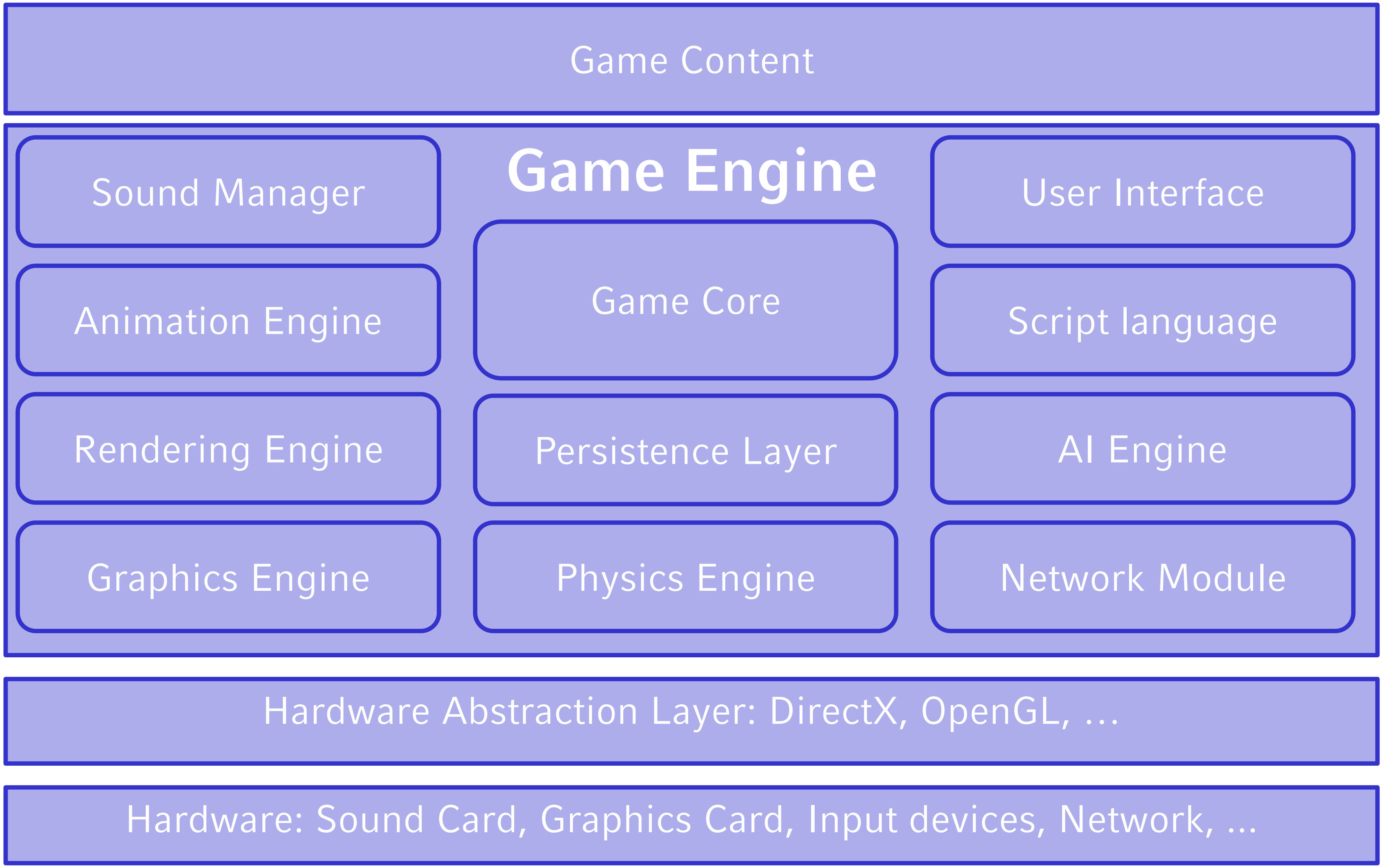
But: Games are complex software system

⇒ Use of standard components (*Engines*)

⇒ Layered architecture and modular build

⇒ Reusability of code for other games (*Mods*)

Building Blocks in Game Architecture



Game Content

Game Engine

Sound Manager

Animation Engine

Rendering Engine

Graphics Engine

Game Core

Persistence Layer

Physics Engine

User Interface

Script language

AI Engine

Network Module

Hardware Abstraction Layer: DirectX, OpenGL, …

Hardware: Sound Card, Graphics Card, Input devices, Network, ...

Hardware Layer and Hardware Abstraction Layer

- Hardware components
  - graphics controller and GPUs
  - sound processors
  - input devices (keyboard, mouse, joysticks, gamepads, steering wheels, microphone, camera,...)
  - main memory and CPU
  - SSDs and hard drives (secondary storage)

- Device Drivers:
  - low I/O between hardware and the operating system
  - often device specific

Hardware Abstraction Layer

- encapsulates hardware functionalities for different hardware configurations (PC, Android, ...)
- offers a universal interface to hardware
- offers functionality allows basic commands

- **Examples**:
  - Glide: 3D-Graphics SDK (obsolete)
  - DirectX: Microsoft's 3D-Graphics SDK
  - OpenGL: Common SDK, implemented on many platforms
  - libgcm+edge: Playstation 3 Graphics Interface

Graphics Engine

- higher order access to graphics functions
- usually tailored to a specific graphics display
  - sprites
  - isometric
  - 3D
- works with models on a greater abstraction layer
  - sprites
  - characters
  - solids
- implements complex aspects of the display
  - mini maps
  - multiple views
  - overlays
  - special effects

Rendering Engine

- graphics engine models data only
- rendering turns the models into the display
  - rendering depends on the graphics card abilities
  - rendering is also based on the hardware abstraction layer

- functionality of the rendering engine:
  - low-level building and editing of the scene graph
  - visual effects (particle, dynamic shadows, Highly Dynamic Range rendering (HDR-Effects), Mirror effects, fog, …)
  - presenting the front end (GUI, Video, menus, Head-Up-Display (HUD))

Physics Engine

- Simulates “real” world rules
  - physical base rules (gravity, motion, inertia, ...)
  - collisions (often: Physics Engine = Collision Engine)
  - acting force on objects (Destruction, Breaking, Deforming, ...)
  - explosions
  - ragdoll characters (e.g. dynamic death animations)

- Physics is becoming an important aspect of many games
  - games become similar to real-time simulations
  - animations are based on physics
  - object interactions use physics (Collisions, ...)

Physics Engine

- Use is dependant on genre and perspective
  - simulation games: highly detailed, comprehensive Physics Engine
  - turn based Strategy games: No Physics Engine per se, possible moves are derived from more abstract rules of the game

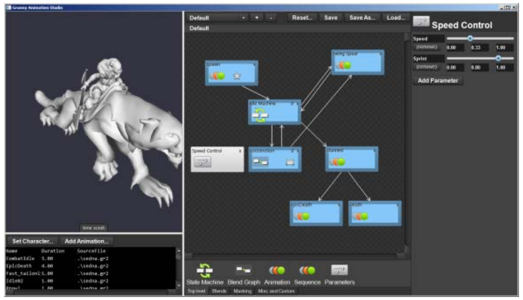
- Some available SDKs:
  - havok
  - open dynamics engines (ODE)
  - Tokamak
  - PhysX (Nvidia, Angeia) Software used in combination with specialized hardware
    (Physics Processing Unit (PPU))

Animation Engine

- necessary for seemingly natural movement of objects and characters
- offline-tasks:
  - *Motion Capture and Retargeting*: recording the movement of real models and transferring them to computer models
  - *Motion Editing and Adaption*: modification of recorded movement and transferring it from one model to another
- real-time tasks:
  - animation of sprites or textures
  - animation of graph models
  - movement of solid objects or skeletal-movement

Animation Engine

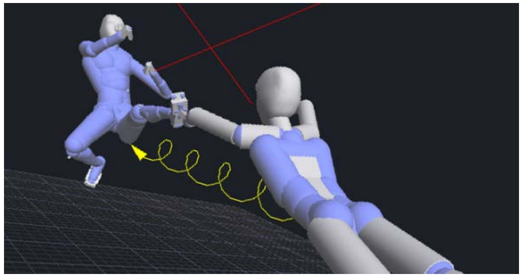
- Packages for implementing animation engines:
  - Granny (common, >2000 Games)



```
Name        Duration    SourceFile
CombatIdle  5.00        .\sedna.gr2
EpicDeath   4.00        .\sedna.gr2
Fast_tailonl1.00        .\sedna.gr2
Idle02      1.00        .\sedna.gr2
```

  - Havok Animation
    (Extension of Havok SDK,
    refers to Physics Engine)

  - Endorphin (for movies and games)



  - Edge (Animation for PlayStation 3)

Sound Engine and Audio Manager

- creation and playing of Sounds and music
- important for the game experience
- sound rendering:
  - playing sound in a predefined simulation
  - synchronizing audio and video
  - audio mixing (relative volume of channels)
  - 3D und distance difference
- common formats: midi, wave, mp3, ogg, acc, ...
- sound manager:
  - xACT for PC and Xbox 360
  - EA: SoundRIOT
  - Sony: Scream for PlayStation 3
  - IrrKlang, OpenAL, FMOD, …

Game Core

- core of the virtual simulated world
- saves the current State of the game world
- controls the transition from one valid game state to the next (Game Loop)
- time management: real-time, turn based
- granularity of presentation: all relevant information for the rules of the game
- traditionally interdependent with other components
- newer design models decouple the components
- for huge game worlds:
    - Assistance for spatial queries
    - Partitioning of the game world
- more in Chapter 2

Network and Multiplayer Modules

- Administrates the data transfer between several devices
- types of multiplayer games:
  - ***Single-Screen Multiplayer*** (e.g. Super Mario)
    - one view for the game world
  - ***Split-Screen Multiplayer*** (e.g. Super Mario Kart)
    - every player has her own view on the same screen
  - ***Networked Multiplayer*** (e.g. StarCraft, Counter Strike, ...)
    - approx. 2-20 Players, usually no dedicated server (dedicated client acts as server)
    - small overhead for data transfers (small game worlds)
  - ***Massive Multiplayer Spiele*** (e.g. Eve-Online, World of Warcraft, ...)
    - approx 1.000-100.000 Players in one game world
    - big overhead for data transfers (big game worlds)
    - costly game management
    - huge game worlds

Network and Multiplayer Modules

- effort is genre dependant: from easy to very elaborate
- distribution of game functionality to several machines:
  - synchronization of the game state
  - separation functionality: client-server model
  - everyone performs all tasks and data is exchanged: P2P
- design of high-level protocols
- user-authentication
- influence on game mechanics and design
  - game speed is dependent on latency
  - game flow is dependent on server capacity
- software platforms for network games: RakNet, GNE, ...
- More to come in Chapter 3

Persistence Layer

- characters and game states have to be saved
- storing replays/course of game
- for online games:
  - server task
  - coupled with account management
  - protection against data loss necessary (logging- and recovery-solutions)
  - problems with efficiency for big game states and real-time games
- more in chapter 4

AI Engine

- Artificial Intelligence (AI) is usually a part of game content and not necessarily part of the game engine
- basic functionality is encapsulated in AI-Engines:
  - pathfinding in obstructed environments: Search for entries and exits, avoid obstacles, ...
  - behaviour and interaction is often based on rules
  - Decision making: similar to expert systems
  - Group behaviour: swarming, independent agents, panic simulations, ...
- products and packages:
  - AI-implant (Presagis)
  - Kynapse (AutoDesk)
  - DirectAI (Masa)
  - SimBionic
  - …

AI Engine

- centralised group behaviour:
  - less system overhead
  - cooperation of entities is made easier
  - group composition is less flexible
  - unrealistic in some situations
- group behaviour as an Multi Agent System:
  - perception: situational input
  - decision: AI (rules are implemented, learned or random)
  - action: attempt to act upon the decision
  - group behaviour is a composition of several decisions
  - communication is needed for cooperative acting
  - since every group member has to be calculated, the necessary computing power is usually increased

Scripting Engine

- programming language of a higher order with direct access to functionalities of game core and AI-engine
- makes it easier for designers to create content
- is an important interface between game engine and game content
- examples:
  - LUA (http://www.lua.org)
  - GameMonkey (http://www.somedude.net/gamemonkey)
  - AngelScript (http://www.angelcode.com/angelscript)
- alternatively, the game can be created in the game engines programming language (this solution is often more efficient)

User Interface (UI)

- An efficient and intuitive interface is necessary for a satisfactory gaming experience:
  - immediate reaction to player input
  - quick access for necessary abilities
  - mild learning curve/learning by doing(tutorial levels)
- closely connected to the graphics engine
- input method as part of the gaming experience:
  - use of special controllers:
    microphone, Wii-guitar, Steering wheels, DK Bongos …
  - motion activated input:
    Wii-Controller, Xbox Kinect, PS Move, ...

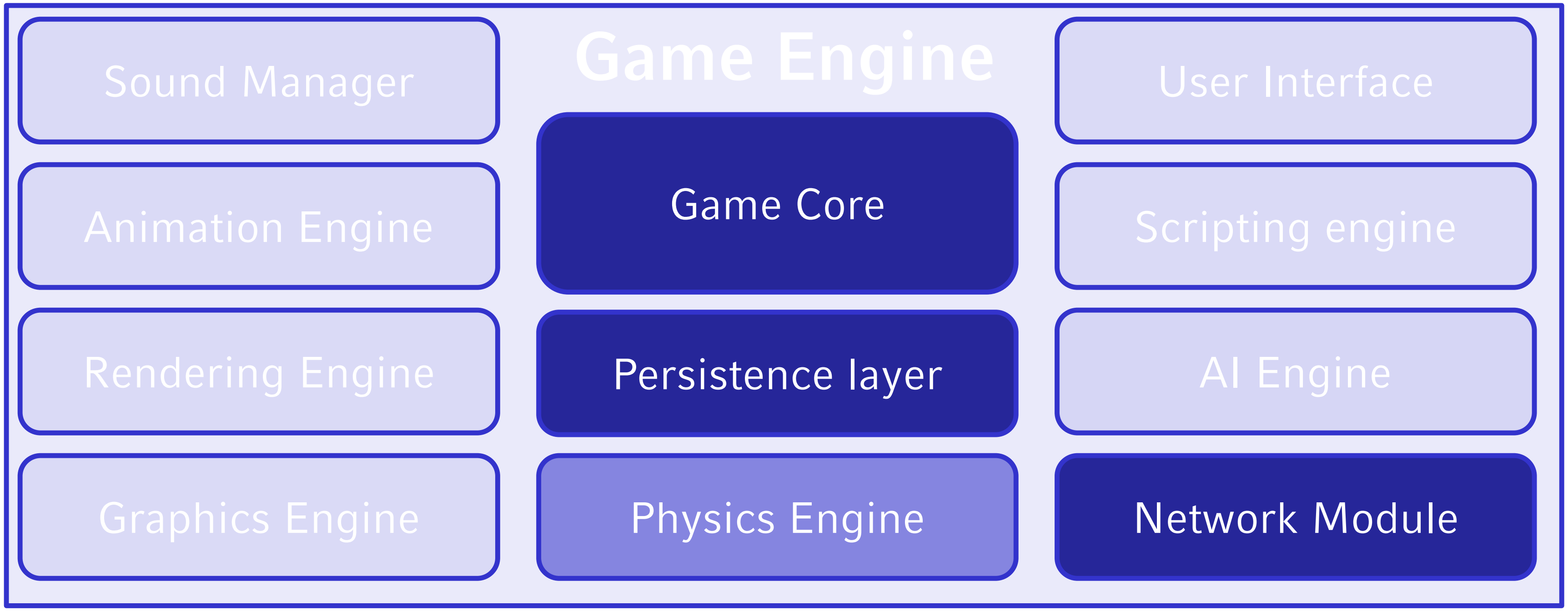
Game Content

- goals, quests, ...
- *entities*: player entities (characters), non-player entities (NPCs, mobs, ...), environmental entities (plants, rocks, etc..)
- levels, maps, dungeons, ...
- background stories
- cut scenes
- *game balance*: units, maps and difficulty
- *presentation*: models, textures, animations, effects, Sound, music (artistic design of the game)

Lecture Overview

1. **Part: Server Architecture**
    1. Game Core
    2. Distributed Games
    3. Persistence
2. **Part: Game Analytics**
    1. Temporal Analytics
    2. Spatio-Temporal Analytics
    3. Player and Team Rankings
3. **Part: AI in Games**
    1. Formalizations of AI
    2. Decision Strategies
    3. Antagonistic Planning

1st part of the lecture: Managing MMOs



- analysis of the dark blue highlighted components
- physics engine will be topic in conjunction with the game core

2nd Part of the lecture: Game Analytics

- successful game design is dependent on user behavior:
  - Is the game interesting in the long term?
  - Are there any unforeseen solutions to game problems, which make the game too easy and therefore boring? (Exploits)
  - Is the level of difficulty as expected by my focus group?
  - Are players gaining unfair advantages or are they circumventing the rules?
  - How do I find adequate opponents in Player vs. Player scenarios?
- Newer business models are increasing the pressure on game developers to create a long lasting gaming experience

⇒ ***Game Analytics***

Observation of player behavior to ensure compliance with the terms of use and to rate game design.

3rd Part of the lecture: Artificial Intelligence

- How to make game entities act intelligent?

  => formalization of states, rewards, actions..

- Path finding and spatial planning
- Scalability of computer controlled entities
- How does an environment react on my actions?
  - collaborative behavior
  - antagonistic behavior
- Many similarities to robotics and autonomous system (environment is much better controlled, no cyber physical links)

Bibliography

- Jason Gregory: Game Engine Architecture, AK Peters Ltd, MA,2009
- Nicolas Pronost: Game Engine Programming (http://www.cs.uu.nl/docs/vakken/mgep/)
- Critical Gaming Project: (https://depts.washington.edu/critgame/)
- Ralph Koster: A Theory of Fun for Game Design, Phoenix, AZ, Paraglyph 2004

What you should know by now

- definition of a computer game
- business models
- classic genres and characteristics for classification
- structure of computer games
- game architecture components

  in particular:

  - game core and physics engine
  - network
  - persistence layer
  - AI engine