Lecture Notes

Managing and Mining Multiplayer Online Games

for the Summer Semester 2017

Chapter 1: Computer Games

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

Organisatorisches

- ***Dates***
  - Lecture: Tue, 13.00 - 16.00 Uhr, room A 014 (Main Building)
  - Tutorials:
    - Wed, 14.00 - 16.00 Uhr room A015 (Edmund Rumpler-Str. 9)
    - Wed, 16.00 - 18.00 Uhr room A 015 (Edmund Rumpler-Str. 9)

- ***Homepage***:

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

- ***Tutorials***: Anna Beer
- ***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

⇒ *Challenge, 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.

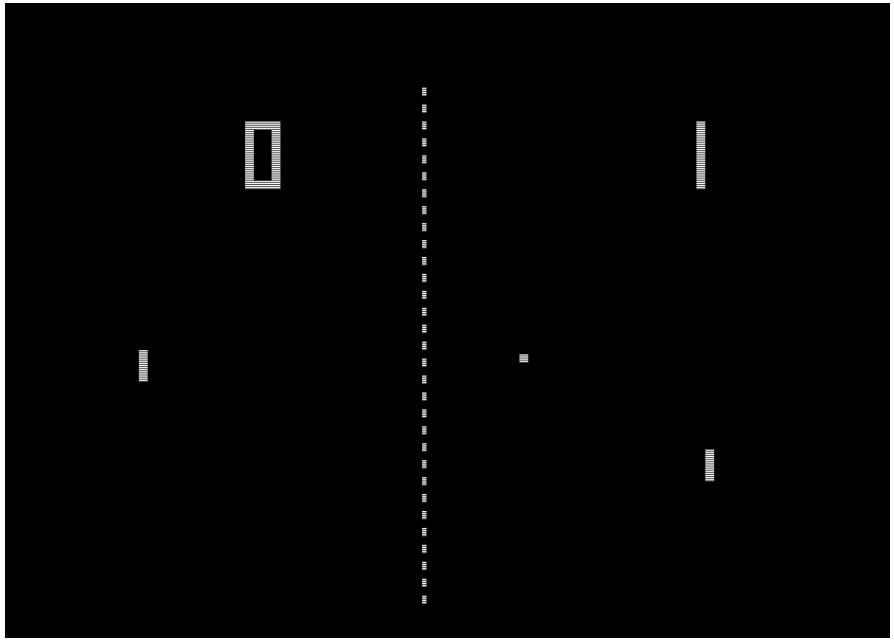
- *Geographic Informations Systems*: Managing spatial information (2D/3D-Maps) and simulating moving objects, e.g. cars.

- *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.

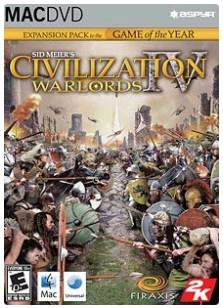
Source : http://www.theesa.com/facts/pdfs/esa_ef_2013.pdf

Business Models

Sale (Buy to Play)

- The customer purchase the software and use with a single payment
- Sold via shops (boxed games) or online
- Suitable for offline games
- High initial risk for the customer/ no or limited possibility for a refund if the customer is dissatisfied
- Danger of pirated copies
- Marketing and Advertising are crucial for success

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

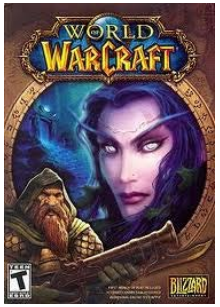


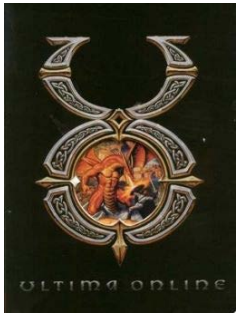
Business Models

Subscription (pay to play)

- The customer has to pay a regular fee for her account
- Suitable only for online games
- Account management is a necessity
- Gives Rise to new forms of criminal conduct (e.g. Account-Napping)
- Guarantees regular income for the game's manager
- Content-Updates and Customer satisfaction are important factors

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



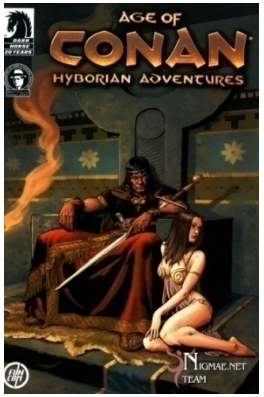


Business Models

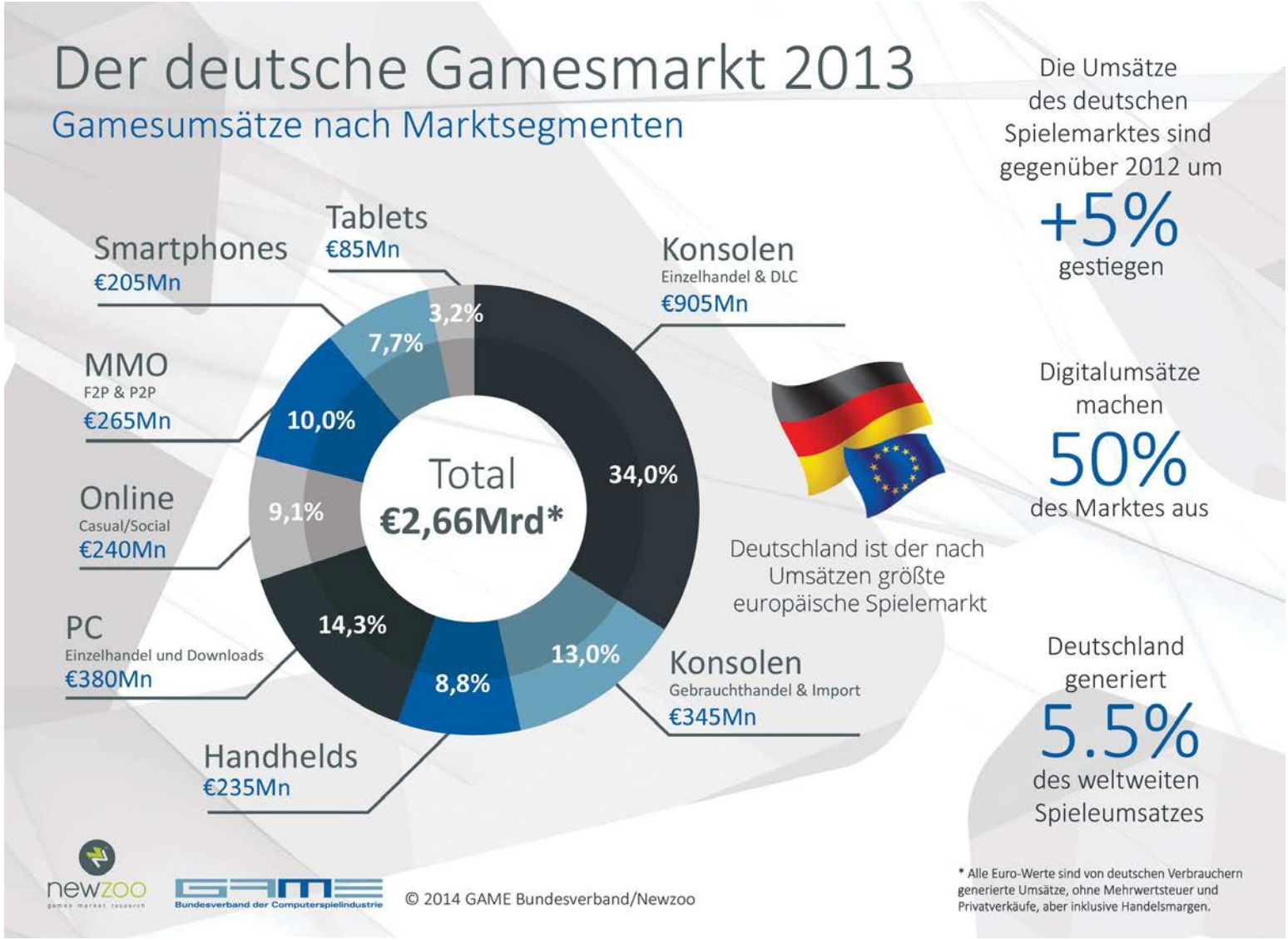
Micro-Transactions

- Sale of virtual currency, items or ability
- Suitable for Free-to-play games
- Greatest development risk but very high margin
- Lowest entry threshold for the customer and low initial risk (Players only invest money if they like the game.)
- Contains the most threats by fraud since ingame economics might be tied to real world economics

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



Marketshares in Germany 2013



http://game-bundesverband.de/index.php/de/neues/140-game-newzoo

Market development in the last years

- The market for Video and Computer Games is one of the biggest of the Entertainment Industry (approx. 93 BN US-Dollar worldwide)

  In comparison the market for database systems has a volume of approx. 20 BN US-Dollars

- In 2013 the percentage of digital only sales increased by 25%

- MMOGs, Casual-Game-Portals and Social Network Games grew between 27-66% in 2010

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 asf.
- 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

- *Quantity of controlled game entities*: Unit, Group, Faction
- *Perspective*: First-Person, Third Person, Top-Down
- *Temporal Sequence*: Real-Time, Game-Time, turn based
- Complexity of Control: Simulation vs. Arcade,
  Manual Aim vs. Auto-Target, …
- *Dimension/Granularity of the gaming world*: 2D, 3D,
  Grids vs. Free Movement
- *Number of Players*: Single-Player, Multi-Player (1-100),
  Massive-Multiplayer (100-100.000)
- *Account development*: virtual abilities, player abilities (RPG vs. Chess)
- *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 over many units, Top Down perspective, low granularity of control, Real-time, 2D-/3D-Welt, 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, Third-Person, mediocre granularity of Control, Real-time, 3D-World, High 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, Herr der Ringe 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, Third-Person View, medium granularity of control, Real-Time, Unit Development, Teamplay,..
- *Goal*: Destruction of the enemy base, sometimes other
- *Examples*: League of Legends(LoL), Dota, Dota2, Heroes of the Storm,..

Classic Game Genres

First-Person-Shooter (FPS)

- Simulation Fire Fights
- One Unit, First-Person, Real-Time, High Granularity of Control, 3D-World, usually 1-20 Players
- *Examples* : Wolfenstein 3D, Doom, Unreal, Half-Life, Team Fortress, Halo, Counter Strike, …

Classic Game Genres

Racing Games

- Controlling Vehicles
- One Unit, First- and Third-Person, Realtime, between medium and very high granularity of control, 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 Close Combat with Enemies
- One Unit, Third-Person View, medium granularity of control, 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 for Conflict Resolutions
- Many Units, Top-Down, Realtime or Turn based, low granularity of control, 2D-World, Single or Massive Multiplayer
- *Examples*: Intopia (1963), M.U.L.E (1983), Civilization, Die Siedler, Sim City, Travianer, Total War Series, ...



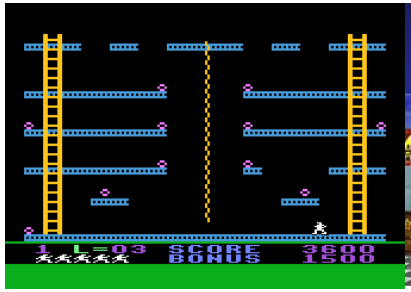
Classic Game Genres

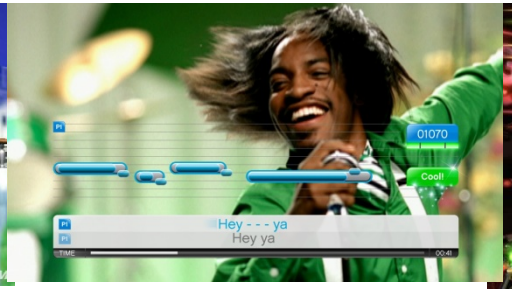
Adventure Games and Interactive Movies

- Games that include an interactive story
- Interaction: solving puzzles and collecting items
- Third- and First-Person, variable Granularity of Controls and Movement
- Often transition to other genres
- *Examples*: Zorc, Monkey Island, Leisure Suit Larry, Deponia, Firewatch, Telltale Game Series, Until Dawn


More Genres

- ***Jump 'n' Run***: Third Person Control of a character in a 2D/3D Enviroment (e.g. Super Mario, Jumpman, Yokai Laddle, ...)
- ***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 wiede 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 differ in complexity and scope dependent 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 extende to an MMOG, due to scaling and the necessity to analyze player behavior

Development Team Composition

- ***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 the 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

These Factors speak against the use of Standard Architecture:

- Focus on content
- Different requirements for different games
- Performance requirements
  (less resources used mean more potential players)
- Hardware-ressources are subject to constant change

Consequently: Architecture is more likely to be integrated and specific than modular and reusable.

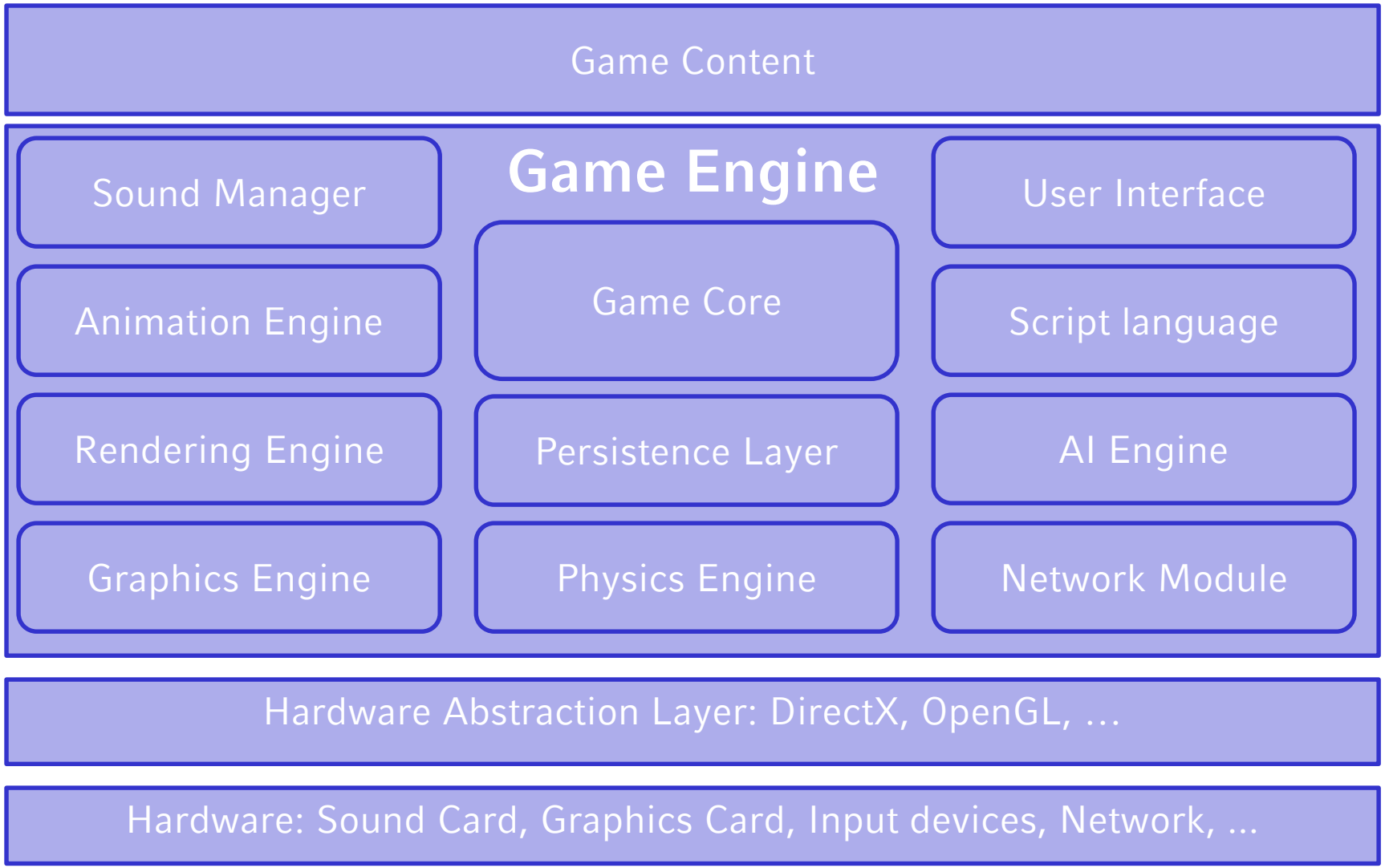
But: Games are complex software system

$\Rightarrow$ Use of standard components (*Engines*)

$\Rightarrow$ Layered architecture and modular build

$\Rightarrow$ Reusability of code for other games (*Mods*)

Building Blocks in Game Architecture



Hardware Layer and Hardware Abstraction Layer

- Hardware components
  - graphics controlleer 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

- Encapsulation of hardware functionality for different hardware configurations (PC, Android, ...)
- Offers a universal interface to hardware
- Offered 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 Vviews
  - 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

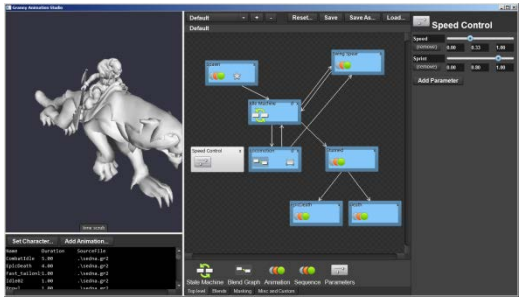
- Some available SDKs:
  - Havok
  - Open Dynamics Engines (ODE)
  - Tokamak
  - PhysX (Nvidia, Angeia) Software used in combination with specialized hardware
    (Physics Processing Unit (PPU))

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

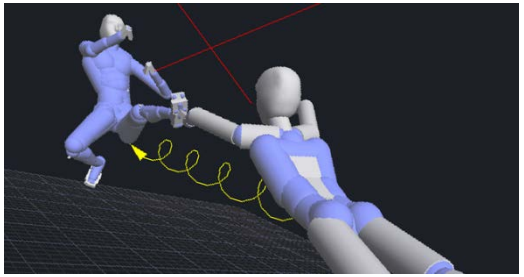
Animation Engine

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



  - 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

- Expenditure is genre dependant: from easy to very elaborate
- Distribution of game functionality to several machines:
  - Synchronization of Game States
  - Separation functionality: Client-Server model
  - Everyone performs all tasks and data is exchanged: Peer-to-Peer (P2P)
- Design of high-level protocols
- User-authentication
- Influencing 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

- Needed to save characters and game states
- Saving 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
- Nowadays basic functionality is encapsuled in AI-Engines:
  - Pathfinding in obstructed enviroments: Search for entries and exits, avoid obstacles, ...
  - Behavior and interaction is often based on rules
  - Decision making: similar to expert systems
  - Group behavior: swarming, independent agents, panic simulations, ...
- Products and packages:
  - AI-implant (Presagis)
  - Kynapse (AutoDesk)
  - DirectAI (Masa)
  - SimBionic
  - ...

AI Engine

- Centralised group behavior:
  - Less System Overhead
  - Cooperation of entities is made easier
  - Group composition is less flexible
  - Unrealistic in some situations
- Group behavior as an Multi Agent System:
  - Perception: situational input
  - Decision: AI (Rules are implemented, learned or random)
  - Action: attempt to act upon the decision
  - Group behavior 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:
    Mikrophon, Wifi-guitar, Steering wheels, DK Bongos …
  - Motion activated input:
    Wii-Controller, Xbox Kinect, PS Move, …

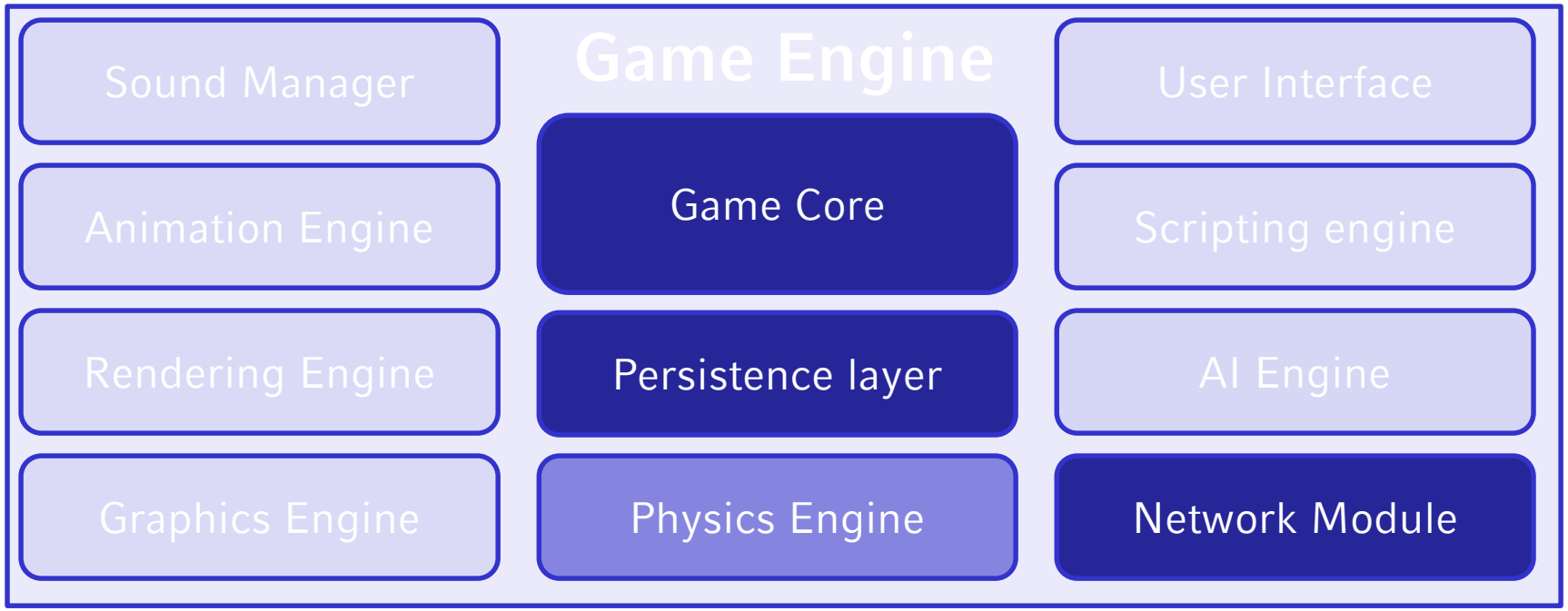
Game Content

- Goals, Quests, ...
- *Entities*: Player entities (characters), Non-Player entities (NPCs, Mobs, ...), Enviromental entities (Pflanzen, Felsen, ...)
- Levels, maps, dungeons, ...
- backround 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

$\Rightarrow$ ***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