GADE Summary (beliebig lang)

17. Februar, 2026; rev. 4. August 2026

Linda Riesen (rieselin)

1 Vorlesung 01

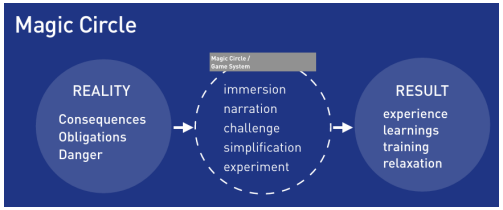

Abbildung 1: Magic Circle

1.1 Play vs Game

Play:

- no rules —> Rules emerge in play (maybe)
- learn concept of rules
- only rule: everything can change
- Examples: minecraft, sandbox games, 'open world games' toys, Lego, etc.

Game:

- Have rules, have win condition / end condition
- Table rules may exist
- Cheating is possible
- Examples: FPS, RPG, MMO, Roguelike, Gacha Board game, euro game, soccer, card game, tabletop rpg

Abbildung 2: Game vs Play

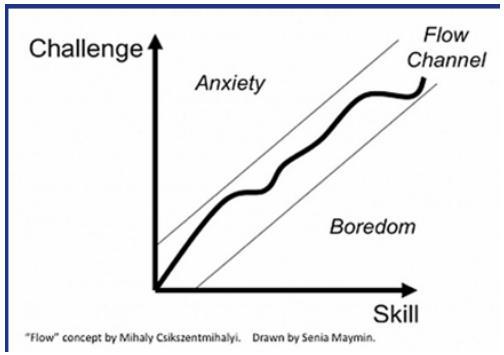

Abbildung 3: Flow Channel

1.2 Game Mechanics

1.2.1 Decision / Choice

Decision is every moment during a game in which there are two or more options available to the player.

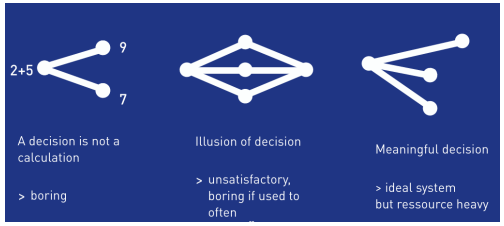

Abbildung 4: Decision

Micro / Macro Mechanics Micro mechanics: Short term (seconds /minutes)

- more challenge
- action
- pressure, etc.

Macromechanics/Long term: 15+ minutes, hours, days

- Progress (minimap)
- Story

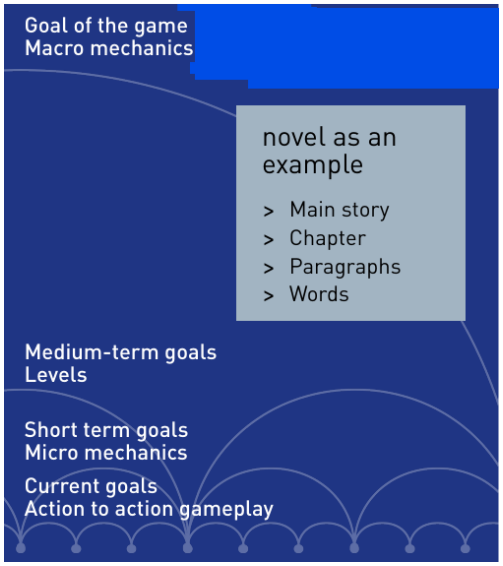

Abbildung 5: Pacing

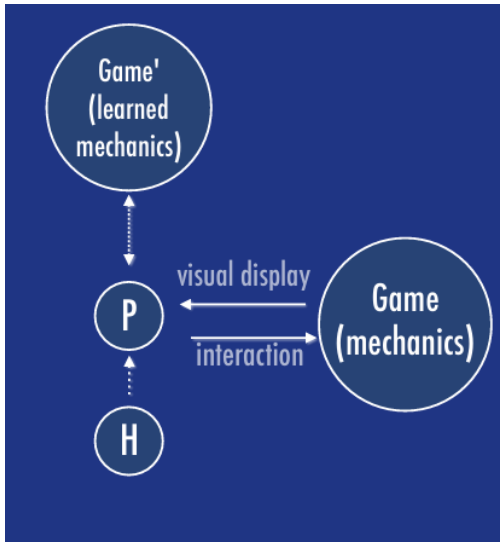

Abbildung 6: Game Mechanics vs Game Play

1.2.2 Motivation

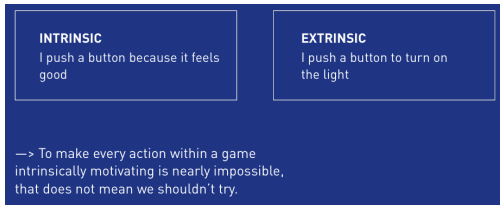

Abbildung 7: Intrinsic v Extrinsic Motivation

Abbildung 8: Types of Gamers

Abbildung 9: Motivation Types

1.2.3 Skinner-Box (conditioning)

Gambling / Loot drops / Loot boxes

1.2.4 Reward / Punishment

How games reward & punish:

- resources (money, life ...)
- feedback: sound, graphics, animation
- progress, story
- Interactions in the game world / cyberspace

Metagame:

- Login rewards
- achievements

Note: Gamification & serious games often use no punishment.

1.3 Game Design Applications

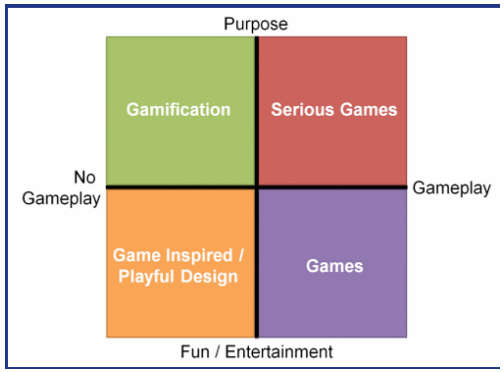

Abbildung 10: Game Design Applications

1.3.1 Serious Game

“Games are fun, but some things are not. How about we use games to make boring things fun, and teach people things they wouldn’t otherwise learn or do?"(have purpose beyond the game)

1.3.2 Gamification

In gamification, non-game real-world processes are enrichted with game design elements, hoping to increase motivation.

2 Vorlesung 02

- Arcade Games: Custom hardware, individual
  - Racing, Tanks, Flippers
  - Design space: infinite possibilities
- Console Games
  - Mainstream product

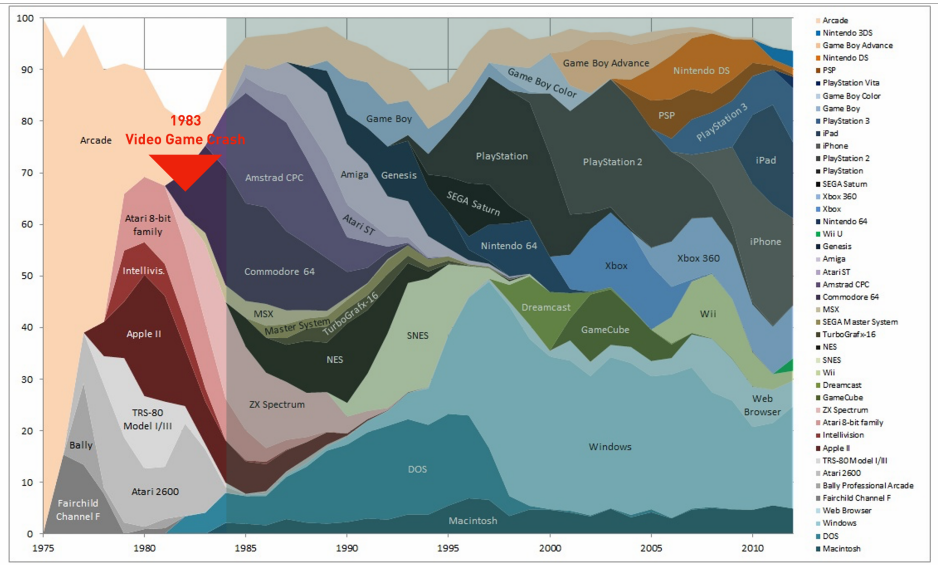

Abbildung 11: Game Platforms

  - Child-accessible

- Computer Games:
  - Geek product, expensive
  - more and more Virtual Machines (VM) Example: Doom/Unreal/Unity3d

- Mobile Games
  - Mass market product & emerging markets
  - cheap | free-to-play

3 Vorlesung 03

3.1 The Hero's Journey

Act 1: The Departure

1. Ordinary World – The hero's normal life before the adventure begins.
2. Call to Adventure – The hero is presented with a challenge or mission.
3. Refusal of the Call – The hero hesitates due to fear, doubt, or obligations.
4. Meeting the Mentor – A guide provides training or tools to prepare the hero.
5. Crossing the First Threshold – The hero commits to the journey and enters the unknown world.

Act 2: The Initiation

1. Tests, Allies, and Enemies – The hero encounters challenges, forms allian-ces, and faces enemies.
2. Approach to the Inmost Cave – the most dangerous part of the journey, where they face their greatest fear or a powerful enemy.
3. The Ordeal – a major battle or crisis where the hero is tested to their limits.
4. Reward – After victory, they receive a Reward: knowledge, treasure, or power.

Act 3: The Return

1. The Road Back – The hero starts the journey home but faces new obstacles.
2. Resurrection – The hero faces a final challenge, emerging stronger. This is a moment of growth.
3. Return with the Elixir – The hero brings back newfound wisdom or a phy-sical
4. reward to benefit the world.

3.2 Freytags Pyramid

1. Exposition (Introduction)
2. Rising Action
3. Climax
4. Falling Action
5. Dénouement (Resolution)

4 Game Worlds

What they are and why we need to think about them!

- Game World: “Space” where the game takes place
- Visual Representation
- Model of the game rules
- Built to be experienced through interaction
- Rule-based world

4.1 Dimensionality

Dimensions of the World:

1. Game mechanics: x dimensions
2. Display: y dimensions

> x and y do not need to match!

4.2 Types of Worlds

- **1D Worlds:** all in one line, game elements: difference in colors and pulse: LineWobbler
- **2D Worlds:** Single Screen, 2D (No Scrolling, fixed Perspective/Camera): PacMan
- **Scrolling 2D worlds:** Worlds bigger than one screen, Can create illusion of space with depth (Doors), minimaps / overviews (e.g. Gris)
- **2.5D Worlds** Worlds are visualized in 3D Mechanics on 2D (e.g. Far)
- **3D Worlds** 3D imports more complexity. Camera (Player) Perspective
- **Non-Euclidean Worlds:** Impossible Geometry, Space “can not be trusted”, Geometrical laws suspended (e.g. Superliminal)
- **4D Worlds:** Additional “Dimensions” (e.g. Time: Does time move linearly forward / haptics / Smell)

4.3 Rules: Bindings, Guidance Systems, Feedback

- Players need to “read” the world
- Bindings: Game mechanic (common visual elements, common behaviors, common interaction models)
- Guidance Systems: (small game mechanics, GUI (minimap, cursors, hotspots, markers), in Game (Colors, Lines, light, camera angles, highlights))

5 Vorlesung 05

5.1 Collision Layers and Masks

This feature controls which objects can collide with each other.

- **Collision Layers:** Assign objects to specific layers (e.g., “Enemies,” “Environment”).
- **Collision Masks:** Define which layers interact using the Layer Collision Matrix.

5.2 Spatial Queries

Spatial queries retrieve data based on location and proximity, analyzing how objects relate to one another within a defined space
Data structures to support fast spatial queries:

- Islands: group of elements without connection.
- Sleeping: rigid bodies with velocity under threshold become deactivated for several time steps.

5.3 Particle System

Particle use simplified assumptions due to their large number of elements:

- Particles do not collide among themselves.
- Shadows cast only on the environment, not among themselves.
- No light reflection; each acts as a point light.
- Randomness prevents dull behavior.
- Finite lifespan for realism.

6 Vorlesung 06

- **Tessellation**: Tessellation refers to the process of subdividing or refining a mesh by adding more details (vertices, edges, and faces) to achieve a higher level of detail (LOD).
- **Material**: determines an object's visual appearance by controlling how it reacts to light. It's a container for settings like color, texture, shine, and transparency
  - **Shader:** A set of math instructions that calculates how light interacts with surface.
  - **Texture**: An image file wrapped around the model for color or detail.
  - **Shininess:** Controls how "glossyör roughthe surface looks.
- **PBR (Physically Based Rendering)**: is a shading method that calculates how light interacts with surfaces using real-world physics.
  - **Albedo:** color of an object's diffuse reflected light.
  - **Texture map:** contains its flat, unaltered color.
  - **Metalness:** specifies how strongly light will reflect.
  - **Roughness**: Controls surface micro-detail; higher values blur reflections and broaden highlights.
- **UV Mapping** is the process of applying a 2D image (texture) to a 3D surfaceto create realistic colors, patterns, and details. It uses UV coordinatesto determine how the texture wraps around the model.
  - U represents the horizontal (left to right) axis of the texture.
  - V represents the vertical (top to bottom) axis of the texture.
- **Gaussian Splatting** is a 3D technique that replaces sharp, solid points with soft, semi-transparent ßplats"(Gaussians)
  - Soft Geometry: Replaces rigid triangles with blurry, mathematical ßplatsfor smoother transitions.
  - Natural Realism: Overlapping soft shapes create realistic surfaces, and complex lighting.
  - Primary Uses: The modern standard for high-fidelity 3D photo reconstruction and cinematic particle effects.

7 Vorlesung 07

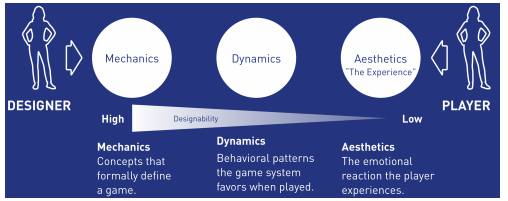

Abbildung 12: Designer can not design aesthetics, only observe them

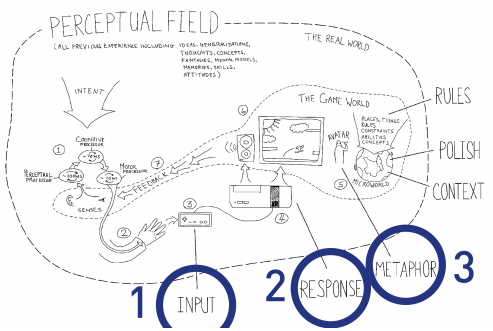

Abbildung 13: Perceptions

1. Controls & Controller
   - Controller = Haptic reality for player
   - Choice controller —> type of game —> platform
   - Avatar = Extension of Player
2. Response & Feedback
   - Responsiveness of controls: Good controls —> Good Toy —> Good game
   - Feedback: You can not put too much of it!
3. Metaphors of Play
   - Avatar: How is the player represented in the Game world?
   - Character: Who am I? How do I relate to Gameworld or PC?
   - Camera: How is the game world framed?

8 Vorlesung 08

8.1 SOLID Principle

- **Single Responsibility** each class or module should have only one responsibility (good for Readability, Extensibility, andReusability)
- **Open-Closed Principle** Classes must beopen for extension but closed for modification.
- **Liskov Substitution** Principle Subclasses must be able to replace their base classes without breaking the game.
- **Interface Segregation Principle** Clients should not depend on unused methods: Instead of one massive interface, use small ones like IMovable, IDamageable, to ensure classes only implement the behaviors they actually need.
- **Dependency Inversion Principle** High-level modules shouldnot import anything directly from low-level modules.

8.2 Delegates

A delegate is a type in C# that allows you to reference a method as a variable. It's like a function pointer: you can assign, pass, and invoke methods dynamically. They're used a lot in events, callbacks, and custom messaging systems.

8.2.1 Event

An event is a special type of delegate used to broadcast notifications to multiple subscribers, while enforcing encapsulation and safety.

8.2.2 Unity Event

A UnityEvent is a serializable event system that allows you to attach functions via the Inspector, acting as Unity's version of the observer pattern in the UnityEngine.Events namespace.

8.3 Asset Bundle

A Unity Asset Bundleis a collection of assets (models, textures, audio, prefabs, etc.) packaged into a single file that you can load at runtime. It's mainly used to reduce initial build size, support downloadable content (DLC), or update content without rebuilding the entire game

8.3.1 Addressable

Addressable is a system for efficient, flexible, and optimized asset loading and management, improving on traditional Asset Bundles

8.3.2 Streaming Assets

Purpose: For files that should stay unchanged and be included in the build exactly as they are (e.g., videos, config files). Access: Read-only; use Application.streamingAssetsPath

8.3.3 Resources

- Purpose: For assets that can be loaded at runtime using Resources.Load().
- Usage: Good for prototyping or loading assets dynamically.
- Downside: All assets in the Resources folder are included in the build, even if unused, which can increase build size.

8.3.4 persistentDataPath

- Purpose: For reading/writing data during gameplay (e.g., save files, downloaded content).
- Access: Read and write; use Application.persistentDataPath.
- Downside: Path varies by platform which makes manual debugging difficult. File Security & Vulnerability, Risk of Data Loss

8.4 Profiler

The Profiler allows developers to analyze game performance, identify bottlenecks, and optimize projects by tracking metrics like frame rate, CPU usage, and more.

8.5 Stats Window

Displays real-time metrics to guide performance optimization.

- FPS: Frames rendered per second; the primary indicator of game smoothness.
- Draw Calls: The number of times Unity commands the GPU to render objects.
- Triangles: The number of triangles being rendered on the screen.
- Vertices: The number of points in 3D space.

9 Vorlesung 09 Visual Experience

Elements of of Visual Experience

1. Visual Style: Look, Environment & Characters
2. Game UI
3. Motion: Effects & Animations

9.1 Visual Style

Art Style

- Set of visual characteristics that allow us to categorize artworks categorize.
- A set of design rules that permit artists to produce artworks of a similar style
- Often associated with a time period (art often shapes our mental image of a time period

Semiotics (Study of Symbols (and their meaning)) Layers: Description, Interpretation, Function

Choice of style

- Tied into Game Concept phase (Gameplay, Theme & Setting, Narrative, Visual Representation, But secondary considerations: Brand, Cost)
- Sense Pleasure: Satisfactory to look at (“Eye Candy”)
- Emotional response: stronger magic circle
- Coherence: One visual language, learnable
- Marketing: Unique Visual ID / branding
- Cost: Certain styles are more costly to make than others.

Tech Art (Game Aspects) Lighting

- Color and mood of scene
- Performance optimization:
- Lighting is very expensive!
- Postprocessing & Color Grading

Shader Development

- Writes Shader Programs
- Shaders define how objects are rendered in a game

Rigging specialist

- Makes 3D Models animatable

10 AI in GADE

Game AI (Runtime / Behavioral AI): set of tools, techniques, and methodologies to create engaging player experiences. Rather than seeking perfectßolutions, it focuses on using AI methods tailored specifically to make games fun and immersive. AI tools for game development (Generative / Productivity AI) Learnable AI: designed to be predictable so players can eventually master and exploit it. Instead of being random, the NPC follows logical patterns that a player can learn, anticipate, and counter. Intuitable AI: It exhibits responsive behavior, with active decision trees operating behind the scenes

10.1 AI Decisions

The state: game snapshot at specific moment action: action that ai can do to change game state state space: map of all possibilities (collection of every single situation the game could possibly be in at any given moment) State Transition System: A state transition system defines how actions change one state into another.

10.2 Decision Making Algorithms

Finite State Machine (FSM): FSM is a computational model with a fixed set of states, inputs, and outputs, used to design systems that transition between defined states. Behaviour Tree (BT): A structured system that breaks down an agent’s decision-making into a clear, multi-layered hierarchy BDI: Beliefs-Desires-Intentions: Beliefs: The AI’s perception of the world, which can be partial or incorrect (player location, health status). Desires: States the AI wants to achieve: preserve health, find / damage player... Intentions: Which states the AI wants to commit to (idle, follow, attack) GOAP: Goal Oriented Action Planning: Goals: Represent the desired states that the AI agent aims to achieve. Actions: defines a set of actions that the AI agent can perform. Planning: find action sequences to reach goals based on preconditions & effects. Execution: The AI follows the planned action sequence in the game.

10.3 Pathfinding

Steering behavior refers to a set of algorithms used to control the movement of characters or entities in video games. (Path following, Waypoints, Path refining) NavMesh: Unity define walkable areas

11 Vorlesung 12 Level Design

11.1 Was ist ein Level

- Classic games: stage / segment
- Open world: spatial area
- RPGs: Character levels
- Other definitions: stage/level/area/location/theme/difficulty/end condition/
- goals

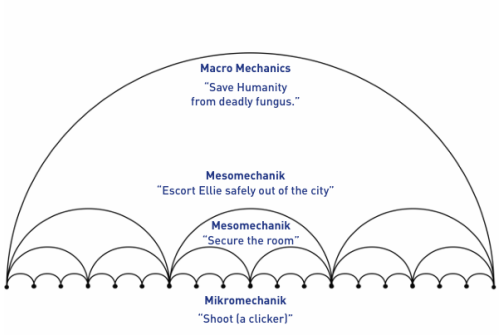

Abbildung 14: Level vs Game Mesomechanics

11.2 Level Design Principles

1. Design of Space
2. Clear Goals, Many Solutions
3. Navigation: Guidance Systems
4. Various Levels of Difficulty
5. Levels constantly teach AND surprise
6. Tell a story
7. Create Emotion
8. Core Mechanic drives the level Design