Design of computer video games

6. Gameplay and core mechanics

Boyan Bontchev

Agenda

- Introduction to game mechanics - history, definitions and basic concepts
- Gameplay – progressive and emergent
- Core mechanics
- Resources, simple and compound entities
- Events, processes, and conditions
- Internal economy of games
- Using random numbers in games
- Examples

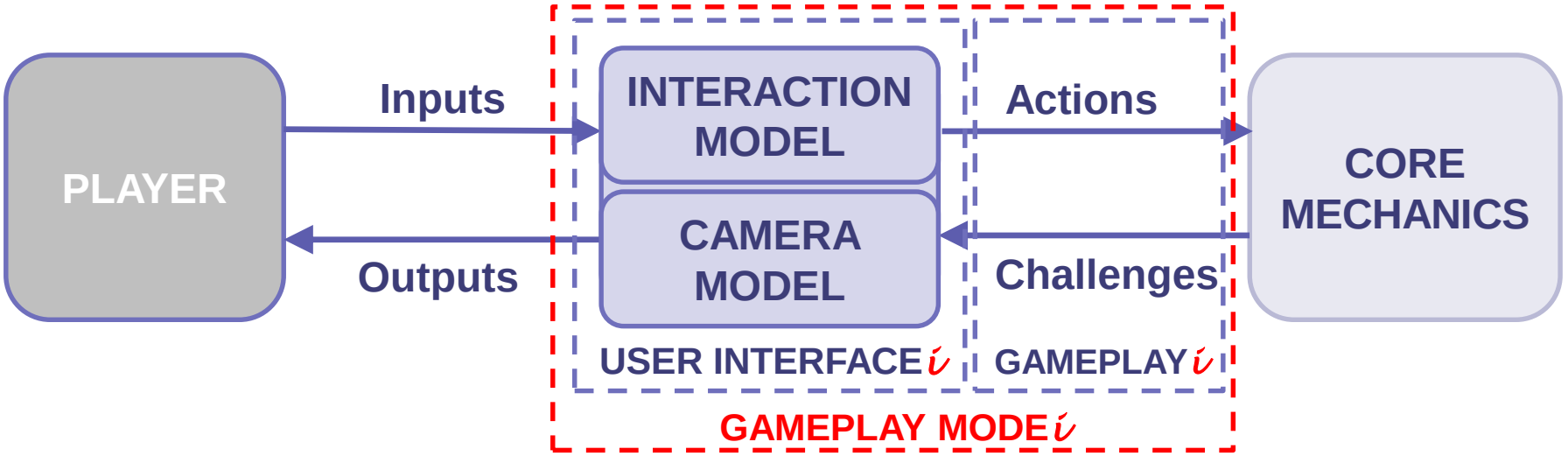
References

- Adams, E. Fundamentals of Game Design, Third Edition, Pearson Education, Inc., ISBN-13: 978-0-321-92967-9, 2014
- Salen, K., Zimmerman, E . Rules of Play - Game Design Fundamentals, MIT Press Cambridge, Massachusetts London, England, ISBN 0-262-24045-9, 2004.
- Fullerton, T., Swain C., Hoffman, S. S. Game Design Workshop: A Playcentric Approach to Creating Innovative Games, Second Edition, Elsevier, ISBN 978-0-240-80974-8, 2008.
- Adams, E., Dormans, J. Game Mechanics: Advanced Game Design, New Riders Games, ISBN-13: 978-0-321-82027-3, 2012.
- Smith, M., Queiroz, C. Unity 4.x Cookbook, Packt Publishing, Birmingham B3 2PB, UK, ISBN 978-1-84969-042-3, 2013.
- Other references are cited in slides

From Module 5

What is the first designer job?

- RULE: First job is to design the *primary* gameplay mode

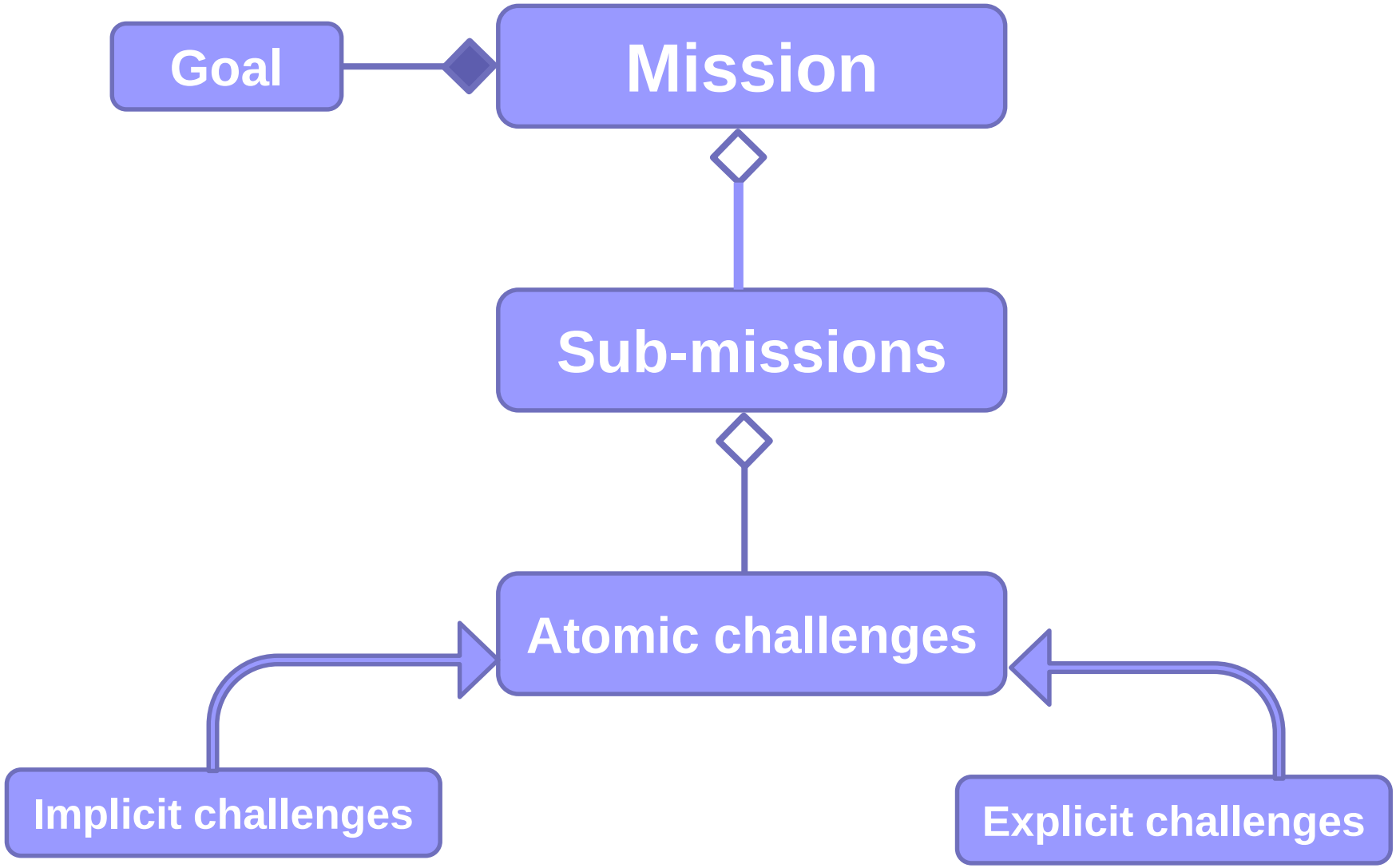


- Only AFTER the *primary* gameplay mode – details of its UI
- Gameplay modes *typically* share a number of UI features, so it's best to define all the modes before you begin UI
- In case of many modes - *wait* until you have planned the game structure and game moves from mode to mode

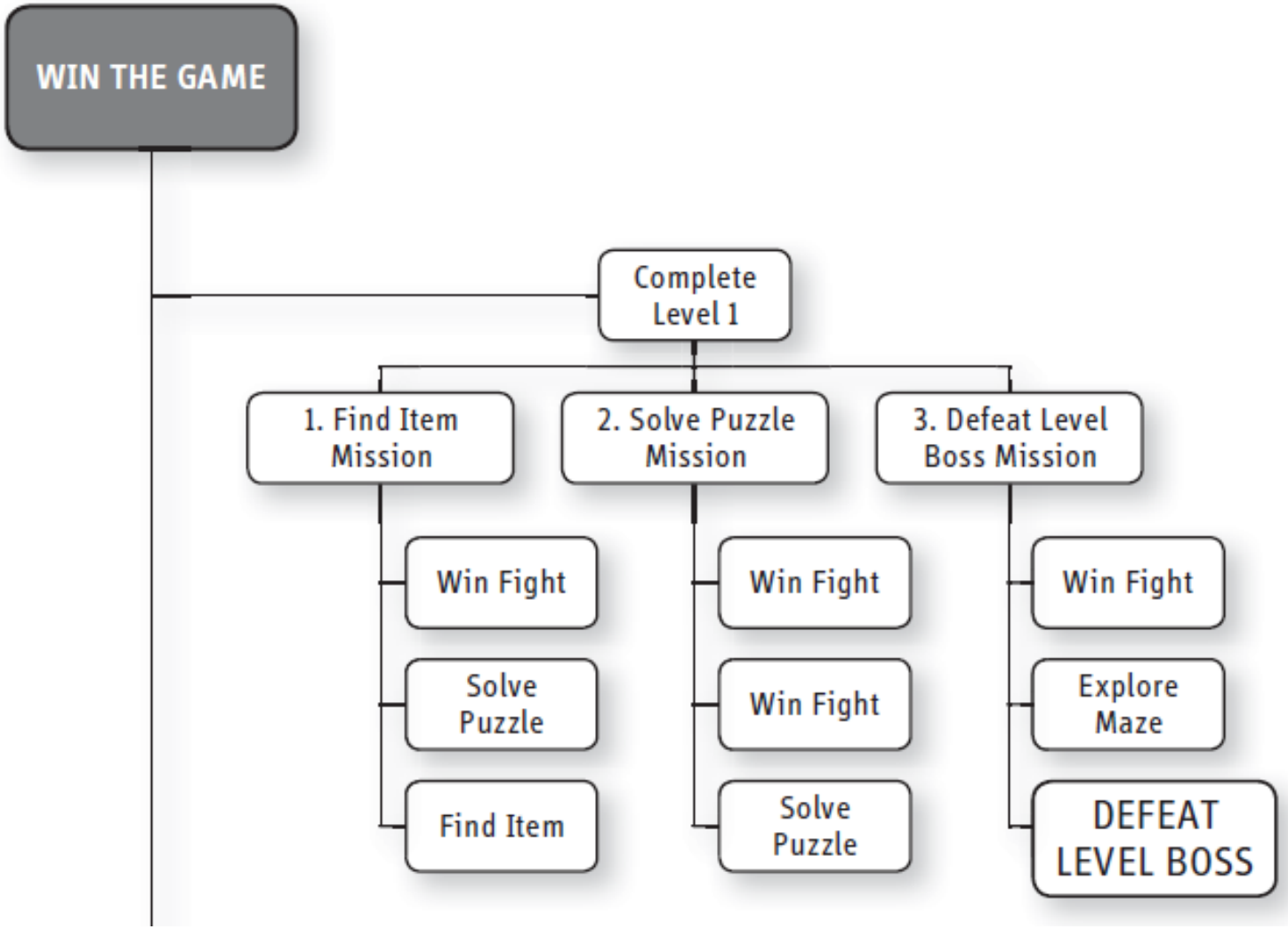
Better gameplay – true fun

- **Gameplay comes first**
- **Get a feature right or leave it out -** it is far worse to ship a game with a broken feature than to ship a game with a missing feature
- **Design around the player** - you must examine *every* decision from the player's point of view
- **Know your target audience** - different groups of players want different things - for any group, *know* what they want and what they think is fun, and then provide it
- **Abstract or automate parts of the simulation that *aren't* fun**
- **Strive for harmony, elegance, and beauty**

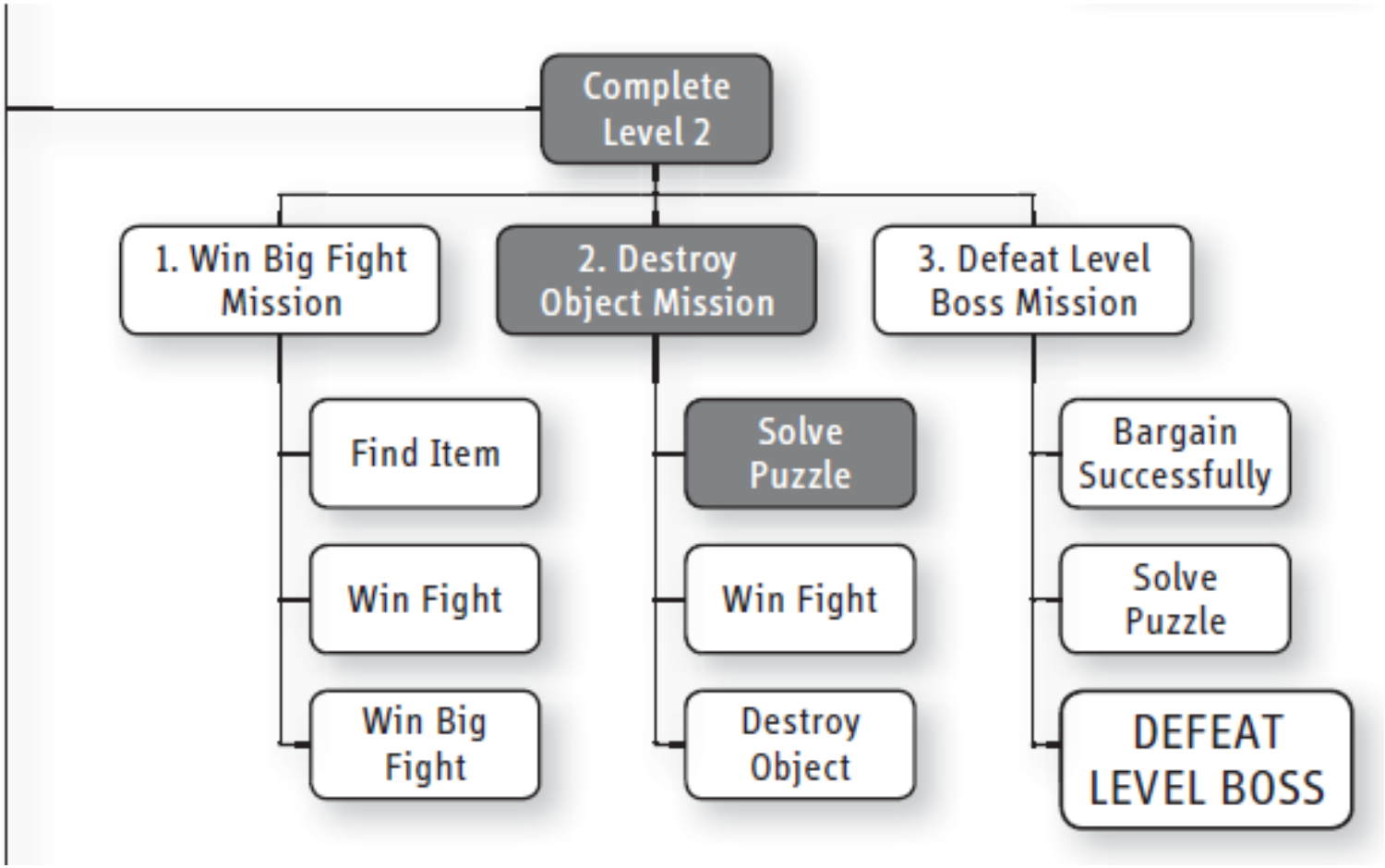
Hierarchic challenges



Challenges in an action-adventure game 1/3

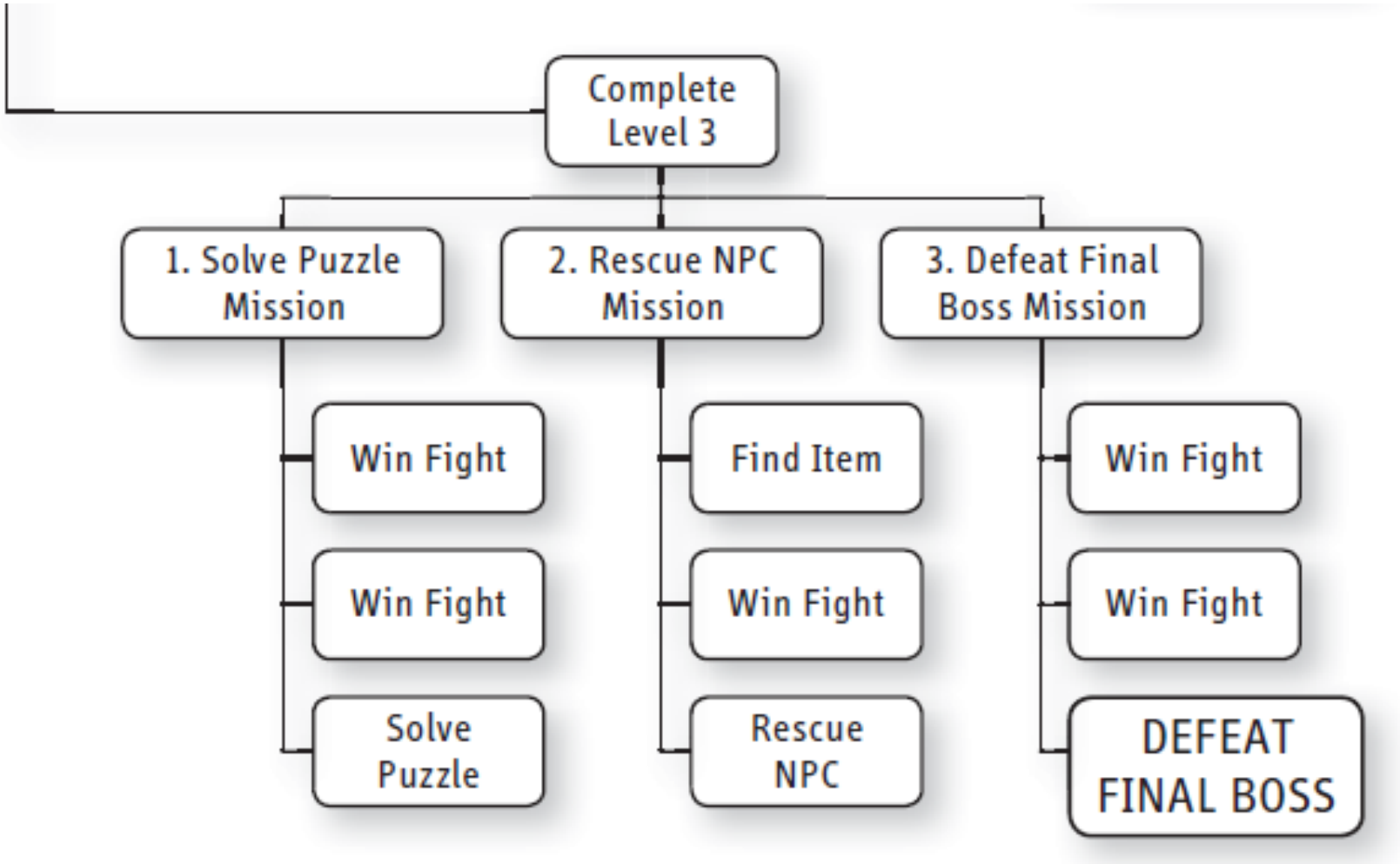


Challenges in an action-adventure game 2/3



Gray boxes indicate the challenges the player faces simultaneously at one particular moment.

Challenges in an action-adventure game 3/3



Example: Rush for Gold gameplay

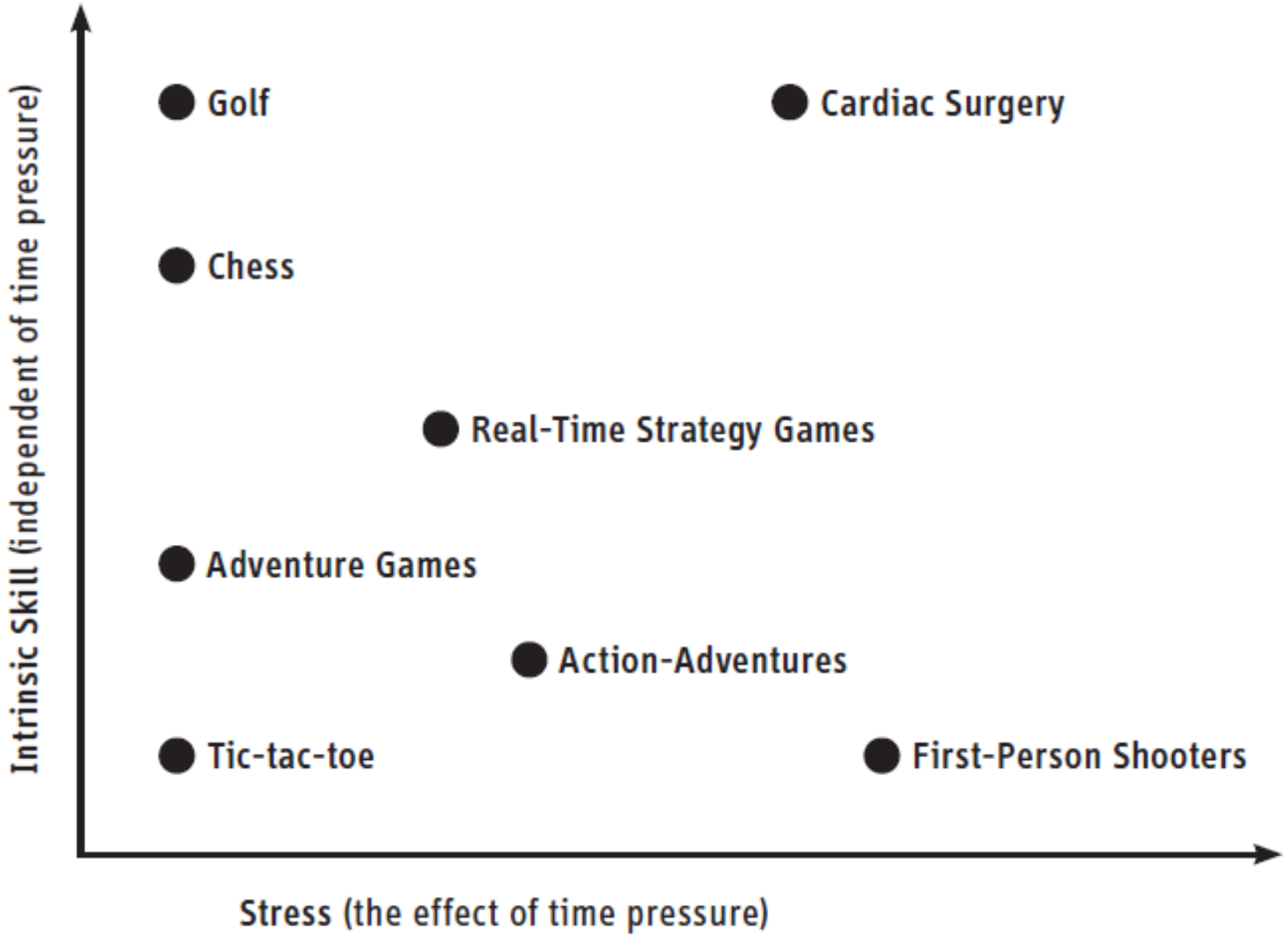
Hints

- The topmost level includes the victory condition for the entire game
- Present the overall victory condition explicitly
- Teach the player explicitly how to meet the atomic challenges:
  - by game's *tutorial levels*
  - by self-explanatory gameplay *and controls*

Absolute difficulty depends on:

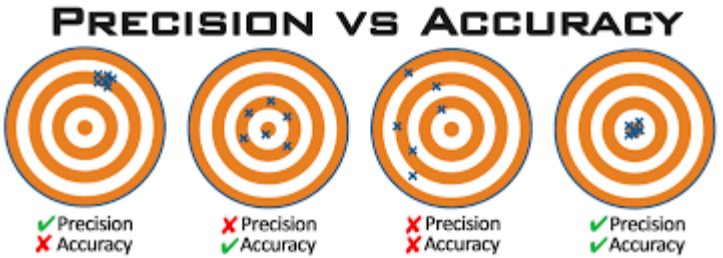
- *intrinsic skills required* – needed to surmount the challenge *if you give the player an unlimited amount of time to do it. Varies for:*
  - *Physical tasks – archer game, shooting*
  - *Mental tasks – sudoku, trivia*
  - *Combined tasks*
- *stress* - measures how a player perceives the effect of time pressure on his ability to meet a challenge requiring a given level of intrinsic skill. Stress in:
  - Tetris – to complete the task under time pressure
  - Golf - ?

Intrinsic skills vs stress for tasks



Physical coordination challenges

- Speed and reaction time
  - Pong
  - Tetris,
  - Candy Crush…
- Accuracy and precision
- Intuition for physics
  - simulators
- Timing and rhythm
  - Dance Dance Revolution
  - Guitar Hero…
- Combination moves



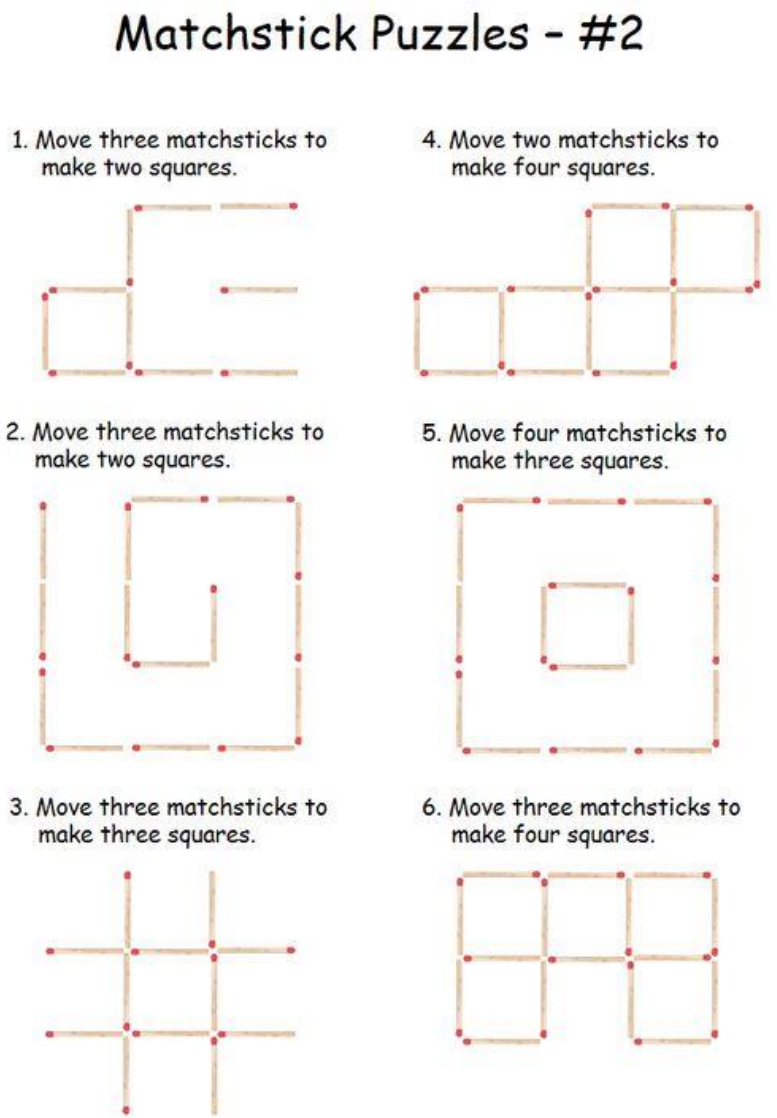
Mental challenges 1/6

- Logic and mathematical challenges
  - Hint: do not make puzzles that can *only* be solved by trial and error
- Time pressure - discourages careful strategic thought and instead encourages direct, brute-force solutions (e.g., in 15 seconds to get through a host of enemies and disarm a bomb)
- Factual knowledge challenges ≠ conceptual reasoning challenges

Design of computer video games

Mental challenges 2/6

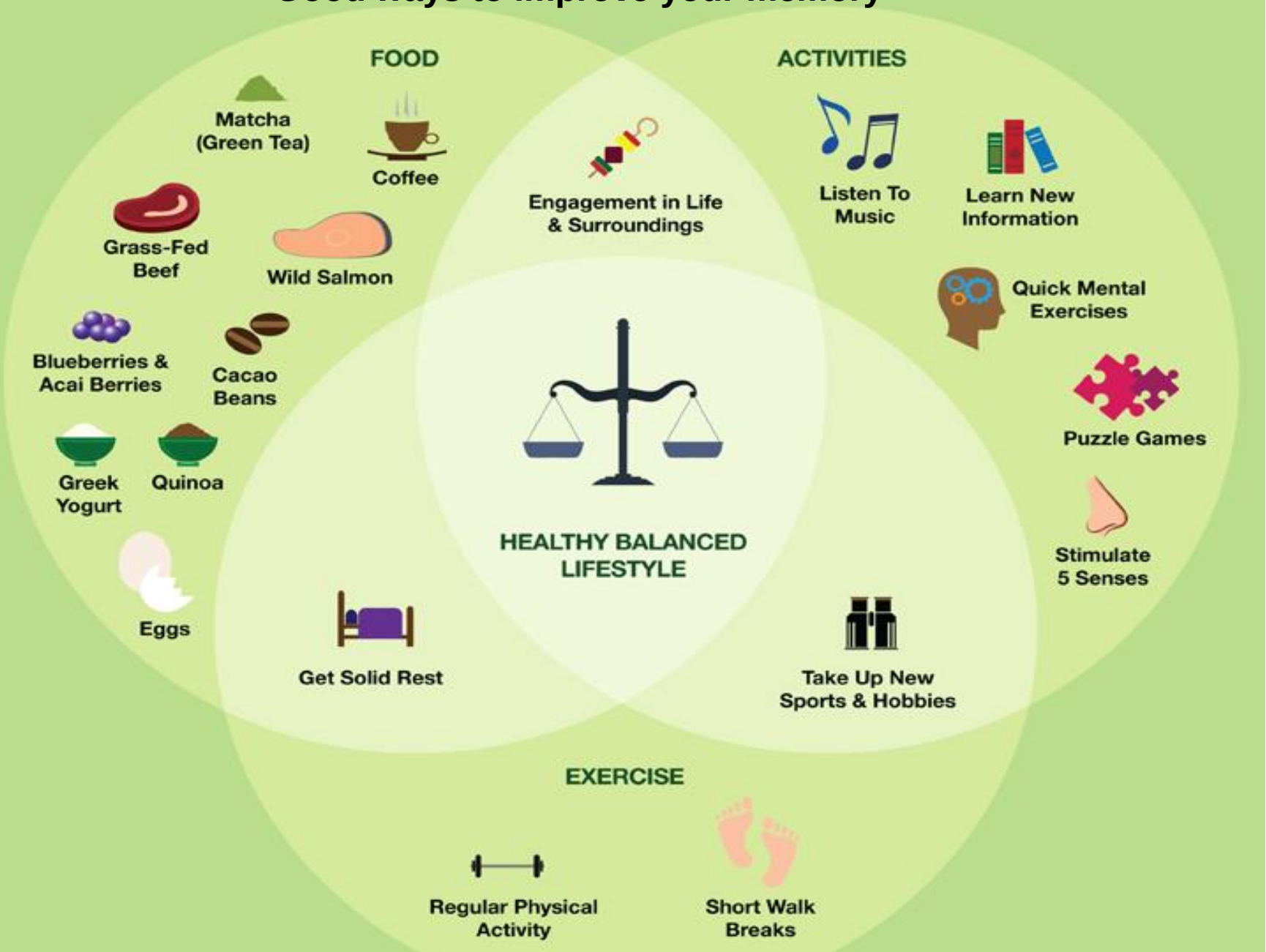
- *Lateral thinking* puzzles – related to conceptual reasoning puzzles, but they add a twist: The terms of the puzzle make it clear to the player that *what seems to be the obvious* or *most probable solution is incorrect* (or the necessary elements to achieve the obvious solution are unavailable).



Mental challenges 3/6

- Memory challenges - test the player's ability to recall things that she has seen or heard in the game

Good ways to improve your memory



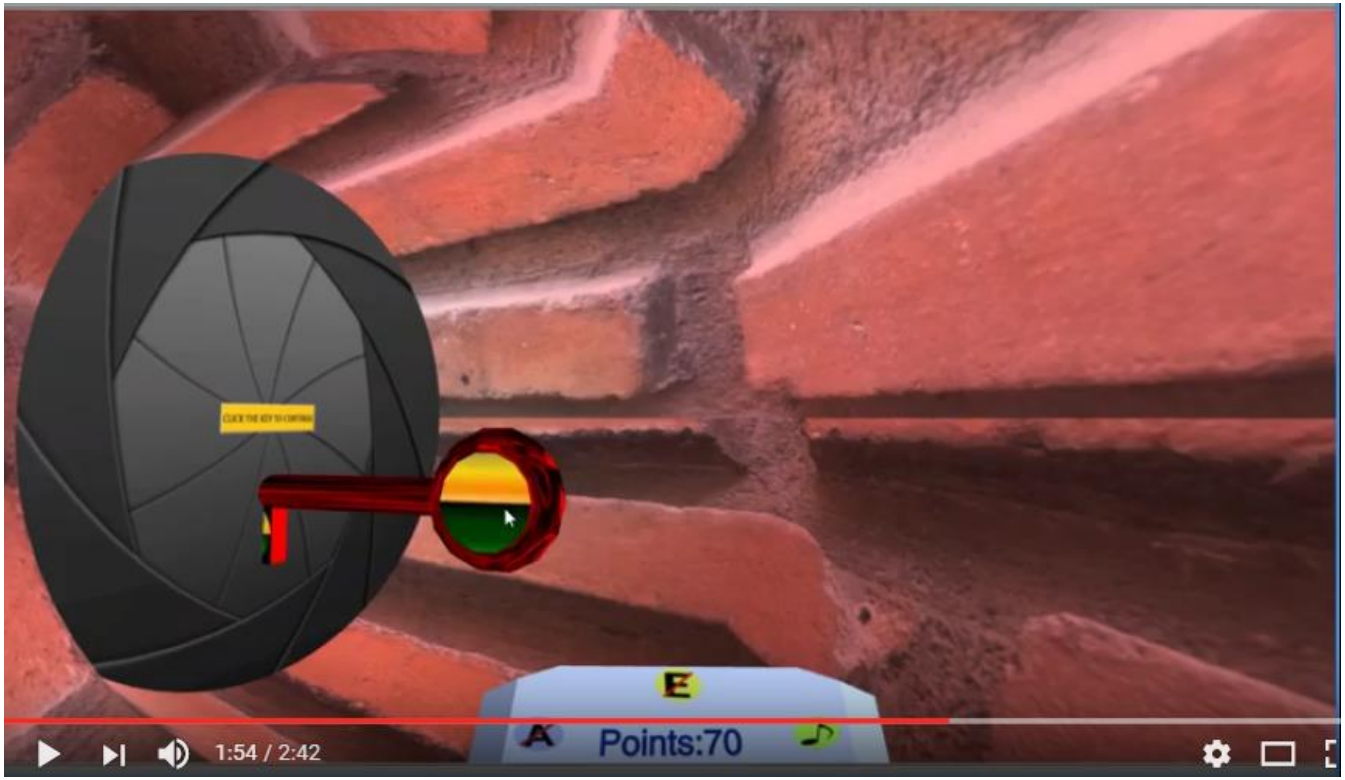
Source: https://ticsyformacion.com/2012/01/16/como-funciona-la-memoria-infografia-infographic-health/

Mental challenges 4/6

- Pattern recognition challenges - test the player's ability to spot visible or audible patterns or patterns of change and behavior – Brain Spa
- Exploration challenges
  - spatial awareness challenges
  - locked door - any obstacle that prevents the player from proceeding through the game until he learns the trick for disabling it
  - traps
  - mazes (labyrinths)
  - illogical spaces - areas do not relate to each other in a way that the player might reasonably expect
  - teleporters
  - hidden objects – e.g. *Easter eggs* (hidden in obscure locations)

Video

- STRAMAG (STRAtegic MAnagement Game) Maze - https://www.youtube.com/watch?v=q_xkshEe4RU&index=3&list=PLs4ZK9XAjRxpROiCNQ1kdDR8SFpH-itmY



Mental challenges 5/6

- Logistics - supporting troops in the field and bringing fresh troops to the front lines
- Conflicts
- Strategy ≠ Tactics

Mental challenges 6/6

- Survival / defending
- Stealth - the ability to move undetected
- Economic challenges
  - Accumulating resources
  - Achieving balance

Gameplay types

- Progressive gameplay
- Emergent gameplay
- Mixed gameplay – both progressive and emergent

Progressive gameplay

- Refers to video game mechanics in which the designer sets a course of action that a player must complete to move forward in the game
- Used for majority of games thanks to its solid storyline
- Depends on checkpoints that a character must reach to advance to the next level, and vary according to the game genre:
  - in action, adventure and RPG - defeating the level boss
  - in puzzle games – solving a sequence of puzzles
  - in racing games – finishing given track/route
- Designers know the course a game will take, so they can build a much deeper and more complex story around that course

Source: https://www.techopedia.com/definition/27044/progression-gameplay

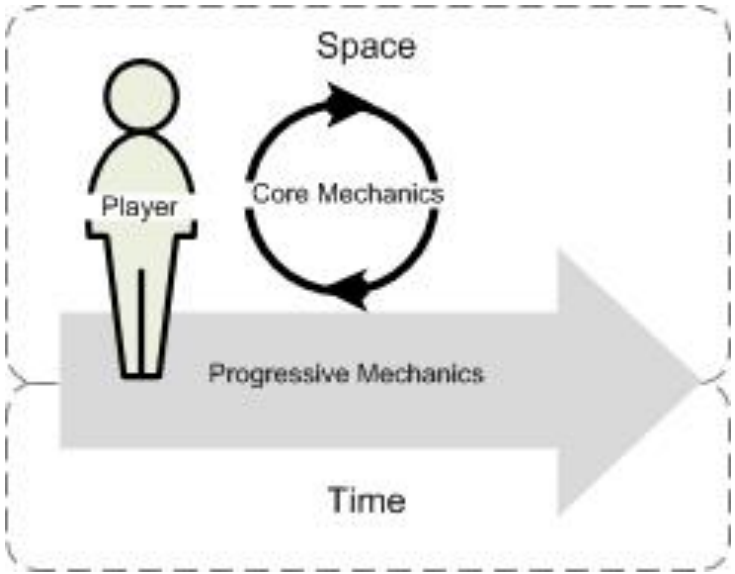
Emergent gameplay

- Refers to video game mechanics video game mechanics that change according to the player's actions
- Includes a number of relatively simple decisions that a player must make leading to more complex outcomes
- Can also be created by adding multiple players to the same game environment and having their individual actions impact the overall game narrative – like in *The Sims*
- As well, more complex AI capable of impacting the storyline in unpredictable ways can be used instead more players
- Becomes more and more popular as online gaming platforms offer more opportunities to create a complex gaming environment with simple gameplay mechanics

Source: https://www.techopedia.com/definition/27043/emergent-gameplay

Progression vs emergency

- Pro's and con's?
- Some RPG allow actions to impact narrative significantly. More fun/immersion?
- Defenders of emergent gameplay want games where random players actions affect both the story and the game world, leading to unlimited possibilities compared to the outcomes of progressive gameplay. +/-?

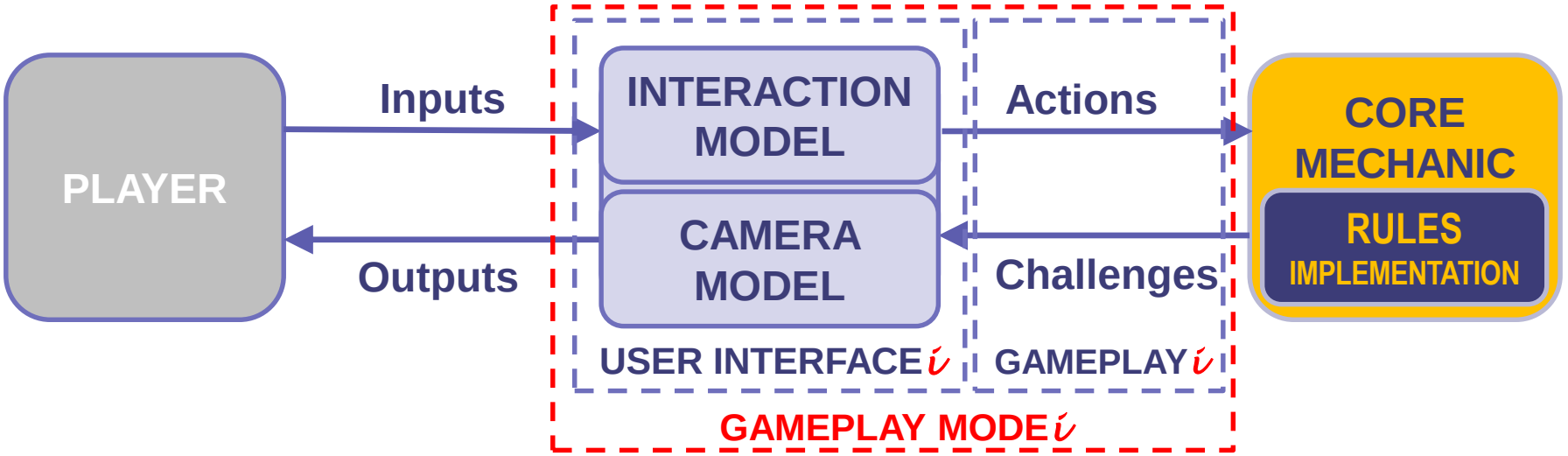


Saving the game

- Saves a snapshot of a game world and all its particulars at a given moment so that the player can later load the same data and play the game from that point
- Reasons:
  - **Play break -** players leave the game and return to it later
  - **Recovery from disastrous mistakes** – e.g., the death of the avatar
  - **Exploration of alternate strategies of playing**
- Techniques:
  - DB
  - Serialization
- Policy:
  - Auto-save
  - Administered save

How gameplay is generated?

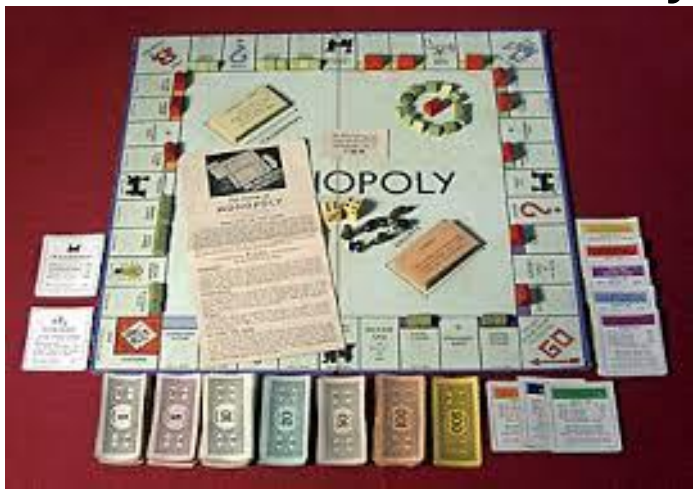
- RULE: First job is to design the *primary* gameplay mode



- The gameplay mode is includes UI is and gameplay is
- Gameplay is generated by core mechanic
- Core game mechanic *is implemented* by the game engine
- Core game mechanics *implement* the rules
- The core mechanics sends triggers to the storytelling engine

Core game mechanics

- Definition*: The core mechanics consist of the data and the algorithms that precisely define the game's rules and internal operations.*
- Example: The rules of *Monopoly* are less than three pages, but are *not sufficient* to build a computer game. To properly specify the core mechanics of *Monopoly*, you should include rules plus prices of each of the properties, the different amounts of rent, money, the layout of the board, and the effects of all the chance and Community Chest cards.



Functions of core mechanics 1/2

- **Operate the internal economy of the game -** specify how the game or the player creates, distributes, and uses up the goods on which the game bases its economy
- **Present active challenges** to the player via UI, as the level design specifies:
  - Active challenges are those governed by mechanics
  - Passive challenges (e.g., a chasm that the avatar must jump over), don't have mechanics of their own

Functions of core mechanics 2/2

- **Accept player actions** from UI and implement their effects upon the game world and other players
- **Detect success or failure in all challenges in the game +** termination conditions of the game
- **Operate the artificial intelligence** of NPC
- **Switch the game from gameplay mode to another mode**
  - keeps track of modes
  - updates gameplay
  - signal UI engine to update UI
- **Transmit triggers to the storytelling engine**

Real-time vs turn-based games 1/2

- Turn-based games - most played on a board
  - Players play in turns
  - In a turn-based game with no artificial opponents, the core mechanics remain idle while a player takes his/her turn
  - In a turn-based game that does have artificial opponents or NPCs, the mechanics don't remain entirely idle between turns because they must compute their behavior

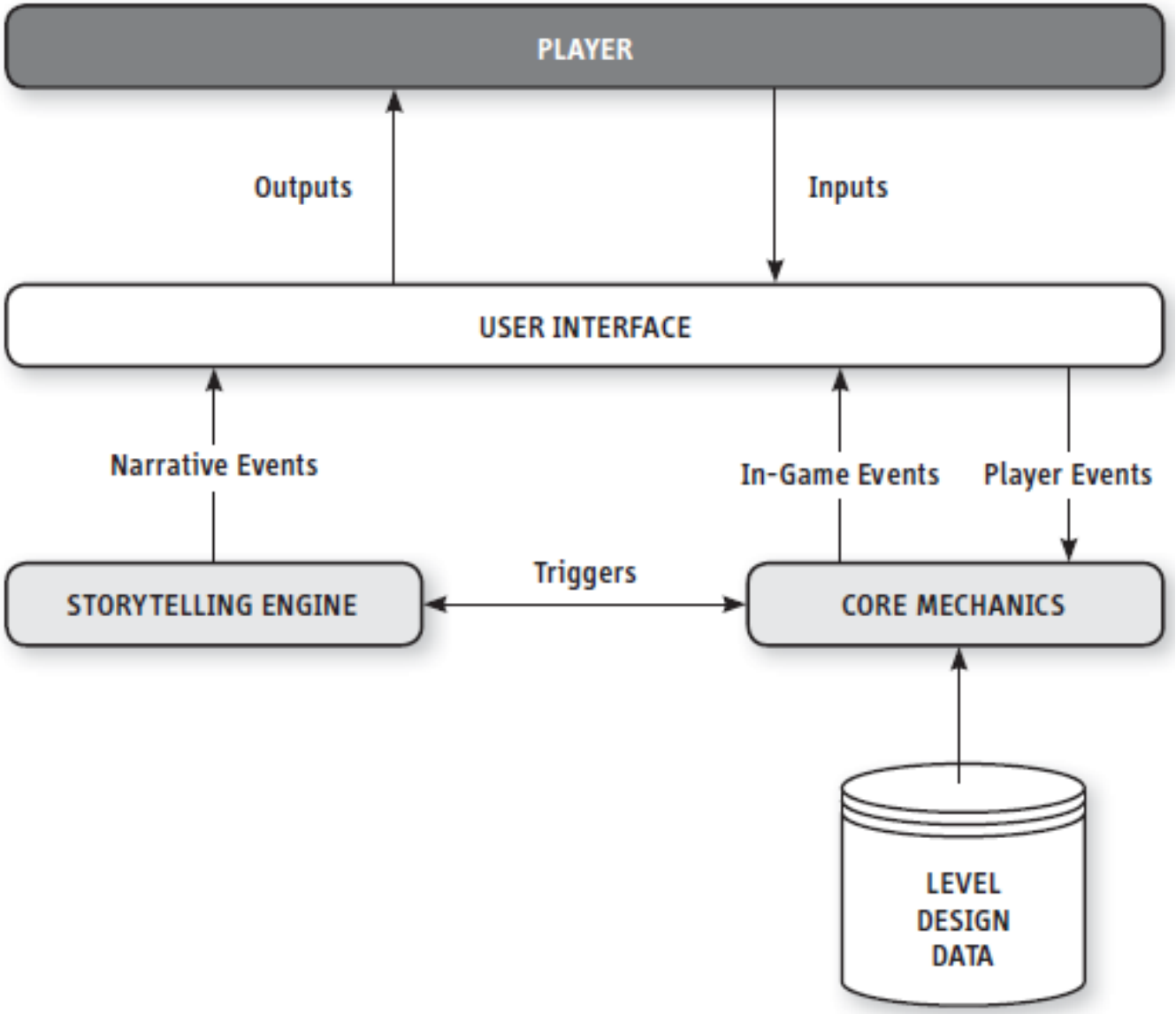
Real-time vs turn-based games 2/2

- Real-Time (RT) games - most games operate in real time
  - All players play simultaneously
  - Mechanics are processes that:
    - operate continuously or for extended periods and
    - specify the parameters of a real world that operates on its own whether the player acts or not
  - NPCs do their business
  - Shot/collect/talk/… events happen at players' action only

Core mechanics and level design

- Level design specifies
  - type, timing, and sequence of challenges that appear at given level
- Core mechanics specify
  - how challenges work upon
    - player actions
    - changes of the game world
  - how level design are read data from the storage
  - not exactly which challenges each level will contain

The same in a picture

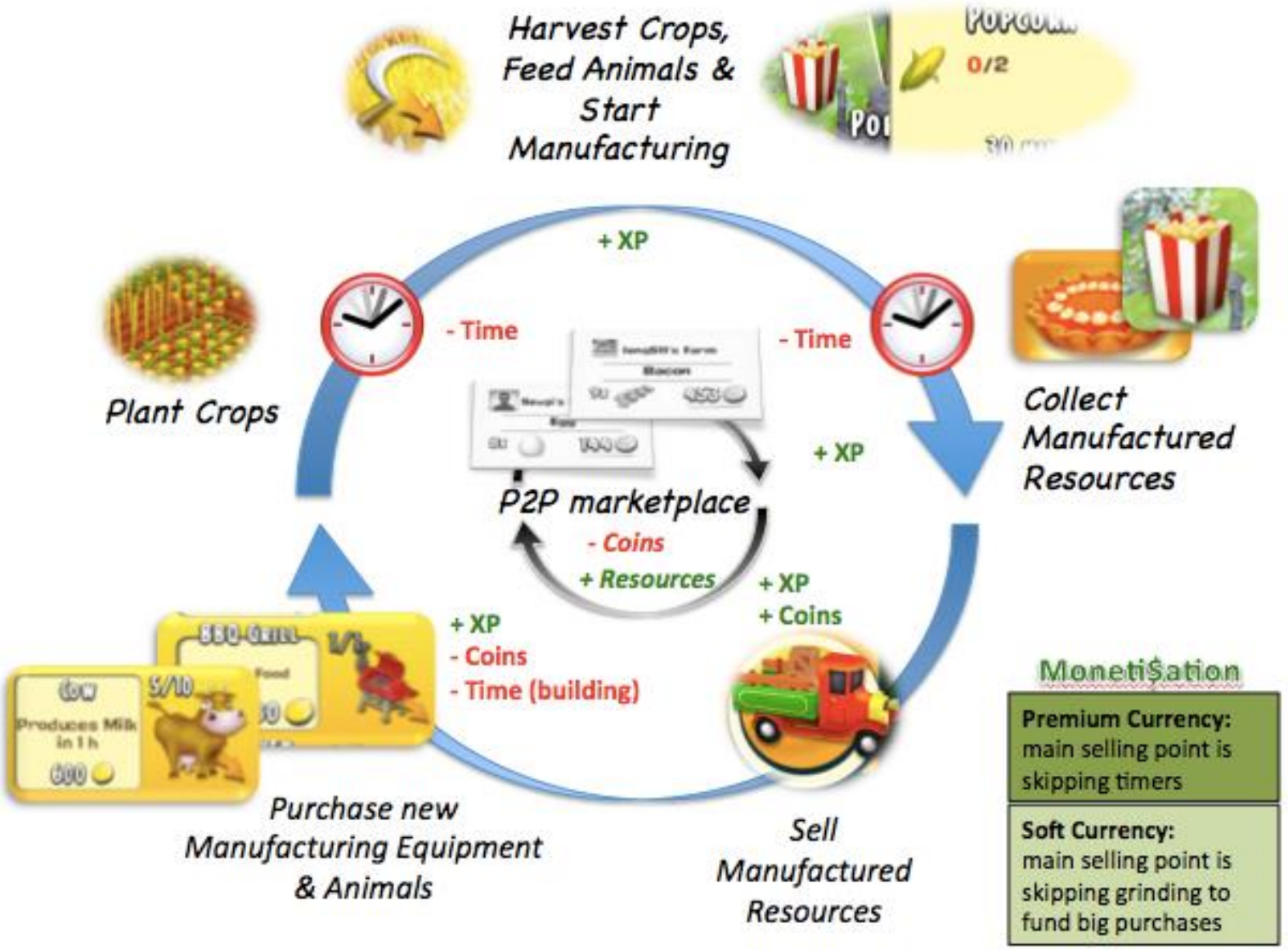


Key concepts of core mechanics design

1. Components
   - You must document the different components that define how your game works
   - Find out the relationship among them
2. Resources – objects/materials created, updated, used or destroyed by core mechanics
3. Entities
   - Simple Entities
   - Compound Entities
   - Unique Entities
4. Attributes of entities

Resources 1/2

- Types of objects or materials the game can create, move, exchange, or destroy
  - The game handles them as numeric quantities
  - May be also
    - Popularity, resistance – non-physical
    - Water, resistance to poison – not countable
- Usually, do not refer to specific instances of these objects
  - But to a collection of resources


Harvest Crops, Feed Animals & Start Manufacturing
+ XP
- Time
- Time
Plant Crops
Collect Manufactured Resources
+ XP
P2P marketplace
- Coins
+ Resources
+ XP
+ Coins
+ XP
- Coins
- Time (building)
Purchase new Manufacturing Equipment & Animals
Sell Manufactured Resources
MonetiSation
Premium Currency: main selling point is skipping timers
Soft Currency: main selling point is skipping grinding to fund big purchases

Resources 2/2

- Core mechanics define the processes by which the game creates, uses, trades, and destroys resources
- A resource is only a type of thing, not the thing itself
  - Specific instances of resources
    - Can legally be moved from place to place
    - Pass from owner to owner
    - Can come into and go out of the game
  - Try to quantify them as numbers so that you can manipulate them

Entities

- An entity is an particular instance of a resource – e.g., marble blocks in avatar's pocket
- Similar to objects in OOP (Id, State, Behaviour)
- Simple Entities
  - Specified by single value
    - Numeric – like score or health status
    - Symbolic – like state of a traffic light
  - Identify simple entities and define them in core mechanics
- Compound Entities
  - More than one data value to describe an entity, e.g. wind (speed and direction) or avatar (has attributes representing other entities)
  - Each value = an attribute

Types of entities

A Simple Numeric Entity

Points Scored
1,755,245

A Compound Entity with Two Attributes

Wind	
Speed	35
Direction	Northwest

A Compound Entity Containing Another Compound Entity as an Attribute

Avatar	
Name	Voldarok
Money	25
Health	117
Race	Elf
Location	Feasting Hall
Inventory	
# Items	5
Item 1	Sword
Item 2	Shield
Item 3	Food
Item 4	Helmet
Item 5	Armor

Attributes of entities

- Attributes of entities
- An attribute is an entity that belongs to, and therefore helps to describe, another entity
- Defining entities for your Game
  - Find out all entities in the game
  - Define how to keep track of them
  - Define how to represent them through user interface
  - Programmers will use these entities

Mechanics

- Mechanics state
  - the relationships among entities,
  - the events and processes that take place among the resources and entities of the game, and
  - the conditions that trigger events and processes.
- Mechanics describe
  - the overall rules of the game but also
  - the behavior of particular entities, incl. AI of a very smart NPC.
- Operate throughout the game
- Apply only in particular gameplay modes
- Global mechanic
  - governs when the game changes from mode to mode (while entities record what modes it is in)

Relationships among entities

- The value of one entity depends on the value of another entity
  - They have a relationship
  - Define it in your core-mechanics
- Types:
  - Numeric relationships - express mathematically, e.g.

  points = collected golden bullions / available golden bullions * 100
  - Symbolic relationships – e.g.,

  behavior of NPC depending on traffic light colors
- Symbolic relationships are relatively easy to define and their results are easier to predict
- Numeric relationships are harder to create and tune.

Events, processes, and conditions

- Event - a specific change that happens once when triggered by a *condition* and doesn't happen again until triggered again.
- Process - refers to a sequence of activities that, once initiated, continues until stopped.
- Conditions
  - To define what causes
    - an event to occur
    - a process to start or stop
  - Conditional statements - often take the form

if (condition) then (execute an event, or start or stop a process);

whenever (condition) take action to (execute an event, or start or stop a process);

continue (a process) until (condition).

  - Mechanics defining victory and loss conditions conform to this style

Internal economy 1/3

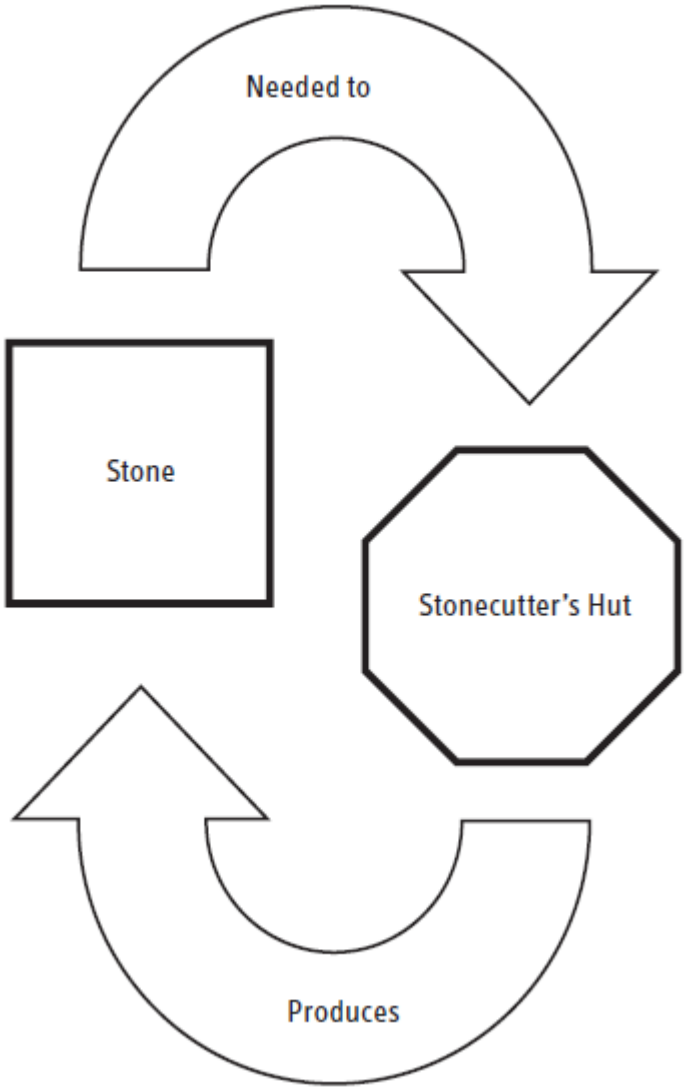
- An economy is a system in which resources and entities are produced, consumed, and exchanged in quantifiable amounts
- Game designers: design and tune the game's economy
- Components
  - Sources – e.g., *spawn points* for insertion of new resources
  - Drains - mechanics determining consumption of resources, e.g. *decay:*
    - *"Each section of road includes a numeric attribute indicating its level of decay as a percentage, with 0 (zero) indicating that the section is new and 100 indicating that the section is fully decayed and impassable. Sections of roads begin to decay 3 months after they are constructed, and 3 percent is added to their level of decay every year, plus an additional 1 percent for every 100,000 car trips over the section in the course of that year. When decay reaches or exceeds 100 percent, the road section becomes impassable and it must be replaced."*

Internal economy 2/3

- Converters – mechanic that turns one or more resources into another type of resource. Example: *The Settlers*
- Traders – such mechanic governs trades of goods, either between the player and the game, or P2P
  - In a stock-trading game, the trader may be a faceless financial construct;
  - In a role-playing game, the trader may be a blacksmith who trades in swords, a NPC, or another player.
- Production mechanisms
  - Tangible resources - possess physical properties within the game world; occupy physical space and have to be transported, e.g., *ammunitions*
  - intangible resources - occupies no physical space and does not have to be transported, e.g. *money*

Internal economy 3/3

- *feedback loops* – e.g. in *The Settlers III*
- *mutual dependencies* - production rate, input to output ratio
- *static equilibrium* – supply and demand for goods are balanced
- *dynamic equilibrium* - conditions are changing all the time, but they always return to the same state after a while because the process is cyclic
- *deadlocks*



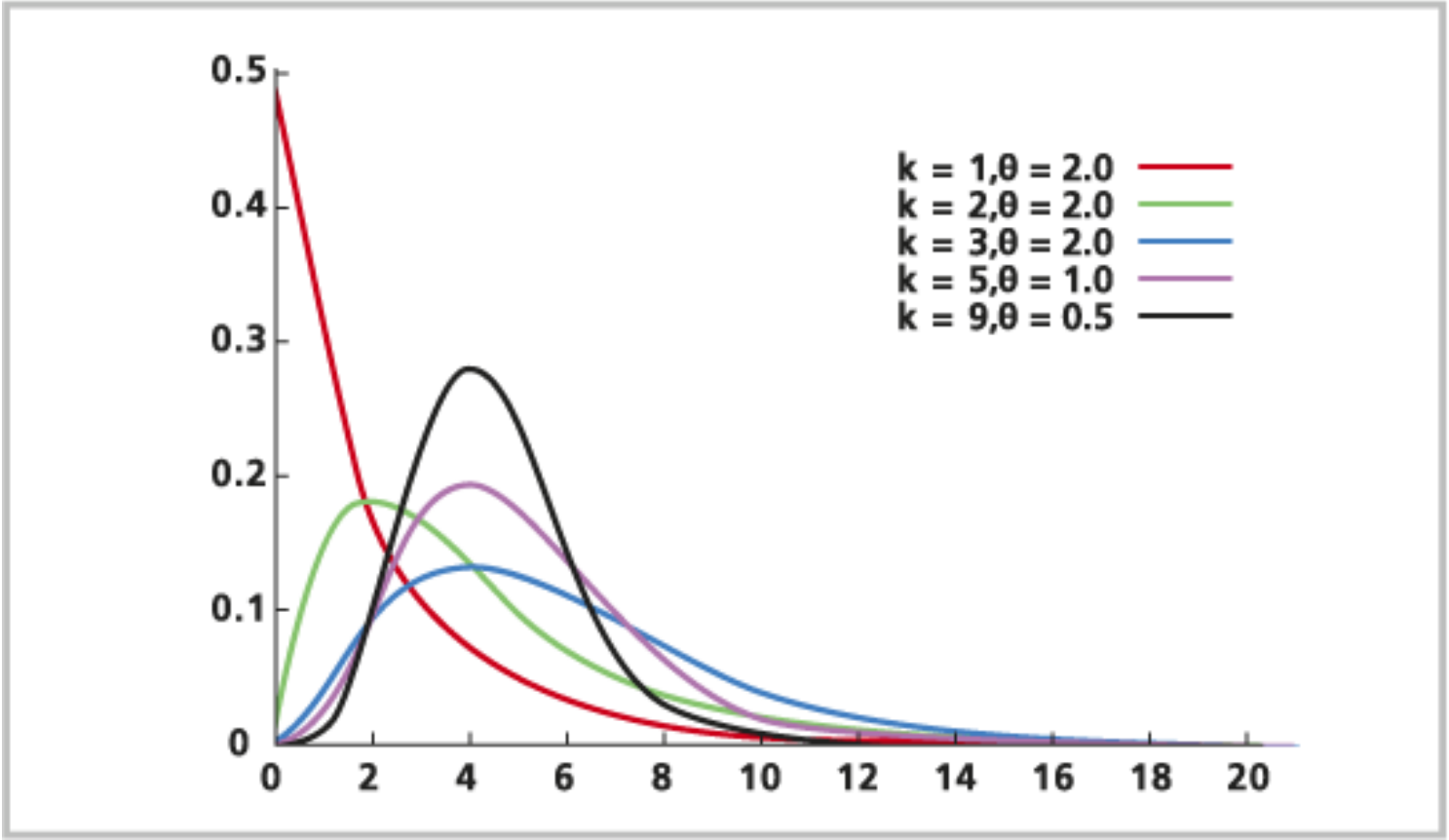
Core mechanics and gameplay

- Challenges and the core mechanics
  - Passive challenges
    - Not presented by core-mechanics – already in the game
    - Can implement the actions, detection, and offer reward
  - Active challenges
    - Offered by the core-mechanic – e.g., a puzzle to open the door
    - Define rules, actions, outcome
- Actions and the core mechanics
  - Player actions trigger mechanics
    - Must specify a mechanic that implements each action in each gameplay mode
    - Initiate an event, start or stop processes – e.g., by pressing a button, UI triggers a mechanic that implements the action, mechanic – change avatar posture
  - Actions accompanied by data
    - Manipulation or storage of data
    - Event mechanic and entity

Random numbers and distributions

- Games use (pseudo) random numbers extensively
- You can use probabilities to
  - Generate events randomly
  - Fix number of enemies or/and entities
  - Set velocity, acceleration, …
- Monte Carlo simulations – e.g., in a game tournament with 20 teams
  - Difficult and time consuming
    - To make people play the game
    - Test if the algorithms resemble fairness or not
  - Rather automate the game playing
    - Select random inputs
    - Analyze the output
    - Think if the output made sense?
    - Did the weak team defeat the strong team? How often?

Distributions



Source: http://desktop.arcgis.com/en/arcmap/10.3/tools/data-management-toolbox/distributions-for-assigning-random-values.htm

Conclusions

- Gameplay
- Core mechanics
- Resources, simple and compound entities
- Events, processes, and conditions
- Internal economy of games


Въпроси