CS 498

Video Game
Design & Development
Methodologies

(For internal use only)

Rational Game Design: In-depth Articles

Gamesutra.com (articles on game design and development)

- **Chris McEntee*, Rational Design: The Core of Rayman Origins***
- **Luke McMillian, *The Rational Design Handbook: An Intro to RLD***

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Rational Game Design

- Seems to have originated in Ubisoft as part of the Prince of Persia/Assassin's Creed creation process
  - Conceived by Lionel Raynaud (Ubisoft worldwide content director) & Eric Couzian (Ubisoft game design conception director)
- Ubisoft developed an internal Design Academy led by Olivier Palmieri (level design director on *Rayman Origins*),
  - Used to train their designers in the field of rational game design.
- Ubisoft recently created an online course in partnership with Concordia University and Knowledge One

What is Rational Level Design (RLD)?

Rational level design (RLD) objectively quantifies elements of user experience to create a consistent game play experience.

- Game elements are modified and created based on observation of user data and informed approximations created by mathematical regression methods.
- RLD is most commonly used to understand how various game elements impact on difficulty.
- Difficulty plays such a significant role in determining user experience (a precarious balance between rage and boredom!)
- RLD uses an objective, number driven system to craft user experience.

Although RLD is now used to create much than game levels, the RLD tag has stuck and is now used to describe design activities using this data driven approach.

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Focus:

Atomic Design: a very low-level game design process wherein the designer examines the small influential factors in the Mechanics and finds clear ways to harness their power in the pursuit of creating a learnable, balanced, fun and exciting experience.

Key Elements:

Player Skills: The physical, social, and mental skills of the player

Mechanic: a challenge for the player based on a specific input and skill which can be altered by Atomic Parameters to increase the inherent difficulty of the challenge.

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Atomic Design

Atomic Design is built around analyzing a Mechanic by breaking it down into its components (methods to come):

- Inputs
- Skills
- Atomic Parameters

We can start then combine raw inputs to build new Mechanics from scratch.

By building Mechanics in this way, a designer can:

- More easily control the inherent difficulty to execute the Mechanic
- Be better prepared for level design
- More effectively define the Game System

Based on what defines a **Game System (***A Game System refers to the balanced relationship between all the Gameplay and Mechanics of a game; the Game System is the game as a whole)* you end up with a bottom-up approach to Game System design.

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Player Skills

Physical skills: Some of the most common skills challenged in games; they are often based on:

- Timing
- Reactions
- Precision, etc.

Social skills: Challenges test the player's ability to communicate and work together with other players. Include:

- Cooperation with another player
- Leadership of a group toward a common goal
- Communication between players to either co-ordinate an action or debate on what to do next (negotiation)

Mental skills: Require logic, memorization and association. Not limited to puzzles, mental skills include:

- Management
- Tactics and Strategy

In deciding which skills we want to challenge and to what degree, we must break down the mechanic into its inputs and the atomic parameters involved. This is done via a matrix of inputs vs. skills.

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Atomic Parameters

Every Mechanic contains at least one Atomic Parameter which upon alteration will influence the amount of challenge associated with the Mechanic at that particular moment in the game.

One Mechanic can share multiple Atomic Parameters

- Each has their own weight of significance on the challenge
- Multiple Atomic Parameters cannot relate to the same aspect as the others

To fully explore the use of an Atomic Parameter it is useful to study five values related to difficulty

- No influence on difficulty (written as ˩)
- Easy
- Normal
- Hard
- Impossible (written as ∞)

- Exploring the non-difficulty case (J) and the impossible case (∞), we can more easily understand how a situation is affected when this parameter is either in full effect or no effect.

- Example: If an enemy's scale fills the entire screen, then the Atomic Parameter for accuracy when shooting is ˩, simply because there is no possibility of missing. What Mechanic could you use to compensate for this?

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Potential Answers:

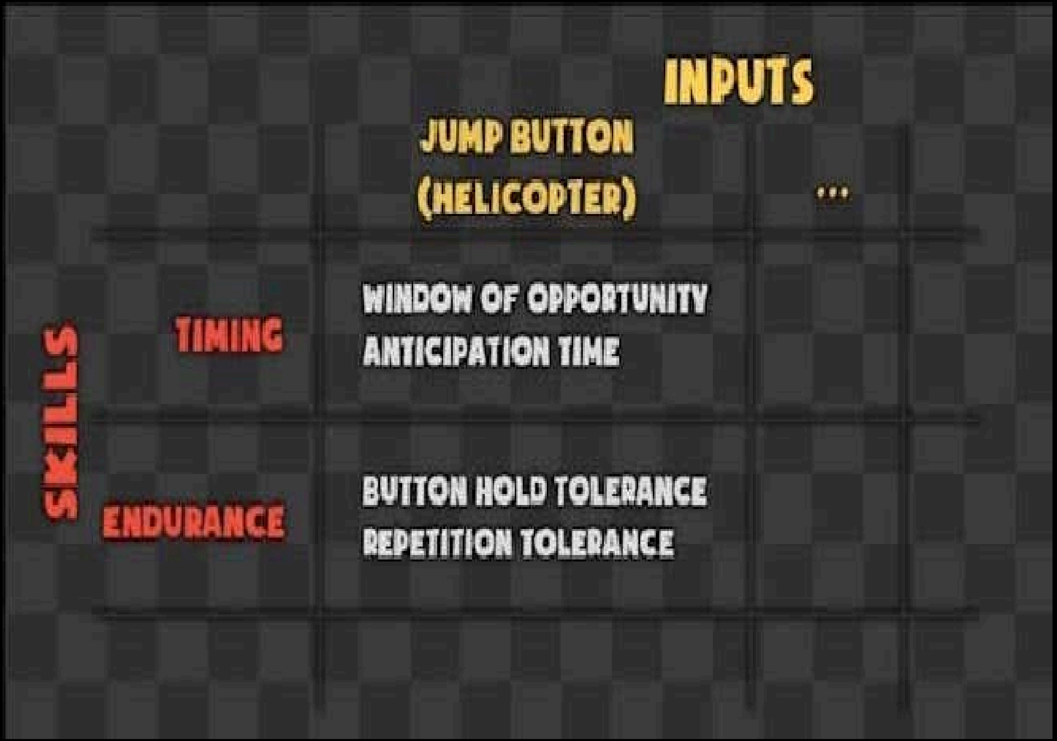
- Require the Player to hit a specific target on the Enemy
- Require splash damage over specific parts of the Enemy
- Offer a window of opportunity to attack the enemy

Similarly, when a parameter is in ∞, such as an enemy having infinite health, it forces the player to take an alternate strategy to defeat them.

With these five values, it is also important to quantify the Atomic Parameters significantly, either in terms of metrics, percentages or times; **there is no such thing as a vague Atomic Parameter.**

Mechanics are defined by skills and inputs

- Skill and inputs are influenced by Atomic Parameters
- Atomic Parameters sit on a Matrix of Inputs vs Skills
- Atomic Parameters relate to player skills, not in-game actions
  - Atomic Parameters should always be factorized
  - If the Atomic Parameter includes a form of syntax from the game context to define, then it is no longer a proper Atomic Parameter.



		INPUTS	
		JUMP BUTTON (HELICOPTER)	...
SKILLS	TIMING	WINDOW OF OPPORTUNITY ANTICIPATION TIME	
	ENDURANCE	BUTTON HOLD TOLERANCE REPETITION TOLERANCE	

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Mechanics

A game mechanic is a challenge based on a specific input and skill which can be altered by Atomic Parameters to increase the inherent difficulty of the challenge.

If no challenge is present (initiating dialogue with an NPC or choosing from a menu) then it is not a mechanic, it is an action.

- To successfully define a Mechanic, we must define a skill to associate with it, so that we know what shall be challenged.
- A player skill is not the same as a character's skill or in-game abilities
- Player skills are something separate from the game world entirely, and are based specific actions:
  - Physical
  - Mental
  - Social
- These actions, when translated into proper inputs, allow the player to overcome a challenge.

Mechanics are the critically important tools for developing good gameplay, flow and learning.

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

Validation

Validation is most effective when done in a safe area, blocking the player's progression to force learning.

By keeping the area safe:

- The player is given total freedom from time and danger constraints to explore the situation
- They can test the full range of possible inputs.
- This also gives the player than didn’t learn the mechanic a chance to “catch up”.

It is also good practice to re-validate specific abilities or gameplay elements to refresh the player's memory in preparation for a nearby challenge.

Challenge

For the sake of the Game Flow, the new Mechanic (Challenge) must start as slightly more complex than a simple validation and progressively to more difficult Gameplay.

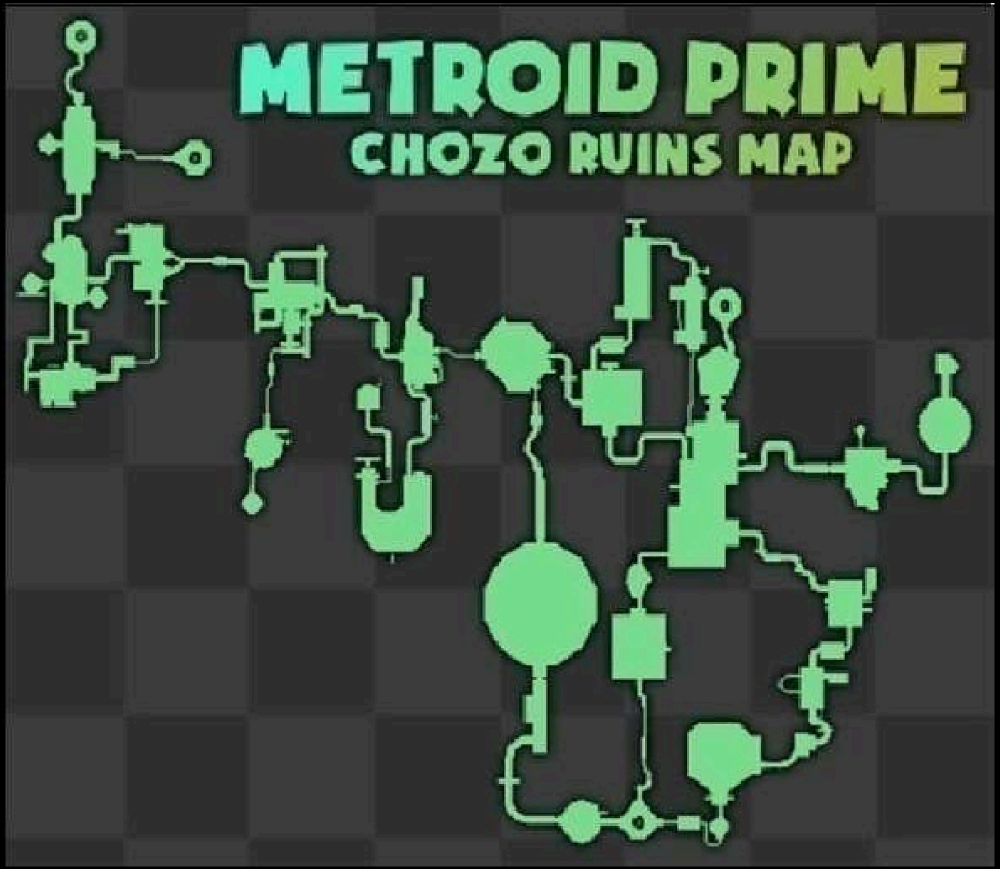
Question: What if the game has a more open world structure rather than a fully linear experience? How is a designer supposed to be able to guarantee that a player has been properly exposed to and taught how to use necessary gameplay elements before a master challenge when he has complete freedom to roam anywhere in the level?

Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

The answer lies in a combination of linear and open world.

- The level structure tapers into short linear sections of gameplay
- Then opens back up into a more expansive environment

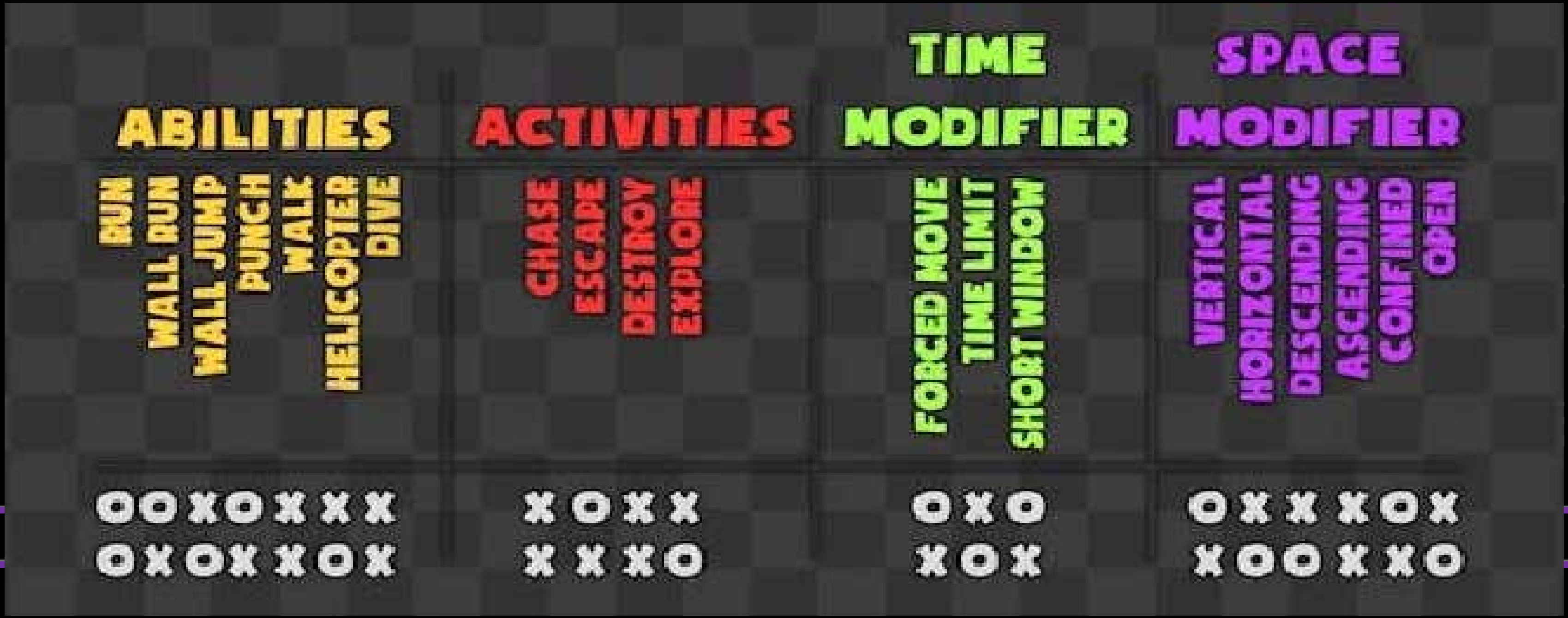
The linear section allowed the designer to teach the necessary skills in a controlled environment before offering the freedom of exploration, movement and choice in the open world.



Rational Game Design: *Rational Design: The Core of Rayman Origins, Chris McEntee*

The Variety Matrix

- A tool for quickly generating gameplay situations that might not be considered due to the illogical combination of abilities.
  - Creates gameplay variety, with combinations normally not considered
  - Could end up being the most memorable sequence in the game
- A variety matrix can be created by listing all the abilities, tools, activities, time and space options
  - Each row consists of x's and o's in a random configuration
- With the result being the possible gameplay variants



ABILITIES							ACTIVITIES				TIME MODIFIER			SPACE MODIFIER					
RUN	WALL RUN	WALL JUMP	PUNCH	WALK	HELICOPTER	DIVE	CHASE	ESCAPE	DESTROY	EXPLORE	FORCED MOVE	TIME LIMIT	SHORT WINDOW	VERTICAL	HORIZONTAL	DESCENDING	ASCENDING	CONFINED	OPEN
O	O	X	O	X	X	X	X	O	X	X	O	X	O	O	X	X	X	O	X
O	X	O	X	X	O	X	X	X	X	O	X	O	X	X	O	O	X	X	O

Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

How do we create the difficulty numbers that fill the Atomic Parameters?

To understand how to create difficulty numbers in RLD process, there are two main issues that we need to think about;

- Mathematically speaking, linear increases in complexity tend to create exponential increases in game difficulty.
  - Think of it as the link between the RLD tables and the psychology of how players interact with the results of the tables.
- RLD is a starting point, there will always be elements of your design which cannot be expressed objectively with numbers.
  - When numbers fail, then look towards chaos to create interesting game level experiences.

Dimensionality

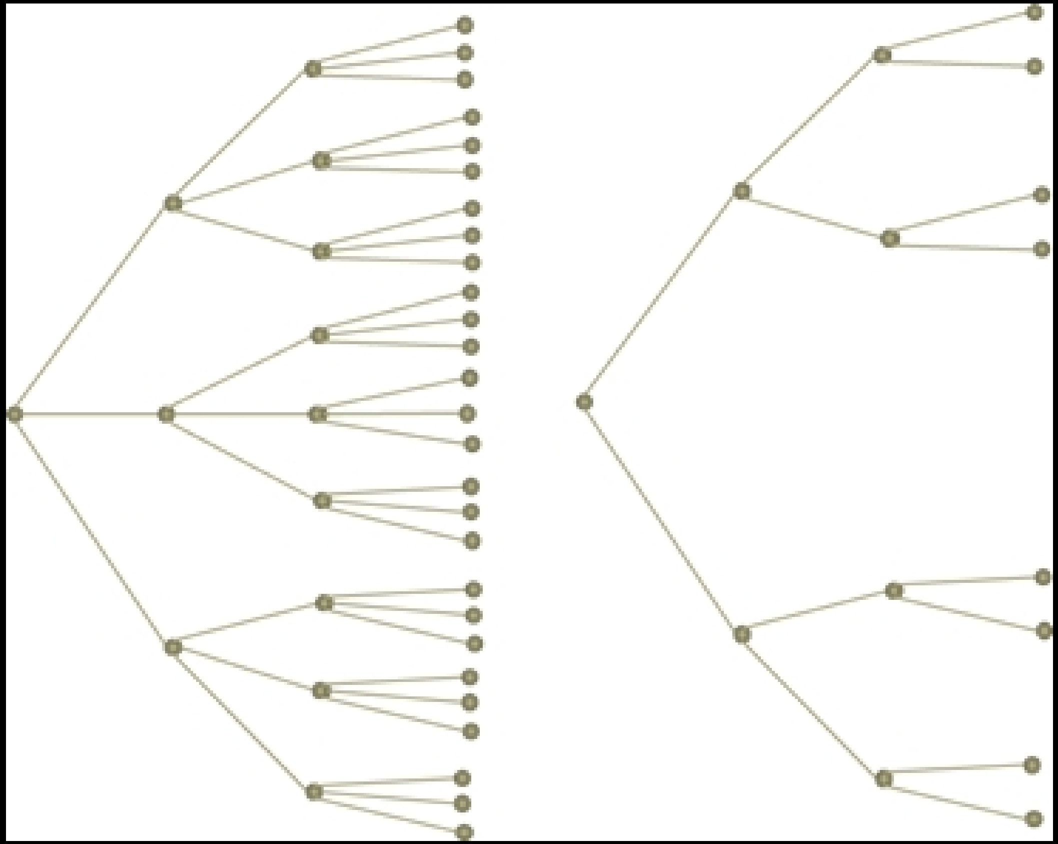
RLD is graph driven and it starts with dimensionality.
Dimensionality is the number of spatial dimensions that we would need in order to enumerate every possible outcome of a set of modifiers.
Example: Dice Game with 1 vs. 2 Dice

Although linear increases in modification lead to exponentially increased probability space, humans are exceptionally capable of dealing with these huge probability spaces.

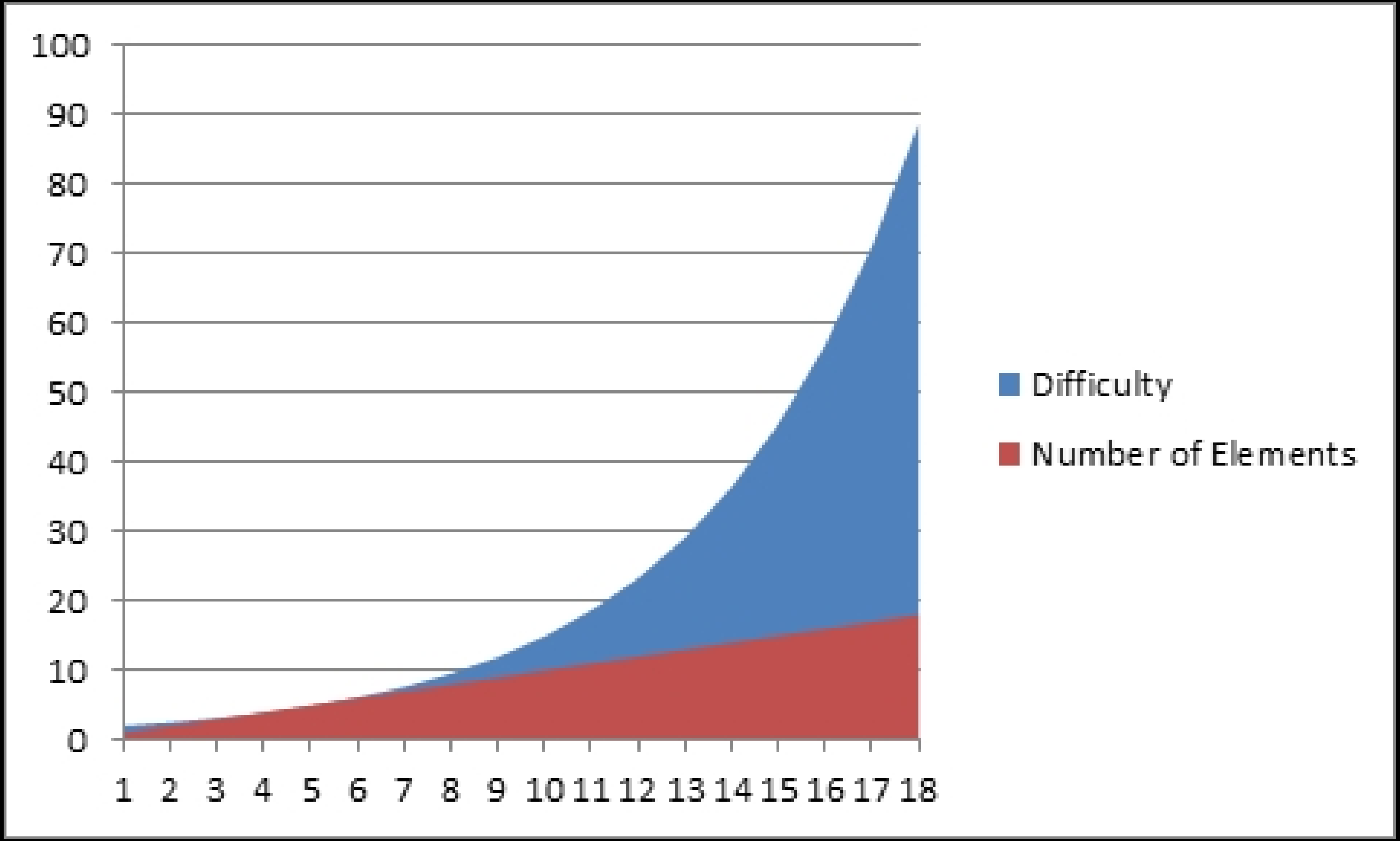
According to many, this is one of the reasons why we find such pleasure in identifying patterns.

Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

We can visualize this concept in a different way by considering the amount of options a player might have at any one point as a branching decision tree. In the image above, we can see how having three options at any one stage (left) is exponentially more complex than only having two (right).



Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD



This frames the RLD problem – whenever we modify or add a single element to a game, the growth rate of difficulty is not linear, but rather exponential to the point where a game will become impossible.

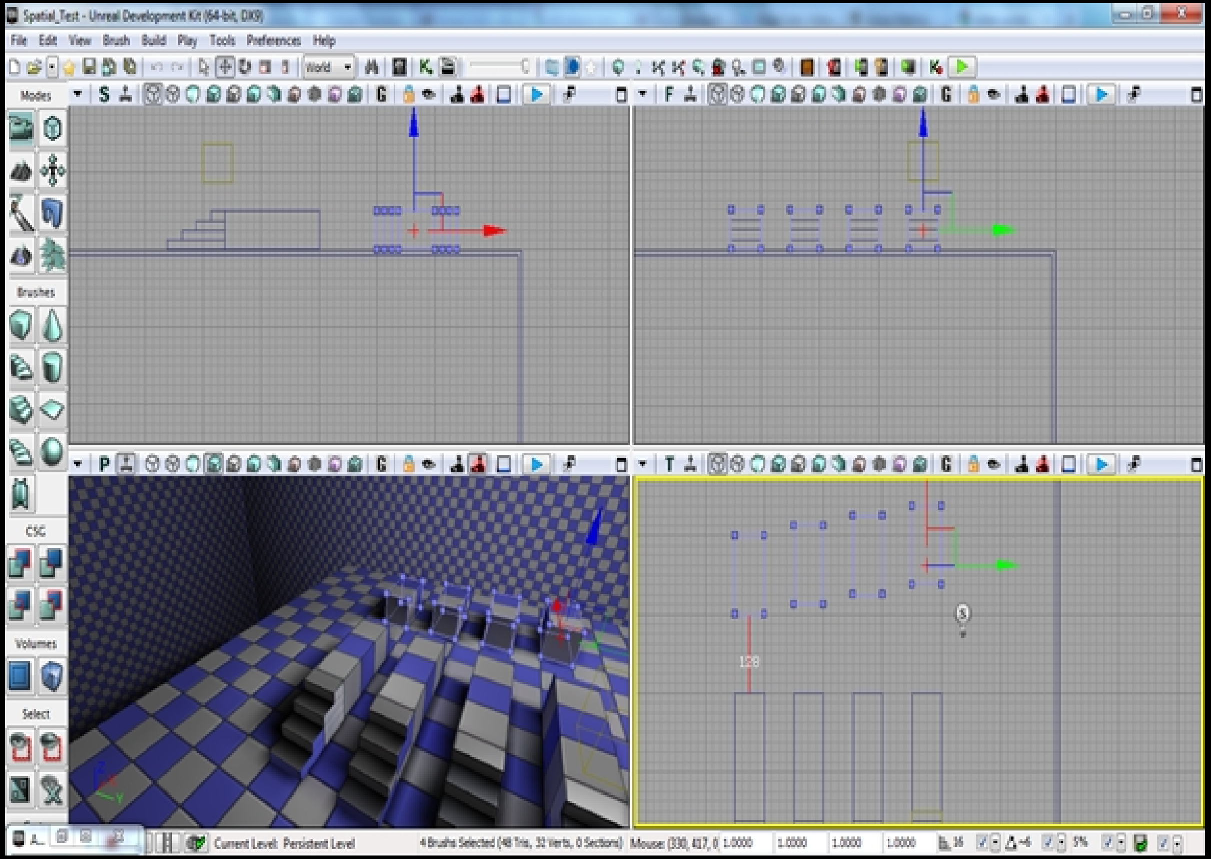
Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

How do you build a rational dataset?

Tools for building out your dataset include:

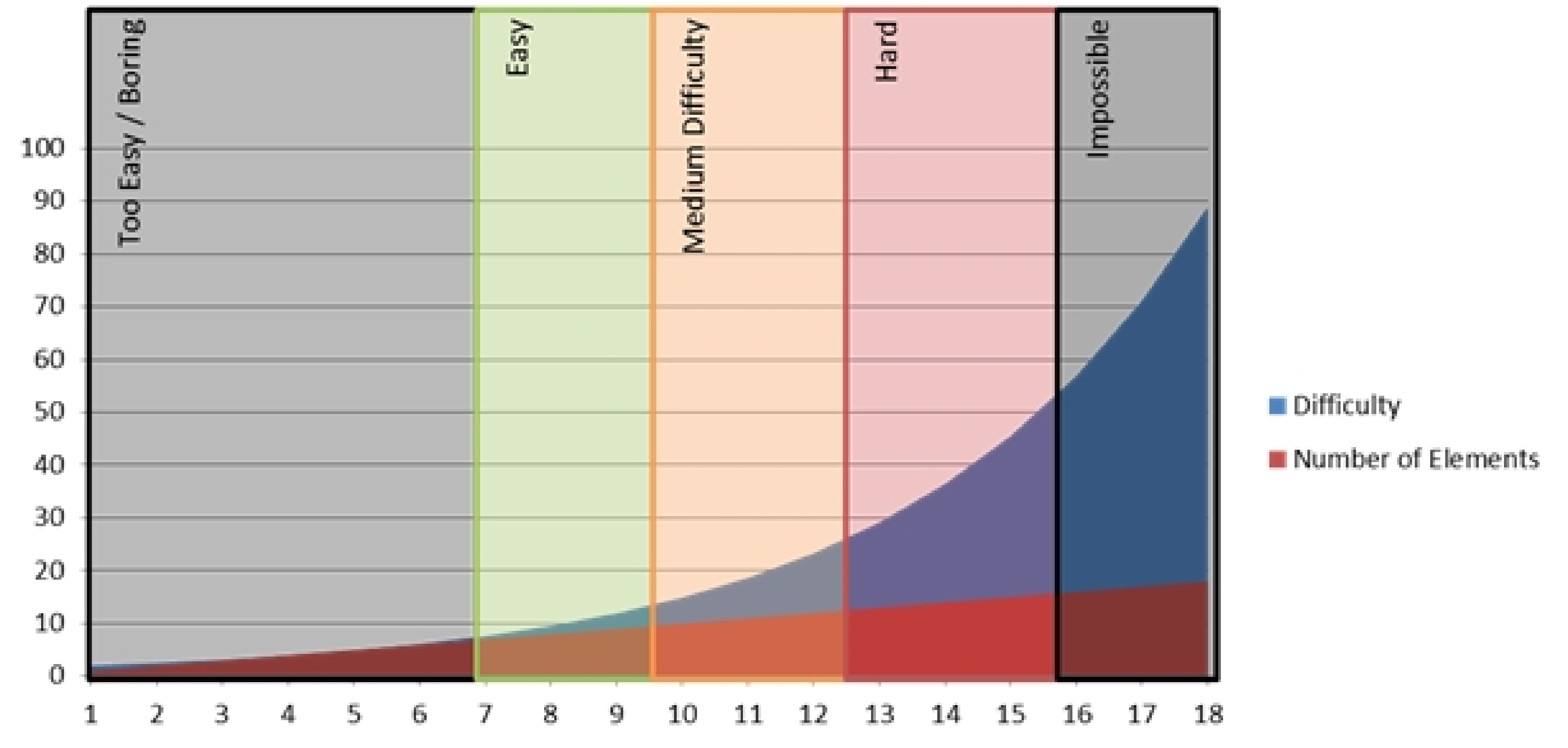
- Setting up a test bed and having testers run through it.
- Graphing the data to determine a rough curve
- Add modifiers
  - Verticality (how far can the player jump up or fall)
  - Number of jumps in a set (this is a bit arbitrary)
- Build out further test beds to quantify more points
  - Various jumps of different heights and distances
  - Related to the initial difficulty sets
- Regress the numbers
- Iterate and refine

- It's a focus on math, testing, and detail


Spatial_Test - Unreal Development Kit (64-bit, DX9)
File Edit View Brush Build Play Tools Preferences Help
Modes
Brushes
CSG
Volumes
Select
128
Current Level: Persistent Level
4 Brushes Selected (48 Tris, 32 Verts, 0 Sections)

Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

Build out your Jump Curve

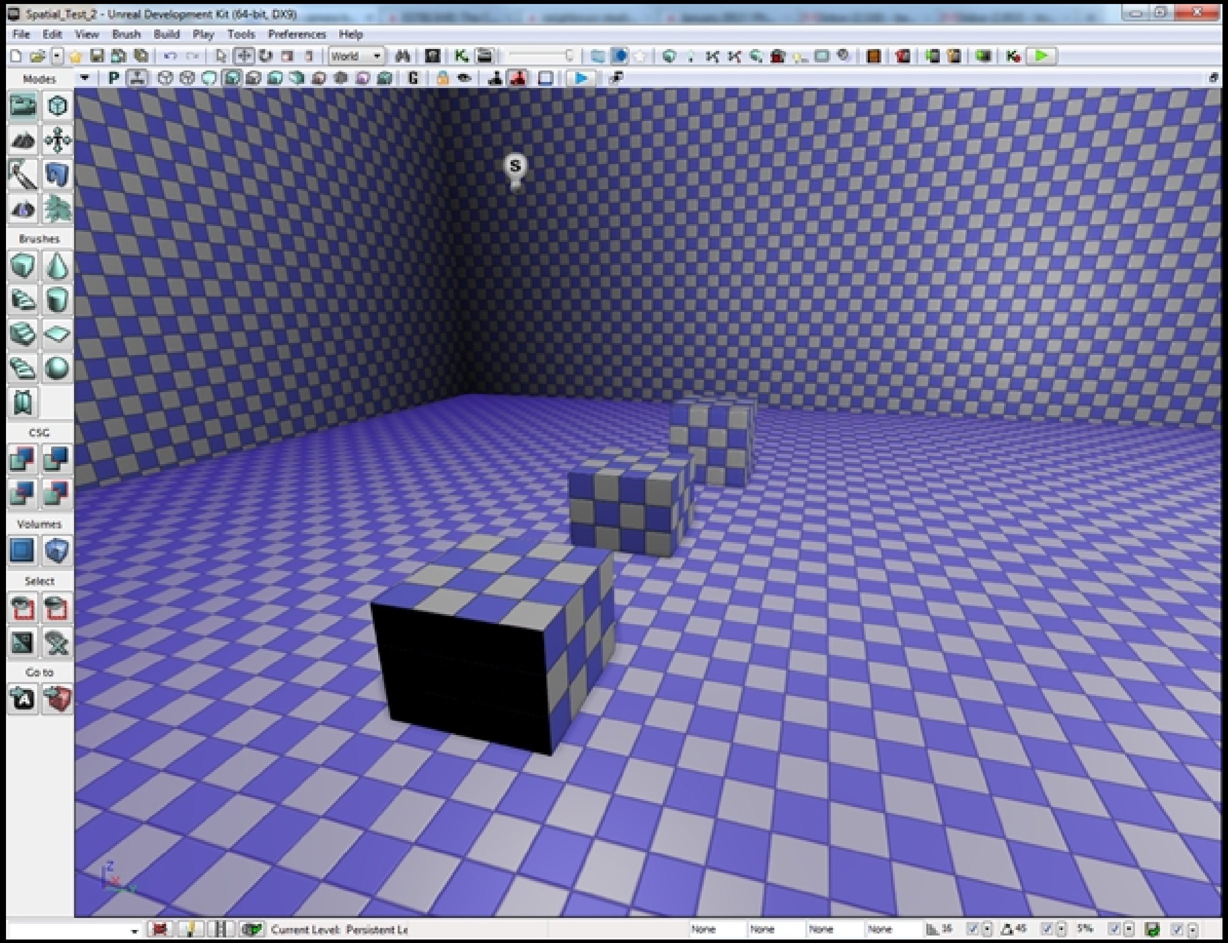


Note that you need some risk even for Easy tasks

Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

Chaos:

Chaos is what happens to a **player's perception of difficulty** when your jumping challenge goes from this:

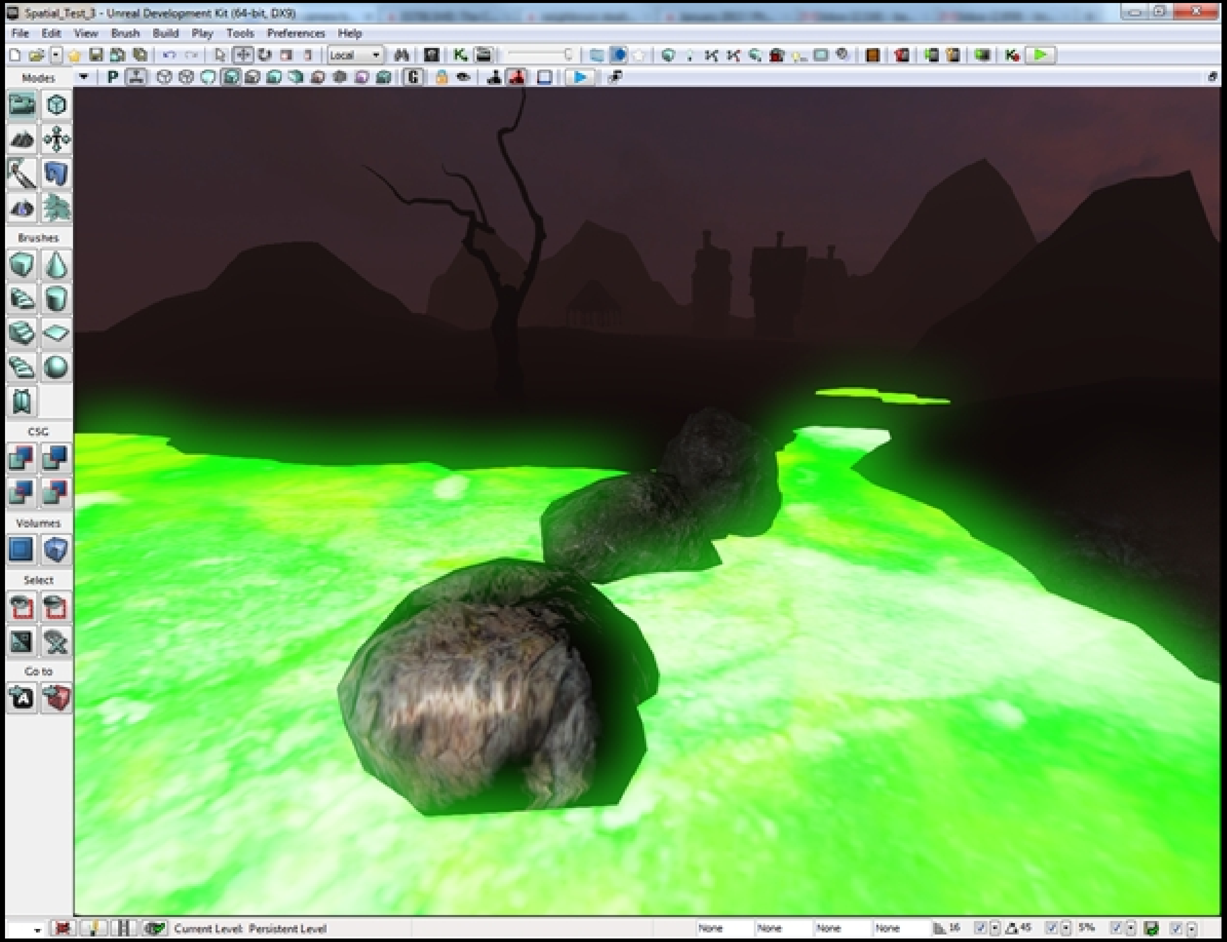

Spatial_Test_2 - Unreal Development Kit (64-bit, DX9)
File Edit View Brush Build Play Tools Preferences Help
World
Modes
Brushes
CSG
Volumes
Select
Go to
Current Level: Persistent Le
None
None
None
None

What the designer sees in a test bed

Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

Chaos:

To this:



What the player sees in game

Rational Level Design: Luke McMillian: The Rational Design Handbook: An Intro to RLD

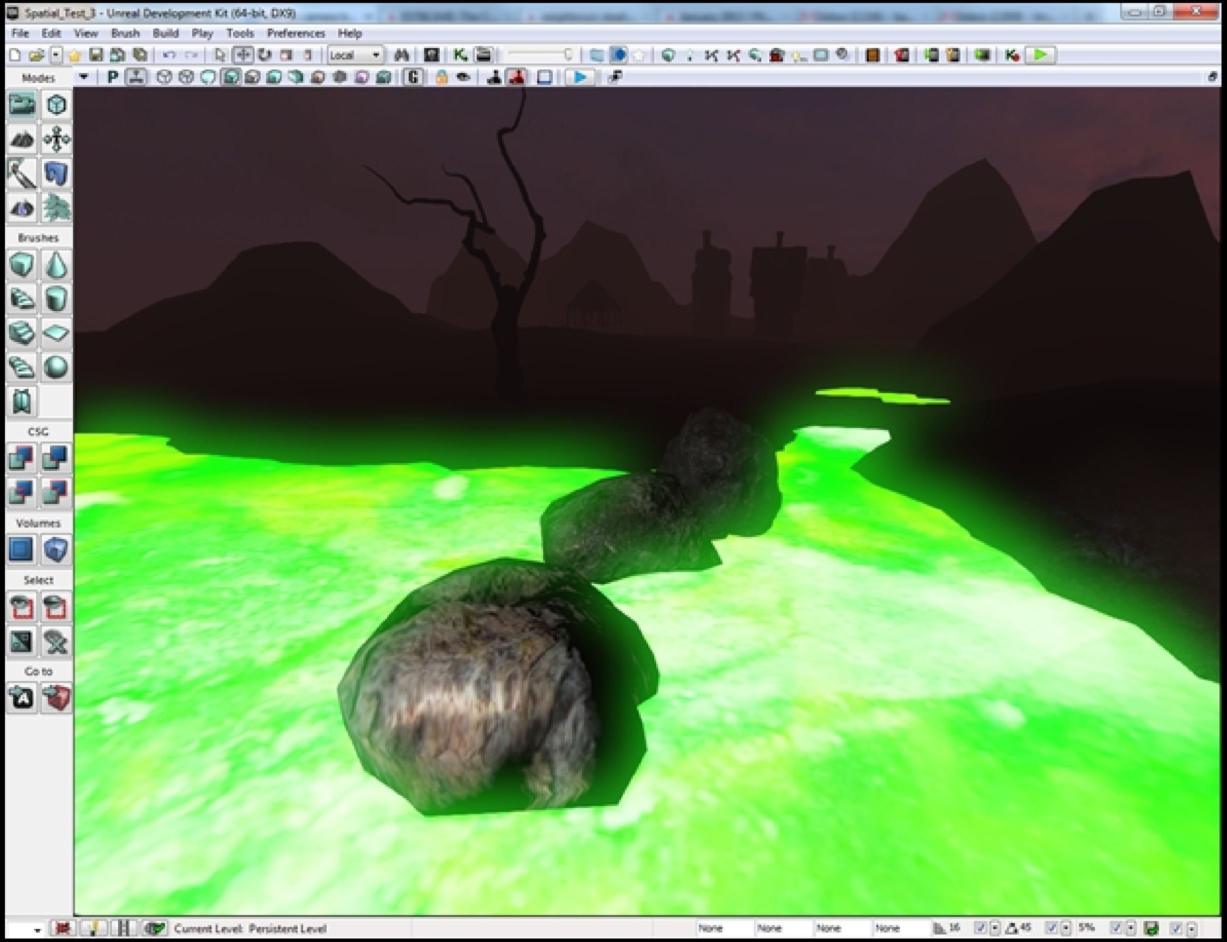
Chaos:

Chaos in the context of RLD is all the modifiers that cannot be expressed in a quantified manner

- Those that are in game
- Those that are in the Player’s mind
- It is the gameplay that is evident in the game play experience

Put another way, chaos is the part of RLD which we do not represent numerically but is nevertheless an integral element of player experience.

Although designers do not quantify these phenomenon, they use them and consider them in the process of RLD.


Spatial_Test_3 - Unreal Development Kit (64-bit, DX9)
File Edit View Brush Build Play Tools Preferences Help
Local
Modes
Brushes
CSG
Volumes
Select
Go to
Current Level: Persistent Level
None
None
None
None

What the player sees in game

Conclusion