Building Non-linear Narratives in HORIZON: ZERO DAWN

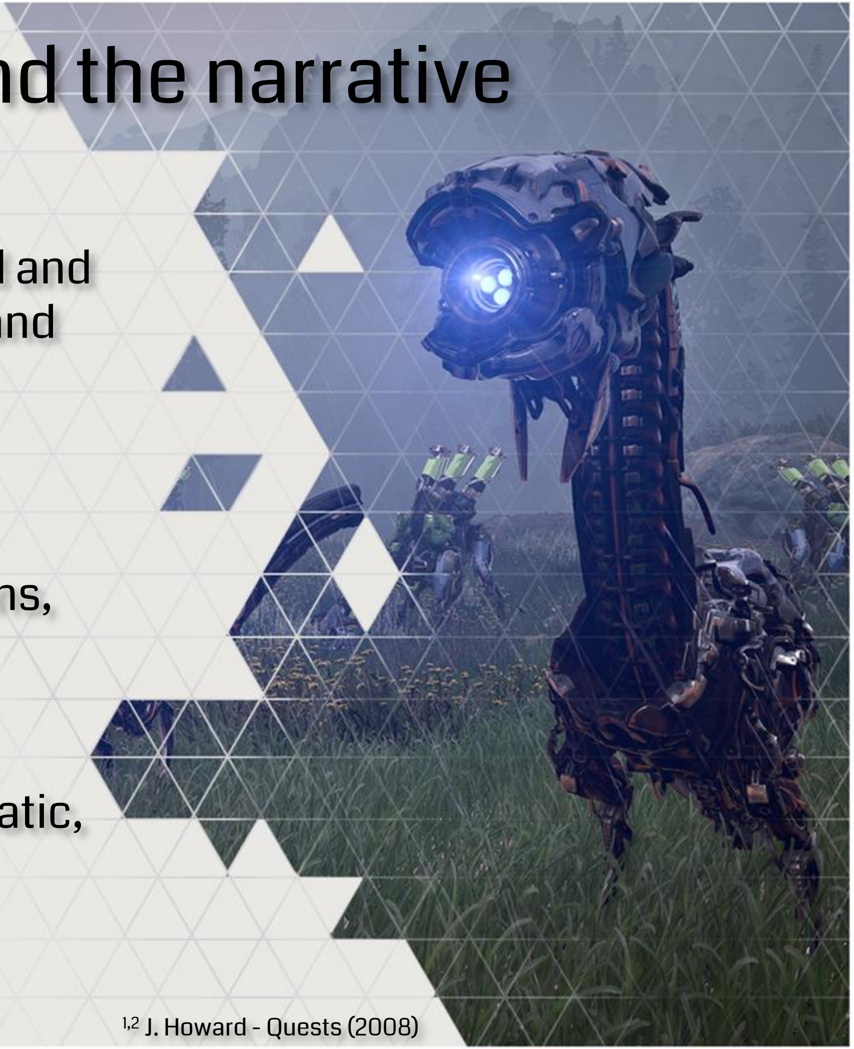
Leszek Szczepański
Gameplay Programmer - Guerrilla Games
@yezu

Quests - between rules and the narrative

Quests are a way to combine narrative focused and rules focused methods of creating, analysing and understanding games.

Quests are about action that is meaningful to a player on the level of ideas, personal ambitions, benefit to society, spiritual authenticity. [1]

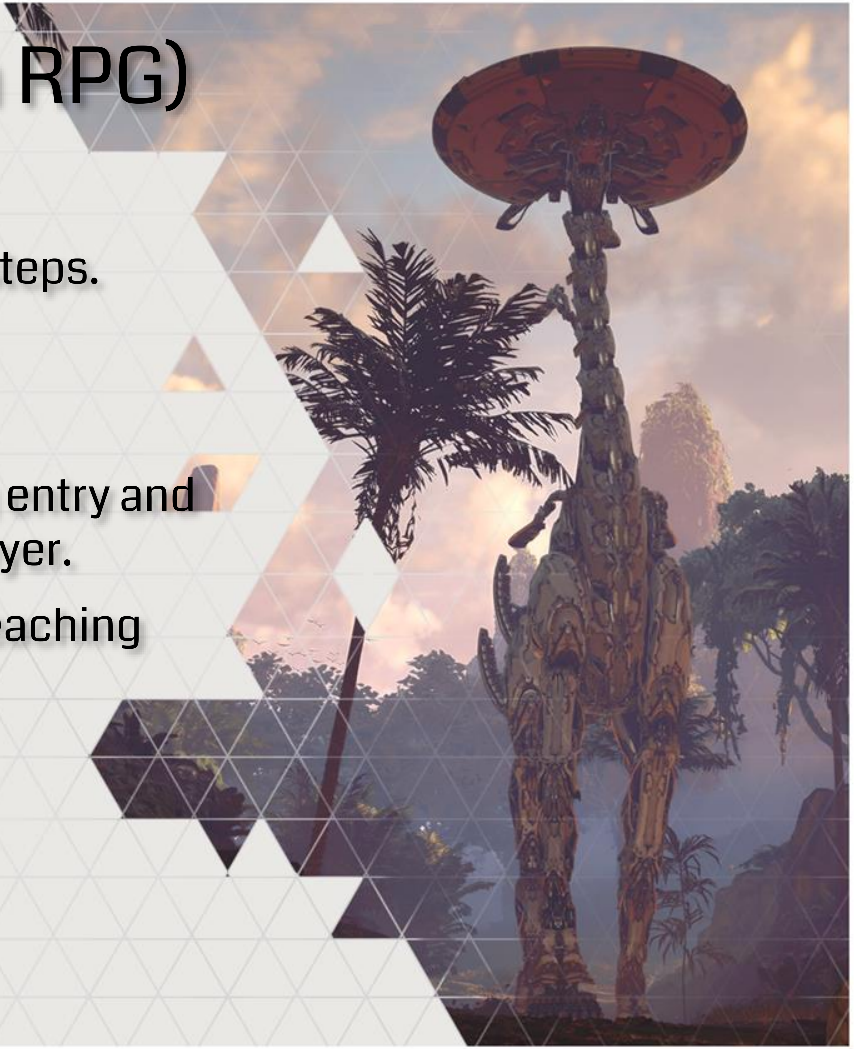
The meaning of quests emerges from strategic actions, but these actions have thematic, narrative and personal implications. [2]



[1,2] J. Howard - Quests (2008)

Structure of a quest (in an RPG)

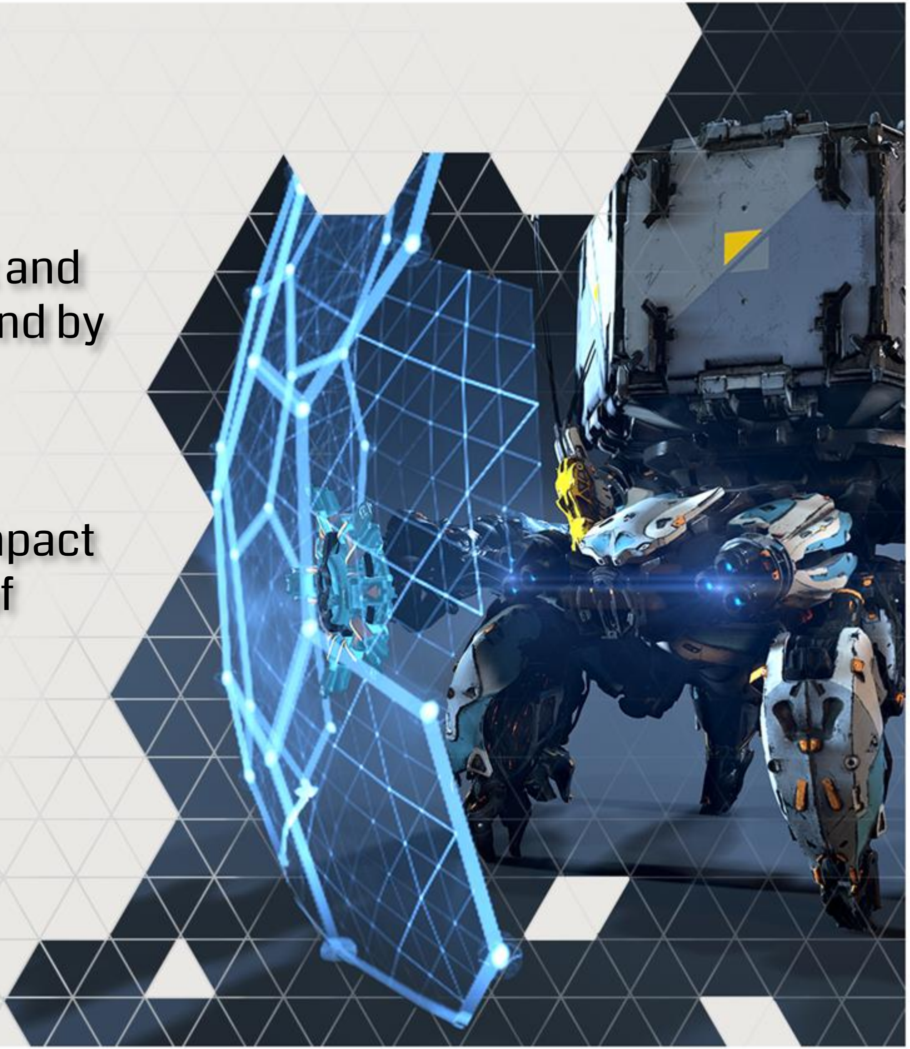
- Composed of multiple discrete phases or steps.
- Has an obvious start, middle and end.
- Each step is a piece of narrative.
- Often a step is connected with a Quest Log entry and one or multiple objectives visible to the player.
- Usually the player is given a reward after reaching specific steps.



Quest Systems

In an RPG a quest system is needed to track and manage player's progress through quests and by turn the game's story.

The shape of the system has a significant impact on how the quests are designed and kinds of stories that can be told.



Quest Systems

When looking at other games we saw multiple different approaches and implementations.

However after a while a certain pattern became visible.

All quest systems could be placed on a scale describing how flexible they are and how strictly they define what a quest is.

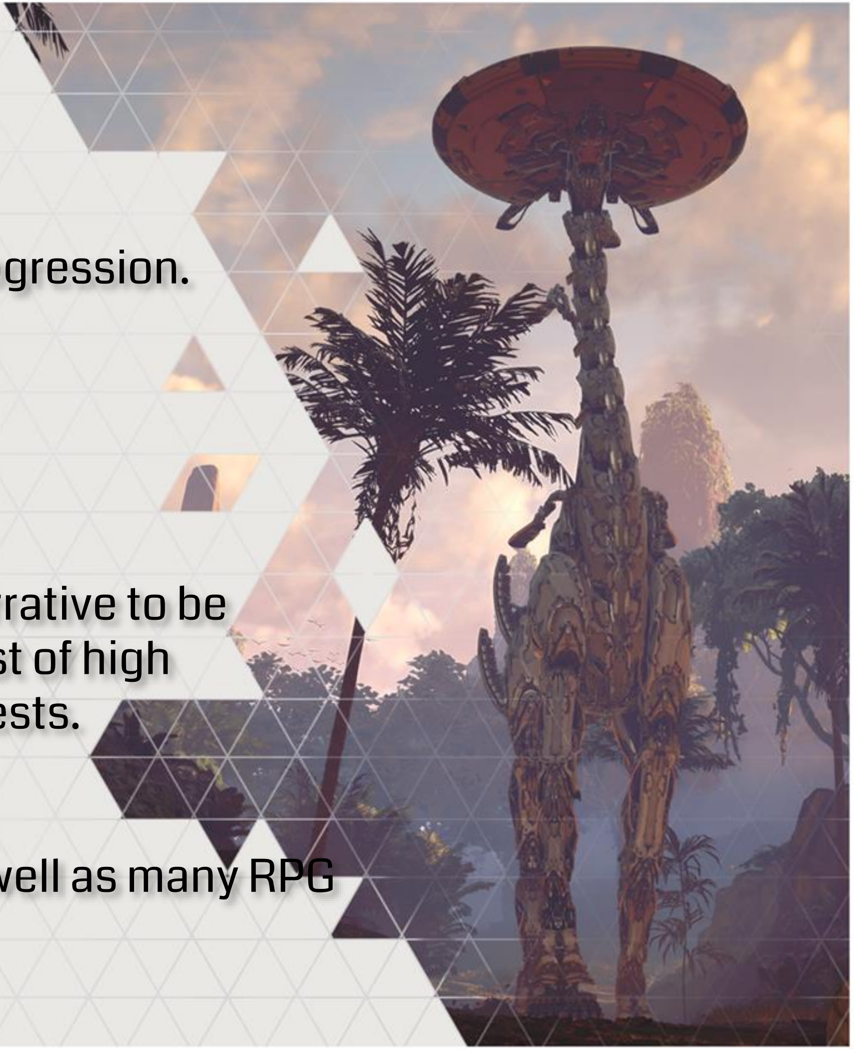
With **Relaxed** and **Strict** systems on opposing poles of the scale.

Relaxed Quest System

- No centralized system managing quest progression.
- No defined concept of a quest.
- All logic fully managed by scripts.

Such systems allow for virtually any kind of narrative to be expressed through a quest. However at the cost of high complexity of implementation of individual quests.

Good examples: Might & Magic, Wizardry and
a lot of other RPGs created before mid 90s as well as many RPG Maker games.

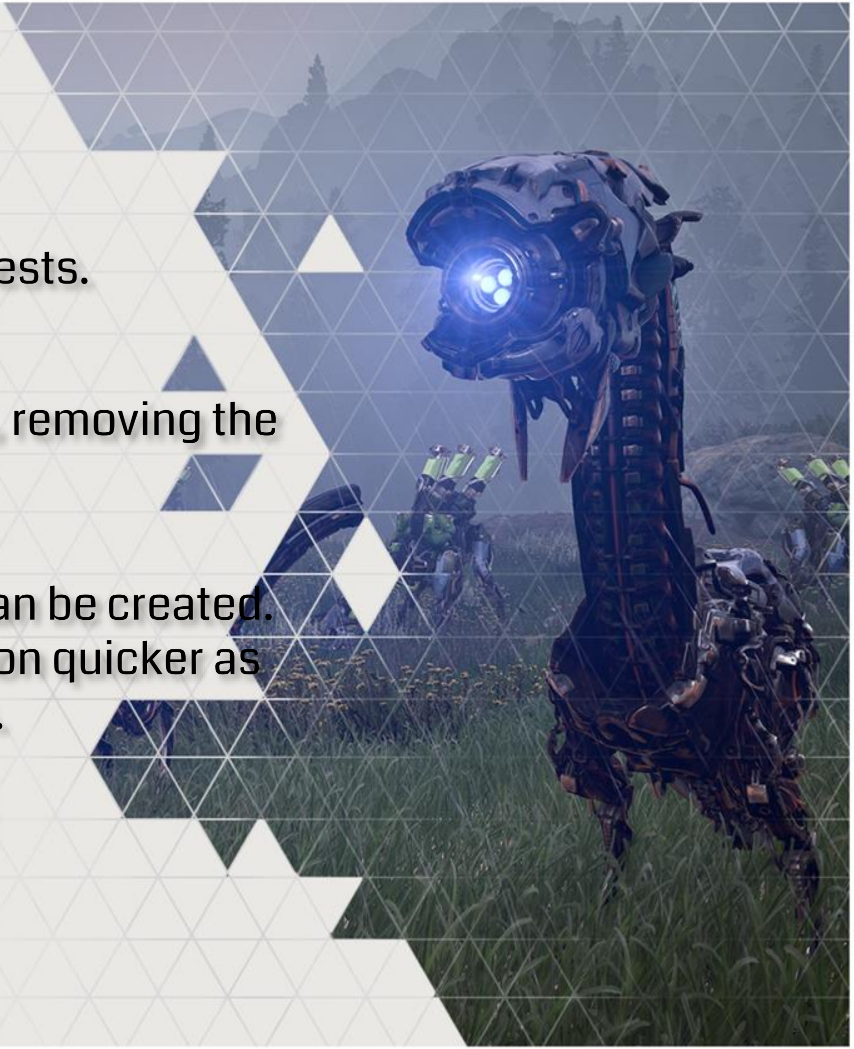


Strict Quest System

- Uses a single, centralized system for all quests.
- Each quest has a defined, rigid structure.
- Quests are built from predefined elements, removing the need for scripting.

Such system severely limit what kind quests can be created. However these limitations make implementation quicker as well improve communications between teams.

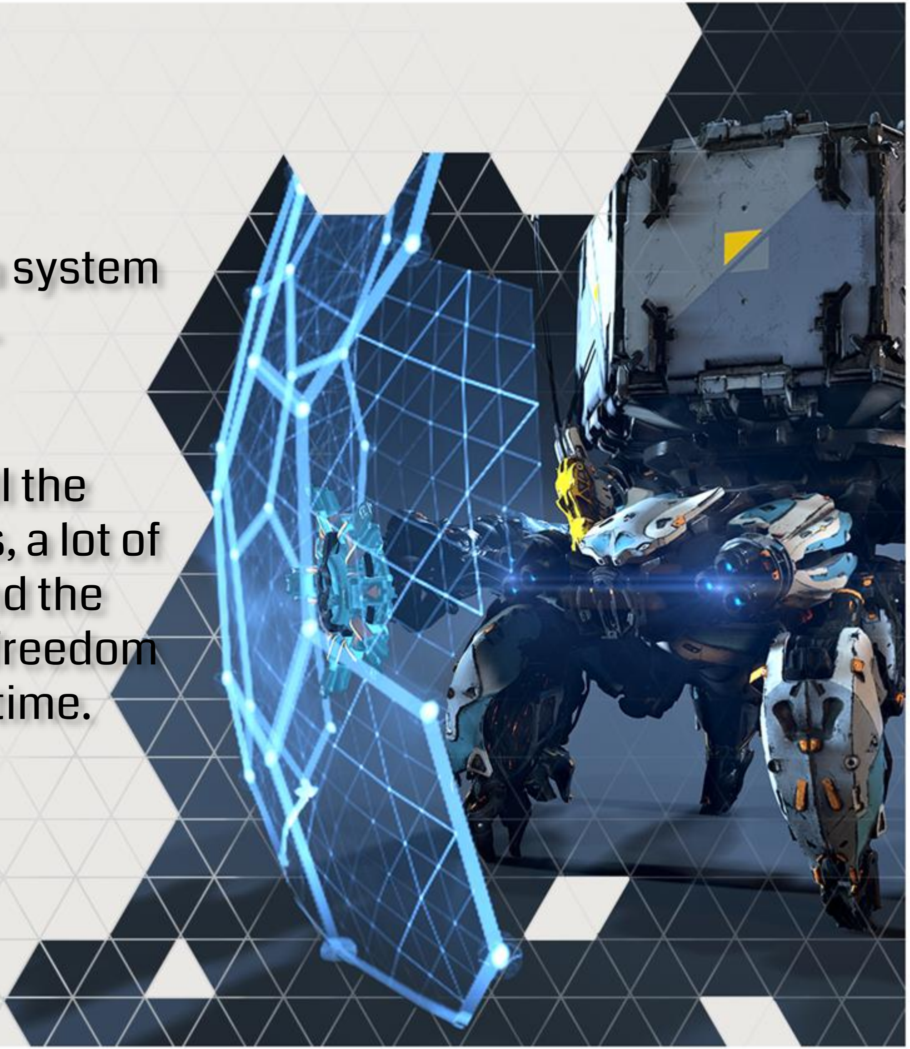
Good examples: Titan Quest, Sacred and other hack’n’slash games.



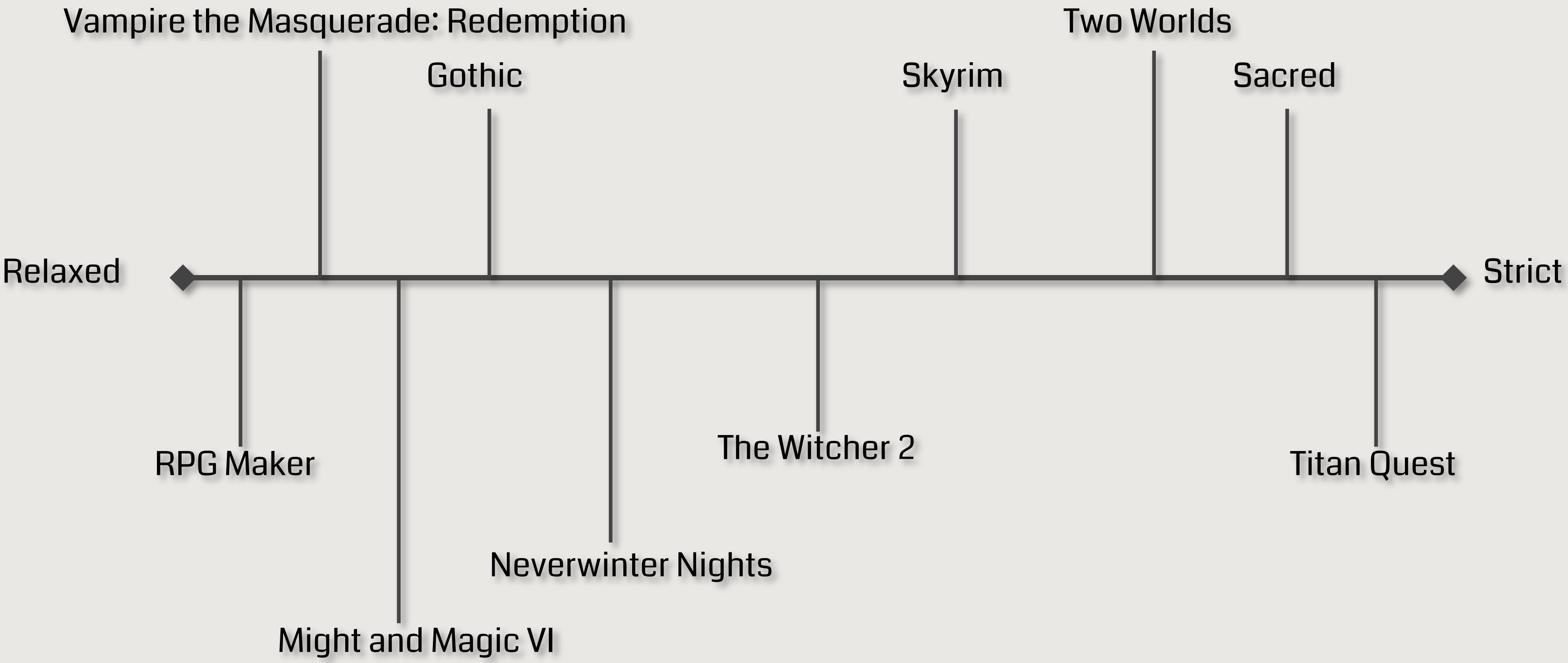
Quest Systems

Nowadays it is quite rare for a game to use a system that lies on any of the extremes of the scale.

As the demand for interesting stories as well the amount of content has grown over the years, a lot of studios try to explore a middle ground, to find the perfect balance between story complexity, freedom of expression, scalability and development time.



Quest Systems



Relaxed

Strict

Vampire the Masquerade: Redemption

Gothic

Skyrim

Two Worlds

Sacred

RPG Maker

Might and Magic VI

Neverwinter Nights

The Witcher 2

Titan Quest

Binding rules with the narrative

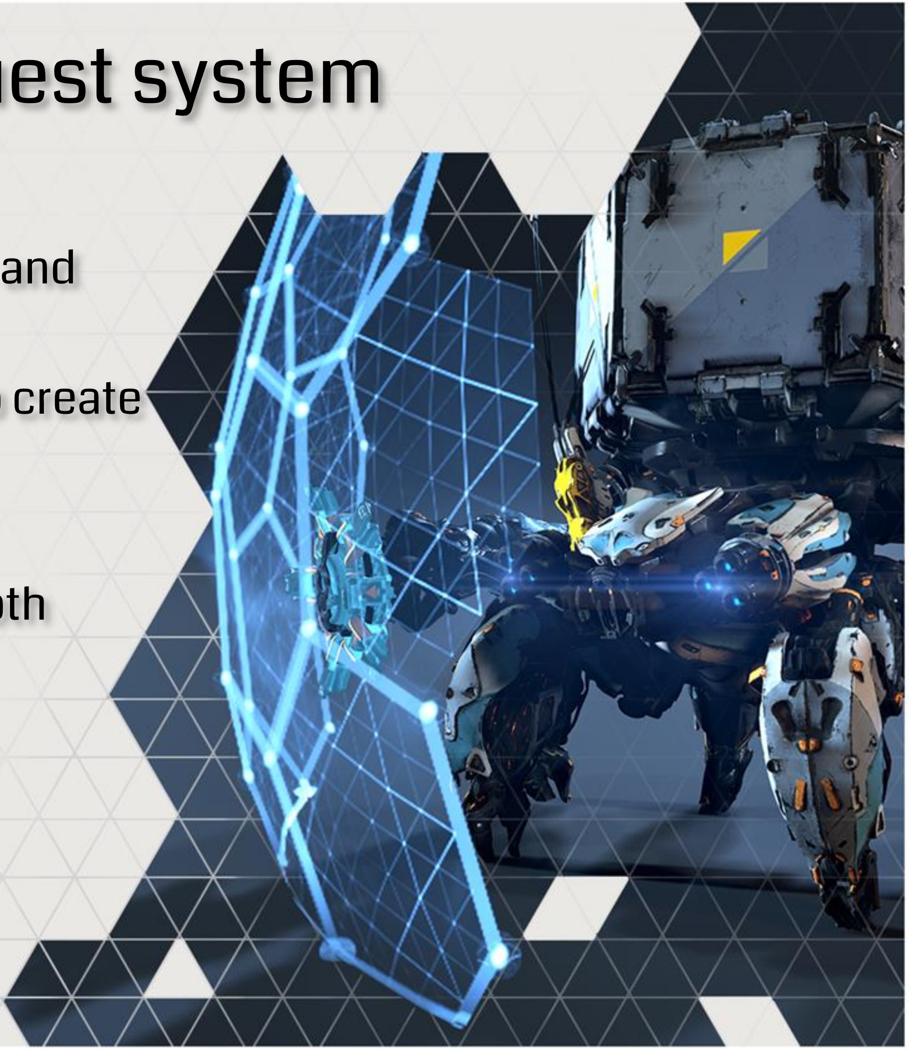
Usually there are two approaches available to merge rules with the story, in order to create a quest:

- Write a story and by injecting mechanics, add gameplay to it.
- Create a set of challenges and wrap a story around them.

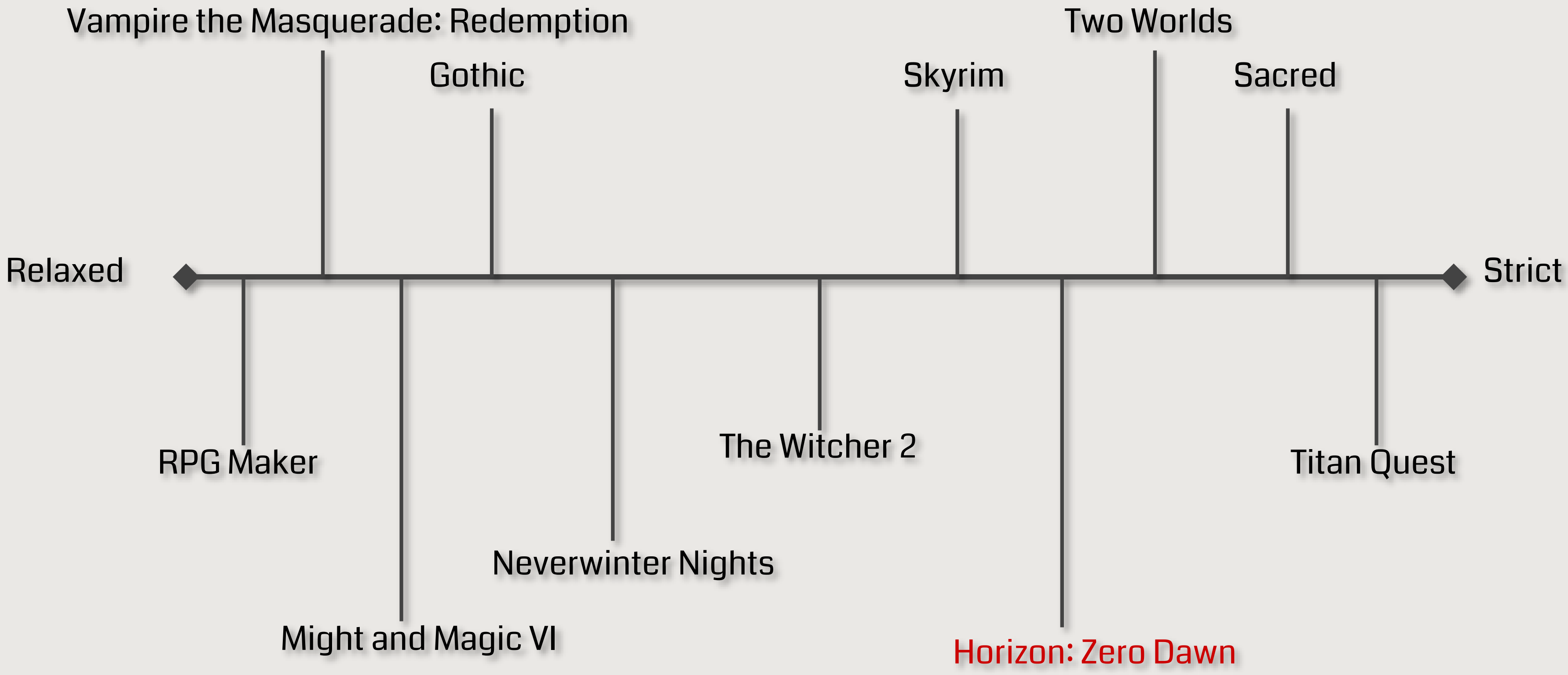
These have been perfected over the years and tend to work well. However, because we are **translating** narrative into logic or vice versa, sometimes cracks and inconsistencies can show.

Requirements for our quest system

- Common types of quests must be quick and easy to create.
- Complex quests must not be complex to create and manage.
- Focused on non-linearity.
- Provide a base common language for both rules and the narrative.

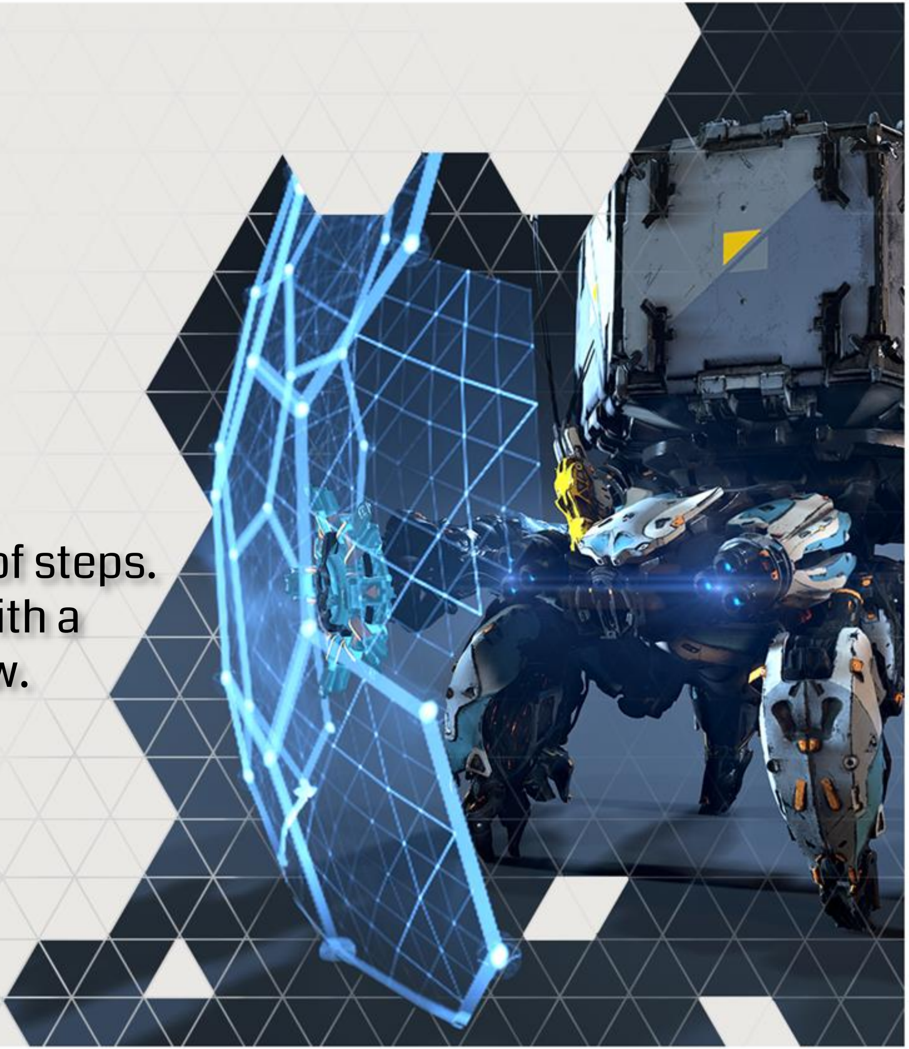


Quest Systems



Our approach

A lot of games, even ones with non-linear narratives, construct quests as sequences of steps. However we wanted to present the player with a **probability space** instead of a script to follow.

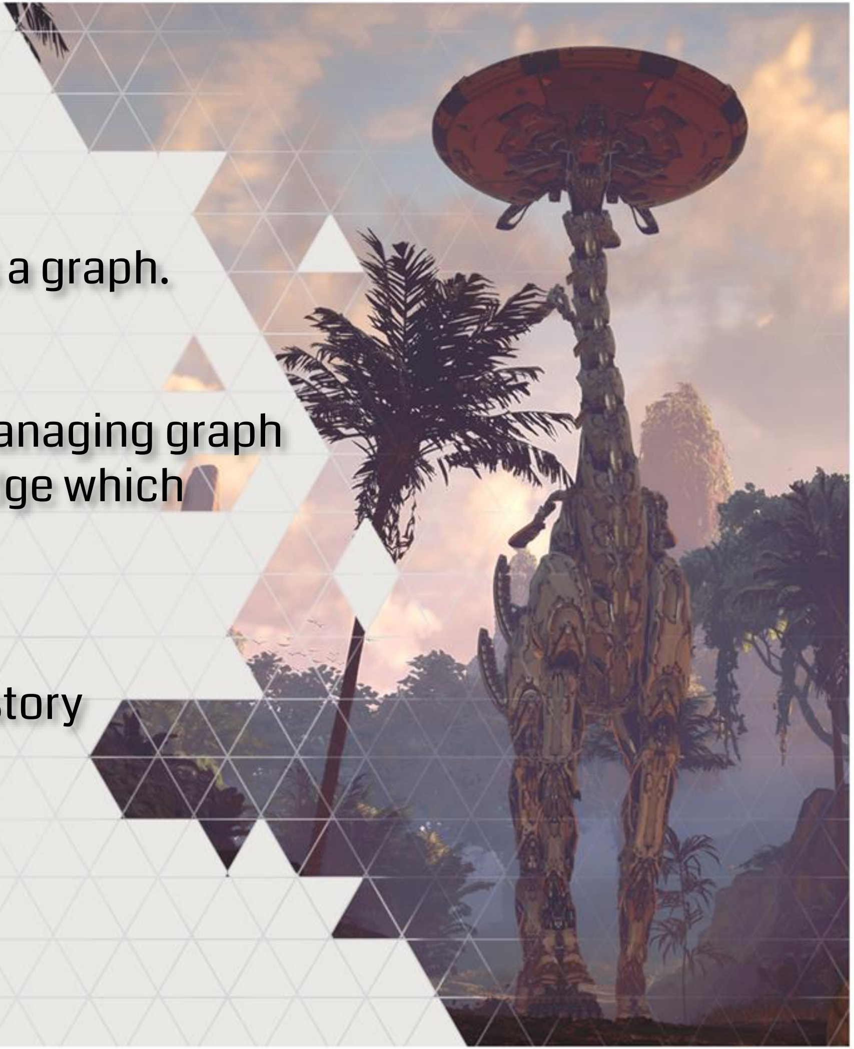


Our approach

A **probability space** can be well represented as a graph.

Graph theory provides many useful tools for managing graph structures, but does require technical knowledge which users of the system might not have.

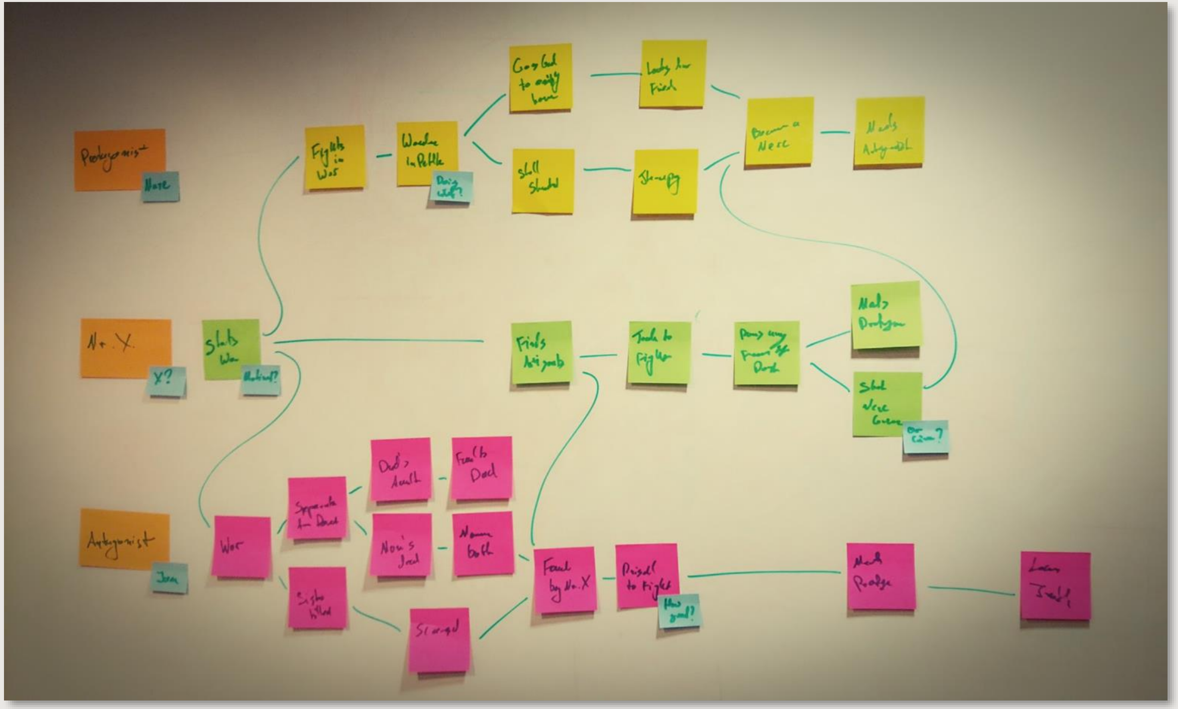
Instead we drew inspiration from the world of story writing and decided to model our system on a technique called **Wall of Sticky Notes.**



Wall of Sticky Notes

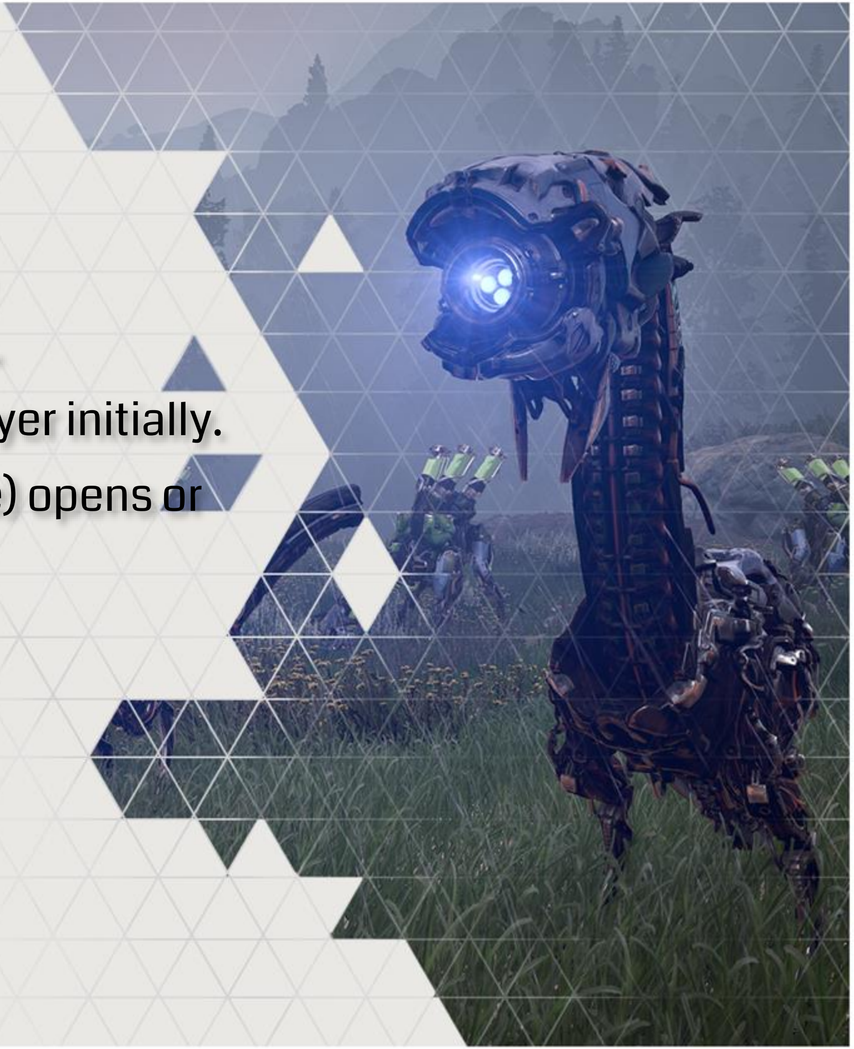
A sticky note represents a plot point or an event in the story.

Lines between notes represent cause and effect between the events.

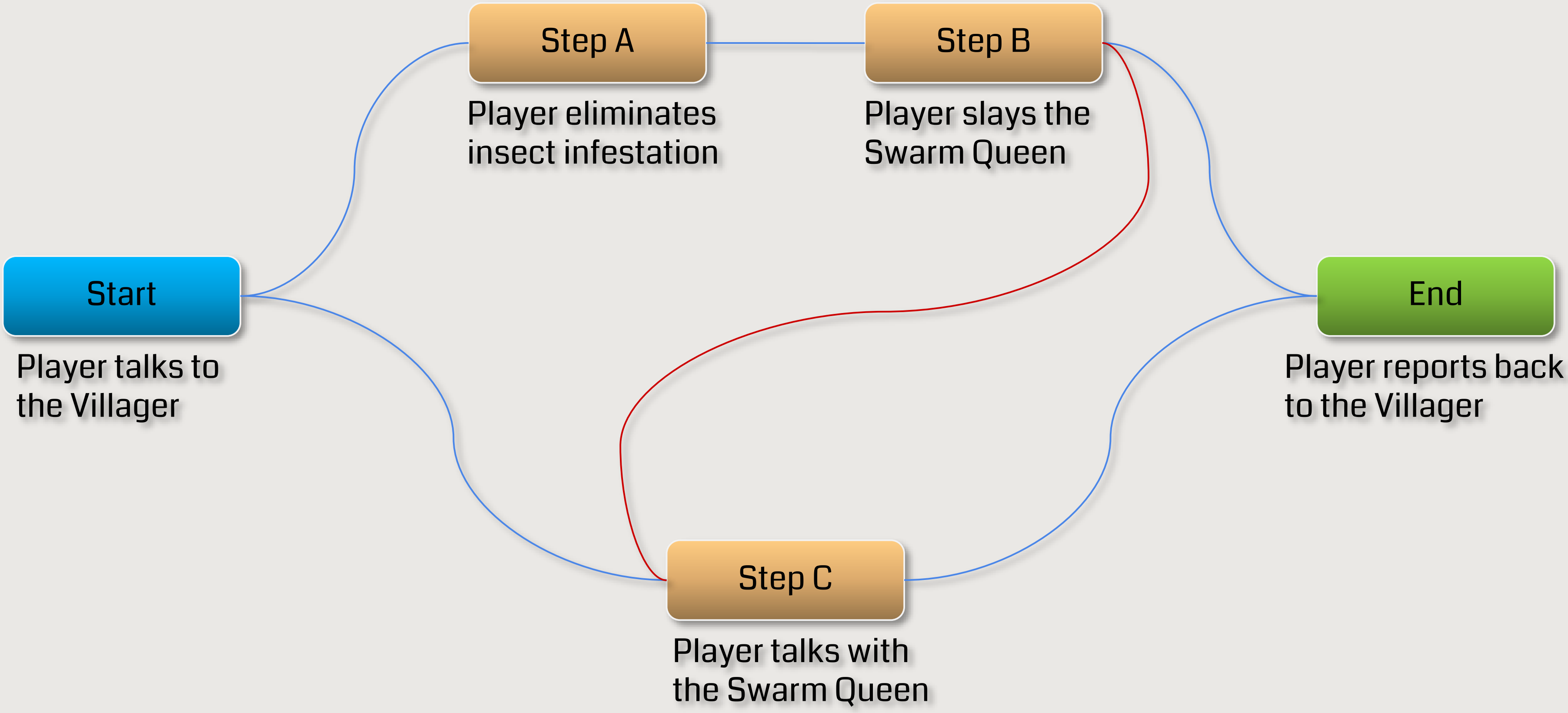


Our approach

- A quest is represented as a **graph** of steps.
- Each step is a scene or an event in the plot.
- Only the start steps are available to the player initially.
- Completing a step (experiencing the scene) opens or closes parts of the graph to the player.
- Quest ends once an end step is reached.



Quest Graph



Start

Player talks to the Villager

Step A

Player eliminates insect infestation

Step B

Player slays the Swarm Queen

Step C

Player talks with the Swarm Queen

End

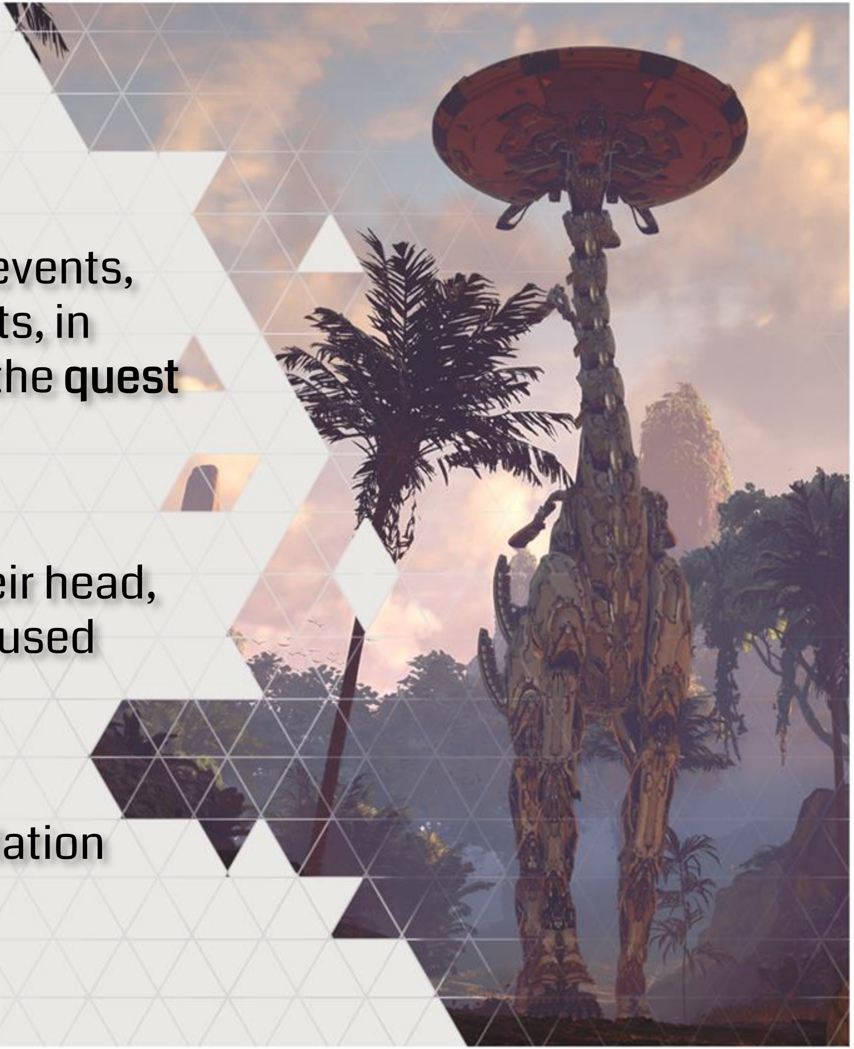
Player reports back to the Villager

Quest Graph

The writer would imagine this web of possible events, where each event unlocks other possible events, in consequence creating the **probability space** - the **quest graph**.

The designer would have the same graph in their head, however the nodes would have a more rule focused context.

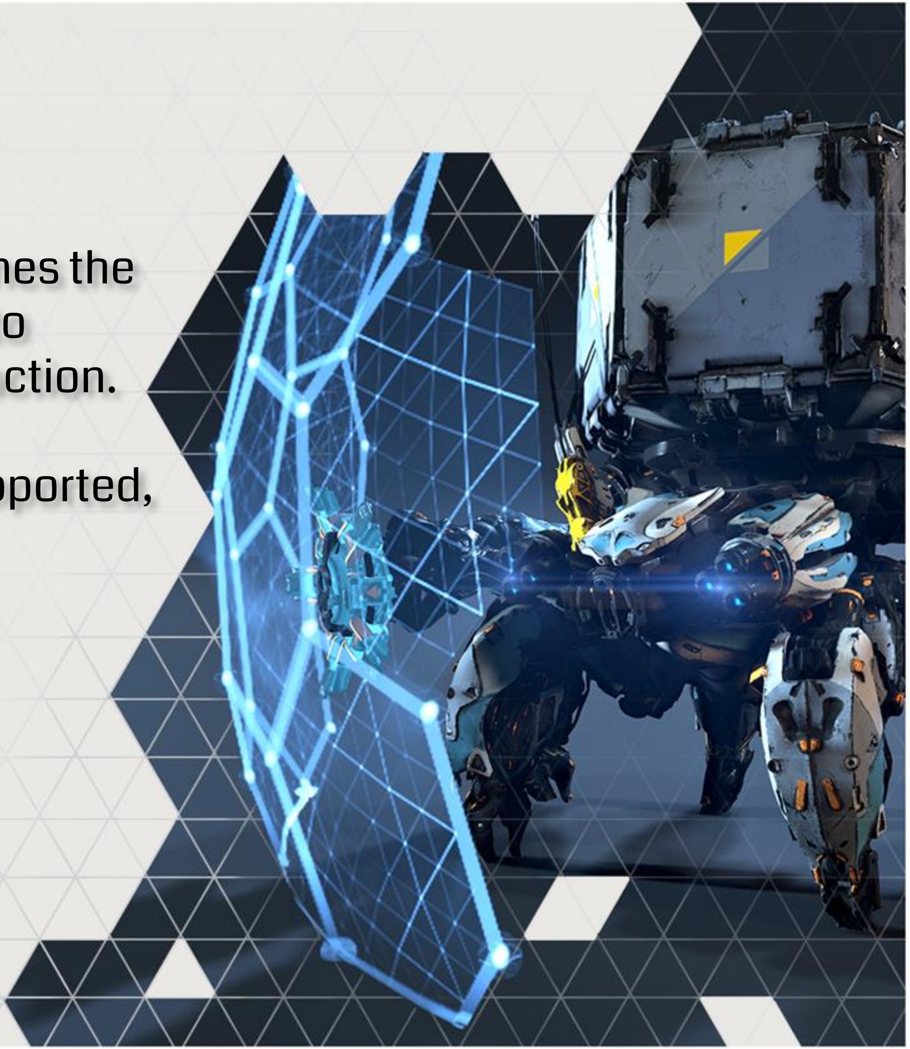
Because the initial step is the same, less translation between the rules and the narrative is needed.



Quest Progression

Each step is connected to a verb which defines the action the player needs to perform in order to progress and if necessary the target of the action.

There is a finite amount of verbs that are supported, for example: obtain, kill, converse etc.

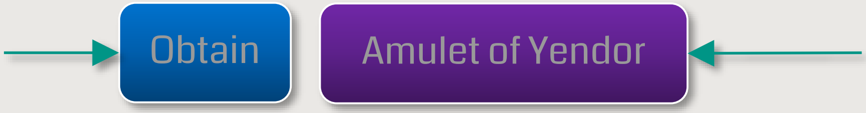


Quest Progression - example

Scene:

The player after searching through rubble finally finds and picks up the Amulet of Yendor.

Quest Step:



Appropriate verb chosen from the list.

Obtain

Amulet of Yendor

Appropriate game object for the action.

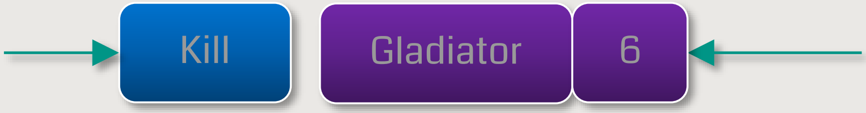
Quest Progression - example

Scene:

In the arena the player **defeats** all **six opponents**.

Quest step:

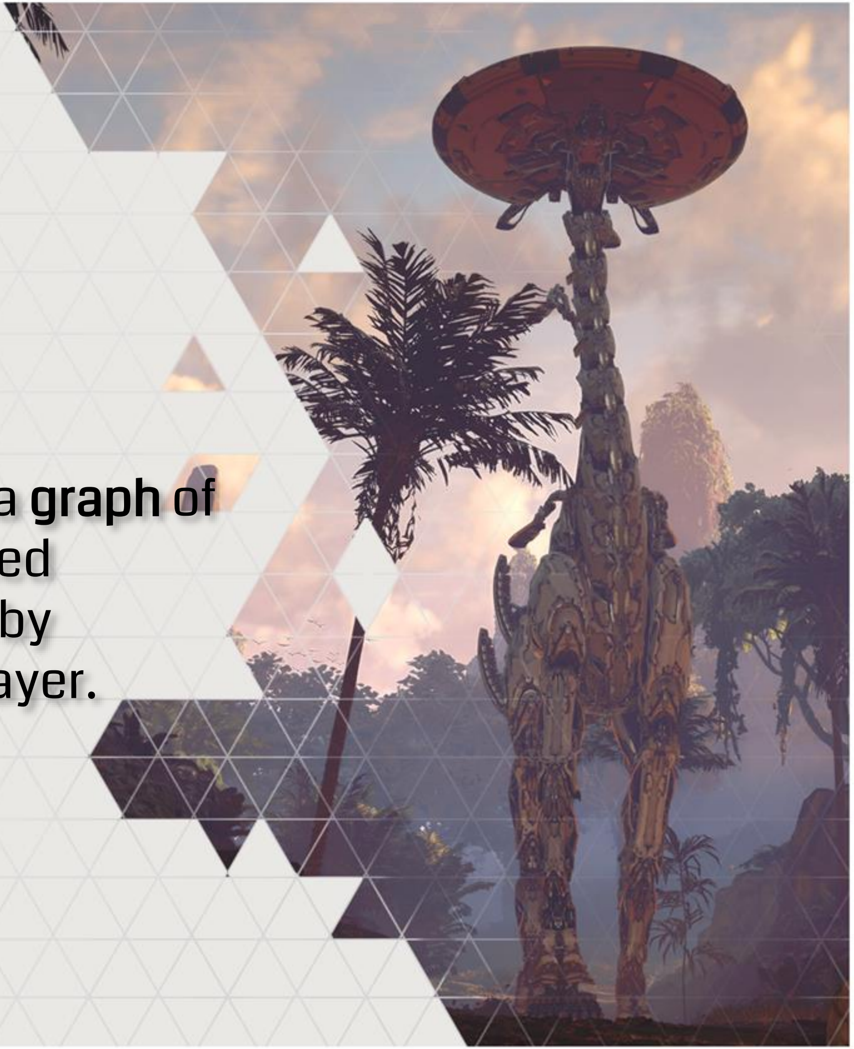
Appropriate verb
chosen from the list.



Appropriate game object
and value for the action.

Summary

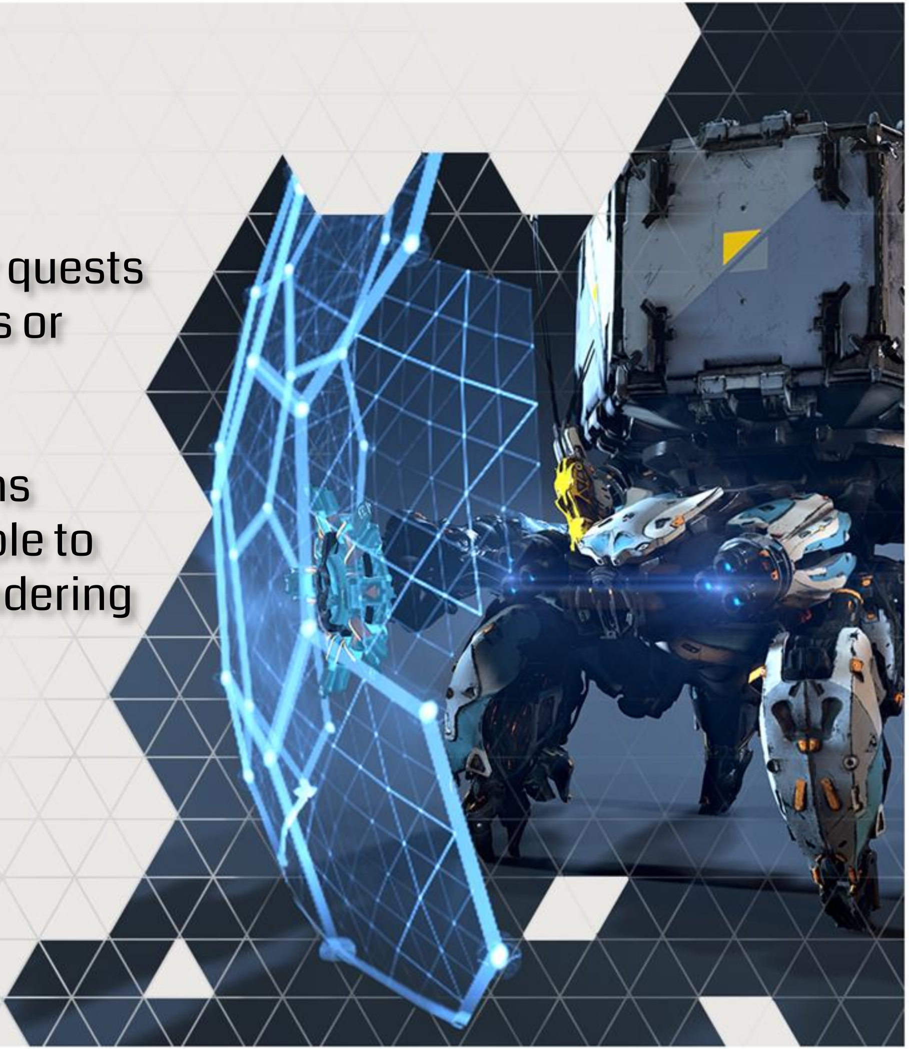
Quests in Horizon: Zero Dawn are built as a **graph** of **steps** (possible plot points) which are linked together by cause and effect and defined by **actions** needed to be performed by the player.



Successes

The graph structure assured that non-linear quests could be created naturally without any tricks or workarounds.

Having a limited set of verbs, defining actions required to progress the quest, made it simple to create common types of quests, without hindering the creation of sophisticated ones.



Player jobs

We noticed that certain simple patterns, which use specific verbs, recur frequently.

Strictness of our system allowed us to automate creation of quests based on such patterns.

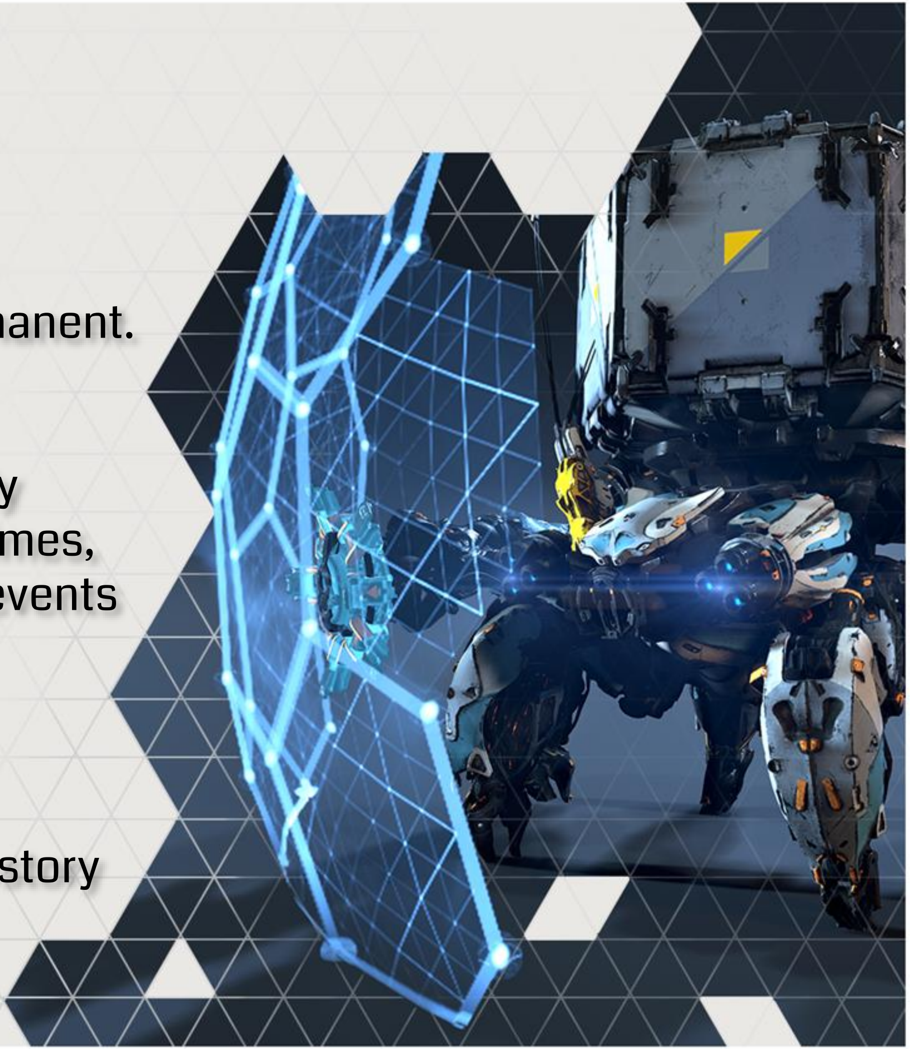
We enabled the player to create simple fetch quests for themselves, in which they can gather items for crafting or for trade.

Time travel problem

In a story once an event happens it cannot "unhappen". Changes to the world are permanent.

However during gameplay, in order to clearly communicate the state of the quest, sometimes, based on player actions, we must allow for events to be erased from time and the quest story reverted.

Because the system was deeply rooted in a story writing metaphor, that was not supported.



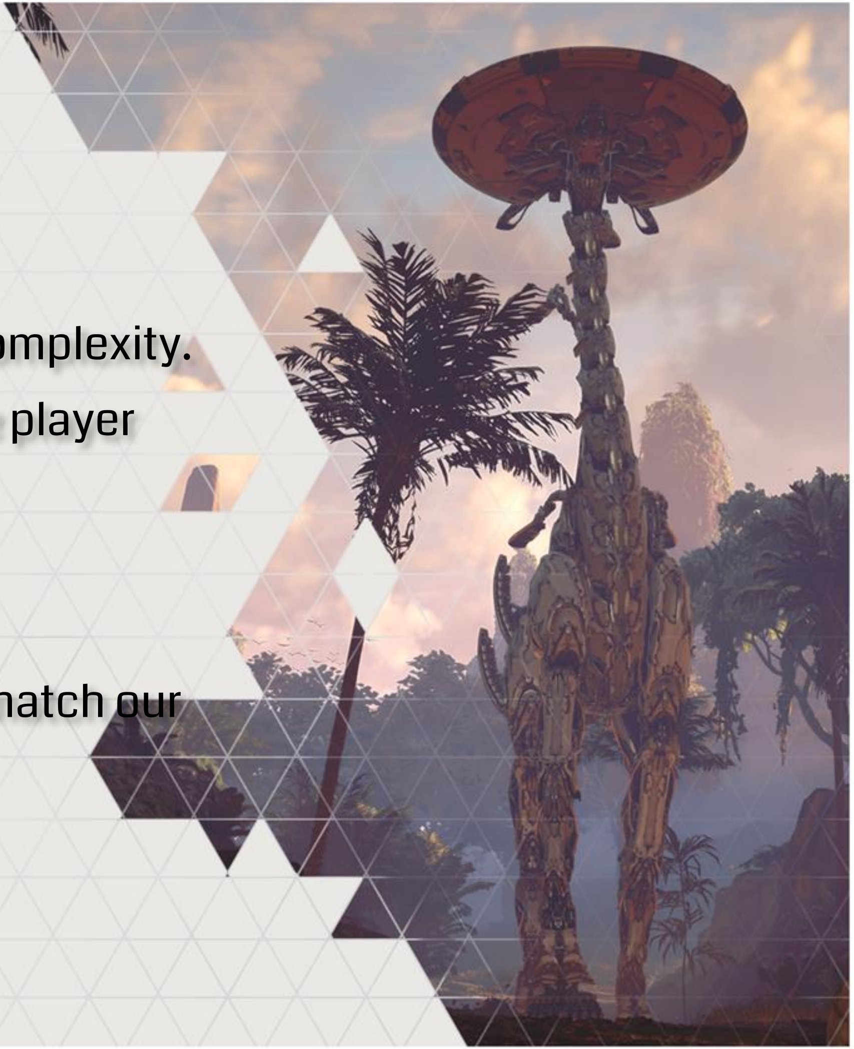
Closing notes

The system served its purpose quite well:

- We created almost 150 quests of various complexity.
- We managed to extend it to support simple player generated fetch quests.

However it still needs improving:

- Time travel problem.
- The list of verbs we had didn’t sufficiently match our needs.



Thank you for listening!