Infernal Engines Pitch Doc

Alexei Othenin-Girard

Chain together alchemical engines to create miraculous concoctions and odd byproducts as you explore this game of logical processes and systems thinking!

## Infernal Engines High Level Overview

Infernal Engines is a puzzle game where players manipulate a series of simple input-output systems, varying the inputs and processes of each, to produce increasingly complex outcomes. The game refers to these input-output systems as "engines," and they're simply defined as something that takes one or more inputs, applies a process or transformation to them, and then delivers one or more outputs. At the beginning of the game, players have only a few options as to how to adjust each of their engines, with the number of steps and variables necessary to produce the desired outcome increasing as the game continues.

Infernal Engines trains players in "systems thinking," the ability to break complex processes down into a series of simple linked steps. Additionally, it builds an understanding of the ways that "upstream" processes and transformations can affect "downstream" results.

### Why Systems Thinking?

Even relatively simple games can be seen as sets of connected processes, which are linked together to create an engaging play experience. Thinking this way lets us consider in-game processesthat use in-game resources, but we can also step back and consider more abstract parts of the game's design that take player inputs like "time" or "chance" to generate in-game resources. Even more crucially, we can see these processes as also having more intangible outputs like "engagement" or, in the case of teaching games, "pedagogical mastery."

Systems thinkinggives designers a bird's eye view of the game they're building, allowing a better understanding both of how the various parts of a design work together but even more importantly how they can maximize the kinds of intangible outputs they're searching for. Systems thinking lets designers identify, for example, which parts of the core loop of a game might be most educational, or most entertaining, and understand how to maximize or modifythose systems to get better player outcomes.

## Gameplay

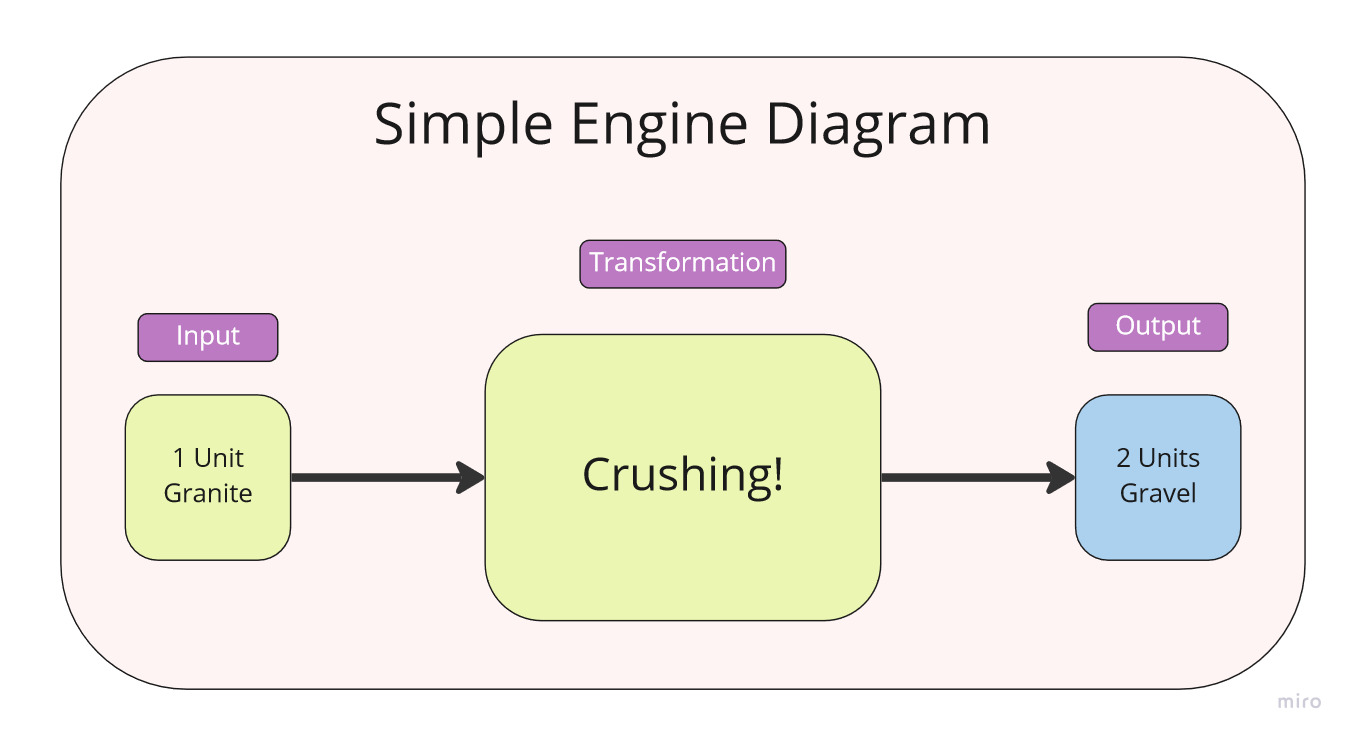
Infernal Engines is a puzzle game that's played in a series of rounds. Each round, players are given an engine (or, in later levels, multiple engines) and a desired output. Players must arrange the internal logic of each engine to ensure that when the engine runs it produces the correct output.

Simple models have one input, one output, and an internal transition that transforms the input into the output. Players assign the input resource and define the type of transformation. Players learn the outcome of transformations as they play the game, building a catalog of inputs, transformations, and outputs. Later models might have multiple inputs, or multiple outputs, or inefficiencies that cause unforeseen byproducts as extra outputs.

### Narrative

The basic concept of Infernal Engines could be developed as a puzzle game, as described above, but the concept also extends itself very nicely into an adventure game/metroidvania-style exploration game, with players acquiring "inputs" (now "materials") from the environment, and using an alchemical transformation process to create outputs (which are simply other materials.) Those new materials could be used to unlock progress gates, complete quests, etc. The theme would involve a young alchemist exiled from the alchemical college who decides to learn to do alchemy on their own and builds a series of "infernal engines" which transform elements in their environments into new products, for the benefit of the local village and to defeat a great evil.

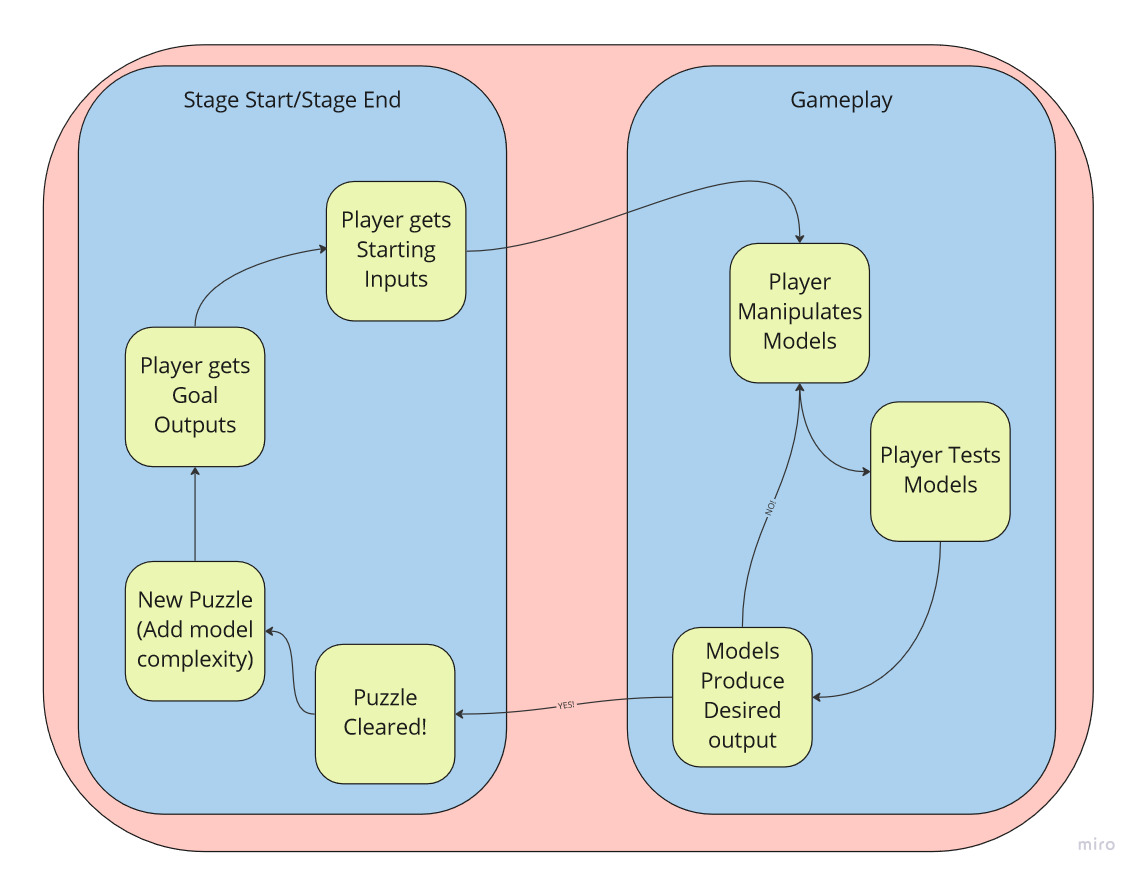
### Simple Model Example



Let's look at a simple engine, like the one above. In this example, the player has chosen one unit of granite as an input, and chosen the transformation "Crushing" for this engine. Running this model produces two units of Gravel. (This interaction would be recorded in an index for players to refer to later.)

Early engines generally take concrete inputs such as "strawberries" or "paper," depending on the puzzle, but later puzzles ask players to incorporate more abstract inputs such as "skill" and "chance" as well. These inputs have special rules associated with them that help to reflect their real-world behaviors. This helps players develop skill and intuition about how those qualities might affect the behaviors of game elements that they're designing.

### Core Gameplay Loop



The core loop for Infernal Engines is quite simple: Players receive their output goals for the new puzzle, begin manipulating engines (choosing inputs, choosing transformations and, in later levels, linking earlier engine outputs to later model inputs.)

Players then run their engines and see if the transformations give the desired outcome. Success means moving on to the next puzzle, while failure just means that players need to tweak their engines and try again.

Successfully hitting output goals solves the puzzle and moves the player on to the next level/puzzle, where they will get new goal outputs, and might also get new inputs, transformations, or even a whole new engine to link onto their process train.

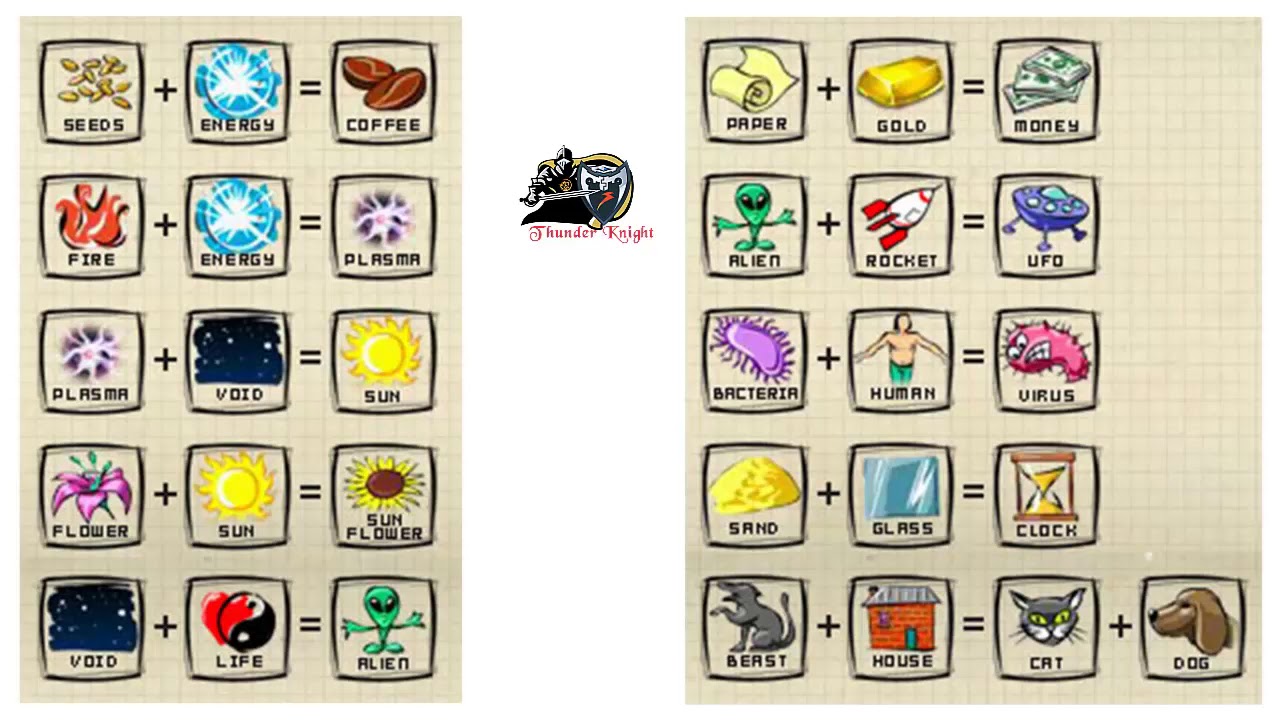
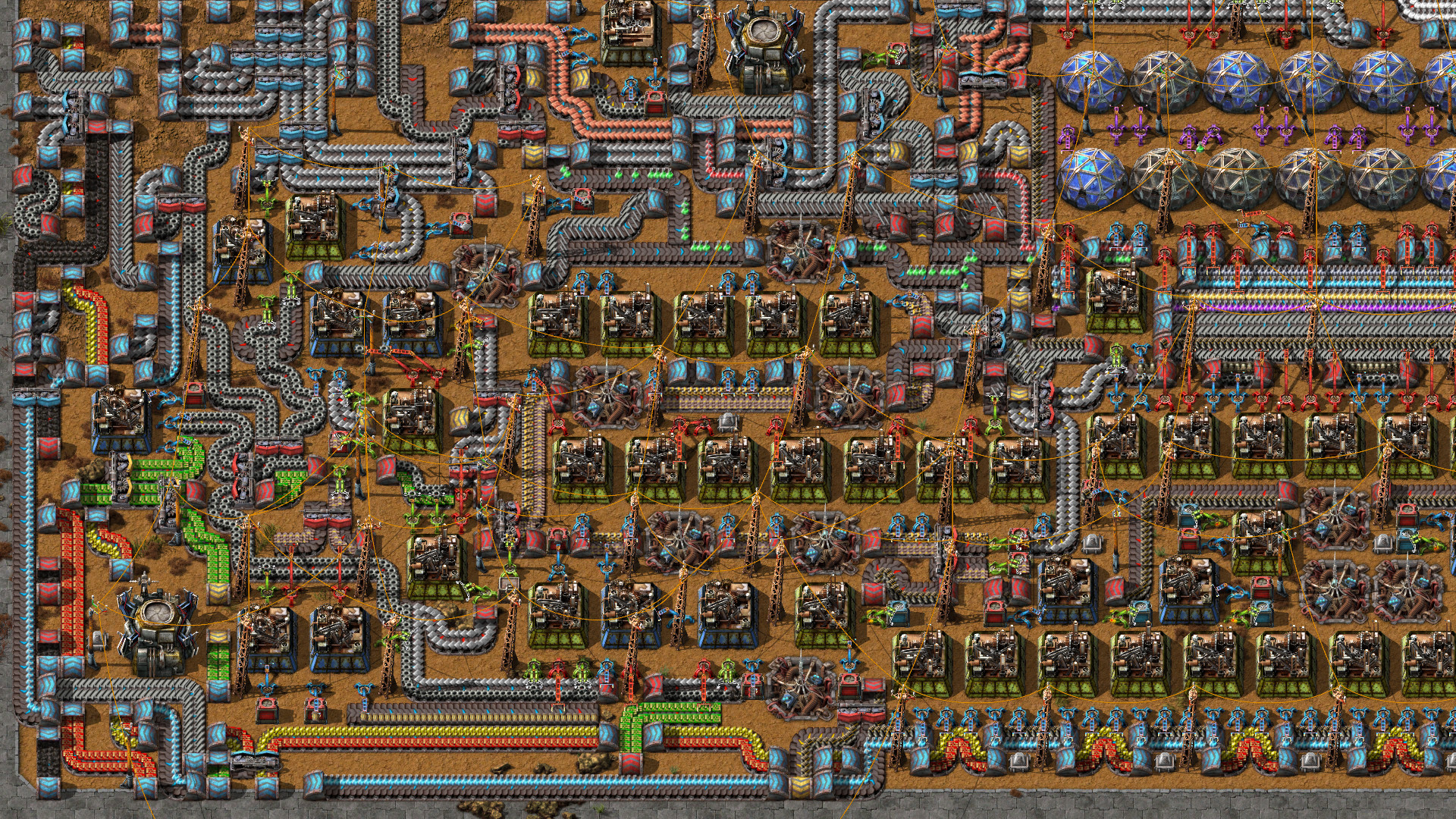
#### "Teach-Test-Challenge"

Taking a step back from the moment-to-moment core gameplay loop, Infernal Engines would offer a very clear opportunity for the creation of a "teach-test-challenge" structure, which would not only add some exciting dynamism to the flow of the game across multiple puzzles, but would also reinforce the pedagogy of the game. In this structure, players would first see a transformation or two, perhaps via a tutorial, and then in subsequent levels need to use that transformation to achieve the requested outcomes. Then, in a "boss" level, players would need to make an intuitive leap about how to use that type of transformation in a new way, that led logically from what they'd seen before but hadn't yet been explicitly outlined to them. This structure rewards players for exploration, and develops feelings of agency and accomplishment in players.

### Progression

The main axis of progression in Infernal Engines is the accumulation of input types and transformations. Very quickly, players will learn that the brute force method of simply trying every input with every transformation is inefficient, and will need to use logic to intuit what outputs they can expect from certain combinations of inputs and transformations. As the game progresses, it introduces linked engines, where the output from one transformation is turned into the input for another, and engines that accept (or require) multiple inputs. Additionally, as the game increases in difficulty, players will learn to avoid certain transformations which produce unwanted "slag" or other undesirable byproducts. Finding models that eliminate or reuse these becomes an important part of game-play in later levels. This teaches designers to be conscious of all of the various factors that might influence a specific gameplay sequence, function, or game element, and take a "systems view" of that element and how it will fit with other parts of their game.

### Where's the fun?

*Elements in Doodle God Factories in Factorio*

Infernal Engines is ultimately a game about exploration and iteration. Taking inspiration from games such as "Doodle God" and "Factorio," Infernal Engines invites players to explore many possible combinations of inputs and transformations, while still grounding the player's goals in logical outcomes. This avoids the tendency for Factorio-style games to increase in complexity too quickly, while reducing the "just randomly put stuff together" play-style that often results from Doodle God-style games. Discovering new combinations is exciting and fun, and realizing that those new combinations unlock puzzle outcomes gives players a powerful sense of being rewarded for both exploration and logical thinking.

## Final Thoughts

Infernal Engines gives players the chance to explore how systems work together to create outcomes, but is as much about the pleasure of discovery as it is about finding the "right" answer. In giving players a safe, encouraging place to test their creativity in finding new combinations of inputs and transformations, it helps game designers learn to think in terms of the processes that comprise the games they're making, making them better game designers overall.