Video Game Kit: how to explain your project in a dynamic way

Context and purpose

This kit is for iGEM teams interested in creating an educational video game inspired by their own project. You do not need to recreate every detail of your system or develop a large game. A short experience built around one important relationship can provide a starting point for explaining a much more complex idea.

In this guide, we offer:

1. Design principles.
2. Designing your video game in steps.
3. Adapting the experience to different audiences.
4. Measuring what players took away.
5. Worked example: Vinesys Pixel Battle.

1. Design principles

1.1. Start with what you want players to understand
1.2. Turn the mechanism into a decision
1.3. Simplify while preserving the important relationships
1.4. Make cooperation matter
1.5. Explain why an action fails
1.6. Leave room for different levels of detail

- **Start with what you want players to understand**. Before choosing a game format, decide what someone should be able to explain after playing. For example: “We want players to understand that our bacterium needs to detect a specific signal before activating a certain response.” This is more useful as a design goal than “explain our project”, because it identifies a relationship that can become a game rule.

  Choose a small number of learning goals. Other details can appear in optional descriptions or in the conversation around the game.

- **Turn the mechanism into a decision**. A scientific mechanism becomes part of the game when it changes what the player can do. If your system requires a signal, make that signal necessary to activate an action. If two components contribute to the outcome, give both a role in reaching the goal. A useful design question is: “Could someone win while completely ignoring the mechanism we want to explain?”. If the answer is yes, consider changing the rules so that understanding the relationship becomes useful and necessary to pass the game.

- **Simplify while preserving the important relationships**. A videogame needs simplifications. Biological processes do not happen in turns, organisms do not have health bars and molecular responses do not consume a fixed number of game energy points. The important task is to preserve the relationship you want to communicate: what activates a response, what depends on an earlier step and what the intended outcome is. Keep a record of what each mechanic represents and where the analogy stops working.

- **Make cooperation matter.** Many iGEM projects involve several components working together. A game can make those relationships visible by giving different characters or actions complementary roles.

  In Vinesys, the vine cannot complete the battle through ordinary attacks alone. Progress depends on the bacterium's support functions. This places the relationship between the two organisms at the centre of the player's strategy.

- **Explain why an action fails.** An unsuccessful move can prompt a useful question: "What was missing?". Give players enough feedback to answer it. A message explaining that a signal or previous step is required is more informative than a generic error without explanation. The aim is to help players connect conditions with consequences, rather than memorise an unexplained sequence.

- **Leave room for different levels of detail.** Players should be able to understand the immediate choice without reading a long explanation every time. Use short action names and clear feedback, then provide optional information about the science. This allows a younger player, a parent and a student with more biological knowledge to explore the same game at different levels.

2. Designing your videogame in steps

2.1. Define the audience and setting
2.2. Write a simple explanation of your project
2.3. Map the mechanism
2.4. Choose a suitable game format
2.5. Turn the science into rules
2.6. Try a paper prototype
2.7. Build a small playable version
2.8. Review science and usability
2.9. Revise and prepare the handover

The following workflow can help you turn your project into an educational videogame. Adapt it to your audience, available time and programming experience.

Step	What to do	Expected outcome
1. Define the audience and setting	**Decide who will play**, the rules and difficulty should be adapted to this decision. Identify what they should understand even if they do not finish.	A short description of your audience, setting and learning goal.
2. Write a simple explanation of your project	Think of the **relevant part of your project** in everyday language before introducing technical terms.	An accessible explanation, with scientific terms added alongside their meanings.
3. Map the mechanism	**Identify the logic** that your game should follow. Distinguish between functions you intend the system to perform and outcomes supported by experiments.	A diagram showing the scientific relationships the game should preserve.
4. Choose a suitable game format	**Select a format** that helps explain those relationships.	A game concept that fits both your mechanism and your audience.
5. Turn the science into rules	For each mechanism, **define what the player does**, what must happen first and what changes afterwards. Record what the mechanic represents and which details you have simplified.	A mapping between scientific mechanisms, player actions and game feedback.
6. Try a prototype	Use cards for actions and counters for resources. Ask someone outside the design group to try the rules. **Check** whether the objective is clear and **whether they can win while ignoring the mechanism you want to teach.**	An initial set of rules and a list of changes to make before programming.

7. Build a small playable version	Start with **one objective**, a **few actions**, **clear feedback** and a simple ending. Use tools your team can maintain, and **review any AI-assisted code** for correct behaviour and scientific meaning.	A working prototype that demonstrates the main learning goal.
8. Review usability	Check that the game preserves the intended relationships. A **game should be understandable** to play and accurate in what it communicates.	Scientific and usability feedback, with specific revisions to address.
9. Revise and prepare the final version	Record what you changed and why. **Organise** the playable files, source code, instructions and scientific mapping.	A documented game that others can understand, use and adapt.

3. Adapting the experience to different audiences

3.1. Younger children
3.2. Older children and early teenagers
3.3. Older students and adults

Age is a useful starting point, but reading confidence, familiarity with games and prior knowledge also affect the support someone needs.

- ### Younger children
  Consider shared play with a parent or organiser of the activity/game. Read choices aloud when needed and focus on concrete ideas such as detecting a signal, helping another character and gaining protection. Introduce technical terms alongside an accessible explanation: “This move represents a protective function. In our project, it is connected to a substance called gamma-PGA, that acts like a sponge.”

- ### Older children and early teenagers
  Ask players to predict outcomes and explain why some actions are unavailable. Encourage them to connect the sequence of moves with the mechanism. Paired play can give one person control while the other explains the next choice, with roles swapped during the activity.

- **Older students and adults**
  Use the game to open a discussion about conditional responses, design choices and the limits of the model. Ask what the game leaves out, how a rule might differ from a real biological process and what evidence would be needed to evaluate the intended function.

4. Measuring what players took away

An enjoyable experience, a completed battle and an explanation of the mechanism are different outcomes. Winning may show that someone found a successful strategy. It does not, by itself, show that they understood the biology behind the game and, therefore, your project.

Ask about the relationship

Choose a short question linked to your learning goal. For example: “Why might the bacterium’s useful response be unavailable at the beginning?”. You can ask a similar question before and after playing. Allow spoken answers or drawings when writing would make participation harder.

Separate learning questions from feedback

Ask what was enjoyable, confusing or difficult to control. These answers can guide improvements even when they do not tell you whether the scientific explanation was understood. When reporting results, describe the setting, participant numbers, assistance provided and questions asked.

5. Worked example: Vinesys Pixel Battle

From our project to a game

For Vinesys, we included the main mechanisms used in our bacterial design and the functions we wanted the bacterium to perform.

The player controls two characters:

- Vineyardax, representing the vine.
- Subtilisk, representing Bacillus subtilis.

Their opponents are Droughtsol and Soilith, fictional representations of environmental pressures associated with drought and adverse soil conditions. This does not mean that sunlight or soil are inherently harmful. The characters just represent challenging conditions within the game. The player chooses an action for the vine and then one for the bacterium. Their actions and the opponents’ responses are resolved in rounds.

Making the mechanism necessary to progress

The game introduces stress as a condition that allows the bacterium to use K^+ Sensing. Two successful uses (following our Two-Factor Authentication system) fill the game's potassium counter and activate the state represented as intein splicing.

This unlocks Gamma-PGA Coating and ACC Relief, which represent protective and stress-relieving functions included in our design. Applying both makes Full Bloom available. Ordinary attacks cannot remove an opponent's final health point. This means the player must use the support pathway to complete the battle.

The health points, energy costs, damage values and cooldowns are gameplay choices. They organise the experience but do not represent experimental measurements. Likewise, the game communicates our intended design functions rather than demonstrating their effectiveness in a real vineyard.

Choosing how much guidance to provide

In Normal mode, actions with unmet requirements are unavailable. Players can consult the information provided to understand what is missing. In Hard mode, they can attempt those actions, but a failed move consumes that character's action, so that only players that completely understood the mechanisms can succeed. The opposing team also presents a greater challenge in this second mode.

Both modes preserve the central mechanism, they just change how much support the game provides and how forgiving the battle is.

Visual design and development

We designed the characters in pixel art and used a vineyard setting to keep the game connected to the project. As Droughtsol acts, the background takes on drier colours. Full Bloom restores its green appearance. This gives players visible feedback about the state of the battle while keeping the same landscape.

Development took place in Visual Studio Code, using object-oriented programming for the main mechanisms and AI assistance during programming. Our team reviewed the mechanics and their scientific interpretation. The game was also prepared for computer and mobile use, with an English edition and a separate edition (oriented to the Cosmocaixa's "Nit de la Recerca" public) offering Catalan and Spanish.

What another team could adapt

The transferable idea is the connection between the project's logic and the conditions for success. Another team could use different characters, a different genre and a different

scientific system while keeping the same design question: “How can reaching the game’s objective require the player to use the relationship we want them to understand?”.