Pipeline Research

Jake O'Connor

UAT MS509

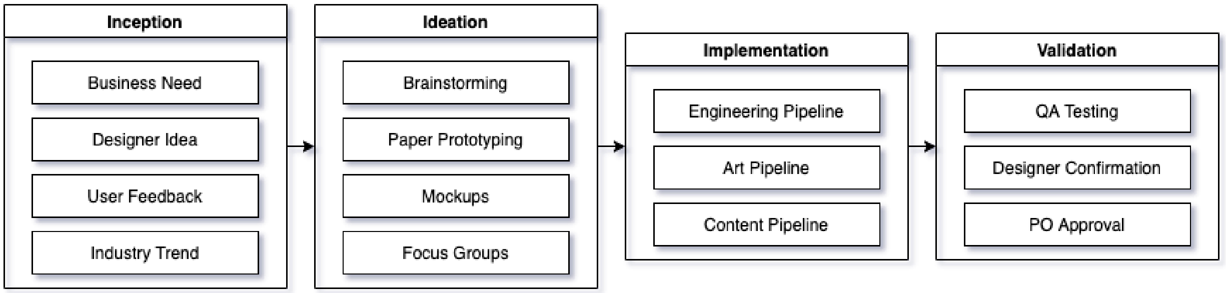
Pipeline Project: Part 2

Overview

There are countless game studios in the world and with them come countless varying pipelines for features, art, and engineering tasks. One thing that they all do share in common is that they are generally hard to pin down, not concrete, and only slightly documented. From the research I've gathered, a decade of personal experience in the games industry, and personal interviews with coworkers about their experiences in the industry, I've decided to divide each category of pipeline into four loose stages of development.

The first stage in each pipeline is Inception, the creation of the original idea, proposal, or need for the pipeline to be enacted. The second stage is Ideation, the development of the idea, its requirements, and determining if it is a good idea to begin with. The third stage is Implementation, the creation of assets, content, code, and any other thing that needs to be done. The fourth and final stage is Validation, the testing, approval, and confirmation of completeness of the pipeline. These four stages of each pipeline are not always linear, often overlap, and very frequently funnel back into one another, but for simplicity's sake each diagram will only contain the best-case pipeline.

Feature Pipeline



The feature pipeline is a blanket pipeline which covers new features to be added to a game and flows into other discipline-specific pipelines in order to complete a goal. Features could be anything from basic requirements of the game to new ideas generated by the team to regulatory necessities for the organization. The feature pipeline is generally the most top-level view of the development pipeline and includes the most disparate stakeholders across the team.

Inception

The basic idea spurring on the feature pipeline could come from anywhere within the development team, but most of the features at this level of pipeline generally come from project-wide decision makers. One example of a feature request that would come from the higher-ups is the implementation of the branding, prompts, and support that are required by COPPA regulations. Another example would be the feature to support in-app payments from different countries and with different currencies, which is a product-level feature that requires multidisciplinary involvement.

The decision of who is allowed to propose an idea at this step of the feature pipeline depends a great deal on the type of game studio. Smaller studios tend to have a much more open proposal process, allowing anyone within the development team to propose features. Larger studios, or studios that are part of a larger organization, often have a stricter proposal process wherein anyone attempting to propose a feature request must have it approved by one or more stakeholders before the idea can even move on to the ideation stage. A third category of studios with a slightly varying inception stage of the feature pipeline are studios working on licensed 3rd party titles; these studios generally have to defer a significant amount of the feature-level decision making to the publisher or IP owner.

Ideation

The ideation stage of the feature pipeline includes the brainstorming of the feature and how it fits into the existing structure of the game and the game's design. Depending on the type of feature, mockups and paper prototypes might be created in order to fully vet the feature before beginning active development. Some studios make use of internal or external focus testing to fine tune a feature's design before pushing it onto the next stage of development. The primary artifact of this stage in the pipeline is the success criteria for the feature, which will be used by the various teams to complete the work.

The ideation stage often includes stakeholders from across the organization, especially for features with broad implications. Engineers will vet the feature’s feasibility and offer insights into the required timelines for the implementation. Artists will identify the required assets for the feature. Designers will ensure the feature’s design is sound and meshes with the overall design of the game. Producers and product owners will confirm that the output of the ideation stage is reasonable and that the teams involved have enough bandwidth to handle the required work.

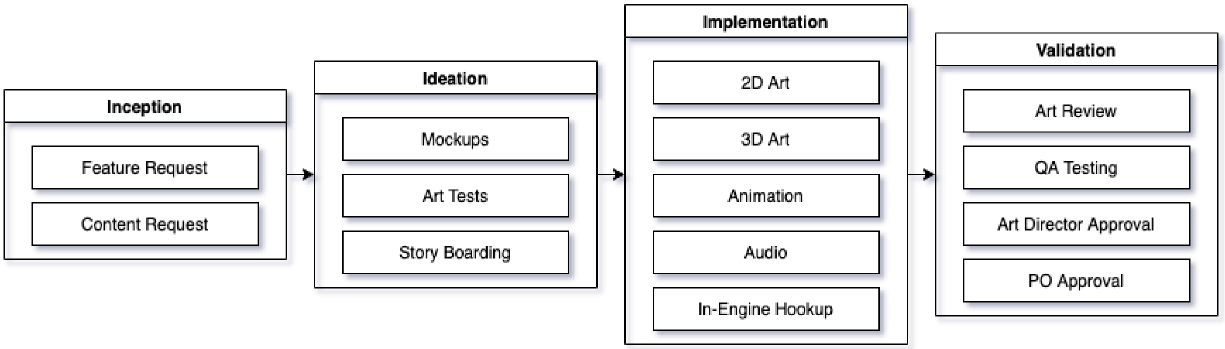
Implementation

Since features generally rely on two or more teams or disciplines in order to be completely implemented, the implementation stage of the feature pipeline defers to the distinct pipelines for the other disciplines. Once the other disciplines’ pipelines have been completed to a satisfactory level, then the feature as a whole can move onto the feature pipeline’s final stage.

Validation

The fourth and final stage of the feature pipeline is the validation that the output from the individual disciplines’ pipelines meet the original feature’s expectations. This generally includes a full QA test pass on all the success criteria identified in the ideation stage as well as halo testing around the feature’s affected area to ensure that no unintended changes were made. This stage is also when the feature as a whole receives approval and confirmation from the designer(s) who developed the idea and the product owner for the feature.

Art Pipeline



This general art pipeline covers the creation of new art assets within a game studio. The basic flow is generally the same for all types of art assets whether 2D or 3D, as the overall process is not that different on the macro scale. The creation of audio assets is also included in this pipeline, as generally audio assets are created corresponding to art assets and have much the same requirements.

Inception

The inception stage for the art pipeline in most studios is one of the simplest of all the different disciplines' pipelines. Art assets generally don't exist on their own, they are created to serve a function within the game or its ecosystem. As such, most art pipelines are kicked off by an associated feature or content request.

Ideation

The ideation stage of most studios' art pipeline is often one of the most complex. Some studios such as Riot Games employ an extremely open and free form brainstorming process during the ideation stage, where dozens of people across different specialties all contribute to fleshing out a design before it moves on to being implemented (*Creating League of Legends Champions: Our Production Framework Revealed* 2019). More rigidly structured studios, or those with much smaller teams and budgets, will sometimes have only a single person involved in the ideation process before submitting it on to the implementation stage.

The goal of the ideation stage, regardless of how long or complex it is, is to determine what assets need to be made during the implementation stage and how those assets will fit into and mesh with the current assets within the game. Artists in this stage will often create storyboards and mockups for animation sequences that need to be created and establish color palettes for new art pieces and characters. The output of this stage is a series of tasks for individual artists to create the assets needed to support the original need.

Implementation

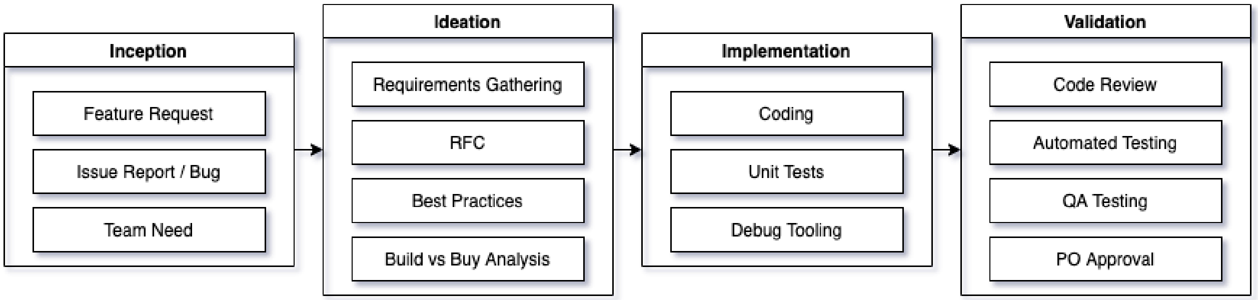
The implementation stage of the art pipeline greatly depends on the type of art asset being created and the purpose it serves in the game as a whole. For a simple sprite, this stage will be incredibly quick and only involve a single artist. For more complicated assets, like an entirely new character being added to the game, this stage will involve multiple artists from different specialties including modelers, riggers, and animators.

For most studios this stage involves the in-engine implementation of the art assets as well, but that often depends on the type and size of studio, as well as what sort of tools they are using for development. In my experience, and based on secondhand information, the distinction between studios which require artists to implement their own assets into the game engine and those which do not largely depends on both the size of the studio and how niche the game engine is. In small studios using popular game engines, artists are almost universally expected to implement their own assets in the engine. This is generally due to smaller budgets and a higher expectation of familiarity with the tools being used. Larger studios using more specialty software (engines built in-house, etc.) trend towards pipelines which don't expect artists to implement their own assets in engine, but instead rely on tech artists or engineers to implement them.

Validation

The validation stage for the art pipeline is generally characterized by the review, testing, and approval of the newly implemented art assets. This stage often overlaps with the implementation stage in many studios, with regular art reviews being done throughout the implementation process to ensure that the ongoing work adheres to the team's quality standards. For almost all studios, this stage of the pipeline is when the QA team validates the functionality of the art in-engine and that it satisfies the success criteria laid out in the ideation stage. Studios with art directors also use this stage to ensure that the art adheres to the team's and studio's standards and fits within the game's overall artistic direction. Like all pipelines, one of the final steps across nearly every studio is the confirmation by the product owner that the new assets achieve the goal that the process was set out to.

Engineering Pipeline



This pipeline is a general process for any type of engineering asset added to a game, whether it be a new feature, a bug fix, or a new tool. The engineering pipeline is one of the most consistent across studios, regardless of size or funding. The only minor differences lie within a few optional steps which some larger studios make use of for certain types of engineering assets.

Inception

The inciting idea of the engineering pipeline can come from a wide variety of sources. Like with the art pipeline, a large majority of engineering pipeline tasks originate as feature requests from the feature pipeline. Additionally, the engineering pipeline is often kicked off by a bug report either from the QA department or from users of the game or system. Lastly, many engineering pipelines are started due to some need within the engineering team or within another team working on the project. One example of an internally generated engineering task was the creation of the tools suite which went into building Horizon: Zero Dawn, which according to their tools team was originally conceived before outside requests even began coming in (Creating a Tools Pipeline for Horizon: Zero Dawn, 2019).

Ideation

The ideation stage of the engineering pipeline is the one which varies most based on the type of work and the size of the development team and studio. Regardless of studio size, this stage is characterized by the identification of the success criteria for the engineering task, and by identifying the individual necessary components of the task. Generally during this stage, the engineers on the team will consider existing tools and frameworks within the studio that can be used and consult engineering best practices from both within and outside of the studio.

Two practices during the ideation stage of the engineering pipeline which appear less frequently, but are still fairly common especially among larger studios, are Requests for Comments (RFCs) and Build/Buy Analyses. RFCs are documents shared throughout the studio to gather feedback, comments, and critiques of a potential engineering solution, which allows the engineer or team which is set to implement the task to consult a wider audience than their direct coworkers. Build/Buy Analyses are done most often for engineering tasks which are common within the industry or have well-known solutions, and compare the cost and complexity of developing and maintaining an in-house solution to a problem with using a pre-built solution.

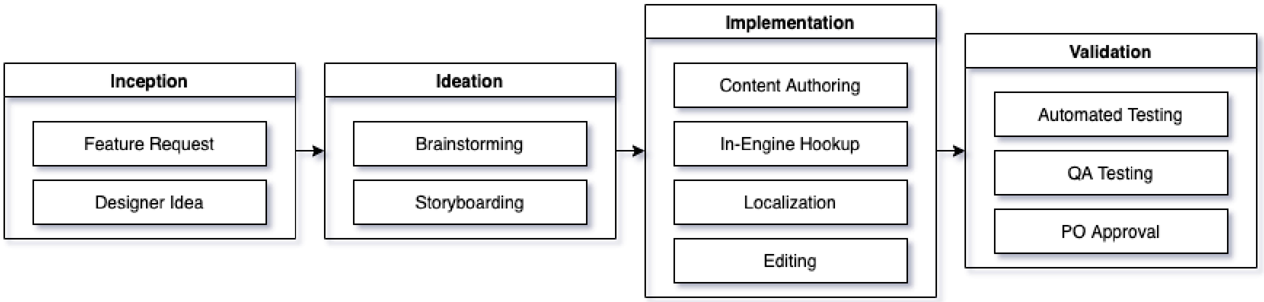
Implementation

The implementation stage of the engineering pipeline is, at least process-wise, very simple. Engineers complete the task or tasks necessary to achieve the success criteria established in the ideation stage. In studios where unit tests are used, which is fairly common, the implementation of unit tests is also expected during this stage of the pipeline. Finally, depending on the type of task being completed, many studios expect debug tooling to be implemented alongside the implementation of new engineering tasks.

Validation

The engineering pipeline ends with the validation stage, wherein multiple different steps guarantee that the engineering task was completed successfully. Code reviews are required or encouraged by most professional game studios; wherein fellow engineers review completed code to ensure it adheres to code standards and is free of mistakes. Automated test suites are run, including unit tests and integration tests, depending on the level of automation within the studio. Like all other pipelines, the final two steps are generally approval by the QA team and by the product owner for the pipeline to ensure that the task functions as expected in context and meets the intended goals.

Content Pipeline



The content pipeline covers the completion of tasks which are neither directly engineering nor art, and do not change the overall functionality of the game. Content pipeline tasks are things such as the addition of a new level to an existing level-based game, or the addition of a new narrative into a game. This pipeline category is inherently very broad and mostly depends on what kind of game is being made, but there are many shared steps regardless of content type.

Inception

Like engineering and art pipelines, the inception stage of the content pipeline is often kicked off by the feature pipeline. For example, creating the personality, backstory, and dialog of a new playable character would be the content pipeline's portion of the overall feature pipeline goal of creating that character. For content pipelines which aren't kicked off by a feature pipeline, the originating idea generally comes from someone on the design or content team.

The level of flexibility in the inception stage, who can propose new concepts and ideas, largely depends on the culture of the studio. Massive studios with equally massive teams generally have a much more rigid definition of who is allowed to propose new content ideas, sometimes leading either to ideas proposed by people outside of that definition being ignored or those ideas having to be funneled through someone who does fit the definition. Likewise, smaller studios working on 3rd party licensed games also frequently constrain the content inception stage, but mostly due to the tight time and content constraints imposed by the publishers and/or IP owners.

Ideation

The ideation stage of the content pipeline greatly depends on the type of content being created within the pipeline. The two most common processes within this stage are brainstorming and storyboarding, with the design and/or content team generating and iterating on ideas that satisfy the original concept. Depending on the type of game and idea, this may involve comparing the idea to the game's lore to ensure it fits adequately, or consulting with product owners and designers to ensure that the task does not stray from the overall vision of the game.

Implementation

The implementation stage of the content pipeline is when the content authors will do their authoring. Depending on the tools being used at the studio, this may involve in-engine hookups or secondary tools which automatically link to the engine. Depending on the type of task, this stage may also require the editing and potential localization of the content.

Validation

The final stage of the content pipeline is the validation stage. In most larger studios, especially those with a high content-to-feature ratio, automated testing during the validation stage ensures that the newly added content is functional and adheres to the team's standards. Depending on the type of content this might include JSON Linting, unit tests, or some other bespoke automated testing tools. Like all pipelines, some form of QA team testing and product owner acceptance are the final steps in this stage of the pipeline.

References

Appelman, R. L. (2009). Defining the Development Pipeline for Meaningful Play. In Proceedings of International Simulation and Gaming Association Conference. Singapore.

The Animation Pipeline of Mario + Rabbids Kingdom Battle. (2019). YouTube. https://youtu.be/qxLR8qbD5JE.

The Animation Pipeline of Overwatch. (2018). YouTube. https://youtu.be/cr7oO8kDu8g.

Creating a Tools Pipeline for Horizon: Zero Dawn. (2019). YouTube. Creating a Tools Pipeline for Horizon: Zero Dawn.

Creating League of Legends Champions: Our Production Framework Revealed. (2019). YouTube. https://youtu.be/dhJXtPPfKbg.

George, Ryan. (2021, July 16). Senior Software Engineer at Wizards of the Coast, former Software Engineer at NetherRealm Studios. Personal communication [Online chat].

Making the World of Firewatch. (2016). YouTube. https://youtu.be/hTqmk1Zs_1I.

Moy, Kevin. (2021, July 16). Senior Digital Production Artist at Wizards of the Coast, former Senior Environment Artist at Turbine. Personal communication [Online chat].

The Motion Capture Pipeline of The Last of Us. (2016). YouTube. https://youtu.be/2GoDlM1Z7BU.

Narrative and Mission Design in Assassin's Creed Iii. (2016). YouTube. https://youtu.be/pOh8cGFu4Os.

Up Sh*t Creek: Pro Tips for Managing the Unmanageable Project. (2019). YouTube. https://youtu.be/dNlEZZlmIcw.