WHITE PAPER

Streams Adoption Guide

Easier Branching and Merging with Perforce P4

Executive Summary

Perforce Streams is the branching feature within Perforce P4 (formerly Helix Core). Unlike traditional branching models, Streams has a built-in framework that eliminates complexity and manual errors, so it's easier to manage assets at scale. It's a better way to branch and merge, making it easier to manage concurrent development, dependencies, and other common branching and release activities. It provides projects with a workflow framework based on best practices. Plus, it is flexible enough to accommodate many branching strategies and development models, including the mainline branching model. Streams is both lightweight and powerful. Only P4 has Streams, giving you a unique built-in branching advantage you won't find anywhere else.

Contents

Understanding Perforce Streams

Before reading this paper, it's helpful to understand Streams, how they work, and their associated commands and tools.

Traditional branching models require manual workspace setup, careful tracking of merge conflicts, and directory structures that impose rigidity on projects. But Streams automates workspace configuration, enforces best practices, and provides a clear visual model of how changes flow between different branches.

Dependence on Directory Structure

In most systems, a directory structure conveys important contextual information about the intended use and relative stability of its branches. For example, consider the Titan project, which features a main branch and two release branches. A common directory structure is shown in Figure 1. The REL container directory indicates that the subdirectories are release maintenance branches.

After moving the Titan project into Streams — and adding a couple of new branches — the directory structure is flat (see Figure 1).

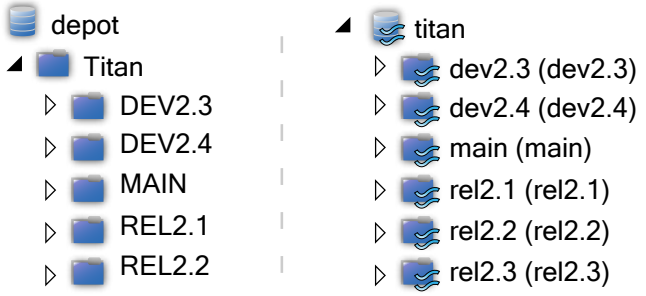


Figure 1: Titan branches (L) and the Titan branches as streams (R).

Branch Strategies for Stability

According to the mainline model, a release stream follows a particular flow-of-change pattern. This structure governs when changes are merged back to the main branch. Before moving to Perforce Streams, the REL container level in the Titan project conveyed important information about the branches in that container.

The branches were release maintenance branches, implying a higher level of stability than development branches or the main branch. With Streams, this information is captured in the stream meta-data. It is presented visually in the stream graph, which is a visualization that is built into the P4 Visual Client (P4V). See Figure 2.

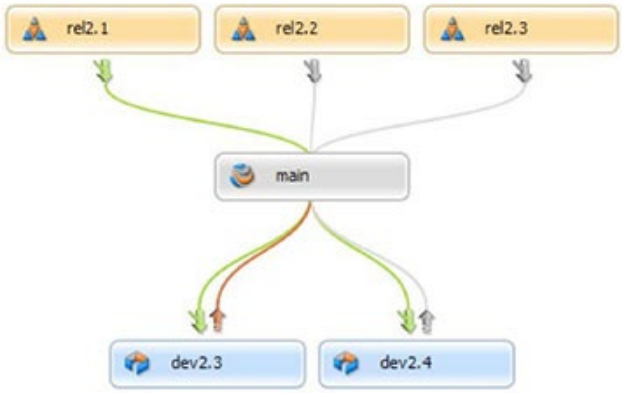


Figure 2: Stream graph conveys branch stability and the flow of change.

Perforce Streams Composition and Inheritance

Streams, like workspaces and branches, have views. A product architect can use the stream view to define the set of modules or components in a stream. In other words, a stream view defines the purpose and origin of files, including:

- Files branched for work.
- Files imported or excluded from the parent.
- Files imported from other parts of the repository.

The stream view is inherited by all child streams and workspaces, which simplifies the startup work for new developers in projects. When a new user creates a workspace from a stream, the workspace view is generated automatically. Workspace views are also updated automatically when moving from stream to stream.

Expert Insights

Ryan Maffessoli, Senior Sales Engineer

"One of the most powerful features of Streams is its inheritance model. When a new workspace is created from a Stream, its structure is automatically inherited. This keeps teams aligned and reduces manual errors. So developers can dive in and start working right away, without worrying about misconfigurations. Plus, with automated workspace updates, managing assets is easier than with traditional branching models."

Move Existing Projects to Perforce Streams

Mechanically moving data into a Streams depot from an existing project is straightforward. Although a detailed history import (DHI) strategy could be used, in most cases it is not necessary. You can simply integrate the tips of the relevant branches into new streams. P4 tracks and respects the indirect branching history between the newly created streams.

The following provides a high level overview for migration:

1. Define a new Streams depot for the project. Administrative access is required.
2. Choose the relevant branches to copy into the Streams depot. You have the option to include all branches or only those that are still actively being used.

Perforce Streams Best Practices:

For each branch, define an equivalent stream. Start with the mainline stream and keep in mind the following:

- Consider which types of streams to use. A mainline stream is the main branch and there is usually only one mainline per Streams depot. Release streams are assumed to be more stable than the parent. Development streams are less stable than the parent.
- Review the flow of change, which is defined by the Streams type. Most development branches allow a bidirectional flow of change, while release maintenance branches usually do not accept changes from the parent.
- Choose stream names carefully. Stream meta-data captures the most important information about a stream. Stream depths allow architects to adjust the organization structure for Streams. But formalizing a naming convention serves as a powerful and easier method for referring to Streams and their locations.
- Determine the implications of parent-child relationships between Streams. Consider how you perform merges between branches. For example, if you normally merge bug fixes from oldest to newest and then merge through to the mainline, you may want the oldest release to use the next oldest as its parent. In a more advanced branch model, an older 1.0 release receives bug fixes first, followed by the 1.5 release, and then the main branch. See Figure 3.

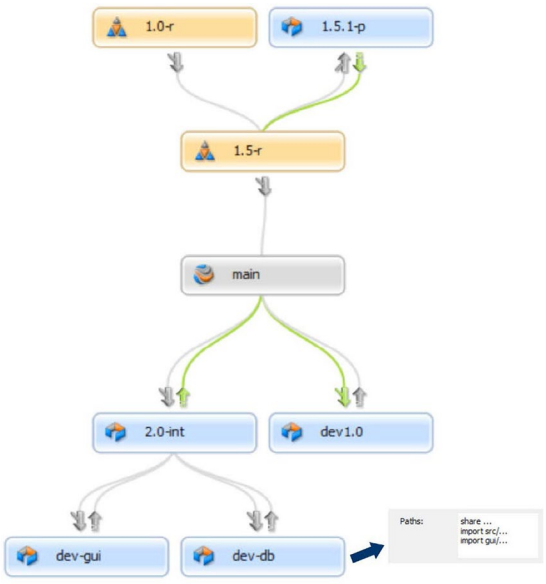


Figure 3: A more advanced branch model in the stream graph.

- Capture any relevant information about stream composition in the stream views. This information may be currently stored in a branch spec, or it may be implied. For instance, the dev-db stream in Figure 3 imports two modules from the integration stream, as shown in the stream paths.

Migrating Branches into Streams

Now you are ready to start moving your data into Perforce Streams. Start by copying each branch from its original location to the equivalent stream using the p4 copy command. For example:

```
p4 copy -v //depot/Titan/MAIN/... //titan/

main/...

p4 submit -d “seeding Titan stream

mainline”
```

The revision graph for a file shows that its pre-streams history is readily accessible. See Figure 4.

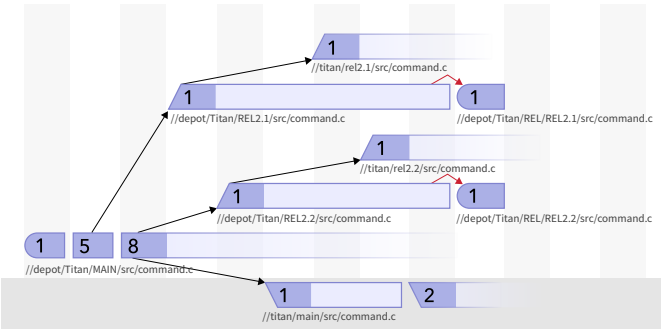


Figure 4: Revision graph showing pre-streams history of the Titan branch.

In most cases, merging between two streams will return results similar to merging between two original branches. This is due to the indirect history between the newly copied streams. Once copied from their legacy location, the streams could change moving forward. But their merge history will remain constant.

For example, say there is one bug fix waiting to be merged from the Titan REL2.1 branch back to MAIN. After integrating the current state of MAIN and REL2.1 into streams to show the pending bug fix, run the command:

```
p4 integ -S //titan/rel2.1
```

In cases with more complex merge history, you should preview any merge operations immediately after moving into Streams to ensure that the results are as expected.

Analysis of Perforce Streams: What to Expect

For most people, Perforce Streams will be easy and straightforward to use. However, Streams are a departure from classic branching strategies in P4. Advanced training and documentation should be provided for your teams. Review and analyze how Streams may impact your users.

Benefits for Users

Streams provides a simplified workflow for:

- Workspace creation and management.
- Stream or branch creation.
- Merging using the merge-down/copy-up paradigm.
- Stream view management.
- In-place branching and fast workspace switching.

Learning the new workflow and commands is the main impact for users. New workspaces, other tools, and scripts may need to be updated. Users will need to understand the new directory locations and the structure of their Streams.

Impact on Release and Project Management

Perforce Streams simplifies many release management activities. The flow of change is guided by the Streams framework, and the stream view allows for dependency management and code reuse. Build and release scripts and tools may need updating to take advantage of these new features.

If your previous branching model was well-designed with supporting scripts and tools, then the overall workflow will feel familiar. An inefficient branch-and-release model will become more visible in the Streams framework. Users may find themselves working outside of the expected guidelines frequently. Moving to Streams is a good opportunity to identify and fix any problems in the legacy release model.

Planning Your Team's Migration

Another important choice is how to migrate your many teams and when to move them. Moving the entire organization over to Perforce Streams at once simplifies some of the planning, since legacy tools and processes can be retired. However, doing so requires careful planning and management.

Transitioning your teams gradually after a successful pilot allows for more time to develop new processes and tools. If there are strong dependencies between the work done by different teams, additional work will be required to ensure that teams can still collaborate with those using Streams, and vice versa. Your migration schedule may depend on the level of collaboration, your release schedule, and other important milestones. It will take time for each team to adapt to new processes.

Review Branching Strategies for Each Team

You may find that some teams should not move to Streams. Teams that simply use P4 as a document repository and perform little or no branching would not benefit. Parallel work would also not benefit much from the Streams workflow.

Some teams may already have a comprehensive set of scripts and tools in place to support their unique development process. In this case, moving to Streams could prove disruptive. Streams provides built-in workflows based on observed best practices. If your needs are currently satisfied, you may not realize any gains by changing your existing branching strategies.

Managing Requirement Tools

There are many Perforce ALM (formerly Helix ALM) tools that interact heavily with P4: code review tools, defect trackers, and continuous integration engines all interface with P4 to some degree. Therefore, it is vital to analyze and understand how Perforce Streams could potentially impact your toolchain in advance.

Any tool that relies on knowing and understanding the product directory layout will need to be changed when you migrate to Streams. The directory structure will be flat, so tools and scripts cannot rely on the same directory conventions in use to convey structural context. Perforce Streams captures the information in a different way, so the tools that rely on this information must account for this change.

Supported Releases for Perforce Streams

Transitioning to Streams should be done using the most recent version of P4. All clients, APIs, and plugins should also be on the most recent and supported release. Administrators will need to manage a server upgrade and make sure that all affected users have appropriate client software.

Conclusion

This document details, at a very high level, some of the considerations involved when moving existing users and projects to Perforce Streams. Planning and preparation are key to any successful process transition and moving to Streams is no exception, particularly if you want to realize the potential productivity gains.

Explore More

Ready to adopt or migrate to Streams? Contact us here.

Command-Line Documentation

Get the full breakdown of how Streams works, read the docs.

Streams 101 Guide

New to Streams? Learn the basics in our beginner-friendly 101 guide.

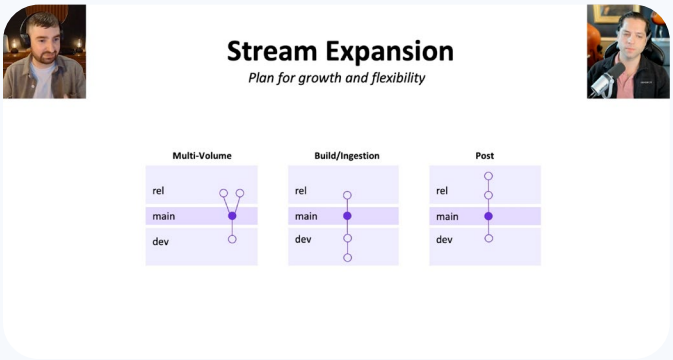
Quickstart Video

Get started fast with this short overview video.

Streams in Action

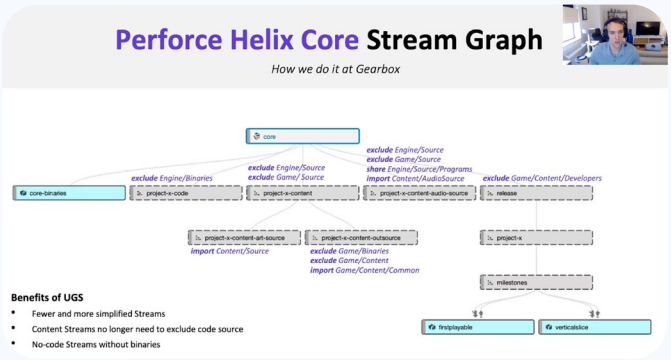
For Media & Entertainment:

Watch this webinar with DNEG to see how they structure complex projects with Streams.



For Game Development:

Watch this webinar with Gearbox to see how they use Streamstosupporthundredsofartistsanddevelopers.



About Perforce

The best-run DevOps teams in the world choose Perforce. Powered by advanced technology, including powerful AI that takes you from AI ambition to real results, the Perforce suite is purpose-built to handle complexity, maintain speed without compromise, and ensure end-to-end integrity across your DevOps toolchain. With a global footprint spanning more than 80 countries and including over 75% of the Fortune 100, Perforce is the trusted partner for innovation.

Harness the power of AI and accelerate your technology delivery without shortcuts. Build, scale, and innovate with Perforce—where efficiency meets intelligence.

Request Support ▶

Contact Us ▶