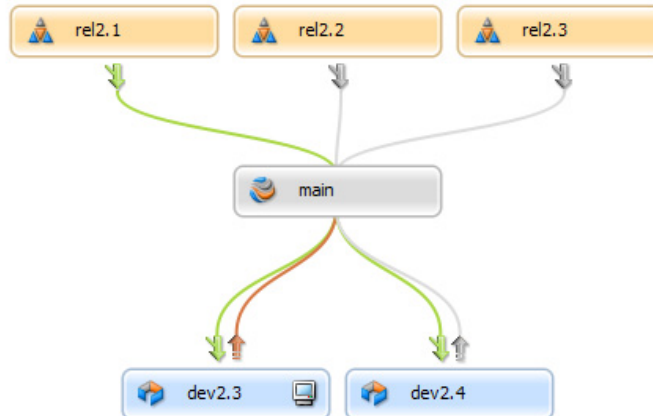


# Perforce Streams



An innovation in workflow that delivers flexible process and best practices to codeline management, right out of the box.



## Branches with Brains: Enforce Best Practice Branching Strategies Out of the Box

Streams intelligently organize the modules that comprise your projects, along with the policies that govern their workflow. Embodying a flexible best practice branch-and-merge strategy, streams ensure that change flows in the right direction and order.

## Eliminate Overhead. Simplify Common Processes. Increase Agility and Scalability.

Streams promote efficiencies such as code re-use, automated merging, fast context switching, efficient workspace updates, and inherited workspace and branch views. While the Perforce Server handles these logistics, users can focus on their work. In projects with a large volume of data, the time and performance savings are considerable.

## Required Components

Streams require a 2011.1 Perforce Server and a compatible client.

## Availability

Perforce Streams are available in the latest release of Perforce v2011.1. Visit [perforce.com/streams](http://perforce.com/streams) for more information.

## Key Features

Streams build on Perforce's strong integration and data management fundamentals, offering many compelling new features.

### Model the Flow of Change

Streams and the associated visualization tools model the relationship between codelines. Using streams, it is readily apparent that a release stream is more stable than the mainline, and that change only flows from the mainline to the release stream.



### Fast Context Switching and In-Place Branching

Streams let you quickly switch from one branch to another in a single workspace. Only the files that actually differ between two branches are updated, making for an extremely efficient process. A stream itself can easily be moved to a new parent, providing additional flexibility for rapid development workflows.

### Stream Composition and Dependencies

The stream view captures and defines the composition of a stream, including:

- The level of sharing between streams
- Which modules are being actively developed
- Which modules are made available from a parent stream for read-only use
- Which dependencies are imported from other parts of the repository

### Inheritance: Easy Management of Streams and Workspaces

Once a product architect has defined the stream view, that information is inherited by child streams and by the users working in the stream. Workspaces are created and updated automatically based on the stream view. New child streams can be created with a minimum of configuration.

### Powerful Visualization Tools

The information in the stream model is used to build powerful visualization tools such as the stream graph. The stream graph provides a quick overview of the stream model, visual cues for pending merges, and easy access to branching tools.

### Best Practices Built In

Streams embody and help to enforce branching best practices like the merge down, copy up paradigm. Using the information embedded in the stream model, streams help to ensure that change flows in the right direction, in the right order.

### Flexible and Agile

Streams are both Agile and flexible to handle realistic, complex development models.