Component-Based Development, Lego, and the Internet of Things

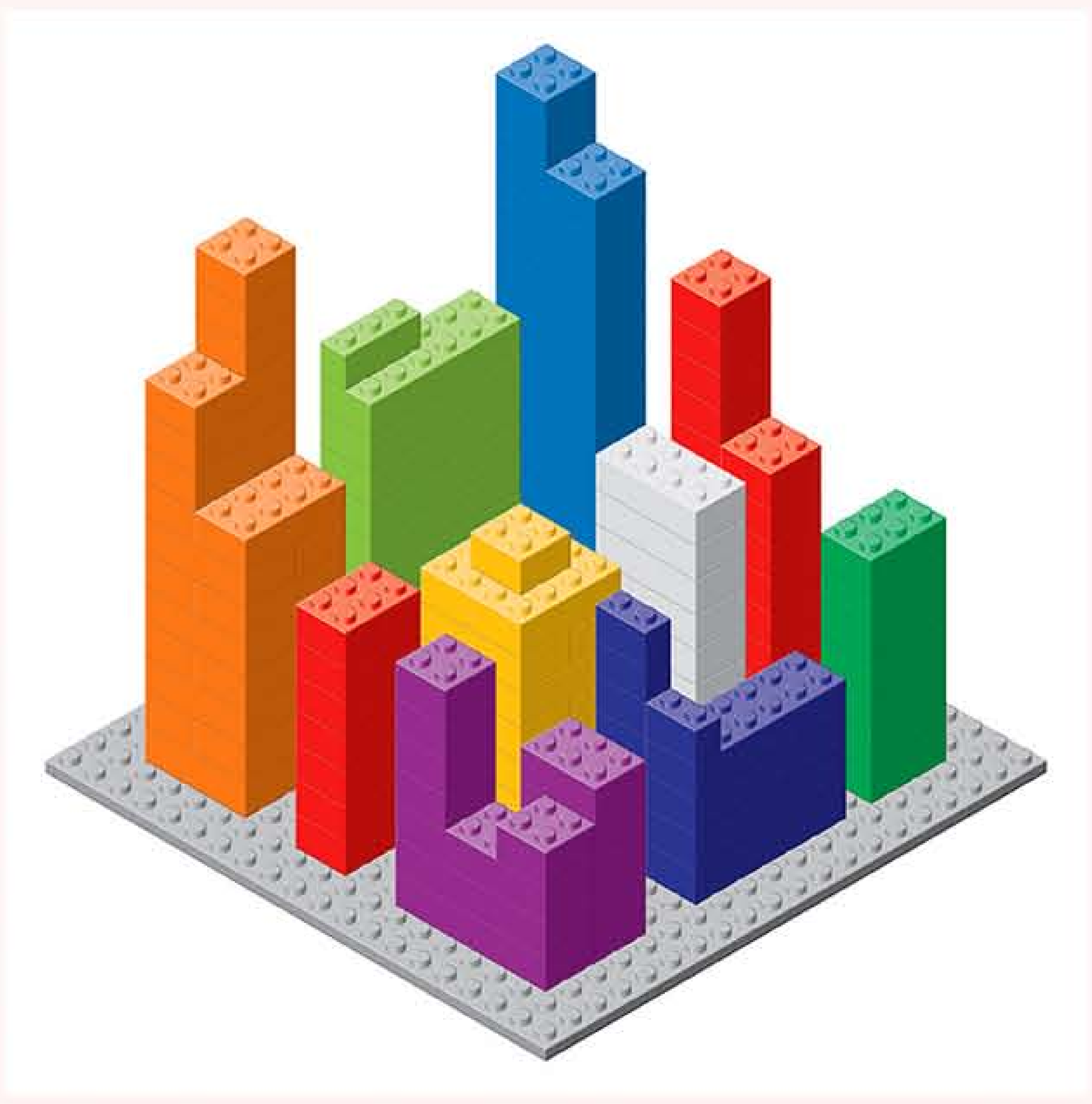
Blog Categories

By Author

Newsletter

Get articles by our experts, and Perforce news monthly Sign up now or view online.

Dec 15, 2015
Anthony Bartlett, Marketing Coordinator at Perforce Software



2015 may well be remembered as the year the "Internet of Things" (IoT) went mainstream. From cars to refrigerators, the tools we use every day are becoming increasingly connected to each other and the wider world. However, as various software failures and hacks have shown, this new connectivity comes at a price.

In most cases, embedded systems are real-time safety or mission-critical systems, e.g. vehicle management, traffic control, and medical equipment. They demand reliability, robustness, availability, and dependability. Hence, a successful deployment of real-time systems depends greatly on low development costs, a short time-to-market, and a high degree of configurability and quality.

This is where Component-Based Development (CBD) can come to the rescue.

In much the same way that individual Lego bricks can be assembled in a myriad of ways to realize a larger design, CBD enables software systems to be assembled from a pre-defined set of components explicitly developed for multiple usages. The introduction of CBD into real-time and embedded systems development offers significant benefits, including:

- **Rapid development and deployment:** Many components, if properly designed and verified, can be reused across applications, offering a substantial reduction in development cost and time-to-market.
- **Reduction of complexity:** Software for a specific application can be configured using components chosen from an existing library to provide exactly the functionality needed by the application.
- **Design evolution:** Components can be replaced or added to the system as needed, allowing for the continuous hardware and software upgrades required by complex, embedded real-time systems.
- **Increased reliability and maintainability:** Each component can be tested independently, making problems easier to isolate and fix. In addition, bug fixes can benefit many projects, and reused components tend to be more stable and mature than any new development.
- **Higher developer efficiency:** Each development team can focus on its own specialized task—domain experts can concentrate on creating components, and integration experts on assembling those components into products.

The Streams feature of Perforce Helix is ideally suited for Component-Based Development. It enables versioning of a single 'body of code' across multiple branches, allowing users a consistent view of the code base. Also, branch views that manage flows of change between these code bases provide a safe, rule-based flow to maintain version stability.

Using Streams, a component's history can be traced, making it easy to identify in all the projects in which it is shared. This shifts the emphasis toward the management of components rather than the components themselves. If components are shared across multiple projects, it is often beneficial for their lifecycles to be independent. With an independent lifecycle, a component can release new functionality and patch updates without affecting the projects that rely upon it. It is then up to the project owner to use the correct configuration of each component version.

Each project will need to effectively subscribe to its required components, as well as reference a specific release. This allows the project to remain isolated even as new component versions are released—it will only adopt newer releases by updating the subscribed components. Not imposing component updates on a project is vital in the embedded software development world as specific versions of software components are often tied to specific hardware releases.

We're adding new functionality to Perforce Helix all the time, so take a look today and see how you can use Streams to incorporate CBD in your own development workflow now. Stay tuned for more details on an out-of-the-box CBD solution.

POSTED IN: **PRODUCT AND INDUSTRY**

Related Articles:

- Component-Based Development with Perforce Helix: Some Assembly Required
- The Thing with the Internet of Things
- Single Source of Truth in Component-Based Development

« Stay Secure for Success with Git in the Enterprise   Helix Swarm's New Search Feature and More... »