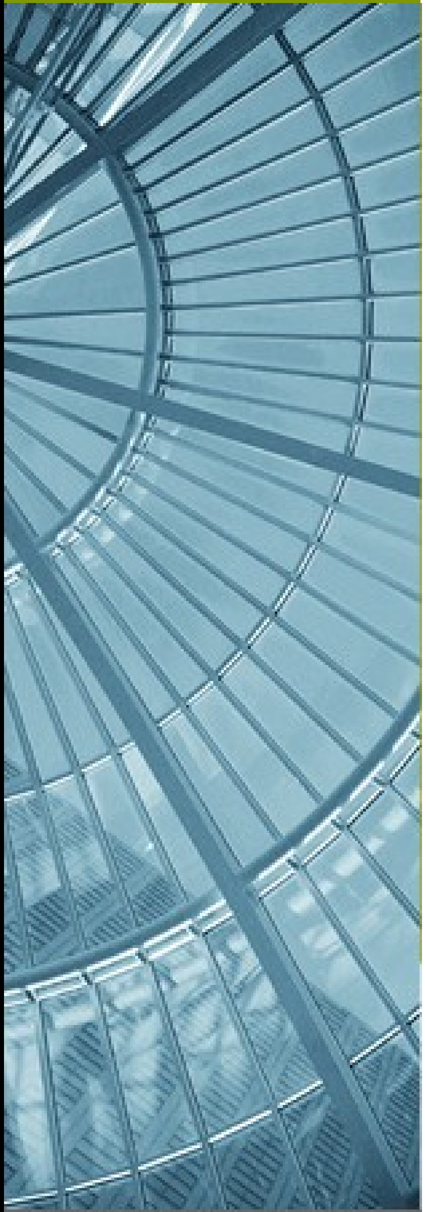


STP Service Creation & Deployment Workflow

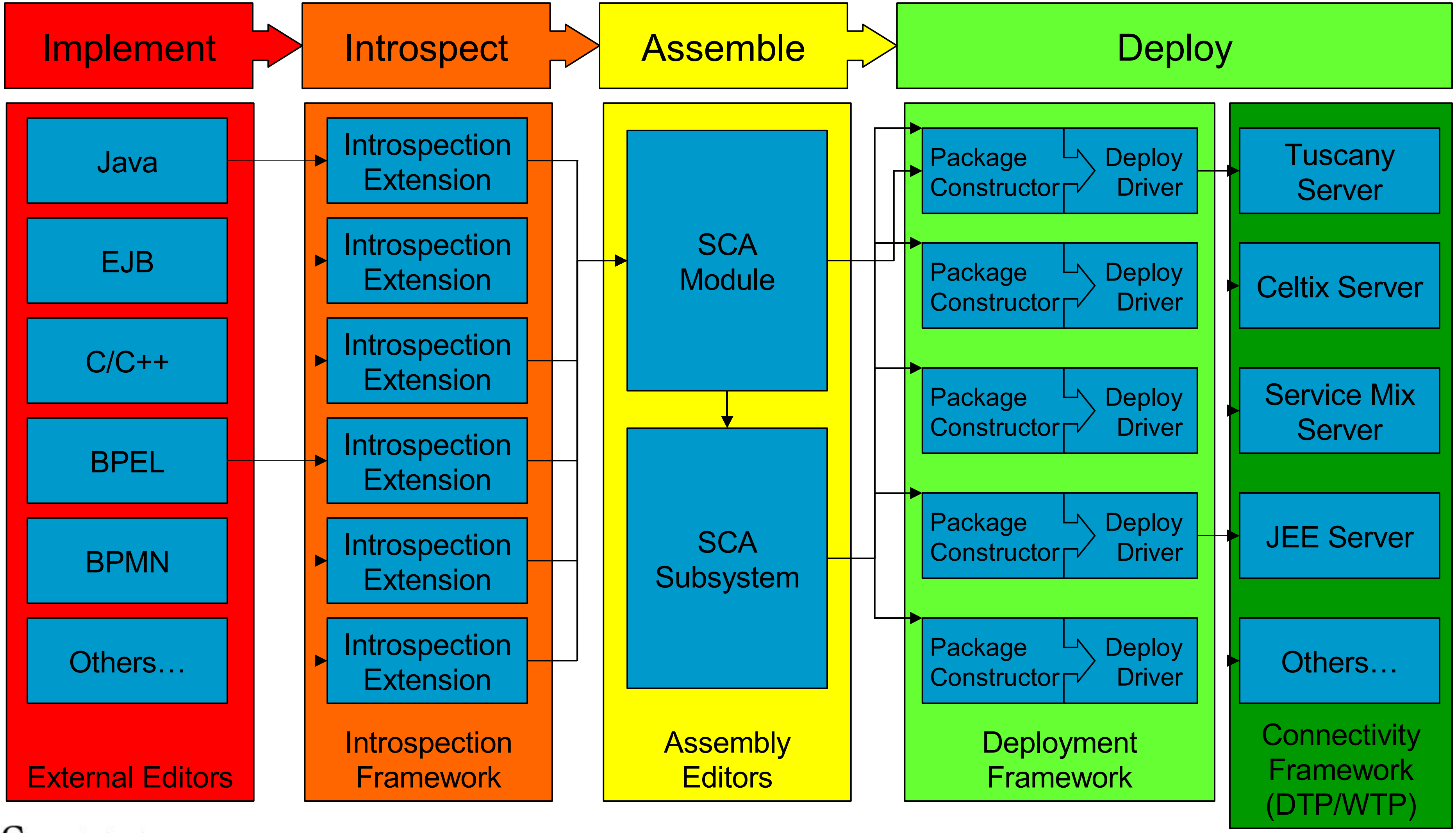
Rob Cernich – April 14, 2006

Purpose

- Define how the larger components within STP interact with one another
- Provide context for deployment framework contribution (how it would be used within the system)
- Provide a starting point for discussion on the overall system architecture
- A bottom-top workflow pattern is used for the example. It is assumed the component interactions will still be present, as described for a top-bottom workflow, however there may be subtle differences and there may be a need to define more extension points

- Disclaimer: This is based on my understanding (assumptions) of the system, which may be 100% inaccurate.

Bottom-Top Workflow



SYBASE

Step 1 - Implement

- User creates service implementation using external (i.e. not necessarily supplied by STP) editor
- Editor does not need to be implemented using any STP provided functionality or extension points
- For integration with STP, addition data may be required for use by the introspection framework (i.e. the raw files may need to be adorned, either internally or externally for proper introspection)

Step 2 - Introspect

- Introspection extension points are implemented for each implementation type, which allow the resources to be used within STP
- Implementations may need to provide additional editing capabilities for supplying out of band data required by introspection extensions

Step 3 - Assemble

- ComponentTypes output from introspection extensions are used to define components within module assemblies
- Module assemblies are used to define module components within subsystem assemblies

SYBASE®

Step 4 – Package/Deploy

- Module assemblies and subsystem assemblies are deployed to servers
- Deployment extensions use information within the assembly to construct a package for deployment
- Deployment extension may or may not support specific implementation types which may be included in an assembly
- Deployment extensions may wish to provide their own extension points to allow third parties to plug-in packaging logic to support their implementation type (for extendable runtimes)

SYBASE