Continuous Release and Deployment Capabilities for CICS Customers

Luis Carlos Silva
Continuous Testing and Delivery for z Systems Offering Manager
lcsilva@ca.ibm.com

Mark Cocker
CICS Technical Strategy and Planning
mark_cocker@uk.ibm.com

Stay ahead.

Customer's Release and Deploy Challenges Impact Their Entire Business

CHALLENGES

Costly, error prone manual processes and efforts to deliver software across an enterprise

Slow deployment to development and test environments leave teams **waiting and unproductive**

Upgrade risk due to managing multiple application configurations and versions across servers

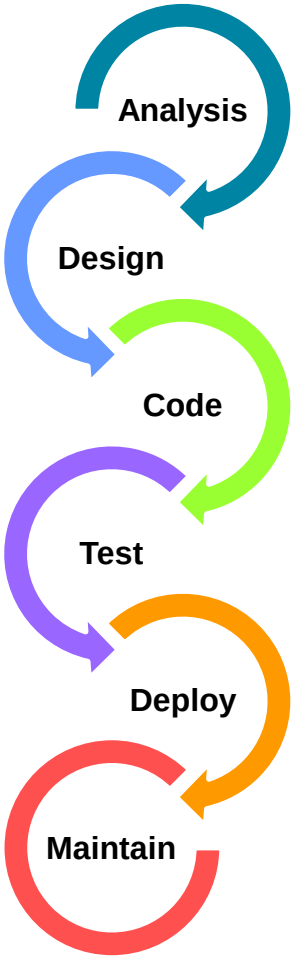
Knight Capital lost **$440 million in 45 minutes** due to a misconfigured release

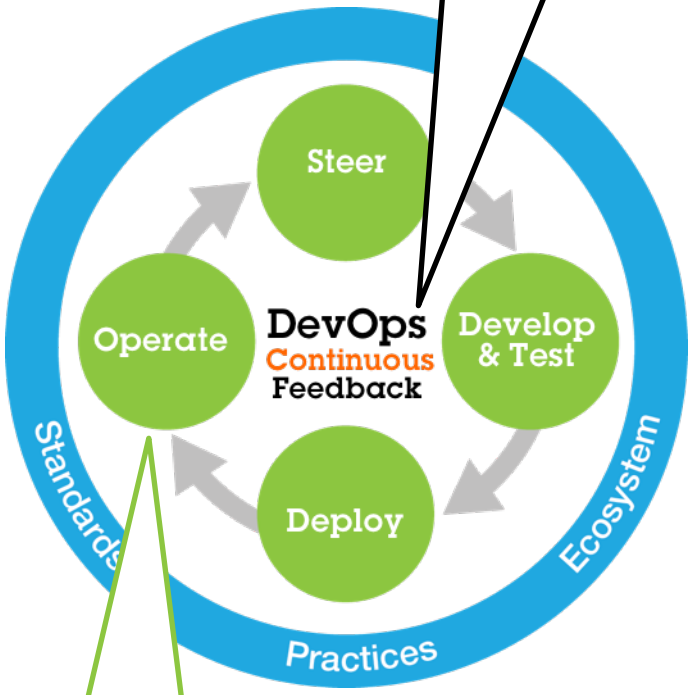
New Zealand's biggest phone company, Telecom paid out **$2.7 million** to some **47,000 customers who were overcharged** after a software glitch

A bad software upgrade at RBS Bank left **millions unable to access money for 4 days**

Transformation is key

By the end of 2015, 75% of large organizations are expected to have adopted agile DevOps practices (IDC)

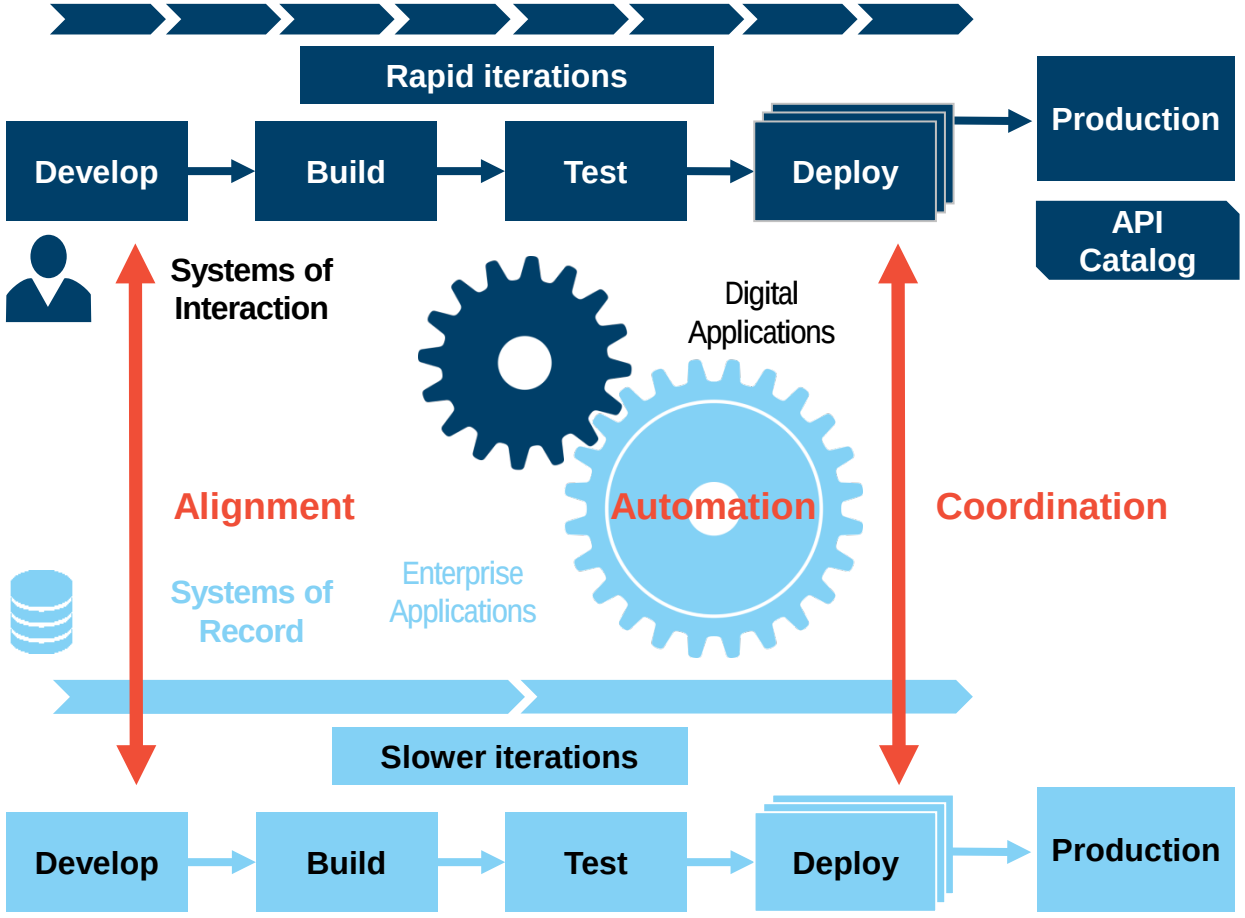




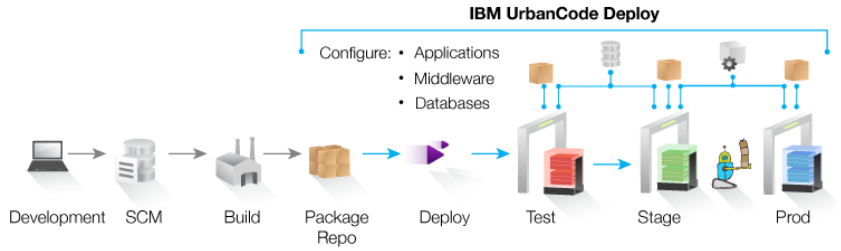
The **key difference** being, there is no one team or group at the center. Rather all dev teams get elevated as active (equal) participants throughout the entire process.

Operations is now a first class citizen and also an active (equal) participant throughout the entire process.

Release and Deploy Challenges in a Variable Speed IT



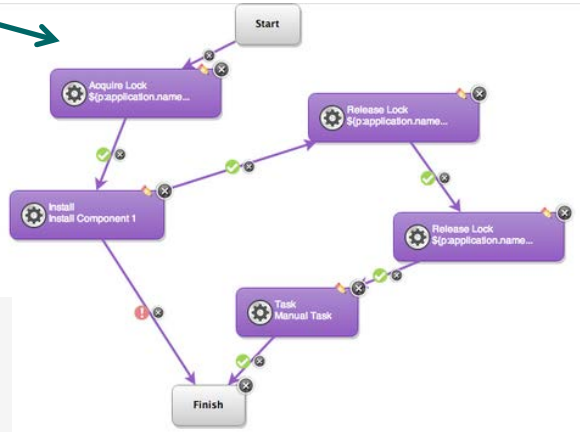
IBM UrbanCode Deploy



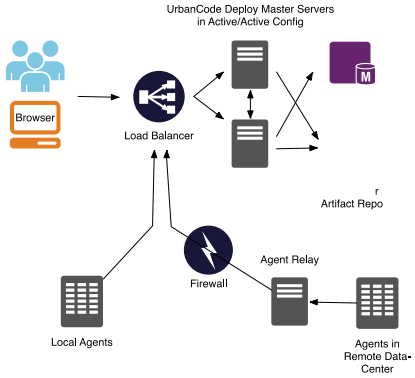
Continuous Delivery
Across Environments

Re-useable / Extensible
Integrations & Workflows

Easy to use process designer



IBM UrbanCode Deploy



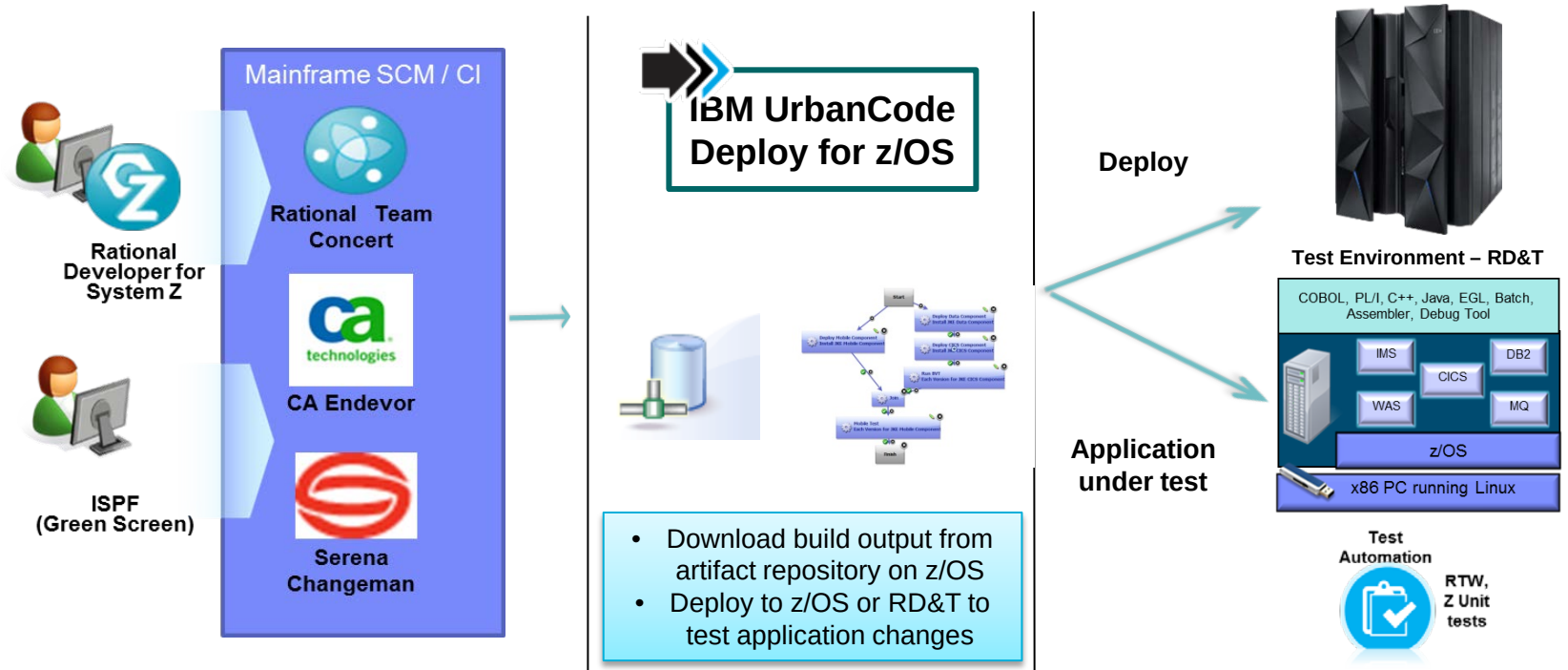
Scalable Architecture

1.2.3

System of Record
Everything is versioned & auditable

Push Button Deployments
Role based security & gates

Continuous Release and Deploy for the z/OS



- Provides a **unified solution** for continuous delivery of heterogeneous enterprise applications
- **Accelerate delivery** and reduces cycle time to develop/test multi-tier applications across heterogeneous environments and platforms
- **Reduce costs** and eliminate delays for delivering mainframe applications
- **Minimize risk** and improve productivity across disparate teams with cross-platform release planning

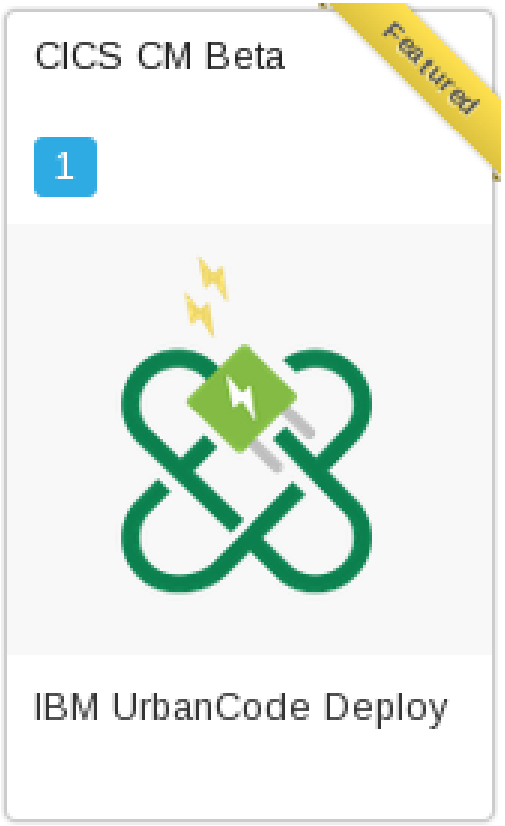
UrbanCode Deploy Mainframe Capabilities Overview

- z/OS and Linux on System z platform support
- Incremental deployment and rollback
- Security features: Impersonation, PassTicket authentication
- Deployment of sequential data sets
- Deployment of data set deletions
- Deployment of PDS
- SCM agnostic artifact packaging
- z/OS artifact repository in USS
- Deployment of Java applications to USS
- Support for UCD CodeStation
- RTC build integration
- z/OS Utility plug-ins for TSO/ISPF/MVS command, JCL, REXX and PDS processing
- Deployment support for:
  - DB2 for z/OS
  - IMS
- Deployment and configuration support for CICS
- Deployment and configuration support for WAS for z/OS

UrbanCode Deploy CICS CM Plug-in

- IBM CICS Configuration Manager is a tool for managing and maintaining CICS Transaction Server definitions. It provides audit, reporting and lifecycle change management control facilities to support the build, management, and deployment of complex mainframe CICS applications.

- The CICS CM plug-in for UrbanCode Deploy can be used to automate the resource management process. The resource management process can be simplified by modelling the processes, that might be complex, across each of your environments. For example, these environments might be development, test, and production in different data centers.



UrbanCode Deploy CICS TS Plug-in

Overview

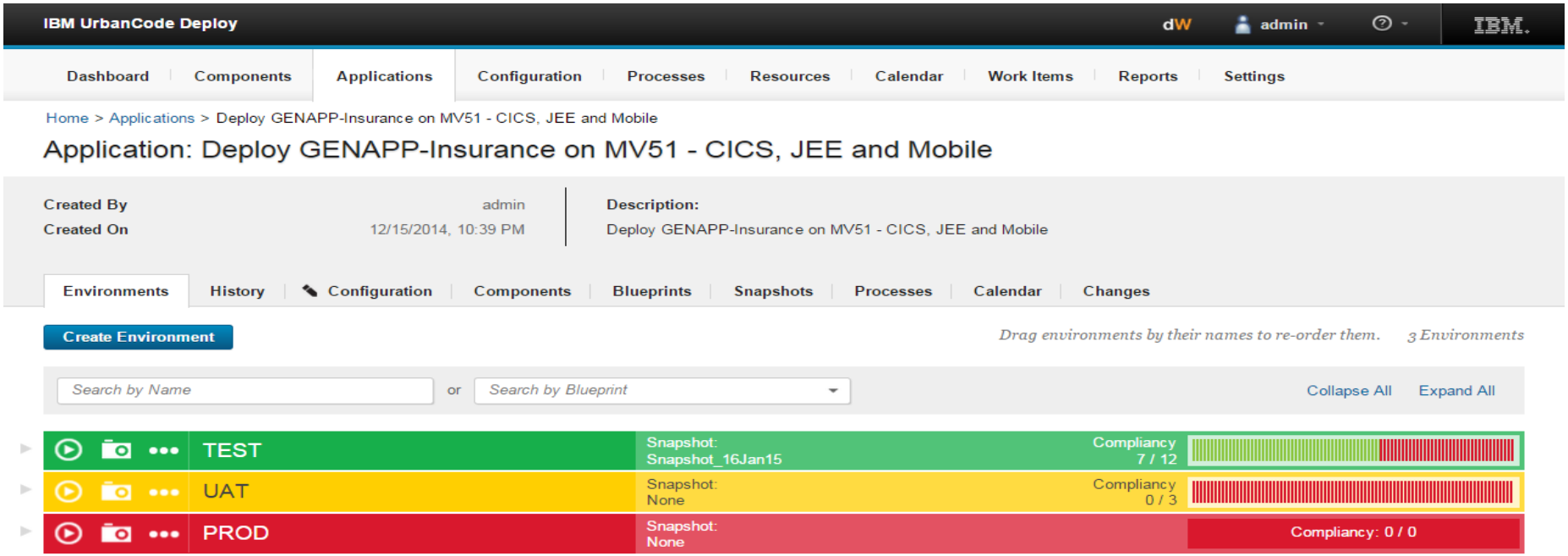
- IBM CICS Transaction Server for z/OS provides scalable, general-purpose, transaction processing solutions for high-volume updates to shared data. Businesses have used CICS to differentiate themselves by creating, extending, and evolving their critical applications.
- You can use the CICS TS plug-in for UrbanCode Deploy to automate the deployment and undeployment of CICS applications and CICS bundles.
- In a wider context, the CICS TS plug-in can be used in conjunction with other CICS tooling to automate the build and deployment of CICS artifacts as part of a continuous delivery environment.

Features

The plug-in includes steps to:

- Install CSD resources, groups, and lists
- Install BAS resources, resource descriptions, and groups
- Discard resources
- Enable and disable resources
- Open and close resources
- New copy and phase in resources
- Make resources available and unavailable
- Check the available, enable, and open status of resources
- Scan pipelines
- Deploy and undeploy a bundle
- Deploy and undeploy an application
- Check the status of an application
- Enable and disable an application
- Make an application available and unavailable

Deploy the same application to different environments



Demo

www.ibm.com/software/products/en/ucdep

Flexible and Secure

- Easily integrate into existing DevOps pipeline with SCM/build agnostic packaging utility
- Artifact repository “on the box” to ensure data governance and compliance
- Design dynamic and versioned release processes

Transparency and Control

- Standardize release planning, tracking, and deployment governance across disparate teams and platforms
- Reliably package, deploy and promote incremental native file system changes across environments
- Confidently track “what is where” and rollback to any prior version at ease

Extensible

- Streamline deployment process with out-of-the-box z/OS utility integrations
- Support for middleware application deployment and configuration for WAS
- Leverage existing deployment scripts or extend with intuitive plugin framework

Achieving Faster Time to Market with a 482% ROI

The Total Economic Impact™ of IBM UrbanCode Deploy

Forrester Consulting interviewed four IBM UrbanCode Deploy customers, gathered and analyzed the data, and created a "composite organization" based on the insight

FIGURE 1
Financial Summary Showing Three-Year Risk-Adjusted Results

ROI: 482%	NPV: $3,053,639	Delivery automation resources saved: at least 30%	Time-to-deployment: as much as 75%

Source: Forrester Research, Inc.

FORRESTER

3 Year Benefits:

- **IT operations resource savings** – *$430k in allocated resource headcount*
- **Faster time-to-market with improved productivity** – *$2.65m in savings over 3 years*
- **Reduction in risk of failed deployments** – *$608k in savings from reduced failed deployments*
- **Reduced cost per release by 97%**
- **Improved employee satisfaction** by eliminating repetitive manual processes and easing the workload
- **Improved scalability and increased transparency** into the release process

"The cost benefit to us of implementing UrbanCode was a reduction of 97% in the cost of a release. From an average of eight people working for 3 hours for every release, it's now one person working 10 to 15 minutes." ~ *Enterprise architect, global specialist insurer*

Download the commissioned study conducted by Forrester Consulting
https://ibm.biz/urbancodeTEI

Investment	Savings & Benefits	
$633,427	$3,687,066	~5X Payback

Under Consideration for Future Releases

Expanding DevOps Solutions for Enterprise Systems

- DB2 z/OS as option for UCD server's database
- Automated rollback
- Deployment of non-file (virtual) resources
- Tivoli Workload Scheduler Plugin
- MQ z/OS Plugin
- z/OSMF Plugin
- Deployment of CICS Java applications

Additional Information

- Forrester's study Total Economic Impact of UrbanCode
- Webinar: *Agility, Velocity and Innovation: Adopting DevOps for 2-Speed IT*
- Booklet: *Mobile to Mainframe DevOps for Dummies book*
- White Paper: *IBM UrbanCode Deploy, Docker and Linux on z Systems*
- YouTube: *DevOps for System z*
- YouTube: *z/OS Application Build and Deployment RTC + UrbanCode Deploy 6.1*
- YouTube: From BlueMix to Mainframe with IBM UrbanCode Deploy
- YouTube: Multi-platform Deployment Automation with IBM UrbanCode

Questions?

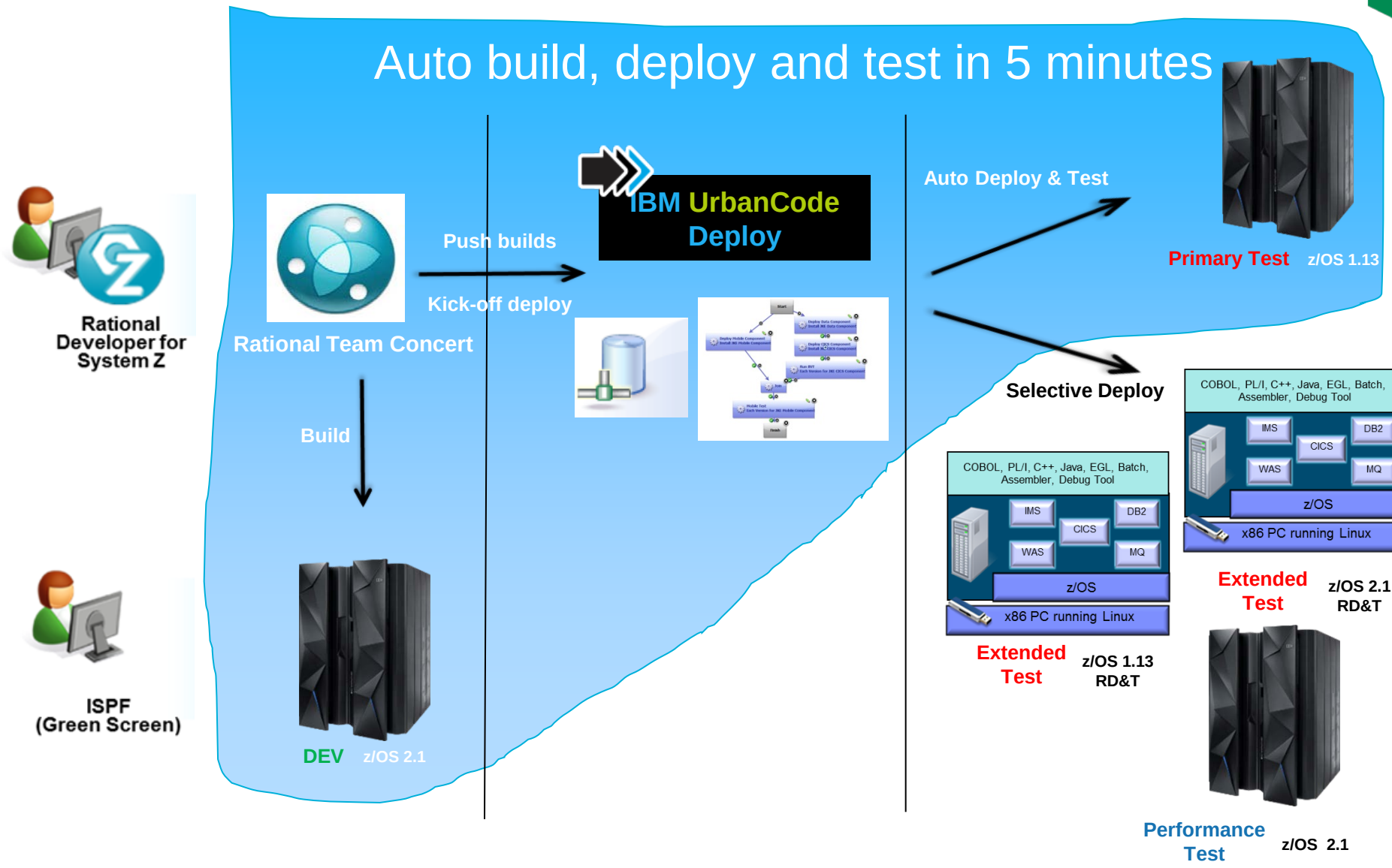
www.ibm.com/software/products/en/ucdep

Thank You!

www.ibm.com/software/products/en/ucdep

Backup

Reference Case – An IBM Team



Challenges for the team (1)

- Deploying and Testing a build takes long time
  - Installing the SMP/E build takes hours
  - Testing takes another couple hours
  - There's also a zip build which doubles the install effort
  - SVT has to test in multiple z/OS environments

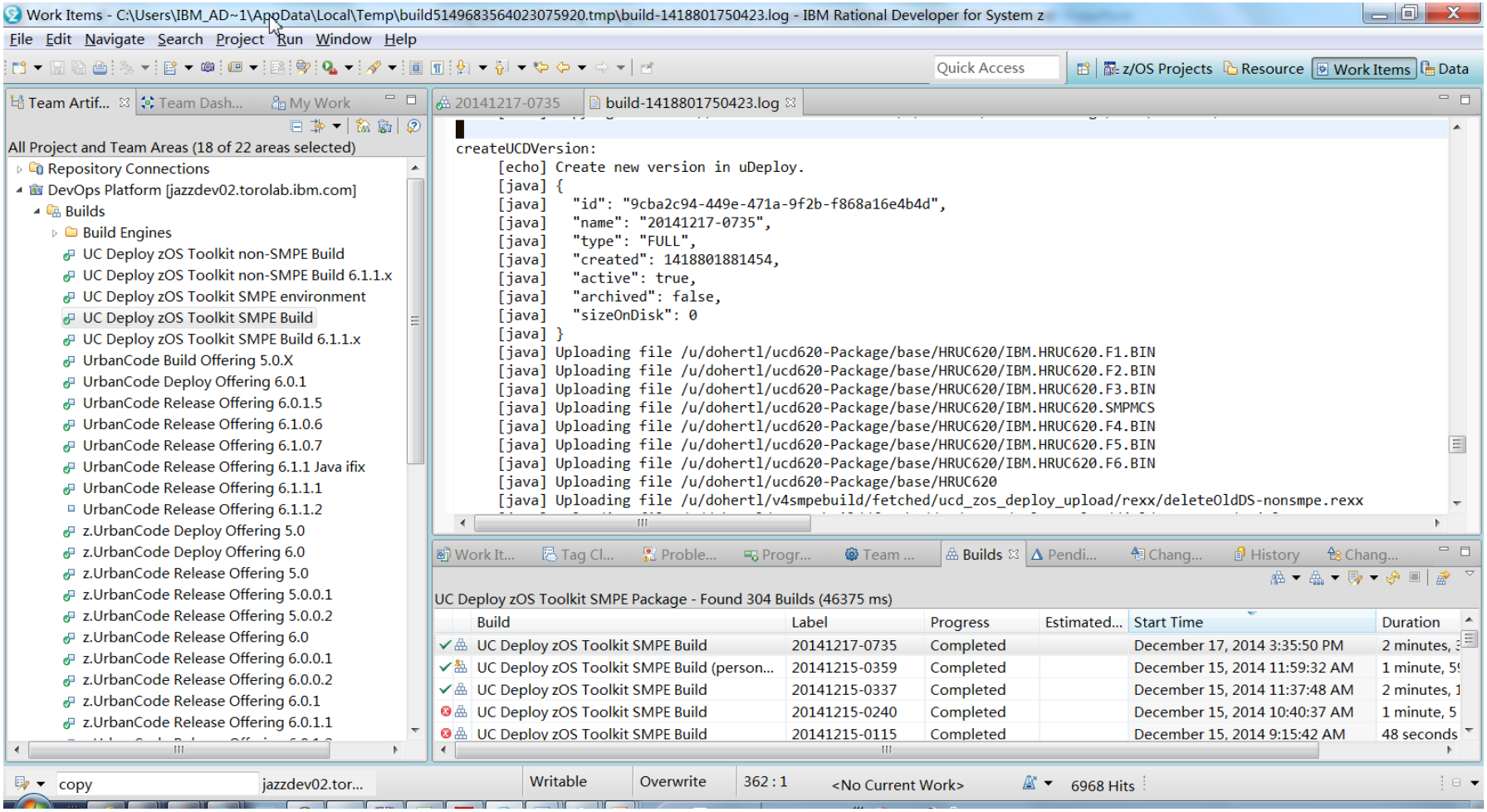
Challenges for the team (2)

- Difficult to track deployed versions
  - Many times, we asked around the team ‘what version is installed?’ before doing bug verification and testing

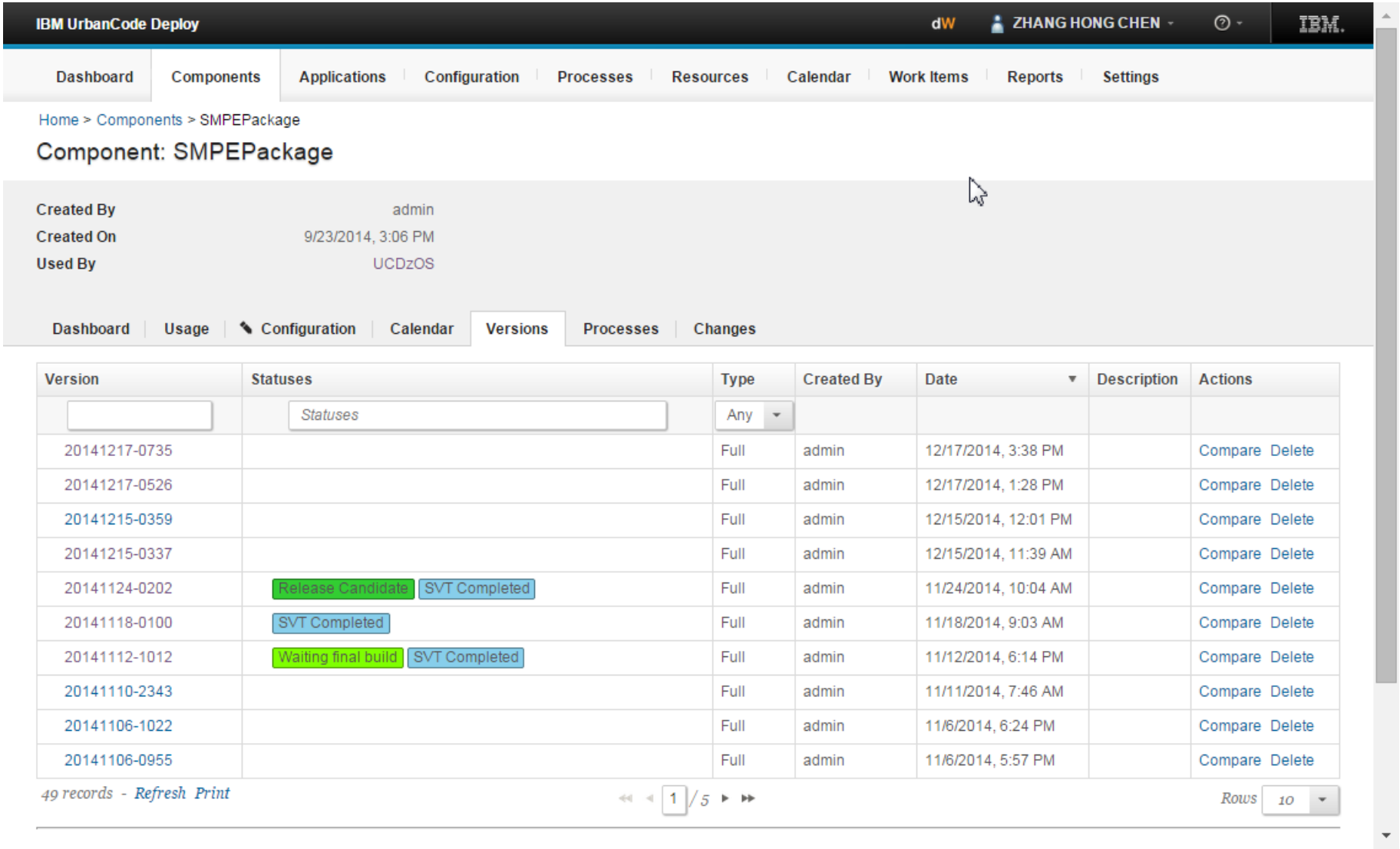
Challenges for the team (3)

- Git doesn’t work very well for z/OS
  - Git is the SCM for UCD product. The z team inherited the infrastructure at the beginning
  - Build using JCL and scripts in z/OS, developer has to sync up code with Git manually
  - We need a real z SCM & Build tool

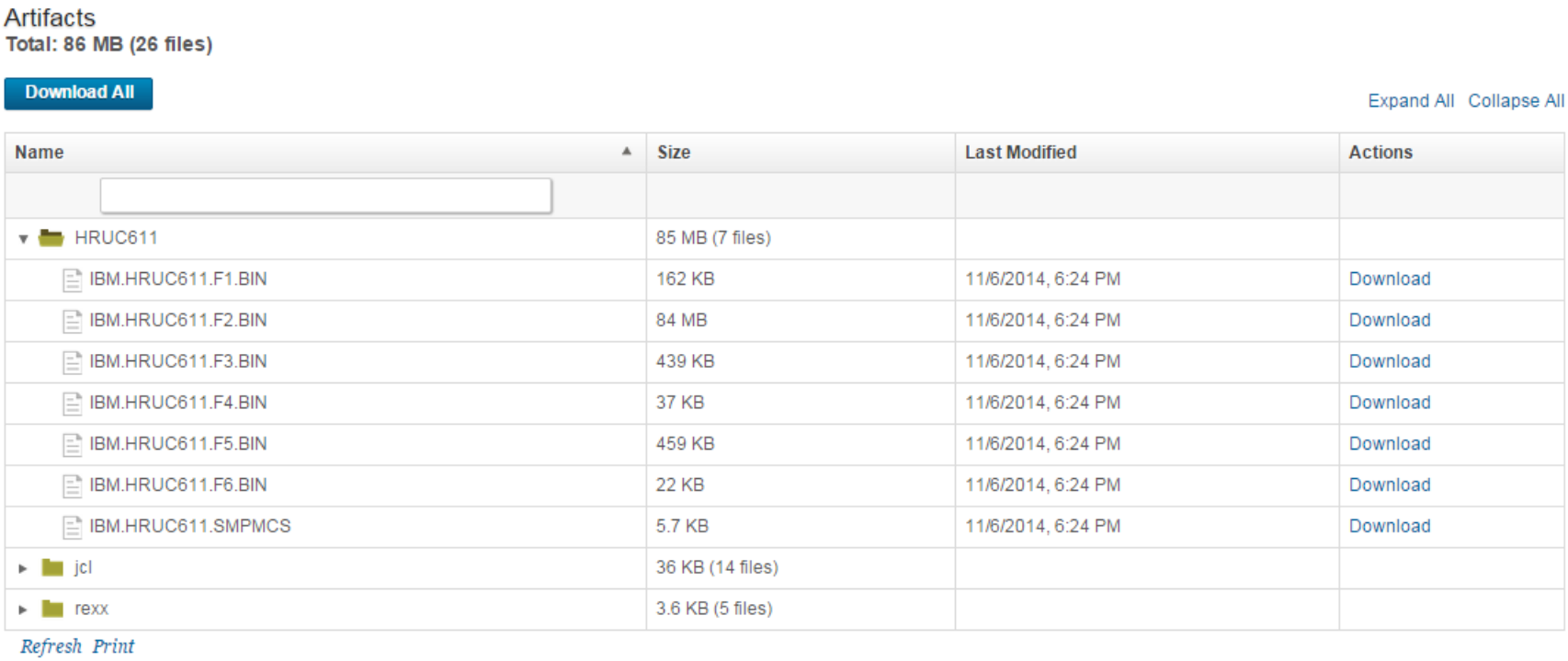
Build for every change in RTC



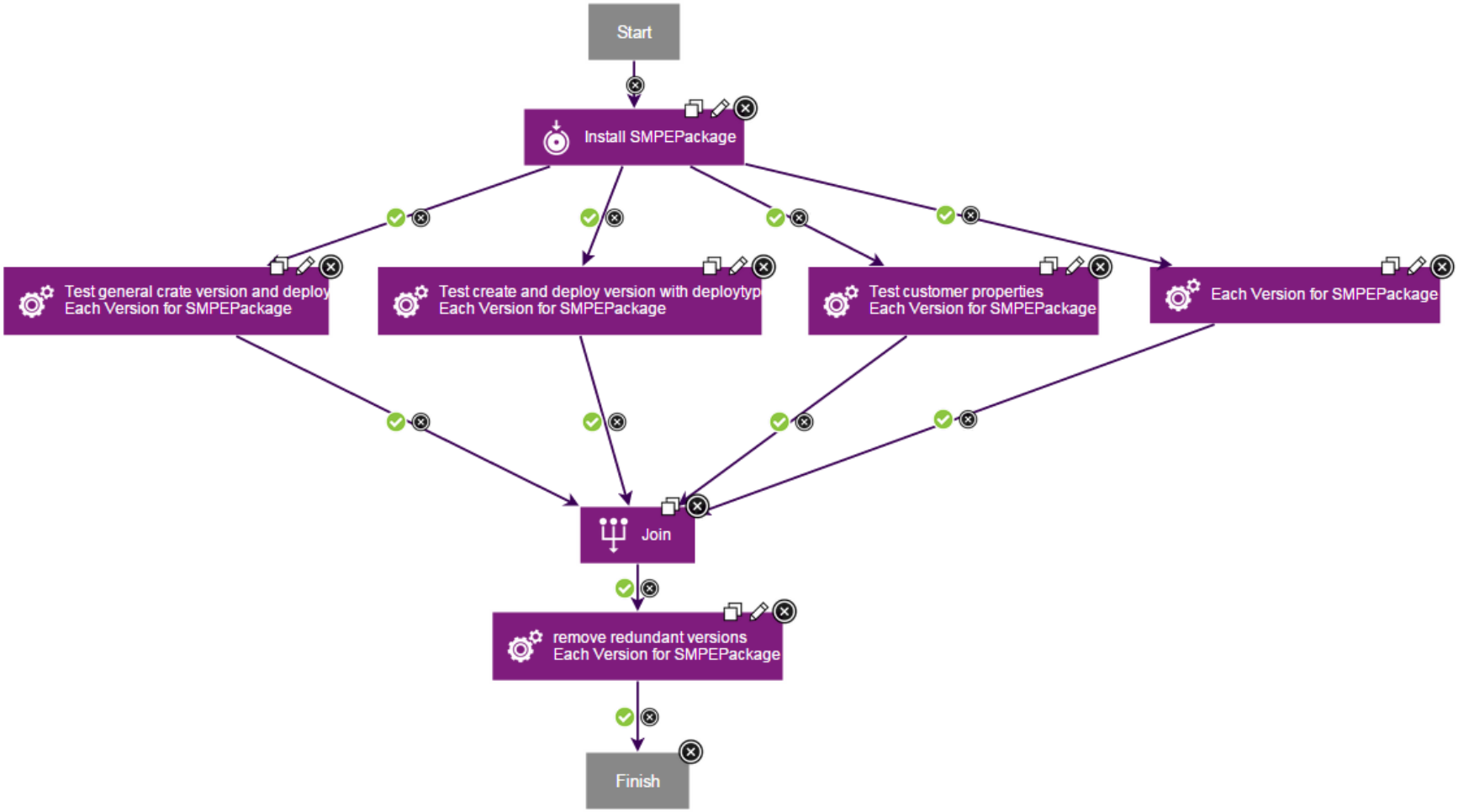
Each build creates a deployable version in UCD



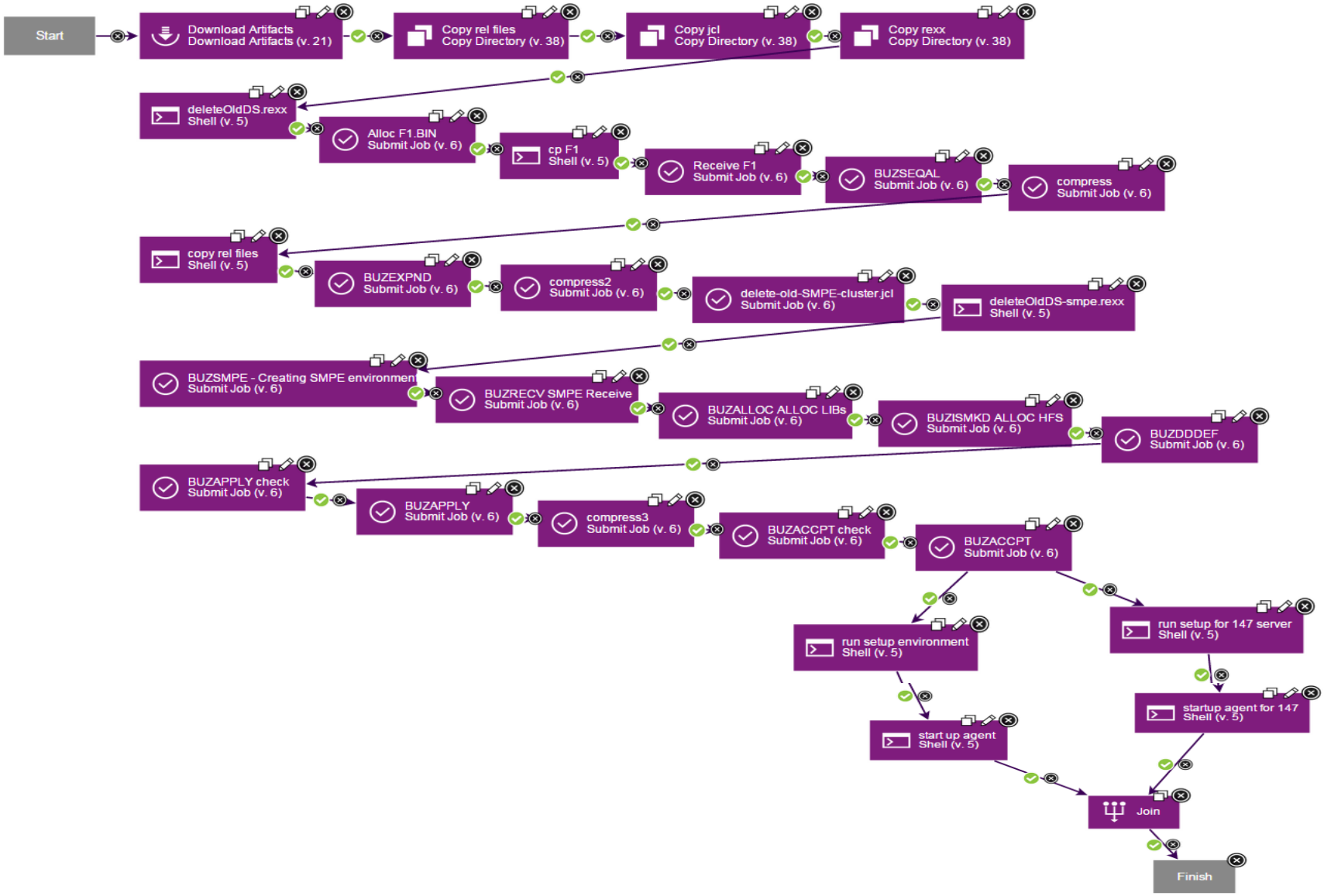
Artifacts in a version



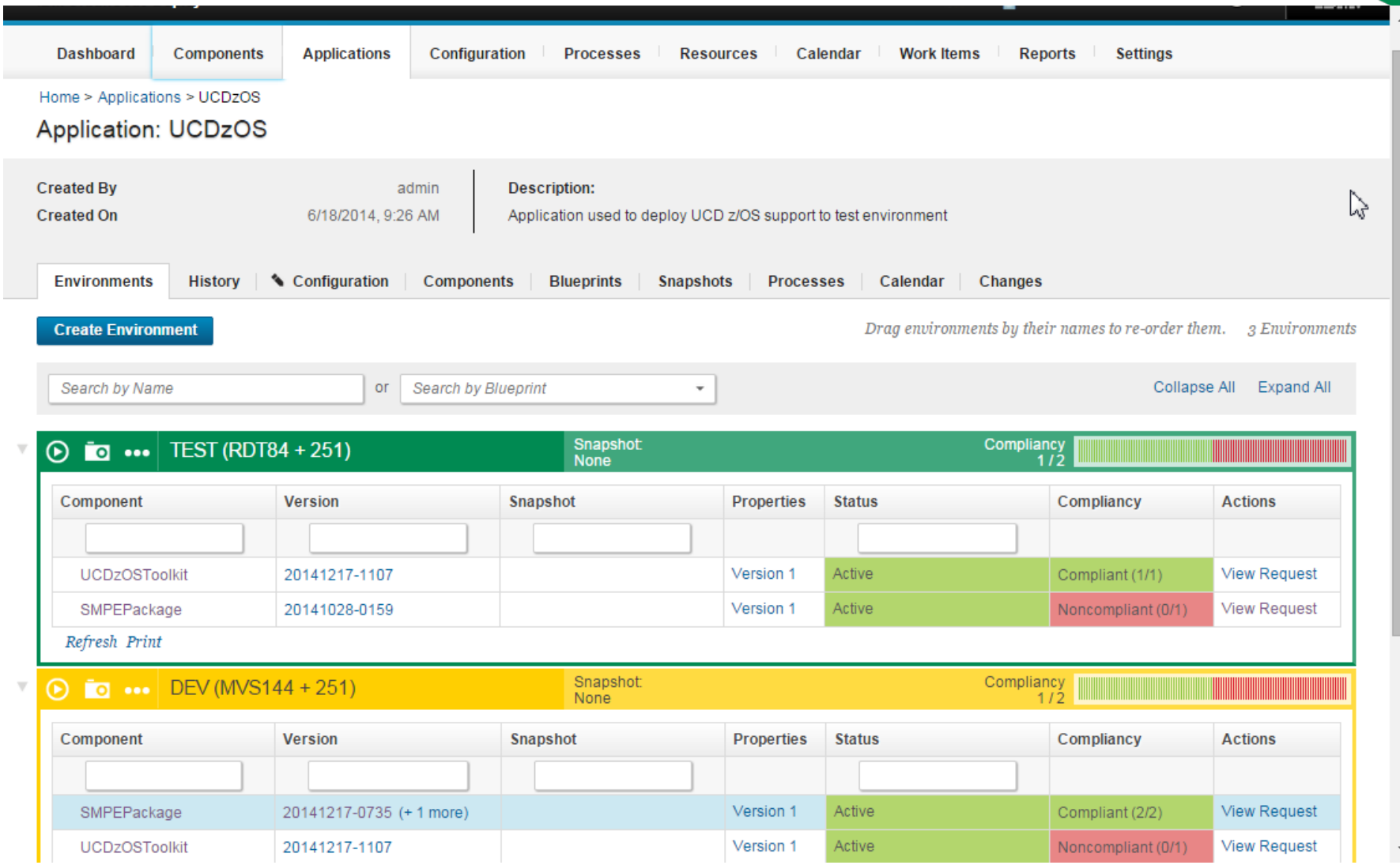
Application deploy process (deploy + test)



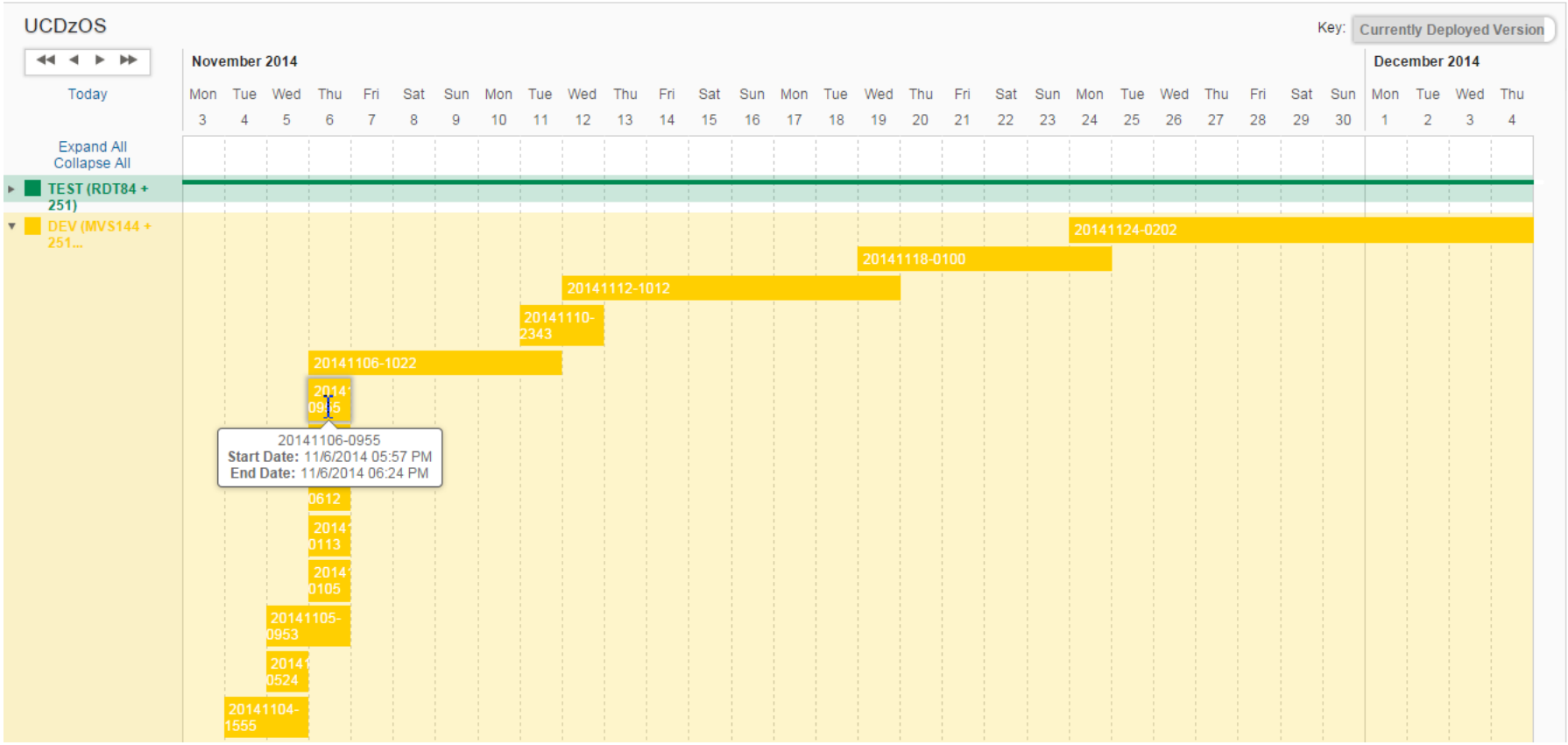
Deploy process (Install SMP/E package)



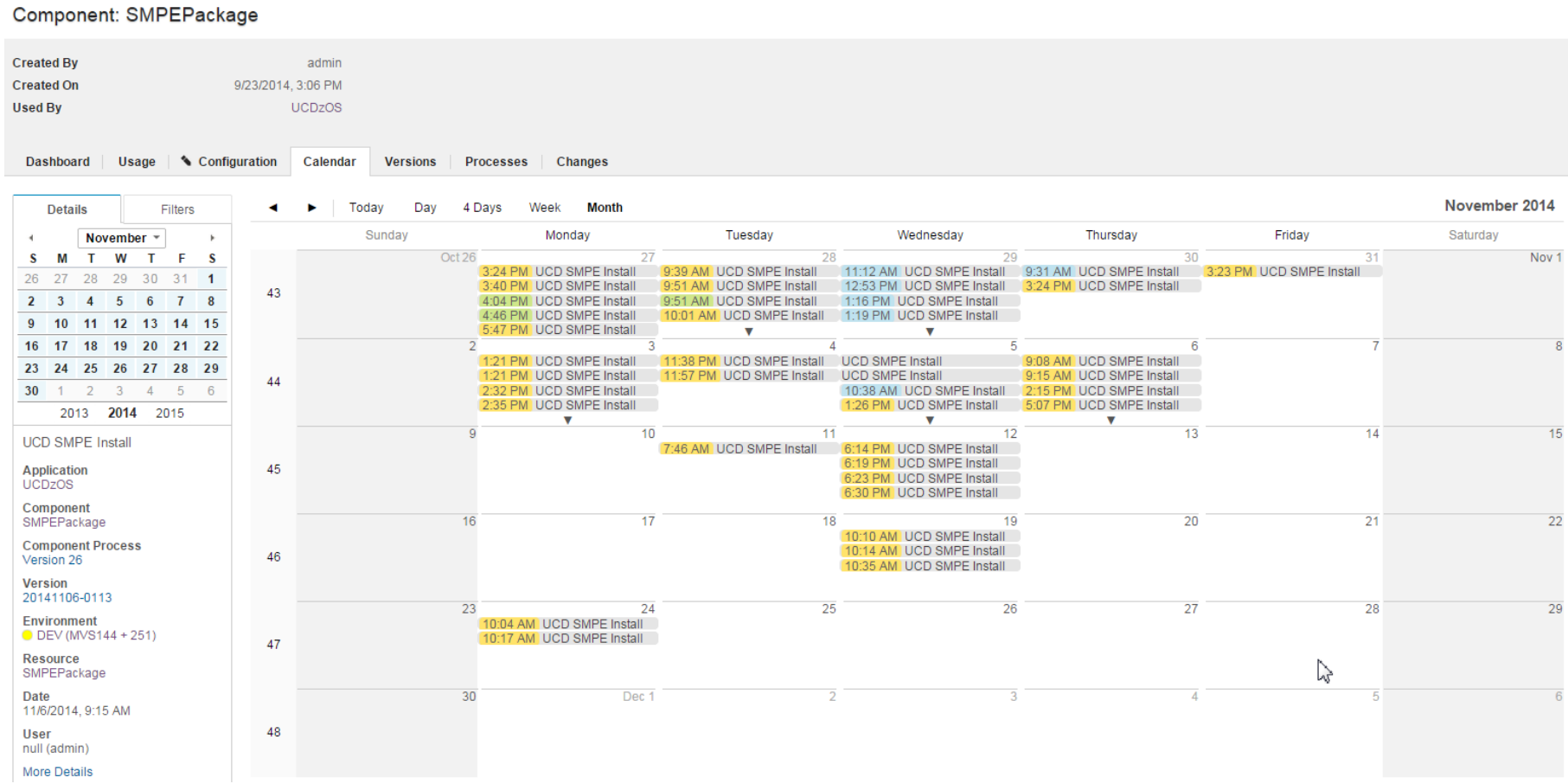
What's deployed where



Usage View – See the deploy history



Calendar View



The benefits of the solution

- Deploy time greatly reduced
  - Build, deploy, build verification test

    5 minutes compared with several hours before
  - No manual deploy effort at all
  - Up to 7 deployments a day in our past release development
- Verification test effort greatly reduced
  - Automated build verification, failures will be notified by email
- Less/no conflicts
  - Before we have this, we merge multiple changes into one deploy, leading to conflicts and complexity in problem diagnosing
- The real benefit: developer now focus on developing new features and fixing bugs. Much more productive and enjoyable

Lessons learned, recommendations, tips and techniques, etc.

- Smart development infrastructure doesn't come for free
  - Effort to build the process. Taking a day to turn the manual deploy steps into a process. Revised a number of times after the initial setup
  - Effort to build the test process and test automation. Bigger then deployment automation
- Build your process so that it can be re-run when failed
  - Continue from failed step is difficult
- Include deploy scripts in your version
  - JCL, REXX etc
  - Infrastructure as code practice
- Plan the network
  - Build environment -> UCD
  - UCD -> deploy targets