



Best Practices for Model-Driven Approach to Application Release

Chris Doucet

Overview

- The Evolution of Maturity
- Benefits
- Best Practices
- Getting Started

Maturity Level 1: Manual Approach

A person involved at every process stage

- Slower
- Human processes = Error-prone processes
- Inconsistent results
- Less visibility into what's really going on



Maturity Level 2: Scripted Approach

- Difficult to maintain
- Brittle / Not flexible
- Not visual
- Not widely understood, documented, or transferred

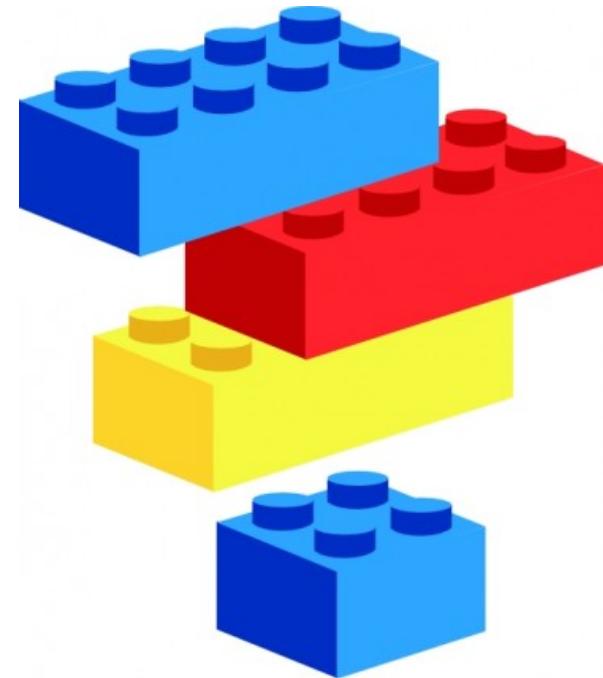
BRACE YOURSELF



Maturity Level 3: Model-Driven Approach

The next phase of the evolution:

- Model out pieces of release and process
- Hook up the models to build solution
- Add to or change out pieces as needed
- Reusable components
- Predictable outcomes

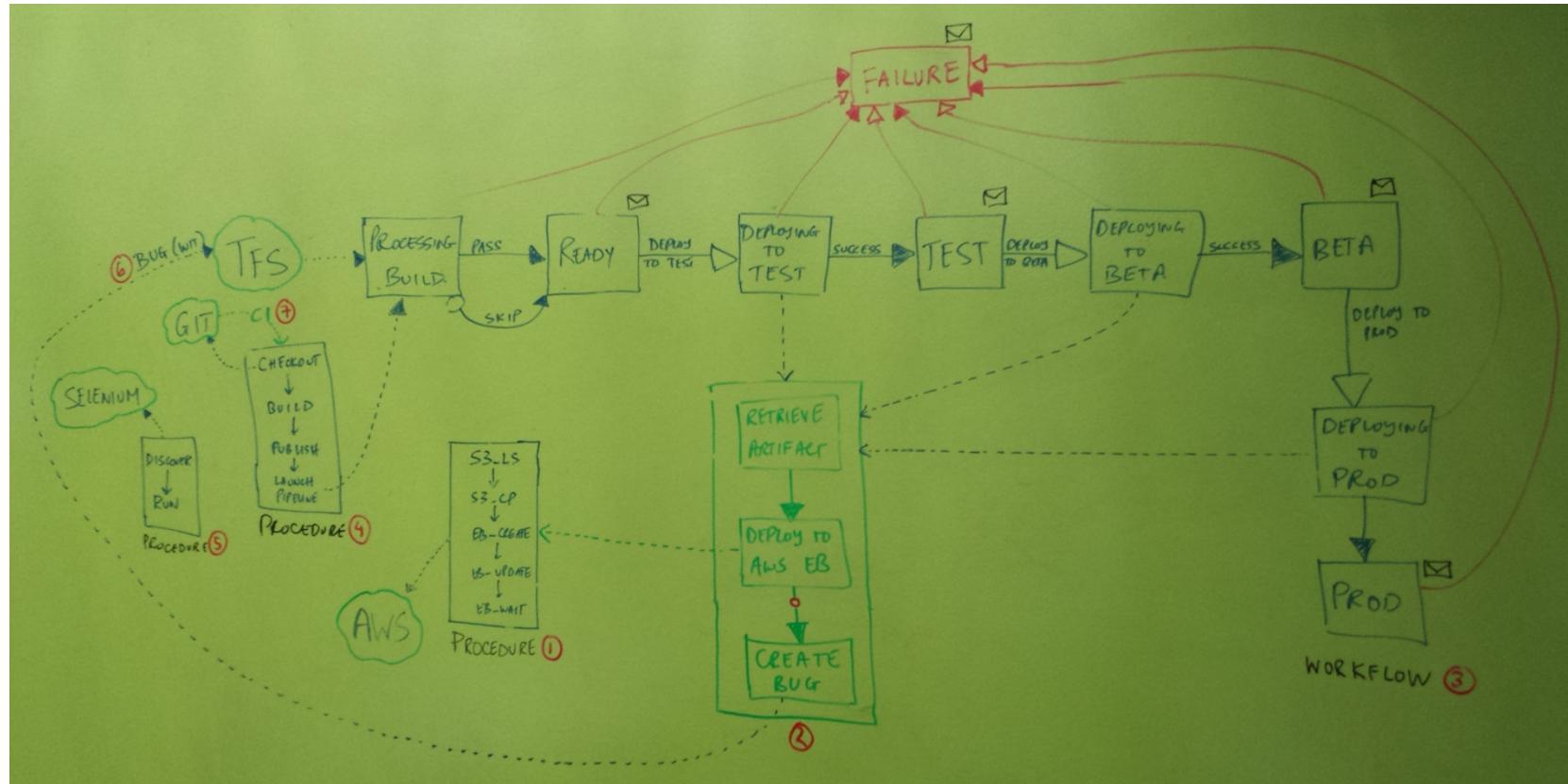


Benefits - The ‘-ilities’

Model-driven ARA benefits:

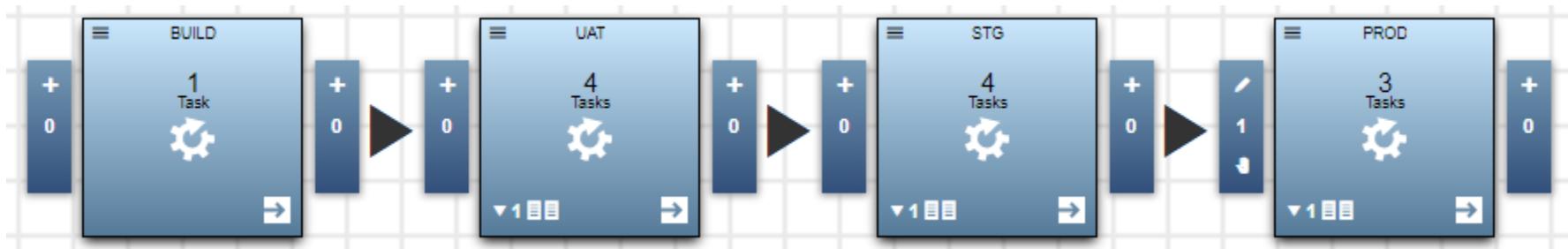
- Increased understanding
- Visibility
- Flexibility
- Consistency
- Quality
- Auditability
- Reusability
- Maintainability
- Reliability
- Simplicity
- Speed
- Control

Best Practices: Capture Existing Process



Best Practices: Value Stream Mapping

Map your value stream and create models to visualize your process, achieve lean principles, enforce standardization (consistent quality), and control & improve the outcomes of your releases over time



Best Practices: What to Model

What: The components that comprise your deployments

- Applications
- Artifacts
- Dependencies

Where: The environments to where you are deploying

- On-Prem
- Virtual
- Cloud
- Containers

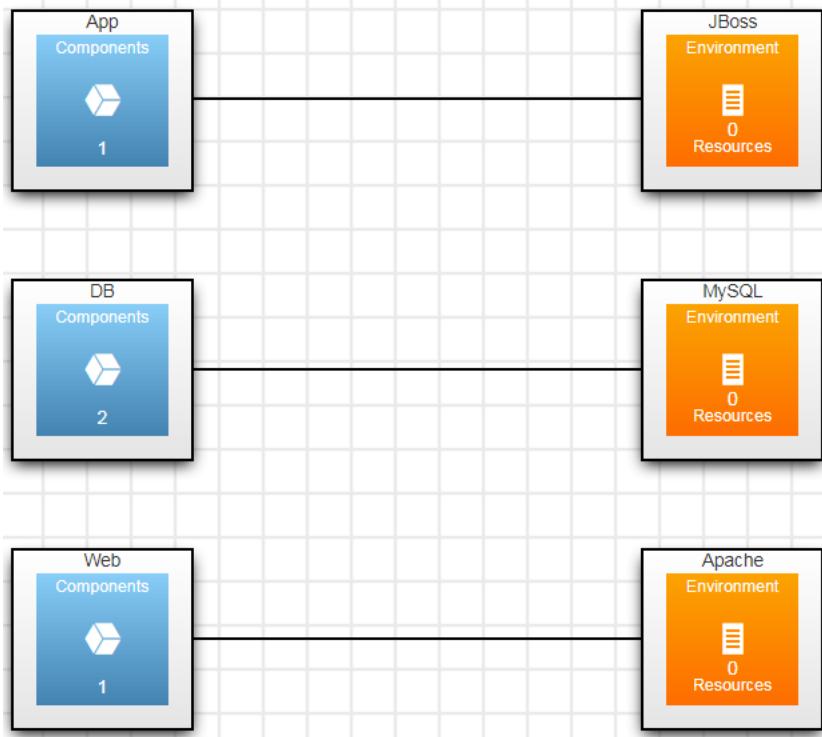
How: The process by which you deploy your components to environments, including gates, approvals, audit trail, etc.

- Deployment processes
- Pipelines
- Releases

Best Practices: How to Model

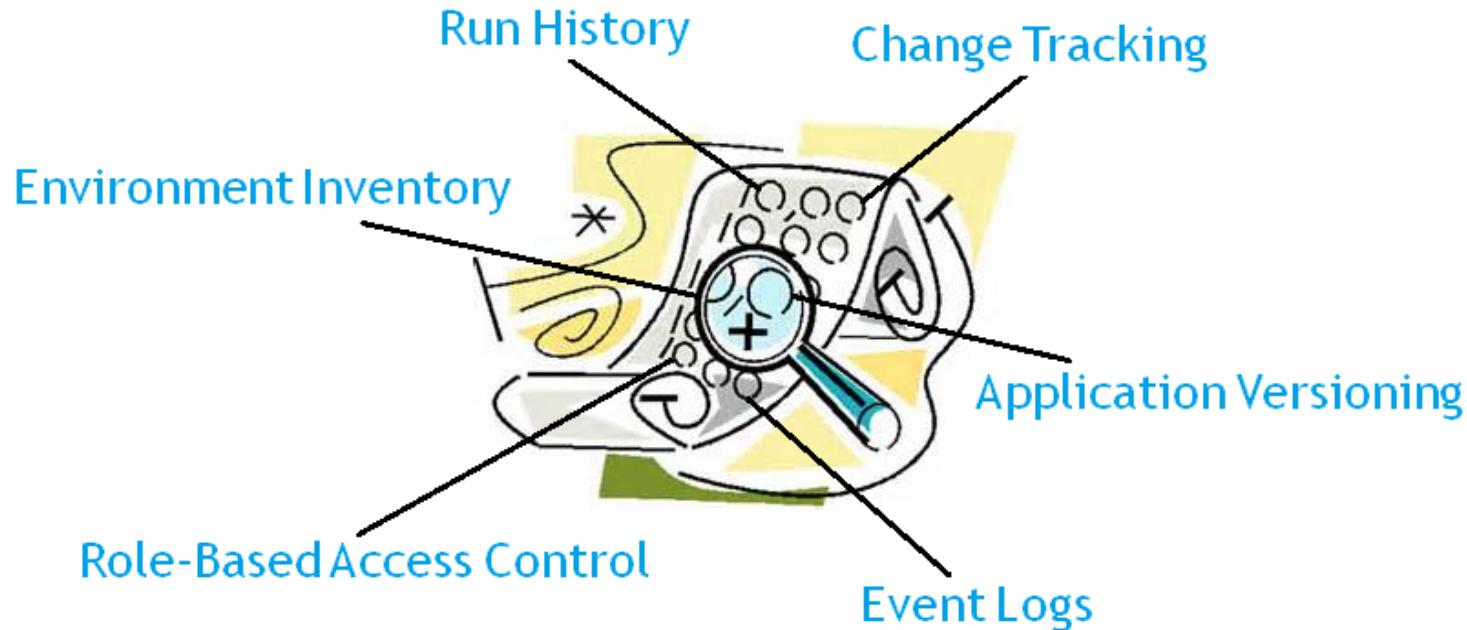
Modeling exists in many forms, from documents and spreadsheets to modeling tools, but are not practical because they are **not runnable**.

Application Release Automation (ARA) platforms are the best method for modeling the **What**, **Where**, and **How**, as well as incorporating the **Who** and **When**.



Best Practices: Bring in Compliance & Governance

Use models to show intent. Run them to show built-in governance and audit trails.



Best Practices: Build for Reusability

Model the components of your release process for maximum practical reusability

- Runtime Parameters
- Metadata References
- Component Templates
- Self-Service Catalog Items

2  Parameters

 1/1 Required 



AWS Dynamic Environment
Create a new environment by dynamically provisioning and configuring VMs on Amazon EC2 [More...](#)

[Provision](#)



Docker Based Microservice
Create a simple microservice application model to deploy a docker container to an environment with Amazon ECS or Google GCE [More...](#)

[Create](#)



Multi-stage CD Pipeline
Create a Pipeline with Dev, QA and Production stages with approval gates to manage path to production from Dev commit to Production deployment [More...](#)

[Create](#)

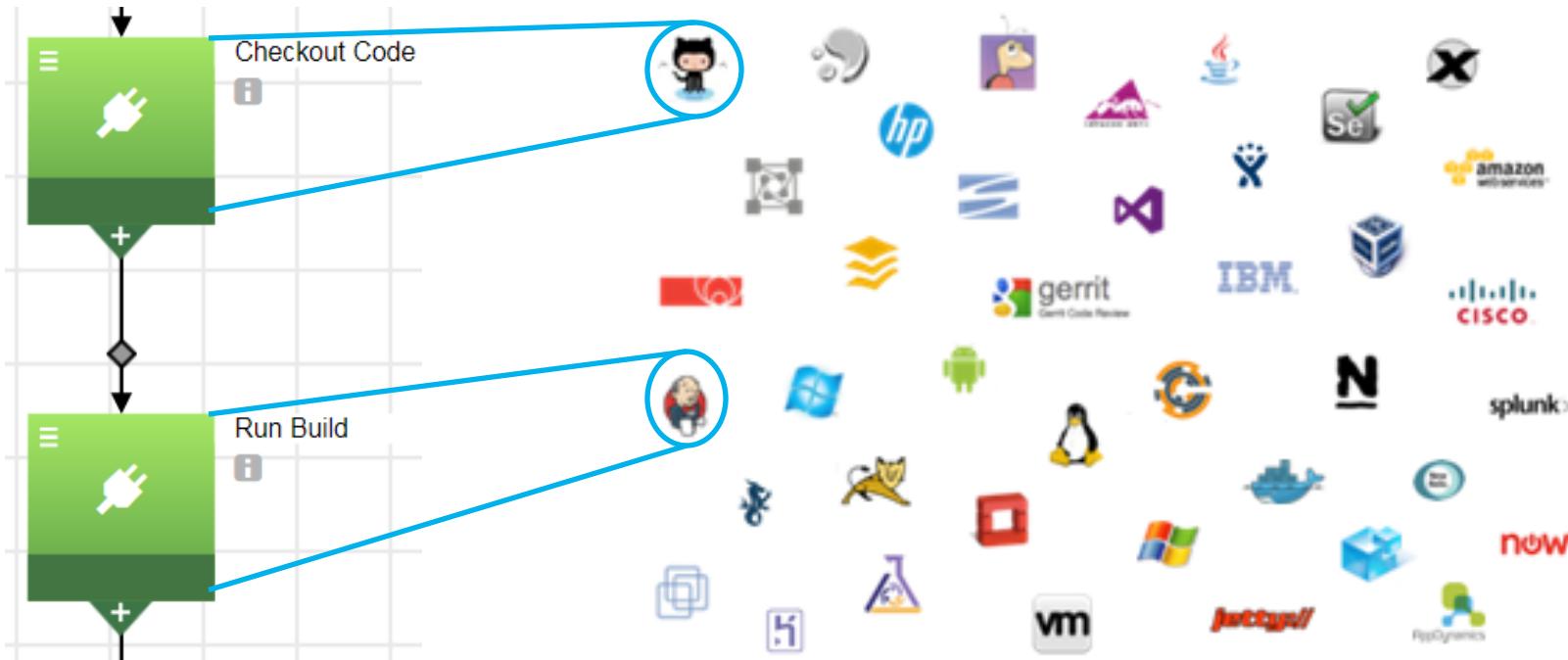


Tomcat Application Model
Create a simple Java application model to deploy to Tomcat Application Server [More...](#)

[Create](#)

Best Practices: Using Existing Tools

Integrate and orchestrate existing tools into modeled approach



Best Practices: Where to Start?

- Have a vision and set of goals first, instead of doing it just because others are doing it
- Begin with a real use case
 - Specifically, where the pain is greatest
 - Show value and expand
- Don't underestimate the culture challenge



I want ‘Boring’!

A model-driven approach to application release automation translates to boring outcomes.

Boring outcomes translate to ability to focus on what matters...your business!





A large, intense lightning bolt strikes vertically from a dark, cloudy sky. A second, slightly smaller bolt is visible to the left. The sky is filled with dark, billowing clouds, and the ground below is dark with some scattered trees and lights.

Thank You!