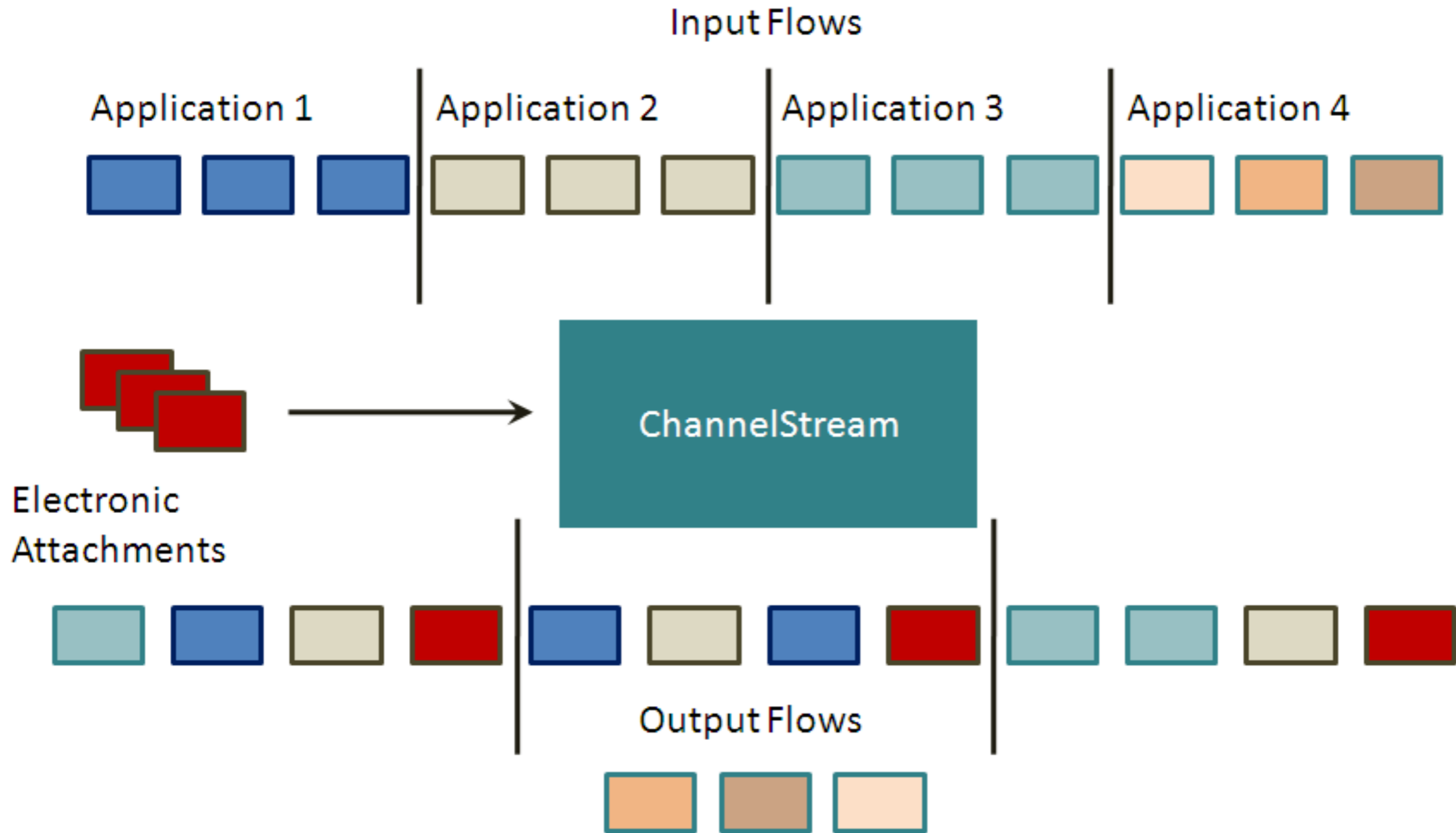# The Challenge of Continuously Delivering

Julian Fitzell, Cincom
ESUG, Edinburgh
August 24, 2011
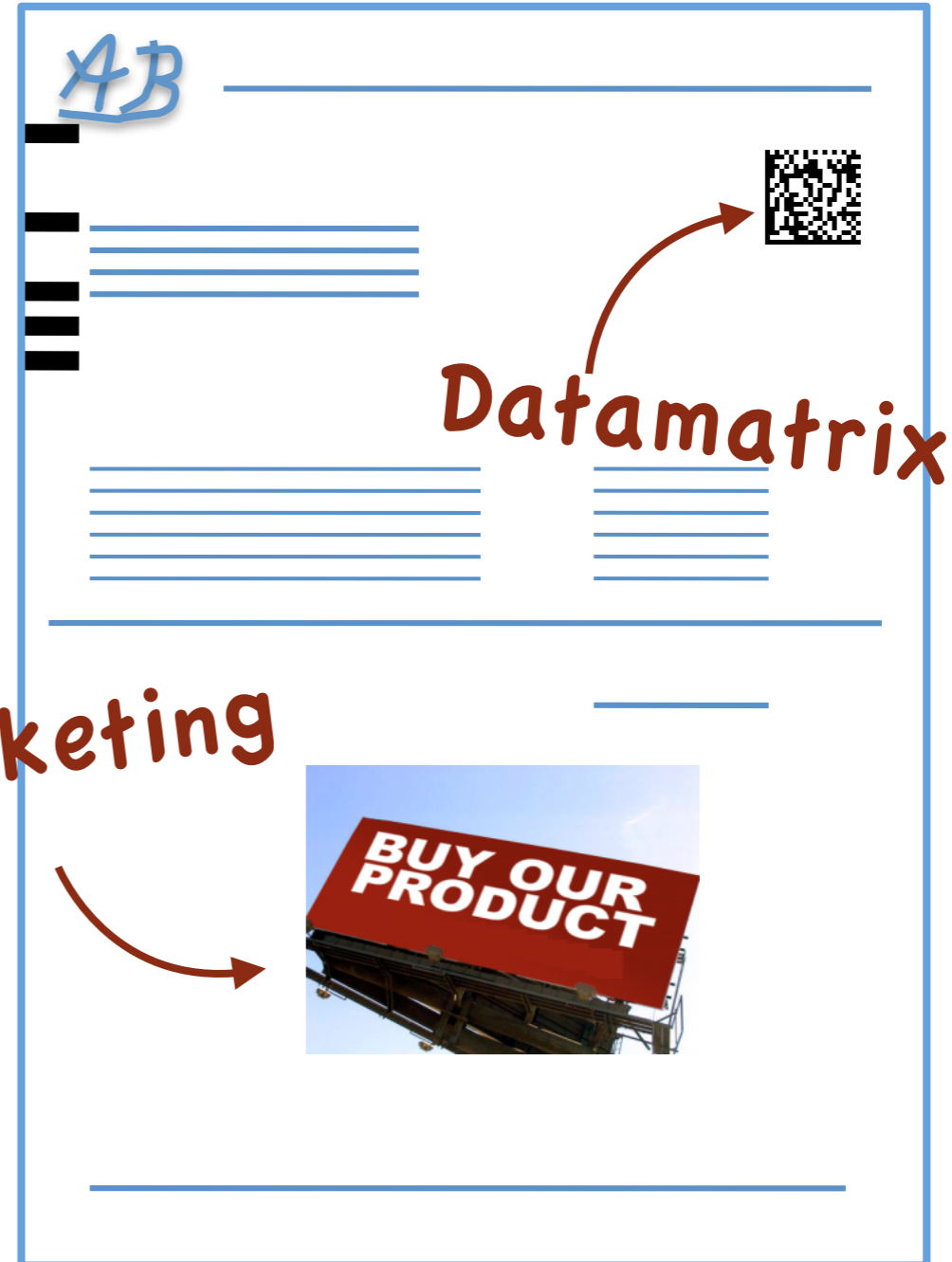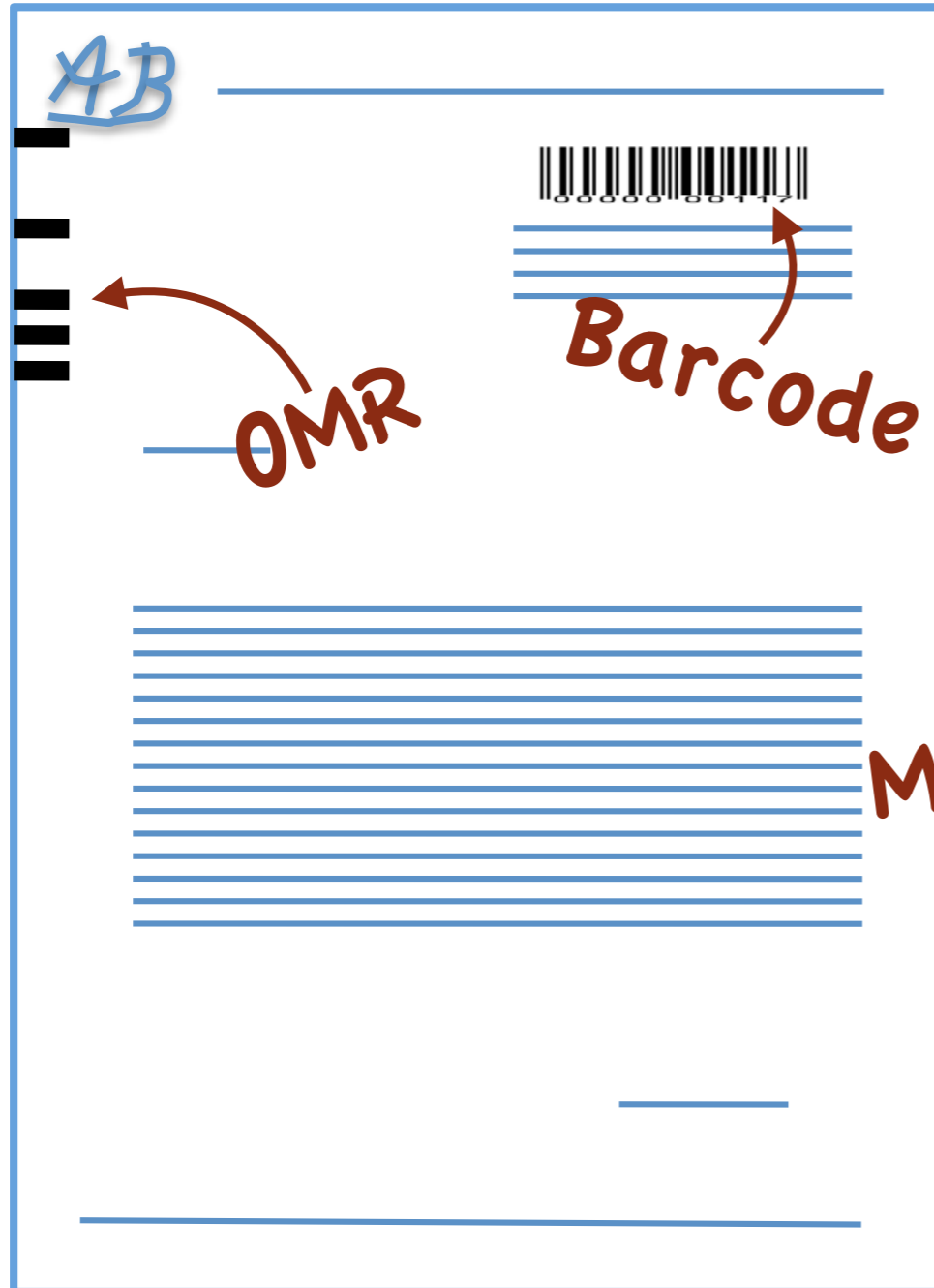
Cincom® ChannelStream™

Document Output Management
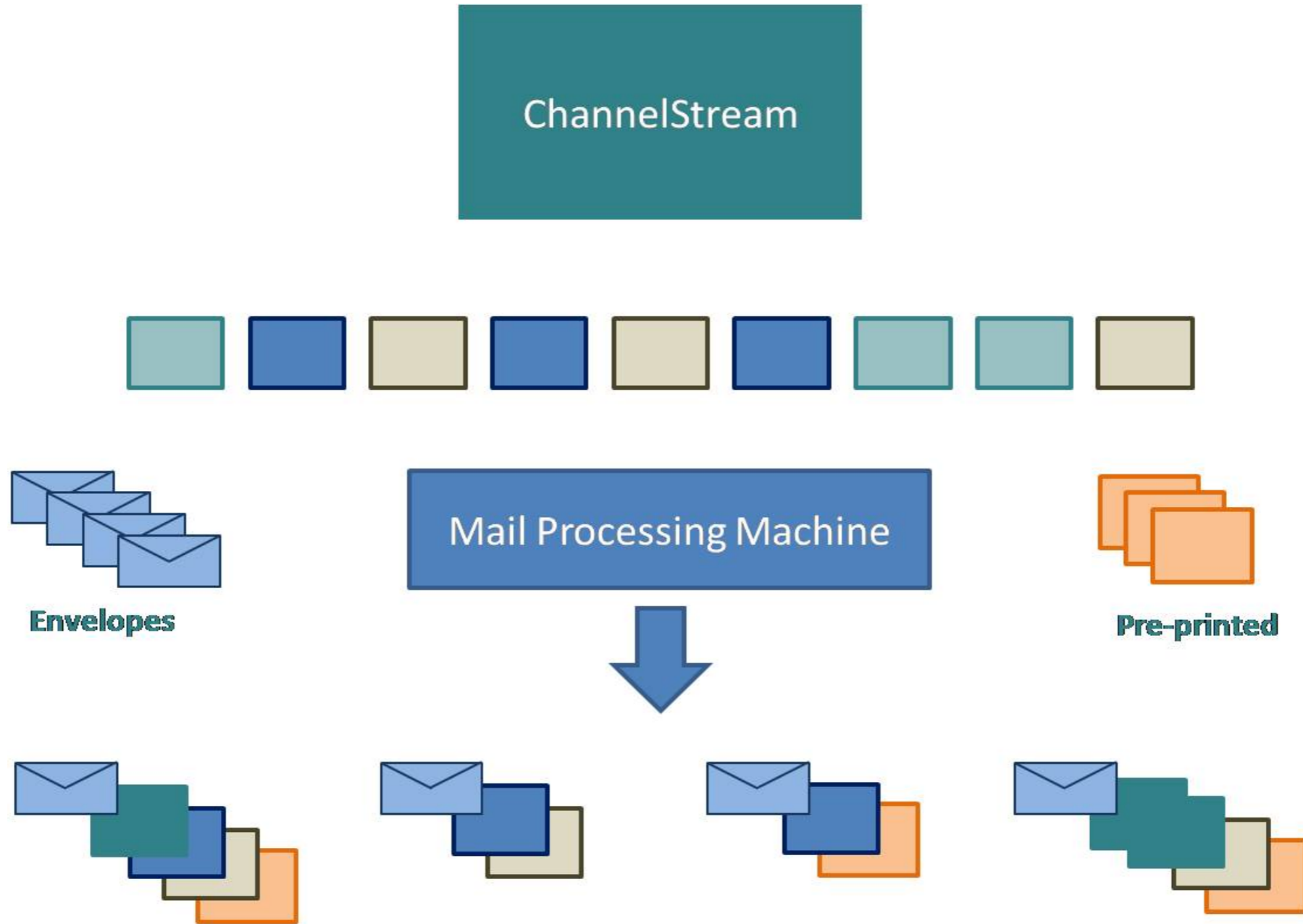
Control

Enrichment

OMR

Barcode

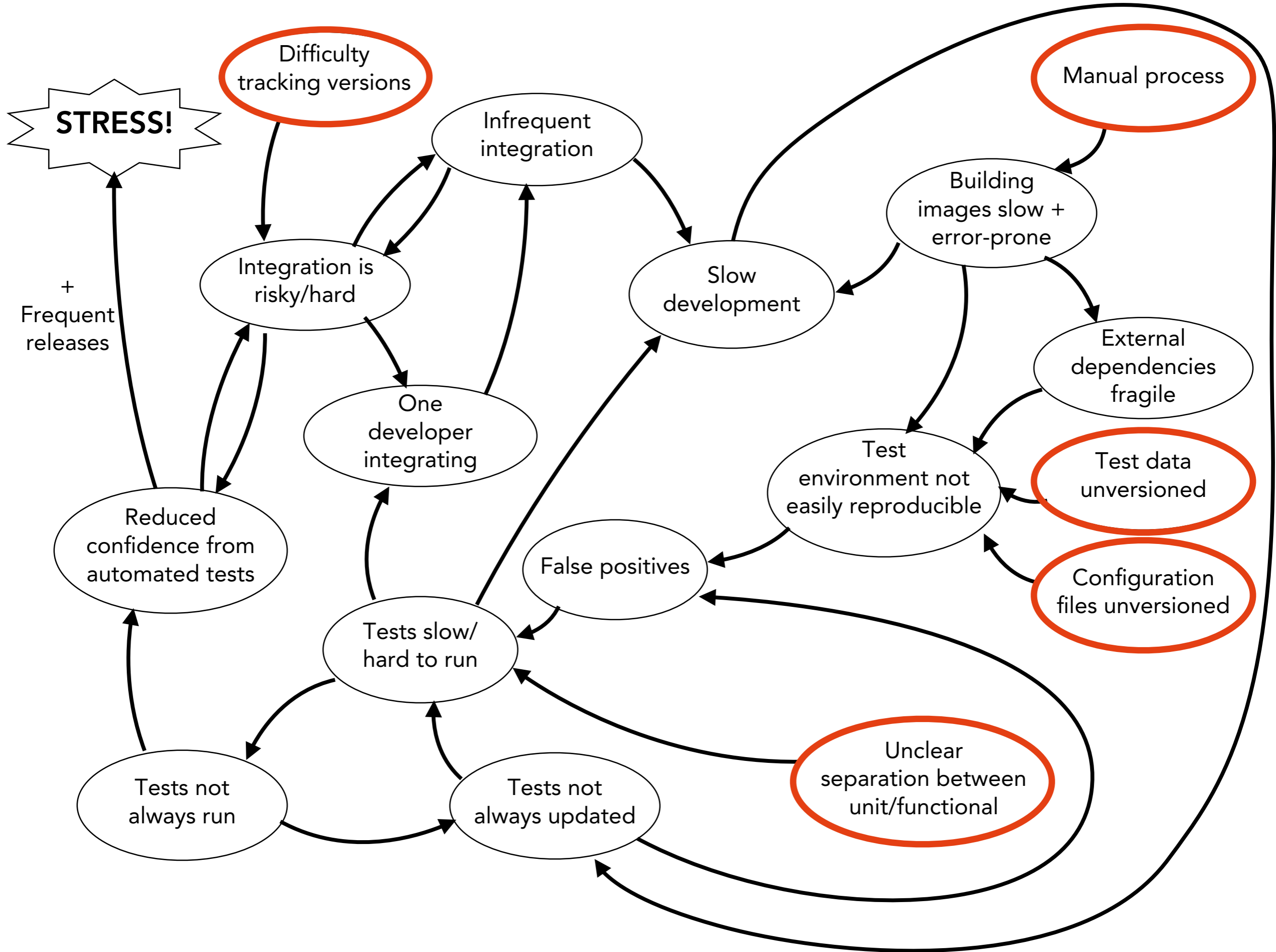Datamatrix

Marketing

BUY OUR PRODUCT

Why?

- Postal contract savings

- Saves licking envelopes

- Personalization and cross-marketing

- Avoid IT/supplier for customization

- Enhance legacy output

Challenges

STRESS!

Difficulty tracking versions

Infrequent integration

Manual process

Building images slow + error-prone

Integration is risky/hard

Slow development

External dependencies fragile

One developer integrating

Test environment not easily reproducible

Test data unversioned

Reduced confidence from automated tests

False positives

Configuration files unversioned

+ Frequent releases

Tests slow/ hard to run

Tests not always run

Tests not always updated

Unclear separation between unit/functional

So...

- Make building images easier

- Make running tests easier

- Make integration easier

con·tin·u·ous de·liv·er·y

- putting the release schedule in the hands of the business
- ensuring your software is always production ready throughout its entire lifecycle
- making any build potentially releasable to users at the touch of a button

Jez Humble, http://continuousdelivery.com/2010/08/continuous-delivery-vs-continuous-deployment/

CONTINUOUS DELIVERY

RELIABLE SOFTWARE RELEASES THROUGH BUILD, TEST, AND DEPLOYMENT AUTOMATION

JEZ HUMBLE
DAVID FARLEY

Foreword by Martin Fowler

Goals

- Reduce *cycle time* between an idea and its availability to users

- Deliver *high-quality* software *reliably* and *efficiently*

Frequent, automated releases:

- Decrease stress
- Eliminate errors and improve auditability
- Make feedback more immediate
- Protect against reliance on individual expertise
- Reduce wasted time on repetitive tasks

Make releases boring

- Automate (almost) everything

- Version everything

- Deploy everywhere the same way

- If it hurts, do it more often

- <u>Everyone</u> is responsible for delivery

Development

+

Integration

Problems

- Contention over "head"

- One person doing most of the integration

- Delayed integration => harder integration

- Poor visibility of changes

- Hard to predict/analyze failure

Issue tracker + email

Task/merge tools
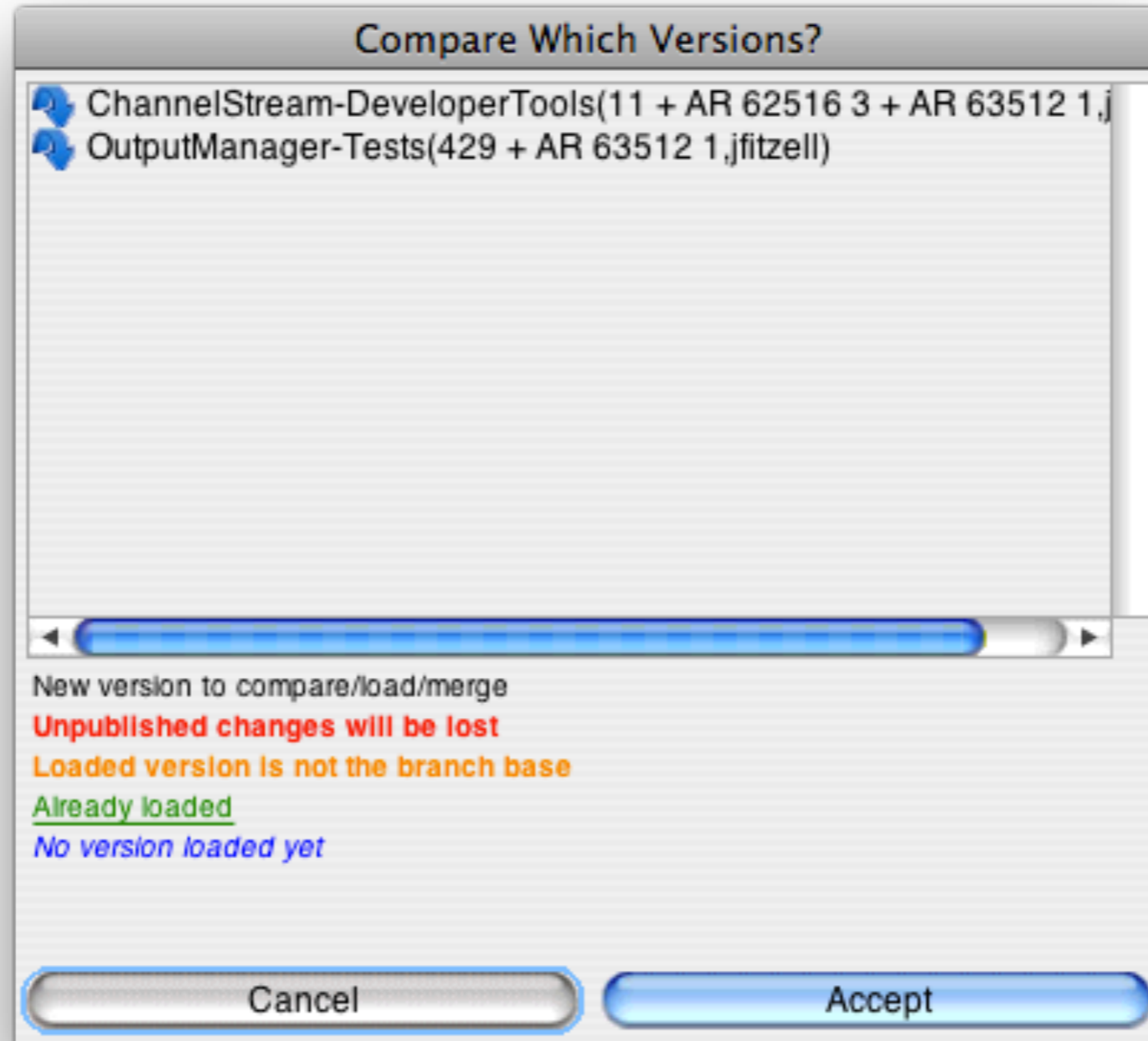
Integrate often

- Hide functionality (feature toggles)

- Incremental, releasable changes

- "Branch by abstraction" for large-scale changes

- Use components to decouple parts

Remaining Issues
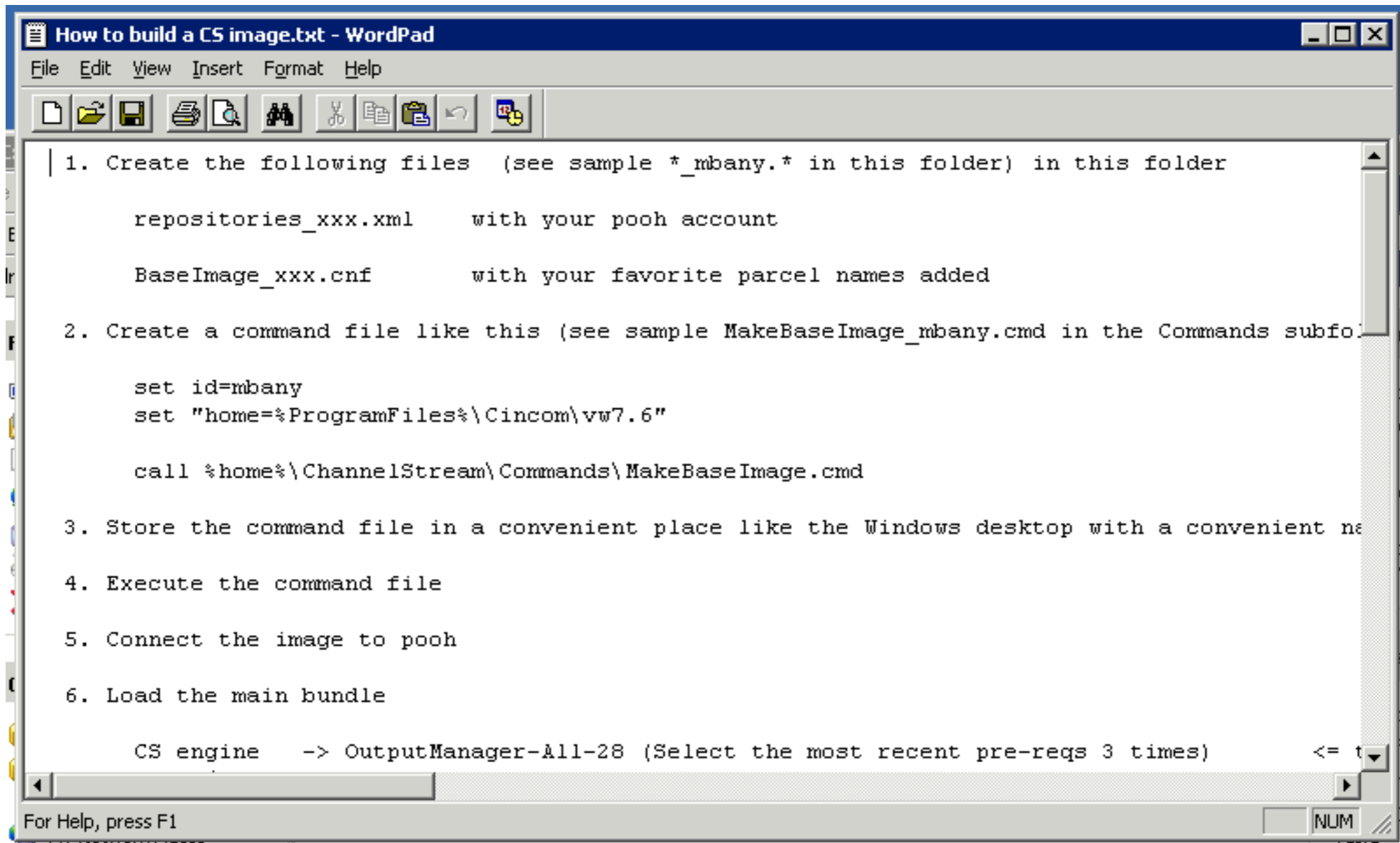
- Restructurings can still be tricky when branches are outstanding

- "Integration token" still just informal conversation

Building

Problems

- Manual builds on developer machines

- Wasted developer time

- "Here, take my image"

- Taking shortcuts

- "Works on my machine"

Helper scripts + docs

How to build a CS image.txt - WordPad

File   Edit   View   Insert   Format   Help

```
1. Create the following files  (see sample *_mbany.* in this folder) in this folder

        repositories_xxx.xml      with your pooh account

        BaseImage_xxx.cnf         with your favorite parcel names added

2. Create a command file like this (see sample MakeBaseImage_mbany.cmd in the Commands subfo

        set id=mbany
        set "home=%ProgramFiles%\Cincom\vw7.6"

        call %home%\ChannelStream\Commands\MakeBaseImage.cmd

3. Store the command file in a convenient place like the Windows desktop with a convenient na

4. Execute the command file

5. Connect the image to pooh

6. Load the main bundle

        CS engine   -> OutputManager-All-28 (Select the most recent pre-reqs 3 times)        <= t
```

For Help, press F1   NUM

Fully automated scripts

Builder

Target

-filein foo-wrapper.ws

foo.ws

foo-wrapper.ws
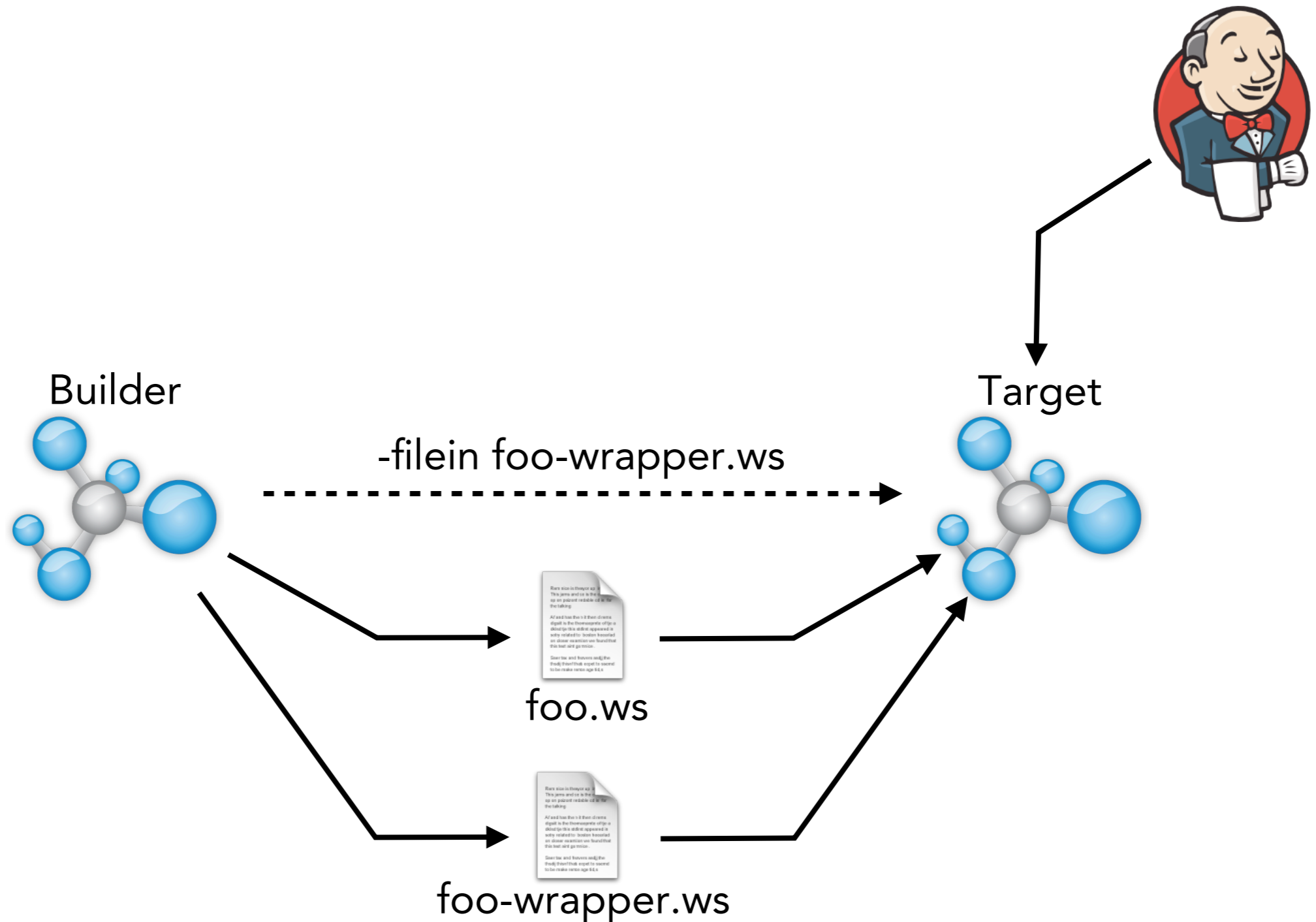
Builds run (at least)daily

S	W	Name ↓	Last Success
🟢	☀️	builder	7 min 44 sec (#12)
🟢	☁️	eloquenceProxy	7 min 1 sec (#4)
🟢	☀️	emailAgent	7 min 1 sec (#7)
🟢	⛅	engine	2 hr 11 min (#8)
⚪	☀️	engine-OLD	18 hr (#23)
🟢	☀️	reportProxy	4 min 19 sec (#2)
🟢	☀️	vw76-base	7 min 14 sec (#9)
⚪	☀️	vw76-base-OLD	21 days (#2)

All +

Archive built images

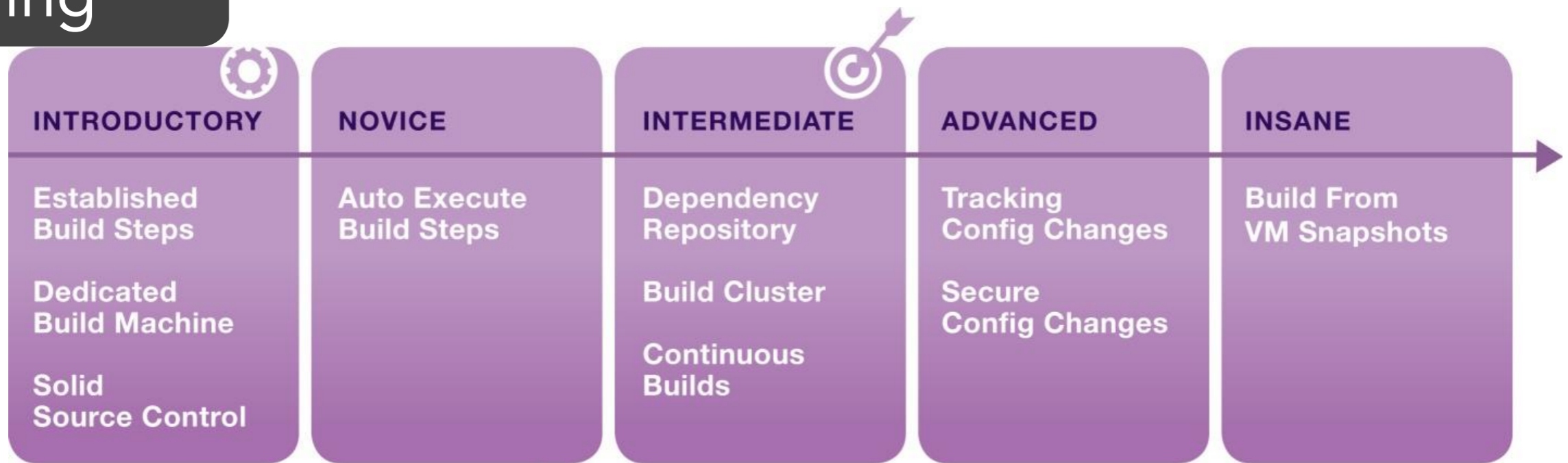| INTRODUCTORY	NOVICE	INTERMEDIATE	ADVANCED	INSANE
Established Build Steps	Auto Execute Build Steps	Dependency Repository	Tracking Config Changes	Build From VM Snapshots
Dedicated Build Machine		Build Cluster	Secure Config Changes	
Solid Source Control		Continuous Builds		

Image © 2009 by Urbancode

Run automatically on ALL code changes

Avoid versioning split between SVN + Store

Rationalize build/version numbering

urban{code}

Enterprise Continuous Integration Maturity Model

by Eric Minick & Jeffrey Fredrick

http://www.urbancode.com/html/resources/white-papers/

Testing

Problems

- Tests took up to 16 hours to run

- Test environment hard to create

- Failing tests sometimes pass when re-run

- Not everyone could run tests or be sure of results

- High cost of test maintenance

- Sometimes quicker to test manually than fix the test environment

Done

- Optimize tests (now run in ~ 2 hours)

- Separate unit/regression from functional/integration tests

- Store test data in a central location

Run tests automatically

INTRODUCTORY	NOVICE	INTERMEDIATE	ADVANCED	INSANE
Some Test Automation	Regression Tests Run with Build	Some Static Analysis Used Automated Functional Tests	Extensive Unit Tests Extensive Automated Function Tests Risk-Based Manual Tests Security Scans	100% Test Coverage

Image © 2009 by Urbancode

Run regression and smoke tests as part of build and notify of errors by email

Run functional tests after build

Version control test data

Automate creation of testing environment

Releasing

Problems

- One person is:
    - best person for build
    - best person for integration
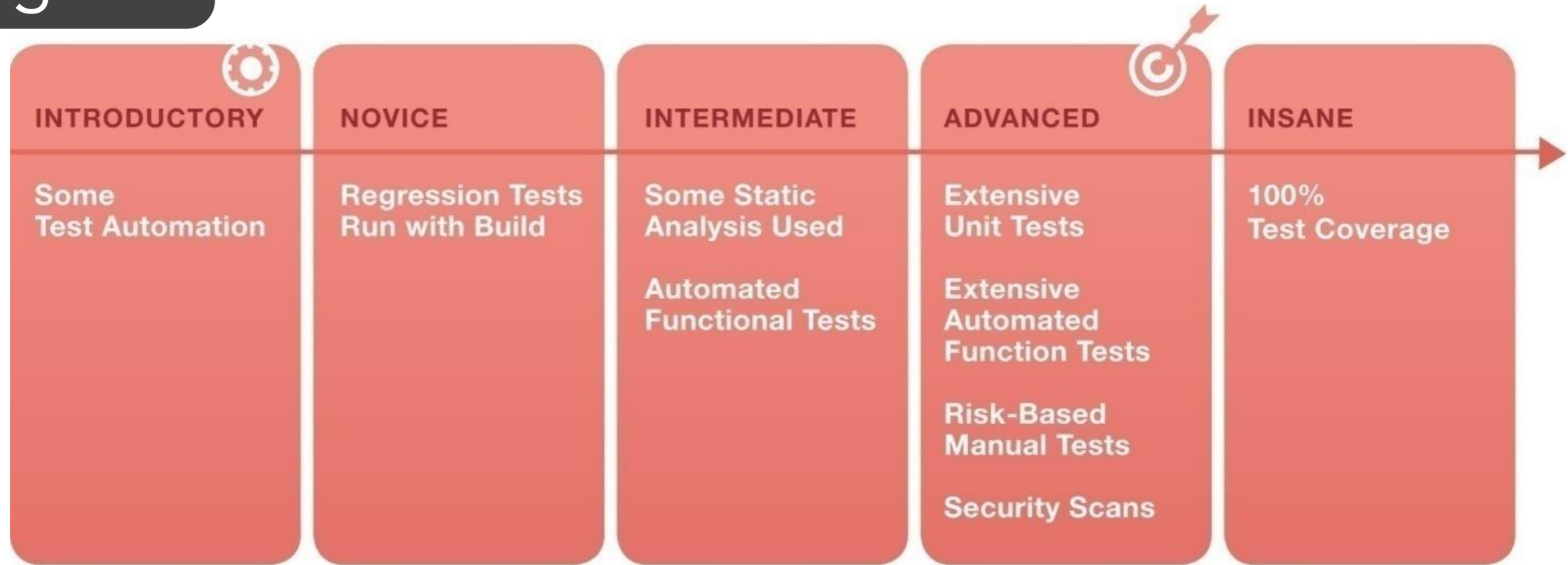
    (so, one can delay the other)

- Forced to branch when close to release

Done

- Helper scripts

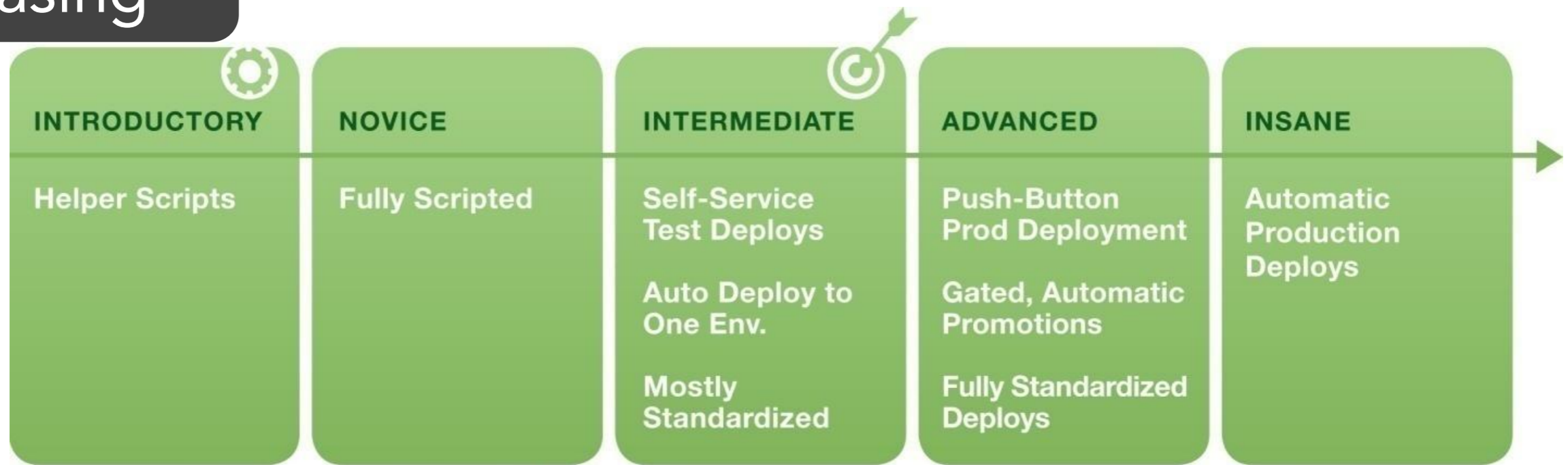| INTRODUCTORY	NOVICE	INTERMEDIATE	ADVANCED	INSANE
Helper Scripts	Fully Scripted	Self-Service Test Deploys Auto Deploy to One Env. Mostly Standardized	Push-Button Prod Deployment Gated, Automatic Promotions Fully Standardized Deploys	Automatic Production Deploys

Image © 2007 by urbancode

Fully automate runtime image build

Automate installer creation

Automate installation into test environments

Changelog generation

Julian Fitzell

jfitzell@cincom.com

Twitter: @jfitzell

Cincom