

Reeling in Outsourced Fraud Mitigation Development

Paul Ireifej / Principal Member of Technical Staff
October 15, 2019

The views expressed in this presentation are those of the presenter Paul Ireifej and do not necessarily represent the views of AT&T.

Framework for managing / migrating a software project



Agenda

- 1: Overview
2. What We Did
- 3: Challenges We Had
- 4: What We Wish We Did
- 5: Production Support

Powerful and sustained change requires **constant communication**, not only throughout the rollout but **after the major elements of the plan are in place**. The more kinds of communication employed, the **more effective** they are.

DeAnne Aguirre

advisor to executives on organizational topics



Overview



7

Outsourced
Developers



1

Internal
Developer



3

Months

What We Did

Skill Set Inventory



Take stock of resource, knowledge and skill set

Single Team Call



Day-long call and reviewed individual team member responsibilities

Recurring Knowledge Transfer



Transfer sessions, share screen with development team

Name	Role	Technology	System/Module
Joe Bloggs	GUI Developer	JavaScript/AngularJS	GUI
John Doe	GUI Services	JavaScript/Java	UI-Backend
Jane Doe	Database Engineer	<u>Hbase/Hive/Java</u>	<u>Datalake</u>
Tom Kenny	Data Engineer	Java	Ingestion
Harry Ireifej	Tester	Automated Testing	Integration

Challenges We Had

Skill Set Inventory



Inaccurate with lots of overlap

Single Team Call




Too high-level and not enough detail, sounded boiler plate

Recurring Knowledge Transfer



Only a subset of team with no agenda


Challenges We Had



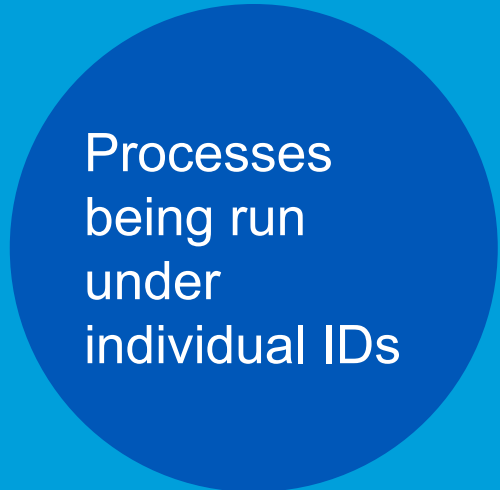
No useful
knowledge
transferred

Skill set inventory not
helpful, the right
knowledge was not
transferred

Challenges We Had



No useful
knowledge
transferred




Processes
being run
under
individual IDs

Skill set inventory not
helpful, the right
knowledge was not
transferred

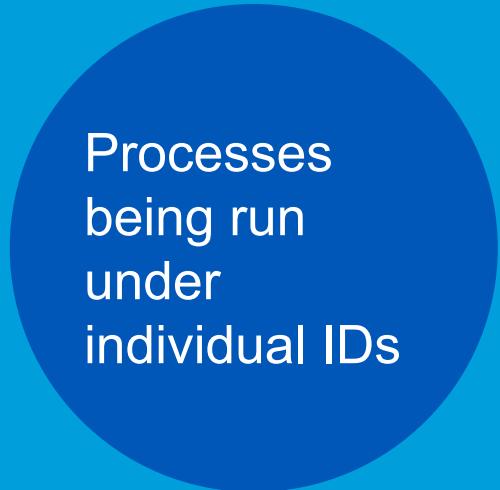
Production processes &
services could not be
manipulated

Challenges We Had

A solid purple circle containing the text "No useful knowledge transferred".


No useful
knowledge
transferred

Skill set inventory not
helpful, the right
knowledge was not
transferred

A solid dark blue circle containing the text "Processes being run under individual IDs".

Processes
being run
under
individual IDs


Production processes &
services could not be
manipulated

A solid light blue circle containing the text "In the dark with production issues".

In the dark
with
production
issues

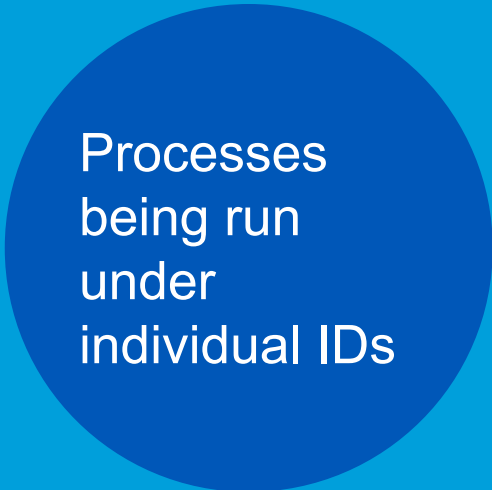
When the application
went down,
investigation was
grueling

Challenges We Had




No useful
knowledge
transferred

Skill set inventory not
helpful, the right
knowledge was not
transferred



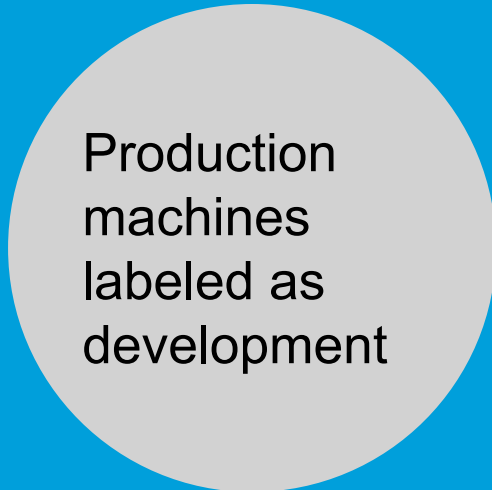
Processes
being run
under
individual IDs

Production processes &
services could not be
manipulated



In the dark
with
production
issues

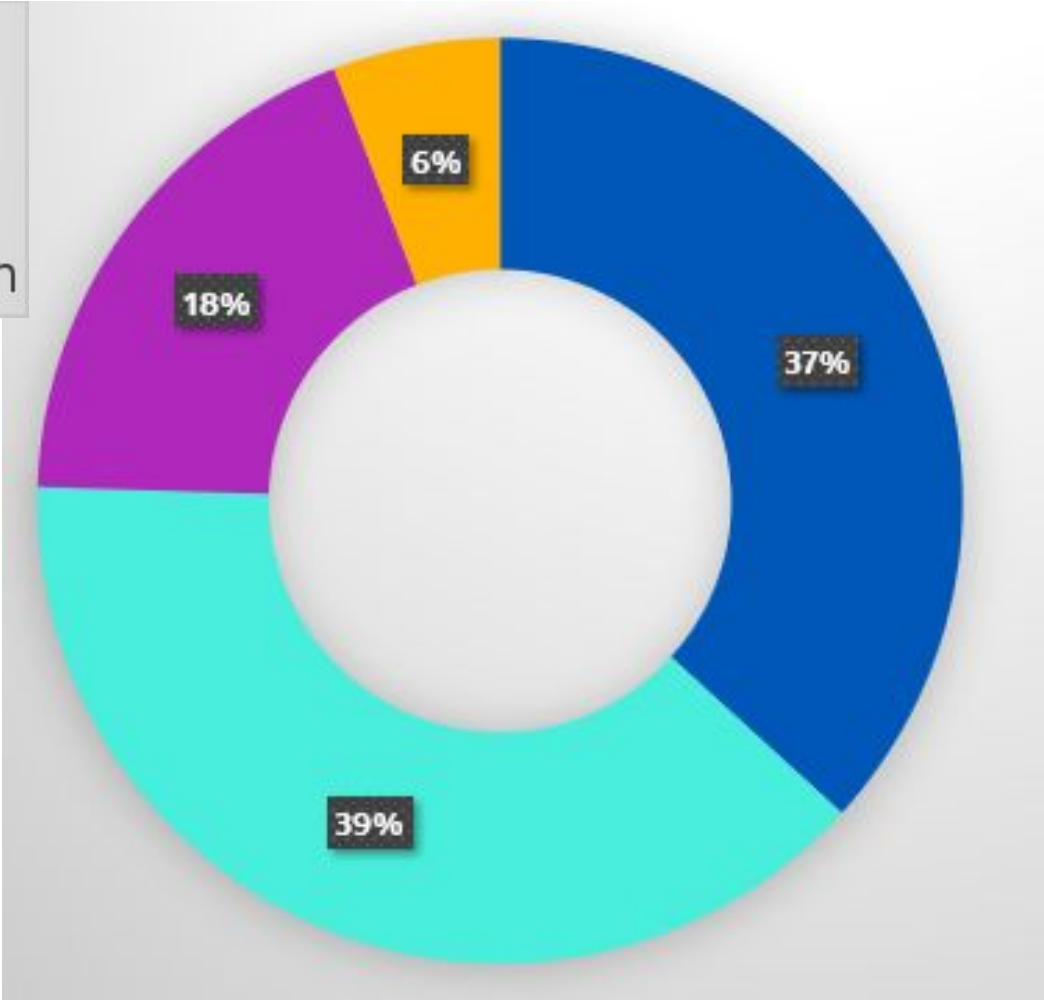
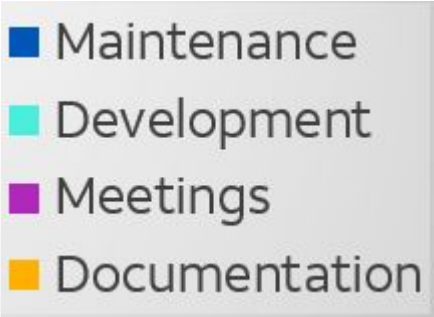
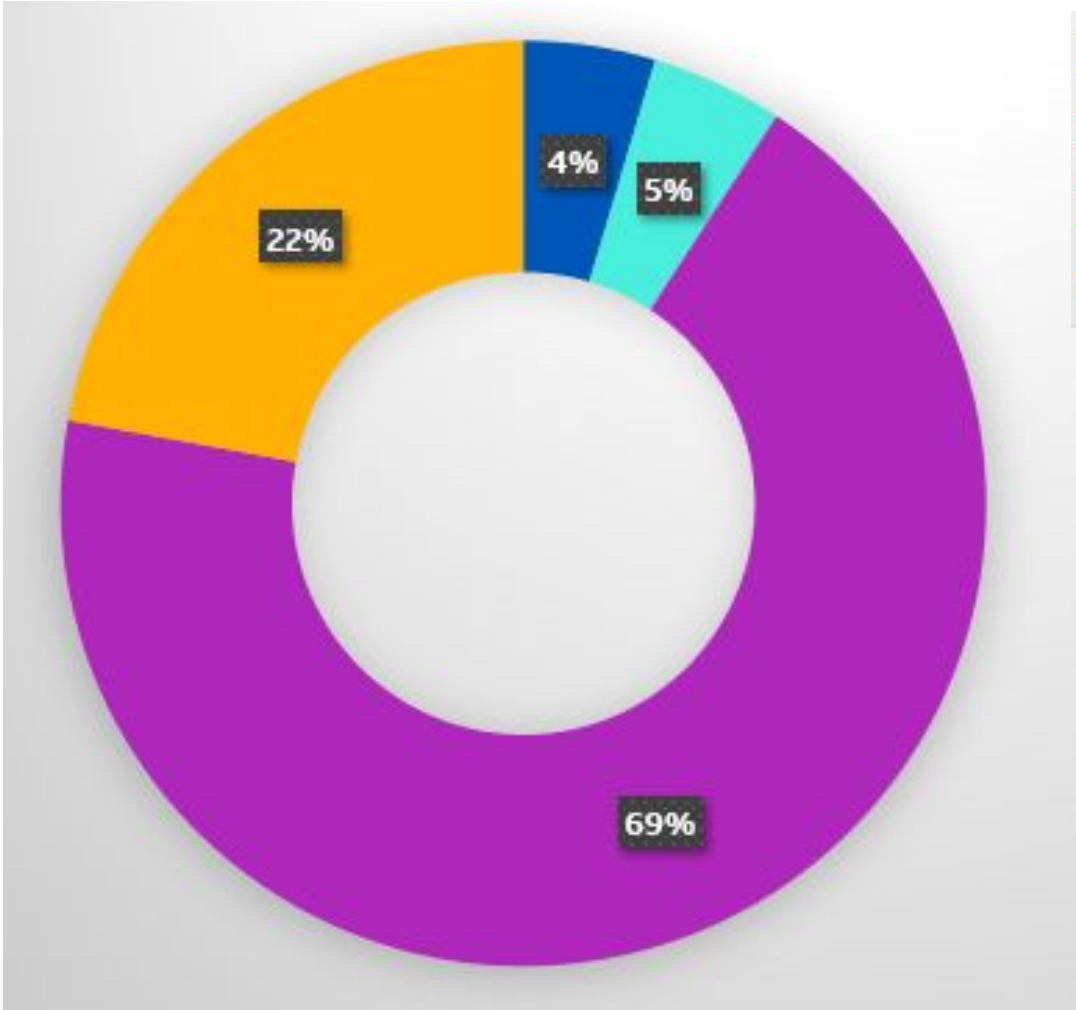
When the application
went down,
investigation was
grueling



Production
machines
labeled as
development

Production support did
not take issues on our
application servers as
high priority

What We Wish We Did: Knowledge Transfer Category Ratios



Maintenance & Design Improvements

1

Scrub all hard-coding

Search code
thoroughly for
ALL hard-coded
things epically
people's names,
user IDs,
passwords, IP
addresses,
server names

Maintenance & Design Improvements

1

Scrub all hard-coding

Search code thoroughly for ALL hard-coded things epically people's names, user IDs, passwords, IP addresses, server names

2

File-based configuration

Move all config info into a sperate file and make them env specific (dev / test / UAT / prod)

Maintenance & Design Improvements

1

Scrub all hard-coding

Search code thoroughly for ALL hard-coded things epically people's names, user IDs, passwords, IP addresses, server names

2

File-based configuration

Move all config info into a sperate file and make them env specific (dev / test / UAT / prod)

3

Use Mech IDs

Nothing should be dependent on an individual user name, use mech ID instead

Maintenance & Design Improvements

1

Scrub all hard-coding

Search code thoroughly for ALL hard-coded things epically people's names, user IDs, passwords, IP addresses, server names

2

File-based configuration

Move all config info into a sperate file and make them env specific (dev / test / UAT / prod)

3

Use Mech IDs

Nothing should be dependent on an individual user name, use mech ID instead

4

Processes

Understand all processes that are running, note user ID that it's running under and switch to mech ID

Maintenance & Design Improvements

1

Scrub all hard-coding

Search code thoroughly for ALL hard-coded things epically people's names, user IDs, passwords, IP addresses, server names

2

File-based configuration

Move all config info into a sperate file and make them env specific (dev / test / UAT / prod)

3

Use Mech IDs

Nothing should be dependent on an individual user name, use mech ID instead

4

Processes

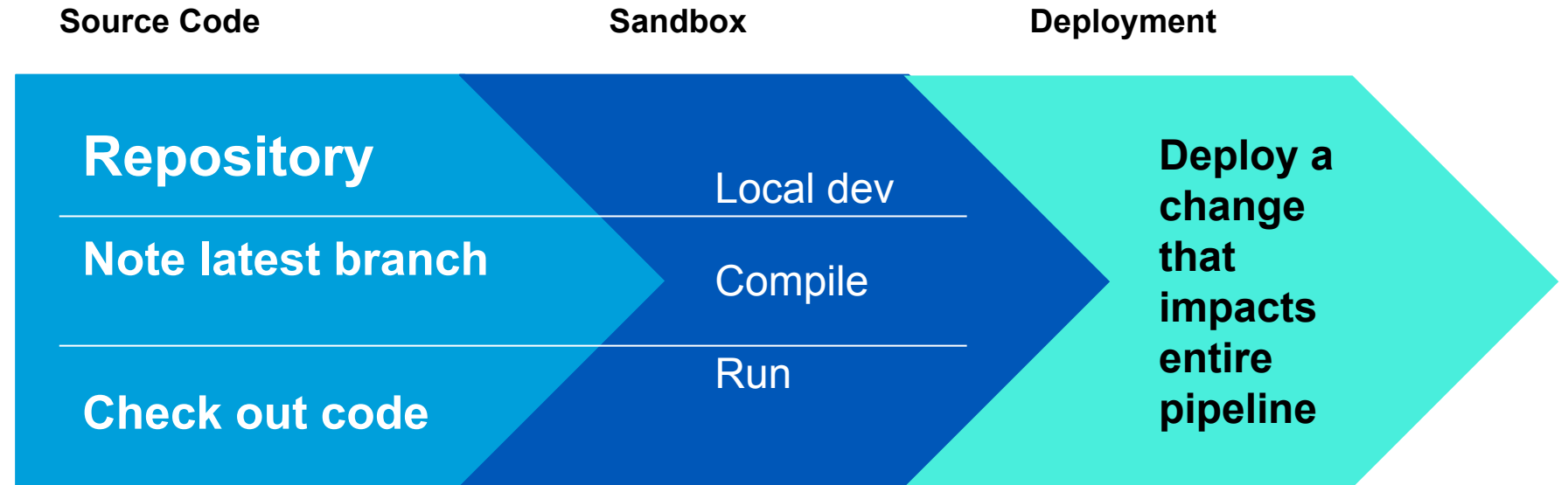
Understand all processes that are running, note user ID that it's running under and switch to mech ID

5

Architecture Diagram

Draw a picture, from your perspective, on how you understand the system and its piece-parts. Tweak as you learn more information

Development & Source Code Management



Meetings



Documentation

Update existing document

-- or --

Create documentation from scratch

Start early, document as you learn

Machine names, server information, mech IDs, permissions needed, IP addresses, environment-specific details

MY PROJECT

Paul Ireifej, 10/15/2019, 3:35 PM

**SYSTEM DESIGN,
ARCHITECTURE DIAGRAM,
SERVER INFORMATION**



Production Support

**Digest information,
Formalize &
Transition continuously**



THANK YOU





AT&T