DEEP DIVE

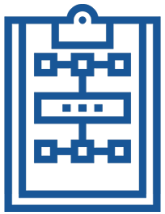
→ SERVICE TRANSITION

SERVICE TRANSITION – PRIMARY BREAKDOWN

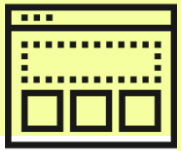
Service Validation and Testing

Change Evaluation

Change Management



Service Asset And Configuration Management



Application Development

Release And Deployment Management

Transition Planning And Support (Project Mgmt.)

Knowledge Management

RELEASE AND DEPLOYMENT MANAGEMENT

- Release and Deployment Management is used when larger number of complex changes to the service environment are needed
- Typical example is a migration of customer infrastructure between locations

  (Moving each Configuration Item as a separate change would be impractical)

Release Management tries to **minimize service disruption caused by large-scale changes**.

Construction of a shopping center in the middle of a city

- Roads must be closed
- Ground excavated
- Water supply pipes moved

- Plan alternate routes
- Inform public in advance
- Ensure water supply

RELEASE AND DEPLOYMENT MANAGEMENT

Terminology

Releases are made of Release Units – one, or several

Release Unit contains multiple Configuration Items that need to be moved

Ex.:

Migration Release =

Release Unit A (Database Systems) +

Release Unit B (Storage Systems) +

Release Unit C (Application Systems)

Construction Release =

Release Unit C (Construction) +

Release Unit TR (Traffic Rerouting) +

Release Unit WS (Water Supply) +

Release Unit NM (Noise Mitigation) +

Release Unit COM (Communication to Public)

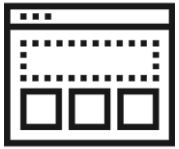
RELEASE AND DEPLOYMENT MANAGEMENT

Terminology

Release Unit may contain a Development Work Order – this triggers Application Development process.

Release Unit may contain an Installation Work Order.

Release Management focuses on changes which require:



Application Development

Build

- Development Environment

Test

- Testing Environment

Deploy

- Live Environment

RELEASE AND DEPLOYMENT MANAGEMENT

Release Planning	• Necessary changes are divided into Release Units • Scope of Release Units is defined • Release Units are scheduled for building, testing and deployment
Release Build	• Development Work Orders and Installation Work Orders are submitted for "construction" • Hardware and Software is purchased, Applications developed...
Testing Phase	• All built elements are subjected to pre-defined testing. • Includes various types of testing, such as functional, performance, security, and user testing, to ensure a smooth service deployment.

RELEASE AND DEPLOYMENT MANAGEMENT

Release Deployment

- Configuration Items are moved from Testing Environment to Live Environment.
- Users are trained in using and operating the components.
- Documentation is provided as needed.

Early Life Support

- Temporary support phase after a new service or big change goes live.
- During this time, the service is **closely monitored**, and any issues are **quickly fixed** to make sure everything runs smoothly.
- Helps the operations team get familiar with the new service before it moves to regular support.

Release Closure

- Final step in the release process, wrap-up after a new service or change has been successfully deployed.
- Includes **checking if the release met its goals**, documenting lessons learned, making sure any remaining issues are handed over to regular support. Once everything is confirmed to be stable, the release is officially closed.
- Pre-condition: All Configuration Items must be up-to-date in the CMDB.

APPLICATION DEVELOPMENT

Full title: Application Development and Customization

Frequently triggered by **Development Work Order** from **Release And Deployment Management**

Identify Market Need	• Which need shall this application satisfy? • Ex.: Send email from cell-phones, Kill zombies in a game
Planning Phase	• Developers break the application into **manageable chunks** • What resources will be needed for each chunk?
Design	• Detailed design of how the chunk will function • Followed by repeated {implementing, testing, documenting}

APPLICATION DEVELOPMENT – ABOUT DEVELOPMENT DEBT

This lesson is applicable in all other areas

Development Debt:

> All the shortcuts taken during development and never changed to their full paths; “cutting corners”

Typical case: Not creating documentation for features (for users) and code (for programmers)

Developer saves times by not spending it on writing detailed documentation.

Wisdom: Development Debt accumulates interest over time, and it will have to be repaid.

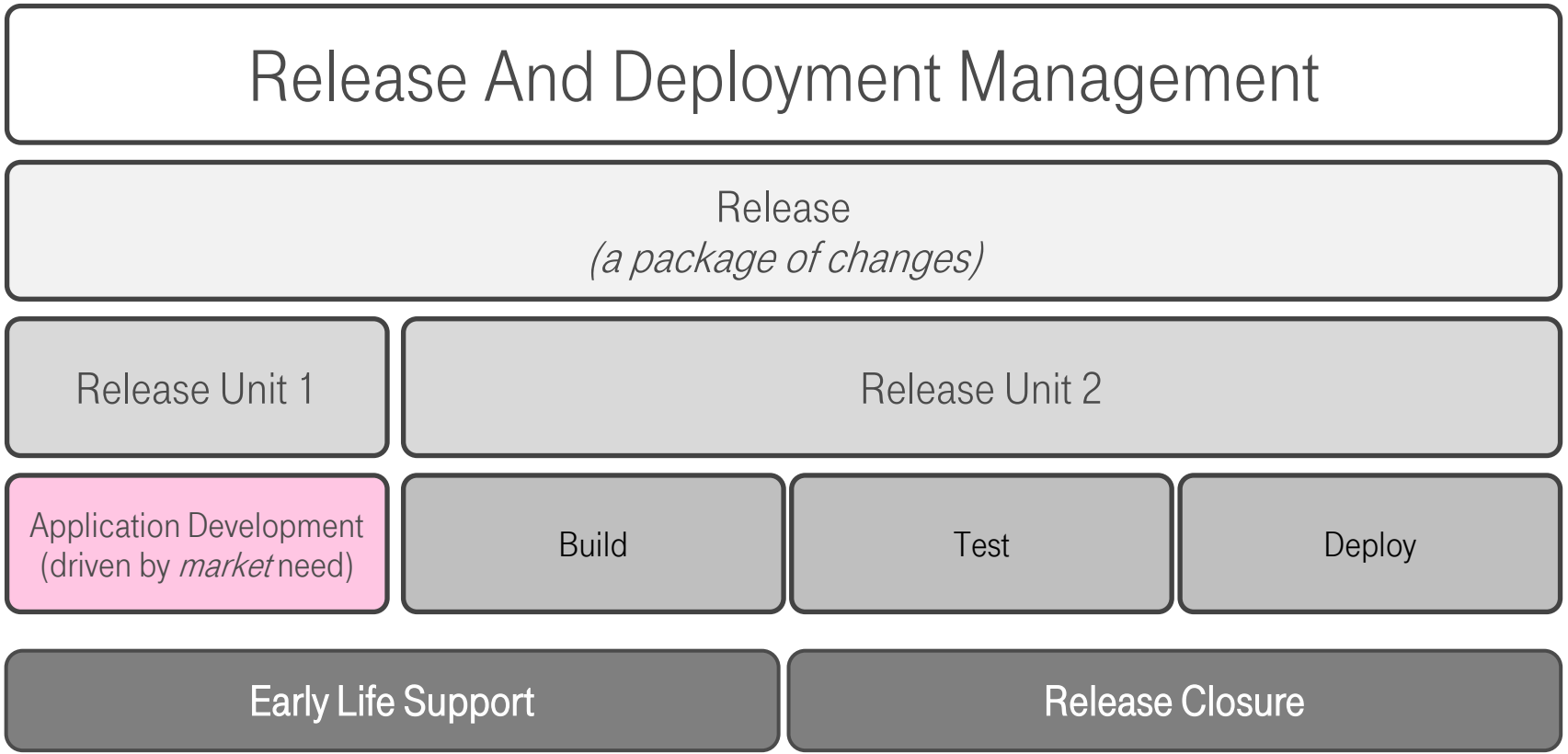
Ex.:

> 2 Weeks saved by not writing documentation now = 10 weeks of figuring out how it works later.
>
> Operator saves 2 minutes by not writing description of incident resolution,
> and then has to reinvent that solution a month later.

APPLICATION DEVELOPMENT

Identify Market Need	• Which need shall this application satisfy?
Planning Phase	• Developers break the application into **manageable chunks**
Design	• Detailed design of how the chunk will function • Followed by repeated {implementing, testing, documenting}
Deployment	• Application is launched – made accessible by customers/users.
Application Maintenance	• Post-launch maintenance is performed: Bug fixes, improvements to User Experience, enhancements of functionality

RELEASE MANAGEMENT AND APPLICATION DEVELOPMENT

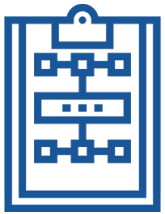


SERVICE TRANSITION – PRIMARY BREAKDOWN

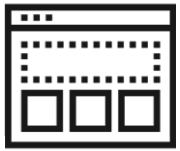
Service Validation and Testing

Change Evaluation

Change Management



Service Asset And Configuration Management



Application Development

Release And Deployment Management

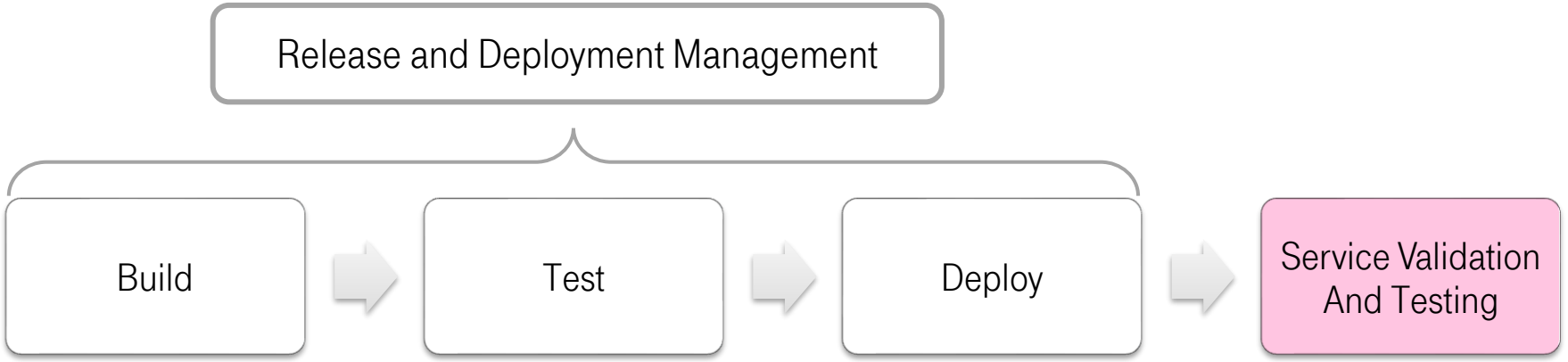
Transition Planning And Support (Project Mgmt.)

Knowledge Management

SERVICE VALIDATION AND TESTING

Objective: Check if a service is operating as expected.

It can take place during and after Release and Deployment Management, but it is often carried out before the service is deployed to production to ensure that it meets all requirements.



SV&T is separated from RDM to ensure objective quality control and minimize the risks associated with deploying a new or changed service.

SERVICE VALIDATION AND TESTING

A software testers enters a bar.

He orders a beer.

He orders -1 beer.

He orders 0 beers.

He orders 99999999999999999 beers.

He orders a lizard.

He tries to leave without paying.

SERVICE VALIDATION AND TESTING

- Service Validation and Testing must verify that
  - Valid inputs by authorized users result in expected outcomes.
  - Valid inputs by unauthorized users are handled properly – typically informing them they need to be authorized.
  - Invalid inputs do not cause disruption.

- Flaws in services may have dire consequences.

How is this testing done?

SERVICE VALIDATION AND TESTING

Naive approach:

Let users test the service the way they want and report anything unusual

Problems:

1. Does not guarantee that all valid interactions will be tested
   Rare or unusual usage
   Unattractive element
2. Test users behave as honest customers and don't try to break the service or breach its security
   How many people can perform X-site Javascript injection?
3. This approach is unmanageable in terms of cost and duration
   How long will it take?
   Have we tested enough?

SERVICE VALIDATION AND TESTING

ITIL Approach:

Using Test Models – Detailed descriptions of how to test the release.

Test Model

Defined by Service Validation and Testing, but
Created during Release Planning (!)

→ Note that **Release Management** and **Service Validation** are running in parallel at this point!

Release And Deployment Management

Service Validation and Testing

SERVICE VALIDATION AND TESTING – SUB-PROCESSES

Test Model Definition

- Defines the Test Model
- The Test Model is **created** in Release planning phase

Release Component Acquisition

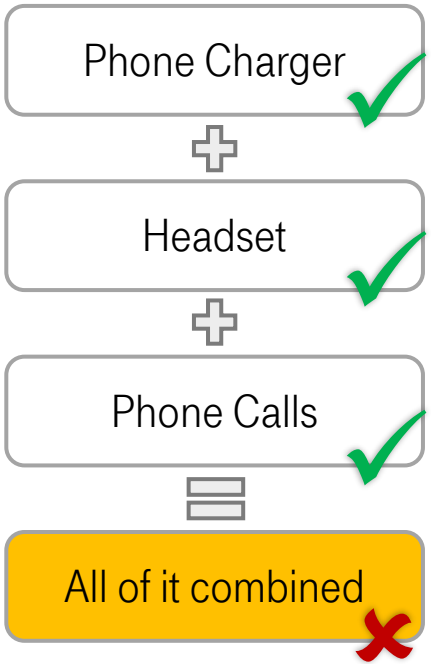
- Makes the components which must be tested available
- Components are checked for manufacturing flaws which would make the testing unreliable

Release Test

- Actual testing according to the Test Model
- Tests both valid and invalid inputs

Service Acceptance Testing

- Tests the ENTIRE SERVICE, i.e., a chain of components that together deliver the service
- Ends when customer signs Acceptance Protocol

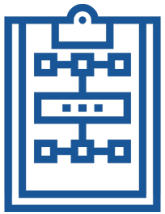


SERVICE TRANSITION – PRIMARY BREAKDOWN

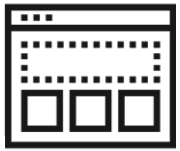
Service Validation and Testing

Change Evaluation

Change Management



Service Asset And Configuration Management



Application Development

Release And Deployment Management

Transition Planning And Support (Project Mgmt.)

Knowledge Management

PROJECT MANAGEMENT – TRANSITION PLANNING & SUPPORT

- Methodologies such as PM-BOK or PRINCE2 are well established for Project Management
  For this reason, ITIL does not collect and provide actual PM best practices.

- Transition Planning And Support has the objective of planning and coordinating resources for Releases in an effective and efficient way.
- Observes: Time, Quality and Costs of Major Releases.

TRANSITION PLANNING & SUPPORT

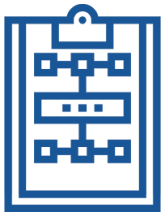
Project Initiation	• Identifies Stakeholders, nominates project team • Resources are committed and Risk analysis performed
Project Planning And Coordination (ongoing)	• Plans activities and coordinates them • Organization's PM guidelines are followed
Project Control	• Also known as Steering Committee or Quality Gate • Checks project's progress, resources • Initiates corrective measures if needed
Reporting and Communication (parallel)	• Responsible for regular reporting on the project (towards organization's management) and communication to affected parts of the organization

SERVICE TRANSITION – PRIMARY BREAKDOWN

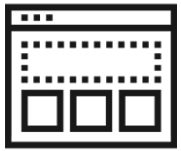
Service Validation and Testing

Change Evaluation

Change Management



Service Asset And Configuration Management



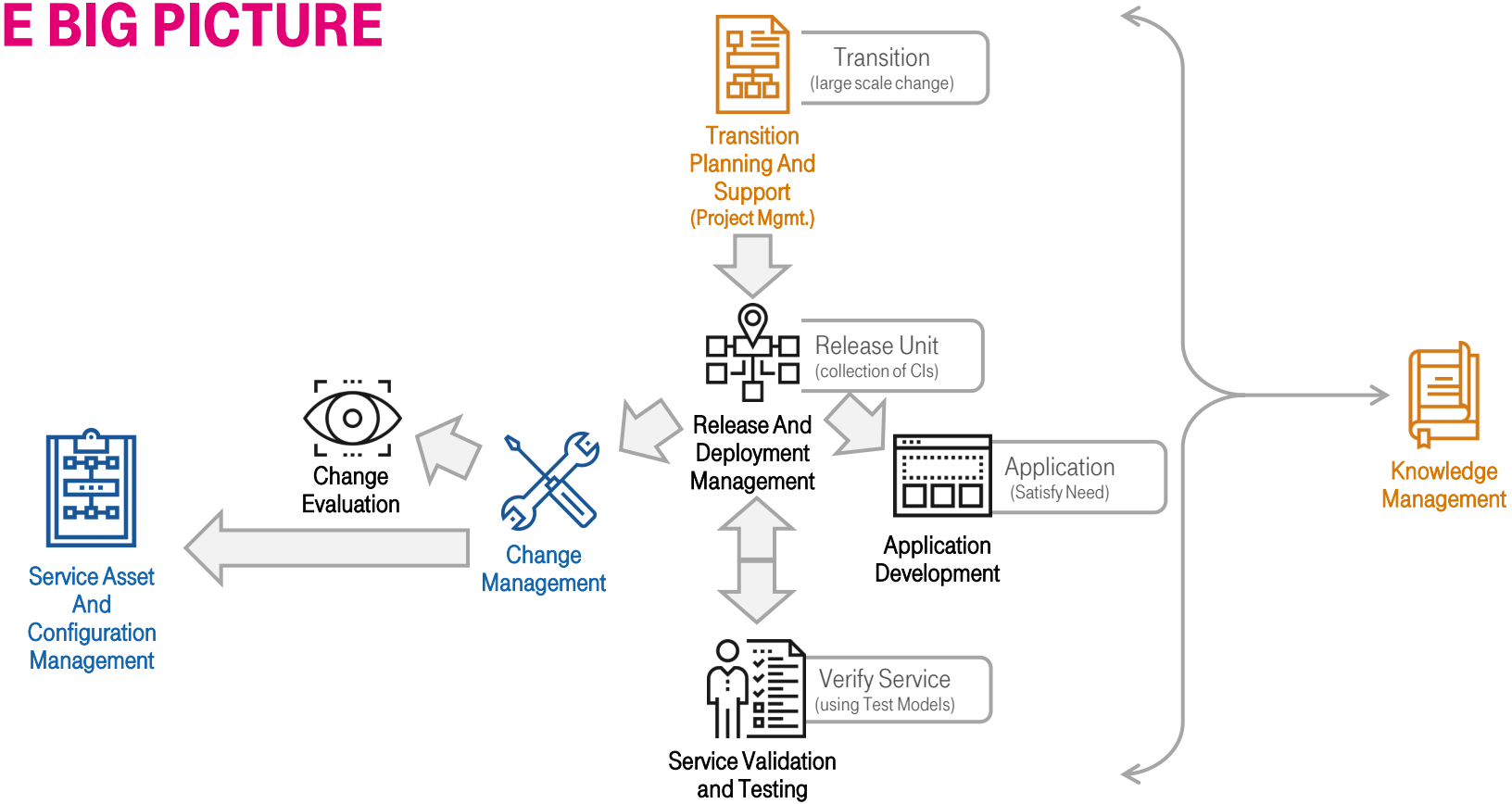
Application Development

Release And Deployment Management

Transition Planning And Support (Project Mgmt.)

Knowledge Management

THE BIG PICTURE



SERVICE TRANSITION - SUMMARY

Core Take-Aways for Service Transition:

1. Whenever any part of our service has to change, then that part enters Service Transition, and at some point, a Change will be executed.
2. As changes increase in complexity, additional processes get involved to provide support.

   Release Management – minimizes disruption during larger changes

   Transition Planning and Support – for large-scale changes
3. Making the right decisions requires perfect information.

   Configuration Management System holds information about elements.

   Service Knowledge Management Systems govern all other knowledge

DEEP DIVE

→ SERVICE TRANSITION