Governance & Elements.Cloud

Svet Voloshin

Governance = Discipline = Freedom

Prevent project chaos. Control outcomes.

Objectives

- Understand various governance topics

Why cover these topics?

- To maximize chances of project success

Scope

Governance topics to include

- Project Methodology
- Risk Management
- Test Management
- Center of Excellence
- Continuous Integration
- Release Management
- Regulations

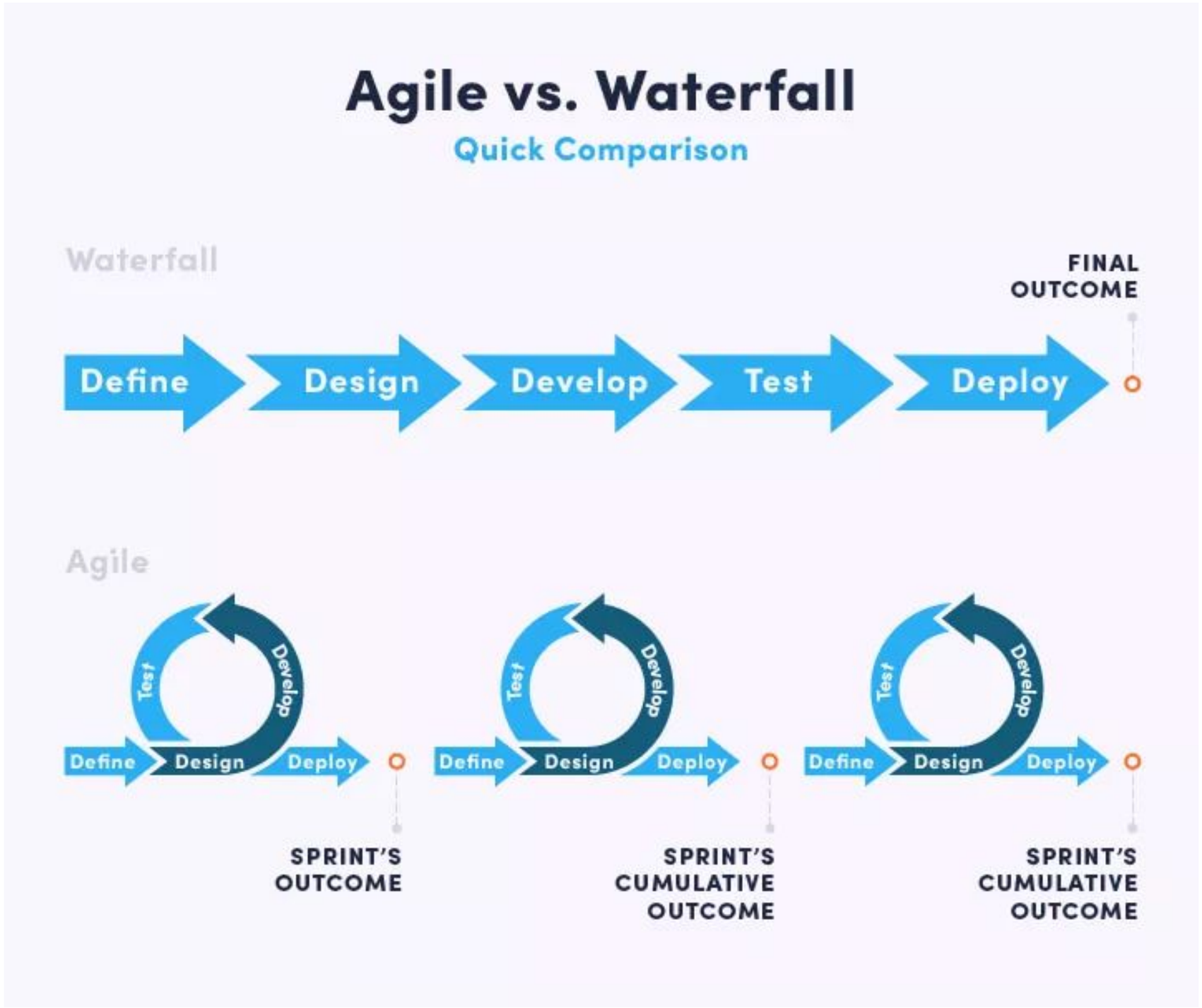
Key Points

- *“If you fail to prepare, you are preparing to fail.”*

  -Anonymous

- I work as part of a team – you will learn how to work with your Project Manager, rather than taking orders
- I work as an individual contributor – you will learn how to guide the client through the project and keep your sanity.

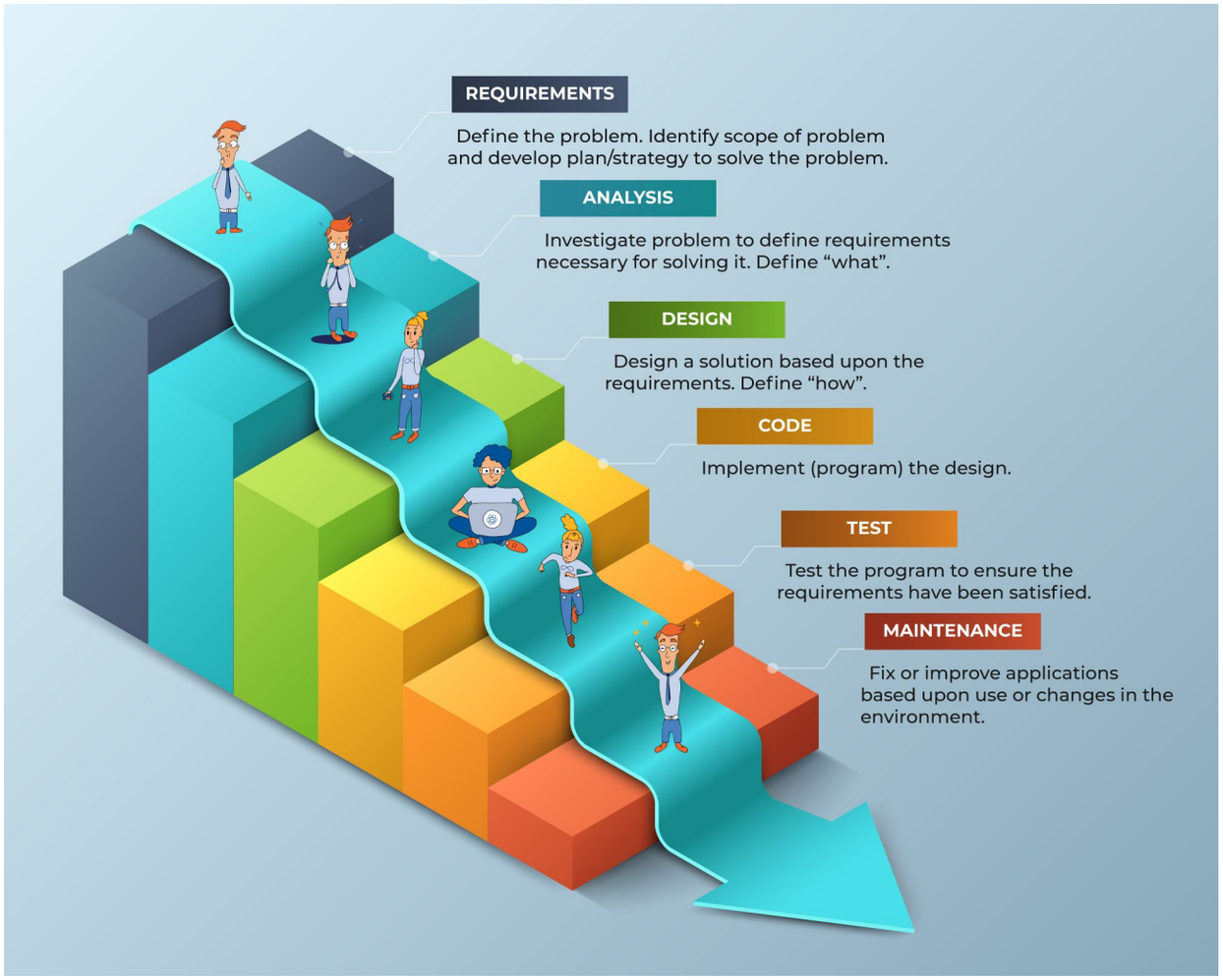
Agile vs. Waterfall



Waterfall Methodology

Waterfall

The waterfall methodology is a linear project management approach, where stakeholder and customer requirements are gathered at the beginning of the project, and then a sequential project plan is created to accommodate those requirements. The waterfall model is so named because each phase of the project cascades into the next, following steadily down like a waterfall.



Advantages:

Regular reviews (milestones) to assess project success and necessary adjustments.Easy to use and implement. Easy to manage, no special training or artifacts needed.

Disadvantages

Problems or mistakes which are found later in the process can only be fixed with major effort. High amount of risk and uncertainty since no working software is produced until the end. Can`t handle complex requirements well. Poor model for ongoing projects or improvements.

Best suited for:

Small projects with known requirements. Requirements are well understood. Fixed scope and timeline. Well suited for initial project. **Known technology and requirements.**

Regulatory requirements for certifications and so forth.

Risk of Waterfall:

Changing requirements -> Must rigorously be monitored with change management process.

Project progress is hard to measure -> Milestones

Agile Methodology

Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches.

Advantages:

Rapid, continuous delivery of useful delivery. People over process. Working software over documentation. Fast feedback cycles. Regular adoption to changing customer demands.

Disadvantages:

Hard to estimate effort and timeframe. Bad for fixed feature / scope combination. Lackluster documentation. Self-organization of team and people need some experience and time.

Best suited for:

Changes to existing system. **Fast iteration** and small adoptions.

Risk of agile:

Complex integrations and processes cannot be built in short sprints. ->

Small requirements or changes do not fit into overall strategy -> CoE

TRAILHEAD

Waterfall-Agile

W-Agile combines (freely) elements of Agile and Waterfall as needed.

Waterfall for the methods which need fixed planning and upfront definition and/or are known.

Like Backend integrations, Complex milestones and … Within these fixed elements use a Agile approach to experiment with more flexible elements.

Advantages

Can combine best from Agile and Waterfall.

Disadvantages

Needs know-how for both methodologies. Cannot be repeated.

Best suited for:

Transition from Waterfall to Agile.

Regulation or contractual requirements for specific pieces of work but flexibility with other pieces.

Risk of W-Agile:

Neither methodology is implemented properly and there is chaos and confusion leading to project failure.

Scrum or Kanban - how about Scrumban?

	Scrum	Kanban
Who prioritizes it?	Product owner prioritizes on product backlog	Product owner prioritizes on product backlog
Where does it go?	The product backlog is reordered for the next sprint.	The product backlog is continuously reordered for the next available person with capacity.
When does the work start?	During sprint planning, the team commits to the work in the next sprint.	As soon as there is capacity to work on it.
Why is there a delay?	Scrum focuses the team to deliver on their sprint commitments, and interruptions mid-sprint are discouraged.	Kanban focuses on efficient flow of work, so the top of the backlog is always the next thing to be worked on.
How long does it take to deliver?	It can be 2 weeks or more, depending on sprint status.	As soon as it is completed.

You Can Use Both: Scrumban

Many teams at Salesforce benefit from using parts of both Scrum and Kanban to manage their workload. Often, teams like the structure of the regular planning and review cadence of Scrum, this allows them to manage progress easier. They also use the work-in-progress limits of Kanban to respond to urgent work while minimizing disruptions that Kanban provides.

TRAILHEAD

Project Manager - Important Ally

Key Responsibilities

- Business requirements success
- Project completed on time
- Project completed within budget
- Project goes smoothly/minimize friction
- Manage people/resources
- Foresee and mitigate risks
- Manage expectations
- Be the diplomat/adult in the room

Can a PM do it all alone?

NO!

- PM is rarely as technical as a Developer/Engineer/Architect
- PM relies on the tech/dev team to get accurate feedback
- If a PM cannot get straight answers, PM starts to lose control and everyone starts sharing the stress
- Common PM Questions
  - Where are we with regard to deliverable “X”?
  - What are our blockers?
  - Have you documented the changes?
  - Who is covering for you when you’re out?
  - Do you foresee any risks?

Good vs. Bad PM

Good PM

- Does not commit the team unilaterally
- Carefully manages scope creep
- Embodies team spirit
- Welcomes challenges
- Assumes responsibilities
- Calms the team and the client
- Inspires the team
- Encourages the team
- Leads by example
- Does not blame
- Does not throw teammates “under the bus” in meetings
- Minimizes number of meetings
- Keeps stand-ups short and sweet
- Removes friction

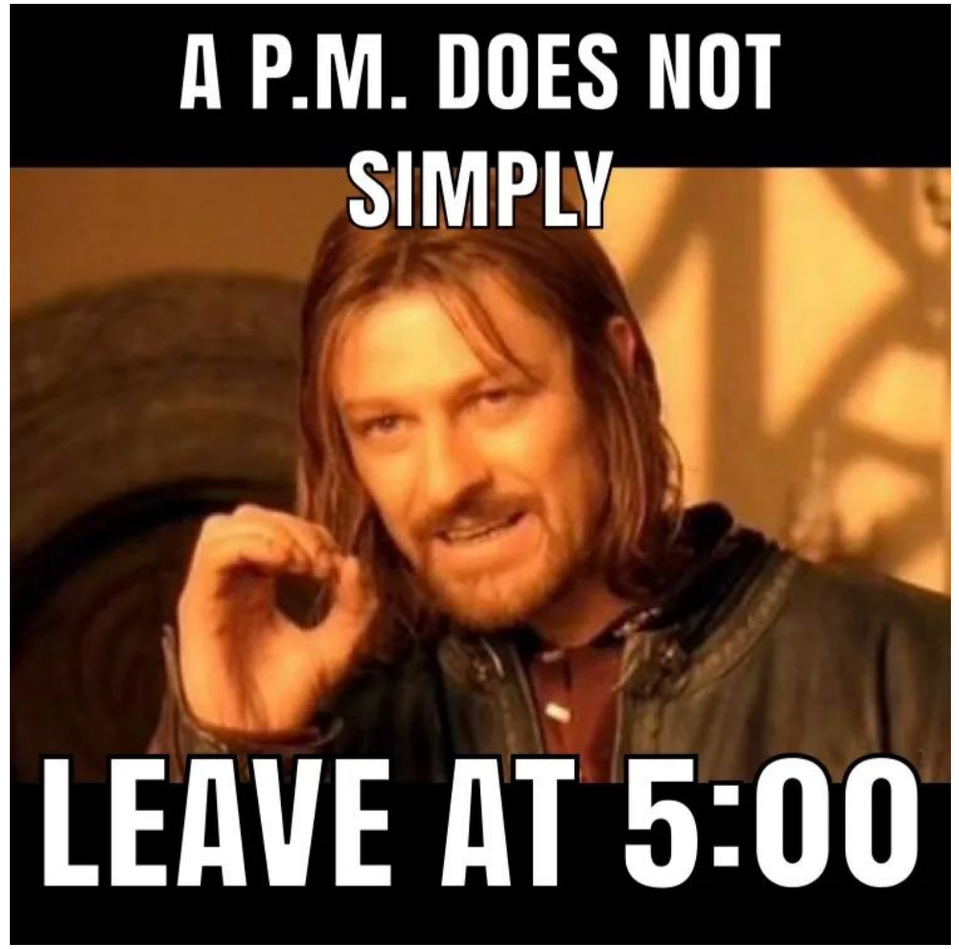
Bad PM

- Commits the team unilaterally
- Does not manage scope creep
- Dictatorial
- Afraid of challenges
- Offloads responsibilities onto others
- Stresses out the team
- Depresses the team
- Sets a poor examples
- Blames others for own and client’s shortcomings
- Frequently throws people “under the bus” publicly
- Creates unnecessary meetings
- Extends stand-ups
- Increases project friction

Working with PMs

Good PM Alliance

- Realize that being a good PM is hard, being a great PM is magical!
- Share responsibilities
- Align early
- Make your PM trust you by keeping them in the loop
- Communicate risks and doubts to your PM
- Think of mitigation strategies and communicate them to your PM
- Help the PM manage deadlines
- Identify scope creep early and often
- Take pride and responsibility for your work
- Be a friend
- Realize that your project may often not be the only project your PM is handling
- Be ready to handle project issues when your PM can’t be there



Managing your PM

Why is the PM such a nightmare?

- Realize that not all PMs are good at what they do
- Not all PMs are true PMs
- Most likely a PM started out with the best intentions, but then they evolved into a nightmare because they became jaded and afraid

Nightmare PM Management Strategies

- Use the same strategies as you would with a good PM
- Get the PM on your side quickly
- Remember that it’s not your job to be the PM
- Set boundaries - unless something is your fault, your time outside of work is your own
- Stand up for yourself and for your team when appropriate
- Do your job well
- Document everything
- Trust, but verify
- Power move: go over the PM’s head to their manager
- Nuclear option: leave the project

Project Management Tools

Why use Project Management tools?

- They keep everyone aligned and accountable
- They help maximize the chances of success

- Rule #1: any project management tool is better than no tool
- Rule #2: spreadsheets are a horrible idea
- Rule #3: Google Sheets are less horrible, but still sub-optimal
- Rule #4: Transparency is key

Strategies

- Learn best-in-breed tools as early in your career as possible
- Invest time in Jira and Confluence - Gold Standard
- Invest time in Smartsheet
- Look at other options, like Rally
- Try using spreadsheets and Google Sheets and identify pitfalls

Questions to ask

- How do you keep them synchronized?
- How do you ensure version control?
- How do you track who did what?
- How do you share project updates with the client?

- How do you report on progress, risks challenges?
- How will your tool help keep you on track?
- How will your tool help keep you out of trouble?

Risk Mitigation Strategies - Dependencies

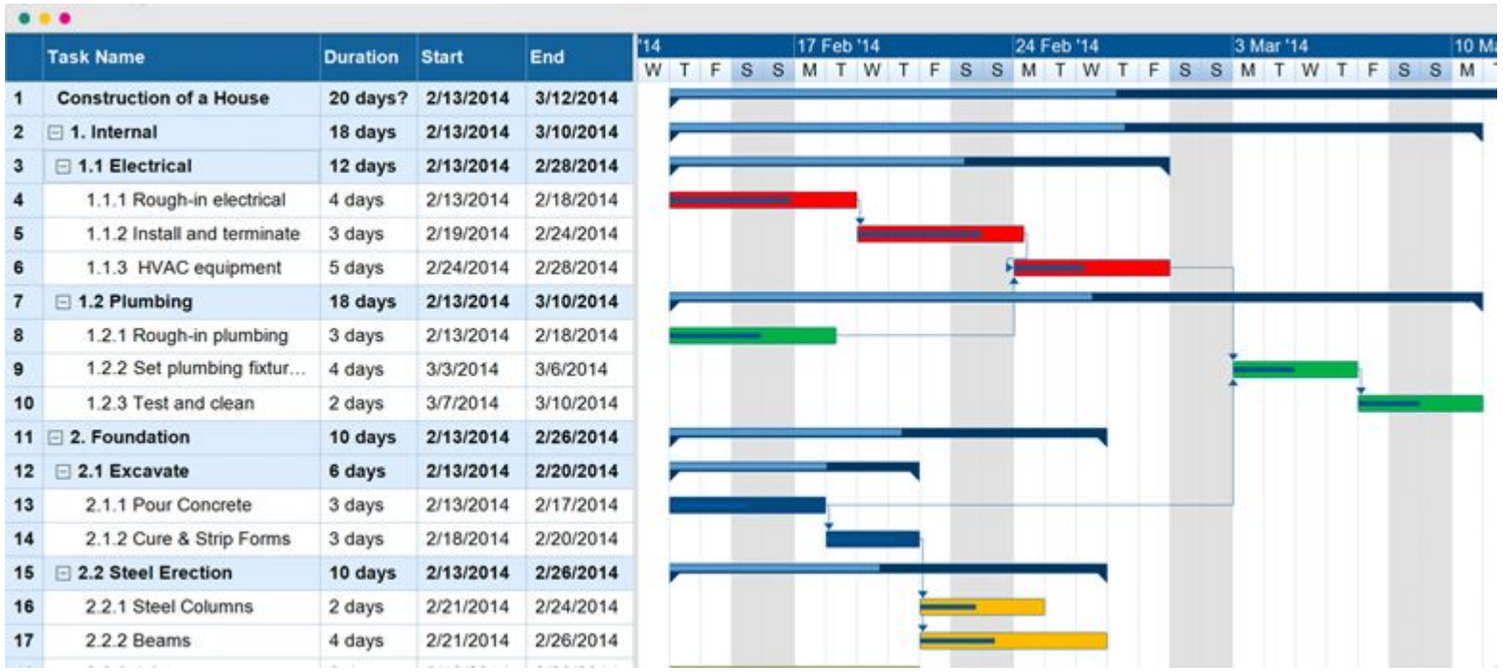
Dependencies

Problem:

Project success can be at risk since we are dependent on progress of something else outside of our control. Example: Mobile App Development (especially native apps), Software Migration.

Mitigation:

Implement strong Project Management Office. Identify dependencies. Plot dependencies on a Gantt Chart. Constantly monitor and if necessary, adjust dependent parts of the projects. Build supporting parts early.



	Task Name	Duration	Start	End
1	Construction of a House	20 days?	2/13/2014	3/12/2014
2	1. Internal	18 days	2/13/2014	3/10/2014
3	1.1 Electrical	12 days	2/13/2014	2/28/2014
4	1.1.1 Rough-in electrical	4 days	2/13/2014	2/18/2014
5	1.1.2 Install and terminate	3 days	2/19/2014	2/24/2014
6	1.1.3 HVAC equipment	5 days	2/24/2014	2/28/2014
7	1.2 Plumbing	18 days	2/13/2014	3/10/2014
8	1.2.1 Rough-in plumbing	3 days	2/13/2014	2/18/2014
9	1.2.2 Set plumbing fixtur…	4 days	3/3/2014	3/6/2014
10	1.2.3 Test and clean	2 days	3/7/2014	3/10/2014
11	2. Foundation	10 days	2/13/2014	2/26/2014
12	2.1 Excavate	6 days	2/13/2014	2/20/2014
13	2.1.1 Pour Concrete	3 days	2/13/2014	2/17/2014
14	2.1.2 Cure & Strip Forms	3 days	2/18/2014	2/20/2014
15	2.2 Steel Erection	10 days	2/13/2014	2/26/2014
16	2.2.1 Steel Columns	2 days	2/21/2014	2/24/2014
17	2.2.2 Beams	4 days	2/21/2014	2/26/2014

What is a Gantt Chart?

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. **On the left of the chart is a list of the activities and along the top is a suitable time scale.** Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity.

Risk Mitigation Strategies - Distributed Teams

Problem: You have geographically distributed teams which worked historically with great autonomy.

Mitigation

Distributed Tasks: Have separated scrum teams which keep each other on the same page through a Scrum of Scrum. The Scrum of Scrums one or more people from each Scrum Team update on each teams status.

Share knowledge: Have regular, team led, knowledge sharing. Encourage horizontal exchange of knowledge between teams using tools like Slack, Fail Fridays, Demo Jam and so forth. Implement a knowledge sharing platform like Confluence.

Produce coherent, standardized work: Quality control define standards for each of the work parts: Requirements, Architecture, Coding and Configuration. The Center of Excellence supports Quality control by implementing processes to encourage and enforce these standards.

Risk Mitigation Strategies - Company Conflicts

Problem: Different Stakeholders of different needs and requirements.

Mitigation:

Steering committee: Align requests and requirements to the Company strategy

Project Management Office (PMO): Strategic project management, allocate resources

Center of Excellence: Different Business Units communicate with each other.

What are responsibilities of a CoE and what are the functions/roles?

Executive Committee:

C-Level & VP-Level (Quarterly)

Manage strategic priorities, Goals and Funding

Steering Committee

Sponsors of Business & IT (Below Executive)

Tactical roadmap, review and prioritize requests

Defining the standards and best practices (Technical & Delivery)

CoE Lead: Coordinates internal & external resources

Members: Admins, Business Analysts, Solution Architects, Technical Architect, Developers, QA Leads, Release Managers

Architectural Review Board: creates guidelines and standards for architecture

QA Team: Helps and supports the Quality

Important Project Roles and Responsibilities

Executive Sponsor: Carry responsibility, approve project scope changes, approve project deliveries, approve funds.

Project Sponsor (Senior Mgmt): Key Business decisions, approve project budget, resources, communicate goals to company

Project Manager: Project plan, manage deliverabilities, recruit staff, lead and manage project team, manage project, identify risks, update management

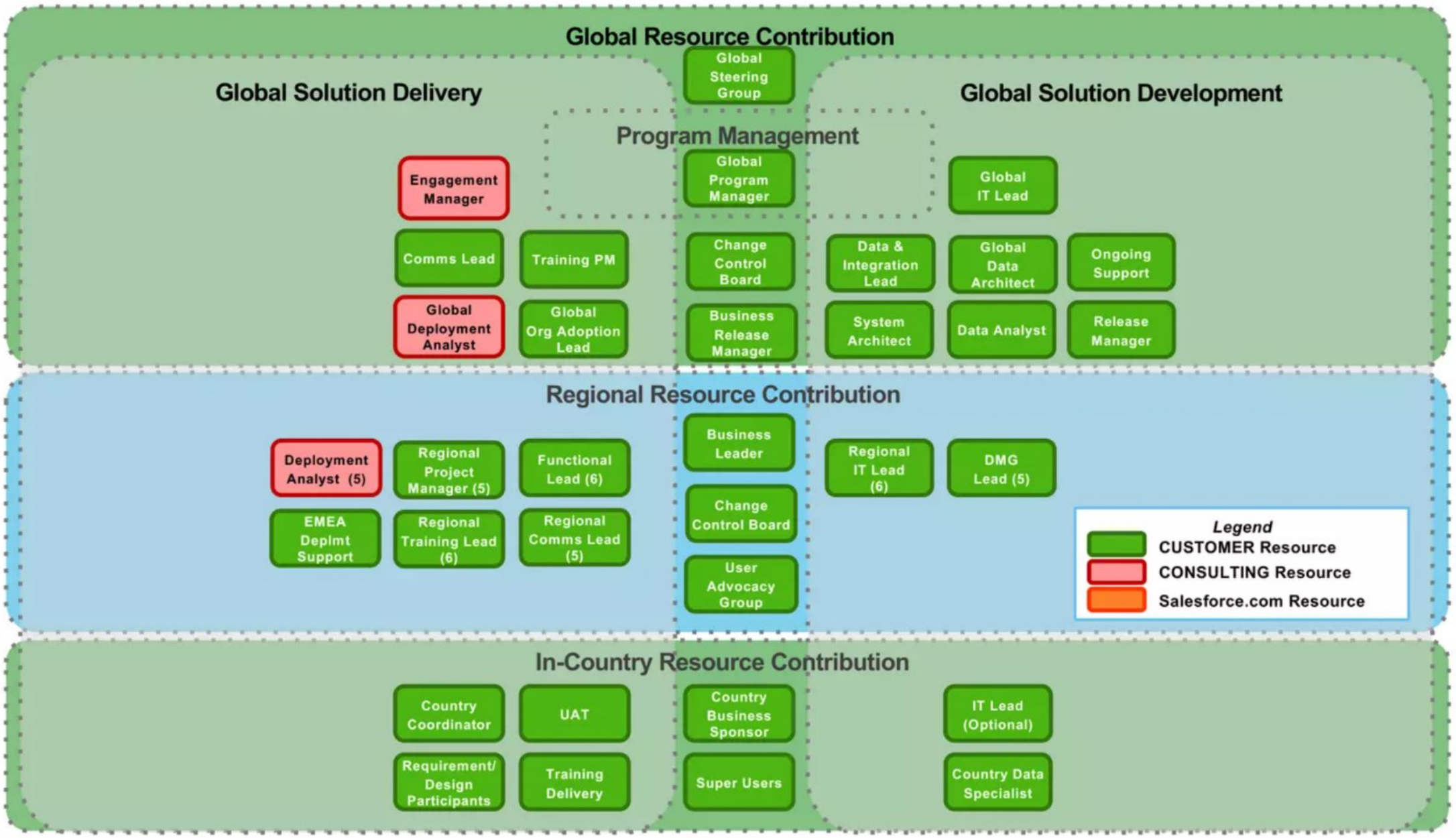
Solutions Architect: Like a main building architect and designer. Come up with an end-to-end solution and own it. Identify technical risks and communicate to the team and especially to the PM. Be client-facing.

Technical Architect: Structural engineer - understand the full extent of the solution, integration components and how they fit into the Salesforce architecture. Give the Solution Architect a reality check Develop strategies to solve complex technical challenges. Lead the technical delivery of Salesforce implementations. Be client-facing.

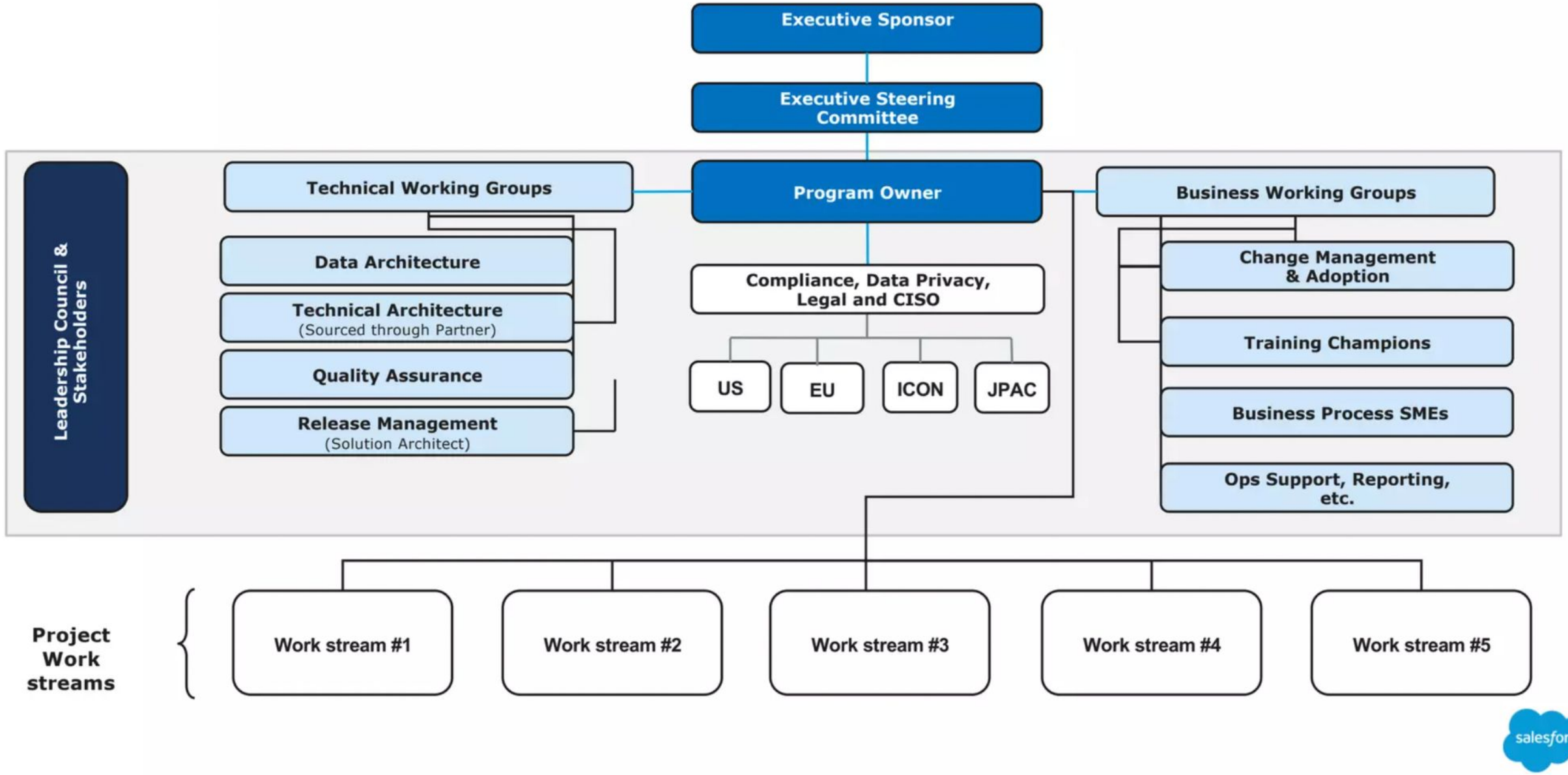
Business Analyst: Assist defining project, gather requirements, document business requirements, verify project delivery.

Project Team Member: Do the best work possible.

Global Sample Model

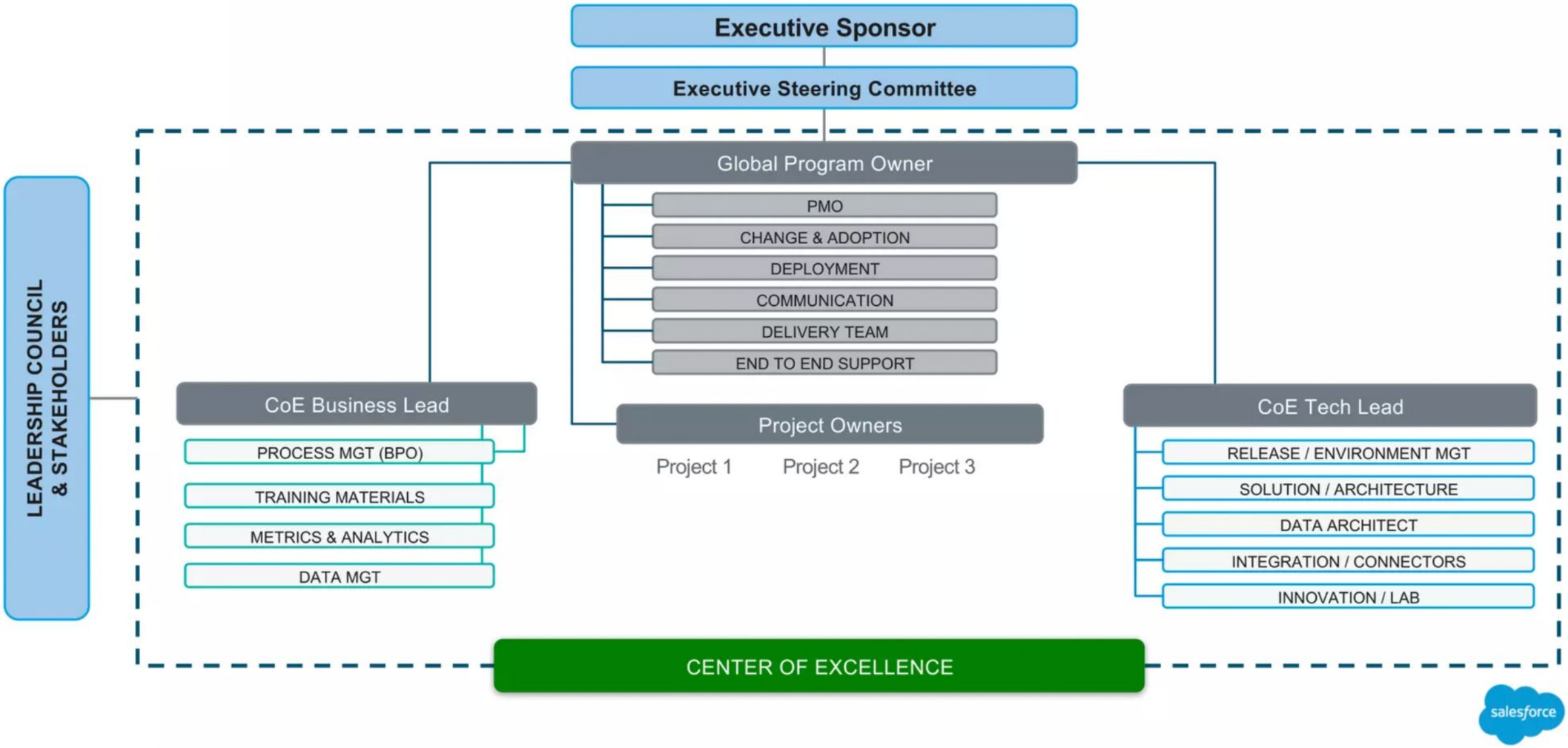


Localized Sample Model



CoE

One Center of Excellence



Executive Sponsor

Executive Steering Committee

CoE Business Leader

Global CoE Owner

CoE Technical Leader

Change Management

Business Operations

Functional Architect

Training and Deployment

Metrics and Analytics

Community

Release Mgmt/QA

Data

Developer

Technical Support

Technical Architect

Portal

Growth Initiatives

Call Center

Marketing

EBC

Field Sales

Inside Sales

Tele Sales

Sales Engineers

Finance/Comp

Renewals

Business Advisory Team
Capability Leads

Customer Sat

Channel

Services

Theatres

US

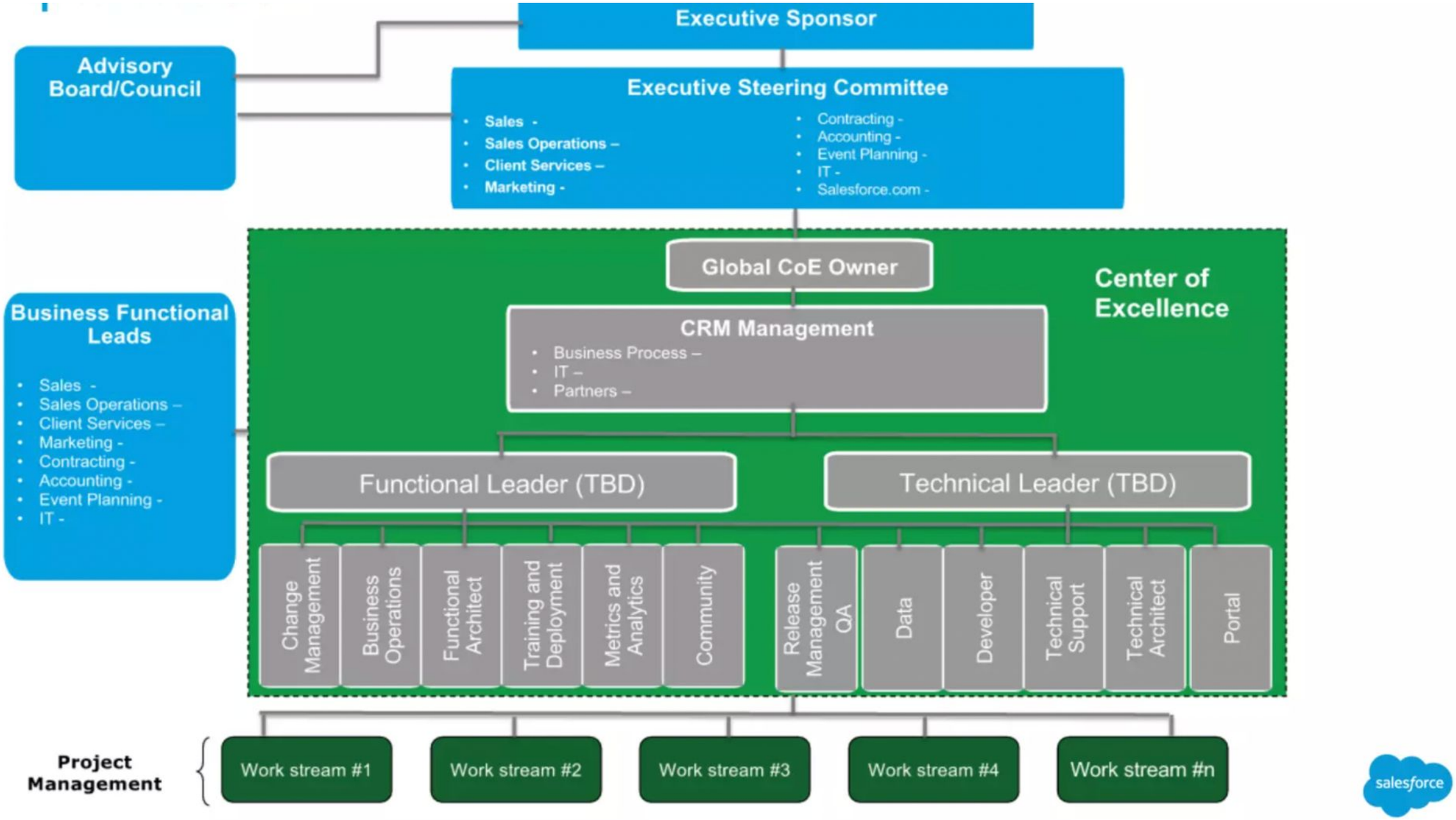
America's International

EMEA

APAC

GOV

salesforce


Executive Sponsor
Advisory Board/Council
Executive Steering Committee
Sales -
Sales Operations –
Client Services –
Marketing -
Contracting -
Accounting -
Event Planning -
IT -
Salesforce.com -
Global CoE Owner
Center of Excellence
CRM Management
Business Process –
IT –
Partners –
Business Functional Leads
Sales -
Sales Operations –
Client Services –
Marketing -
Contracting -
Accounting -
Event Planning -
IT -
Functional Leader (TBD)
Technical Leader (TBD)
Change Management
Business Operations
Functional Architect
Training and Deployment
Metrics and Analytics
Community
Release Management QA
Data
Developer
Technical Support
Technical Architect
Portal
Project Management
Work stream #1
Work stream #2
Work stream #3
Work stream #4
Work stream #n

Testing

Testing type	Description	When & where
Unit testing	The process of testing each unit of code in a single component. This testing is carried out by the developer as the component is being developed.	During development on a developer environment
Code review	Systematic examination of computer source code. It is intended to find mistakes overlooked in software development, improving the overall quality of software. Reviews are done in various forms such as pair programming, informal walkthroughs, and formal inspections.	At the moment of merging, commiting code on a developer environment
Functional Testing	A type of black-box testing that bases its test cases on the specifications of the software component under test. Functional testing usually describes what the system does.	After development of a certain piece of functionality, on the QA environment
Integration testing	Testing the interface between the modules; it can be top down, bottom up, big bang.	After merging multiple functionalities or features on the CIT environment
System Integration testing	A high-level software testing process in which testers verify that all related systems maintain data integrity and can operate in coordination with other systems in the same environment. The testing process ensures that all subcomponents are integrated successfully to provide expected results.	SIT phase on the SIT environment
User Acceptance testing	Last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications.	UAT phase on the UAT environment
Performance testing	Includes Stress and Load testing. Coordinated with Salesforce,. Done using LoadRunner, SilkPerformer, Red View, ...	Scheduled during development, SIT, UAT
Smoke testing	Preliminary testing to reveal simple failures severe enough to reject a prospective software release	After deployment to any environment
Regression testing	A type of software testing that ensures that previously developed and tested software still performs the same way after it is changed or interfaced with other software	After deployment to any environment
Data migration testing	Testing that data is correctly migrated and that data integrity is maintained between systems.	UAT or Staging phase on the UAT or Staging environment

Key Points

- Test early and often
- Resolve code conflicts ASAP
- If tests fail don't ignore them - this will come back to haunt you later
- Not all tests have to be complicated
- Guide the client through testing
- Tests minimize "surprises"
- Tests build confidence

DevOps

DevOps is a set of practices that combines software development (*Dev*) and IT operations (*Ops*). It aims to shorten the systems development life cycle and provide continuous delivery with high software quality.[1] DevOps is complementary to agile software development; several DevOps aspects came from the *agile* way of working. (Wikipedia)

What are different source control branching strategies?

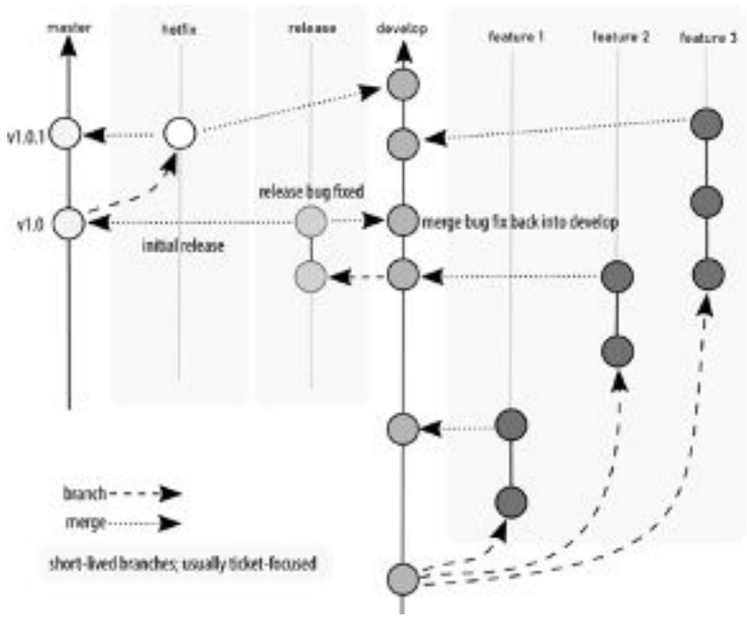
GitHub Flow:

- One master branch which is deployed as fast as possible

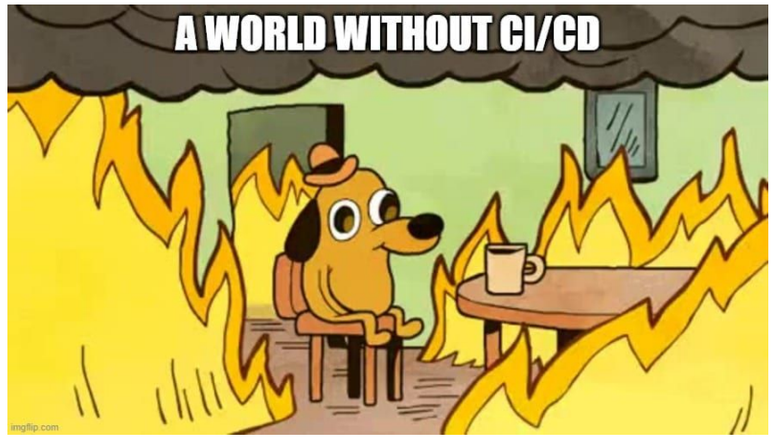
GitFlow

More complex flow:

- Developers develop in “local” feature branches.
- Development branch is the main branch.
- Release branch is fed by the develop branch. Integration and regression testing is done on the release branch.
- Master branch: As soon as Release branch is stable it’s fed into master branch where it’s deployed to Production.
- Hotfix branch is created directly from master branch and fed back into master and dev branch.



DevOps - Continuous Integration & Source Control



What benefits does Continuous Integration (CI) provide?

- Constantly validate deployments and unit tests run on Production prior to deployment day
- Accelerate speed of actual deployment but pre-validating and ensuring deployment success
- Handles merge conflicts with multiple developers / teams
- Requires Source Control: Significantly increased governance (to review changes per engineer/dev/admin; pull reviews to Reject/Accept work

Describe a CI driven deployment process

1. Dev/Admin pulls metadata from sandbox
2. Dev/Admin commits work to source control (check in of changes)
3. Validate deployments are run against a sandbox that mimics production (or Production itself) to ensure new work continues to deploy
4. Dev/Admin raises pull request after successful CI
5. Merge/Approval of Pull Request triggers CD process (deployment to higher environments)"

What are the benefits of Source Control Management

- Allow you to move the source of truth from Salesforce into a centralised repository
- Able to deploy your application to other Salesforce Orgs (that aren't necessarily connected)
- Track changes
- Enable easier rollback
- Allow versioning of changes
- Manage conflicts of having multiple developers working on shared components
- Manage code reviews

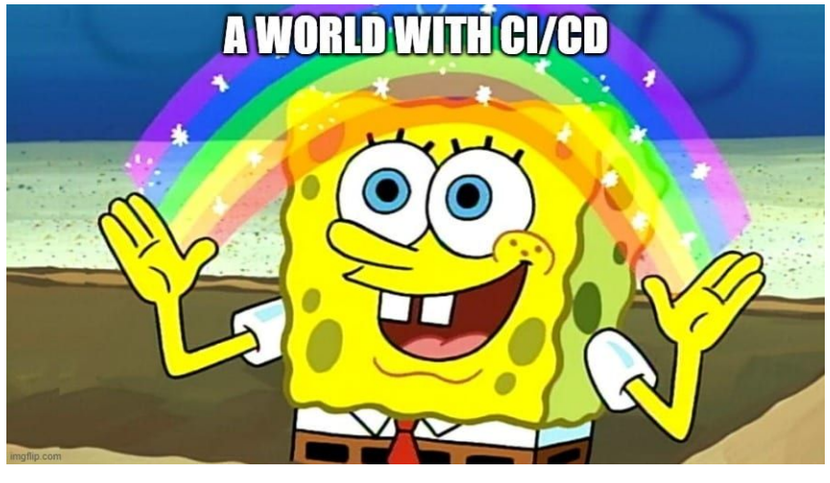
DevOps - Continuous Delivery

What is Continuous Delivery?

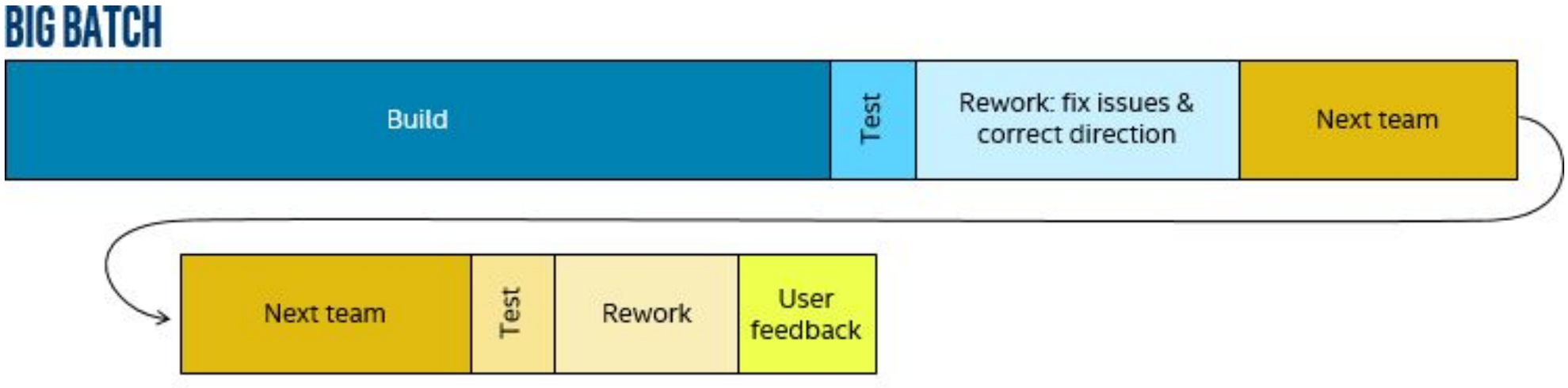
- CD is a method that allows you to deliver changes to production in a quick, safe and sustainable way.

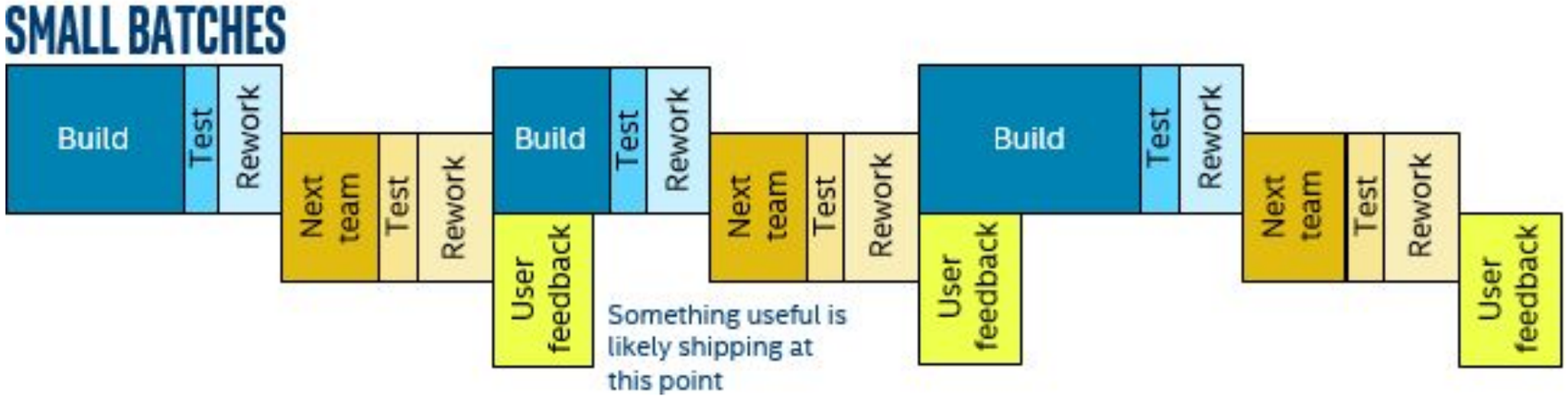
What is the difference between Continuous Integration and Continuous Delivery? (Atlassian)

- Continuous delivery is an extension of continuous integration since it automatically deploys all code changes to a testing and/or production environment after the build stage.
- On top of automated testing, you have an automated release process and you can deploy your application any time by clicking a button.
- In theory, with continuous delivery, you can decide to release daily, weekly, fortnightly, or whatever suits your business requirements. However, if you truly want to get the benefits of continuous delivery, you should deploy to production as early as possible to make sure that you release **small batches** that are easy to troubleshoot in case of a problem.



Big vs. Small Batches





Rule of Thumb: small batch releases are faster and more effective.

Why?: you are likely to uncover problems/conflicts earlier in the process.

DevOps Tools

Why use them?

- To not go crazy and hate life (e.g. enjoy your evenings & weekends)
- To automate the deployment pipeline as much as possible
- To ship code faster
- Infinite other good reasons: fill in the _____ (discussion topic)

Enterprise-level Tools

- Copado
- Gearset
- Prodly
- Flosum

Static Code Analysis

- Clayton
- Codescan
- PMD

DevSecOps

- Checkmarx

Quick to get started

- Copado Essentials

Questionable, why?
Need to build deployment logic and figure out how to resolve conflicts.

- Jenkins
- GitHub Actions
- CircleCI
- Salesforce DevOps Center

The Worst

- ChangeSets - flat “no”

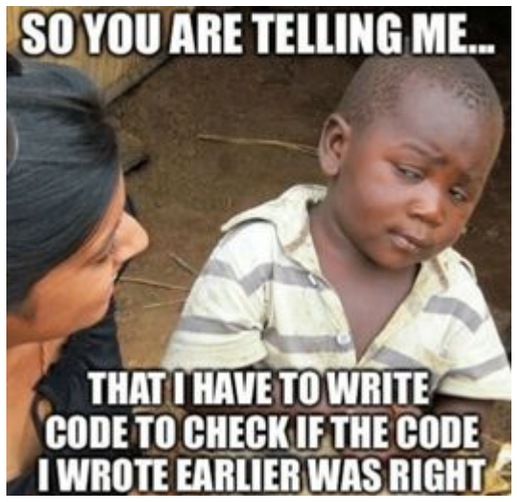
What other tools do you use/like and why? (discussion topic)

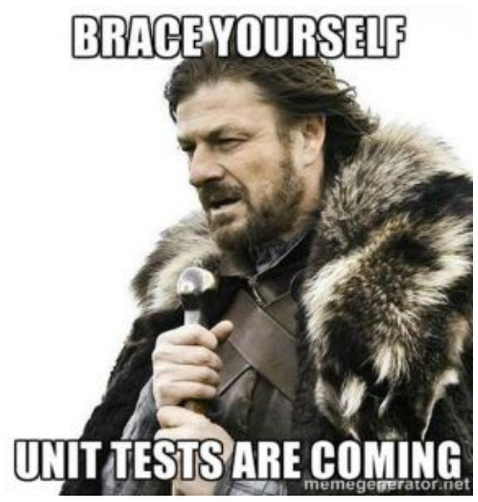
Salesforce Testing Tools

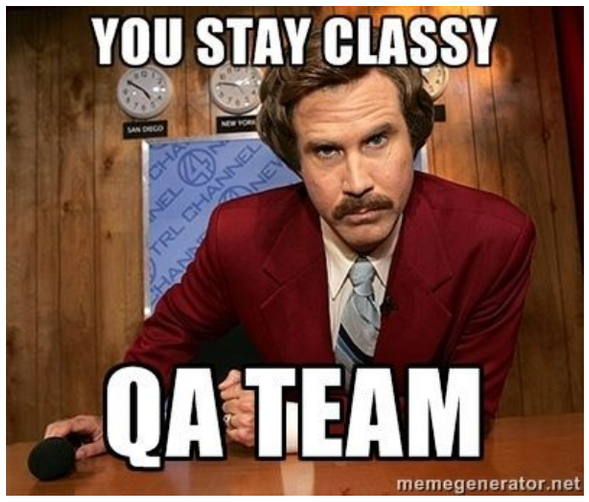
- Provar
- Copado Robotic Testing
- QASource
- AccelQ
- Keysight Eggplant
- Opkey
- Salesforce itself
- People
- *...many more*

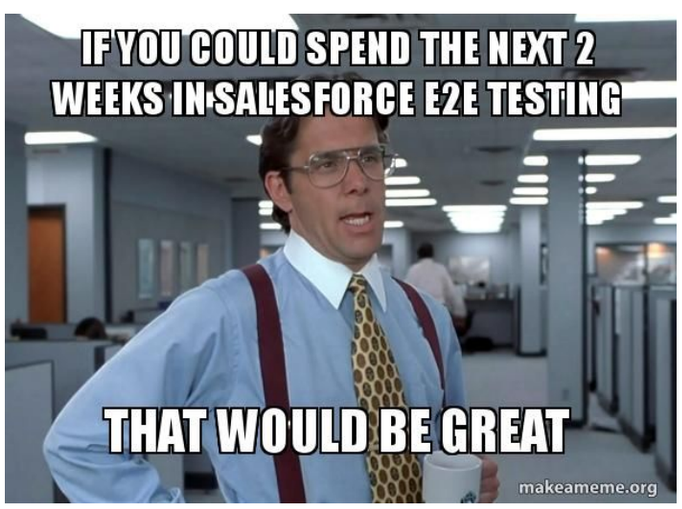
Key Points:

- Test tools are better than no test tools
- No tool will replace people completely
- Human-powered User Acceptance Testing is always going to be important









Elements.cloud

What is it? - Cloud Intelligence Platform (huh?)

- Org Discovery
- Impact Analysis
- Multi-cloud metadata dictionaries with automated documentation, dependency trees, where-used & %filled.
- Requirements & feedback
- Process maps
- Architect diagrams
- User stories & Releases.
- DevOps & Jira integration
- In-app help & adoption monitoring

What it is not

- Deployment tool
- Silver bullet - *one-stop-shop solution*
- Write all of your documentation for you

Pricing

Competition

- Strongpoint
- Salto
- Sonar
- Metazoa

Thank you!

Stay tuned in the Slack channel for the next topic and please feel free to suggest areas of interest.

Ways to get in touch...

Connect with me on LinkedIn
Email: svet@dc3me.com

Obvious People Slack Channel:
#salesforce-academy