scram™
SCHEDULE COMPLIANCE RISK ASSESSMENT METHODOLOGY

Scheduling Agile Development Management Challenges

ProjectChat
17-18 September 2019

Angela Tuffley
SCRAM Principal and
Director, RedBay Consulting Pty Ltd

RedBay

Topics

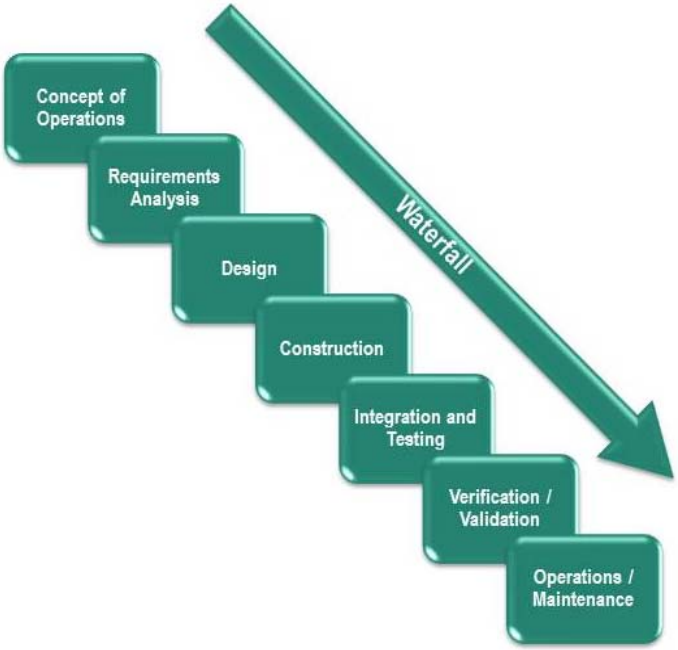
- Traditional vs Agile Methodologies
- SCRAM and Agile
- Scheduling Agile

We're going Agile!

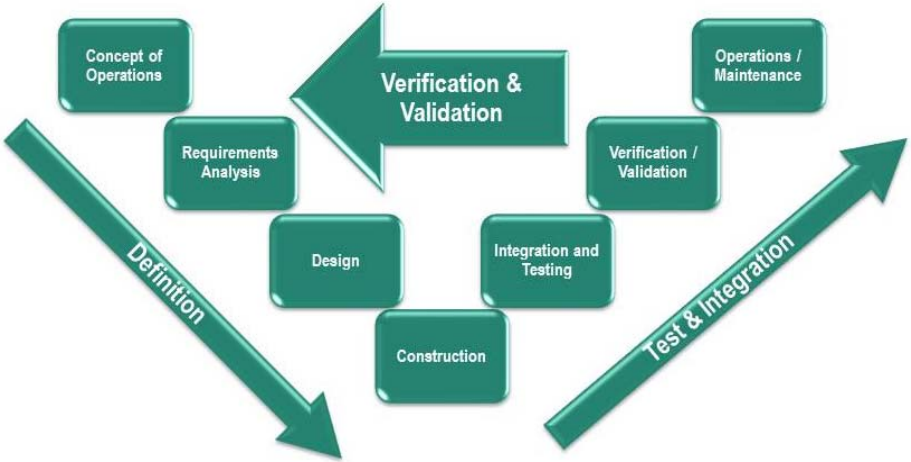
- 13th annual State of Agile survey (2018)
  - 97% of respondents reported their organisations practice agile development methods

Traditional Development Life Cycles

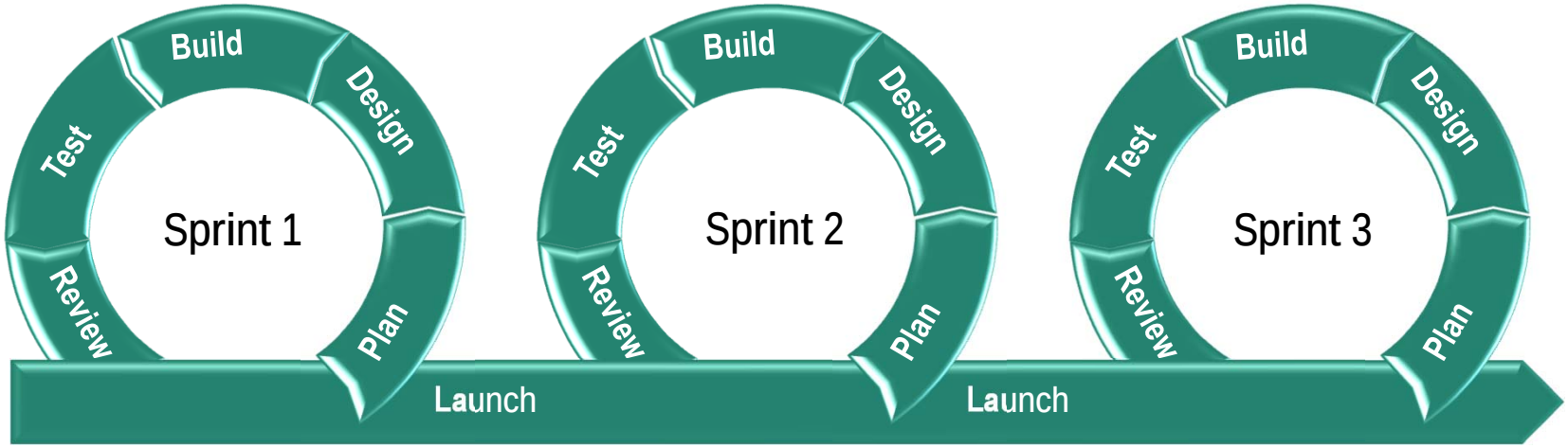
Waterfall



"V" Model



Agile Development Life Cycle



- The Agile Manifesto four key values
  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan

Balancing the two

- The worst of Traditional is to over-emphasise
  - process and tools
  - comprehensive documentation
  - contract negotiation and
  - following a plan
- But if completely ignored development effort will fail
- The worst of Agile is to over-emphasise
  - individuals and interactions,
  - working software
  - customer collaboration and
  - responding to change
- Can result in “cowboy coding”

Agile is

- The group of various software development methodologies that emphasise most or all of these points:
  - requirements evolution
  - iterative development
  - continuous test and integration
  - frequent progress demonstrations
  - frequent delivery of working code
  - on-going, direct communication between the customer and developer
  - self-organizing, cross-functional teams
  - frequent retrospectives promoting continuous improvement

» CMU/SEI-2013-TN-021 *Parallel Worlds: Agile and Waterfall Differences and Similarities*

Some Agile Terminology

A Disciplined Agile process in action

- The customer develops a prioritised *Product Backlog* of high-level requirements
- The customer and the developer jointly agree a high-level plan (goals, budget, schedule) or *Roadmap*
- High-level requirements are put into context and a *Story* created about each requirement
- The developer refines estimates of duration and cost to implement each *Story*
- The *Product Backlog* is *Groomed* to balance customer priorities and developer risks
- The *Roadmap* is broken down into a series of *Releases* to create a *Release Backlog*
- *Releases* are then broken down into *Iterations* or *Sprints* (typically 2 weeks)
- *Iterations* are broken down into *Daily Work* assignments or *Tasks* for the development team
- The development team holds a *Daily Stand-up* meetings
  - Each team member reports what work they completed, what they will work on next, and any issues or roadblocks

A Disciplined Agile process in action

- The code base is *Continuously Integrated* and often tested at least once a day
  - Must pass all tests after integration
- During each *Iteration*
  - The customer cannot add requirements but can clarify requirements already allotted
  - The development team maintains visual displays of progress and issues
  - The *Release Backlog* is groomed and the plan for the next *Iteration* is refined
    - *Grooming* ensures the next few *Iterations* of *Stories* are ready for planning
- At the end of each *Iteration*
  - The developer demonstrates or delivers some working capability to the customer
  - Once the customer is satisfied, the capability is prepared for *Release*
  - Any unfinished capabilities are put back into the *Release Backlog*.
    - Reprioritisation may occur at this point as priorities may have changed
  - The customer and the developer hold a *Retrospective* to review what worked and what didn't work.
- At the end of each *Release*
  - The developer delivers at least one working capability to the customer

Getting Results

Traditional

Results at the end of development
(could be years)

Agile

Results every Sprint (or Iteration)

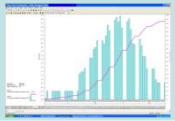
Topics

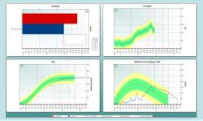
- Traditional vs Agile Methodologies
- SCRAM and Agile
- Scheduling Agile

What is SCRAM?

An independent review to identify issues and risks to schedule

- Quantifies the schedule impact of issues and risks using
  - Schedule Monte Carlo Simulation
  - Software Parametric Modelling





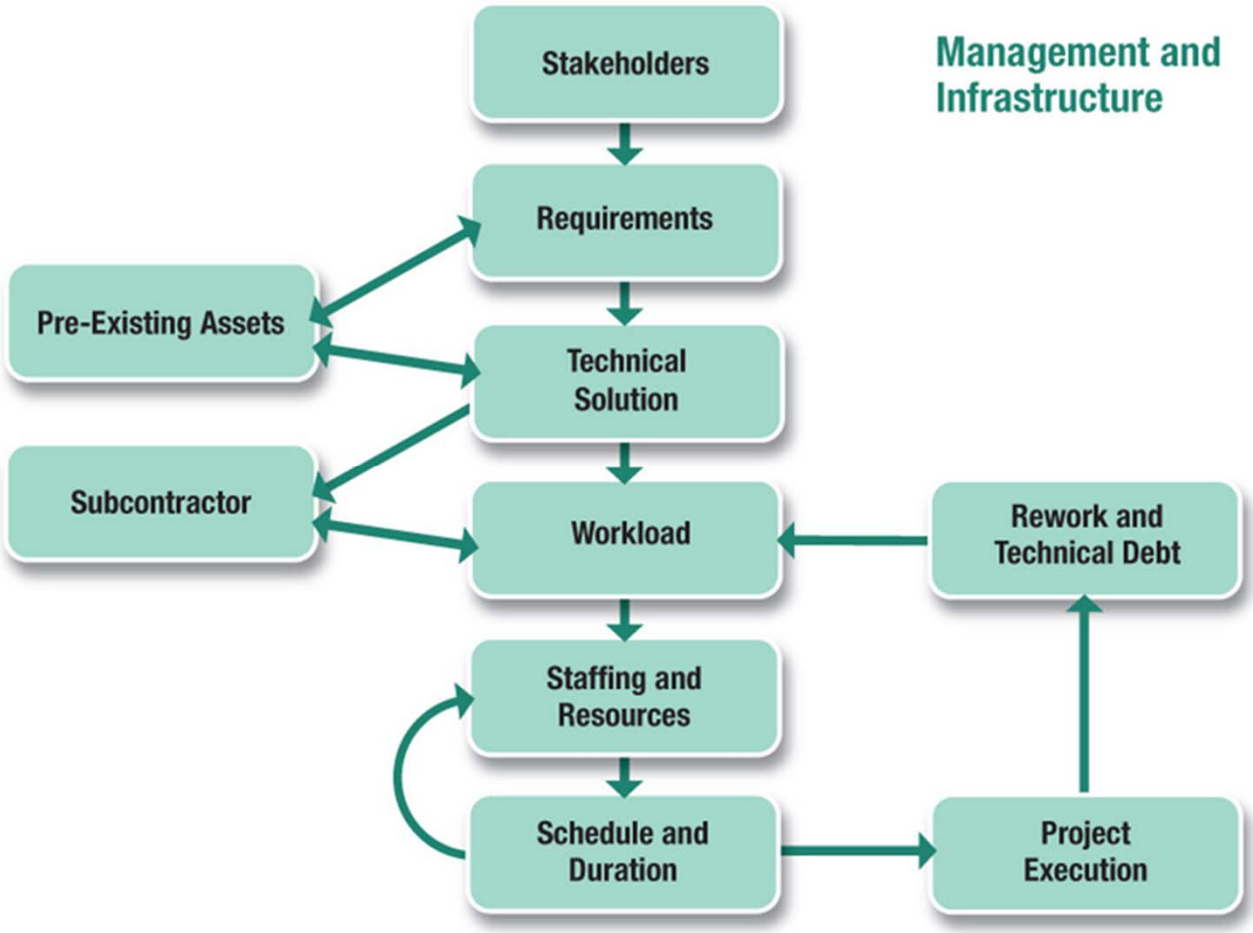
Embodies best practices

- Systems and software engineering
- Schedule development and project execution

Facilitates improved business practices

- Based on feedback from reviews
- Identification of systemic root causes / issues

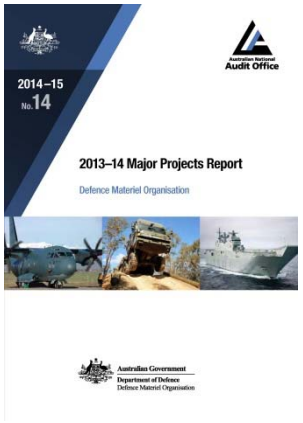
Root Cause Analysis of Schedule Slippage (RCASS) Model



SCRAM Usage

Sponsored by the Australian Department of Defence

- To improve Project Schedule Performance in response to Government concern as identified by the Australian National Audit Office (ANAO)
- Successfully applied to the F-35 JSF Program in the USA and has been used to monitor software development performance on the program (web search “F-35 Australian SCRAM”)



Diversity of SCRAM Reviews

Aerospace

Satellite Ground Stations

Maritime

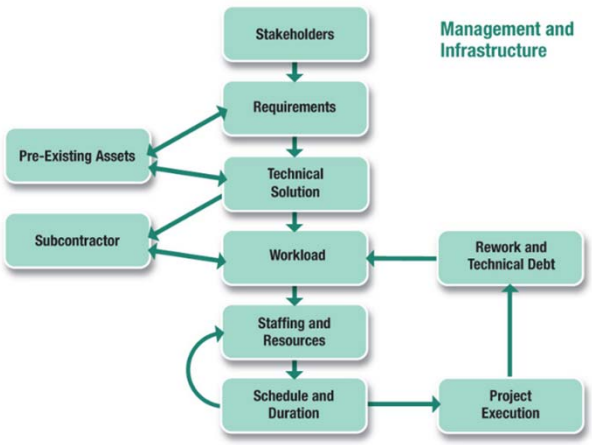
Enterprise Resource Planning

Telecommunications

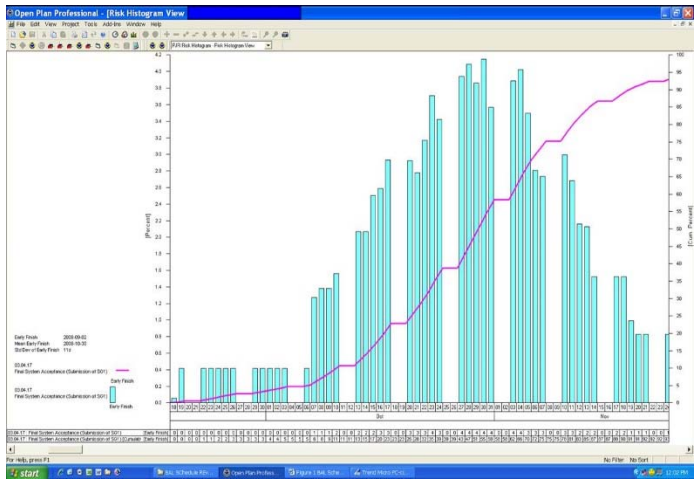
Training Systems

Aligning SCRAM with Agile

- Risk and Issue identification
  - Root Cause Analysis of Schedule Slippage (RCASS)



- Schedule Risk Analysis
  - Monte Carlo Simulation
  - Predicting milestone probability



Why practice Agile?

- Accelerate software delivery
- Enhance ability to manage changing priorities
- Increase productivity
- Improve business/IT alignment
- Enhance software quality
- Enhance delivery predictability
- Improve project visibility
- Reduce project cost
- Improve team morale
- Reduce project risk
- Improve engineering discipline
- Increase software maintainability
- Better manage distributed teams

– *13th Annual State of Agile Survey Report*, VersionOne, 2018; http://stateofagile.versionone.com

Topics

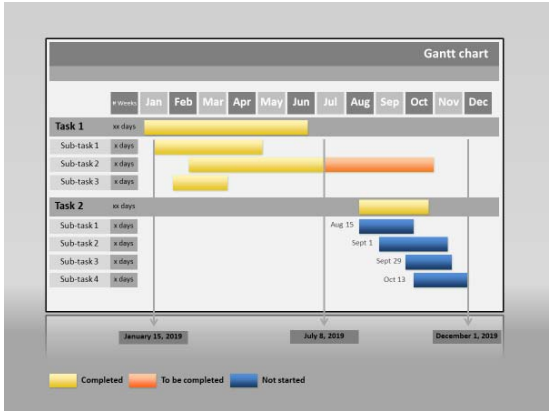
Traditional vs Agile Methodologies

SCRAM and Agile

Scheduling Agile

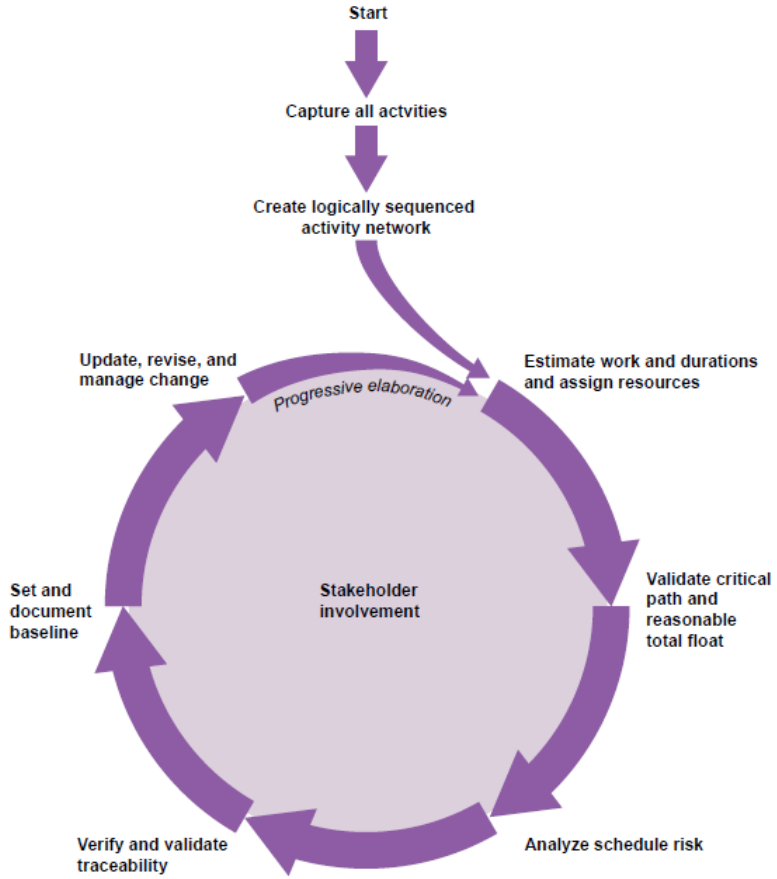
We need a schedule!

- An important part of every SCRAM Review is an assessment of the schedule to ensure it is
  - Properly constructed
  - Aligns with the WBS
  - Can be used for project execution
  - Can be used for Schedule Risk Analysis



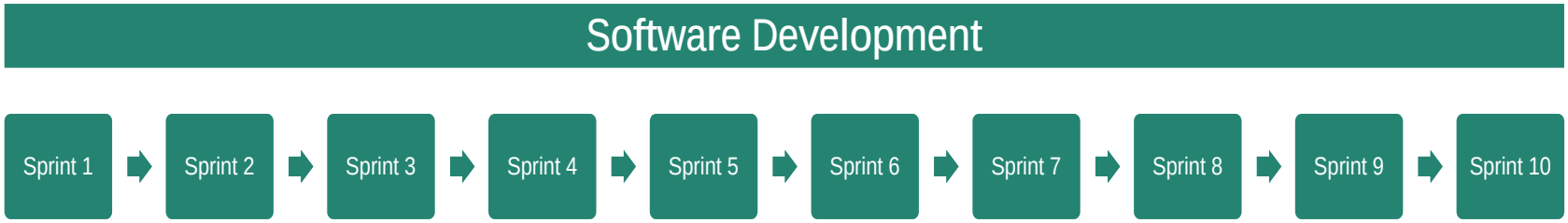
GAO Schedule Assessment Guide

- Describes 10 best practices that should result in a high quality, reliable schedule.

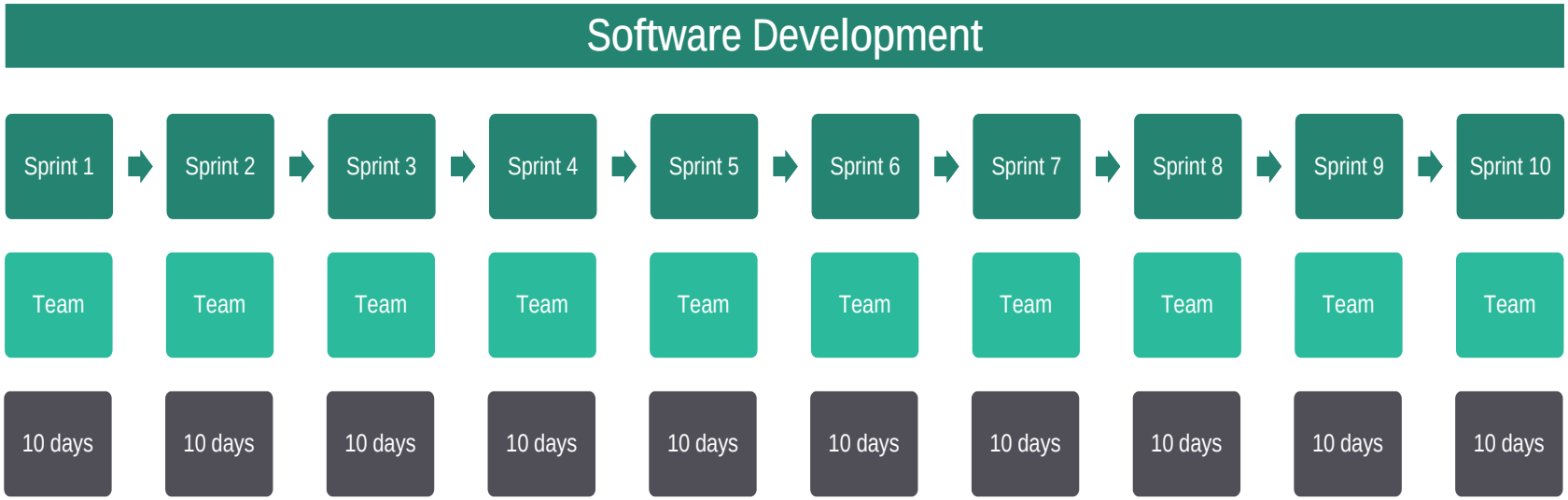


Source: Adapted from Keith D. Hornbacher. | GAO-16-89G

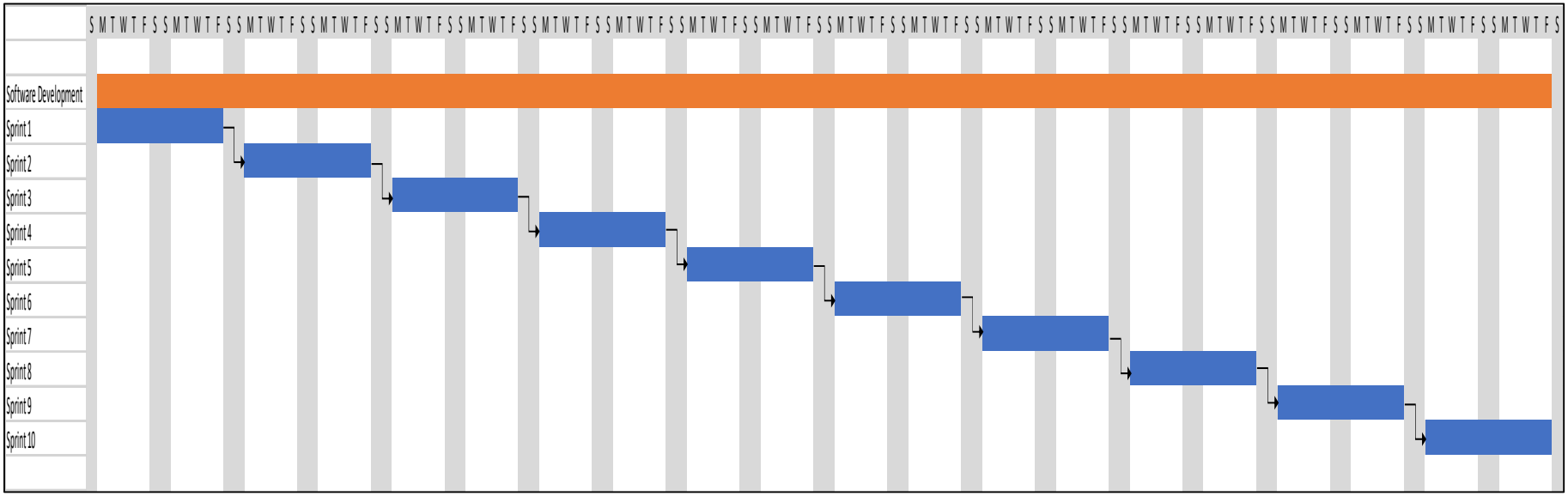
Step 1: Capture all activities and
Step 2: Sequence activities



Step 3: Assign resources and
Step 4: Establish durations



Create the schedule!



Let's STOP there!

Can anyone see a problem with this?

The five levels of Agile planning

- Vision
  - The highest level is strategic in nature and is infrequently changed
- Roadmap
  - The roadmap distils the vision into a high-level plan that outlines work spanning one or more releases; requirements are grouped into prioritised themes, each with an execution estimate
- Release
  - A release is a planning segment of prioritised requirements, along with execution estimates
- Sprint (or iteration)
  - An iteration is a predefined, time-boxed and recurring period of time in which working software is created
- Daily Work (Stand Ups)
  - a brief, daily communication and planning forum where the development team and other stakeholders evaluate the health and progress of the iteration/sprint

» CMU/SEI-2013-TN-021 *Parallel Worlds: Agile and Waterfall Differences and Similarities*

Some more Agile Terminology

- Epic or Epics
  - A connected or bundled set of stories that result in a definable capability or outcome. An epic is a large user story. It is possible to break up an epic into several user stories.
- Features
  - A customer-understandable, customer-valued piece of functionality that serves as a building block for prioritization, planning, estimation, and reporting
  - Different meaning depending on the Agile methodology
    - SCRUM: Backlog Item
    - XP: Story
    - Dynamic systems development method (DSDM): Requirement
- Burn-Up/Down Chart
  - A visual tool displaying progress via a simple line chart representing remaining work (vertical axis) over time (horizontal axis).
- Retrospective
  - A team meeting at the end of every iteration to review lessons learned and to discuss how the team can be more efficient in the future

» CMU/SEI-2013-TN-021 *Parallel Worlds: Agile and Waterfall Differences and Similarities*

Agile Work Breakdown Structures

- This is not a series of two-week sprints

- Recommended to break down the work into three successive levels of effort
  - Start with Epics
  - Then by Features
  - Then by Stories
    - but Stories can be Features depending on the Agile Methodology

- Note:
  - The GAO Agile Assessment Guide: Work Breakdown Structure in an Agile Environment (under development – public release due 2020) plans to expand the WBS discussion based on forthcoming guidance

GAO Agile Assessment Guide: Agile and Scheduling Best Practices (under development – release due 2020)

Step 1: Capture all activities	• Include epics (bundled sets of stories) and features (customer-understandable functionality) from the roadmap that are linked to the backlog
Step 2: Sequence activities	• Reflect work at the epic and feature levels and align with the prioritisation in the roadmap and iteration backlog • Identify any key dependencies between features
Step 3: Assign resources	• During release planning, each team member assesses their availability for development activities
Step 4: Establish durations	• As releases are broken down into iterations with time-boxed durations in Agile, establish the number of iterations to determine the duration of a release • (DO NOT include iterations as tasks)

GAO Agile Assessment Guide: Agile and Scheduling Best Practices (under development – release due 2020)

Step 5: Ensure horizontal and vertical traceability	• The sequenced plan for developing all epics and features, along with all dependency information (Horizontal) • Align the roadmap, prioritised backlog, and burn up/down charts (Vertical)
Step 6: Ensure that the critical path is valid	• Done at the epic and feature levels • Critical path is managed during iteration planning and daily stand-up meetings
Step 7: Ensure that total float is valid	• Tracked at the epic and feature levels (Burn-up/down charts)
Step 8: Conduct a schedule risk analysis	• Provides confidence to deliver all must have features • Identifies risks and opportunities (e.g. team size, management support, availability of tools, etc.)

GAO Agile Assessment Guide: Agile and Scheduling Best Practices (under development – release due 2020)

Step 9: Update the schedule using actual progress and logic

- Update feature development progress at the end of each iteration
- Display cumulative results for all features and epics through burn up/down charts

Step 10: Maintain a baseline schedule

- Baseline against the roadmap and release plans to measure schedule variances
- At the end of each release, conduct retrospectives to capture lessons learned, reduce future risks, improve customer commitment, and motivate development teams

Wrapping Up

- Doesn't have to be this way!

But does one size fit all? – NO!

What to do

- Understand the Agile process being used
  - Clarify the terminology being used
    - Features, Stories, Releases
- Plan to control at the Release / Feature / Story level
- Participate in Retrospectives
  - At the end of every iteration
- Utilise Agile project management tools like JIRA

What not to do

- Schedule all the sprints (iterations)
- Insist on detail before coding begins
- Control the Daily Stand Ups
- Leave them to get on with it

Finally – Beware!!

- Fake Agile (Fragile)
  - A project that declares itself Agile but doesn't embrace Agile; such a group typically dictates its own delivery schedule and stops writing documentation but doesn't adopt test-driven development or any other practice they dislike

- ScrumBut
  - "We do Scrum – but we don't have a product owner"
  - "We do Scrum – but the project manager allocates tasks."

» CMU/SEI-2013-TN-021 *Parallel Worlds: Agile and Waterfall Differences and Similarities*

RedBay