GAO Scheduling Best Practices Applied to an Agile Setting

by Juana Collymore and Brian Bothwell

April 15, 2015

GAO
Accountability • Integrity • Reliability

Outline

- Why is scheduling important?
- GAO Schedule Assessment Guide Overview
- Status of the Final Schedule Guide Version
- Agile Appendix Overview
- Scheduling 10 Best Practices

2

A Reliable Schedule is Necessary for Successful Program Management

- Developing an integrated schedule is key for managing program performance and is necessary for determining what work remains and the expected cost to complete it.
- The success of any program, therefore, depends in part on having a reliable schedule of
  - When the program's set of work activities and milestone events will occur,
  - How long they will take, and
  - How they are related to one another.
- Among other things, a reliable schedule provides
  - A road map for systematic execution of a program
  - The means by which to gauge progress, and
  - A way to identify / address potential problems and promote accountability.

GAO's Schedule Guide – May 2012

- The GAO Schedule guide further develops the scheduling concepts introduced in the GAO Cost guide.
  - Outlines 10 scheduling best practices for developing and maintaining high-quality schedules that forecast credible dates.
  - Contains explanatory text, illustrations, and detailed case studies to help program staff identify a schedule's appropriate schedule logic and risk elements.
  - Includes appendices that list key questions, documentation, etc.
- The project team that develops a project's schedules will find the guide indispensable.
  - Agencies that do not have a formal policy for creating schedules will benefit from using the guide because it will inform them of GAO's criteria for assessing a schedule's credibility.
- GAO is incorporating comments from the Exposure Draft of the guide and will be incorporating them into the final version.
  - GAO has set up a sub group of experts to help develop an appendix that addresses scheduling in an Agile environment.
- The GAO Schedule guide can be downloaded for free at *www.**gao**.gov/products/**GAO-12-120G***

Schedule Assessment Guide Status

- Development of Exposure Draft (Nov 2010 – May 2012)
  - Subject of 5 cost expert meetings
  - Comments received: 548
- Development of Final Draft (May 2012 – Fall 2015 expected)
  - Subject of 3 additional expert meetings
  - Additional comments received to date: 1000+
- Reviewing organizations span private industry (80), government departments/agencies (29), and trade groups/universities (4).
- Final draft will include updated figures, schedule risk analysis, and an appendix devoted to scheduling in an Agile development environment

Agile Appendix Overview

- Many GAO audits for IT systems did not have schedules because they were using the Agile method to develop software
- Purpose of the appendix is to identify common misconceptions about scheduling for Agile projects and dispel them
  - Appendix will describe the applicability and benefits of scheduling best practices for Agile projects with various considerations
  - It will also identify key document differences between Agile and traditional scheduling
- Idea is to get the point across that Agile measures progress through the implementation of working software

Shared Values of Traditional and Agile Schedules

- **Traditional:** Processes and Tools
  - ***Agile:*** *Individuals and interactions*
- **Traditional:** Comprehensive Documentation
  - ***Agile:*** *Working Software/Solutions*
- **Traditional:** Contract Negotiations
  - ***Agile:*** *Customer Collaboration*
- **Traditional:** Following a plan
  - ***Agile:*** *Responding to Change*

7

Scheduling Best Practices Identified in the GAO Schedule Assessment Guide

Our research has identified ten best practices associated with developing and maintaining a reliable schedule.

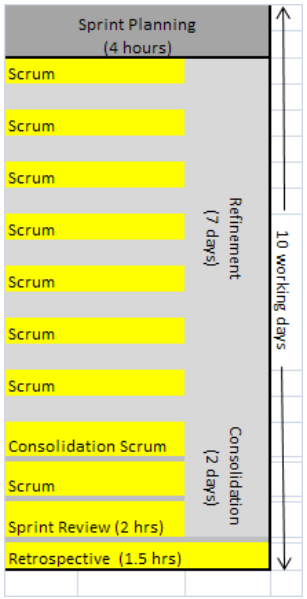
1. Capturing all activities
2. Sequencing all activities
3. Assigning resources to all activities
4. Establishing the duration of all activities
5. Verifying that the schedule can be traced horizontally and vertically
6. Confirming that the critical path is valid
7. Ensuring reasonable total float
8. Conducting a schedule risk analysis
9. Updating the schedule using actual progress and logic
10. Maintaining a Baseline Schedule (new)

Best Practice 1 – Capture All Activities

- Roadmap outlines the basic features (Must Haves) that are planned for the project
  - Understanding is that features will be detail planned over time as more information becomes available including customer/stakeholder feedback
  - Product Backlog contains list of features prioritized as Must Haves, Should Haves, Could Haves and Nice to Haves
  - Agile relies on rolling wave planning where only the highest priority work is selected for detail planning for the current sprint cycle
    - Teams and customers learn more about what the project should deliver as software is prototyped and demonstrated which results in refined requirements after each sprint
- Lowest level for the IMS will be the Release level with the understanding that a specific number of sprints will be planned for each release
  - Number of sprints can change based on user needs and priorities
  - Releases will identify desired features
  - Backlog items are tied to sprints and specific WBS products in order to track what work is expected to be done

Best Practice 2 – Sequencing All Activities

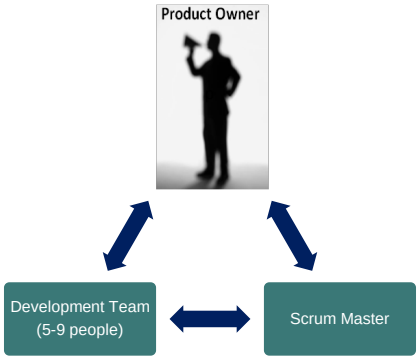
- In an Agile setting, time is fixed with a steady team to complete the work so tracking sprints in a schedule means you are only monitoring level of effort

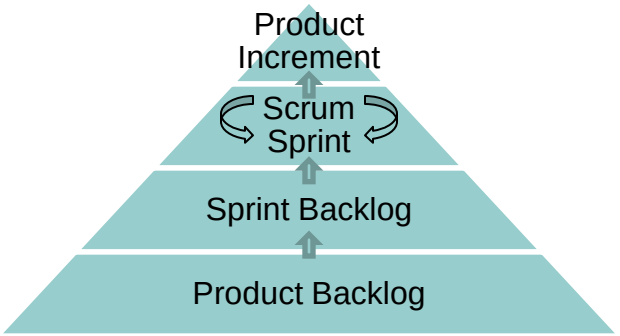


  - You will not know the progress of the effort using a schedule that tracks sprints
    - Therefore, dependencies should be identified at the release level
    - Critical task external dependencies needed within a release (e.g., need a procurement item before starting) then you could model that in the schedule
    - Work is prioritized using the product backlog which ranks features and user stories according to customer needs and preference
  - Better to use Agile metrics (e.g., daily standup meetings, completed stories, burndown charts, etc.) to determine the amount of working software scope being delivered in each sprint

Best Practice 3 – Assigning Resources to All Activities

- Does not apply to Agile sprints since the development team is usually stable, only the scope will be variable.
- Team size should be small (between 5-9 people); there can be several sprint teams working in parallel
  - Management and teams should receive training in the Agile Method
  - Teams should be led by an Agile coach or Scrum Master
  - Teams should be cross functional and self organizing
  - Teams pull work from the prioritized product backlog





Best Practice 4 – Establishing the Duration of All Activities

- In an Agile setting, time is fixed with a steady team(s) to complete the work so tracking sprints in a schedule means you are only monitoring level of effort
  - Releases will be modeled in the schedule and will often have long durations (> 2 months)
    - Release duration will be driven by the number of sprints planned to deliver a set amount of features identified in the Release plan.
      - Duration can be shortened by adding more Agile teams to do the work, by the customer reprioritizing work to be implemented in later releases, or by the customer determining the work is no longer necessary
      - Releases could end up being longer than planned if desired functionality takes more sprints than originally identified in the Release plan
      - Each sprint starts with planning followed by coding, testing, and ending with a retrospective
        - Test driven development means that testing will be done continuously for each sprint resulting in higher quality code because defects are discovered early
    - Sprints will be the same amount of fixed time, typically 2-3 weeks in length so that a cadence can evolve
    - Agile metrics are used outside of the schedule to measure progress
      - After several sprints you can track a team's velocity to better estimate remaining effort

Best Practice 5 – Verifying the Schedule Can Be Traced Horizontally and Vertically

- This best practice only applies to the activities in the schedule down to the release level
  - **Horizontal Integration**: Below the release level, horizontal integration can be determined by examining whiteboards which show using sticky notes or index cards what work has been done, what work is underway, and what work is still left to be accomplished in a sprint.
  - **Vertical Integration**: Below the release level, the reliance on Agile metrics like the burn down chart will provide management and oversight officials information on what work is done and how this corresponds to work status in the release

Best Practices 6 – Verifying that the Critical Path is Valid

- The schedule should reflect the sequence of releases that identify "Must Haves" features so that a critical path can be identified and tracked
  - There will be no critical path below the release level
  - Instead the reliance on Agile metrics is necessary for determining what features and user stories can be delivered in each sprint cycle

Best Practice 7 – Ensuring Reasonable Total Float

- Float will be monitored only to the Release level of the schedule

Best Practice 8 – Conducting a Schedule Risk Analysis

- An SRA can be still be done at the Release level but the nature of Agile is that it is always addressing risks
  - At the end of each release the team asks the customer whether they really need the outstanding features in the backlog or is what they have done good enough
    - Customer decides whether a release should be extended or not to capture desired features
  - Risk in an Agile setting is based on:
    - How good the original estimate is based on assumed velocity
    - Uncertainty regarding new work that arises through the customer discovery phase of completing sprints
  - Scope flexibility is one way to manage schedule slippages in an Agile environment
  - Agile has more feedback loops so you deal with risks sooner and can mitigate them faster
  - During Sprint planning the team assesses what can impact their ability to deploy Must Have promised features
    - Agile develops smaller chunks of software which helps to minimize risk

Best Practice 9 – Updating the Schedule Using Actual Progress and Logic

- This best practice is maintained at the Release level
- For sprints, the following metrics are used to track progress
  - Burndown / burn-up charts, number of story points completed, velocity, ect.
  - Daily standup meetings allow the team to organize their work and identify any impediments they are encountering
  - Stories that are not completed at the end of the sprint are returned to the product backlog
  - If a sprint feature is relying on a dependency from an outside source that is not ready, the team can pick a different story point to work on so that no one is sitting idle.
    - Product owners should be available to reprioritize the work

Best Practice 10 – Maintaining a Baseline Schedule

- In an Agile setting time is fixed with a steady team to complete the work so tracking sprints in a schedule means you are only monitoring level of effort
  - At the end of each sprint you will have data about whether or not you overestimated or underestimated sprint velocity
  - Conducting a retrospective allows the team and the customer as well as stakeholders learn more about the requirements implemented and left to be done
    - Retrospectives also allow for continuous learning by recording lessons learned
  - Customer satisfaction and software quality are captured for better estimating future effort and to prioritize the remaining backlog items
    - Must Haves need to be addressed while Should Haves and Could Haves may never get developed if they are not considered a priority by the customer

How's the government performing?

	Comprehensive	Well Documented	Accurate	Credible
Veterans Affairs (VA)	3.7	3.6	2.3	2.2
DOT	3.5	3.5	3.2	2.3
DOD	3.8	3.5	3.4	2.7
Missile Defense (MDA)	2.8	2.3	2.3	1.8
IRS	3.4	2.8	3	2.3
DHS	3.3	3	3	2.5
DOE	3.4	3.2	2.8	2.2
Agriculture	2.3	1.3	1.3	1.3
Commerce	3.3	2.3	2.7	1.3

19

Invitation to Participate in Further Updates to the Guide

- GAO invites interested parties to meet with us and other experts to discuss further updates to the Guide so that it continually reflects best practices
  - If interested, please e-mail your
  - contact info to:
    - Juana Collymore – collymorej@gao.gov
    - Brian Bothwell – Bothwellb@gao.gov

Backup

Agile Background

The appendix will define what terms GAO will be using to represent Agile development efforts

Term	Definition	Also called
Project Roadmap	High level view of the features the project will set out to accomplish along with the expected business value	Project Vision
Release Plan	Schedule for developing working software that identifies the expected number of sprints and features that will be included in a release	
Epics	High level capabilities	High level Requirements
Features	Next level below an epic which represents items of specific business value. Some features may need several stories to be complete.	Capabilities
User Story	Small chunk of software that identifies business value and success criteria that can be completed within a sprint timebox. A user story defines the work to be done to satisfy a feature.	
Story points	Assessed value of effort for an epic, feature, or user story based on team consensus	
Sprints	short-term, timeboxed effort for delivering an agreed upon number of story points	Iterations, increments
Product Backlog	List of prioritized user stories identified as Must Haves, Should Haves, Could Haves and Nice to Haves	Requirements backlog, feature list
Burn down Chart	Burn down charts represent completed user stories and reflect the rate of progress over time. Can be compared to estimated number of stories to be completed during each sprint for variance analysis.	Burn up chart
Retrospective	Final review of what was accomplished during a sprint and documentation of lessons learned (Agile team and customers/stakeholders attend)	
Velocity	The rate of progress accomplished by the team during a sprint (measures number of story points delivered per sprint to better estimate future work). Velocity reflects a team's cadence and will vary among teams.	Cadence