

Agile Systems Engineering Approach to Software Project Development

Chris Ritter, Daniel Hettema, and Steven H. Dam,
Ph.D., ESEP, SPEC Innovations, 571-485-7807
chris.ritter@specinnovations.com

October 2013

Overview

- What is “Agile?”
- How is agile development applied to software?
- What is the difference between Systems Engineering and Software Engineering?
- What lessons from Agile make sense to apply to systems engineering?
- Summary

What is “Agile?”

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

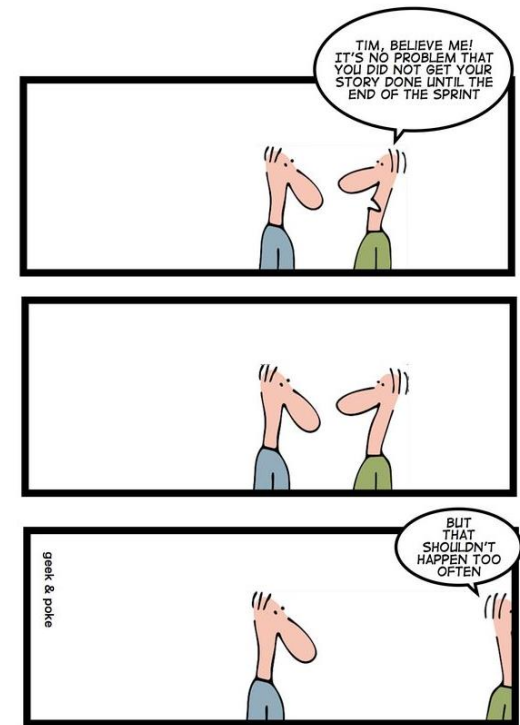
- Time-boxed iterative approach
- Incremental
- Self-organizing, cross-functional teams
- Adaptive planning
- Rapid and flexible response to change

See <http://agilemanifesto.org/>

We have used many names for this over the years: “build a little, test a little” and “extreme programming” to name a couple

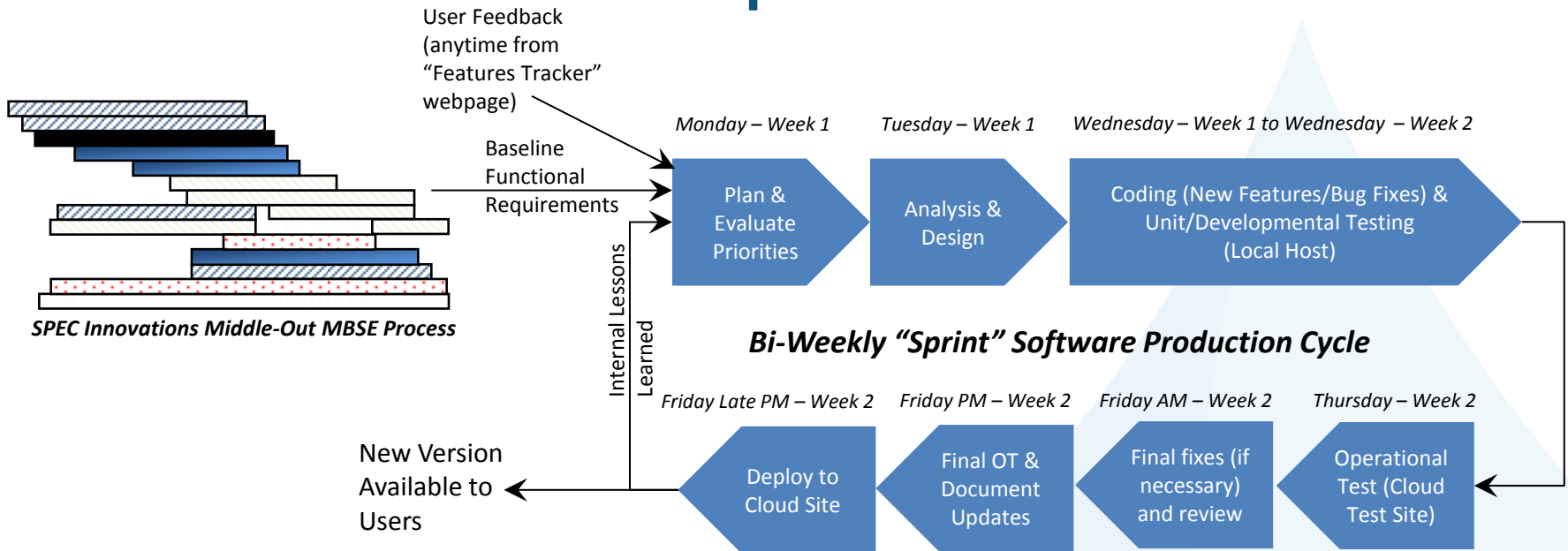
How is Agile Applied to Software Development?

- Many techniques have been derived (e.g., Kaisen, Scrum)
- But does this kind of software development approach translate well into the DoD/NASA world where software makes life and death decisions?



From **AGILE MANIFESTO**
<http://geekandpoke.typepad.com/geekandpoke/2011/12/agile-manifesto.html>

SPEC's Agile Software Production Process – Used for Commercial Software Development



Innoslate SLOC ≈ 370k

We use a small, co-located team who went to school together and have worked together many years



DoD Software Systems

- Characterized by
 - Millions of lines of code
 - Much of it using older software languages
 - Real-time
 - Standards-based
 - Many, many reviews
 - Many, conflicting “requirements” from many, varied stakeholders



*All driven by a very non-Agile
acquisition system*

What Does This Mean to DoD?

- Agile clearly can be applied in non-mission critical activities, for example:
 - Websites
 - Specialized analytical tools
 - “Commercial-like applications” (although it would be better usually to just use COTS)
- It may be applicable to mission-critical software, but only when it’s functional requirements are well-defined and the contracting environment allows for flexibility

What Is the Difference between Systems Engineering and Software Engineering?

- Systems Engineering
 - Capabilities-based
 - Decomposition to many levels
 - Must support all aspects of the lifecycle
 - Hardware SE requires “measure twice, cut once” approach
- Software Engineering
 - Needs to have well-defined functional requirements
 - Verifiable
 - Clear
 - Concise ...
 - Code can be executed quickly

Is There Such a Thing as Agile SE?

The screenshot shows a LinkedIn group page for "Agile Systems Engineering - Wicked Problem Forum" with 1,460 members. The main post is titled "Systems Engineering vs. Software Engineering" by Jesper Nasström. The post discusses the differences between software engineering (decomposed into smaller parts) and systems engineering (integrating sub-systems). A comment by Jeffrey Banks, CSEP, Sr. Systems Engineer at OPS Consulting, LLC, is visible at the bottom, discussing common practices in systems engineering.

Agile Systems Engineering - Wicked Problem Forum
1,460 members

Systems Engineering vs. Software Engineering
Jesper Nasström
ERTMS Signalling and Systems Engineer

We touch on this in some threads, but let's try to get better clarity.

In Software Engineering, the project is about making some software to solve a real problem. Which involves issues such as user interface and the process to be supported. Software almost always is so large that it is decomposed to smaller parts, engineered individually. So in practice, Software Engineering is pretty much a Systems Engineering on a system-of-systems.

In Systems Engineering, I note that system component in the top of the architecture is some application software that glues together the sub-systems. Software allows very sophisticated mechanisms to integrate the system.

What enables software development to be agile is that it is relatively easy to deliver new functionality in small frequent steps. This is not always an option for other systems.

Either you can start the paper mill or you should not. But even if working on a paper mill, engineering teams may show their results frequently. So frequent deliveries are possible, but perhaps not deployed on a working system.

So, what am I missing? When is there a justification to differ Systems Engineering from Software Engineering?

Like (2) • Comment (24) • Follow • Reply Privately • April 5, 2012

Comments

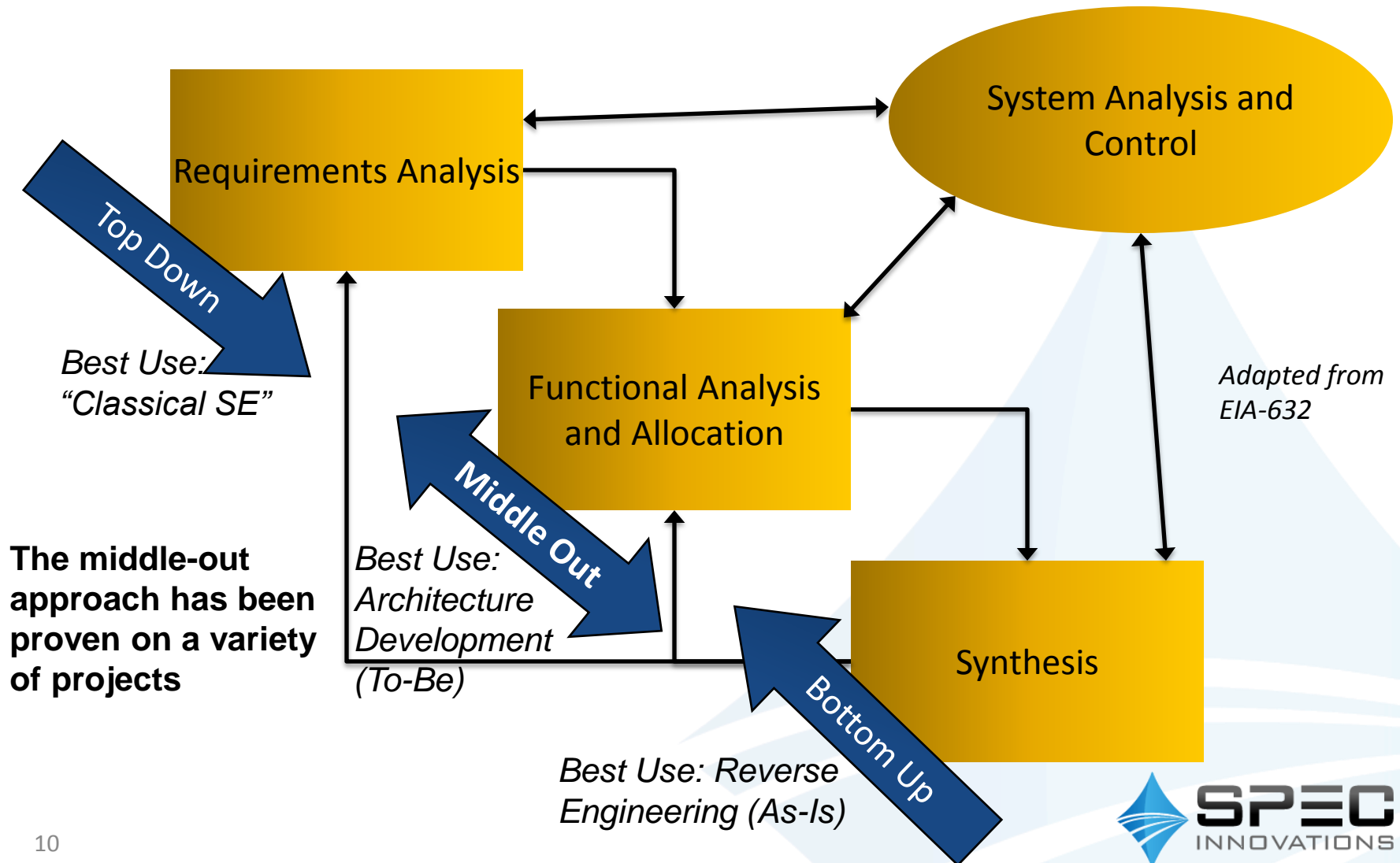
Jeffrey Banks, CSEP, Eirik Midtton like this
24 comments • Jump to most recent comment

Jeffrey Banks, CSEP
Sr. Systems Engineer at OPS Consulting, LLC

Although software and systems engineering do share some common practices, systems engineering is about engineering systems that normally comprise of much more than software. SEs to think about the type of hardware it must run on, the environment it must operate in, the kinds of missions it must support, how well it must support those missions, how it's going to be maintained over it's operational lifecycle, what other system interdependencies it must accommodate, what policies and regulations it must adhere to and so on....

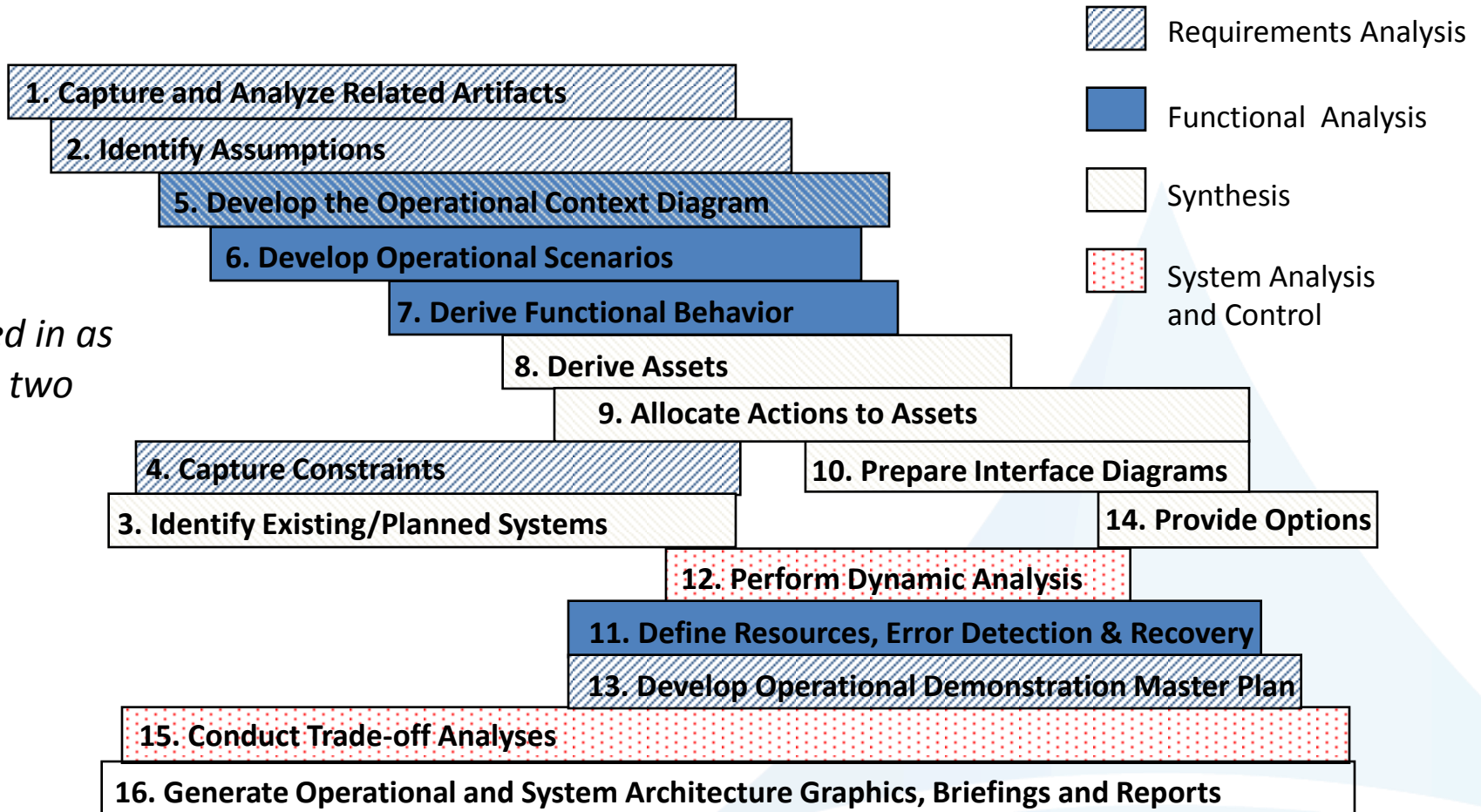
- Many people are trying to apply the Agile principles to SE
- Many forums on this subject (see LinkedIn)
- How have we done this?

We Focus on a Middle-Out Process



Middle-Out Timeline

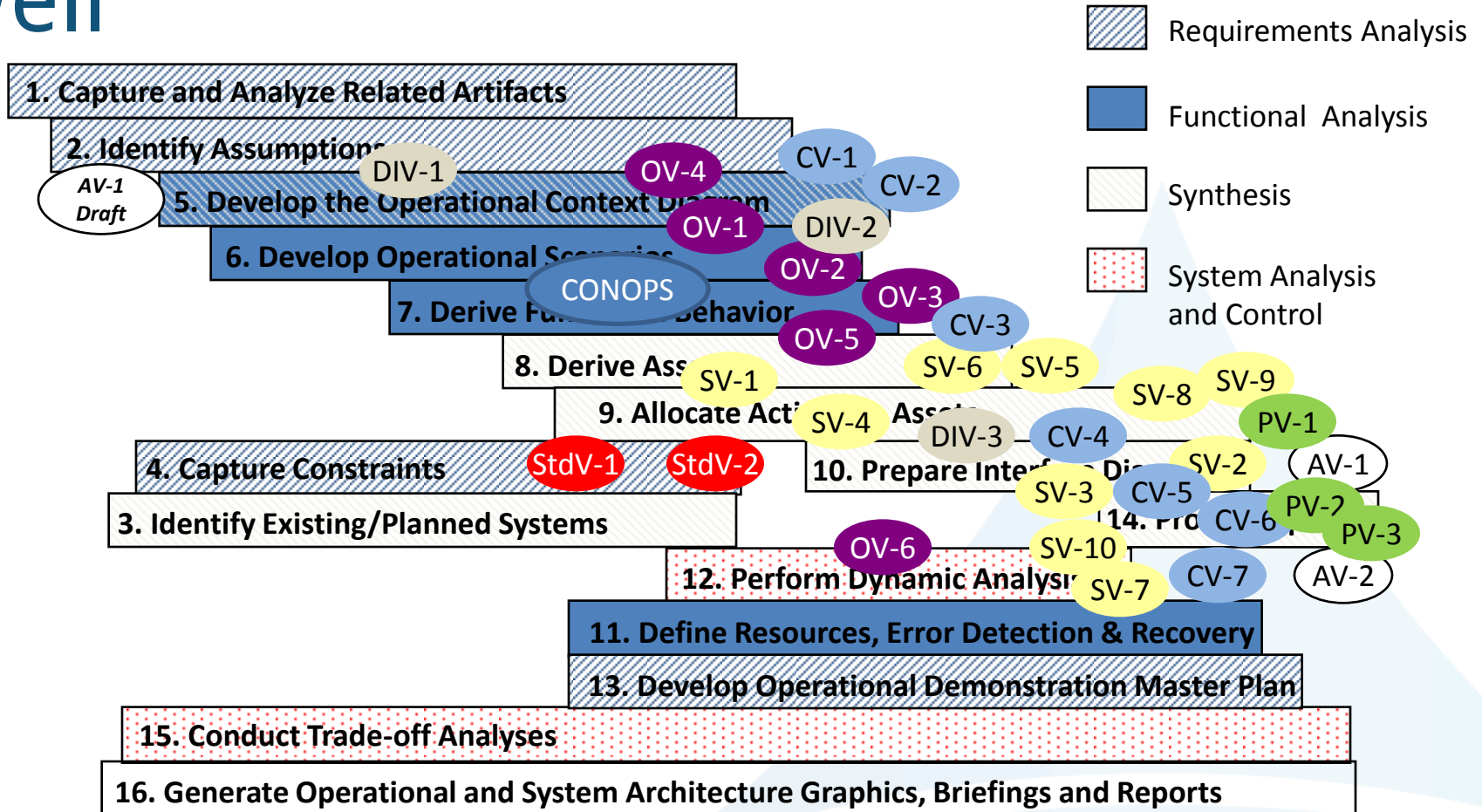
Can be executed in as little as two weeks



Time →

Originally developed to support ACTDs in the 1990s

Agile Architecture Development as Well



Time →

Fits with DoD requirements

What Lessons from Agile Make Sense to Apply to Systems Engineering?



- Individuals and interactions over processes and tools
 - However, you need to apply a process for repeatability and tools to capture the information
- Working software (design) over comprehensive documentation
 - Implies MBSE with simulation
- Customer collaboration over contract negotiation
 - Need to work better as a Government-Contractor team
- Responding to change over following a plan
 - Need flexibility in the SOW and deliverables to make this work
 - Implies CPFF or T&M?

Summary

- Agile is the latest buzzword for something we have been trying to do for decades
- The problem really comes from people being inflexible and suspicious of each other
- A measure of trust is needed to actually improve the situation