



# Thinking About Agile in DoD

**Stephen Welby**  
Deputy Assistant Secretary of Defense for  
Systems Engineering

**AFEI Agile for Government Summit**  
**November 21, 2013**



# US Department of Defense (DoD)



- **World's largest engineering organization**
  - Over 99,000 Uniformed and Civilian Engineers
  - Over 39,000 in the Engineering (ENG) Acquisition Workforce
- **DoD Systems Engineering focused on helping programs succeed**
  - Design, develop, construct and operate complex systems
  - Forecast their behavior under specific operating conditions
  - Deliver their intended function while addressing economic efficiency, environmental stewardship and safety of life and property

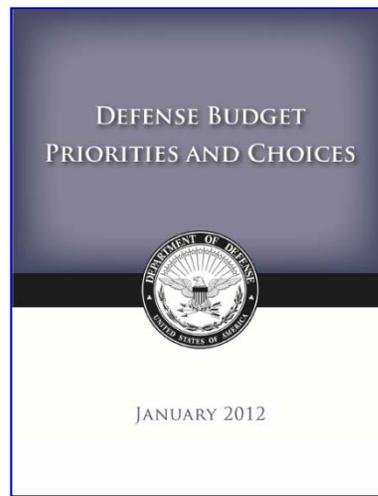
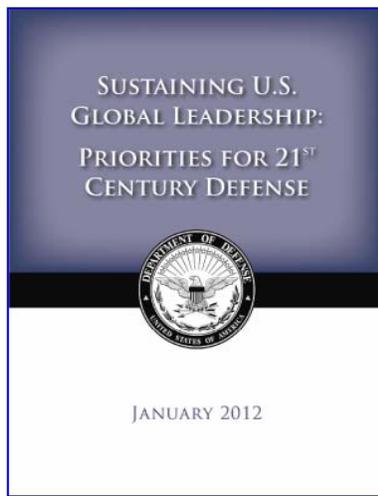
**A Robust Systems Engineering Capability Across the Department Requires Attention to Policy, People and Practice**



# Key Elements of Defense Strategic Guidance



- The military will be smaller and leaner, but it will be agile, flexible, ready and technologically advanced
- Rebalance our global posture and presence to emphasize Asia-Pacific regions
- Build innovative partnerships and strengthen key alliances and partnerships elsewhere in the world
- Ensure that we can quickly confront and defeat aggression from any adversary – anytime, anywhere
- Protect and prioritize key investments in technology and new capabilities, as well as our capacity to grow, adapt and mobilize as needed

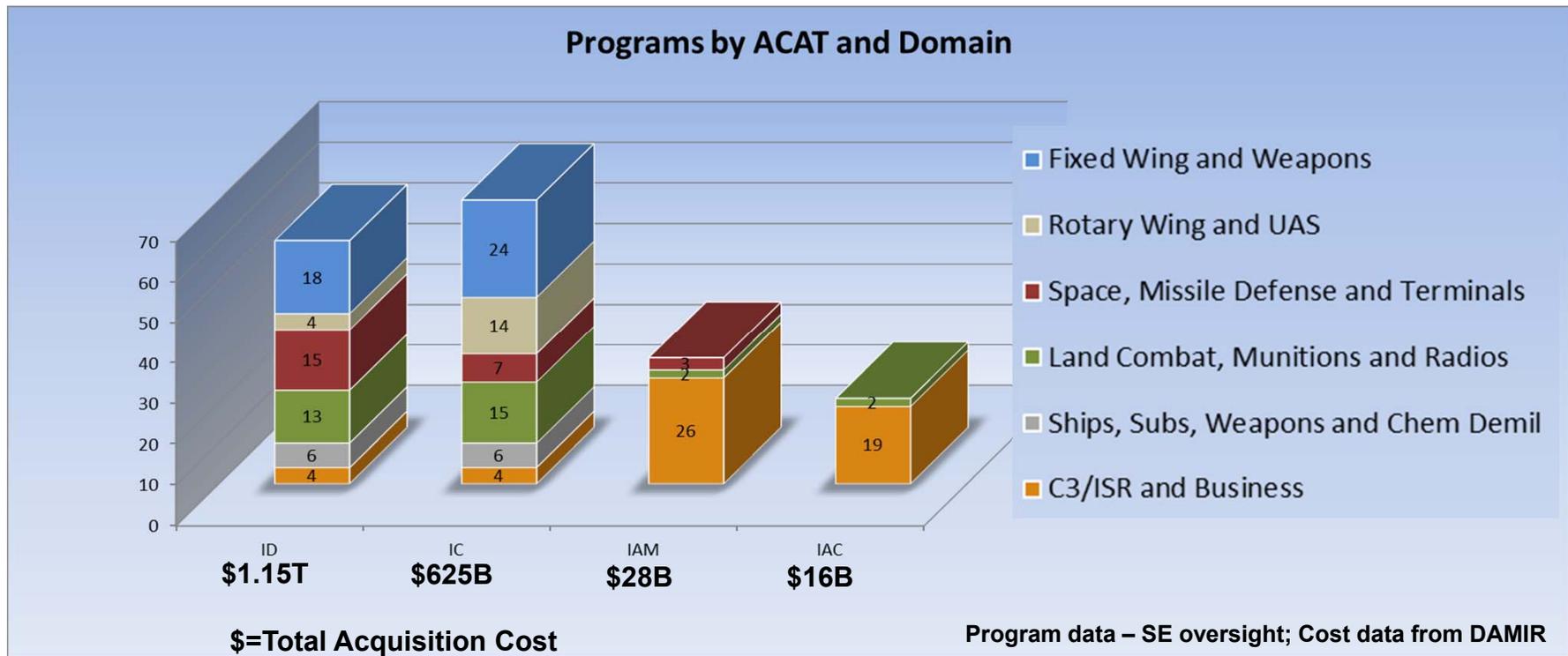




# DASD(SE) Portfolio



- Perform system engineering oversight of **182 programs** with acquisition costs of **\$1.8T**
- Approve System Engineering Plans (SEPs)
- Assess preliminary and critical design reviews (PDR, CDR)





# Preponderance of Acquisition Funding is for ACAT\* ID and IC Programs



- **97% of acquisition funding is in MDAP ID and IC programs**
- **Software applications are components or sub-components of large, complex systems**
- **Software must:**
  - Support system / component / sub-component **requirements**
  - Support overall program / component / sub-component **schedules**
  - Support **integration** with other system software and hardware
- **Software acquisition for 1D and 1C programs poses some of our toughest systems engineering challenges**

\* **Acquisition Category (ACAT) I** programs are Major Defense Acquisition Programs (MDAPs) designated by the Under Secretary of Defense for Acquisition, Technology and Logistics (USD(AT&L)) as a MDAP; or estimated to require eventual expenditure for research, development, test, and evaluation (RDT&E), including all planned increments, of more than \$365 million (Fiscal Year (FY) 2000 constant dollars) or procurement, including all planned increments, of more than \$2.19 billion (FY 2000 constant dollars).

**ACAT ID:** the Milestone Decision Authority (MDA) is USD(AT&L). The “D” refers to the Defense Acquisition Board (DAB), which advises the USD(AT&L) at major decision points

**ACAT IC:** the MDA is the DoD component head or, if delegated, the DoD component acquisition executive (CAE). The “C” refers to component

Source: ACQuipedia <https://dap.dau.mil/acquipedia/>





# Design Agility



- **The only constant for DoD systems is change:**
  - Evolving threats
  - Strategic and tactical innovation
  - Rapid technological change
  - Increased Defense leverage of commercial systems
  - Resource and demand uncertainty
- **These factors all demand increased agility – ability for military systems to be designed explicitly to have capacity to adapt and adjust to maintain relevance and operational advantage in an environment of change**

**Software development agility is a key contributor to Program success**



# DoD Acquisition Environment



- **Governmental regulatory and statutory requirements**
  - Some can be waived
  - Some can be modified
  - Some require an Act of Congress (literally)
- **Broad spectrum of applications and environments**
  - Large, complex systems and system-of-systems
  - Diverse missions, uses, and stakeholders
  - Distributed, heterogeneous development teams





# Software Is Everywhere



- DoD relies on software to provide decisive advantages to our forces
- The complexity required to achieve this advantage demands specific capabilities and tight coupling
- Partial solutions are inadequate
- We can't omit requirements
  - Because they don't fit the schedule
  - Because it simplifies refactoring

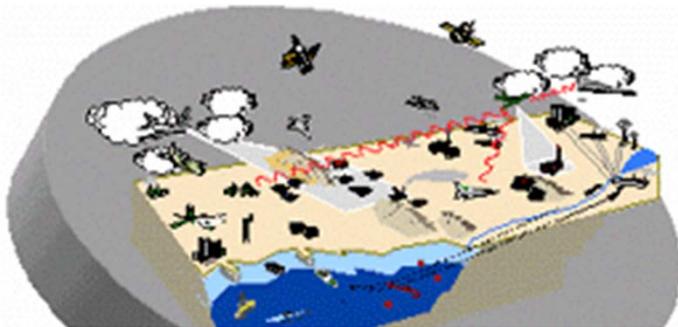


image credit SPAWAR.navy.mil

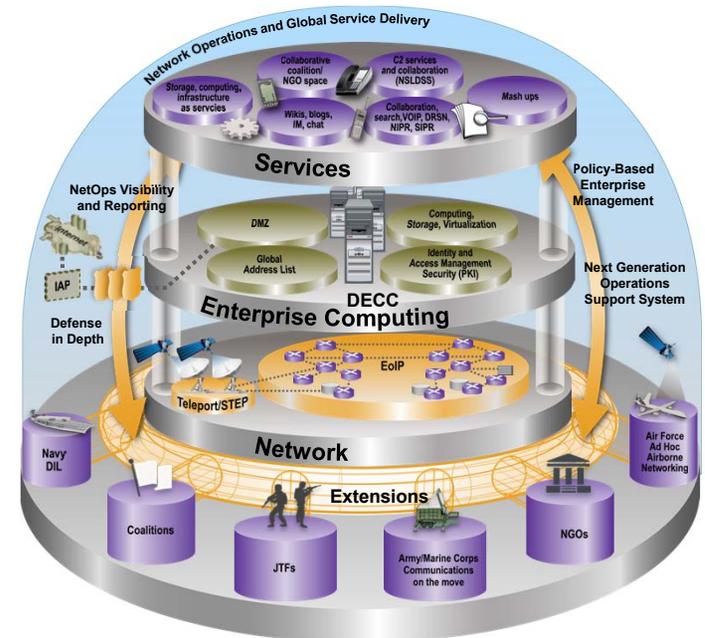


image credit DISA.mil



# Challenges of Software Acquisition

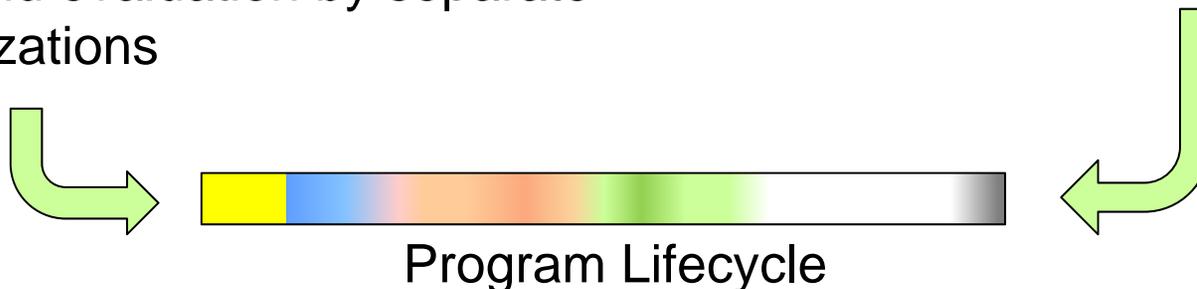


## Governmental statutory and regulatory requirements

- Upfront
  - Significant analysis and needs justification prior to the decision to fund a program
  - Program begins with in-depth solution analysis
- In-progress
  - Numerous decision points for moving to next phase
- Test and evaluation by separate organizations

## Broad spectrum of applications and environments

- Documented technical requirements developed by broad user community
- Limited demonstration opportunities for large, complex systems
- High test and deployment costs that may limit the number of increments and releases

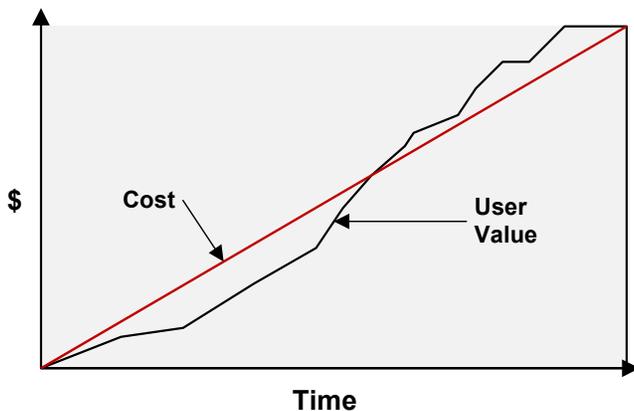
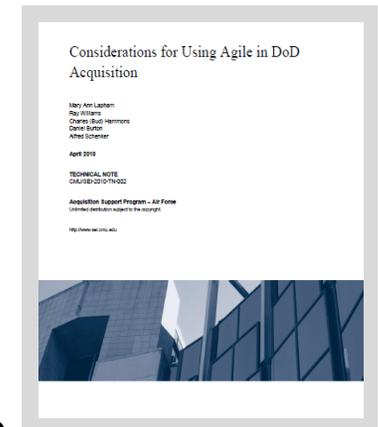
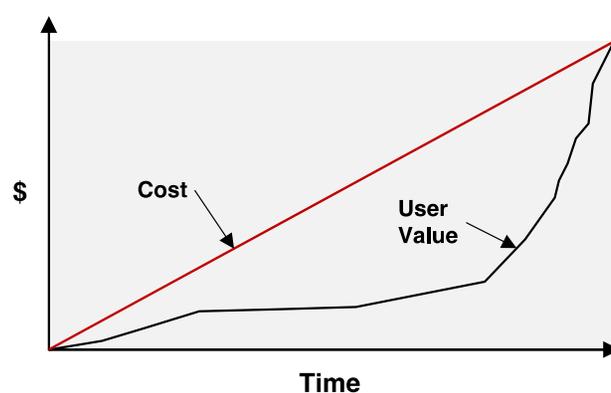




# Software Development Life Cycles

- **Comprehensive Planning**

- Models like Waterfall and Spiral
- Early baseline and lock down of design slows requirements change and cost growth.
- Capability appears at the end



- **Incremental Planning**

- Agile Models like Scrum and TDD
- Evolving plans can adapt to changes without causing rework, waste, and development cost growth
- Capability appears uniformly through the program

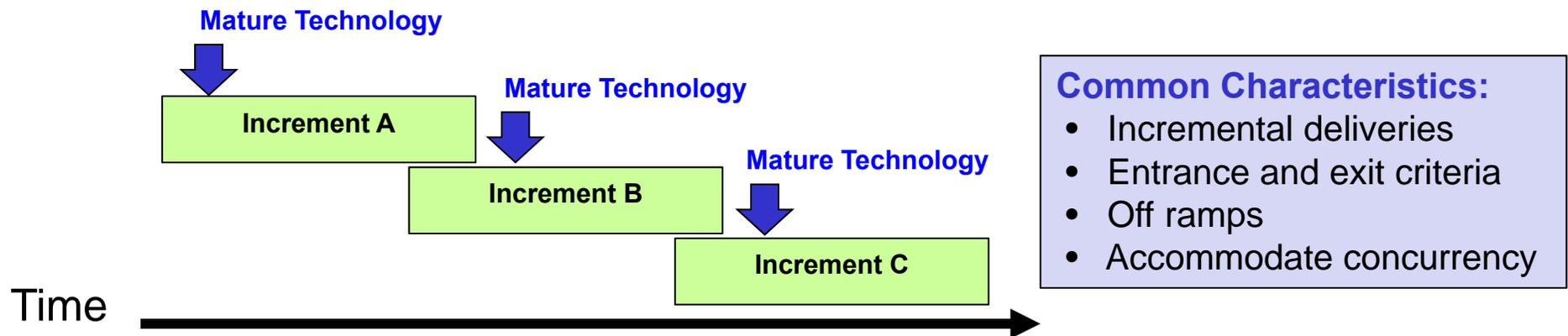
**Major DoD Programs will likely select different Life Cycle models for specific portions of the development effort**



# Evolutionary Acquisition— Preferred DoD Strategy

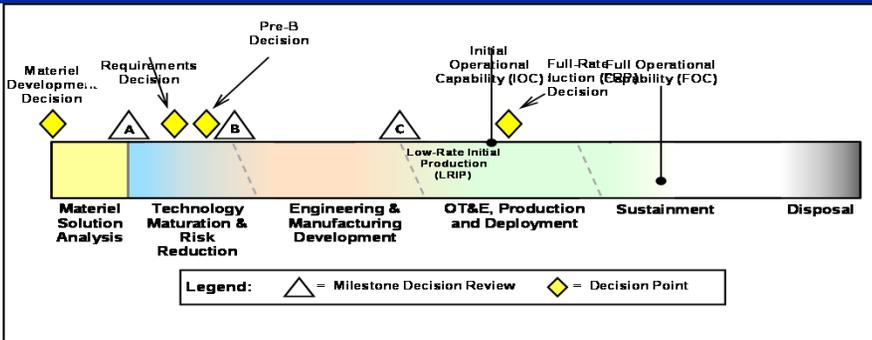


- **Rapid acquisition of mature technology**
- **Incremental acquisition of capabilities**
  - Recognizes the need for future capability improvements
  - Each increment is useful and supportable
  - Planned upgrades managed as separate increments

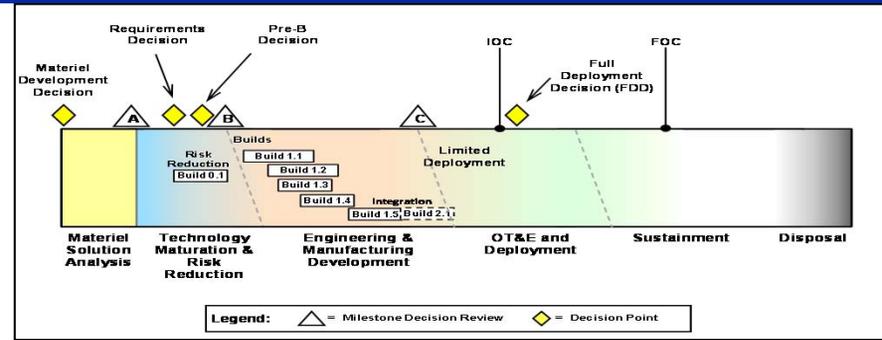




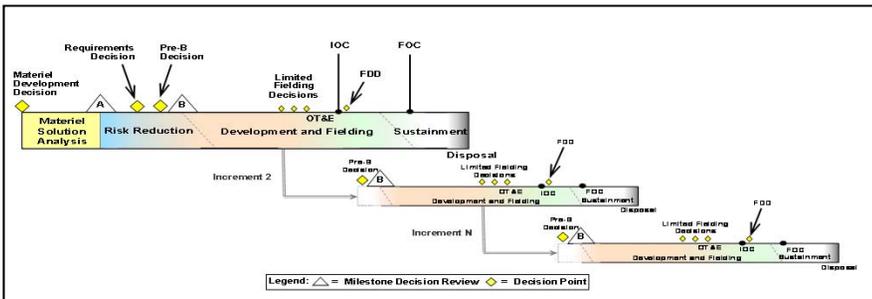
# Six Acquisition Models (Draft DoDI 5000.02)



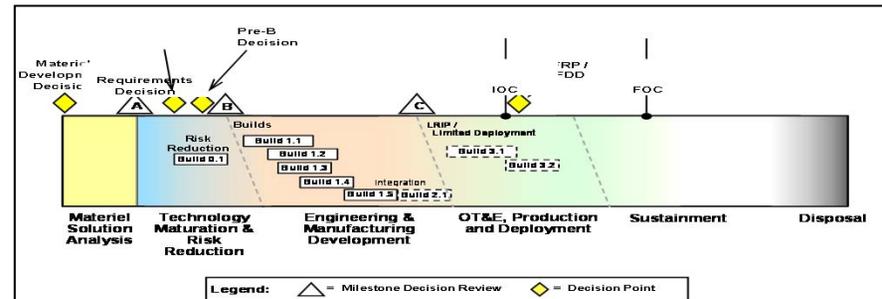
**Model 1: Hardware Intensive Program**



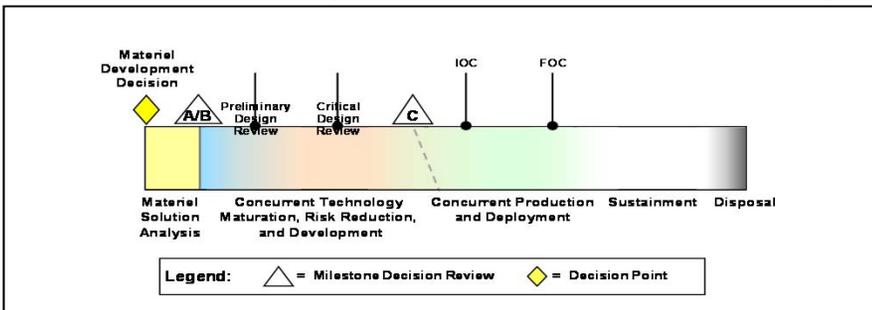
**Model 2: Defense Unique Software Intensive Program**



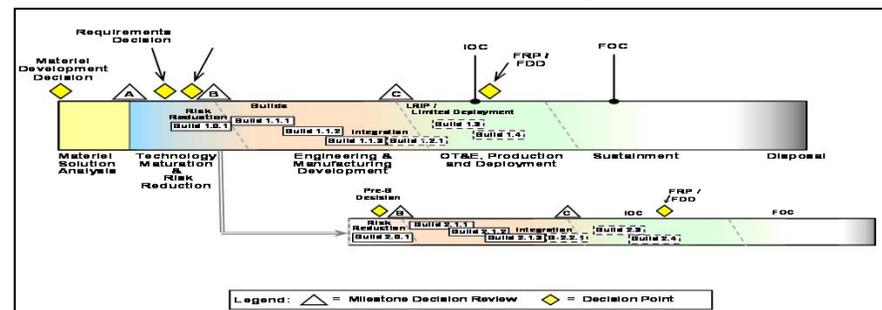
**Model 3: Incrementally Fielded Software Intensive Program**



**Model 4: Accelerated Acquisition Program**



**Hybrid Program A (Hardware Dominant)**



**Hybrid Program B (Software Dominant)**



# Draft Acquisition Models

- **The draft acquisition instruction (DoDI 5000.02) proposes six acquisition program models as a starting point for program-specific planning**
  - Model 1: Hardware Intensive Program
  - Model 2: Defense Unique Software Intensive Program
  - Model 3: Incrementally Fielded Software Intensive Program
  - Model 4: Accelerated Acquisition Program
  - Hybrid Program A (Hardware Dominant)
  - Hybrid Program B (Software Dominant)
- **All models recognize the critical role of software**

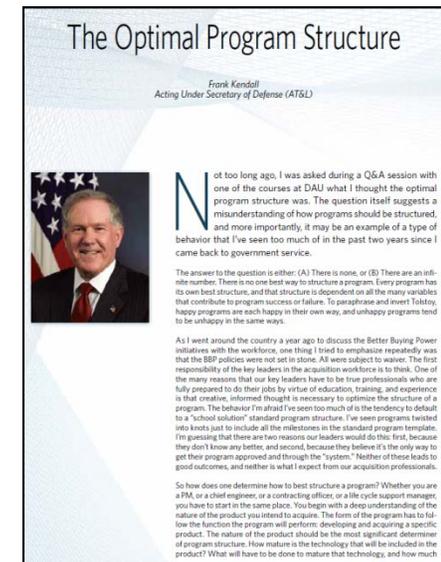
**Acquisition programs should use these models as a starting point in structuring a program to acquire a specific product**



# Guidance from the Defense Acquisition Executive (DAE)



- **First responsibility of key acquisition leaders is to *think***
  - “The behavior I’m afraid I’ve seen too much of is the tendency to default to a “school solution” standard program structure. I’ve seen programs twisted into knots just to include all the milestones in the standard program template.”
- **To determine the need, scope, and duration of the Technology Development (TD) Phase, ask:**
  - How mature is the technology (to develop/manufacture)?
  - How much risk is involved?
  - How complicated is the design? Is it novel?
  - How difficult is integration?
- **Also consider a range of other factors:**
  - How urgent is the product needed?
  - How much uncertainty in balancing cost and capability?
  - What are the customer’s performance priorities?
  - What resource constraints will affect program risk?
  - What are the right contractor incentives?



<http://www.acq.osd.mil/docs/Optimal%20Program%20Structure.pdf>



# Attributes of Agile Development

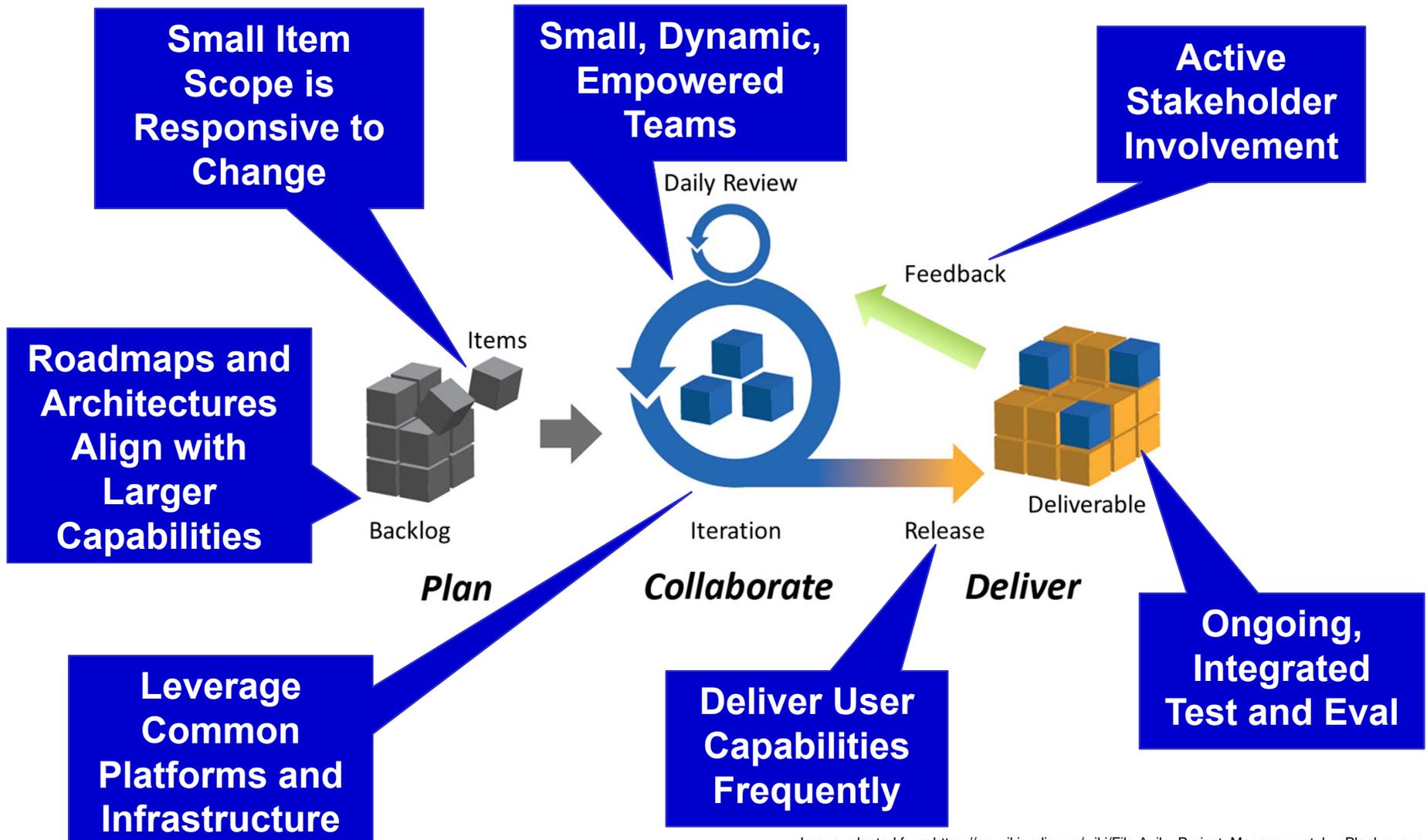


Image adapted from [https://en.wikipedia.org/wiki/File:Agile\\_Project\\_Management\\_by\\_Planbox.png](https://en.wikipedia.org/wiki/File:Agile_Project_Management_by_Planbox.png)



# What About Agile?

- **Can Agile address the complexity of DoD systems?**
  - Can we **decompose** tightly-coupled technical requirements into Agile user stories and controlled interfaces?
  - Can we **identify authoritative customers** - among many diverse stakeholders, including the Adversary - for feedback and iteration?
  - Can we **learn** from small, agile teams and scale to complex projects?
  - Can we support formal, **independent testing** over long test cycles?
  - Can we deliver **capabilities**?
- **Can Agile address regulatory challenges?**
  - Can we provide enough **“up-front” cost, schedule, and risk** analysis to satisfy DoD regulatory and statutory requirements?
  - Can we support the **persistent oversight and management** requirements of DoD acquisitions?
  - Can we mix **contractual negotiation** with customer collaboration?

**DoD Systems tend to be complex, with independently-developed, highly-coupled components**



# Criteria for Software Development Process Model Selection



**PMOs take on risk if processes don't fit their program.**

Environment & Situation	Agile Indicator
Is the team for the work unit collocated or geographically dispersed?	Collocated
How many teams design, code & test, and integrate the effort?	Fewer is Better
Are you willing to rapidly change the system based on customer feedback?	Feedback more effective
Can incremental deliveries provide useful capability?	Easier to execute
Is there a fixed set of requirements against a fixed schedule?	Less backlog management benefit
Are all requirements the same priority? ( <i>Closed Scope</i> )	Less prioritization benefit
Are change requests to requirements handled by the team or a separate organization?	Team management
Was the overall program schedule estimated assuming rolling-wave planning, or detailed bottom-up estimate?	Rolling Wave

**Project Specific Decision: Reject One-Size-Fits-All Models**



# Developmental Challenges

- **Assessment of Progress Design Reviews**
  - At what point can an Agile software effort have the same degree of confidence in success (based on incremental design and implementation) that a traditional CDR had from 100% design?
- **Scale and Schedule**
  - What techniques have proven effective in managing larger scale teams?
- **Infrastructure Availability**
  - Is the base software architecture stable? Is test infrastructure available?
- **Tightly Coupled Hardware/Timing Requirements**
  - How are hardware with imbedded software developments synchronized?
- **Product Owner sync with Field User (*Voice of the Customer*)**
  - How can the Program provide an effective stakeholder perspective, including users and adversaries as stakeholders?
- **Distributed Teams**
  - How can distributed teams be accommodated?
- **Personnel Skills**
  - How can workforce capacity be expanded and given the maturity of experience?

**Program management attention required to address challenges**



# Estimation Challenges

- **Limited maturity / experience in estimation**
- **Lack of adequate historical data (Context specific)**
- **Measurement of productivity to assess progress**
- **Stable historic metrics form the basis for future estimates**
  - Unstable teams and work scope can impact productivity / ability to effectively estimate
  - PMO cannot accurately estimate velocity until the team completes several sprints
  - If after many sprints cannot accurately predict velocity then it may indicate: X, Y and/or Z
- **Story points do not translate easily into other units of measure**
  - Equating story points to effort hours not supported by metrics, nullifying normalization
  - Parametric models gearing factors translate into SLOC, but with large  $\sigma$
- **Story points are not equal across teams**
  - Story point difficulty is negotiated within each scrum team with varying skills as part of each sprint
- **Using aggregate velocity is not necessarily an accurate predictor**
  - Aggregation of metrics across large scale teams; aggregate velocity to plan overall schedule and delivery
  - Requires normalizing metrics to support aggregation (“Story Points” as common measure)
  - The PMO and development teams will define story points differently
  - Single SCRUM team velocity can create a critical path risk not obvious in aggregate



# Technical Debt



- **The concept of Technical Debt doesn't fit well with traditional DoD development milestones**
  - Defined Initial Operational Capability dates
  - Independent Evaluation and Test against defined operational performance expectations
- **Limited ability to distinguish between Technical Debt and “Progressive Slow-Rolling Failure”**
- **Limited opportunity in a tightly-coupled system to adapt to changes in selective feature deliveries**
  - Software component capabilities can be critical to hardware development or system testing activities



# Combating Misperceptions

- **DoD Management is risk averse**
  - DoD applies significant effort to identifying, quantifying, and managing risk
- **New processes and methods don't mix with existing techniques**
  - This has never been the case. Projects routinely pilot new techniques on small tasks and expand their use gradually. One size does **not** fit all
- **Limited available workforce to conduct Agile software development**
  - Like any emerging technique, training and experience must be built into the workforce incrementally as the approaches are applied to more programs
- **Agile methods only apply in open-scope commercial settings**
  - DoD has been successful in applying Agile methods in closed-scope programs where all software requirements must be satisfied for the program to succeed
- **Agile software development means you don't have a long-term plan**
  - Software methods only apply after capabilities have been defined and allocated to software. Long-term project planning is done at a systems engineering level
- **Agile is difficult to contract for**
  - Contracts provide systems or capabilities, they define work at the systems engineering level
- **Metrics for agile technologies are hard**
  - Process-driven metrics for agile are different, but no harder to calculate or track
- **Agile means no documentation or artifacts**
  - On the contrary, Agile techniques provide design and implementation/test artifacts throughout the development process



# Conclusion

- DoD doesn't do manifestos
- DoD Leadership increasingly appreciates Agile software practices and terminology—many programs are using it
- DoD seeks practical approaches that mesh Agile with DoD's statutory, regulatory, and operational environment
- Upcoming revisions to DoDI 5000.02 should better support tailoring for adoption of Agile software development
- Agile processes also must be tailored to be effective within the closed-scope, schedule-constrained DoD environment
- DoD applauds any methodology that can improve software acquisition and systems engineering
- Bottom Line: DoD will apply the development approach best suited to the particular needs of each program

**“We are not in an easy business.” — Mr. Frank Kendall, USD(AT&L)**



# Systems Engineering: Critical to Defense Acquisition



***Innovation, Speed, Agility***  
<http://www.acq.osd.mil/se>