Paper ID# 044

iccrts@dodccrp.org

2012 ICCRTS
"Supporting Agile C2 with an Agile and Adaptive IT Ecosystem"

Topic 1: Concepts, Theory, and Policy

AUTHORS

Harvey Reed
The MITRE Corporation, M/S S110
202 Burlington Road
Bedford, MA 01730

Pat Benito
The MITRE Corporation, M/S LANG
202 Burlington Road
Bedford, MA 01730

Josh Collens
The MITRE Corporation, M/S C335
202 Burlington Road
Bedford, MA 01730

Fred Stein
The MITRE Corporation, M/S FTHD
MITRE/CTSF
Murphy Rd. & 53rd Street
Site 22009, Bldg 27
Fort Hood, TX 76544-0966

POINT OF CONTACT

Harvey Reed
The MITRE Corporation, M/S S110
202 Burlington Road
Bedford, MA 01730

©2012 The MITRE Corporation. ALL RIGHTS RESERVED.
Approved for Public Release: 12-1273. Distribution Unlimited.

“Supporting Agile C2 with an Agile and Adaptive IT Ecosystem”

Topic 1: Concepts, Theory, and Policy

ABSTRACT

Today, the U.S. military deploys in support of missions that require great operational adaptability.[1] A key element of mission deployment is information support via information technology (IT). Typically the IT is bundled with mission functionality, data, security, etc., into a single, large system baseline that is difficult to adapt in the field. This has forced the military to reconsider how to develop, deploy, and adapt IT capability in order to ensure mission success within fiscal constraints.

The authors propose that the military adopt an “Agile and Adaptive IT Ecosystem” (AAE) approach to system development. The AAE enables component providers to deliver components independently for future assembly into capabilities. Shared Agreements assure system developers that components will meet assembly requirements for certification and accreditation (C & A), authentication, authorization, data semantics, etc. The certification and vetting process is provided as part of market governance and can be partially automated. Once deployed, the component-based capabilities can be easily adapted to meet different mission requirements. The authors see immediate applicability of AAE to capabilities using component types such as mobile apps, pluggable user interfaces, widget frameworks, hosted information services, and shared security.

Introduction

Today, the U.S. military deploys in support of missions that can arise suddenly and change dramatically once underway. These missions require great operational adaptability, especially to make rapid adjustments based on continuous assessment.[2] When deploying, military units bring a range of people and their materiel into the theater, and must often work with other entities (coalition members, non-governmental organizations, etc.) to carry out their missions. A key item of material are information technology (IT) systems, ranging from handheld mobile devices to laptops, desktops, servers, associated networks, software, etc. Traditionally, that IT is packaged as a system baseline and functions as a single capability. The system baseline presumably includes everything needed for mission functionality, including data processing, security, computing hardware, etc.

Current methods for acquiring such system baselines have timelines averaging more than 7 years.

> *An analysis of 32 major automated information system acquisitions, conducted by the Office of the Assistant Secretary of Defense for Networks and Information Integration (OASD (NII)), calculated that the average time to deliver an initial program capability is 91 months...*[3]

[1] “DEFINING ADAPTIVE LEADERSHIP IN THE CONTEXT OF MISSION COMMAND,” thesis for Master of Military Art and Science, Maj. Jeremy Holmes, USAF, Fort Leavenworth, KS, Jan 2011 (p. 36) :: http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA547330

[2] Ibid.

[3] Defense Science Board, "Acquisition of Information Technology," March 2009, p.36 http://www.acq.osd.mil/dsb/reports/ADA498375.pdf

Because of these long timelines, the military has tended to acquire very large systems, which in turn demand large requirement sets. Since the world changes so fast, both in terms of regions and types of conflict and of advances in commercial technology, the military now faces a paradox whereby the time needed to deliver a complete large system exceeds the time over which requirements for that system remain valid.

Further, deploying units can no longer make long-range predictions regarding the full military utility of the system, since they cannot know in advance who their partners will be; the exact nature(s) of the conflict, what actions of agile adversaries will take, etc. At most, therefore, the military may have a general idea of how it wishes to use such a system, but often systems must be repurposed and integrated with other systems to achieve their full military utility.

The military has frequently responded by trying to accelerate acquisition of the system baselines by using "agile" techniques to shorten the development timelines. While this is certainly helpful, the greater challenge arises after deployment, when the system must adapt to and be integrated with other systems in a unique configuration. The resultant capability needed for that conflict, under that leadership, at that point in time, must be flexible enough to meet the demands of countering an agile adversary. Frequently, however, systems have failed to meet this need, and as a consequence people use email, chat, phone, etc., in ways to make up for this deficiency, striving to give the overall combined force the flexibility it needs, albeit with fragile human-in-the-loop integration. This illustrates the Law of Requisite Variety from cybernetics:[4]

> (paraphrase) *"…when the variety or complexity of the environment exceeds the capacity of a system (natural or artificial) the environment will dominate and ultimately destroy that system…"*

This paper describes an extension to classic systems engineering called Multi-Party Engineering (MPE), which enables the military to build and deploy the requisite variety of capabilities it needs by assembling smaller, reusable components, with the intentional ability to adapt afterwards by interchanging components. This gives the military a repeatable and persistent ability to *deploy agile and adaptive mission capabilities*. In this context, "agile" means fast initial deployment, "adaptive" means the ability to quickly adapt after deployment, and "mission capability" refers to the assembled IT capability. Applying MPE at scale with governance and feedback loops gives rise to emergent properties, grows an AAE.

Multi-Party Engineering

Current Challenges

Current methods deliver IT systems against a complete set of requirements over a long period of time as a singular tested and accredited capability. The obvious challenges are:

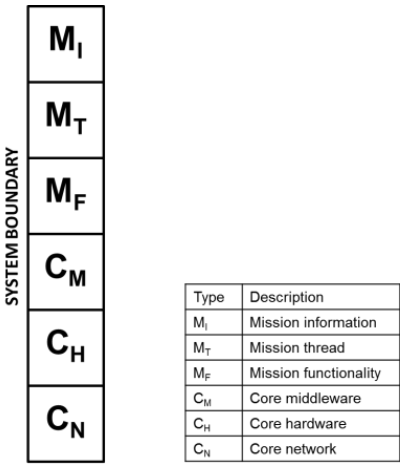
- Large systems have long delivery times, exceeding 7 years.
- Requirements may be out of date by system delivery.

[4] William Ross Ashby, 1956, An Introduction to Cybernetics, London: Chapman & Hall

- Systems are difficult to modify in the field.

Decouple into Components

The basic approach of MPE is to move away from large, single system baselines, as shown in Figure 1a. First, system developers must decouple the components that normally make up a baseline. The most important decoupling separates the mission-unique functionality from the infrastructure, which often could be shared (see Figure 1b). Decoupling provides the opportunity to package the functionality or infrastructure as components, as described below.



Type	Description
M_I	Mission information
M_T	Mission thread
M_F	Mission functionality
C_M	Core middleware
C_H	Core hardware
C_N	Core network

Figure 1a. Single System Baseline

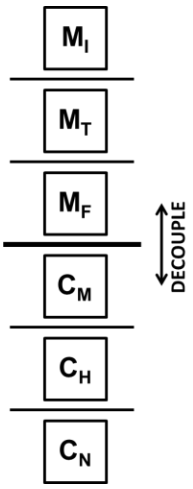


Figure 1b. Decoupled Baseline

The characteristics of mission-unique and shared infrastructure components are sufficiently different to make this decoupling obvious and intuitive:

- Mission unique components
  - High need to address specific mission needs
  - High need to tailor in the field
- Shared core infrastructure components
  - High need to share across missions
  - High need to consolidate

Multiple Component Providers

Once we decouple the parts of a baseline into components, we can consider enlisting a number of component providers (see Figure 2). Since these multiple providers may ultimately supply components for the same assembled capability (see below), we need to ensure that independently produced components can later be assembled. Shared Agreements serve as a means for independent component providers (and capability assemblers) to establish how to cooperate and provide components suitable for assembly. The Shared Agreements can cover numerous areas, including certification and accreditation (C&A), authorization, data semantics, etc.

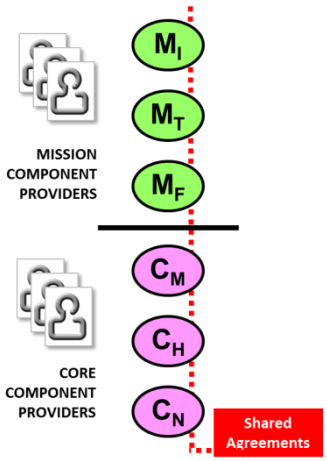


Figure 2. Multiple Component Providers

Among the various component providers we will see a distinction between mission component providers and core infrastructure component providers. These distinctions reflect whether the provider focuses on the end user and delivers a mission component, or on domain-agnostic reuse and efficiencies and delivers a core infrastructure component.

The key characteristics of independent component development are:

- Large number of mission-specific component providers
- Smaller number of core infrastructure component providers
- Reusable core components
- Shared Agreements that ensure components can be assembled into capabilities

Assemble Capabilities from Components

Decoupling components gives developers the opportunity to assemble capabilities independently (see Figure 3). This is one of the most important concepts underlying AAE, because this approach lets developers tailor capabilities to the needs of the mission. This concept also enables capability assembly to be tailored in the field by interchanging components, as well as receiving component upgrades. Once a capability is assembled and later tailored, direct feedback from the users guides component producers as they subsequently update/replace components, which can then be reintegrated into the capability. This feedback can be collected both real-time as the mission capability is used, as well as deliberately in interviews and/or feedback sessions. Overall, because the changes are small, the time required to gather feedback from the field and subsequently update the capabilities affected is shorter than that required for the single baseline method. Since components are certified to applicable shared agreements prior to assembly, time used for risk assessments, IA control apportionment, etc. is used prior to assembly. This helps to minimize the actual time of assembly and shifts risks to be addressed earlier in the overall process.

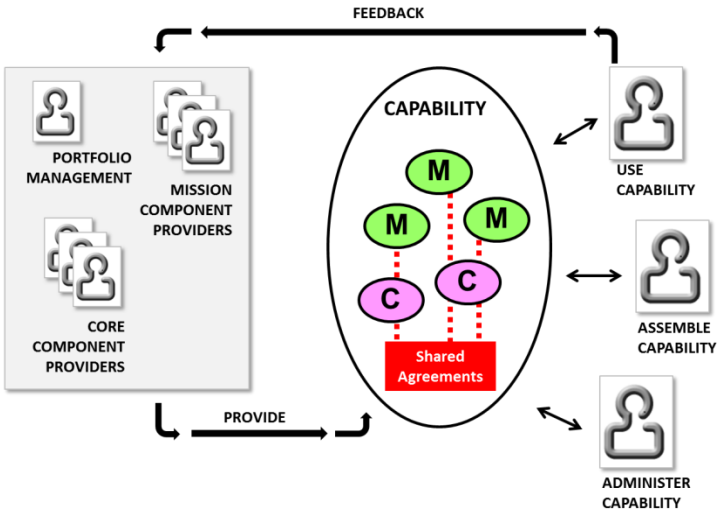


Figure 3. Assemble a Capability

The key characteristics of capability assembly are:

- Capabilities assembled using independently provided components
  - Enabled by Shared Agreements
- Components are a mix of core and mission-specific components
- Feedback from end user to component providers
- Assembled capability can be adapted and tailored in the field

Tenets

The tenets that characterize this approach are (a) provide small components, (b) offered in markets, (c) assembled into capabilities, and (d) feedback-driven refinement.

#1 Provide Small Components

Provide small components that are later assembled into capabilities.

The alternative to building complete large system baselines is to provide small components that can later be assembled into capabilities. This leads to a very beneficial decoupling of the early production of components and late assembly of capabilities. The components are delivered in a short timeframe, and are versionable.

#2 Certify Components to Shared Agreements

Certify components to Shared Agreements to assure future ability to assemble.

Shared Agreements constrain usage to ensure compatibility among components. Components can be certified to Shared Agreements that in turn address security, accreditation, testing, data semantics, etc. Component developers can also be certified, attesting to their ability to provide components that conform to various Shared Agreements.

#3 Offer Components in Markets

Certified components are offered in markets.

Shared Agreements capture assembly requirements (e.g., C&A, authorization, data, etc.), and components are certified as meeting the Shared Agreements. The components are then offered in markets. These markets are conceptually similar to markets we see in commercial smartphones, extended to accommodate many types of components. This decouples the providing components, and assembling capabilities in time, yet maintains integrity in the final assembly. The market governance assures Shared Agreements are versioned and current.

#4 Assemble Capabilities

Capabilities are assembled from components offered in markets.

Components are assembled into capabilities close to the time when the capability is initially needed, and the capability can subsequently be adapted and tailored. Shared Agreements shorten the time necessary for assembling certified components into capabilities. Assembling reusable components avoids costs, and the resources saved can be applied to tailor the capability to the needs of the deployed situation.

#5 Feedback Loops

End users give direct feedback to the markets and component producers.

Capabilities evolve as components are updated and new components are introduced on the basis of direct feedback from end users. The feedback is captured through end user engagement, made available to the component providers, and captured in markets that serve as repositories of knowledge regarding how components and Shared Agreements should be refined.

Components, Shared Agreements, and Governance

Both Shared Agreements and components require governance throughout their lifecycle in order to maintain stable relationships among components and ensure the efficacy of capabilities. The relationship among components, Shared Agreements, and governance is shown in Figure 4.

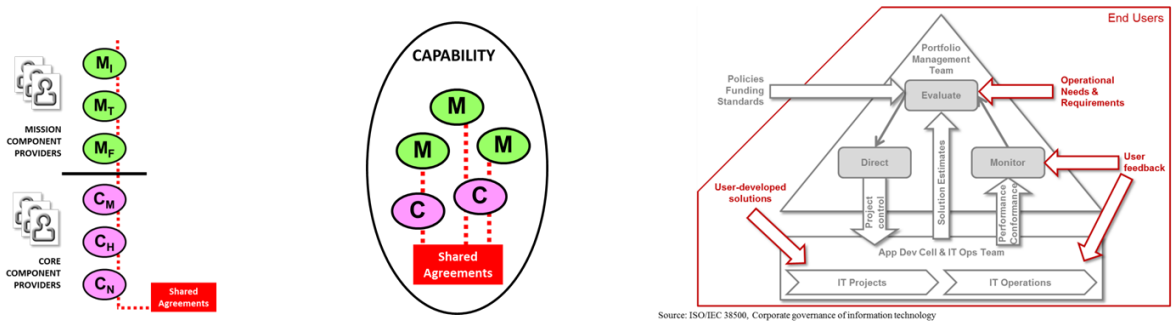


Figure 4. Components, Shared Agreements, and Governance

The Shared Agreements are used between components (vertically and horizontally), as a set of components in themselves, and also within governance structures. For example, when an entity is created to govern a market or a framework, the charter is typically agreed to and signed by the charter members. This charter represents a high-level Shared Agreement among the signing parties.

Between components Shared Agreements are used:

- Vertically between components, to assure users that mission components can properly use core infrastructure and vice versa: that the core infrastructure can support the mission component.
- Horizontally between components for two primary purposes: (1) mission-to-mission, as in "threads" of activity, and (2) core-to-core, as a means enabling infrastructure to work together—anything from compatibility between hosting and messaging to federation between shared security areas (see next bullet).[5]
- Capability-wide as a binder. A Shared Agreement for incremental accreditation, such as within a shared security area, offers one example. In this case a set of core components can be assembled and accredited to meet a certain subset of information assurance (IA) controls. At a future point in time, the assembly can incorporate a mission component that meets its own subset of IA controls and then inherits the IA controls of the core components. This enables accelerated accreditation because the bulk of the components is already accredited and need not be reaccredited.

Figure 5 shows the uses of Shared Agreements between components.

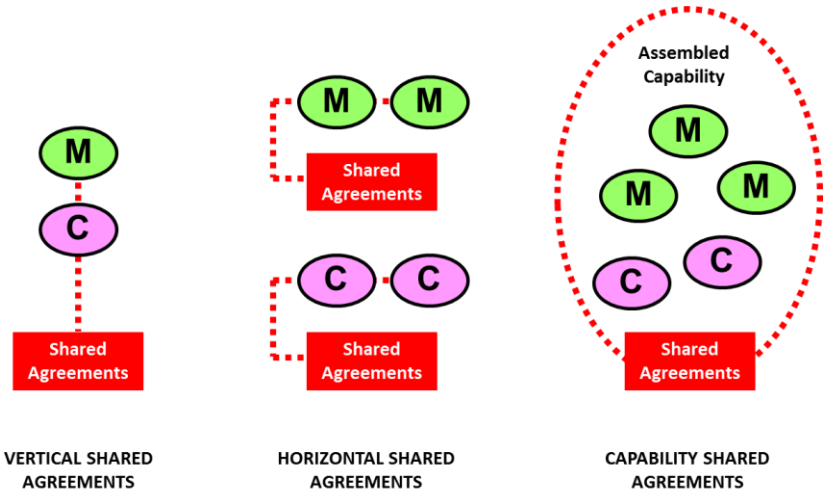


Figure 5. Shared Agreements among Components

The next section describes the governance functions applied to the parts of the ecosystem related to providing components, certifying components to meet Shared Agreements, and assembling capabilities from components. The governance functions, markets, components, capabilities and Shared Agreements taken together comprise what we describe as an AAE.

[5] Shared security area is a construct of the Joint C2 Objective Architecture.

Agile and Adaptive IT Ecosystem

Overview

The MPE described above includes the concepts and tenets underlying an AAE. The terms “agile” and “adaptive” are deliberately chosen to indicate that capabilities can be quickly assembled and deployed (agile) and can be changed rapidly once deployed (adaptive). As illustrated in the figure below, the AAE consists of independent component providers, capability assemblers, people governing market operations, portfolio managers, etc. In fact, many of the roles will be familiar through the present system baseline delivery mode, except that in AAE the components are finer grained and assembly is decoupled and occurs later in the cycle. The new roles of market governance and certification to Shared Agreements reflect the new activities inherent in coordinating independent and concurrent development activities.

The AAE consists of four main ecosystem governance areas:

1. Component providers (components can be mobile apps, widgets, plugins, services, core infrastructure, etc.)
2. Markets, component offerers, Shared Agreements, certifications
3. Capability assembly, end user engagement, feedback
4. Deployment of capability; copying and tailoring capabilities by interchanging certified components

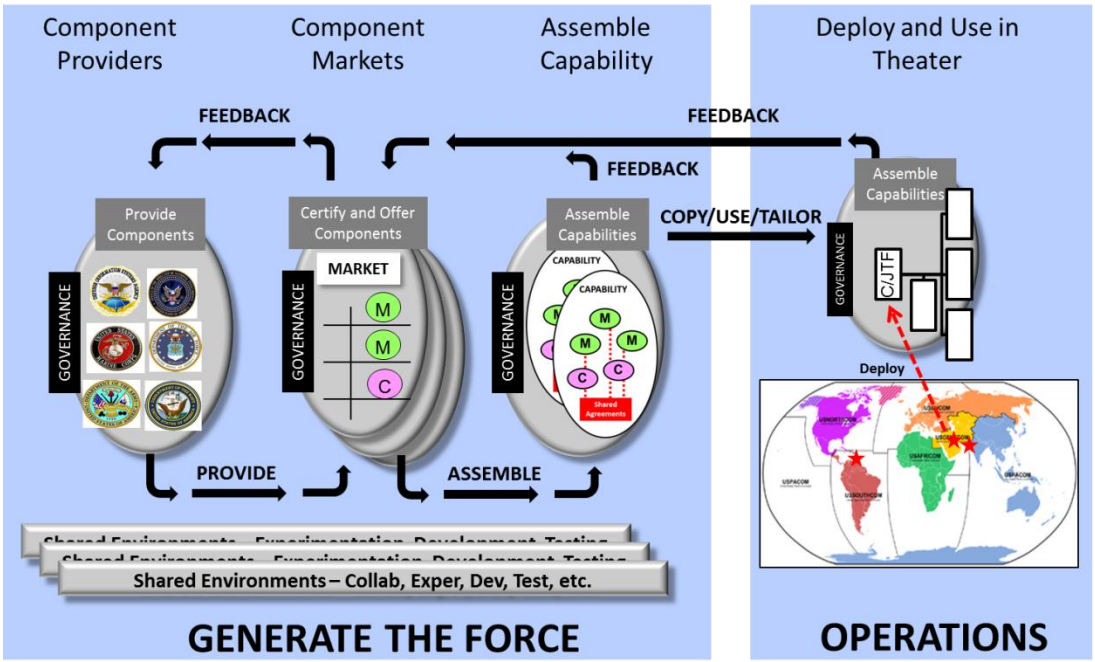


Figure 6. Agile and Adaptive IT Ecosystem

Figure 6 illustrates the AAE. The operations panel in the figure shows that a task force can copy and tailor a capability assembled by interchanging certified components after deployment. This meets the need for variety to match the variety in deployed missions. In addition, engineers can just as easily copy/use/tailor a capability assembly into a Defense Enterprise Computing Center (DECC) in the Continental United States (CONUS) for non-deployed operational needs. This sets the stage for providing components and assembling capabilities in the cloud and other enterprise means of delivering capability.

Because of the unique combination of stakeholders and funding typically associated with component providers, each component type has its own market and governance. For example, Ozone Widget Framework (OWF) and ozone widgets, mobile development frameworks and mobile apps, and Agile Client Framework and Agile Client plugins are all sets of components, and have different markets and governance. These markets can inform each other; in fact, all of these communities (ozone, mobile, Agile Client) already cross-share market knowledge and experience.

Since the AAE encompasses many independent component providers, marketers, and assemblers, their actions are inherently multi-party, as opposed to the single-party (single program office) actions typical of stovepipe development. Why is this important? Currently, when we build large system baselines, we try to avoid and minimize risk[6] inherent in the construction of the baseline by bundling as much as possible in one package, thus ensuring that everything in the baseline can interoperate. However, this approach does not address the substantial risk that the system baseline may not fit the operational context, or may fail to work with other system baselines in the field, including systems from other countries in the case of coalition operations.

MPE shifts component risk earlier in the process toward development, so that components available in a market are already certified to applicable shared agreements. The capability is then assembled from pre-certified components, thus shortening the time for final assembly including final testing and accreditation. MPE shifts risk associated with the fit of a capability to an operational context by assuring a degree of adaptability by interchanging components.

MPE does not try to minimize risk of any particular delivery. Rather MPE manages risk, and pulls risk in as early as possible. For example assembly-time requirements are captured in Shared Agreements. Component providers certify their components against such Shared Agreements, thus managing assembly-time risk before assembly-time.

Another aspect of risk management is to use Shared Agreements to express agreement in both a human- and a machine-readable portion. For example, a Shared Agreement for boundary protection of an enclave may include a PEP (Policy Enforcement Point). In this case, the agreement would spell out the authorization policy in plain English so that the various parties could review, concur with, and sign the agreement. The agreement can also have an associated machine-readable artifact such as an XACML file that is input into a PEP to configure its authorization rules. In this way the same binding Shared Agreement can align both the human and machine parties to the Shared Agreement.

These Shared Agreements form the basis of component certification, which is enforced by market processes. Thus, when the capabilities are assembled, the components offered in the markets are

[6] For example in DoD 4245.7 M “Transition from Development to Production”: https://acc.dau.mil/CommunityBrowser.aspx?id=25653

already certified to meet basic assembly requirements (e.g., C&A, authorization, data semantics, etc.) and developers can pay proportionally greater attention to tailoring the capability to meet the needs of the operation (requisite variety) by interchanging certified components.

Over time, engineers can shift their focus from trying to predict and create a perfect end state (a paradox of unreachable requirements) to creating Shared Agreements that capture the minimum assembly requirements but leave sufficient degrees of freedom to adapt and tailor the final assembly of capability by interchanging certified components. Assembly of the capabilities can likewise be staged so that the 80% of the capability that is generic can be delivered early as a template for multiple deployments, leaving the last 20% for tailoring to meet the needs of a particular deployment.

If AAE encourages early assembly of 80% of the complete capability templates in some cases, why not simply go back to building system baselines? The reason is that we want to reuse fine-grained components across many capabilities. Furthermore, we want to enable a variety of providers, both large and small, to supply the reusable components. This approach ensures greater reuse than prior models, where system developers received large baseline chunks to use as the foundation of a large system baseline. The AAE/MPE model offers variety on both the component provider side and the capability assembly side, thus effectively decoupling component development from capability assembly, mediated via markets.

When a capability (either relatively generic or highly tailored) is put into operation, further Shared Agreements may be needed to address operational characteristics. These are commonly referred to as service level agreements. Whether or not Shared Agreements exist only for capability assembly, or also for operations, the end user at any time can give direct feedback regarding the suitability and performance of the capability and its components. This feedback then drives refinement of constituent components, and subsequently of market component offerings and the resultant future versions of assembled capabilities.

The following is a summary list of principles for AAE as a whole:

Principles for Component Providers

The operations and governance of component providers follow these principles:

1. Components are as independent from each other as possible.
2. Components deliver either infrastructure or mission functionality, but not both.
3. Components are certified to meet the requirements of applicable Shared Agreements.
4. Certified components are offered in markets.
5. Components are refined based on feedback from end users.

Principles of Shared Agreements

The use and governance of Shared Agreements follow these principles:

1. Shared Agreements cover (a) assembly requirements (e.g., C&A, authorization, data semantics, etc.), (b) market governance, and (c) performance (functional, non-functional).
2. Shared Agreements have both human-readable and machine-readable parts as needed.
3. Shared Agreements evolve based on feedback.
4. Shared Agreements are managed throughout their lifecycle in their applicable markets.

Principles of Markets

The operations and governance of markets follow these principles:

1. Primarily markets organize components and Shared Agreements. Markets can also be used to organize generic templates of assembled capabilities for subsequent reuse and tailoring.
2. The primary commodities of the markets are (a) components, (b) Shared Agreements, (c) and commentary, including end user feedback and ratings.
3. The primary members of the markets are (a) component providers, (b) capability assemblers, (c) end users, and (d) market governance bodies.
4. Markets support the lifecycle of Shared Agreements that facilitate the assembly of capabilities.
5. Markets enforce certification of components to applicable Shared Agreements.
6. Exchange mechanisms used in the component market: (a) provider uploads component, (b) certify component, (c) download component, (d) promote component (reassign a component from one provider to another with a larger scope, e.g., move a local scope component to an enterprise scope component).
7. Shared Agreements in the markets: (a) Content is managed through the lifecycle of the Shared Agreement (both human and machine readable content), (b) Are referenced by applicable certification, and (c) Are referenced by audit trails of certification.

Principles of Capability Assemblers

The operations and governance of capability assemblers follow these principles:

1. Assemblies must comply with applicable Shared Agreements.
2. If assembly requires changes to Shared Agreements, those changes must occur prior to assembly (i.e., agile Shared Agreements).
3. End users must have mechanisms for giving direct feedback on the assembled capability.

Summary

The authors consider creation of an AAE an important response to current mission needs for variety, a shorter time to field, and fiscal constraints on deployed IT. MPE represents a practical extension to systems engineering that enables development of an AAE. We are currently working several use cases within the DoD to refine the MPE and AAE constructs.

Currently the bulk of activity centers on mobile apps and pluggable user interface frameworks. This is natural, since much of the initial demand for “requisite variety” will arise in the areas of situational awareness and personal information tools. The authors see further potential in more sophisticated use cases of sharing information in theater and sharing core infrastructure with clouds.

Acknowledgments

Lynne O’Riorden, MITRE lynne@mitre.org

Mark Bergman, MITRE mbergman@mitre.org

Disclaimer

This paper reflects the views and opinions of the authors and not necessarily the view or positions of The MITRE Corporation or the Department of Defense.