14th ICCRTS
"C2 and Agility"

"Implementing an Open Business Model and Open Architecture Approach to Enable Agile Technology Selection"

Paper ID 177

Topic #3: Information Sharing and Collaboration Processes and Behaviors

Author(s):
Megan Cramer, Jason Stack, and Rich Ernst

Point of Contact: Megan Cramer

Complete Address: 490 M St. SW #403W Washington DC 20024
Telephone: 301-509-1450
E-mail Address: mcramer90@gmail.com

Abstract

This paper presents a Technology Framework for enabling technology insertion within Command & Control (C2) software programs. The identified aspects of this construct include open architecture, technology, and open business perspectives. Each of the presented facets of this framework are explored in context with common technology maturity process for software. Systems engineering is endorsed as a mechanism to facilitate the development of a community of interest and the coherent alignment of technologies within current and future C2 architectures. Implementation of this framework supports a vision of migrating improved C2 capability to sensors towards an objective of ultimately enabling autonomous unmanned vehicles.

1.0 Introduction

Command and control (C2) software is required to efficiently manage systems and incoming situational awareness information. The C2 software for a mission area can be a critical component in the effort to achieve system of systems interoperability. Technology improvements to C2 software capability enable management of multiple systems and facilitate incoming situational awareness information in support of a common mission objective. Through adoption of a common Open Business Model involving scalable Open Architectures with identified Technology Insertions points, it is possible to create an effective government and industry partnership to support rapid capability improvements to C2 software systems. These improvements can be achieved through a well-defined formal structure of reusable services and common standards, which together support a decrease in the required integration time of new technologies and enable agility with respect to technology insertion. It is fundamental for both industry and government to participate in an inclusive business model and align to a common architecture in order to achieve an affordable and consistent foundation for the transition of future capability improvements.

Creating a flexible, cost effective framework for technology insertion continues to be a challenge for defense acquisition programs. At a large scale, government software providers are rapidly searching for ways to adopt new technologies to support legacy systems. These providers are looking to extend their old and new systems into a Service Oriented Architecture (SOA) framework and to adapt their business strategies to an open model environment supportive of new technologies from 3rd party developers. These emerging technologies targeted towards software have the potential to ultimately provide an improved capability to new or existing programs and marked increase in software automation and decrease in complexity of code. Harnessing these technologies and providing a business process that facilitates successful technology transition is essential in order to accelerate software capability and automation improvements and thus reduce the cost of development and decreasing the time to market. This vision has become a reality for many commercial software applications. FOSS (Free Open Source Software) is software that is freely licensed to allow the legal rights to the developer to view, change, and advance its intend development through the accessibility of its open source code. FOSS has gained a lot energy and wide spread audience for the potential benefits of small businesses, Department of Defense (DoD), and large industry partners. One must look no farther than “Living Labs” such as Google, Apple, IBM, or open source software communities such as GNU/Linux etc. to recognize the power of a collaborative research Community of Interest and a structured process for technology maturity (Katzy and Klein 2008). But how are these communities established, and how do these concepts effectively translate to the Department of Defense (DoD) environment?

In September 2006, the General Accountability Office (GAO) published a report to Congress emphasizing the need for improved practices related to the transition of technology to DoD programs (GAO 2006). The report highlights the need to ensure successful hand-off of technologies to acquisition programs, particularly given the tremendous amount of funding spent each year in the Science and Technology (S&T) arena ($13.2 billion in Fiscal Year 2006 on basic, applied, and advanced technology research). In considering this issue, the report compares DoD with commercial best practices for managing technology transition to mature products. One of the major findings from this report was that corroborating tools, such as technology transition agreements (TTAs), are employed to "solidify and document specific cost, schedule, and performance metrics labs need to meet for transition to occur. Relationship managers address transition issues within the labs and product line teams and across both communities. Meaningful metrics gauge project progress and process effectiveness" (GAO 2006). These tools must be included within a process that is owned by both the S&T and acquisition program partners. Mr. Robert Gold, the Deputy Under Secretary of Defense for Science and Technology (DUSD (S&T)) categorizes software technologies into five variants: (1) Unprecedented Functionality, (2) Off-The-Shelf Components, (3) Enabling Run-Time, (4) Aggregation of Components, and (5) Enabling Development (Gold 2009). Each of these flavors of technologies must be addressed and appropriately managed through the transition process. Additionally, transparent architectures and well defined business models for software programs must be appropriately open to address each of these types of technologies. Within this paper, a technology framework is presented to begin to address these challenges within a general construct. This framework contends that incorporating agility, modularity, and competition are integral to achieving the goals of accelerating C2 capability towards the objective of achieving eventually an autonomous unmanned systems capability and ultimately reducing costs for capability improvements. First steps in implementation of this framework for the U.S. Navy's Undersea Mine Warfare (MIW) Community of Interest are presented as an example.

2.0 Creating a Technology Framework

Before a technology framework can be developed for C2 software programs, it is necessary to consider the technology insertion objectives that are ultimately supported. A vision can be described by outlining three major objectives in support of technology insertion: (1) improved capability, (2) increased transition speed, and (3) reduced overall costs. These objectives are achieved by creating an environment of agility, competition, and interoperability. Figure 1 illustrates this vision in what can be described as an Objective Technology Framework.

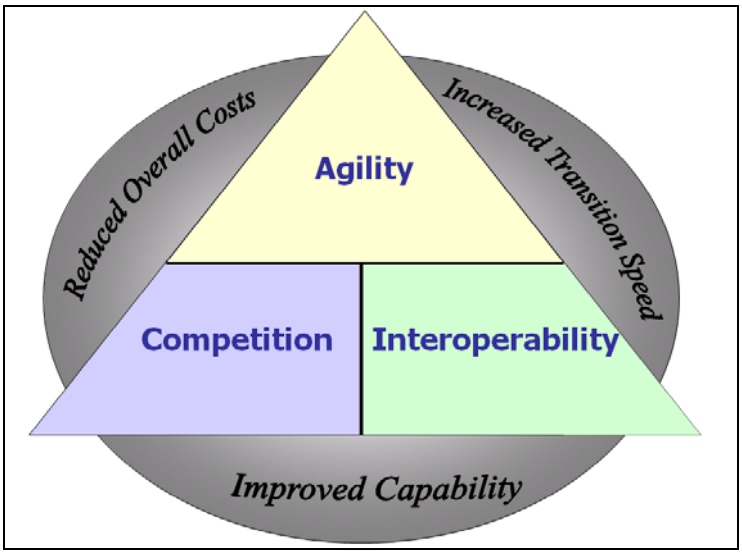

Figure 1: Objective Technology Framework
This graphic illustrates the objectives for efforts towards Open Business, Open Architecture, and Technology Insertion.

This vision for technology insertion for C2 software programs is directly supported by the implementation of both an open business model and an open architecture approach. In order to achieve an environment in which best of breed technologies are carefully considered and efficiently managed from an immature concept into an operational product, it is necessary to build both an open technical and open business foundation by which this migration can occur. Open business can be defined as the creation of a competitive environment by culturing a wide performer base with published product/domain knowledge to maximally leverage relevant technologies from other sectors. A competitive market environment can ultimately drive down costs to the government as prescribed by known economic principles of increasing competition in normal market conditions (Fetter 1918, 74). Open architecture from a software perspective describes the development of a technical architecture that embraces open standards, code reuse, and software modularity designed in such a way as to facilitate technology improvements. The Net-centric Enterprise Solutions for Interoperability (NESI) published by the DoD provides practical guidance on the adoption of these open approaches for software acquisition programs. (NESI 2008) Figure 2 provides a visual representation of these three aspects of a conceptual Technology Framework.

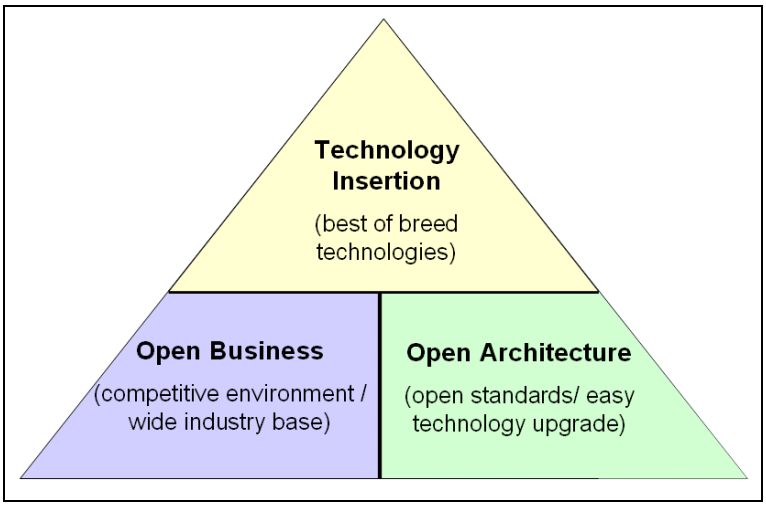

Figure 2: Conceptual Technology Framework
This graphic illustrates a conceptual framework for considering the alignment of Open Business, Open Architecture, and Technology Insertion.

It is important to note that the three aspects of the framework must be aligned to provide for a cohesive technology insertion approach for an acquisition program. Further efficiencies are gained through the development of a common strategies, plans, and processes between multiple acquisition programs within communities of interest. The need to align between multiple programs is further supported by the fact that technologies may often be relevant and targeted at several acquisition programs. Figure 3 illustrates the decomposition of this framework that illustrates the processes within the overall process within these three areas.

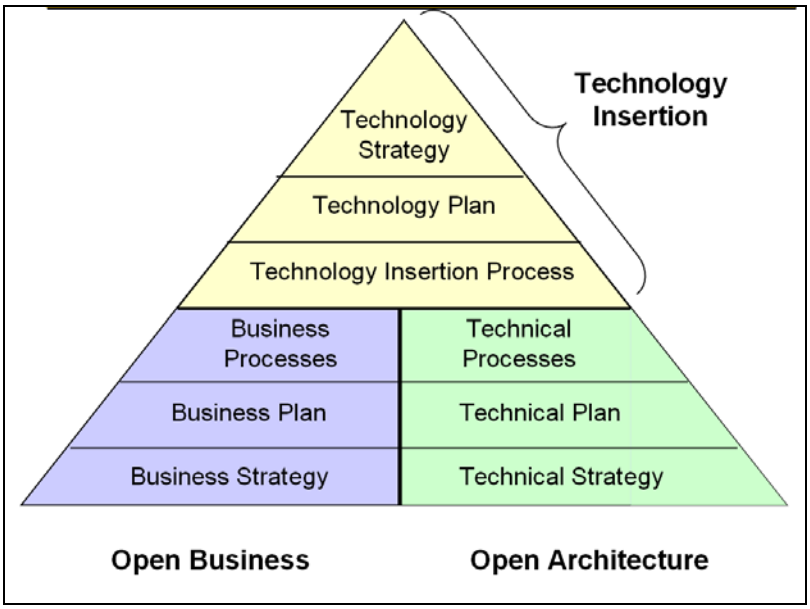

Figure 3: Process-Oriented View
This graphic illustrates a process-oriented view of the proposed technology framework for considering the alignment of Open Business, Open Architecture, and Technology Insertion.

Ideally, all three perspectives of Open Business, Open Architecture, and Technology Insertion within this view of the framework are aligned. It may be challenging to align these various aspects. It is useful to consider several examples of this process alignment. For example, the development of an appropriate data rights strategy for technology modules may be described in an acquisition program from an open business perspective. The software will then need to be developed in a manner consistent with the described data rights for each model (e.g., source code vs. executables). These modules must each be documented and reviewed from a verification & validation perspective as appropriate for that technology before introduction into the targeted acquisition program. Another example is that the inclusion of a license with a particularly technology module must be carefully considered in context with the program's business model. This business perspective should be analyzed in addition to technical and technology considerations.

In each of the three areas, a plan is included. This plan is the funded activities (execution year and over the Future Years Defense Program to support the achievement of measurable objectives supporting the overall vision. The strategy in each area describes this end vision and provides overall guidance towards supporting strategic goals such as reducing cost, increased lifecycle support, increasing competition, and accelerating delivered capability. For example, a technology strategy may describe a vision of delivering common technology modules to multiple mission areas in order to maximize return on investment of delivered technologies. A technology plan may explicitly describe the S&T programs to be funded with identified technologies targeted towards the specified mission area. Similarly, a technical strategy may outline a vision of maximizing software code reuse, improving interoperability through the use of common data standards, and leveraging common services within multiple software applications. The technical plan may describe the schedule for the development of logical data models to support the development of common standards within the community and the plan for community applications to migrate towards a SOA. Finally, a business strategy may describe an acquisition approach for increasing competition, reducing costs, and understanding software data rights. A business plan would describe the contractual mechanisms for creating such an environment for current programs of record. Although each aspect within the framework may involve a variety of technical, business, and technology strategies, the convergence of these perspectives occurs with the alignment of a commonly supported process for technology insertion.

The technology framework can be detailed into a component level view within each aspect. Figure 4 provides a detailed description of program- and community-related activities to support a realization of 'openness' in the areas of business and the technical architecture. These activities directly support agile and rapid technology insertion if aligned appropriately with the processes detailed in Figure 3.

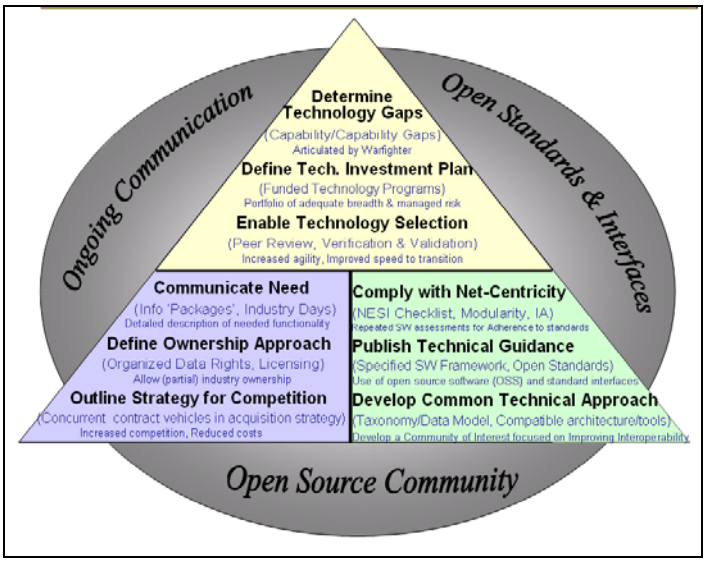

Figure 4: Component Level View

This graphic illustrates a component level of the proposed technology framework for considering the alignment of Open Business, Open Architecture, and Technology Insertion.

Within the following section, the activities highlighted to promote openness in the interest of technology insertion will be described. These activities involve both community and programmatic initiatives to institute and maintain. Examples from current initiatives within the MIW Community of Interest will be illustrated.

2.1 Creating an Open Business Model

To ensure an open business approach, a program should outline a strategy for enabling and increasing competition within the business environment and encouraging broad participation from potential solution providers. An acquisition strategy may include the use of concurrent contractual vehicles to present multiple opportunities for industry involvement and interaction. Multi-award contracts, Small Business Innovative Research (SBIR) projects, or Broad Agency Announcements are procurement mechanisms to facilitate competition between concurrent efforts. This open business approach involving broad participation from industry and academia can support an increase in competition for both the technologies developed as well as the software programs targeted for technology insertion.

Once an acquisition strategy is in place for an individual program or group of programs, the contractual vehicles themselves must define an ownership approach of the technologies developed. These activities include agreement of data rights up-front during the procurement request development process. To ensure business incentive towards innovation, the strategy must allow for at least partial industry ownership of both software and technical data rights. Intellectual property is maintained by the contractor or developing organization. A licensing approach must be considered and should be consistent with the program lifecycle goals and cost constraints. A software license may grant recipients rights to modify and redistribute the software, which would otherwise be prohibited by copyright law. A license should be associated with software components such that it does not impact the overall license strategy of the software as a component of

the larger system. Use of Open Source Software (OSS) and Commercial Software should be supported and a process established in the contract to facilitate government review of licenses per the Naval Open Architecture Contracts Guidebook (IWS 2007). Definition of the ownership approach up-front will not only minimize future confusion but emphasize the importance of ongoing communication involving business interests. Open source software development practices should minimize redundant software development and enabling a more agile development of systems.

Finally a program must engage in an on-going dialogue with the S&T community regarding program technology needs and automation goals. Detailed information describing future functionality needs should be routinely communicated to the performer base to enable appropriate technology investment plans to be developed and aligned to support acquisition programs. Development of information 'packages' with relevant technical, process, tactical, syntax, semantic, and software/data dependencies will provide context by which technologies may be developed for a specific functional gap. Further steps include the development of online research communities and software development kits, etc. to facilitate the communication of not only technology needs, but also the necessary supporting information to enable relevant and appropriate solutions.

The Advanced Processing Build (APB) process developed by the U.S. Navy's submarine community is an example of a defense program intended to improve acoustic performance through rapid introduction of new technology. The APB model is an 18 to 24 month ongoing process of developing new technologies to satisfy identified fleet operational needs. The APB process, illustrated in Figure 5, encompasses advanced algorithms developed in academia, small business innovative research, laboratory contribution and large defense contractor engineering solutions. The APB program differs from typical defense programs in that it evolves existing software year to year rather than designing a completely new system. Key differences between traditional system engineering and the APB process are that the APB does not have a single integrator and utilizes a system of systems approach to examining and incorporating new technology. The result of this methodology is a streamlined process for identifying best-of-breed technologies to fulfill warfighting operational requirements.

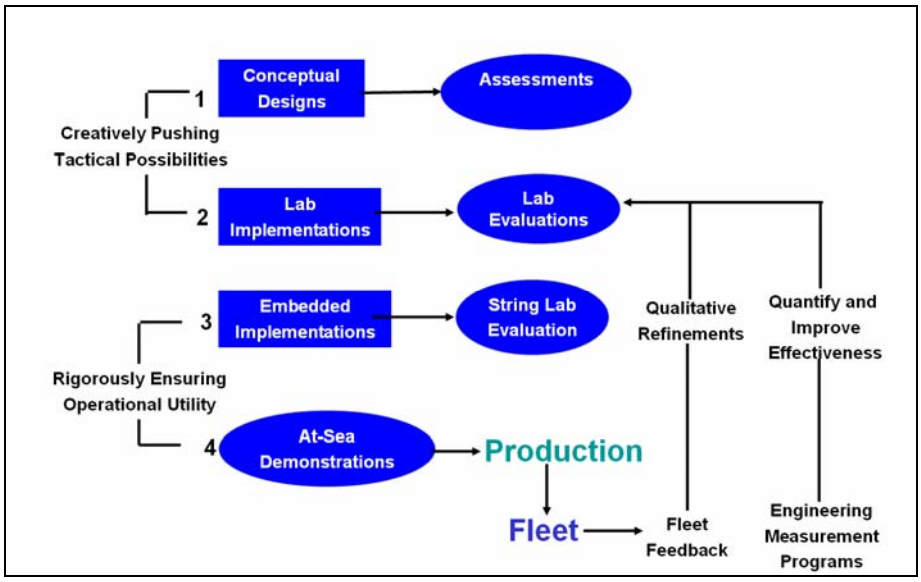

Figure 5: Submarine APB Process *(Figure courtesy PEO Submarines)*

This figure shows the basic outline of the APB process from the start through to Fleet feedback.

Examining effective open business models like APB and those processes in industry is useful to consider various approaches that have succeeded in the commercial arena. Figure 6 provides a spectrum of potential business models with open involvement from their respective communities of interest. Before considering the differences between these models, it is interesting to note some areas of similarity between these successful models. Each model is enabled by a research community of interest that is supported by a web-based collaborative environment. Providing similar web-based collaborative tools (through secure means where necessary) would potentially be useful by research communities in DoD. Additionally in each of these models, such as DoD Techipedia, there is bi-directional communication between the research community and the targeted community for technology insertion. Outside DoD, communication flows more from the targeted community than from the research community. In DoD however, it seems this communication is mainly from the S&T community as acquisition offices manage the crises of the minute and plan immediate efforts towards upcoming milestones. Identified funding and management personnel specifically for communication of program needs and functional information is a potential remedy to the current situation in many programs.

Figure 6: Potential Open Business Models

This graphic depicts a spectrum of potential Open Business Models that have been effective in transitioning technology within software in a commercial environment.

With regard to the potential business models described in Figure 6, these can be analyzed according to the amount of structure. OSS communities such as GNU provide examples of a business model that is unstructured (www.gnu.org 2009). Source code is posted directly to online websites and this code is then improved as determined by individual community members. Benefits of such a model being applied to DoD include full transparency of software, facilitation of peer review, maximum flexibility for innovation, and community ownership. Application of such a model to a DoD environment does have some drawbacks, and these include commercial incentive to participate through the loss of intellectual property, lack of control over development direction, and lack of overall architecture design with significant implication on the ability to implement information assurance controls.

Technical interrelationships emerge when considering the spectrum of potential business models presented in Figure 6. As a system designer allows the developer more flexibility the probability the developer will be interoperable with the designer is reduced proportionately. The inverse is also true, as the designer reduces the flexibly to the developer by restraining them to a specific design they also increase the odds of interoperability to the overall design. This dynamic must be managed with continued systems engineering at a system of systems level.

A model that illustrates a very structured approach in which Software Development Kits (SDKs) are provided and developed as applications that 'snap' into specific interfaces is the development of iPhone applications enabled by Apple Inc. The benefits of this approach are control of the overall hierarchical architecture as well as tighter quality controls. Industry also has incentive to participate as the ability to organize license strategies and data rights is relatively simpler. A negative aspect of this more structured approach is the possible loss of innovation by the limited number of interfaces by which to insert the technology. This leads to limited interoperability and scalability between software segments.

GoogleLabs provides an intermediate approach to structure between these two approaches. The GoogleLabs website provides an excellent example of an online technology maturity process (www.google.labs.com 2009). The community has the ability to experiment with 'beta', or prototype, applications, plug-ins, and services thus enabling ability to leverage upon future technologies. The process facilitates communication and incremental development of maturing technologies. Transparency applies to the process rather than the technology itself. This is a mechanism for protection of intellectual property but may limit government reuse in the interest of follow-on competition. Innovation is encouraged through the potential for multiple identified interfaces through the use of multiple technology options (applications, plug-ins, and services). Structure of the overall software architecture design is carefully managed through adherence to the identified process. This model stops short of providing the maximum re-use and government ownership afforded by the unstructured model, but provides industry incentive to participate and a structured approach to ensuring overall software configuration management.

There are many potential open business models that have been shown to be employed successfully in the commercial arena and these models should be examined for applicability within the DoD environment. Regardless of the model chosen for particular communities of interest, the commonalities between these models with the importance of a research community and a web-based collaborative tool to facilitate communication should be closely considered for DoD application.

2.2 Creating an Open Architecture Approach

With a supporting Business Model, creating an Open Architecture necessitates both programmatic and community approaches in developing a sustainable technical architecture. A software program must not only be open architecture within itself, but must be compatible with co-resident and remote applications and services to ensure alignment and interoperability. From a community of interest perspective, a taxonomy of community information must be developed and associated semantics agreed upon by the community. The taxonomy then becomes a roadmap for development of scalable logical data models, which will ultimately drive physical data representation and common schemas for information formatting and exchange. Compatible software architecture components must be determined to reduce interoperability issues between applications. A technical approach to a common architecture must be developed such that future application can later be plugged together. Physical implementation questions such as core visualization approaches, the enterprise service bus (ESB), database architectures, etc. must be answered to minimize redundancy and preserve an overall sensible physical architecture.

Once a software team has identified its technical design, the program must publish a software architecture document to describe in detail the technical guidance that will enable external software and/or services development by third-party developers. This available technical guidance information will ultimately reduce integration costs by minimizing interoperability issues and redundant functionality. The guidance document describing the software architecture should include specific software frameworks, standards, and development environments.

Finally, throughout the development process the program should institute a repeated assessment of the software to ensure adherence to applicable standards and processes for code reuse. Tools such as the NESI checklist monitor software modularity and consideration for information assurance controls early in software development. (NESI 2008) Repeated assessments to ensure compliance and compatibility throughout the development process encourages continued improvement towards open, modular, and reusable approaches. Specifically, the use of common tools with a common architectural design pattern is very useful with reducing cost and overall integration time.

Figure 7 provides an example of how the development of an open data service by a community can directly support research community interests.

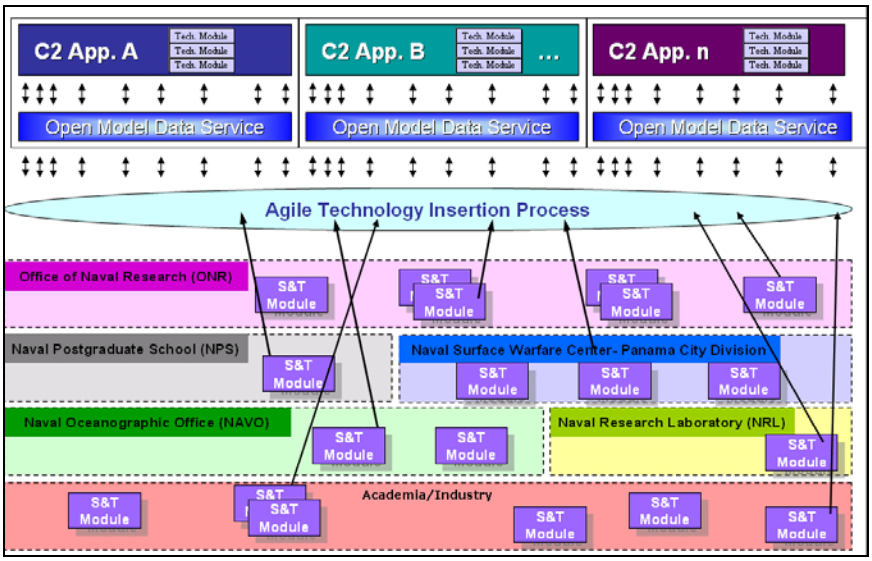

Figure 7: Data Service Support for Mine Warfare Research Community

This graphic depicts an open business model vision of multiple organizations developing capabilities targeted for transition to C2 software programs for the mine warfare mission area. An open data service enables research through enabling access to relevant tactical data and reduced costs through eliminating redundancy in development of multiple data interfaces by each organization.

The figure illustrates multiple organizations and S&T initiatives in the mine warfare community that are vying to bring new technologies into C2 software programs. A well-formed published data service provides a standard interface by which research organizations may subscribe to data in support of supplying algorithmic requirements. In support of the mine warfare open business model approach for software, multiple data services (e.g., mine contacts, bathymetry, plans) were developed and published to provide a mechanism by which organizations conducting S&T activities could access necessary tactical information. This mine contact service has already proved effective in reducing costs as each researcher does not have to build individual data interfaces directly to the C2 applications.

There are many benefits to the use of open architecture as a mechanism for streamlining technology insertion, reducing integration costs, and supporting an agile technology selection process. Development of a common technical approach between application, publishing technical guidance, and compliance with net-centricity are

specific approaches to ensuring programmatic and community open architecture implementation.

2.3 Supporting Agile Technology Selection and Insertion

The establishment of open business and open architecture approaches within software acquisition programs directly enables agile technology selection and insertion. With the knowledge of an available business process supporting an open business model and an accessible open architecture, a technology strategy can be developed beginning with the determination of technology gaps as articulated by the warfighter. Given these identified needed capabilities, a technology investment plan is formulated, executed, and results in the form of funded technology programs by S&T organizations. This investment plan is a maintained portfolio of technologies or potential technologies with adequate breadth of solutions and managed risk of technologies at various states of maturity. The process for transition includes both peer review and verification activities both of which could be facilitated with participants from the previously described research community. Providing mechanisms for coordination, communication, and collaboration will ultimately result in improved speed to transition, increased review of technologies, and flexibility in the offering of potential solutions. The goals are to increase agility in technology selection and to more rapidly field technology improvements to operational systems.

The concept for technology insertion into software programs is illustrated in Figure 8. This graphic depicts a process to enable this agile technology selection in a process that is managed from a capability fielding and configuration perspective. Borrowing from commercial examples, the idea of concurrent baselines each with software in various stages of technology maturity is at the heart of this concept.

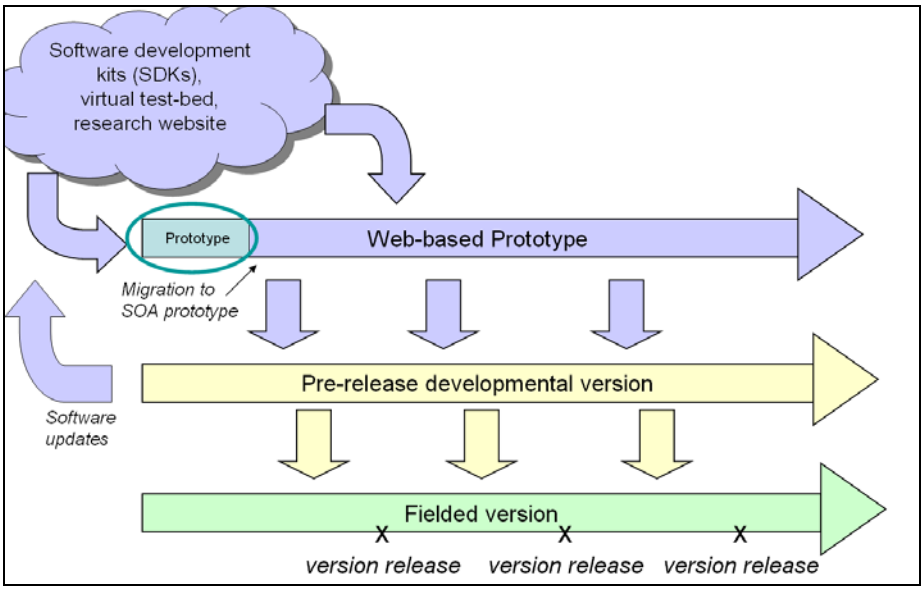

Figure 8: Conceptual Technology Insertion Process for Software

This figure depicts concurrent software development efforts to enable a streamlined transition of capabilities based on technical maturity. New capabilities are brought from a conceptual level through to operational status through a managed software configuration process.

As shown in the figure, a research community (as shown in the cloud) is enabled by tools and information for communication and collaboration. These tools may include software development kits, a web-based collaborative website, and an available test-bed. As the research community matures new technologies, they are rolled into an available prototype version of the acquisition program software. This prototype is particularly useful for incorporating customer feedback into the process before a capability is fielded. The mine warfare community has established a fleet trial process to present prototype versions of C2 software to fleet users for consideration prior to the capability being fielded within the operational build of the software. This trial process with the prototype also becomes a useful mechanism for flowing feedback from the customer and about the current warfighter challenges back to the research community. Once a technology has been approved and funded for inclusion in the software baseline, it is integrated formally within a pre-release version of the software until it is appropriately tested as part of a configuration baseline that is released and fielded to the operational environment.

3.0 Importance of Systems Engineering

A key enabler of an agile technology selection and insertion process is the establishment of a collaborative research community. The ability for a community to provide contextual information with regard to C2 capability gaps is dependent on the organization, common understanding, and availability of that information within the community of interest. Figure 9 provides a graphic of different views of contextual information that together provide support for development of appropriate solutions to support technology and automation shortfalls for C2 software programs. These views are systems view (programs of record), process view (tactics, techniques, and procedures by the warfighter), functional view (purpose-based perspective that is critical for ensuring modular and reusable software), data view (consistent with community of interest data standards), assessment view (feedback on acquisition program performance), documentation (architecture information captured and databased), business view (acquisition strategies and business models such as the ones previously discussed), and of course systems (programs of record).

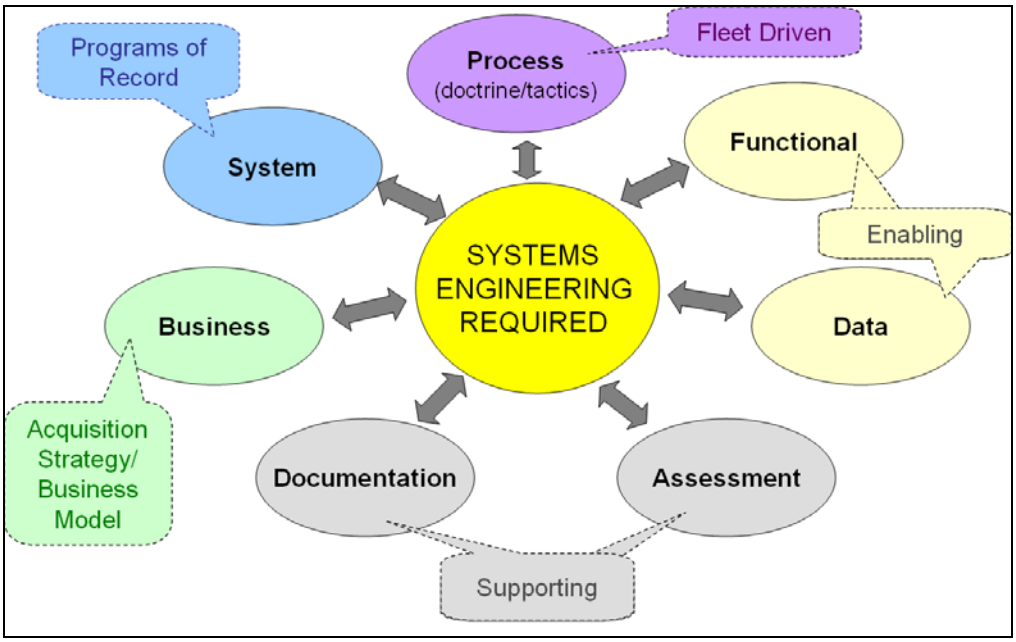

Figure 9: Importance of Systems Engineering

This figure depicts multiple views of contextual information relevant to the architecture that together provide support for development of appropriate solutions to support technology and automation shortfalls for C2 software programs

These different views must be aligned through systems engineering and must be consistent with the architecture views relevant to the open architecture perspective previously discussed. For example, a taxonomy of data (and more detailed logical data models and XML schemas) must be consistent with the data view, which corresponds to semantic definitions provided from a tactical perspective in the process view. Another example is the requirement for systems to support warfighter processes. Inability to fulfill this requirement will result in an identified capability gap in the technology insertion process previously discussed. A community must therefore develop, collect, and agree on this information for it to be made available beyond the community in support of a broader research community. The ability to gain agreement and capture this information is by no means an easy task for a community of interest. It is essential that this level of alignment and organization within the community be achieved in order to realize the vision of agile software technology selection.

4.0 Conclusion

A vision for technology insertion within software acquisition programs can be described by outlining three major objectives: (1) improved capability, (2) increased transition speed, and (3) reduced overall costs. To achieve this vision, this paper presents a technology framework to illustrate the reliance of agile technology insertion on the creation of competition within an open business model and on the achievement of interoperability in implementing an open architecture approach for C2 software programs. Open business can be defined as the creation of a competitive environment by culturing a wide performer base with published product/domain knowledge to maximally leverage relevant technologies from other sectors. Open architecture from a software

perspective describes the development of a technical architecture that embraces open standards, code reuse, and software modularity designed in such a way as to facilitate technology improvements. The consideration of commercial examples and current initiatives within the mine warfare community of interest are examined. The important role of systems engineering in aligning and capturing community of interest information is highlighted. This contextual information directly supports the establishment of collaborative research communities and is a key enabler in ultimately facilitating selection of appropriate solutions through an agile technology insertion process.

Reference List

Fetter, Frank. 1918. *Economic Principles*. New York, NY: The Century Company.

Gold, Robert. 2009. Software Technologies in Technology Readiness Assessments for DoD Acquisition Programs. Software Tech News 11(4). https://www.softwaretechnews.com/stn_view.php?stn_id=48&article_id=120

Google.Labs.com. Google's Technology Playground. 2009. http://labs.google.com/ Accessed 22 March 2009.

GNU Operating System. 2009. http://www.gnu.org/ Accessed 22 March 2009.

Katzy, Bernhard and Stefan Klein. 2008. Special Issue on Living Labs: Editorial Introduction. *The Electronic Journal for Virtual Organizations and Networks* 10: 2-6. http://www.ejov.org/apps/pub.asp?Q=2993&T=eJOV%20Issues

U.S. Department of Defense. PEO-IWS 7. 2007. *Naval Open Architecture Contract Guidebook Version 1.1*. Washington, D.C. https://acc.dau.mil/CommunityBrowser.aspx?id=105662

U.S. Department of Defense. 2008. *Net-Centric Enterprise Solutions for Interoperability V2.2.0*. San Diego, CA. http://nesipublic.spawar.navy.mil/

U.S. Government Accountability Office. 2006. *Stronger Practices Needed to Improve DoD Technology Transition Processes*. Washington, D.C. GAO-06-883. www.gao.gov/cgi-bin/getrpt?GAO-06-883