The George Washington University

Comparative Analysis of Pathways to Changeability

Aditya Singh (asingh25@gwu.edu, 678-656-6919)

Zoe Szajnfarber (zszajnfa@gwu.edu, 202-994-7153)

April 6, 2023

Naval Post Graduate School

Acquisition Research Program

Abstract

Through an examination of three cases of change in the U-2 platform, this paper compares three pathways to changeability: form changes, operational changes, and cyber changes. Each pathway can lead to change in similar properties of a system but have varying levels of performance and time to implement. For each pathway, we describe the design mechanisms necessary to implement change in that pathway. We analyze the tradeoff between performance or extent of change and agility or speed of change and find that form changes offer the highest degree of changeability but take the longest time to implement. Operational changes offer the least degree of changeability but are far quicker to implement. Cyber changes lie in between these two pathways. Understanding the design choices needed and the underlying tradeoff of each pathway can enable decision makers to better select a pathway to change when the need arises. This comparative analysis is especially useful since literature has thus far examined each of these pathways in isolation, not as different paths to the same goal.

Author Bios

ADITYA SINGH is a PhD student at The George Washington University, studying Engineering Management in the Department of Engineering Management and Systems Engineering, where he previously received his B.S. in Systems Engineering and Economics Currently, he is a Doctoral Fellow on National Science Foundation Research Traineeship Program: “Co-Design of Trustworthy AI Systems”.

ZOE SZAJNFARBER is a Professor and Chair of Engineering Management and Systems Engineering at The George Washington University. Her research seeks to understand the fundamental dynamics of innovation in technology-intensive governmental organization, as a basis for decision-making. She received her bachelor’s degree in Engineering Science from the University of Toronto. Szajnfarber conducted her graduate work at the Massachusetts Institute of Technology, earning dual masters’ degrees in Aeronautics & Astronautics and Technology Policy and a doctorate in Engineering Systems.

1.0 Introduction

Complex engineered systems (CES), such as aircraft and ships, often entail protracted design phases and lengthy lifecycles. This gap between system conceptualization and system retirement introduces a great deal of uncertainty over the system lifecycle as new needs arise as the gap grows. To guard against this inherent uncertainty, CES are often required to be changeable, meaning that they can change in response a change in the operating environment. Design for changeability literature has typically focused on mechanisms that make changing the physical form of the system easier. Previous work identified that system users can change *how* the system is used to maintain value in a changing operating environment without risky and expensive form changes. Software design literature has also examined how software can be designed to more easily incorporate changes after the initial design phase. These three pathways to changeability, form, operational, and cyber, have not been connected in the design for changeability literature and have not been compared to each other in terms of agility and performance. This paper shows that form, operational, and cyber changes can be leveraged to achieve similar types of change and compares the speed of implementation and performance each type using three cases of change in the U-2 platform.

2.0 Literature Review

Design for changeability literature is concerned with how systems maintain value in the face of changing operating environments. Changeability is an umbrella term that captures many strategies for how systems can change in response to a change in operating conditions (Fricke & Schulz, 2005). Four key strategies are adaptability, flexibility, scalability, and modifiability. Adaptable systems initiate change through internal change agents, while flexible systems initiate change throughout external change agents. Automatic software updates are an example of an internal change agent, while a technician modifying a system is considered an external change agent. Scalability refers to change the level of some system parameter, like bandwidth. Modifiability refers to the ability to move system parameters from agent to agent, such as using a dongle to connect a new subsystem to an existing computer (Ross et al., 2008). There are several more strategies, collectively referred to as the -ilities (de Weck et al., 2012) (Beesemyer, 2012) (Ross & Rhodes, 2019), but they are not covered for brevity and relevancy.

These strategies need specific mechanisms to be implemented. Changeability mechanisms are specific design choices that enable these strategies to be carried out. One of the most popular mechanisms is modularity, which involves a one-to-one mapping of function and module. Modules are loosely couples with each other and the rest of the system, but modules themselves are often comprised of tightly coupled components (Baldwin & Clark, 2000). Modules rely on common interfaces to be easily swapped in and swapped out. While modularity continues to be a popular changeability mechanism in industry, modularity often comes at the cost of design optimization and performance of the system (Hölttä et al., 2005).

Real options are another popular mechanism for changeability. Stemming from finance, real options in engineering are contract tools that give system buyers the right but not the obligation to implement a change in the future (de Neufville, 2003). A classic example of real options is a parking garage where system buyers might include an option to add additional floors to the structure at some point in the future. This requires an upfront investment in the option, to make

the foundations stronger to accommodate the potential change, and can be executed in the future if the buyers decide there is enough demand to justify the execution cost (de Neufville et al., 2006). Real options are rarely executed perfectly as technical, logistic, and organizational delays can create a gap between when the option is executed and when the option is fully implemented. The value of real options degrades as implementation delays arise (Sapol & Szajnfarber, 2020).

Margin, the excess of a system property beyond its required level, is another significant change mechanism. Margin has been tied mostly to evolvability, the transfer of common system traits from generation to generation (Allen et al., 2016) (Tackett et al., 2014). Building in margin for design is related to adding in safety margin, which is a common practice in many fields like civil engineering (Eckert & Isaksson, 2017). Previous work identified margin as a key enabler of modularity and flexibility as well (Singh & Szajnfarber, 2022), but modern systems face many design requirements that require physically optimized design. Physically optimized design means an elimination of margin, which can limit the amount of form changes a system can accommodate.

Literature identified that changing *how* the system is used can enable changeability (Mekdeci et al., 2015). These operational changes can even provide systems with new capabilities, thus avoiding risky, expensive, and/or time consuming changes to the form of the system (Singh & Szajnfarber, 2022). Operational changes are often generated by system users, who are considered to be agents of changeability within the system (Cox, 2017). While changing how a system is operated has been shown to be a mechanism of changeability, it is still limited by the form of the system. Users can only do so much with the system that they have. This creates a need to change the system without extensive form changes, which can be accomplished through cyber changes.

While changeability literature has largely focused on form changes, there have been some considerations of changeability through software. In their seminal paper, Fricke and Schulz describe how automatic software updates could be a mechanism for achieving adaptability (Fricke & Schulz, 2005). Since then, others have created and discussed changeability as it relates to software, primarily relating to software quality (Brown et al., 2022). For example, researchers have discussed the maintainability of a software system, which is further subdivided into the repairability and modifiability of said system (Chen et al., 2018). Modifiability, the ability of a system to accommodate a change, is most closely related to how systems can add capabilities (Bachmann et al., 2007). Reducing coupling, a strategy to create modular systems, is also a key technique in software design. Delaying binding time, when a flexible software feature becomes fixed (Krisper & Kreiner, 2016), and increasing cohesion within modules to reduce overall module complexity are also key strategies within modifiability. Specific design mechanisms for each of these sub-strategies have been discussed in literature (Bachmann et al., 2007).

Many studies in software changeability are focused on the repair of these systems. Even those that are focused on adding or enhancing capabilities often cite software evolution and the pace of change in software as a key motivation for why change is needed. This is due, in part, to most of these studies focusing on software systems and not cyber-physical systems specifically. Helen Gill coined the term cyber-physical systems, defining them as “systems with integrated computational and physical capabilities that can interact with humans … and expand the capabilities [of] the physical world through computation, communication, and control” (Baheti & Gill, 2011, p. 161). Cyber-physical systems are deployed in very different environments than software only systems and face different change motivators. Cyber-physical systems have been identified as key platforms for changeability since the incorporation of several types of systems

increases the trade space of changes that can be implemented and increases the number of experts due to the variety of systems found in cyber-physical systems today (Colombo, 2016).

Nevertheless, changeability literature falls short on analyzing how software design can enable new capabilities in cyber-physical systems. While software design literature has detailed mechanisms to achieve modifiability and other changeability mechanisms, changeability literature has failed to appropriately appreciate cyber pathways to change, especially in terms of adding or enhancing new capabilities in the field. Complex engineered cyber-physical systems, like many of today's air and spacecraft, face less pressure to change from market forces and technological evolution, and face more pressure from changing operating environments over long lifecycles. Responding to these changes by adding and enhancing capabilities using software will be an important capability for complex engineered cyber-physical system operators and needs further investigation into how it can be enabled and how it is implemented.

3.0 Methods

To investigate cyber pathways for changeability and compare them to other pathways of change, we examine three instances of change where U-2 targeting, imaging, and sensing capabilities were updated. Aircraft are a prime example of cyber-physical systems as modern jets are becoming more cyber reliant, while still relying on their physical form to accomplish their tasks. Cyber components of aircraft are often used to interface with physical components and can enable certain capabilities. Fricke and Schulz characterized systems that have a well-defined core function but highly variable secondary functions, have long lifecycles but rapid technology integration requirements, operate in a system of systems environment, and have "high deployment and maintenance costs as those that are best suited for changeable architecture (Fricke & Schulz, 2005, p. 7). Older military aircraft fit these criteria and have substantial publicly available information that is not available for commercial or modern military aircraft.

One instance is of a form change implemented through the Agile Pod system, another instance is of an operational change implemented during Desert Storm, and the final instance is of a software change implemented recently. Table 1 presents a summary of the three cases of change in the U-2 platform analyzed in this paper. We analyzed what necessitated the change, the extent of the change implemented, and the time required to implement the change. Through this analysis, we find that there is a tradeoff between the extent of the change that is implemented and speed at which it can be implemented. Form changes are the most extensive, providing the highest degree of change but requiring the most amount of time to implement, while operational changes offer the lowest degree of change but require the least amount of time to change. Cyber changes lie in between form and operational changes on the extent and speed tradeoff axis. There is a delay in developing software, but implementation can be instantaneous if over-the-air updates are enabled. Each case is discussed further in this section. For each pathway of change, we also discuss the upfront design requirements to implement, if any.

Table 1: U-2 Results Table

	Need for Change	Extent of Change	Speed of Change
AgilePod (Form)	Need to integrate multiple sensors & cameras onto U-2 and quickly swap	Modular pod created that can swap different sensors in and out;	Useable prototype delivered in 18 months

	equipment for different missions	leverages common mechanical and electrical interfaces	
H-Cam (Operational)	Request for higher resolution on intelligence images from H-cam on U-2; H-cam operates at an angle to capture maximum amount of ground	Camera angle changed to straight down for higher resolution; new flight routes developed	Changes implemented in a matter of days after camera angle was mechanically changed and new flight routes were planned
Kubernetes (Cyber)	Need to account for new types of targets not planned for originally	Improved automatic targeting algorithm developed and installed	Software created in weeks, implemented instantly over-the-air

4.0 Differences in Implementing Different Pathways

U-2 Agile Pod (Form)

Intelligence, surveillance, and reconnaissance (ISR) is a core requirement of the United States Air Force (USAF) which is comprised of several different missions, each with their own equipment needs. This variety of mission and associated equipment creates a difficult logistical environment since not all aircraft are able to accommodate each piece of equipment. The Air Force realized the need to enable aircraft to swap in and swap out ISR equipment easily and quickly (Trevithick, 2018a). To meet the challenge, USAF developed a pod made up of several compartments ranging in size that can be reconfigured to accommodate a variety of ISR equipment. Several iterations of the pod, known as the AgilePod, have been created to match different requirements, primarily focused on size to accommodate what the aircraft can hold and what the aircraft needs for each mission. AgilePod uses common interfaces and creates a single physical and electrical interface that can be mounted on aircraft pylons (Nine et al., 2019) (Shirey et al., 2017).

Recently, the Air Force awarded KEYW a contract to develop an AgilePod to accommodate a variety of ISR equipment. The pod was delivered in prototype form to the Air Force within 18 months (Cogliano, 2015) (Alia-Novobilski, 2016). (Trevithick, 2018a). A recent iteration of an agile pod was installed on an aircraft in a hangar at Wright-Patterson Air Force Base in Ohio for testing in a matter of weeks, showing how rapidly these AgilePods can enable new capabilities (Alia-Novobilski, 2018). Once installed, swapping ISR equipment becomes tantamount to swapping out ordnance on a fighter jet. The AgilePod was installed on U-2s, and contract vehicles have been created to develop new sensors for the AgilePod family of sensors (Trevithick, 2018b).

While AgilePod is one of the most agile and flexible systems in the Air Force acquisition pipline, design and development took over a year and a fit test took weeks. The test was conducted in the United States, but if AgilePod needed to deploy to an international field, additional logistic constraints and delays would arise. AgilePod provides a useful baseline for implementing rapidly needed capabilities even though it is not a fully fielded system on the U-2. Modular systems that provide new capabilities have been shipped to the field without full testing

in the past, as noted in previous with the GPU-5/A sent to Desert Storm (Singh & Szajnfarber, 2022) (Smith, 2021).

U-2 Camera Positioning (Operational)

Desert Storm was the largest U-2 operation in U.S. military history, providing key intelligence and targeting information to allied forces. U-2s operating in Desert Storm and Desert Shield carried a variety of sensors and cameras, including the High Resolution 329 camera (H-cam). The H-cam's normal concept of operations is to place the camera at an angle in the gyrostabilized compartment to provide the maximum amount of coverage. Those in the field relying on the data needed greater resolution for the H-cam data to be useful. To accomplish this, "Lieutenant Colonels Lafferty and Spencer … decided to revise the H-camera's procedures" by shooting the camera straight down instead of at a coverage maximizing angle (Cross II, 2014, p. 41). This required technicians to reposition the camera in the compartment and required planners to redevelop the flight paths to accommodate for the loss of aerial photography coverage area. Through these operational changes, U-2 operators and intelligence officers were able to greatly improve image quality, over what the camera was advertised as offering, without having to acquire a new camera system (Cross II, 2014).

U-2 Targeting Software (Cyber)

A U-2 recently received an over-the-air update that improved the aircraft's automatic targeting system (Trevithick, 2020). The update is the first time that military software was updated on an aircraft while the aircraft was in flight (Insinna, 2020). In-flight updates were made possible by Kubernetes, an open-source software containerization system developed by Google and donated to the Cloud Native Computing Foundation. Kubernetes enables developers to automate a large degree of testing and development through software modularization and reuse (Trevithick, 2020). To use Kubernetes, system functions need to be decoupled so that developers can quickly swap software modules without affecting the entire system. This type software module to system function mapping is the same modularity strategy employed by designers of physical systems. While software modules are mapped to system functions and loosely coupled with each other, the modules are tightly coupled within themselves as each Kubernetes module has all dependencies and libraries within the module. Being able to quickly swap modules in software and hardware are very similar in their design requirements, but they require extremely different logistical considerations to implement (Insinna, 2020). Kubernetes was installed on the U-2's existing computers without the need for new electronics or avionics. Following the U-2 over-the-air update, the Kubernetes system was installed on F-16s in 45 days showing how rapidly software open architecture can installed on a system (Chaillan, 2019). While complete function to module mapping was not completed in this 45-day span, F-16s were subsequently able to receive an over-the-air update that provided new electronic warfare data files. The update was initiated from an Air Force base hundreds of miles away from where the F-16 was flying when it received the update (F-16 System Program Office, 2021).

5.0 Analysis

U-2s have shown that changeability can be achieved through form, operational, and cyber pathways of change. Each case covered related to some aspect of ISR for the same system, showing that each pathway could be used in the same context. The extent of changeability for each pathway of change was quite different. Form changes require the most extensive logistical requirements, with physical systems needing to be procured, produced, and shipped for installation. The AgilePod that was recently developed required 18 months to get to the prototype phase, showing how time-consuming physical system development can be, even when the product is based on an existing product framework. Even if the physical equipment needed is already produced, shipping and installation can introduce heavy tolls on logistical capacity, with large potential for severe delays (Sapol & Szajnfarber, 2020). Equipment like the AgilePod represent a best-case situation for form changes, as it leverages existing common interfaces and is designed to be extremely modular. Being able to add or swap equipment creates the largest trade space of possible changes, creating a tradeoff between agility and the extent of changeability. This tradeoff is reversed with operational changes.

Operational changes can be implemented very easily with system users changing how their system is used without extensive changes to the form of the system. Conceptualizing the change and training enough to ensure that new concepts of operation are effective require a highly variable amount of time but are generally much faster than implementing a new form change, as seen in the U-2 H-cam change and as noted in previous case study work (Singh & Szajnfarber, 2022). Adding to the agility of operational changes is that they do not require upfront design considerations. Systems need to be designed to easily accommodate future form changes but do not require such design considerations. While extremely agile, operational changes are restricted in degree of change they can create in a system. Operational changes that aim to improve capabilities or gain new capabilities in the field are generally initiated when system users face an urgent need and do not have time to wait on a form change to be initiated and implemented. This means that system users have to work with the system they have, not the system they want. While the H-cam changed showed how changing how a system is used can increase its capabilities even beyond what system designers were willing to advertise, operational changes are still constrained by the physical limitations of their physical systems.

Cyber changes are a newer pathway of change that seem to be in the middle of form and operational changes on the agility and extent of changeability continuum. Similar to form, software requires system design choices that enable future changes to be easily implemented. The case discussed in this paper leveraged software modularity is a key strategy for changeability, requiring many of the same design considerations as physical modularity including loose coupling between modules and tight coupling within modules. A key difference however is when systems can be made modular. Decoupling physical components is far more difficult than decoupling software systems and this can be done after the fact, as the U-2 and F-16 software components were not explicitly designed with software modularity in mind. Physical systems are more defined by their initial design than software systems, representing a timeline shift in when these design choices need to be made.

In terms of agility of implementation, software has been created and installed on platforms like the F-22 through over-the-air updates in a matter of just 60 days (Hadley, 2022). When software is already created and need to be transmitted, over-the-air updates enabled almost instantaneous implementation. This is not to say that software implementation does not require extensive logistical capabilities to be in place. The F-16 update used a satellite to implement, and other platforms hoping to take advantage of the agility of over-the-air updates need to have

reliable access to transition and enough computing power available to implement. If these capabilities are in place, cyber changes can be implemented rapidly, but if they are not, cyber changes would require systems to return to a central depot, making them more akin to slower form changes.

In terms of extent of changeability, the limits of cyber change for CPS are being pushed constantly. Recently, Tesla and Mercedes released optional software updates that could be implemented over-the-air that would make their cars faster, meaning that software changes can impact the maximum physical performance of a system (Gerken, 2022). Making cars faster and improving targeting software are both examples of improving a system's existing capabilities, but the F-16 change represented "the first time a fighter aircraft has received a software update and gained new capability all while in flight" (F-16 System Program Office, 2021). As software is increasingly used to control and manipulate physical system properties, the trade space of changes that can be implemented through software only changes will increase. Additionally, software updates may have unique interactions with other forms of change. For example, battery optimization software might be able to create margin in power supply where there was none before, enabling physical changes that take advantage of newly created margin.

6.0 Conclusion

By examining three cases of change, we showed that form, operational, and software changes enhanced capabilities in the same mission area for the same platform. We additionally examined the design choices required to implement each change, the speed at which the change was implemented, and the extent of the change. Through this examination, we reveal the tradeoff between agility and extent of change. Form changes are least agile but have the highest extent of changeability and require upfront design considerations. Operational changes are the most agile but have the least extent of changeability as system users must work within the constraints of the system. These changes do not require upfront design choices. Cyber changes lie in between form and operational changes on the agility and performance tradeoff axis. Implementing cyber changes in the field requires modular design, but modularity can be superimposed on existing cyber physical systems after production. Additionally, over-the-air updates require infrastructure investments to relay updates from some location to the system in the field. If proper design and infrastructure is in place, cyber change implementation is only delayed by the time required to develop software. For practitioners, understanding these pathways and their associated tradeoffs can enable better decision making about the type of change that should be undertaken based on the extent and urgency of the change needed.

Works Cited

Alia-Novobilski, M. (2016, December 28). *AgilePod 'reconfiguring' ISR mission*. Wright-Patterson AFB. https://www.wpafb.af.mil/News/Article-Display/Article/1038723/agilepod-reconfiguring-isr-mission/https%3A%2F%2Fwww.wpafb.af.mil%2FNews%2FArticle-Display%2FArticle%2F1038723%2Fagilepod-reconfiguring-isr-mission%2F

Alia-Novobilski, M. (2018, January 2). *AFRL's AgilePod shows ISR versatility during Scorpion fit test*. Wright-Patterson AFB. http://www.wpafb.af.mil/News/Article-Display/Article/1406999/afrls-agilepod-shows-isr-versatility-during-scorpion-fit-test

Allen, J. D., Mattson, C. A., & Ferguson, S. M. (2016). Evaluation of System Evolvability Based on Usable Excess. *Journal of Mechanical Design*, *138*(9). https://doi.org/10.1115/1.4033989

Bachmann, F., Bass, L., & Nord, R. (2007). *Modifiability Tactics:* (Technical Report CMU/SEI-2007-TR-002). Software Architecture Technology Initiative. https://doi.org/10.21236/ADA472581

Baheti, R., & Gill, H. (2011). Cyber-Physical Systems. *The Impact of Control Technology*, 161–166.

Baldwin, C. Y., & Clark, K. B. (2000). *Design Rules: The power of modularity*. MIT Press.

Beesemyer, J. C. (2012). *Empirically characterizing evolvability and changeability in engineering systems* [Thesis, Massachusetts Institute of Technology]. https://dspace.mit.edu/handle/1721.1/76092

Brown, M., Dey, S., Tuxworth, G., Co, J., Bernus, P., & Souza, P. de. (2022). An Ility Calculation for Satellite Software Validation. *2022 IEEE Aerospace Conference (AERO)*, 1–20. https://doi.org/10.1109/AERO53065.2022.9843603

Chaillan, N. (2019, November 22). *How the Department of Defense Moved to Kubernetes and Istio*. KubeCon + CloudNativeCon North America 2019, San Diego, California. https://www.youtube.com/watch?v=YjZ4AZ7hRM0

Chen, C., Lin, S., Shoga, M., Wang, Q., & Boehm, B. (2018). How Do Defects Hurt Qualities? An Empirical Study on Characterizing a Software Maintainability Ontology in Open Source Software. *2018 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, 226–237. https://doi.org/10.1109/QRS.2018.00036

Cogliano, J. (2015, June 12). *KEYW Corp. Tapped to develop more affordable, flexible aircraft spy pods*. Dayton Business Journal. https://www.bizjournals.com/dayton/news/2015/06/12/exclusive-firm-tapped-to-develop-more-affordable.html

Colombo, E. F. (2016). *Open innovation meets Changeability: Strategic design analyses for Cyber-physical Industry platforms* [PhD, Politecnico di Milano]. https://www.politesi.polimi.it/handle/10589/117765

Cox, A. (2017). *Functional Gain and Change Mechanisms in Post-Production Complex Systems* [The George Washington University]. https://www.proquest.com/openview/f0c8db7bd87e0fdc086e13fbe0ba523a/1.pdf?cbl=18750&pq-origsite=gscholar

Cross II, C. F. (2014). *The Dragon Lady Meets the Challenge: The U-2 in Desert Storm* (p. 125). 9th Reconnaissance Wing Historian.

de Neufville, R. (2003). Real Options: Dealing With Uncertainty in Systems Planning and Design. *Integrated Assessment*, *4*(1), 26–34. https://doi.org/10.1076/iaij.4.1.26.16461

de Neufville, R., Scholtes, S., & Wang, T. (2006). Real Options by Spreadsheet: Parking Garage Case Example. *Journal of Infrastructure Systems*, *12*(2), 107–111. https://doi.org/10.1061/(ASCE)1076-0342(2006)12:2(107)

de Weck, O. L., Ross, A. M., & Rhodes, D. H. (2012). *Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities)* [Working Paper]. Massachusetts Institute of Technology. Engineering Systems Division. https://dspace.mit.edu/handle/1721.1/102927

Eckert, C., & Isaksson, O. (2017). Safety Margins and Design Margins: A Differentiation between Interconnected Concepts. *Procedia CIRP*, *60*, 267–272. https://doi.org/10.1016/j.procir.2017.03.140

F-16 System Program Office. (2021, July 31). *F-16 receives in-flight software update during recent flight test*. Air Force. https://www.af.mil/News/Article-Display/Article/2715206/f-16-receives-in-flight-software-update-during-recent-flight-test/https%3A%2F%2Fwww.af.mil%2FNews%2FArticle-Display%2FArticle%2F2715206%2Ff-16-receives-in-flight-software-update-during-recent-flight-test%2F

Fricke, E., & Schulz, A. P. (2005). Design for changeability (DfC): Principles to enable changes in systems throughout their entire lifecycle. *Systems Engineering*, *8*(4), 342–359. https://doi.org/10.1002/sys.20039

Gerken, T. (2022, November 24). Mercedes-Benz to introduce acceleration subscription fee. *BBC News*. https://www.bbc.com/news/technology-63743597

Hadley, G. (2022, September 1). *F-22 Flies With Third-Party Apps, New Open Software Architecture*. Air & Space Forces Magazine. https://www.airandspaceforces.com/f-22-flies-with-third-party-apps-new-open-software-architecture/

Hölttä, K., Suh, E. S., & de Weck, O. (2005). Tradeoff Between Modularity and Performance for Engineered Systems. *DS 35: Proceedings ICED 05, the 15th International Conference on Engineering Design, Melbourne, Australia, 15.-18.08.2005*, 449–450.

Insinna, V. (2020, October 19). *US Air Force sends software updates to one of its oldest aircraft midair*. Defense News. https://www.defensenews.com/air/2020/10/09/the-air-force-updated-the-software-on-one-of-its-oldest-aircraft-while-it-was-in-the-air/

Krisper, M., & Kreiner, C. (2016). Describing binding time in software design patterns. *Proceedings of the 21st European Conference on Pattern Languages of Programs*, 1–15. https://doi.org/10.1145/3011784.3011811

Mekdeci, B., Ross, A. M., Rhodes, D. H., & Hastings, D. E. (2015). Pliability and Viable Systems: Maintaining Value Under Changing Conditions. *IEEE Systems Journal*, *9*(4), 1173–1184.

Nine, J., Shirey, R., Thompson, B., George, A., Cunningham, J., & Mason, A. (2019). Open Adaptable Architecture (OA2) for Agile Intelligence Surveillance Reconnaissance (ISR). *Open Architecture/Open Business Model Net-Centric Systems and Defense Transformation 2019*, *11015*, 102–112. https://doi.org/10.1117/12.2518819

Ross, A. M., & Rhodes, D. H. (2019). Ilities Semantic Basis: Research Progress and Future Directions. *Procedia Computer Science*, *153*, 126–134. https://doi.org/10.1016/j.procs.2019.05.063

Ross, A. M., Rhodes, D. H., & Hastings, D. E. (2008). Defining changeability: Reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value. *Systems Engineering*, *11*(3), 246–262. https://doi.org/10.1002/sys.20098

Sapol, S. J., & Szajnfarber, Z. (2020). Revisiting Flexibility in Design: An Analysis of the Impact of Implementation Uncertainty on the Value of Real Options. *Journal of Mechanical Design*, *142*(12). https://doi.org/10.1115/1.4047682

Shirey, R. G., Borntrager, L. A., Soine, A. T., & Green, D. M. (2017). Blue Guardian: Open architecture intelligence, surveillance, and reconnaissance (ISR) demonstrations. *Open Architecture/Open Business Model Net-Centric Systems and Defense Transformation 2017*, *10205*, 123–129. https://doi.org/10.1117/12.2263864

Singh, A., & Szajnfarber, Z. (2022). Understanding Post-Production Change and Its Implication for System Design: A Case Study in Close Air Support During Desert Storm. *Naval Engineers Journal*, *134*(3), 87–95.

Smith, P. (2021). *Fairchild Republic A-10 Thunderbolt II: The "Warthog" Ground Attack Aircraft*. Pen and Sword.

Tackett, M. W. P., Mattson, C. A., & Ferguson, S. M. (2014). A Model for Quantifying System Evolvability Based on Excess and Capacity. *Journal of Mechanical Design*, *136*(5), 051002. https://doi.org/10.1115/1.4026648

Trevithick, J. (2018a, January 3). *USAF Uses Textron's Scorpion Jet As the Latest Testbed for Its Modular Sensor Pod*. The Drive. https://www.thedrive.com/the-war-zone/17352/usaf-uses-textrons-scorpion-jet-as-the-latest-testbed-for-its-modular-sensor-pod

Trevithick, J. (2018b, September 10). *USAF Plans To Drastically Boost Flexibility of U-2s and RQ-4s By Adding Modular "Agile Pods."* The Drive. https://www.thedrive.com/the-war-zone/23489/usaf-plans-to-drastically-boost-flexibility-of-u-2s-and-rq-4s-by-adding-modular-agile-pods

Trevithick, J. (2020, October 19). *U-2 Spy Plane Got New Target Recognition Capabilities In First Ever In Flight Software Updates*. The Drive. https://www.thedrive.com/the-war-zone/37131/u-2-spy-plane-got-new-target-recognition-capabilities-in-first-ever-in-flight-software-update