Sustaining Software-Intensive Systems

Mary Ann Lapham
Contributor: Carol Woody, PhD

May 2006

Acquisition Support Program

Technical Note
CMU/SEI-2006-TN-007

This work is sponsored by the U.S. Department of Defense.

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Contents

List of Figures

Acknowledgements

The author would like to acknowledge and thank Lt Col Pat Rizzuto, United States Air Force (USAF), Lt Col Lani Smith, USAF, and Major Trauna James, USAF and their respective staffs for their openness and willingness to share all aspects of their sustainment activities.

Abstract

As today's systems become increasingly reliant on software, the issues surrounding sustainment become increasingly complex. The risks of ignoring these issues can potentially undermine the stability, enhancement, and longevity of fielded systems.

Questions about sustaining new and legacy systems include

1. What does it mean to perform sustainment from a software perspective?
2. What types of development and acquisition activities are required to sustain software-intensive systems?
3. Although the Department of Defense (DoD) has a technical definition of sustainment, does the DoD typically consider sustainment as maintenance?
4. How does the increased use of commercial-off-the-shelf software complicate sustainment?

This technical note discusses these questions and presents definitions, related issues, future considerations, and recommendations for sustaining software-intensive systems. Sustainment done well leads to well-supported software-intensive systems and reduced total ownership costs and should help organizations meet current and new mission area and capabilities requirements.

The information contained in this technical note is based on information that the Software Engineering Institute gathered during work with Air Force software-intensive systems. While the information is pertinent and can be applied to systems in the commercial sector, keep in mind minimal effort was made to convert "DoDspeak" into commercial sector language.

1 Software Maintenance and Sustainment Definitions

As today's systems become increasingly reliant on software, the issues surrounding sustainment become increasingly complex. The risks of ignoring these issues can potentially undermine the stability, enhancement, and longevity of systems in the field.

At the center of this puzzle are disparate definitions. Developers and acquirers have a general understanding that sustainment involves modifying systems and deploying changes to meet customer needs, but does this understanding align with common practice and the DoD's definition of sustainment? DoD Instruction 5000.2 describes sustainment as follows:

> Sustainment includes supply, maintenance, transportation, sustaining engineering, data management, configuration management, manpower, personnel, training, habitability, survivability, environment, safety (including explosives safety), occupational health, protection of critical program information, anti-tamper provisions, and information technology (IT), including National Security Systems (NSS), supportability and interoperability functions [DoD 03a, Section 3.9.2.1].

The terms *software maintenance* and *software sustainment* are often used interchangeably. It is important to make sure that all stakeholders use the same terminology when discussing software sustainment. The descriptions contained in the following sections help distinguish between these terms as they are applied in this technical note.

1.1 What Is Software Maintenance?

The *IEEE Standard Glossary of Software Engineering Terminology* defines "software maintenance" as follows:

> The process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment [IEEE 90a].

Software maintenance consists of correcting faults, improving performance or other attributes, and adapting to a changing organization and technical environment. To be complete, there is usually a fourth category of maintenance activities focused on anticipated problems, or preventive maintenance.

1.2 What Is Software Sustainment?

While DoD Instruction 5000.2 describes sustainment in detail, no authoritative definition of "software sustainment" exists. The SEI's working definition is as follows:

> The processes, procedures, people, materiel, and information required to support, maintain, and operate the software aspects of a system.

Given this definition, software sustainment addresses other issues not always an integral part of maintenance such as documentation, operations, deployment, security, configuration management, training (users and sustainment personnel), help desk, COTS product management, and technology refresh. Successful software sustainment consists of more than modifying and updating source code. It also depends on the experience of the sustainment organization, the skills of the sustainment team, the adaptability of the customer, and the operational domain of the team. Thus, software maintenance as well as operations should be considered part of software sustainment.

2 Criteria to Enter Sustainment

At some point during a system's life cycle, it should enter sustainment, but when? What are the entrance criteria for sustainment? Our research found no specific DoD or Air Force guidance regarding entrance criteria for sustainment. However, there seems to be a qualitative notion of when a system is *supposed to* enter sustainment. In some cases, fielded systems have been declared in sustainment without having undergone any formal readiness planning. So, to fill the void, we developed our own short list of entrance criteria. While it is not all-inclusive, it does provide a useful starting point for deciding when a system is ready to enter sustainment.

Based on our analysis, each of the following criteria is necessary for a system to enter sustainment:

- signed Source of Repair Assignment Process (SORAP) or equivalent document
- completed Multi-Service Operational Test and Evaluation (MOT&E) for the potential production software package (or OT&E if not multi-service)
- stable software production baseline
- complete and current software documentation
- Authority to Operate (ATO) for an operational software system
- current and negotiated sustainment transition plan
- sustainment staffing and training plan

Each of these criteria is discussed in more detail in the subsequent sections of this technical note. In addition to this list of criteria, a sustainment improvement checklist from the Department of the Air Force *Guidelines for Successful Acquisition and Management of Software-Intensive Systems*, included in Appendix A, may provide insight into the multifaceted task of preparing for sustainment [DoAF 03]. Other possible sources of information include the Defense Systems Management College's *Acquisition Logistics Guide* [DSMC 97] and the DoD's *National Security Space Acquisition Policy* [DoD 04].

2.1 Source of Repair Assignment Process

Eventually, a software-intensive system will enter sustainment and it seems only prudent to plan for that event. A Source of Repair Assignment Process (SORAP) document informs stakeholders who is responsible for a program's sustainment. Typically, it takes significant time to complete this document and to obtain approval, so it should be started during the technology development phase. For non-governmental groups, a strategy for how a system

will be sustained should be created before production and deployment starts. The sustainment strategy can impact how a system is designed and implemented.

DoD Directive 5000.1 requires that all acquisition programs include sustainment strategies that incorporate the "best use of public and private sector capabilities through government/industry partnering initiatives in accordance with statutory requirements" [DoD 03b]. These statutory requirements require that a source of repair analysis be completed as part of the acquisition strategy produced for Milestone B or Milestone C within the Defense Acquisition Management Framework[1] [DoD 03a]. This means that before a program starts doing actual development (Milestone B), or at least before it enters production (Milestone C), the program has a sustainment strategy in place. Note that sustainment plans are implemented in a variety of ways across the armed services. For example, the Navy produces an Integrated Logistics Support Plan, while the Army completes Post Production Software Support planning. For the Air Force, a SORAP is required according to Air Force Instruction 63-107 [DoAF 04]. Even with all the requirements, sometimes systems enter sustainment without the prerequisite plans having been completed. Starting sustainment without some type of documented strategy is starting sustainment without a clear plan. Without a clear plan, the odds of sustainment success are reduced.

2.2 Multi-Service Operational Test and Evaluation

A system must successfully complete OT&E, or MOT&E if it is multi-service (e.g., Air Force, Navy, Army), before it can enter sustainment. Successful completion of OT&E verifies and validates that the system is operational or at least is ready to begin operation. Logically, sustainment of a system that is not operational cannot occur.

In addition, the acquisition phases preceding Milestones A, B, and C need to be complete in order for a system to properly be prepared to enter sustainment. These milestones are at the end of the concept refinement, technology development, and system development and demonstration phases, respectively. Each milestone represents a critical review point during the system acquisition.

2.3 Stable Software Baseline

Most sustainment organizations will not accept software into sustainment until the software is stable. Merriam-Webster Online defines *stable* as "a. firmly established: fixed, steadfast b. not changing or fluctuating: unvarying c. permanent, enduring" [M-W 05]. However, in the realm of software stable can mean different things.

If one were to apply Merriam-Webster's definition to software, he or she could infer that a single instance of loss of availability or a system failure would indicate that the software is

[1] More information about the Defense Acquisition Management Framework is available at http://akss.dau.mil/dag/DoD5000.asp?view=framework.

not stable. In other words, software is stable only if it does not have problems that cause it to stop working. For software, unfortunately, the definition of stable can be a subjective one from several different perspectives. One organization may be willing to accept software as stable if it only fails once a week, while others would deem this rate of failure too high and would not accept the software. In other situations, software may be considered stable if no Category 1 or 2 Software Trouble Reports (STRs) exist.

Defining the stability of a system depends somewhat on its intended use, its mission criticality, and the potential consequences if the system fails. For instance, a system such as navigation software or command and control software whose failure could result in loss of life should have more stringent requirements for maintaining stability than one that is business software supporting actions that could be postponed for hours or even days.

The program office should define the criteria for accepting a system as stable in the Sustainment Transition Plan. These criteria should at the very least identify the types of STRs allowed to be active in a system that is entering sustainment.

2.4 Complete Documentation

Complete, current software documentation is paramount for the software sustainment organization. Without it, the sustainment organization has limited insight into how the software was designed and implemented. Incompleteness or omissions increase software maintenance costs because software engineers have to reverse-engineer the code to determine how it works. In addition, this process increases the risk of inadvertently introducing errors into the code.

The program office should determine what constitutes complete documentation for their system. At a minimum the documentation set should include the "why, how, what, and where" of the system. That is, documents should allow the sustainment organization to understand why the system was designed, how the system was developed, what the system consists of, and where functions were allocated to different subsystems. The overall architecture or blueprint for the system needs to be provided. Plans on how the program office intended to handle COTS and configuration management issues are essential for sustainment and continued implementation. Interface definitions need to be documented. Database designs and their documentation are essential to understanding their purpose within the system. Lastly, the development environment needs to be defined so the sustainment organization knows what tools were used to develop and support the system. This information is contained in the following documents:

- Initial Design Document (IDD)
- System/Subsystem Design Document (SSDD)
- system and software architectures
- COTS Management Plan

- Configuration Management (CM) Plan
- Interface Control Document (ICD)
- database design and documentation
- software development environment documentation

Often, these documents exist but lack sufficient detail to ensure a system's successful transition to the sustainment organization. Thus, not only must the sustainment organization be provided these documents, the documents need to be complete and detailed.

2.5 Authority to Operate

Before a system can be considered operational in the field, and thus meet the criteria to enter sustainment, an Authority to Operate (ATO) must be issued. The ATO issuance depends on approval of security requirements by the Designated Approval Authority (DAA). Issuing an ATO means that a DAA has accepted that operation of the system represents a low security risk. An ATO is issued for a fixed period of time (typically three years) and must be renewed. Delay in obtaining ATO approval or renewal could cause the system to be deemed non-operational.

The process for obtaining an ATO is well documented in DoD Instruction 5200.40, the *DoD Information Technology Security Certification and Accreditation Process (DITSCAP)* [DoD 97]. However, it is important to note that the ATO process is time consuming and requires updates either whenever the system's risk potential is significantly changed or at scheduled intervals when the ATO requires renewal.

Other security issues that need to be addressed during sustainment planning are the requirements for information assurance (IA) and how the IA role will be transferred from the development organization to the sustainment organization. Some key questions are

- Are the IA requirements understood by both the development and sustainment organizations?
- Is the IA role in the transition process well understood by both the development and sustainment organizations?
- What additional effort is required from both organizations?

2.6 Sustainment Transition Plan

In many instances, a program has been developed, tested, and declared operational but there is no funding set aside to address creation and subsequent negotiation of the sustainment transition plan. Unfortunately, in an era where budgets are becoming increasingly tight, sustainment planning is postponed and in some instances forgotten.

Both the development organization and the sustainment organization need to be involved in creating the sustainment transition plan. If a contractor is involved in development, that

organization also needs to participate in the development and subsequent negotiation of the sustainment transition plan. In addition, the contract should include tasks that address the contractor's role in the sustainment planning and transition process.

The program office should ensure that while the program is being developed, sustainment tasks are not forgotten or removed from the development contractor's tasking. While the development contractor may not necessarily become the sustainment organization, the development contractor is responsible for developing and maintaining documentation that the sustainment organization will need. It is the program office's responsibility to ensure that the contractor does not create documentation that is proprietary or undeliverable. Even though it was cancelled in 1998, the MIL-STD-498, Section 5.13, "Preparing for Software Transition," contains good background and reference material in this area [DoD 94].

2.7 Staffing and Training Plan

Staffing the sustainment organization is critical. The staff needs to be trained software professionals that can work with the development organization to transfer the necessary system knowledge. One should not assume that any of the development organization staff will transition to the sustainment organization. But rather, adopt a plan to transfer the knowledge from one organization to the other as part of the staffing plan. The staffing and training plan are related to and should be coordinated with the Sustainment Transition Plan.

As with many other areas associated with sustainment, training for the sustainment organization is often treated as an afterthought and is usually an under-funded activity. Even though the sustainment staff is composed of trained software professionals, they still need training on the specifics of the system entering sustainment. This is especially true for the increasingly complex systems that contain a mixture of COTS, government off-the-shelf (GOTS), and organic (government-developed) software code. "On-the-job" training is not sufficient for personnel sustaining these types of complex systems. The system's specific architecture, design decisions, and other nuances need to be communicated in some depth. Providing complete documentation as discussed in Section 2.4 helps, but it is not a substitute for a formal training plan that addresses the needs of the users, administrators, operators, and developers in the sustainment organization.

3 Sustainment Challenges

As with any type of change or transition, a variety of issues need to be considered. Our research uncovered a variety of issues or challenges that we have grouped into six categories. This is not to say that one would not find other issues that must be addressed when a system is entering sustainment. In addition, no priority is implied by the order in which these topics are discussed.

The following categories of sustainment challenges are discussed in detail in the sections that follow.

- sustainment with COTS software
- programmatic considerations
- system transition to sustainment
- user support
- information assurance
- development versus sustainment

3.1 Sustainment with COTS Software

Organizations have been sustaining software for decades. The processes and procedures for sustaining legacy software systems have evolved over time and are usually well defined. These legacy software systems are typically monolithic systems that were built to support a single enterprise or client using approaches specifically tailored for that enterprise or client. However, sustainment today is more complicated due to the increased use of commercial off-the-shelf (COTS) software products. A COTS software product is one that is

- sold, leased, or licensed to the general public
- offered by a vendor trying to profit from it
- supported and evolved by the vendor, who retains intellectual property rights
- available in multiple, identical copies
- used without modification of its internals

A COTS-based system is a composite of one or more off-the-shelf components (reuse, legacy, COTS) from one or more vendors, plus any custom components integrated to achieve new or expanded system functionality. COTS-based systems can range from *COTS-solution systems* to *COTS-aggregate systems*. A COTS-solution system consists of one product or a suite of products, usually from a single vendor, tailored to provide system functionality. A

COTS-aggregate system consists of multiple products from multiple vendors integrated to provide system functionality [Brownsword 00].

Choosing a sustainment organization that has a limited understanding of COTS product maintenance and sustainment can present a high risk to the program. Additional issues that present significant risk during sustainment include system obsolescence, technology refresh, source code escrow, vendor license management, and COTS-aggregate system architectures. The complexity of supporting each COTS product on a different timetable complicates sustainment even further. Each of these potential issues is discussed in more detail in the sections that follow.

3.1.1 Sustainment-Ready COTS-Intensive Systems

Some acquirers and developers might argue that due to the very nature of a COTS-intensive system, it shouldn't be a candidate for entering sustainment because the system is effectively undergoing continuous spiral development. In reality, the resolution of this argument significantly depends on the integration and sustainment organizations' understanding of COTS products and the organizations' abilities to plan projects effectively. The integration organization's application of well-managed development processes using effective requirements and configuration management provides a strong foundation for transitioning a COTS-intensive system to sustainment. How well the sustainment organization has prepared itself for handling COTS-specific maintenance issues has a direct bearing on the success of the sustainment effort. Another factor is how the sustainment organization is equipped and organized. While DoD directives indicate an increase in and preference for using COTS software products, the sustainment support organizations for these types of systems may not be equipped to provide the level of support required for COTS-intensive systems. Sustainment support organizations can exhibit the following deficiencies:

- They assume that annual code refresh cycles are adequate.
- Their method of configuration management is primarily hardware-focused.
- There is a minimal, if any, vendor management plan to ensure key functionality remains available in the COTS products being used.
- They rarely use any formal evaluation criteria to determine or confirm when a system is ready to enter sustainment.

3.1.2 The Impact of COTS-Intensive Systems on Sustainment

The following subsections describe four critical areas that need to be addressed when sustaining a COTS-intensive software system. Undoubtedly, there are many more, but these provide unique challenges:

- system obsolescence, technology refresh, and upgrade planning
- source code escrow
- vendor license management

- architecture and COTS software interfaces

Information security is another significant challenge when sustaining COTS-intensive systems. This issue is addressed in Section 3.5, Information Assurance.

3.1.2.1 System Obsolescence, Technology Refresh, and Upgrade Planning

COTS-based systems are composed of one or more COTS products. As shown in Figure 1, each COTS software product life cycle includes updates, refreshes, and obsolescence (i.e., unsupported releases). This life cycle is not based on the users' requests or budgetary cycles, but rather on marketplace demands and COTS software vendors' business plans. In fact, most COTS software products undergo a major change with a substantially new version being released at least every two years. In addition, the technology on which COTS software relies is constantly advancing, which requires the program office to maintain an orderly refresh plan.

Without careful planning and thought in designing a COTS-intensive system, the redesign, development, and retesting efforts due to the differences in configuration and tailoring of the replacement COTS products versus the original COTS products can be substantial. It is essential for organizations to create and maintain a thorough COTS management plan that addresses these issues and provides clear plans to incorporate updates and new releases of COTS software products. The evolution of the system must account for how all the COTS software components work together and how changes to one component can affect the others. Adequate development tools, training and skills, and parallel testing and production environments are needed to integrate and test updates and new releases.

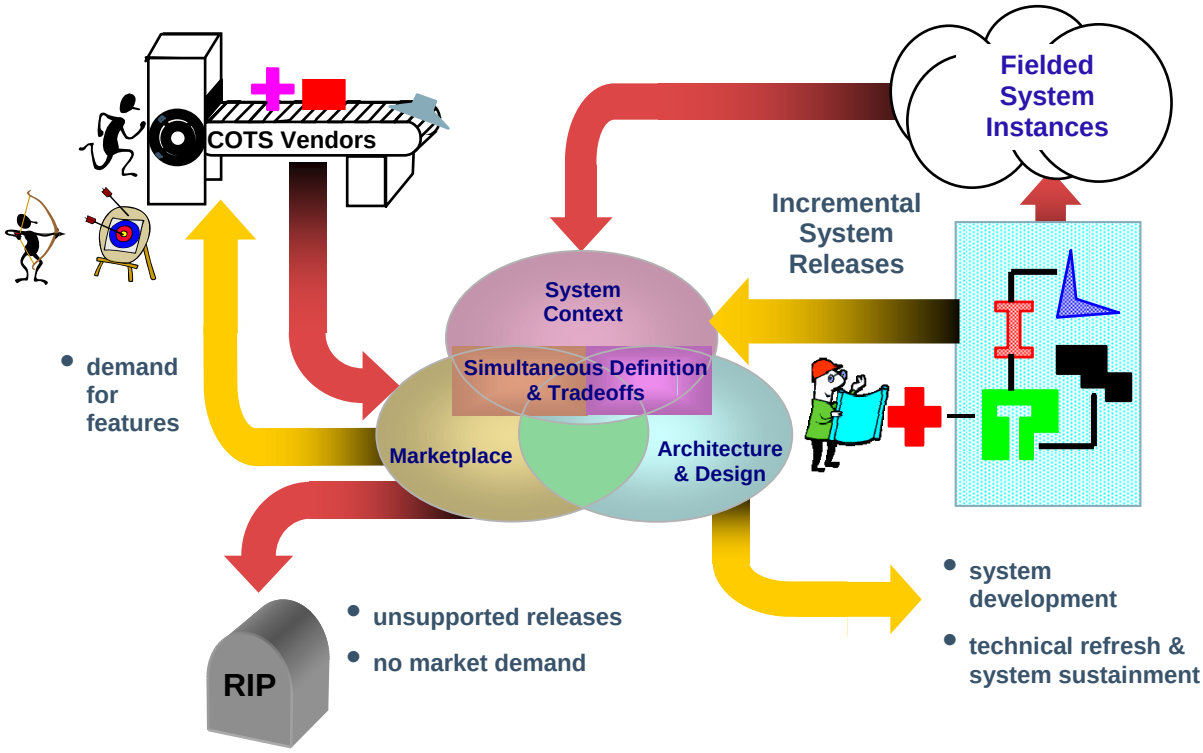

Figure 1: Cyclic Nature of COTS-Based Systems

3.1.2.2 Source Code Escrow

Many organizations buy COTS-based systems or more typically, have COTS-based systems integrated and delivered by a third-party organization. In these instances, the source code is owned by the COTS software vendor or perhaps the third-party integrator. Problems can arise when the COTS vendor goes out of business or no longer exists due to a business merger or acquisition. If the customer, which may be the program office or the sustainment organization, does not have rights to the source code, sustainment of the system is impossible. In addition, access to the source code alone is not sufficient; access rights need to include the software development environment in which the components were developed.

One way to counteract this issue is to ensure that there are escrow clauses in any contracts with COTS software vendors or system integrators. Escrow clauses require the vendor to place a copy of the code in escrow for release to the customer under certain circumstances that are defined in the contract, for example, if the vendor goes out of business. However, specifying escrow clauses in the contract is no guarantee that issues regarding access to the source code and development environment can be easily resolved.

3.1.2.3 Vendor License Management

License management can pose several issues to the sustainment organization, depending on how the licensing is handled. During system development, product licenses are sometimes managed by the system integrator. The transition of license management tasks to the sustainment organization needs to be jointly planned by the program office and sustainment

organization. In large organizations, a centralized approach to license management makes the most sense for economies of scale and overall holistic system view. However, not all organizations are equipped to use a centralized approach and may assume that each individual system user is responsible for managing his or her license(s). Again, planning and forethought can prevent licensing issues from becoming a "show stopper." In addition to defining the roles and responsibilities for transitioning software licenses, the budget for maintaining the licenses needs to be specified in fiscal plans for the organizational unit designated with ownership of the licenses.

3.1.2.4 Architecture and COTS Software Interfaces

Software architecture is "the structure or structures of the system, which comprise software elements, the externally visible properties of those elements and the relationships among them" [Bass 03]. Software architectures define the context in which software elements will be designed and acquired.

Sometimes during system development, third-party integrators or development organizations will capitalize on informal relationships with COTS software vendors to acquire system-specific capabilities. The resulting capabilities may not be in the official version of the product and there is no guarantee that these "extra" features will be maintained as the product evolves. In fact, some COTS software vendors charge ongoing license fees to support the special version of the product or the custom features that are not part of the product baseline. Beware, as this can become costly and may undermine many of the long-term benefits of using COTS software products. Over time, the customized version can significantly diverge from the supported product and from the original architecture of the system.

Another architecture-related issue to consider during sustainment planning is the interchangeability of different COTS software products. The system architecture should be designed such that the system as a whole is insulated from COTS product interfaces. If this is done, when the COTS product changes, there should not be a ripple effect of changes to the interfaces. In addition, consideration needs to be given to COTS products that perform the same or similar functions so that options are identified. For example, sometime during a systems' life cycle, a COTS product may need to be replaced for fiscal or functional reasons. Programs should ensure that alternative products are available to provide the required functionality within the architectural constraints. If the system depends on one specific COTS product that has no acceptable alternatives, performing future upgrades and sustainment could be increasingly difficult or even impossible.

Finally, in order to track and influence which capabilities are included in COTS software products, the sustainment organization, like the program office before them, must be prepared to participate in the COTS software vendors' user groups. If the sustainment organization or the program office doesn't have sufficient clout to force a vendor to change a product, a larger coalition could be formed to convince the vendor to implement a certain change if others in the user group are also interested in the capability. User group

participation also gives the program office or sustainment organization an opportunity to become familiar with the vendor's product roadmap, so that it has early warning about intended future product changes that could impact the system.

3.2 Programmatic Considerations

Many regulations, policies, and acquisition best practices stress that sustainment needs to be considered early in the acquisition process. However, in many DoD programs, there is a long period of time between contract award and when the system enters sustainment. Given the budget and schedule pressures during the development phase, sustainment requirements are sometimes relegated to minor importance at best; at worst, they are deferred, ignored, or dropped from the requirements entirely. Unfortunately, deferred or ignored requirements are the root cause of many sustainment woes that a program faces.

Even when sustainment planning survives program cutbacks and pressures, there can be a lack of discipline in continuing sustainment planning throughout the early phases of the program. It's not that the program office lacks discipline, but rather that the lesser requirements get postponed and fall into the "I'll worry about it later because sustainment is a long time off" category. Once the sustainment requirements are lowered in importance or become ignored altogether, the system is no longer being designed to be sustainable. This is the beginning of future sustainment woes. For example, questions addressing sustainment issues will not be asked at design reviews. Other symptoms that indicate there may be problems sustaining a system in the future include the following:

- Documentation (especially in the case of COTS software) is eliminated, leading to poor quality or incomplete documentation to pass on to the sustainment organization. For instance, rigorous system design documentation may not be completed since specific information is not available for the COTS products. Thus, the designers produce limited design documentation at best, with the excuse "it's a COTS product and I don't have the information."
- System support documentation and databases are not contractual deliverables from the development contractors, leading to incomplete system documentation.
- There is little or no coordination across different segments of a program to use the same design standards, which may produce a system that is difficult to sustain.
- Little thought, if any, is given to how the sustainment personnel will be trained, potentially leading to a poorly prepared cadre of sustainers.

Once sustainment is placed in the category of "minor requirement," there seems to be a never-ending spiral of potential issues. These issues can be addressed, but they require consistent, persistent management. Program offices must consider sustainment when they are allocating the budget; funding for the contractor and sustainment organizations to prepare the transition plan is essential. In the balancing act of trading off requirements, especially non-functional requirements, sustainment needs to be considered as an important factor. The program office must also remember that planning for and executing the SORAP requires a

long lead time; a SORAP needs to be started and completed long before sustainment becomes a reality.

3.3 System Transition to Sustainment

Transitioning a system to sustainment requires careful planning, and there are a variety of areas that need to be addressed. The six areas discussed in the sections that follow have proven to be problematic despite (or maybe because of) the obvious nature of the inherent issues.

3.3.1 Support Database Transition

Support databases are databases being developed that the sustainment organization will need, in addition to those inherent to the system. Support databases contain configuration management data, software trouble reports (STRs), enhancement requests, registered system user data, and other information used by the development organization while developing the primary system. In many cases, support databases and the tools used to manipulate them are owned by the development contractor. The database structures and the associated tools may even be proprietary, but the actual data belongs to the government.

Eventually, the data contained in the support databases needs to be transitioned to the sustainment organization. Key questions to be addressed include

- Which support databases need to be transitioned?
- Who owns the databases?
- What's the specific content of the databases?
- What are the tools used to create, maintain, and manipulate the databases?
- How are the databases maintained?
- What is the purpose of each database?

A plan to migrate the support databases to the sustainment organization can be developed after these questions have been answered. The transition plan should include details about the need to purchase proprietary tools or to find COTS software replacements for them.

3.3.2 Development and Software Support Environment Infrastructure

The development environment usually includes tools, test suites and harnesses, support databases, a software test lab, configuration management, hardware spares, process and procedure documentation, and delivered source code. The program office needs to ensure that the sustainment organization uses an equivalent development environment. Portions of the development environment may be transferable. However, if there are any contractor-owned tools within the environment, they will not be provided to the sustainment organization and plans for alternative options must be developed and executed. In addition,

if any COTS software products are used, it is critical to note that both hardware and software is affected by product version changes. Thus, the sustainment organization needs compatible hardware and software.

The sustainment organization needs to know what hardware and software is being provided and what is not, so it can assess if it is prepared to sustain the system. A simple inventory of items should be provided to the sustainment organization by the development organization. Any items missing from the inventory may have to be purchased.

3.3.2.1 Software Test Lab

If the development organization is finished using the test lab, then it could be transferred to the sustainment organization. However, where to locate the software test lab and how and when to move it needs to be decided and planned. Often, the sustainment organization assumes that the program office has planned this transfer, but this may or may not be the case. The transfer of the lab should also include training about the contents of the lab for either organic or contractor logistic support (CLS) personnel in the sustainment organization.

3.3.2.2 Hardware Spares

The availability of hardware spares in the sustainment environment can impact the overall system including the software and documentation. This is especially true when old versions of the hardware are no longer available, which may mean that the software needs to be adapted to work with newer hardware. The number of depot spares available can impact this situation. Having additional older spares available lowers the probability that the software will need to be modified. The plan for administering any existing hardware spares needs to be included in the transition plan. In addition, the acquisition of additional spares for sustainment needs to be planned for and funded.

3.3.2.3 Release Processes and Procedures

During the technology development phase, several system releases can occur, especially if the development approach is spiral or incremental. Users can become accustomed to receiving releases containing new capabilities and defect fixes at certain intervals. A change in the number of releases executed by the sustainment organization could impact user satisfaction and the "health" of the system. Questions that need to be answered include

- What is a "reasonable" number of releases?
- What will the users expect?
- How do the various COTS product releases fit into a model that calls for one release per year?
- What is the impact to security risks if upgrades are delayed?

Release processes and procedures need complete investigation so plans can be made to sustain the system at the appropriate frequency to maintain effective operations and to

manage associated risks. All stakeholders need to be involved with updating and/or finalizing the release processes and procedures that will become effective once sustainment starts.

The release process and procedure review should include a thorough review of the software release approach. There seem to be two general approaches to releasing software. One method is a full release (i.e., completely replace the old version of the software) and the other is an incremental release (install a "patch" to add functionality or fix defects). There are pros and cons to each approach that should be considered within the context of the system entering sustainment. The software release model that will be used during sustainment needs to be documented, coordinated, and concurred.

3.3.3 Staffing

As the technology development phase ends and sustainment approaches, staffing issues that must be addressed will surface. Staffing issues need to be addressed regardless of the organization that is going to perform the sustainment effort. Issues include who should perform the sustainment, development staff retention to meet transition needs, and learning curve for sustainment personnel.

Some program office personnel profess that the development or integration organization is the only qualified sustainment organization. However, it could be done by a separate organization. Explicit and implicit risks are associated with both approaches. Explicitly, traditional sustainment organizations may not have the prerequisite skills or knowledge required to sustain the system. This can become even more complicated if the system contains integrated COTS software products and the sustainment organization is not experienced in COTS product sustainment. Implicitly, the developing or integrating organization will most likely employ different personnel to support the sustainment effort. The reason for this is that different personnel perform and are interested in doing development/integration work than those interested in performing sustainment work. This difference can be exacerbated by the fact that the development organization personnel are interested in staying "on the cutting edge" of software development, especially with COTS software products, and would most likely become bored with performing pure sustainment tasks. In either case, sustainment personnel need to be trained on the system, and knowledge transfer from the development organization to the sustainment organization is key for this training.

Availability of key development personnel to perform knowledge transfer raises another staffing issue. As the development phase winds down, retaining the subject matter experts for the system becomes increasingly difficult. Personnel leave the program to look for the next challenge or simply to stay employed. This attrition puts knowledge transfer at risk, especially if any type of "contractor partnering" is part of the planned transition to sustainment. Staffing retention plans need to be developed and implemented to ensure

experienced developers are available to act as subject matter experts during the transition of the knowledge base and to build training plans for each sustainment role.

By keeping development staff on board and building training plans, the learning curve for sustainment personnel should be shortened. Planning, training, and documentation all help in shortening "ramp-up" time. If COTS software products are included in the system, extra training may also be required to reduce the sustainers' learning curve.

3.3.4 Operations Training

Today's computing environment of highly integrated systems with heavy COTS product usage does not easily lend itself to effective separation of operational support and software sustainment. This can be a risk to the system in sustainment if the sustainment organization does not plan to provide operational support. Remember, the overall definition of sustainment includes operations!

If the sustainment organization does not support operations, make sure that training materials are developed for the operators before the program enters sustainment. Creating training materials may require additional work and, consequently, additional funding to compensate the development organization or contractor. Establish a relationship between the operational support organization (including the help desk) and the software sustainment organization to ensure full coverage of the system.

3.3.5 Transition Planning

Section 2 discussed criteria for entering sustainment. One of those criteria was completion of sustainment transition planning, which is one of the key elements to starting a successful sustainment effort. Unfortunately, sustainment transition planning can also be one of the issues. Many times the program office does not fund a transition plan or have one on contract for the development organization to accomplish. And even when a program office does ensure that a transition plan is created, it can be woefully incomplete. For example, the plan may not include ramp-up time or a complete roadmap for all segments of the system. It's most problematic when the transition plan mentions software only in passing, if at all.

To add to the dilemma, the development organization may believe that the sustainment organization is not sufficiently experienced or trained to sustain the system. Such a perception can hamper knowledge transfer by affecting interpersonal and inter-organizational communications. In order to mitigate this perception, the sustainment organization needs to demonstrate to the development organization their proficiency. Both organizations need to be contractually required to cooperate in transition planning, and ultimately, system hand off. The program office should support this goal by ensuring that system training and current, accurate documentation is supplied to the sustainment organization. The documentation must contain a roadmap for sustainment that addresses all the hardware, software, and

firmware segments across all possible configurations of the system. Most of all, the program office needs to include funding for transition in the budget.

3.3.6 System Documentation

A key element of transitioning to sustainment is the turnover of system documentation. Not just high-level documentation, but detailed system documentation from both functional and technical vantages. The detailed documentation should address dependencies and interactions of components, corporate knowledge of COTS software tuning, comprehensive architecture views, technical orders, and training information including help desk procedures. The technical orders should address the types of data that the operational staff will need to perform their jobs.

There are several pitfalls to watch for with regard to system documentation. The first is when documentation is available only from the contractor and it is not a deliverable specified in the contract. Another problem is documentation that is hardware-oriented and contains little information about the software. While having any documentation is a good start, it will need enhancements to be useful for software sustainment. Having multiple versions of similar documentation is also a problem, especially if the different variations were created by different sources. This indicates poor document management and implies that it will be difficult to provide current, accurate documentation to the sustainment organization. An indication of incomplete documentation is “TBD” (to be determined) or “TBS” (to be supplied) sections. Documentation that contains missing information should not be accepted as final.

Poor or incomplete system documentation can be rectified. In an ideal world, the documentation production task would be included in the original contract. However, this is not always the case or the documentation delivered is inadequate. The earlier the documentation problem is discovered, the easier it will be to fix. First, the program office needs to determine the level of documentation available today. Then, the program office should decide the appropriate level of detail that should be included in the documentation. The program office then should develop a plan to create/augment the documentation to meet that required level of detail. If contractor partnering is part of the sustainment plan, then the documentation updates should be targeted for completion during the first six months of the “contractor partnering” period.

3.4 User Support

Depending on the system and its intended use, user support issues need to be addressed when the system enters sustainment. Three key areas are help desk, user documentation, and user training.

3.4.1 Help Desk

Initial deployment of a system is usually accomplished by the development organization. As part of this initial deployment, this organization may resolve user issues, questions, and problems in the form of a help desk. Users of the system become accustomed to the level of support that is provided. During the transition to sustainment the users' help desk support may be transitioned to a different organization.

The level of support the users receive from the help desk must be maintained or improved during the transition to and during sustainment. Several issues may arise during the transition time frame:

- the amount of "hand-holding" expected by the users
- the mechanics of contacting the help desk
- overall user satisfaction

Let's look at these issues from a user's point of view. The development organization has just deployed a system (call it System Upgrade) to the users and is responsible for supporting them during this initial deployment. As the deployment becomes final, System Upgrade is now ready to enter into sustainment. Some users will still probably consider System Upgrade new to them, even though the system is moving to another phase in its life cycle.

At first, the users may be reluctant to use System Upgrade for many reasons. First, it represents change and most people do not like change. Second, the users are accustomed to the legacy system and while it may have faults, they know about them and know what to expect. When a new system such as System Upgrade is introduced, there is a learning curve and growing pains associated with becoming familiar with new software and hardware. Users expect and most likely need a lot of hand-holding to work through issues. Even after users learn System Upgrade, they are accustomed to having someone at their beck and call to help them resolve issues. However, when System Upgrade is transitioned to the sustainment organization, the amount of hand-holding will most likely change, and from the users' perspectives this change is a negative one. The sustainment organization may assume that the users are trained and are capable of working autonomously, while the users expect that the sustainment organization will provide the same level of support that the development organization did.

During the initial installation and execution of System Upgrade, it is likely that all issues with the system will be reported to a single entity. This approach allows users to communicate with a single point of contact (usually a help desk) for all their needs, whether they are having hardware problems, software problems, or operational problems (operator errors, inadequate training). However, sustainment organizations are not usually set up in this manner. In sustainment organizations, the most common logistics scenario for requesting help is for hardware and software to have separate lines of support (different physical locations, different contact information, etc.) It is also likely that operator errors will be relegated to training issues and the help desk personnel will not address them. Again,

this could be perceived by users as a lack of responsiveness and as a result, they feel less satisfied with the support they receive. Thus, with the advent of sustainment, the users could become less satisfied with the system. User satisfaction may not be formally measured, but it will be reflected in the users' attitudes and the number of complaints about the system.

In order to avoid the issues described in this section, users need to be informed and trained about how their support will change in sustainment. Planning and forethought can eliminate negative user perceptions and possible system downtime. In addition, developing a method to measure user satisfaction can allow the sustainment organization to measure and correct issues. If the sustainment is done by a contractor, this measure should be part of the contract performance metrics and rewards.

3.4.2 User Documentation

Documentation for users is just as important to a successful transition to sustainment as the system documentation is to the sustainment organization. User documentation usually takes the form of Technical Orders (TOs). TOs should be created and managed under configuration management before the transition to sustainment. TOs are usually a deliverable to the sustainment organization, but in some instances, a separate organization maintains them. In the case of multi-service-oriented systems, different organizations will maintain different sets of TOs. In any case, the configuration management of the user documentation must ensure that it is accurate across all versions, no matter who is maintaining the documents.

3.4.3 User Training

The development of user training materials is another area that is sometimes neglected during system development. Many program offices fall into the trap that training is something that can be deferred until later. This deferral can lead to the creation of inadequate training materials and a lack of coordinated training as the system enters sustainment.

Other symptoms of inadequate user training include no formal training for operators, no "train the trainer" program, and for multi-service systems, each service being responsible for its own training. In many cases, if the system is not officially operational, training for it will not be in the official government training pipeline. As a result, the help desk is burdened with the task of attempting to compensate for the missing training.

The program office should determine and plan who will provide user training for all system roles. The plan needs to include the impact of sustainment changes on user training and support. In addition, the government trainers need to know that the system is operational or will soon be so that the system can be added to the government school house pipeline to be included in the normal training curriculum and the development/maintenance of that curriculum. The training groups within different military organizations are responsible for creating and maintaining training for operational systems. The training group is another

stakeholder that needs to be informed of the system status so they can perform their training mission.

3.5 Information Assurance

Information assurance (IA) is critical to the continued viability of any fielded system. If the system no longer meets IA guidelines, it could be pulled from use, with disastrous implications for the users. Thus, appropriate IA support must be in place to ensure continued system security during sustainment. Changes to any system component, whether it's a custom component or a COTS component, must be investigated to ensure that no new vulnerability and subsequent risk has been introduced into the system. The next two sections discuss the unique challenge that COTS components present for IA in today's systems.

3.5.1 IA and COTS Software Products

Component changes, which are expected for COTS products, can affect the functioning of other components unpredictably. Each COTS software product is constructed with a set of simplifying requirements and context assumptions for interfaces that may change as the product evolves, rendering it inconsistent with the initial COTS-intensive system design. Such changes introduce opportunities for functional processing errors to occur, as well as potential vulnerabilities, such as buffer overflows and race conditions. In addition, COTS software vendors cannot cost-effectively test every possible product use case. Instead, testing focuses on the use cases considered to be most critical, which may not match a product's use by the system in sustainment. Also, due to the vendor continually seeking increased market value by expanding the range of capabilities for its products, functionality remains static in very few vendor products between upgrades. As with any software component, adding new capabilities increases the opportunity for defects, undocumented features, and the breaking of existing functionality.

As new vulnerabilities in COTS components are identified by CERT® and other reporting organizations, vendors provide patches and workaround options for critical security holes. The risk inherent with these changes is that they may break something else; patches and workarounds must be applied to base products with caution. The sustainment organization must be able to analyze the risks and determine if the patch application can be deferred until the next upgrade cycle or if it requires immediate application. Performing an annual upgrade is no longer sufficient for maintaining system security. This decision-making role requires that the sustainment organization has access to in-depth knowledge of the system security requirements, threat potentials, and the impact of a security compromise. Also, if COTS components include specially licensed features (features unique to the system and not necessarily available in the commercial version), these versions are usually patched last, if at all, depending on the negotiated level of vendor support. This potential lack of available

® CERT is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University. The CERT Coordination Center was the first computer security incident response team.

software patches may increase the potential for an entire system to be compromised, depending on the level of risk presented by the stale component.

3.5.2 Testing for IA

The sustainment organization must be prepared to perform regression testing of the entire system for each component upgrade or patch installed to ensure continuing functionality. If a test fails, they must be prepared to analyze each component and rework the system to establish functional equilibrium. Many sustainment organizations do not have parallel production environments in which to perform this level of validation, and most do not have sufficient insight into the internal workings of the COTS-intensive system to identify and correct problems. Because sustainment organizations do not typically develop software, the development tools, skills, and trained staff needed to adjust system components to accommodate a new or changed COTS component may not be available and have to be obtained. Also, without a robust regression test suite or environment, problem identification may require the involvement of functional users, a resource not typically available to sustainment organizations.

3.6 Parallel Development and Sustainment

Many of today's systems are fielded in an incremental manner. *Incremental* means that an increment or version of the system that provides partial capabilities is developed and fielded. The remaining capabilities are developed later depending on budget, requirements definition, and technology advancements. For the sustainment organization, this means that it will be sustaining a system in parallel with one that is still under development.

Development in parallel with sustainment is not a new concept; however, many sustainment organizations may not have experience with this mode of operation. To ensure continued operation of the system, the sustainment and development organizations need to develop processes and procedures, coordinate them with all parties, and obtain concurrence on their use.

Historically, in organizations that are successful in performing development and sustainment concurrently, groups within the organizations report to the same person. Given the organizational structure of the development and sustainment organizations, this can be problematic. In many instances, the person who has enough experience to oversee both the development and sustainment groups does not have the desire or the time to be involved in this level of oversight.

To better align parallel development and sustainment efforts, the program office needs to consider the current sustainment structure. With that in mind, it should then determine how the system being developed is evolving and how it can fit into the sustainment structure. Sustainment organizations should plan to adapt their processes to handle an evolving system, especially if it implements COTS hardware or software products.

In addition, a joint Configuration Control Board (CCB) needs to be created and given the authority to act. All decisions for changes to the baseline must go through the CCB **without exception**. The operational software **must** be driven from the CCB approved baseline. Last, a clear, documented path of escalation up to senior-level personnel **must** be created to address issues. It is not a question of if there will be issues, but when they will occur. Being prepared to handle issues reduces the impact problems have on the overall development and sustainment of the system.

4 Assessing Risks During Transition and Sustainment

Successfully sustaining a software-intensive system is a balancing act that is weighted on each side with the myriad of issues that surround such a system. Paying too much attention to one area often results in deficiencies in another. The key to successfully balancing the level of attention given to the issues is risk management.

Risk management is critical for both the development and sustainment organizations to ensure that the transition to sustainment is feasible and that it is as smooth as possible. The PMO must coordinate and perform the risk management to ensure that business risks are properly considered (i.e., the strategic risk assessment needs to cover development and sustainment). Both the development and sustainment organizations must be prepared to identify and analyze risks based on the functionality provided in the system, security concerns, COTS product supportability, product obsolescence, and a host of other related issues. When any of the potential risks reaches an unacceptable level, the responsible organization must have planned for and established the ability to make the necessary changes to reduce or remove unacceptable risk throughout the sustainment process. The organization responsible for the transition must also possess and transfer to the sustainment organization an appropriate level of system knowledge, COTS component knowledge, system validation capability and knowledge, and system support capability and knowledge.

System knowledge includes a comprehensive understanding of the system purpose from a business perspective. This applies not only to the technology pieces, but to the true business or mission-related reasons for having developed the system in the first place. These business or mission-related reasons are the major drivers for evolving and sustaining the system.

5 Conclusion

In addition to development and integration challenges, software–intensive systems present unique, multifaceted challenges in sustainment. The issues facing these systems are a result of their very nature. Software systems, especially those that use multiple COTS software products, are meant to evolve quickly to respond to users' needs. These characteristics lead to complications during sustainment. Organizations can more effectively sustain software-intensive systems by being aware of the issues and addressing them proactively. The top ten issues that must be addressed are

1. lack of funding for transition planning
2. lack of signed SORAP or equivalent document
3. unclear IA requirements
4. unaddressed support database and tools transition logistics
5. unclear COTS product license management
6. organic (Government) organization inexperience with software-intensive systems, especially COTS-based systems (there seems to be a lack of experience when organic military resources try to sustain COTS-intensive systems)
7. loss of key development staff and expertise
8. over-reliance on contractor development processes and proprietary tools
9. insufficient formal training for sustainment personnel
10. lack of parallel sustainment and development plans (this applies if development of the system continues during sustainment)

Appendix A Sustainment Improvement Checklist

The following checklist is excerpted from the Air Force *Guidelines for Successful Acquisition and Management of Software-Intensive System* document [DoAF 03, Section 16.4.1].

This checklist is included here to help you understand the sustainment issues that can affect your project. If you cannot answer a question in this list affirmatively, carefully examine the situation and take appropriate action.

1. Is all your software developed with a goal to facilitate its future sustainment?
2. Do you understand the four types of sustainment and their purposes?
3. Do you understand the place and purpose of the sustainment phase in the software life cycle?
4. Do you understand your sustainment process?
5. Is there a sustainment plan?
6. Is there a process in place to gather problem reports and upgrade requests for the software?
7. Does the plan provide for reviewing, evaluating, and prioritizing upgrade requests?
8. Are all sustainment activity steps included in the plan?
9. Is there a transition plan to move to the upgraded system?
10. Have all activities been planned and organized to keep interference and downtime to the operating system to a minimum?
11. Does the plan call for running critical systems redundantly during testing and installation?
12. Do the deliveries include source code, documentation, and all else that is needed in addition to the software itself to continue maintaining the software?
13. Are all products under configuration control?

Appendix B Tailorable Top-Level Guidance

The U.S. Air Force *Software Management Guidebook* is a product of the Air Force Software Intensive Systems Strategic Improvement Program (AFSSIP) working group that "provides the next level of definition in setting Air Force acquisition expectations for systems that have significant software content or software-driven functionality" [USAF 04]. This guidebook was created in response to multiple policies, studies, and Public Laws including a Weapon System Software Sustainment Study by Air Force Materiel Command completed in 2001, Public Law 07-314 (Section 804 of the FY03 Defense Authorization Act), and a joint policy memorandum from Dr. Marvin Sambur and Dr. Peter Teets on "Revitalizing the Software Aspects of Systems Engineering." This memorandum provided ten software focus areas including

> 9. Life Cycle Support: Address sustainment capability and capacity needs during the system design and development phase, and balance overall system acquisition and sustainment costs. Ensure you plan, develop, and maintain responsive life cycle software support capabilities and viable support options [SAF/AQ 04].

The *Software Management Guidebook* provides tailorable top-level guidance for organizations that acquire systems that require significant software development or integration. The following list summarizes the majority of the applicable guidance that can be tailored and included in a sustainment plan:

- Planning should consider all required software-related acquisition, development, and sustainment activities (Section 3.1).
- Planning for Operations and Support may be accomplished in a separate software or computer resources life cycle management plan (Section 3.1). Items appropriate to include in this plan are as follows:
  - Identifying roles and responsibilities of all stakeholder organizations in planning and implementing software sustainment capability
  - Planned source of life cycle (post deployment) software support and identification of the life cycle support infrastructure
  - Identification of computer systems hardware, software, documentation, and software engineering environment that will be delivered
  - Human resources required for software support with supporting assumptions and rationale
  - Intellectual property rights

  - Software test and integration considerations
  - Transition (if applicable) of all operational software and support tools from the developer to the post-deployment support organization
  - Interim support (if applicable) subsequent to the completion of system development but prior to the availability of permanent post deployment support
  - Security classification, certification, and accreditation considerations
  - Required facilities including buildings, integration labs, and test ranges
- Managing software related risks (Section 3.4). Specific Computer Systems & Software (CS&S) risks that programs should consider and manage include:
  - COTS/GOTS suitability, integration, and sustainment
  - New and unprecedented requirements such as extensive CS&S security requirements
  - Technical obsolescence of underlying computing architectures and hardware
  - Safety critical requirements
  - Uncontrolled sources of software (foreign developers, open source, etc.)
- Considering application and sustainment of non-developmental software (NDS) (Section 3.10). "Often this software has risk, and contractual and long-term support implications, particularly when programs find they need to modify the software."
  - General considerations such as rights to modify the software and potential software defects both known and unknown.
  - Consider how to handle impacts of fixes supplied to NDS as functionality may change which in turn may impact other software within the program.
  - Consider impact of limited or no access to source code (locking you into a particular vendor for continued use and support)
  - Address NDS maintenance and technology refresh in post deployment software support planning
  - Address unavailability of NDS when needed especially during sustainment
  - Contractually address NDS issues involving licensing, data rights, development cost, and warranty/support
- Planning for Software Support (Section 3.13.1). The planning should begin early, be documented and coordinated with all stakeholders.
  - "Determine and recommend a support concept and a source of post-deployment software support, based on analyses of the preliminary systems operational concept, other operational and support requirements, and Source of Repair Assignment Process (SORAP), as appropriate."
  - Identify and assign responsibilities to all affected organizations required to support the system
  - Identify all computer systems hardware, software, and engineering data that require support
  - Establish a strategy to respond to Diminishing Manufacturing Sources
  - Identify training requirements

- Identify human resources required to support the software
- Identify security requirements
- Define required computer resources support procedures, processes, and methods
- Establish criteria for transition from the developer to the post deployment support organization, and include consideration of government/contractor partnering opportunities
- Provide interim support between development completion and availability of permanent sustainment support (if applicable)
- Document initial software verification activities in order to form a baseline for future integration
- Procure and deliver all facilities, equipment, and data required to support the software

- Planning for Software Product Engineering Data (Section 3.13.2)
  - "Ensure minimum essential set of software product engineering data (documentation) is developed, acquired or escrowed, and maintained"
  - Lack of quality documentation can lead to inefficiencies or re-work during sustainment resulting in increased life-cycle costs. "This problem can occur whether the product is maintained organically, through the original developer, or through an alternate support contractor." (quotation marks added)
  - Steps to take include:
    - Plan, program and budget the development and test of System/Software Engineering Environments. Consider transferability of assets from development to sustainment
    - Determine and contract for sufficient quality documentation to support operations and sustainment needs
    - Ensure the developer's process defines and captures the required Software Product Engineering Data
    - Ensure the Software Product Engineering Data is delivered and/or escrowed to enable effective and efficient long-term system support
    - Establish appropriate government data rights
- Establishing Systems/Software Engineering Environments and Associated Facilities (Section 3.13.3). Adequate planning for software tools and facilities is critical. Key steps include:
  - Ensure developer identifies complete set of System/Software Engineering Environment development and test tools
  - Consider transferability of assets from development to sustainment and budget accordingly
  - Plan and budget for:
    - Development and test of environments and integration labs
    - Adequate spares or LRUs
    - Security requirements

- Selecting Computer Programming Language (Section 3.13.5)
  - Take into account the planned language(s) for the system and consider the sustainment needs
- Reviewing Software Integrated Master Plan (IMP) (Appendix A)

  Software sustainment is not specifically addressed in the *Software Management Guidebook,* but should be included in the IMP. However, several other IMP events set the stage for software sustainment. These are:
  - System Requirements Review (SRR) exit criteria include identifying the support concept and defining preliminary software support data.
  - Software Specification Review (SSR) exit criteria include defining life-cycle software support requirements
  - Preliminary Design Review (PDR) exit criteria include updating life-cycle software support requirements
  - Functional Configuration Audit (FCA) exit criteria include complete operational and support manuals and documents
  - Physical Configuration Audit (PCA) exit criteria include verifying software support and operational data is complete and accurate
- Suggested Software Content for Request for Proposal (RFP) Section L, "Instructions, Conditions, and Notices to Offerors" (Appendix C)
  - Include a statement such as "Describe the software product engineering data that you will develop and maintain as part of the software development/sustainment effort. Provide rationale."
- Suggested Software Content for Request for Proposal (RFP) Section M, "Evaluation Factors for Award" (Appendix D)
  - The technical factor for systems engineering requires the system meets needs for supportability.
- Contracting Considerations for Software (Appendix E)
  - "For Government Off-The-Shelf (GOTS) software, execute the acquisition management processes of Attachment 8, 'Accommodate Application and Sustainment of Non-Developmental Software.'"
- Computer Systems and Software (CS&S) Criteria for Technical Reviews (Appendix F)
  - CS&S Support to the System Requirements Review (SRR). The SRR should place particular emphasis "on ensuring that adequate consideration has been given to logistics support, software, test, and production constraints." Exit criteria that applies:
    - "programming languages and architectures, security requirements and, operational and support concepts have been identified"
  - CS&S Support to the Software Specification Review (SSR). The SSR has the following applicable exit criteria:

    - "Life-cycle software support requirements are compatible with, and incorporated into, the system lifecycle resource requirements"
  - CS&S Support to the System Preliminary Design Review (PDR) and CS&S System Critical Design Review (CDR). The PDR and CDR both include exit criteria that is applicable:
    - "Life-cycle software support requirements updated"
  - CS&S Support to the System Verification Review (SVR). The SVR confirms the completeness of the system development and readiness for production and sustainment. Consider including this review prior to entering sustainment. Specific exit criteria are:
    - "required operational and support manuals/documents/electronic data bases, to include software development files, are complete and available"
  - CS&S Support to the System Physical Configuration Audit (PCA). The PCA includes a detailed audit of design documentation, listings and operation and support documents. Exit criteria include:
    - "software support and operational information is completed and verified for accuracy"
- Process Considerations for Safety Critical/High Assurance Systems (Appendix G)
  - This section discusses system supportability and how to obtain it in safety critical / high assurance systems. This includes early architecture definition to include supportability, hardware selection, and software selection/design. Standardization is also key in improving supportability.

Appendix C Example Sustainment Plan Outline

This appendix contains an example sustainment plan outline.

Sources for the Sustainment Plan Outline and Description

The Product Support Management Plan (PSMP[2]) is a life cycle plan that documents the support strategy of a product or a system. The requirement for the PSMP is from AFPD 20-5, *Air Force Product Support Planning and Management*, and is implemented by AFI 63-107, *Integrated Product Support Planning and Assessment*.

Policy and Procedures

The Single Manager (SM) is required to develop the PSMPs for their systems that provide a combat mission or support a warfighter capability. PSMPs are living documents that are required for all programs whether in acquisition or sustainment. SMs will determine the requirement for a PSMP for each of their individual programs, but must, as a minimum, develop a single PSMP for their program or suite of programs to document planned life cycle product support.

The PSMP will place emphasis on life-cycle (cradle-to-grave) sustainment, to include eventual disposal of the item. SMs/PMs will request the support of major stakeholders to participate in the development of product support requirements and strategies to support its implementation. A major modification accomplished as an acquisition program requires an individual PSMP or an update to an existing PSMP. The Merged Acquisition Fielded System Portfolio will be used as a tool to identify programs that must be considered for PSMP development.

The PSMP shall be updated biennially, as a minimum. However, updates shall occur when significant changes occur in the program baseline or acquisition or sustainment strategies and at major milestone decision points. Plans should include roles and responsibilities as well as processes, procedures, and methods. The level of detail included in the sustainment plan should reflect the needs of the system. This should include monitoring the amount of overlap between different documents referenced in this outline to ensure no duplication of effort occurs.

[2] The PSMP format and name was changed to the *Life Cycle Management Plan* (LCMP) March 2005.

Draft Outline

DOCUMENT HISTORY

TABLE OF CONTENTS

I. SYSTEM DESCRIPTION

Current PSMP information should be augmented with the following data*:*

- Reference documents such as JCIDS documents (ICD, CDD, CPD), other life-cycle documents, etc.
- The topic Business Case Analysis should be moved to this section or put in an appendix. It should be expanded or reference documents containing more detailed information.

II. PROGRAM BASELINE

A. Baseline Management – Describe the configuration management and control process for controlling the program baselines. It can be defined here or a reference to another document can be documented.
B. Performance Requirements Baseline Thresholds
C. Program Funding Baseline Thresholds
D. Program Schedule Baseline Thresholds
E. System Technical Baseline – Summarize or reference baselines such as requirements, architecture, design, etc.
F. Risk Baseline – This section describes the risks and their mitigation plans for sustainment.
G. Security Baseline
  - Mission Assurance Category (MAC) and Designated Approval Authority(s) must be established
  - If a Program Protection Plan is required, the reason for the requirement (critical system resource or critical program information) should be carefully explained
  - National Information Assurance Partnership (NIAP) Type 2 packages for COTS products as required by COTS 140-2 should be referenced
  - Restrictions on who can work on the system (e.g., level of clearance required) should be defined
  - Incident handling System Security Authorization Agreement (SSAA) Appendix K from the DITSCAP should be referenced and the process by which this agreement will be implemented described at a minimum

III. SYSTEM SUPPORT STRATEGY

The system support strategies section defines the system sustainment sub-element strategies and plans. Reference the SORAP and other related documents here.

A. Cross-Cutting Sustainment Strategies

This section describes the system sustainment strategies and plans that apply to all components including operations, documentation, staffing, training, security, technology refresh and sustainment environment. Hardware Management and

Maintenance and Software Management and Maintenance are discussed separately. When planning the strategies, make sure that an iterative approach is included for the COTS strategies due to the nature of COTS.

1. Operations Strategy and Plan

   This section addresses the operational strategies and plans for sustainment including help desk, deployment, configuration management, and stakeholders, particularly users.

   a. Help Desk Strategy and Plan
      This section describes how the help desk will be operated during sustainment. It should include a discussion of the problem reporting and resolution process including tier 1, tier 2 and tier 3 operations.

   b. Deployment Strategy and Plan

      This section describes any deployment activities necessary during sustainment. This includes patch deployment mechanisms (security and COTS patches are especially critical).

   c. Configuration Management Strategy and Plan

      This section describes how all components will be controlled and managed during sustainment. This includes the processes as well as the necessary governance structure.

   d. Stakeholder Involvement

      This section describes how the different stakeholders, particularly users, are involved with and participate in formulating the operational strategies. This information could be contained in the other subsections described here.

2. Technology Refresh Strategy and Plan

   This section describes the plans for determining if and when technology needs to be refreshed in the system and how new technologies will be introduced over time. The technologies can be hardware, software, COTS, etc. These plans include a technology strategy, technology roadmap, technology lab, and technology refresh activities.

3. Sustainment Environment (includes facilities, development, test, etc.)

   This section describes the environment necessary to support all sustainment activities for the system including facilities, development environments, test environments, etc. A critical part for success in all of these strategies is a technology integration lab. There must be funding for a lab that contains current system configurations that can be used to investigate and analyze system changes. The lab is useful for technology refresh as well as deployment, maintenance, and training activities.

4. Documentation Strategy and Plan

   This section describes all of the documentation that will be maintained to support sustainment. It needs to discuss how often documents will be updated and who has the responsibility to update the document and approve the updates.

5. Staffing Strategy and Plan

   This section describes the overall staffing requirements to support sustainment. This includes roles and responsibilities of all staff positions and how the positions will be staffed.

6. Training Strategy and Plan

   This section describes the transition and training strategy necessary to ensure that all staff has the skills needed to fulfill their sustainment responsibilities.

7. Security Strategy and Plan
   - This section outlines the processes necessary to maintain Authority to Operate, network connectivity and other security certifications.
   - Issues with sensitive but unclassified information (SBU) should be carefully described.
   - Level of adherence to DISA Gold baseline, waivers, and service unique agreements should be described.
   - Processes for identifying changes that would increase the operational risk (assumed to be low when ATO is issued) should be described.

B. Hardware Sustainment

1. Hardware Management and Maintenance
2. Spares

C. Software Sustainment

1. Custom Software Management Plan – This section references the SDP
2. COTS and Software Reuse Management Plan (CSRMP)

   The CSRMP can be a separate document or can be put in this document as an appendix. Note that the COTS Obsolescence plan is included in the CSRMP. In addition, it will describe the drivers, challenges, risks, and issues facing the project along with the reuse strategies and management and engineering strategies that can mitigate these issues. The roles, responsibilities, and relationships for the key players are defined. Any key artifacts will be described such as Make-Buy Decision, Component Evaluation Record, Reuse Evaluation Analysis Report, Component Lifecycle Plan, Component Health Check Report, Software Reuse Item Database, and Reuse Component Matrix. Any key processes used with COTS/Reusable Software Strategy should be described. Description of the COTS/Reusable Component Lifecycle Management plan should also be addressed. This includes lifecycle planning, risk management, requirements management, license management, problem reporting and management, upgrade management, configuration management, market watch, component tracking, training and support, reuse component provider relationships, reuse component metrics, cost estimation and modeling, and health check.

3. Information Support Plan (ISP)

   This section references the ISP (formerly the C4ISP). The ISP can be a separate document or can be put in this document as an appendix.

Appendix D Acronyms

AFSSIP	Air Force Software Intensive Systems Strategic Improvement Program
ASP	Acquisition Strategy Panel
ATO	Authority to Operate
C4ISP	Command, Control, Communications, Computers, and Intelligence Support Plan
CCB	Configuration Control Board
CDD	Capability Development Document
CDR	Critical Design Review
CLS	Contractor Logistics Support
CPD	Capability Production Document
COTS	Commercial Off-the-Shelf
CM	Configuration Management
CSRMP	COTS and Software Reuse Management Plan
CS&S	Computer Systems and Software
DAA	Designated Approval Authority
DISA	Defense Information Systems Agency
DITSCAP	DoD Information Technology Security Certification & Accreditation
DMI	Depot Maintenance Interservicing
DoD	Department of Defense
DSOR	Depot Source of Repair
FCA	Functional Configuration Audit
GOTS	Government Off-the-Shelf

GSAM	Guidelines for Successful Acquisition and Management
ICD	Interface Control Document
IDD	Initial Design Document
IEEE	Institute of Electrical and Electronics Engineers
IA	Information Assurance
ICD	Initial Capability Document
IMP	Integrated Master Plan
ISO	International Organization for Standardization
ISP	Information Support Plan
IT	Information Technology
JILSP	Joint Integrated Logistics Support Plan
JPO	Joint Program Office
LCMP	Life Cycle Management Plan
LRU	Lowest Replacable Unit
MAC	Mission Assurance Category
MOA	Memo of Agreement
MOT&E	Multi-Service Operational Test & Evaluation
NDS	Non-Developmental Software
NIAP	National Information Assurance Partnership
NSS	National Security Systems
OJT	On-the-Job Training
OPR	Office of Primary Responsibility
PCA	Physical Configuration Audit
PDR	Preliminary Design Review
PSMP	Product Support Management Plan

RFP	Request for Proposal
SBU	Sensitive But Unclassified
SEI	Software Engineering Institute
SM	Single Manager
SOP	Standard Operating Procedures
SOR	Source of Repair
SORAP	Source of Repair Assignment Process
SPO	System Program Office
SRR	System Requirements Review
SSAA	System Security Authorization Agreement
SSDD	System/Subsystem Design Document
SSR	Software Specification Review
ST&E	Security Test and Evaluation
STR	Software Trouble Report
SVR	System Verification Review
TO	Technical Order

Bibliography

URLs are valid as of the publication date of this document.

[Albert 02] Albert, Cecilia; Brownsword, Lisa; Bentley, Col David; Bono, Thomas; Morris, Edwin; & Pruitt, Debora. *Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview* (CMU/SEI-2002-TR-009, ADA405844). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002. http://www.sei.cmu.edu/publications/documents/02.reports/02tr009.html

[Bachmann 03] Bachmann, Felix; Bass, Len; Carney, David; Dietrich, Sven; Feiler, Peter; Garcia, Suzanne; Klein, Mark; Lattanze, Tony; McHugh, John; Meyers, B. Craig; Morris, Ed; Place, Patrick R. H.; Plakosh, Daniel; & Seacord, Robert. *SEI Independent Research and Development Projects* (CMU/SEI-2003-TR-019, ADA418398). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu/publications/documents/03.reports/03tr019.html

[Bass 03] Bass, Len; Clements, Paul; & Kazman, Rick. *Software Architecture in Practice, Second Edition*. Reading, MA: Addison-Wesley, 2003 (ISBN: 0321154959).

[Brownsword 00] Brownsword, Lisa & Place, Patrick. *Lessons Learned Applying Commercial Off-the-Shelf Products* (CMU/SEI-99-TN-015, ADA379746). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. http://www.sei.cmu.edu/publications/documents/99.reports/99tn015/99tn015abstract.html

[DSMC 97] Defense Systems Management College. *Acquisition Logistics Guide*. http://www.dau.mil/pubs/pdf/alg1.pdf (1997).

[DoAF 01] Department of the Air Force. *Air Force Policy Directive 20-5 Air Force Product Support Planning and Management.* http://www.e-publishing.af.mil/pubfiles/af/20/afpd20-5/afpd20-5.pdf (2001).

[DoAF 03] Department of the Air Force. *Guidelines for Successful Acquisition and Management of Software-Intensive Systems: Weapon Systems, Command and Control Systems, Management Information Systems–Condensed Version 4.0.* http://www.stsc.hill.af.mil/resources/tech_docs/gsam4.html (2003).

[DoAF 04] Department of the Air Force. *Air Force Instruction 63-107 Integrated Product Support Planning and Assessment*. http://www.e-publishing.af.mil/pubfiles/af/63/afi63-107 /afi63-107.pdf (2004).

[DoD 88a] Department of Defense. *DOD-STD-2167A Defense Systems Software Development*. February 1988. (Cancelled December 1994)

[DoD 88b] Department of Defense. *DOD-STD-7935A DOD Automated Information Systems (AIS) Documentation Standards*. October 1988. (Cancelled December 1994)

[DoD 90] Department of Defense. *MIL-HDBK-347 Mission-Critical Computer Resources Software Support*. May 1990.

[DoD 94] Department of Defense. *MIL-STD-498 Software Development and Documentation.* December 1994. (Cancelled June 1998)

[DoD 97] Department of Defense. *DoD Instruction DoD Information Technology Security Certification and Accreditation Process (DITSCAP) (DoDI 5200.40)*. December 1997.

[DoD 03a] Department of Defense. *DoD Instruction Operation of the Defense Acquisition System (DoDI 5000.2)*. May 2003.

[DoD 03b] Department of Defense. *DoD Directive The Defense Acquisition System (DoD 5000.1)*. May 2003.

[DoD 04] Department of Defense. *National Security Space Acquisition Policy (Number 03-01)*. December 2004.

[IEEE 90a] Institute of Electrical and Electronics Engineers. *IEEE Standard Glossary of Software Engineering Terminology* (IEEE Std. 610.12-1990). New York, NY: IEEE, 1990 (ISBN: 155937067X).

[IEEE 90b] Institute of Electrical and Electronics Engineers. *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. New York, NY: IEEE, 1990.

[IEEE 98] Institute of Electrical and Electronics Engineers/Electronic Industries Alliance. *Standard for Information Technology – Software Life Cycle Processes* (IEEE/EIA 12207.0-1996). New York, NY: IEEE Computer Society Press, 1998.

[ISO/IEC 04] International Organization for Standardization/International Electrotechnical Commission. *ISO/IEC 90003:2004, Software engineering – Guidelines for the application of ISO 9001:2000 to computer software*. 2004.

[JCS 00] Joint Chiefs of Staff. Ch. 1, "Authorities and Responsibilities for Logistic Operations," Section 4, Paragraph c. *Doctrine for Logistic Support of Joint Operations*. http://www.dtic.mil/doctrine/jel/new_pubs/jp4_0.pdf (2000).

[OUSD 04] Office of the Under Secretary of Defense for Acquisition, Logistics & Technology. *Product Support Boundaries Memo dtd 23 SEP 04.* https://acc.dau.mil/simplify /ev_en.php?ID=54169_201&ID2=DO_TOPIC (2004).

[SAF/AQ 04] Secretary of the Air Force, Acquisition. *Memorandum 04A-003 Revitalizing the Software Aspects of Systems Engineering.* https://www.safaq.hq.af.mil/contracting/affars/5334/mandatory /aq-memo-20sep04.pdf (2004).

[Seacord 03a] Seacord, Robert C.; Plakosh, Daniel; & Lewis, Grace A. *Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices*. Boston, MA: Addison-Wesley, 2003 (ISBN: 0321118847).

[Seacord 03b] Seacord, Robert C.; Elm, Joseph; Goethert, Wolf; Lewis, Grace A.; Plakosh, Daniel; Robert, John; Wrage, Lutz; & Lindvall, Mikael. "Measuring Software Sustainability," 450–461. *Proceedings of the 19th International Conference on Software Maintenance.* Amsterdam, Netherlands, September 22–26, 2003. New York, NY: IEEE Computer Society Press, 2003.

[SMC 05] Space and Missile Systems Center. *SMC Depot Maintenance – Source of Repair Assignment Process.* http://ax.losangeles.af.mil/axl/sorap.htm (2005).

[USAF 04] U.S. Air Force. *Software Management Guidebook, Version 0.9.* https://acc.dau.mil/simplify /ev.php?ID=61457_201&ID2=DO_TOPIC (2004).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	May 2006	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Sustaining Software-Intensive Systems	FA8721-05-C-0003

6. AUTHOR(S)

Mary Ann Lapham

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2006-TN-007

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

As today's systems become increasingly reliant on software, the issues surrounding sustainment become increasingly complex. The risks of ignoring these issues can potentially undermine the stability, enhancement, and longevity of fielded systems. Questions about sustaining new and legacy systems include

1. What does it mean to perform sustainment from a software perspective?
2. What types of development and acquisition activities are required to sustain software-intensive systems?
3. Although the Department of Defense (DoD) has a technical definition of sustainment, does the DoD typically consider sustainment as maintenance?
4. How does the increased use of commercial-off-the-shelf software complicate sustainment?

This technical note discusses these questions and presents definitions, related issues, future considerations, and recommendations for sustaining software-intensive systems. Sustainment done well leads to well-supported software-intensive systems and reduced total ownership costs and should help organizations meet current and new mission area and capabilities requirements.

The information contained in this technical note is based on information that the Software Engineering Institute gathered during work with Air Force software-intensive systems. While the information is pertinent and can be applied to systems in the commercial sector, keep in mind minimal effort was made to convert "DoDspeak" into commercial sector language.

14. SUBJECT TERMS	15. NUMBER OF PAGES
acquisition, commercial off-the-shelf, COTS, database, documentation, integration, operation, sustainment, software architecture, software-intensive systems, security, spiral development, test, training, regression testing, risk, risk management	52

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL