AD-A260 109

Jul 15, 92 FINAL

DoD Software Reuse Initiative, Vision and Strategy

Ms Linda Brown, 703-746-7928

OASD(C3I)/DDI
1225 Jefferson Davis Highway, Suite 910
Arlington, VA 22202-4301

Same as 7 above

Distribution Statement A:
Approved for Public Release; Distribution is Unlimited.

This document describes the vision and strategy for a DoD software reuse initiative which will make a reuse-based paradigm the preferred alternative for developing and supporting software. The strategy applies to all types of software-intensive systems managed by the Department of Defense.

DTIC
ELECTE
FEB 12 1993
S B D

Software reuse, domain, reusable component, product line, domain analysis, architecture, reuse metrics, asset certification.

UNCLASSIFIED UNCLASSIFIED UNCLASSIFIED UL

DoD

Software Reuse Initiative

Vision and Strategy

July 15, 1992

93-02823

TABLE OF CONTENTS

DTIC QUALITY INSPECTED 3

Accession For
NTIS GRA&I ☑
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1

EXECUTIVE SUMMARY

Premise The Department of Defense has evidence that software reuse principles, when integrated into acquisition practices and software engineering processes, provide a basis for dramatic improvement in the way software-intensive systems are developed and supported over their life-cycle. This document describes the vision and strategy for a DoD initiative which will make a reuse-based paradigm the preferred alternative for developing and supporting software. The strategy applies to all types of software-intensive systems managed by the Department: Information, Command and Control, and Weapon Systems.

Goals The specific goals of the initiative are to:

- Improve the quality and reliability of software-intensive systems,
- Provide earlier identification and improved management of software technical risk,
- Shorten system development and maintenance time, and
- Increase effective productivity through better utilization and leverage of the software industry.

Vision To drive the DoD software community from its current "re-invent the software" cycle to a process-driven, domain-specific, architecture-centric, library-based way of constructing software.

Concepts The underlying concept for the vision is to create a business environment where families of related systems in a domain are designed so they share a common structure. There are four fundamental principles intrinsic to this concept:

- Focus on reuse in specific domains and exploit those domains to support reuse-in-the-large,
- Ensure that reuse is treated as an inseparable part of software engineering,
- Employ reuse-oriented architectures to spur investment in components to populate the architectures, and
- Utilize an interconnected network of reuse libraries to drive the capture, storage and reuse of components within and across domains.

Strategy The strategy to realize this vision is based on systematic reuse: where opportunities are predefined and a process for capitalizing on those opportunities is specified. There are ten elements of this strategy:

- Specify the domains where reuse opportunities exist and identify criteria to prioritize, qualify and select domains for application of reuse techniques;
- Define the types of products suitable for reuse and develop criteria to validate these components for new applications;
- Determine what ownership criteria pertain to these components and require conscious decisions regarding their ownership;
- Modify the current acquisition process so that reuse is integrated into each phase of the acquisition process and into the overall system/software life-cycle;
- To guide business decisions, define models which may suggest novel strategies and require tailored acquisition approaches to support reuse;
- Establish procedures to collect metrics which: (1) measure the payoff from the reuse initiative and (2) aid developers in the selection of reusable components;
- Define standards for the various types of components which will permit their certification for reuse;
- Pursue a technology-based investment strategy which identifies, tracks, and transitions appropriate reuse-oriented process and product technologies;
- Conduct comprehensive training to ensure that practitioners and policy makers capitalize on the initiative; and
- Exploit near-term products and services which facilitate movement to a reuse-based paradigm.

Summation This *DoD Software Reuse Vision and Strategy* describes software reuse as a powerful concept. The value of this approach has been demonstrated in many other engineering disciplines where the use of standard concepts, processes, and components allows prior accomplishments to be leveraged and speeds innovation for future systems. At this point, many different techniques which support reuse are being tried. This document provides the focus under which these efforts will continue, lessons-learned will be collected, and experiences shared. Taken together, the ten strategic steps will bring about the cultural change necessary to make software reuse effective for the DoD.

1. BACKGROUND

The Department of Defense (DoD) believes that reuse principles, when integrated into its acquisition practices and software engineering process, provide a basis for dramatic improvement in the way software-intensive systems are developed and maintained over their life-cycle. Therefore, the Department has elected to accelerate the incorporation of software reuse into its business strategy. The reasons for doing so are to: increase productivity, improve the quality and reliability of software-intensive systems, enhance system interoperability, identify and manage technical risk, and shorten development and maintenance time. This document describes the vision and strategy for a focused, cooperative reuse effort that will result in software reuse being pursued in a consistent, coordinated manner throughout the Department. It will be used as the basis for an integrated reuse initiative within the Department. The strategy outlined herein applies to all classes of software-intensive systems managed by the Department, including those characterized as:

- Information Systems
- Command and Control Systems, and
- Weapon Systems.

The notion of software reuse is not new. The concept of treating software elements as black box components and composing systems based on well-understood architectures which use those components was first proposed almost twenty-five years ago. It is only recently, however, that significant progress is being made in this area. Companies are discovering that a reuse-based business strategy improves their competitive and profit base.

Software reuse will eventually happen whether the DoD takes an active role or not. The challenge is to position the DoD to accelerate its use and to reap its benefits. The Department must make a careful and deliberate analysis of the business, technical, and developmental context in which a DoD software reuse strategy will be implemented. In particular, within this context it needs to consider the infrastructure investment which must be made to permit effective reuse and establish those principles central to its effort. The types of infrastructure investment include: advancing technologies that support reuse, incorporating reuse into management and engineering processes, and creating generic sets of components to (re)use in new systems or in software maintenance. The following considerations must be taken into account:

- The objective is systematic (planned), not opportunistic, reuse.
- There is no singular approach to software reuse.
- Libraries facilitate, but do not enable, reuse.
- Reuse is a process, not an end-product.
- Domain analysis/models/architectures are the primary focus.

- Near term cost savings will be offset by infrastructure investments.
- Ada provides a foundation upon which to base reuse efforts.

2. DOD REUSE VISION

At the highest level, the DoD vision for reuse is to drive the DoD software community from its current "re-invent the software" cycle to a process-driven, domain-specific, architecture-centric, library-based way of constructing software. Within this vision, planned reuse becomes an essential facet of each phase of the software development life-cycle. This section highlights the principles, organization and assumptions that will be critical factors in implementing this vision within DoD.

2.1 Terminology

- **Reusable Component**
  A representation of some aspect of a system which may be used in different applications. A component may consist of requirements, architectures, design, or implementation (e.g., code, tests, documentation) information.

- **Software Reuse**
  The application of a reusable component to more than one application system. Reuse may occur within a system (e.g., F-14A, B, ...), across similar systems (e.g., M1, Bradley, ...), or in widely different systems (e.g., user interface services).

- **Domain**
  The functional area covered by a family of systems (e.g., the aircraft navigation system domain) or across systems where similar software requirements exist.

- **Domain Analysis**
  A study which identifies the similarities and differences among related systems within a domain.

2.2 Basic Principles

The underlying concept for all DoD software reuse activities is to create an environment where families of related systems in a domain are designed so they share a common structure (architecture). When populating these architectures by using an appropriate mix of existing, tailored, generated, and new components becomes the predominant way to build and maintain systems, the DoD will start to enjoy the benefits of software reuse identified previously. At that point, the DoD can take advantage of the flexible nature of software to improve the performance of its systems.

The DoD's near-term investment strategy must identify and capitalize on areas where substantial reuse is possible; its long-term strategy must lead to the creation of a true "black box" components' industry. (The term "black box" indicates that while the functionality of a

component must be known, the internal workings of that component need not be understood. Seen another way, *what* the component does is important, while *how* it does it is not. This concept of "black box" components implies a library of interchangeable parts that can be tied together to create new systems). This involves: experimenting, capturing experience, setting the policy and procedures, and establishing the organizational structure and mechanisms which support a reuse-based software engineering process. There are four fundamental principles intrinsic to this concept:

- **Domain Specific Reuse**
  Support the analysis of problems and systems in specific domains leading to the definition of "conventional" architecture concepts for these domains. Exploit these domains to support reuse-in-the-large by building domain-specific tooling, such as program generators, very high-level languages, knowledge-based approaches, and domain-specific reusable component parts.

- **Process Driven Reuse**
  Ensure that reuse is treated as an inseparable part of software engineering. Software reuse must evolve into an integral and transparent part of both the software engineering process and the broader acquisition (procurement) context in which that process occurs.

- **Architecture-Centric Investment**
  Employ reuse-oriented flexible architectures for DoD domains which are well supported by industry and the R&D community, then spur investment in creation of generic software components and tooling which facilitates development of systems complying with approved architectures. The creation of generic components must be independent of development of fieldable production systems. One of the principal challenges of reuse is to develop processes and standards that can facilitate development of the convention that enables effective sharing of components. The goal is to achieve "black box" reuse, where reverse engineering is replaced by examination of a specification that is supported by conformance analysis and validation.

- **Interconnected Reuse Libraries**
  Provide the ability to locate and share reusable components across domains and among the services. Utilize evolving technology to provide a network of interconnected reuse library systems to support capture, storage and reuse of components within and across specific domains.

2.3 Assumptions

- Software reuse is a means to an end -- to reduce risk, improve quality and reliability, shorten development and maintenance time, adapt to new requirements and changing technology, and improve productivity. Although one goal is to reduce life-cycle cost, a significant up-front investment in an infrastructure supporting reuse is necessary.

- There are both technical and non-technical barriers to reuse. The nature of these barriers varies with the specific reuse strategy chosen for a particular class of systems,

or even for elements of the architecture within those systems.

- Software-intensive system development will evolve to use a process-driven, domain-specific, reuse-based, technology-supported model.

3. STRATEGY

DoD reuse strategy will capitalize on systematic reuse, where opportunities are predefined and a process for applying those opportunities is fully specified. It includes the following key elements:

3.1 Establish Domains

Because payoff from a reuse-based software process increases with properly structured and defined domains, the initial and most important step in a DoD reuse strategy will be to properly define those domain boundaries. Although a domain may be bounded by the knowledge and concepts that pertain to a particular computer application area (such as logistics, tactical command and control, or avionics), the DoD will focus on "areas of business" for its initial domain decomposition. These domains may later be decomposed into subdomains to support a specific purpose or goal of a particular domain, or integrated vertically or horizontally where reuse opportunities are identified. Initially, however, the DoD will require business managers to establish plans to manage reuse across their systems. In most cases, these business managers will be Program Executive Officers.

Prior to selecting domains to pursue, the DoD must have a defined procedure to evaluate domains for reuse potential and to qualify them for further work. The following are examples of the types of questions that must be answered:

- Do well defined software processes exist in the domain?
- Is there a sufficient knowledge base to support the domain analysis?
- How will the products of the domain analysis be used (i.e., do they meet the business objectives established)?
- Is the domain under consideration mature or stable?
- How can the domain take advantage of the industrial base and how can its future direction be influenced?
- Will the domain form a key part of an organization's future "business"?

The criteria for domain selection do not currently exist and must be established.

Because of the evolving nature of reuse-based software development and the limited number of experts in the area, the DoD will prototype and evaluate reuse techniques and strategies in selected domains before making wholesale changes to its way of doing business. Potential

domains will be analyzed based on the qualification procedures noted above. Of particular concern is the stability and maturity of the domain. Domains will be prioritized according to their potential for supporting systematic reuse. Selected domain managers (i.e., PEOs) will be provided assistance in analyzing their domains and establishing business plans. Then a continuous process improvement procedure must be established to feedback the results of research, pilot projects, and component usage.

3.2 Define Reuse Products

It is essential that domain managers use domain analysis techniques to identify the information needed for reuse. This includes identifying the components that have greatest utility within an application area and providing guidance on how to adapt them if they do not meet needs exactly. Many approaches for domain analysis are starting to appear. The DoD must define criteria for comparing various domain analysis approaches, identifying the conditions under which different techniques are appropriate, and describing the precise products that are produced and managed.

Since reuse may occur with many different components, each type of component discussed below is a candidate for reuse and has a definite but currently undefined representation which supports reuse.

- The **domain model** describes the problems within the domain that are addressed by software. The domain model identifies the generic requirements, represents the formal definition of the domain, and provides the general rules and principles for operating within the domain. It indicates the boundaries of the domain, the primary inputs and outputs, and the standard vocabulary used for the domain.

- A **software architecture** implements solutions to the problems in the domain. It becomes a model for constructing applications and mapping requirements from the domain model to reusable components. A generic architecture provides a high-level generic design for a family of related applications as well as a set of components intended to be reused for any instance of that application. The generic design eliminates the need to develop a high-level design for each application within the domain. As a result, domain developers use these representations as specifications for reusable components.

- **Product design**, which is derived from the specification of the architecture, describes the relationship between the domain model and the work products; it is used to develop reusable components and build systems from such components.

- The types of **implementation components** which can be reused include: specifications, detailed designs, code, tests, and documentation. In reusing code, actual code is taken from one application and reused "as is" or modified for use in another application regardless of the system design. Specification reuse is currently being performed, but on a limited basis. Reuse can also be based on generic architectures and the generation of software specifically to satisfy such architecture.

3.3 Establish Criteria for Deciding Ownership

Current acquisition policy indicates that the Government will obtain unlimited rights (or close to unlimited in the case of mixed funding) to most software products. The only exceptions are commercial software, and unpublished software and related documentation in existence at the time of contract award. Yet part of a DoD reuse strategy should be to encourage the development of a "black box" software components industry. The DoD wants contractors to invest funds in development efforts with the intent of later capturing greater market share, or gaining a technical and/or monetary advantage. By denying rights to such contractors, the Government is indirectly stifling the growth of a reusable software components industry, discouraging widespread competition for DoD contracts (since many contractors are unwilling to sacrifice their rights to developed software products), and inhibiting the supply of trusted, reusable software products at a time when demand for such products is rapidly escalating.

The DoD must determine how it will change acquisition policy to stimulate creation of reusable components. This includes establishing procedures which allow for a reasonable return on investment for those components created by industry in response to DoD's need to populate software architectures. It also requires that each domain manager make a conscious decision as to the type of components owned and managed by the Government. Each domain's reuse business strategy will identify the level at which Government ownership is prudent: requirements (what it does), architecture (how it works), design (how it is built), or implementation (what is built). This decision will vary by domain and may vary within a domain. It will be driven by the technical factors, the potential market for a particular component, the health of the commercial marketplace, and the acquisition strategy within the domain.

When products are commercially viable, this explicit and basic ownership decision will be used to implement a policy of extending the maximum rights to contractors for domain models, architecture documentation, as well as software design and products and their documentation. As the software components industry grows, it is expected to become the norm for the Government to define the specific requirements and, perhaps, the architecture, then enjoy a free and open competition for the remainder. As an alternative, the Government could elect to allow contractors to retain rights for the duration of the contract; in the event that the contractor did not commercialize within a stated period of time after contract completion, ownership would revert back to the Government. At the very least, the use of Government Purpose Rights should be pursued in lieu of unlimited rights.

3.4 Integrate Reuse into the Development and Maintenance Process

Software reuse must be integrated into the entire life-cycle process, so that all possible software reuse opportunities in the software development life cycle can be utilized. This includes, for example, evaluating architectures before selecting an acquisition strategy, using off-the-shelf components for construction of evaluation prototypes to negotiate requirements, and enforcing standards at the time of Critical Design Review to ensure they will be reusable on future acquisitions. Therefore, reuse must be considered an integral part of the software and system development processes, which, in turn, are conducted within the context and constraints of the DoD acquisition process. To achieve the goal of widespread reuse, reuse considerations must be integrated into the acquisition life cycle for systems as outlined in DoD Directive 5000.1

(Defense Acquisition), DoD Instruction 5000.2 (Defense Acquisition Management Policies and Procedures), and DoD Directive 7920.1 (Life-Cycle Management of Automated Information Systems). Reuse must also be incorporated into the successors to DoD Standards 2167/7935, the Military Standards governing defense system software development.

Software reuse must be systematically evaluated during the examination of alternative concepts at the inception of a software development project. At specific critical junctures during the acquisition life-cycle, reuse must be re-evaluated and/or incorporated to attain the following goals: accelerate system development and deployment, reduce overall life-cycle costs, improve reliability, and provide well-structured components to serve as the basis for future system maintenance.

Reuse considerations must be incorporated into the contractual process, ranging from planning the acquisition strategy prior to contract award, through the entire contractual management process to contract completion. Reuse must be considered when an Acquisition Strategy Panel is convened, and throughout the Request for Proposal process. Guidelines need to be developed and provided to assist in evaluating proposals and the reusability of a system's components. Training and guidance relative to reuse should address the following issues: types of personnel needed for reuse projects, software ownership rights that pertain, and types of contracts appropriate for software development incorporating reuse.

3.5 Define Model for Business Decisions

In order to create an acquisition environment within which software reuse and commercial software integration can flourish, business incentives must be designed to reward software developers who make reusability a design constraint. In addition, regulatory changes must be undertaken and guidelines established which use the acquisition process to stimulate the development of reusable software components. The DoD recognizes that the specific barriers which may inhibit reuse will vary, depending on the model of ownership selected for a given domain. The DoD will identify those business decisions which have the greatest impact on the emergence of a reuse-based software process and components industry and are driven by the domain "ownership decisions." It will identify specific strategies which should be considered in each case. For example, if it is decided to rely on commercially available (COTS) components to populate parts of a domain architecture, a key strategy will be the protection of the Government's interests, in the event the supplier goes out of business. On the other hand, if the Government decides to sponsor development of and own those same components, the issue may focus on the impact of patents pending at the time of initial development.

The DoD must use suitable cost and business models for identifying proper business decisions which implement reuse. Several models are available for use in estimating costs relative to software development efforts. The selection of particular costing methodologies is dependent upon many factors, which include: the extent of reuse to be incorporated; whether development of reusable components is anticipated; the point within the development life-cycle at which cost estimating is undertaken; the software engineering approach to be undertaken; and, the language in which development will occur. Therefore, the DoD will provide guidelines which can be used to identify the particular amalgam of unique requirements associated with a reuse-based development effort to determine which costing methodology or methodologies should be utilized.

3.6 Define Metrics to Evaluate Reuse Success

Measurements for software productivity, in general, lack standardization. Measurements for gauging reuse effectiveness and efficiency are even less mature. The DoD must establish a baseline upon which to gauge success and measure improvement that serves as a basis for comparison among alternative approaches. Metrics to evaluate reuse success will become an integral part of a software measurement process. To effectively manage and control software development and to integrate the software measurement metrics into daily program management activities, the DoD will establish procedures and guidelines to:

- Collect measurements that are targeted to specific program issues and that will aid in understanding and improving the technical and management processes;
- Use the metrics together with other management information to improve insight into process and risks;
- Establish means to further investigate both management and technical issues; and
- Identify and implement changes to improve the program.

Two classes of reuse success metrics will be defined: technical and managerial. In the near-term, management primarily needs to know if reuse resulted in improved productivity and reduced risk, while more powerful predictive metrics are needed in the long-term. Engineers need to focus on specific asset performance, such as reliability, usability, portability, security, user/implementation documentation, configuration control, and adaptability. General standards for the application of software metrics to reuse are necessary.

3.7 Define Component Guidelines for Different Reuse Products

Guidelines need to be set for the different reuse products discussed in Section 3.2. Although the content and structure of the domain model and domain architecture are determined by the domain itself, how these are represented in a model should be similar to increase sharing of knowledge among domain-specific applications.

There are two aspects of guidelines related to components; guidelines that outline (1) design characteristics, and (2) evaluation criteria for certifying components.

Although some guidelines exist for developing reusable components throughout the software life-cycle, they must be standardized. These guidelines should not be so restrictive as to diminish the creativity of software developers, but should be goal-oriented. Some design goals include: generality, modularity, quality documentation, reliability, adaptability/flexibility, completeness of requirements and designs, application independence, extendibility/augmentability, performance/ efficiency, portability, robustness/fault tolerance, understandability/clarity, independence from machine/compiler/operating system, reusability, and extensibility.

Various evaluation criteria for certifying components for reuse exist in both industry and the DoD. Existing criteria should be evaluated, modified if necessary, and adopted for use. Since

reuse can be accomplished at various levels (such as requirements, architecture, design, and code), the criteria must also apply to all levels and not just to code. Some examples of certification criteria include:

- Have requirements been validated in an operational system?
- Do architectures support easy distribution across hardware elements?
- Does the component include documentation to enable modification (e.g., Program Design Language (PDL))?
- If code, does it follow any endorsed coding guidelines?
- Does the component achieve metrics standards for reusability, complexity, and portability?
- Is there documented evidence of successful, frequent reuse?
- Is the component warranted by some organization?
- Are there any disclaimers attached to the component?
- Is the component submitted with test plans, procedures, and results?

One possible plan is to implement a method of multiple levels of certification for components. A multi-level certification process will allow more components to be reused, depending upon the application. The highest level might be: reviewed, approved, complies with standards, contains documentation and test materials, meets requirements and has been cleared for security purposes.

3.8 Identify Technology Base Investment Strategy

In order to implement domain analysis and software reuse within the DoD, various technologies and methodologies must be developed and implemented for operational use. Methodologies to represent a domain's knowledge, software development tools that apply specifically to software reuse, and multi-level security features must be developed. The DoD will institute a formal process to identify and track reuse-related research and development. The Department will leverage off the industrial research and development base to the maximum extent but, in some cases, will need to push the state-of-the-art to see its vision of software reuse implemented. In most cases, however, the Department will need to focus its limited resources on identifying promising technologies and reducing them to practice. Whatever the source of technology, DoD must place a major emphasis on technology transition. A sampling of the areas where an active DoD presence is necessary include:

- **Tools**
  Many different software development tools (such as requirements analysis and design, software documentation, and software test tools) exist to assist during the various stages of the software development life cycle. However, many of these tools do not

take software reuse into consideration. Software development tools that consider software reuse must be developed to ensure the benefits of reuse. Tools exist that generate actual code based upon user-defined specifications. Some of these tools may include capabilities to either prototype the user interface or simulate the system's constraints in addition to generating code. Tools are needed to support both domain analysis and the process of utilizing domain analysis products to develop reusable components and reuse existing components. These tools are necessary to aid in understanding the volume and complexity of information gathered during domain analysis and the presentations of that information in specific domain analysis products.

Selected software development tools should also support evaluation criteria for certifying components. To do this, the component certification criteria must direct the requirements and specifications of software development tools. The resulting software components would automatically meet the evaluation criteria developed.

- **Knowledge Representation**
  Methodologies and tools to represent and access the knowledge captured during domain analysis must be specified to assist software developers in efficiently and effectively reusing software components. Specifically needed are notations to describe architectures, and tools to manipulate these notations. A knowledge base representation of a domain can support software developers in determining constraints and the relationships among components.

- **Information Systems Security.**
  The reuse initiative in the DoD has important information systems security concerns that must be addressed up front and throughout this initiative. The paramount concern is for the establishment and maintenance of the integrity of reuse components. The most effective means for establishing integrity in reuse components is through the development process. However, much of the initial body of reusable components will require post-development evaluation to establish integrity levels. Once components are accepted and stored in a reuse repository, effective mechanisms must be in place to maintain the integrity of the reuse components. The distribution of reusable components must also ensure the integrity of the components retrieved from the reuse repository.

  There is also the need within DoD to address the confidentiality concerns in reuse. Reuse infrastructures must provide the ability for projects running at classified levels (e.g.Secret) to access all pertinent reusable components (that meet or exceed a specified integrity level) at the same level or below (i.e. Secret, Confidential, Unclassified). Inability to provide multi-level repositories will result in fragmented reuse with islands of reuse technology bounded by classification levels.

3.9 Education and Training

Education and training for all parties involved in reuse is absolutely critical. Since reuse of software represents a radical transformation in present software engineering practices, and signifies a dramatic reversal of deeply-ingrained beliefs regarding the development of software, much education is required to effect a marked reversal in attitudes.

Several cultural factors can be expected to directly impact the assimilation of new technology and a novel approach to software engineering practices. These include the following: lack of formal instruction in reuse (not part of most engineering curricula), resulting in the perception of reuse as an alien concept; the existence of the "Not Invented Here" syndrome, which leads individuals to distrust outside products; the lack of upper management support (or more significantly, the lack of a powerful champion/reuse advocate) for pilot projects intended to demonstrate the success and cost-effectiveness of reuse; typical resistance to change; and the view that reuse runs counter to creativity. Since the magnitude of change is anticipated to be significant, organizational difficulties may constitute the most critical factor in impeding the success of reuse initiatives.

Training must be undertaken on several levels, gradually filtering down to the lower ranks. Managers within Government organizations must take the initiative to influence the adoption of reuse within their organizations and within the contracts which they administer. In addition, acquisition and contracting executives must be educated regarding the application of current acquisition regulations and the effect of recent modifications on future reuse activities. They should also receive instruction on DoD policies that encourage reuse, and must also be trained regarding the use of current regulations to foster the expansion of reuse throughout the DoD.

Much groundwork must also be laid to promote the creation of an internal environment which is receptive to the integration of software reuse. Education must take place within the work place, as well as through academic institutions. Management within DoD and contractor organizational layers must institute policies and procedures to support a change process; therefore, guidance must be provided to those executives unfamiliar with reuse to ensure its future success. Upper-level management must also advocate the institution of reuse processes, and must offer a visible presence in support of such initiatives in order to effect a reversal of engineers' negative perceptions. Change should be planned and gradual, to reduce the impression of threat to the status quo, thereby enabling a shift in attitudes and beliefs.

The business practices environment functions as the essential foundation for the reuse infrastructure. Without adequate efforts to address and resolve the issues that constitute this foundation for reuse, the entire reuse structure could be undermined, resulting in failure of widespread reuse and seriously impaired acceptance of major technological change. Although many impediments to reuse currently exist, several tools (e.g., guidebooks, technology transition support, training and regulatory change) could be developed to counterbalance these barriers. Critical objectives would then be realized: creation of a regulatory environment that fosters reuse; elimination of the effects of negative attitudes and perceptions; and confrontation of organizational issues prior to instituting change. Ultimately, resolution of these business practice issues will lay the groundwork for ready acceptance of new development methodologies.

3.10 Provide Near-Term Products and Services

As organizations attempt to move their software engineering process to one supported by extensive reuse, they need to follow a balanced approach. Attempts to foster higher levels of reuse within organizations have often floundered because they only dealt with a subset of the problem. A reuse maturity model has been proposed.[1] It is loosely based on the Software Engineering Institute's Capability Maturity Model (CMM) and prescribes five levels of software reuse maturity: Initial/Chaotic, Monitored, Coordinated, Planned, and Ingrained.

At the lower levels, organizations take an ad hoc view of software parts; while at the higher levels, they achieve a strategic exploitation of software components. This staged model postulates that all the conditions at one level of reuse maturity must be met to provide the foundation for the next. The model suggests the logical next steps for an organization depend, in large measure, on where they are today. While the DoD needs to put a process and infrastructure supporting reuse in place, the products and services offered to each organization ought to be tailored to their needs and capabilities. The DoD should help define and validate a software reuse maturity model to focus organizational efforts appropriately along the following dimensions of maturity:

- Motivation/Culture,
- Planning for reuse,
- Breadth of reuse involvement,
- Responsibility for making reuse happen,
- Process by which reuse is leveraged,
- Reuse inventory,
- Classification activity,
- Technology support,
- Metrics, and
- Legal, contractual, and accounting considerations.

---

[1] Phil Kolton and Anita Hudson, "A Software Reuse Maturity Model", The Fourth Annual Software Technology Conference, USAF STSC, Salt Lake City, Utah, 13-17 April 1992.

4. MANAGEMENT STRUCTURE

This vision and strategy will be used as the basis for an integrated reuse initiative within the Department. Recognizing that organizations play different roles in making reuse a reality (i.e., providing infrastructure, investing in a component base, building systems from those components) and that many different approaches to reuse currently are being tried, the initiative will be managed as a coordinated, integrated effort among individual programs and cognizant organizations. To this end the Department is establishing a DoD Software Reuse Executive Steering Committee to oversee the software reuse initiative. The Executive Steering Committee will provide focus for the initiative and guide it as the necessary cultural changes occurs, technology is developed, and a reuse-based paradigm is institutionalized. The Executive Steering Committee will be supported by working groups that will address the various technical and managerial topics and issues raised by the strategies, and recommend approaches or solutions. Within the context of this Department-wide initiative, individual programs should be allowed to continue, lessons-learned should be collected, and experiences, shared.

5. CONCLUSION

Software reuse is a powerful concept. The value of this approach is demonstrated in many other engineering disciplines where the use of standard concepts, processes, and components allows prior accomplishments to be leveraged and speeds innovation for future systems. The DoD recognizes the potential benefits form this approach and is establishing its own corporate strategy to apply leverage and capitalize on software reuse.