SAE Aerospace
An SAE International Group

AEROSPACE RECOMMENDED PRACTICE

SAE ARP9005
Issued 2005-06

Aerospace Guidance for Non-Deliverable Software

FOREWORD

Non-deliverable software is defined as software used in the design, manufacture, inspection, test, acceptance, or calibration of a deliverable product, and is not generally delivered under a contract.

As Industry efforts increase to improve product quality and reliability and reduce production costs, use of computer software programs for automation, control and monitoring of production processes and product test, acceptance and calibration is also increasing. It is therefore vital to assure these software programs are controlled to ensure product conformity requirements are properly supported.

The intent of this document is to provide industry recommended guidelines for non-deliverable software that are focused on its control and ensuring validation of released software in its functioning environment, appropriate configuration control of non-deliverable software through application of unique identification, and retention of relevant records for internally developed software documenting its development and validation prior to application or use.

TO PLACE A DOCUMENT ORDER:
Tel: 877-606-7323 (inside USA and Canada)
Tel: 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: custsvc@sae.org
SAE WEB ADDRESS: **http://www.sae.org**

SAE 100 Leading Our World In Motion

TABLE OF CONTENTS

INTRODUCTION

General

This SAE Aerospace Recommended Practice (ARP) document defines recommendations for the use of non-deliverable software in the Aerospace Industry based on best practices and effective control methodology.

Process Approach

This document deals with the subject of non-deliverable software as a series, or system of processes, linked and managed to bring effective control to the varied aspects and types of software utilized in supporting and enabling production. This linked series or system of processes is referred to as the “Process Approach” and is utilized in developing the control methodology for non-deliverable software.

1. SCOPE:

General:

This document contains recommended practices for the effective control of non-deliverable software. It addresses practices for control during the development, production, release maintenance, and retirement of non-deliverable software, as well as for software procured from outside manufacturers and incorporated in the production, evaluation, test, acceptance or calibration of processes.

For the purposes of this document, the terms software and non-deliverable software are considered synonymous.

Application:

The guidelines in this ARP apply to non-deliverable software that:

- directly relates to design, manufacture, inspection, test or calibration of a deliverable product, and
- directly affects the configuration, conformity or quality of a deliverable product.

The following are not within scope of this ARP:

- deliverable software (reference AS9006),
- business systems or office software,
- information systems software for business applications,
- prototype software that is not intended for delivery, and
- software that only enables the operations of other software or devices in the performance of required tasks that do not affect product conformity.

2. REFERENCES:

The following publications form a part of this document to the extent specified herein. The latest issue of SAE publications shall apply. The applicable issue of other publications shall be the issue in effect on the date of the purchase order. In the event of conflict between the text of this document and references cited herein, the text of the document takes precedence. Nothing in this document, however, supersedes applicable laws and regulations unless a specific exemption has been obtained.

2.1 SAE Publications:

Available from SAE, 400 Commonwealth Drive, Warrendale, PA 15096-0001.

AS9100	Quality Systems - Aerospace - Model for Quality Assurance in Design, Development, Production, Installation and Servicing
AS9006	Deliverable Aerospace Software Supplement for AS9100A

3. TERMS AND DEFINITIONS:

The following terms are important to the understanding of this recommended practice and are included here for ease of use.

Acceptance Testing: Testing conducted in an operational environment to determine whether a product satisfies its acceptance criteria.

Archive: The long-term storage to assure that the software, documents, and life cycle data associated with the software product are retrievable.

Business Systems/Office Software: General-purpose use software that is widely used and commercially obtainable.

Contract: (Software Contract): A binding agreement between two parties, enforceable by law, for the supply of software or the development, production, operation, or maintenance of a software product.

Commercial-Off-The-Shelf (COTS): Commercially available applications sold by vendors through public catalog listings.

Dataset(s): Information prepared and maintained by electronic means, and provided by electronic data access, interchange, transfer, or on electronic media.

Development: A software life cycle process that contains the activities and support for requirements analysis, design, coding, integration testing, installation and acceptance of software products.

Developed or tailored: Software that is developed, modified or customized by or for the user. This includes new development, modification, re-engineering, and maintenance activities that result in software used in product realization.

EARS: Export Administration Regulation.

ITAR: International Traffic in Arms Regulation.

3. (Continued):

Life cycle: A framework containing the processes, activities and tasks involved in the design, development, operation and maintenance of a software product, spanning the life of the system from the definition of its requirements to the termination of its use.

Non-deliverable software: Software used in the design, manufacture, inspection, test, acceptance, or calibration of a deliverable product.

Part Specific: Software that retains approved configuration information. This may include program files, product definition datasets (Authority Models), Numerical Control (NC) programs, Coordinate Measurement Machine (CMM) programs, and dataset derivatives from Computer Aided Design (CAD) models, etc.

Porting or Migrating: Modifying software to run on a different computer and/or operating system than those on which it has been verified and validated.

Prototype Software: Software in the development stages not used in product realization.

Purchased/Vendor Supplied: Software that is not modified or customized. This includes Commercial off the Shelf (COTS) software used for product realization. Software that is modified for use in production, or not used in its original off-the-shelf state, exclusive of selection of options and features, is not considered COTS.

Release: An approved version of a configuration item made available for a specific purpose (for example, test release).

Replication: A process of copying a software product from one medium to another.

Software: Computer programs and, possibly, associated documentation and data pertaining to the operation of a computer system. This definition includes executable programming logic and data that are embedded in hardware devices known as firmware.

Software item: Any identifiable part of a software product.

Software product: An output that results from a software process. Products can be tangible or intangible, a thing or an idea, hardware or software, information or knowledge, a process or procedure, a service or function. When ISO uses the term product, it also means service.

Stage: A defined segment of development.

NOTE: A stage does not imply the use of any specific life cycle model.

3. (Continued):

Supplier: An organization that enters into a contract with the acquirer for the supply of a system, software product or software service under the terms of the contract.

NOTE: The term supplier is synonymous with contractor, producer, seller, or vendor.

TAA: Technical Assistance Agreement.

Validation: The process of determining that software complies as follows: (1) conforms to its development standards (2) requirements and or specifications are fulfilled, (3) works as intended in the target environment. Validation determines that the fully integrated software functions correctly, completely, and consistently with system specifications and requirements.

Verification: Processes of evaluation to assure input requirements are achieved at the end of a development stage. Verification can occur at various times during a software process. Verification includes review, analysis, inspection, and test.

4. NON-DELIVERABLE SW QUALITY PLANNING:

A Quality Management System should be in place prior to implementing a specific system for non-deliverable software. The non-deliverable software system or plan should provide a documented process that addresses, at a minimum, the following items:

a. responsibility and authority within the organization,

b. identification of requirements,

c. analysis of risks and criticality,

d. verification and validation,

e. version control,

f. training and maintenance,

g. documentation,

h. supplier requirements and oversight.

5. REQUIREMENTS:

The organization should determine and document the software requirements, including business, organizational, safety, and security requirements. Requirements should be complete and unambiguous. The requirements should include, as appropriate:

a. performance criteria,

b. acceptance criteria,

c. system architecture (i.e., hardware/software compatibility, portability),

d. procedure, specification, and standard compliance

e. customer specific requirements (export controls such as ITAR, TAA, EARS, third party escrow agreement, etc.).

A process to decide whether to develop or purchase software should be documented. In either case, the organization should ensure acceptance testing to requirements is performed, including any customer specific software requirements.

6. NON-DELIVERABLE SOFTWARE REALIZATION:

Software realization is the process of developing or purchasing software for use in the design, manufacture, inspection, test, acceptance or calibration of a deliverable product. Non-deliverable software can be developed, tailored, purchased or vendor supplier.

Part specific software can either be developed or procured for use in the manufacturing environment. Part Specific software retains approved configuration information. This may include program files, product definition datasets (Authority Models), Numerical Control (NC) programs, Coordinate Measurement Machine (CMM) programs, and dataset derivatives from Computer Aided Design (CAD) models, etc.

6.1 Developed/Tailored Software:

Software that is developed, modified or customized by or for the user. This includes new development, modification, re-engineering, and maintenance activities that result in software used in product realization. This includes purchased or vendor supplied software that is modified or customized.

6.1.1 Development Planning: Development planning is the processes of establishing tasks associated with the development of non-deliverable software products. Development planning should include the following items:

a. the primary function of the software,

b. identification of inputs and outputs,

c. identification of software,

d. traceability of software requirements throughout the lifecycle

e. identification of applicable standards, conventions, tools, and techniques,

f. computer hardware requirements, limitations, and constraints,

g. interfaces between different groups involved in design and development to ensure effective communication and clear assignment of responsibility,

h. risk mitigation and software criticality.

6.1.2 Development Inputs: Development inputs are a comprehensive set of requirements used to define and develop software products. Inputs should include:

a. functional and performance requirements,

b. key characteristics needed to develop the deliverable product,

c. information derived from planning documentation (i.e., sketches, drawings, dataset),

d. applicable statutory or regulatory requirements,

e. environmental requirements (including operating systems and platforms),

f. system requirements (if the software does not function as stand-alone),

g. information derived from previous similar designs,

h. maintenance requirements.

6.1.3 Development Outputs: Development outputs are the expected result of the development and should be defined, documented, and be traceable to development inputs.

Development outputs should:

a. be verifiable to product requirements,

b. contain or reference acceptance criteria,

c. specify essential characteristics for safe and proper use,

d. provide pertinent data to allow identification, inspection, use and maintenance.

6.1.4 Development Review: Development reviews should be planned and conducted to identify problems, to determine if development outputs meet the input requirements, and to determine whether changes to the plans or requirements are necessary. Progression to the next phase should be approved when the input and output requirements have been met, and action items have been closed or mitigated. Records of the reviews should be documented and maintained.

6.2 Purchased or Vendor Supplied Software:

Purchased or vendor supplied software is not modified or customized. This includes Commercial off the Shelf (COTS) software used for product realization.

6.2.1 Purchasing Process: A software source selection and approval process should be used to evaluate the supplier's ability to meet the requirements for the software product. All impacted functional organizations should be included (e.g., purchasing, tooling, procurement, software quality, configuration management, etc.).

6.2.2 Purchasing Information: Purchase documentation should adequately describe the product to be purchased, and be approved prior to communication to the supplier.

Documentation should include requirements for:

a. product, process, equipment and practice approvals,

b. unique product identification,

c. applicable specifications, drawings or technical data,

d. inspection/acceptance requirements,

e. testing and associated test requirements,

f. nonconforming product requirements,

g. change notification requirements.

7. VERIFICATION AND VALIDATION:

Verification and validation of all non-deliverable software should be planned. The capability of software to satisfy the intended application should be confirmed. Necessary testing should be completed, results documented and reviewed. This should be undertaken prior to initial use and reconfirmed as necessary.

7.1 Verification:

Verification is the process of evaluating developed software to assure meeting input requirements at the end of a development stage. Verification can occur at various times during software development. Verification includes review, analysis, inspection, and test.

When certification and/or conformity reports are used to verify purchased product, the data in those reports should be in accordance with applicable specifications.

7.2 Validation:

Validation is the confirmation that the software requirements are fulfilled and the software works as intended in the target environment. Validation determines that the integrated software operates correctly, completely, and consistently with specifications and requirements.

Validation should be performed at a level of integration to sufficiently ensure product conformance.

7.2 (Continued):

When software outputs cannot be verified by monitoring or measurement, alternative validation processes should be established and approved. Applicable methods, procedures and criteria should also be defined.

New hardware environments may necessitate re-validation. Re-validation should occur when porting or migrating software into a new hardware or software environment.

7.3 Documentation of Verification and Validation:

Records of the reviews should be documented and maintained. The documentation should contain evidence that the software meets the identified operational parameters, and can include source code, user and support documentation, data, drawings, test data, and process planning.

8. CONFIGURATION CONTROL:

8.1 Identification and Traceability:

The organization should establish and maintain a configuration control process for non-deliverable software that includes the capability to:

a. uniquely identify each version of software,

b. identify each item that makes up a software product,

c. trace software to requirements,

d. manage change request and change control,

e. manage problem reporting, corrective action, and disposition of nonconforming software,

f. document and maintain the software's history and changes,

g. identify persons authorized to approve changes,

h. identify changes affecting processes, tools and programs,

i. identify approved production documentation,

j. identify associated development tools,

8.1 (Continued):

k. identify interfaces to other software and to target computer hardware,

l. identify the target computer hardware and software environment.

8.2 Control of Changes:

All software changes should be identified and documented to maintain the integrity and traceability of the software product. The impact of any changes made to software, data, operating environment, or dependent processes should be evaluated. Re-verification and re-validation of software should be employed to ensure that the modified software meets requirements. Additional validation may need to occur to ensure changes do not have any adverse effects on integrated software and products released for use.

The impact of vendor-supplied changes on the conformity of the software product (i.e., hot fixes, patches, temporary fixes, service packs, and vendor upgrades) should be determined.

8.3 Nonconforming Software:

Nonconforming software is software that does not meet requirements. Nonconforming software should be removed from production when the organization determines that the nonconformance is critical to configuration, conformity, or quality of the deliverable product and an approved disposition is not available.

A documented procedure for the review and disposition of nonconforming software should be established and include:

a. determination of the impact and the criticality of nonconformance,

b. identification of impacted areas,

c. appropriate notification,

d. disposition of software, and

e. tracking of corrective action including re-verification and re-validation to prevent recurrence.

8.4 Nonconforming Products:

Validated and conforming software that produces nonconforming products should be reviewed to ensure that the input and output requirements were adequately defined. If it is determined that the requirements were not adequately defined, then corrective action should include redefinition of requirements.

Corrective action for software should include re-verification and re-validation of the identified defective software in its operating environment to ensure that requirements are met, and assure, that the nonconformance will not recur. There should be a method or system in place to allow for retrieval of nonconforming parts/products.

9. PROVISIONS FOR PRODUCTION SOFTWARE:

9.1 Controls and Utilization:

Non-deliverable software should be controlled for proper utilization in a production environment.

9.1.1 Release and Distribution: The documented release and distribution processes should address the following provisions:

a. identification of the released software,

b. notification to users,

c. distribution procedures,

d. installation procedures,

e. control of the replication environment,

f. verification of the software copies,

g. user documentation,

h. master and copy storage information.

Subsequent to release, periodic inspections should be performed to ensure approved configurations are in use, and that applicable preservation and storage practices are followed.

Proprietary, usage, ownership, copyright, warranty and licensing rights should also be considered.

9.1.2 Access Control: Non-deliverable software should be protected from unauthorized changes, and access controls should address:

a. physical access,

b. password protection,

c. read and write access of master,

d. read and write access of copies,

e. Edit Key restrictions.

Protection of the system should also consider external contamination via network, modem, portable media or other means.

9.1.3 Obsolete Software: Non-deliverable software that is no longer required for product realization should be restricted and/or removed from all systems so that it is no longer available for use. Master copies, duplicates, and user copies should be restricted and/or removed from all areas except the archive. Obsolete software in the archive should have restricted access to prevent unauthorized use.

9.1.4 Preservation of Product: The organization should establish documented procedures for non-deliverable software, with requirements for preservation of conformity, segregation and safeguarding, which include:

a. marking and labeling,

b. archive and retrieval of software,

c. verification that software is not corrupt prior to storage, including free from viruses,

d. safeguarding software media from contaminates considered a potential cause of deterioration such as temperature, humidity, magnetic fields, or other causes of media deterioration,

e. handling of software media to protect from change or corruption,

f. encryption/decryption, compression/decompression,

g. storage of software media in separate protected environments to ensure disaster recovery.

9.2 Production Documentation:

Production documentation for software should include approval for use, release information, description of the product, support documentation, and operating instructions that may include necessary work instructions.

The organization should establish documented procedures to ensure the accuracy and repeatability of software controlled measurement equipment during data collection (e.g., scale bar, offsets, temperature compensation, coordinate system manipulation, field calibration, etc.). Records of the results of calibration and verification should be maintained.

10. OTHER CONSIDERATIONS:

10.1 First Article Inspection:

The First Article process should specify the unique inspection features of the product in cases where product characteristics are embedded in the software, (e.g., digital modeling definitions, autoclave recipe, etc.).

10.2 Internal Audit:

Internal audit or review processes should be employed to ensure compliance to established software development, procurement and control procedures.

10.3 Customer Communication:

The organization should ensure that a process exists for effective communications with the customer.