PROJECT PERFORMANCE INTERNATIONAL
helping projects succeed…
www.ppi-int.com

DATA ITEM DESCRIPTION

1. TITLE	2. IDENTIFICATION NUMBER
SOFTWARE REQUIREMENTS SPECIFICATION (SRS)	PPA-002237-11 23 November 2020

3. DESCRIPTION/PURPOSE

3.1 The Software Requirements Specification (SRS) specifies the requirements to be satisfied by a software item (e.g., software system, subsystem, Computer Software Configuration Item (CSCI), component or other item), and, if required, corresponding verification requirements. Requirements pertaining to the software item's external interfaces may be presented in the SRS or in one or more Interface Requirements Specifications (IRSs) invoked by reference from the SRS.

3.2 The SRS, possibly supplemented by IRSs, is used as the basis for procurement, design, verification testing and acceptance testing of the software item. Throughout this DID, the term "software item" may be interpreted to mean "CSCI", "software system", "software subsystem", "software component" or any other software item, as applicable.

4. APPLICATION/INTERRELATIONSHIP

This Data Item Description (DID) may be cited in a Statement of Requirement (SOR), Task Specification (TS), a Contract Data Requirements List (CDRL), within a standard invoked by a SOR or SOW, or within a plan or procedure.

5. PREPARATION GUIDELINES

5.1 General Instructions

a. **Automated techniques.** Use of automated techniques is encouraged. The term "document" in this DID means a collection of data regardless of its medium.

b. **Alternative presentation styles.** Diagrams, tables, matrices, and other presentation styles are suitable substitutes for text when data required by this DID can be made more readable using these styles.

continued next page

6. SOURCE

5. PREPARATION GUIDELINES *continued*

5.2 Acronyms

Acronyms used in this document shall be interpreted as follows:

CC	Creative Commons
CDRL	Contract Data Requirements List
CSCI	Computer Software Configuration Item
DID	Data Item Description
HWCI	Hardware Configuration Item
ICD	Interface Control Document
IRS	Interface Requirements Specifications
SI	International System of Units
SOR	Statement of Requirement
SOW	Statement of Work
SRS	Software Requirements Specification
TS	Task Specification

5.3 Abbreviations

Abbreviations used in this document shall be interpreted as follows:

SI	International System of Units

5.4 Foreword

This Data Item Description (DID) for a Software Requirements Specification (SRS) is intended to provide guidance and instruction on the preparation of a requirements specification for any required software item. A SRS can be used in relation to any software product or system whatsoever, from a cell phone application to software within an IT system for internal use, to embedded software in a train control system. The software that is the subject of a SRS may be new software or changed to existing software. The subject of the SRS may be executable software, or a database.

Hereinafter, the word “software” is used to refer to the item that is the subject of the Software Requirements Specification.

A SRS specifies the essentially solution-free requirements to be satisfied by any acceptable software solution.

The SRS may also specify goals to be pursued during software procurement and/or software item development (if applicable).

The SRS is usually the single most important artefact in the development or acquisition of software. Creation, capture and specification of the information content of the SRS should be done with the utmost care, and with appropriate skills applied, to avoid problems such as:

a. software that does not satisfy the needs of the enterprise at all, or falls significantly short of satisfying the needs;
b. avoidable inaccuracies in cost estimation; and
c. delays due to rework, contractual dispute, or the need to undertake supplementary developments or procurements.

In an environment of genuinely changing needs for the software, the SRS will evolve accordingly.

5.5 Content Requirements

Content requirements begin on the page 3. The numbers shown designate the paragraph numbers to be used in the document. Each such number is understood to have a prefix “5.5” within this DID. For example, the paragraph numbered 1.1 is understood to be paragraph 5.5.1.1 within this DID.

TABLE OF CONTENTS

1. INTRODUCTION AND SCOPE

This section may be divided into the following paragraphs where the volume and content of relevant information justifies sub-paragraphing.

1.1 Identification

This paragraph should contain a full identification of the software to which this document applies, including, as applicable, identification number(s), title(s), abbreviation(s), version number(s) and release number(s). Where the software to which the document applies includes variants of the software, the above information should be provided for each variant. Where variants apply, this paragraph should establish nomenclature for the variants, and any rules for applicability in section 4. of requirements to variants. Where the software to which the document applies includes incremental builds of the software which are subject to individual specification within the overall software requirements specification, the above information should be provided for each such build.

1.2 Intended Use

This paragraph should briefly state the intended use of the software to which the document applies, possibly referring to an Operational Concept Description or comparable description for more detail.

1.3 Background

This paragraph, if used, may summarize the history of software development, operation, and maintenance (if any); and identify, as applicable, the project sponsor, software acquirer, user, supplier and support organizations.

1.4 Software Overview

This paragraph, if used, should identify, if applicable, and list any major software components which are required by 4., and which have end-use significance.

Where the software is required to be structurally configurable, this paragraph should state so, and should state the nature of the configurability.

1.5 Document Overview and Use

This paragraph, if used, should summarize the purpose and contents of this document and should describe any security or privacy considerations associated with its use.

2. APPLICABLE AND OTHER REFERENCED DOCUMENTS

This section should list the number, title, revision, and date of each document referenced in this specification. This section should also identify the source of each document not available through normal channels.

2.1 Applicable Documents

This paragraph should list each document which is invoked in whole or in part within 4. as containing requirements information. The paragraph should contain any applicable rules for establishing precedence in the event of conflict of requirements between 4. and the applicable documents, and between applicable documents. The paragraph should also contain, where applicable, rules for establishing the applicable issue number of documents invoked in 4., and identify any applicable security classification.

2.2 Other Referenced Documents

This paragraph should list each document which is referenced in the specification but which is not invoked in whole or in part by 4. as containing requirements information.

3. DEFINITIONS, ACRONYMS AND ABBREVIATIONS

This section should be divided into the following paragraphs.

3.1 Definitions

This paragraph should list alphabetically and define each word or term used in 4. for which reliance on dictionary definitions is not appropriate. As a guide, terms which are not likely to be in the vocabulary of the intended users of the specification, terms which have multiple dictionary meanings but only a single specification meaning, technical terms and terms which are used with special meanings should be defined in this paragraph.

The following definitions, or similar, should be incorporated into this section:

Shall expresses a characteristic which is to be present in the item which is the subject of the specification, i.e. "shall" expresses a binding requirement.

Should expresses a target or goal to be pursued, but not necessarily achieved.

May expresses permissive guidance.

Will may be used to express a declaration of intent on the part of a party, usually the sponsoring or contracting organization. "Will" does not express a requirement. "Will" may also be used in cases where the simple future tense is required, for example, "Details of the hardware platform will be supplied upon request." Any statement which employs the term "will", if used in 4., should be present as a note so as to be clearly distinguishable from requirements.

This paragraph should also identify by name and issue the dictionary to be used in the interpretation of terms used in 4. The common dictionaries which apply for English language are:

United States of America	Merriam-Webster's Dictionary
Australia	Macquarie Dictionary
New Zealand	Concise Oxford Dictionary.

3.2 Acronyms

This section should list alphabetically each acronym used in the document, together with the acronym's expanded meaning.

3.3 Abbreviations

This section should list alphabetically each abbreviation used in the document, together with the abbreviation's expanded meaning, except that abbreviations within the International System of Units (SI) should not be listed.

4. REQUIREMENTS

This section should be divided into the following paragraphs to specify the requirements of the software item, that is, those characteristics of the software item that are required to be present in the final product. In some cases, such requirements will also be conditions for acceptance of the software item. Each requirement should be assigned a project-unique identifier to support verification and traceability, and should be stated in such a way that an objective, finite and cost-effective test or other verification activity and pass criterion can be defined for it.

If there are no requirements corresponding to a given paragraph of the DID, the DID paragraph may be deleted in the specification and other paragraph numbers adjusted accordingly. Alternatively, "Not used" may be inserted under the paragraph heading. If a given requirement fits into more than one paragraph, the requirement should be stated once and referenced from the other paragraph(s). Duplication of requirements should be avoided.

The degree of detail to be incorporated in specifying requirements should be guided by the following principle: include those characteristics of the software item that are necessary for the software item to satisfy its intended use; defer to design those characteristics that the specifier is willing to leave up to the developer or supplier.

In determining characteristics necessary to satisfy intended use, the criterion which should be used is the level of risk associated with satisfaction of the following ideal: "that any software item which is supplied which satisfies the requirements in 4. will satisfy the need". The level of acceptable risk with respect to attainment of this ideal should be determined as a prerequisite to preparation of the specification for the software item.

Typically, for a given acceptable level of risk, lesser rigor of specification will suffice for software items which are non-developmental, whilst for developmental software items a more rigorous specification is called for. Similarly, where the specification is for internal communication of requirements only then lesser rigor is needed, whilst to satisfy the legal demands and address the financial implications of contracting, greater rigor is called for.

4.1 Identification of External Interfaces

This paragraph should identify all required external interfaces of the software item. The identification of each interface should include a project-unique identifier for the interface and should designate the interfacing entities

by name, number, version and documentation reference, where these exist and are applicable. The name of the interface may serve this purpose.

A diagram which depicts the interfaces may be included for information with interfaces shown which correspond to the specified interfaces. The context diagram is a suitable form of representation for this purpose. If used, a context diagram should be conceptual in nature.

Note that paragraph 4.5 rather than this paragraph should be used to specify the requirements applicable to each external interface identified in this paragraph.

4.2 Identification of States and Modes (or "Configurations, States & Modes")

If the software item is required or permitted to exist or operate in more than one state or mode having requirements distinct from those in other states or modes, this paragraph should identify each state and mode which is permitted or required. Examples of states and modes include: uninstalled state, installed state, executing state, error state, demo mode, training mode, beginner mode, expert mode. Sub-modes may be used. With respect to states and modes, a software item may be described in terms of states only, modes only, modes within states, or any other scheme that is useful. Alternatively, a software item may be specified without reference to states and modes.

A states and modes schema which has states at the highest level of requirements organization, and normally mutually exclusive of other states, together with modes within states, modes not necessarily being mutually exclusive of other modes and being able to exist in multiple states, has been found to be useful for the specification of software items of many types.

If no states or modes are required, this paragraph should be omitted or so state, without creating artificial distinctions. If states or modes or both are required, each requirement or group of requirements in this specification which relates to a state or mode should be correlated to that state or mode. The correlation of a requirement to one or more states and modes should be specified by inclusion of such correlation in the specification of the requirement where it appears in the specification (somewhere in 4.3 to 4.8). This may be achieved in the requirement sentence (preferable) or paragraph (if necessary for clarity) in which the requirement is stated.

Any statements about the applicability of requirements in general to the states and modes identified in this paragraph should be included in this paragraph.

If the software is required to be structurally configurable, the required configurations should also be identified in this paragraph.

4.3 Software Item Function and Performance Requirements

This paragraph should be divided into subparagraphs to specify each function required to be performed by the software item, together with associated required performance. Each requirement should reference as necessary any external interfaces, states or modes identified in 4.1 or 4.2.

4.3.x (Software item) Function

This paragraph should identify in the heading a required software item function and should specify the required function. The word "Function" should appear in each leaf subparagraph heading. The requirements should specify required behavior of the software item and should include, for each function, applicable performance parameters, such as response times, throughput times, other timing constraints, sequencing, accuracy, capacities (how much/how many), priorities, continuous operation requirements, and allowable deviations based on operating conditions. The requirements should include, as applicable, required behavior under unexpected, unallowed, or "out of bounds" conditions, and any requirements for error handling.

Functional and performance requirements may be organized in a structure of section, paragraphs, subparagraphs, etc. Requirements should be placed only in the leaf subparagraphs.

The word "Mode" may also be used in a paragraph heading.

If the requirement is a requirement of a subordinate element of the software item rather than of the software item overall, the requirement should be placed in 4.9.2 and not in this paragraph.

4.4 Relationships Between States & Modes

This paragraph, if states and modes are used, should specify the required relationships between the various states and modes, including temporal relationships, the conditions that are required to cause state to state and

mode to mode transitions and the external response(s) that the software item is required to produce as a reflection of each required transition having taken place.

Where sub-modes are used, the corresponding requirements relating to the sub-modes should be included.

4.5 Software Item External Interface Requirements

This paragraph should be divided into subparagraphs to specify the requirements, if any, for each of the software item's required external interfaces. This paragraph may reference one or more Interface Requirements Specifications (IRSs) or other documents containing these requirements, which may be either annexes to the software item specification or separate documents.

Where an interface is inherently simple, its specification may be contained, rather than referenced, in this paragraph.

4.5.x (Name/Project-Unique Identifier of Interface)

This paragraph (beginning with 4.5.1) should identify a software item external interface by name and any additional project-unique identifier, and should briefly identify the interfacing entities.

Each paragraph should be divided into subparagraphs as needed to state the requirements which the interface must satisfy. The interface should be specified from the viewpoint of the interface as a surface through which inputs and outputs pass. The paragraph should specify all required characteristics at the surface, of the inputs, and of the outputs, including relationships between these items. The paragraph may reference other documents (such as data dictionaries, standards for communication protocols) in place of stating the information here. Requirements should note any differences in these characteristics from the point of view of the interfacing entities (such as different expectations about the size, frequency, or other characteristics of data elements).

Requirements may be structured in a manner which adopts an open software items approach. An open software system is a software system in which interfaces are defined in a series of abstract layers, accessible from the interfacing entity(ies), which progressively and in a structured manner, provide the required interface mechanism. The OSI Reference Model for Open Systems Interconnection is one such schema.

The requirements should include the following, as applicable, presented for each interface in any order suited to the requirements (subject to the open systems guidance contained above):

a. requirements on the types of interface (such as real-time data transfer, storage-and-retrieval of data, etc.), to be implemented.

b. required characteristics of individual data elements that the software item must input/output, such as:
    i. names/identifiers;
        1. project-unique identifier;
        2. non-technical (natural-language) name;
        3. standard data element name;
        4. technical name (e.g. variable or field name in code or database); and
        5. abbreviation or synonymous names;
    ii. data type (alphanumeric, integer, etc.);
    iii. size and format (such as length and punctuation of a character string);
    iv. units of measurement (such as meters, dollars, nanoseconds);
    v. range or enumeration of possible values (such as 0-99);
    vi. accuracy (how correct) and precision (number of significant digits);
    vii. priority, timing, frequency, volume, sequencing, and other constraints;
    viii. security and privacy constraints; and
    ix. sources (setting/sending entities) and recipients (using/receiving entities).

c. required characteristics of data element assemblies (records, messages, files, arrays, displays, reports, etc.) used to present information that the interfacing entity(ies) must input/output, such as:
    i. names/identifiers;
        1. project-unique identifier;
        2. non-technical (natural language) name;
        3. technical name (e.g. record or data structure name in code or database); and
        4. abbreviations or synonymous names;
    i. data elements in the assembly and their structure (number, order, grouping);
    ii. relationships between data elements within a data element assembly, and relationships between data element assemblies;

iii. visual and auditory characteristics defined within data element assemblies (such as colors, layouts, fonts, icons, beeps, lights and other display driver element assembly characteristics);
iv. relationships among assemblies, such as sorting/access characteristics;
v. priority, timing, frequency, volume, sequencing, and other constraints;
vi. security and privacy constraints;
vii. message formatting; and
viii. sources (setting/sending entities) and recipients (using/receiving entities).

d. required characteristics which the interface must satisfy to enable organization and synchronization of connections between interfacing entity(ies), such as:
   i. project-unique identifier(s);
   ii. session-connection establishment—creation of an exchange between interfacing entity(ies);
   iii. session-connection release;
   iv. session-connection synchronization;
   v. exception reporting—permitting the interfacing entity(ies) to be notified of exceptional situations;
   vi. data transfer rate, whether periodic/aperiodic, and interval between transfers;
   vii. message formatting; and
   viii. safety/security/privacy considerations, such as user authentication and auditing inputs/outputs.

e. required characteristics of data flow methods that the interfacing entity(ies) must use for the interface, enabling the interfacing entity(ies) to assume cost-effective and reliable data exchange such as:
   i. project-unique identifier(s);
   ii. transmission services, including priority and grade;
   iii. message formatting; and
   iv. safety/security/privacy considerations, such as auditing inputs/outputs.

f. required characteristics of communication methods that the interfacing entity(ies) must use to establish, maintain and terminate connections between interfacing entities, such as:
   i. project-unique identifier(s);
   ii. message formatting;
   iii. flow control (such as sequence numbering and buffer allocation);
   iv. routing, addressing, and naming conventions;
   v. synchronization, including connection establishment, maintenance, termination; and
   vi. safety/security/privacy considerations, such as auditing inputs/outputs.

g. required characteristics of protocols the interfacing entity(ies) must use for the interface, such as:
   i. project-unique identifier(s);
   ii. priority/layer of the protocol;
   iii. packeting, including fragmentation and reassembly, routing, and addressing;
   iv. legality checks, error control, and recovery procedures;
   v. synchronization, including connection establishment, maintenance and termination; and
   vi. status, identification, and any other reporting features; and

h. the required interface characteristics of the computer hardware that must be used by the software item.

Any requirements related to the interface that are of the nature of system functionality should be incorporated in 4.3 and not in this paragraph, except where ease of use of the SRS would be enhanced by incorporation of such requirements in this paragraph. This can occur when, for example, a particular communications protocol is required to be used across an interface. If such requirements regarding functionality of the interfacing systems are included in the IRS, a pointer to the required functionality should be included in the Section 4.3 "Functional and Performance Requirements" of the system (or software) requirements specification for each of the interfacing systems.

Any requirements which specify the consumption or usage of externally supplied resources should be incorporated in 4.6 and not in this paragraph.

External interface requirements should be specified only to the degree necessary to bound the design of the external interface. For a developmental project, this degree will often increase throughout the course of the software project, i.e. external interfaces will be initially specified at a high level of abstraction and will eventually be specified at the level suitable for coding to achieve the interface characteristics.

An external interface may also be specified in terms of achieving functional interoperability with interfacing external software (e.g., operating system, database management system, communications/network software, utility software, input and equipment simulators, test software and manufacturing software) or hardware (e.g.

processor). Care should be taken if using this form of specification, as it relies on the specifier defining required interface characteristics in terms which include the characteristics of external software or hardware items, characteristics which may not be under the control of the specifier, and which may not be stable with time. Explicit nomenclature, version and documentation references for each such software and hardware item shall be provided.

4.6 Environmental Requirements

This paragraph should state the host environments within which the other requirements of the software are to be met. A host environment is a computing hardware/operating system combination.

4.7 Computer Hardware Resource Utilization Requirements

This paragraph should specify the requirements, if any, regarding the utilization by the software item of resources provided to the software item by computer hardware. Examples are processing capacity, memory capacity, input/output device capacity, auxiliary storage device capacity and communications/network capacity. The requirements should be stated to include the conditions under which resource utilization constraints apply.

4.8 Other Software Item Qualities

This paragraph should specify the requirements, if any, pertaining to other qualities of the software item as a whole. Example other software item qualities include:

a. reliability (the ability to perform with correct, consistent results);
b. maintainability (the ability to be easily corrected);
c. availability (the ability to be accessed and operated when needed);
d. portability (the ability to be easily modified for a new environment);
e. reusability (the ability to be adapted for use in multiple applications);
f. testability (the ability to be easily and thoroughly tested);
g. useability (the ability to be easily learned to use, and easily used);
h. expandability (the ability to be easily modified in response to potential areas of growth in requirements);
i. flexibility (the ability to be easily adapted to changes in mission, threat or technology); and
j. interoperability (the ease of interfacing in general with external software and hardware items. Specific interfaces with external software and hardware items should be specified in 4.5 and not in this paragraph).

Any requirement related to other software item qualities and which is functional in character should be specified in 4.3 and not in this paragraph.

Any requirement for other software item qualities which is a requirement of a subordinate element of the software item rather than of the software item overall should be placed in 4.9.2 and not in this paragraph.

4.9 Design and Coding Requirements

This paragraph should specify the requirements, if any, which *direct* aspects of the design and coding of the software item. This paragraph, where used, should be divided into the following subparagraphs.

4.9.1 General Design and Coding Requirements

This paragraph should specify the general aspects of design and coding which apply software item-wide. Examples include requirements concerning:

a. use of a particular software item architecture, or requirements on the architecture, such as the incorporation of specific software components, including COTS software components;
b. use of customer-furnished software;
c. use of particular software languages;
d. use of particular design or coding standards; and
e. use of particular internal data elements and database structures.

4.9.2 Characteristics of Subordinate Elements

This paragraph should be divided into subparagraphs which correspond to a simple, flat, alphabetical list of software components, modules, classes, etc., to which requirements apply as requirements on the software item. Each element should be specified only to the extent justified by the reasons for directing design, using the paragraph and subparagraph titles and content corresponding to 4.1 to 4.8.1, as applicable.

Only those characteristics of the software item which the specifying enterprise *requires* be implemented by the above subordinate elements should be specified in this paragraph. Note that by specifying the existence or

characteristics of subordinate elements, the specifier assumes responsibility for aspects of software item design. Care should be exercised in taking on this responsibility.

If training devices or training materials in soft form are to be included in the software item, such devices or materials should be specified in this paragraph.

If user documentation or maintainer documentation in soft form are to be included in the software item, such documentation should be specified in this paragraph.

If logistics support devices and logistic support materials in soft form are to be included in the software item, such devices or materials should be specified in this paragraph.

Media or packaging materials should not be specified in a Software Requirements Specification.

4.10 Precedence of Requirements

This paragraph, if applicable, should state whether all paragraphs have equal precedence in the event of conflict between requirements. If all requirements are not to be of equal precedence in the event of conflict, this paragraph should state the rules of precedence.

Note that the possibility of conflict between requirements is increased where design requirements are specified. In this case, the paragraph should state rules of precedence between 4.9 and other requirements paragraphs.

5. VERIFICATION REQUIREMENTS (OPTIONAL)

This section, if used should define a set of verification requirements, each verification requirement being a statement of the extent and nature of the evidence required that the corresponding software requirement has been met. Verification requirements may be expressed in statistical terms, e.g., confidence level to be accomplished by the verification activity, or may direct how a verification activity is to be conducted. This section may also specify, for each requirement in 4., the generic method(s) to be used to evidence that the requirement has been met. A table may be used to present this information, or each requirement in 4. may be annotated with the method(s) to be used.

Verification methods may include:

a. Test - The operation of the software item, or a part of the software item, using instrumentation or other special test equipment to collect data for later evaluation;
b. Demonstration - The operation of the software item, or a part of the software item, that relies on observable functional operation not requiring the use of instrumentation, special test facilities, or subsequent analysis;
c. Analysis - The processing of accumulated data obtained from other qualification methods. Examples are reduction, interpolation, or extrapolation of test results, or deduction from theory;
d. Inspection - The visual examination of software item code, documentation, etc.;
e. Analogy - Use of past verification evidence relating to a comparable item, for example, a previous version;
f. Certification - Written declaration by the developer that the requirement has been satisfied;
g. Simulation - Using the execution of a model of the software as a source of evidence of compliance; and
h. Special verification methods - Any special verification methods for the software item, such as special tools, techniques, procedures, facilities, acceptance limits, or the use of process control.

6. NOTES

This section should contain any general information that aids in understanding or using the specification (e.g., background information, rationale).

This section may include the following paragraphs, as applicable.

6.1 Intended Use

This paragraph, if used, should contain or reference a description of the intended use operational concept of the software item, i.e., intended users, uses, how it is intended the software item be used, and the relevant external conditions during use.

6.2 Requirements Traceability

This paragraph should contain, if applicable:

a. data which details traceability from each software item requirement in the specification, including each requirement in annexed IRSs, to the higher-level requirement(s) it addresses, fully or partly. Alternatively, this traceability may be provided by annotating each requirement in 4. Each software item requirement should trace to one or more higher-level requirements, as applicable; or alternatively
b. reference to the document or database which contains requirements traceability information.

Note: Higher-level requirements are requirements which are source requirements contained in source documents. Source documents may include operational requirements documents, policy documents, standards, legislation, requirements clarification records, etc., or the specification (including applicable IRSs/ICDs) of the higher-level element of which the item is a part, e.g., a system, subsystem, HWCI, hardware component, CSCI.

6.3 List of Safety Requirements

This paragraph, if used, should list the software item requirements, if any, specified in 4. and concerned with preventing or minimizing unintended hazards to personnel, property and the physical environment.

Alternatively, safety requirements may be annotated as such in 4.

Safety requirements are typically listed only if they are subject to special actions, for example, increased verification as to their satisfaction.

6.4 List of Information Security Requirements

This paragraph, if used, should list the software item requirements, if any, specified in 4. and concerned with maintaining information security, viz confidentiality and integrity of information. Such requirements may include, as applicable, the security/privacy environment in which the software item must operate, the type and degree of security to be provided, the safeguards to reduce security risks, the security/privacy policy that must be met, the security/privacy accountability the software item must provide, and the criteria that must be met for security/privacy certification/accreditation.

Alternatively, information security requirements may be annotated as such in 4.

Information security requirements are typically listed only if they are subject to special actions, for example, increased verification as to their satisfaction.

6.5 Summary of Adaptation Requirements

This paragraph, if used, should identify the requirements, if any, specified in 4. and concerning installation-dependent data that the software item is required to use (such as site-dependent latitude and longitude or site-dependent post codes) and operational parameters that the software item is required to use that may vary according to operational needs (such as parameters indicating operation-dependent language constants or data recording).

Alternatively, adaptation requirements may be annotated as such in 4.

6.6 Ordering Details

This paragraph, if used, should provide any information necessary for identification of the software item and any variants of the software item for ordering purposes.

6.7 Criticality of Requirements

This paragraph should specify, if applicable, the criticality, or assigned weights, or both, indicating the relative importance of the requirements in this specification. An example is identification of those requirements deemed critical to mission, or to safety, or to security, for purposes of singling them out for special treatment, e.g. a higher level of independent verification and validation.

6.8 Value Model

Where goals (design goals) are expressed as goals above a specified minimum standard, this paragraph should state the relative importance of the difference between the minimum standard and the goal, for each design goal, as related to the value perceptions of the relevant stakeholders in the software, together with how the value changes between the minimum standard and the goal. Alternatively, this paragraph may reference an external value model, typically a data file accessed via value modeling software.

A. ANNEXES

Annexes may be used to provide information published separately for convenience in document maintenance or use (e.g., charts, databases). As applicable, each annex should be referenced in the main body of the document where the data would normally have been provided. Annexes may be bound or prepared digitally as separate documents for ease in use. Annexes should be lettered alphabetically (A, B, etc.).

Appendices may be used to annexes. Appendices should be numbered numerically (1, 2, etc.).