TabLan.15926

The Methodology of a Technical Document Semantic Modeling

Ver. 2.3

06.08.2013

Moscow

Contents

Revision history

Version 2 (published 08.03.2012)
First English release of the methodology.

Version 2.1 (published 30.09.2012)
Manual 8 Data table transformation updated to reflect interface changes in .16926 Editor version 1.00.

Version 2.2 (published 19.11.2012)
Methodology and sample file changes as new Methodology reference data schema elements became available in TechInvestLab.ru RDL sandbox.
References to the release of open-source TabLan extension for .15926 Editor version 1.10.
Minor typos corrected.

Version 2.3 (published 06.08.2013)
Namespace for PCA reference data library changed to *http://posccaesar.org/rdl/* as PCA changed namespace in library export file to the one used at endpoint.
Namespace for p7tpl templates changed to *http://standards.iso.org/iso/ts/15926/-8/ed-1/tech/reference-data/p7tpl#* as it was changed in files accompanying official ISO release of Part 8.
References to PCA RDL and TechInvestLab sandbox provided in the documentation.

General information

This methodology (hereafter the Methodology) describes a transformation of a natural language technical document into a formal model compliant to the standard **ISO 15926 «Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities»**. The Methodology is tailored for technical documents developed at early stages of the system life-cycle – requirements definition and architectural design. The use of this Methodology at later stages requires adjustments to data models and modeling rules, reflecting a different nature of more specialized technical documentation.

Formal model built according to this Methodology consists of the set of reference data items and relationships between them – all derived from the modeled document, where they are described by humans and for humans. Formal model is computer-interpretable and can be applied to the configuration and data population of various software applications used in the management of engineering data, procurement/asset management data or master data (systems otherwise known as requirements management systems, CAD\CAM, PLM or EAM, ERP, etc.).

This Methodology describes several steps to create reference data items, statements on their interrelationships and their relationships to the elements of external reference data libraries. Requirements (statements about "what should be") are modeled as relationships also.

To understand the basic principles of ISO 15926 we recommend reading of FIATECH **"An Introduction to ISO 15926"** (available at http://fiatech.org/images/stories/techprojects/project_deliverables/iso-intro-ver1.pdf). Reference data engineering processes are described in **«ISO 15926 Reference Data Engineering Methodology»** prepared by TechInvestLab.ru (available at http://techinvestlab.ru/RefDataEng).

This Methodology is intended for a domain engineer well acquainted with a particular technical domain to which a modeled document belongs. The Methodology contains necessary information from ISO 15926 and is tailored for as little specific knowledge of ISO 15926 data modeling as possible, allowing completion of some specific and tricky modeling later. Nevertheless this Methodology assumes that its user is at least superficially familiar with Parts 1, 2, 4 and 6 of ISO 15926, and with some systems engineering terminology (as introduced in the **ISO 15288:2008** standard – *system of interest*, *life cycle*, *processes*, etc.).

The Methodology uses a special table language created for formal data model representation – TabLan.15926 or simply TabLan. This language is in part inspired by Gellish engineering language (http://sourceforge.net/apps/trac/gellish/) as a much simplified version.

.15926 Editor from TechInvestLab.ru (available at http://techinvestlab.ru/dot15926Editor) is used for transformation of a TabLan document model to ISO 15926 format and for viewing of modeling results. The software module doing the mapping is distributed as open-source extension to the .15926 Editor. Some familiarity with Parts 7 and 8 of the ISO 15926 standard is expected for the use of this software. The license, terms of distribution and usage can be found with software package.

In accordance with the **ISO 24744** standard this Methodology defines:

- languages/notations used for document characterization;
- life-cycle stages for the system of interest (document model);
- processes performed during the life-cycle;
- tools used and tool usage instructions.

The Methodology describes a life-cycle of reference and project data derived from data initially presented in a textual form, which is different in some aspects from life-cycles described in other sources.

Document characterization languages

Data description table language (TabLan.15926 or TabLan)

The Methodology uses a special language of data description tables – TabLan.15926 or simply TabLan. TabLan is defined by:

- list of standard data description tables;
- standard table column headers;
- restrictions on the data of the table columns;
- optionality of the data in table columns.

Descriptions of TabLan tables used can be found below together with the descriptions of corresponding processes in this Methodology. Understanding of the data description tables' language is mandatory for the Methodology usage.

All the reference data items in TabLan tables should be defined in the core reference data description table – **Class Definition** table. The main requirement for all the tables is the precise matching of names an all the cells in the tables referring to the same concept. Reference data item name anywhere in a table should be entered exactly as it is entered in the Class Definition table columns **Russian Unique Class Name** or **English Unique Class Name** (see also the chapter **Core description of reference data**).

ISO 15926 part 7 template language

TabLan tables' transformation into RDF/OWL files with ISO 15926 compliant data is described in an accompanying document **«TabLan.15926 Data Mapping Specification»** using ISO 15926 Part 7 templates. The knowledge of ISO 15926 part 7 template language is required for the understanding of the **«TabLan.15926 Data Mapping Specification»,** and also for an extension of TabLan data description tables and predefined relationship types, and is not required when using this Methodology.

RDF/OWL

RDF/OWL files containing ISO 15926 data model of TabLan-encoded data use OWL Web Ontology Language in RDF/XML exchange format with RDF Schema. The knowledge of RDF, RDF Schema and OWL is not required when using this Methodology.

Elements of the Methodology reference data schema

Along with reference data items extracted from the technical document during characterization process, this Methodology uses predefined reference data items (classifiers, standard relationships, etc.), extracted from the POSC Caesar Association (PCA) reference data library (https://www.posccaesar.org/wiki/Rds) or developed specifically for the application of the Methodology and registered in the TechInvestLab.ru reference data sandbox (http://techinvestlab.ru/data). All such reference data items are described in the section «Reference data schema elements» in core reference data description table (**Class Definitions** tab). It is not recommended to change this section of the table during the application of this Methodology.

Their usage is described below and further illustrated in the **«TabLan.15926 Data Mapping Specification»**.

Tool support

The TabLan data description tables are maintained in electronic spreadsheets in an Office Open XML format (**.xlsx**), supported by Microsoft Excel, OpenOffice or LibreOffice software tools. Tables in the old format of Microsoft Excel (.xls) are not supported by transformation software! A sample spreadsheet file is distributed with the .15926 Editor.

TabLan tables' transformation into RDF/OWL files with ISO 15926 compliant data is described in the **«TabLan.15926 Data Mapping Specification»**. The transformation is done by the .15926 Editor open-source extension (available at http://techinvestlab.ru/dot15926Editor). This software also does a visualization of ISO 15926 models. The software is developed by TechInvestLab.ru. License, terms of distribution and usage can be found with software package.

Document model life-cycle

General description of a life-cycle

We are working with a technical document in a textual form, prepared for reading by humans.

The life-cycle of a document model consists of the following stages:

1. Selection of the technical document(s) to be modeled.
2. Structural modeling of the document.
3. Semantic modeling of the document.

The first stage of the text document characterization is a construction of a structural model of the document. The structure is ideally modeled down to the level of elementary text statements that only human can interpret.

After the structural model the semantic model of the document is constructed by modeling the machine interpretable "meanings" of text statements. Such a machine interpretation can be, for example, automated loading of the data into a database storing engineering, scheduling, accounting/finance master data – a storage of CAD\CAM, PLM or EAM, ERP system. Such loading can be either a part of those systems configuration (setting vocabularies, references, breakdown structures etc), or a part of filling those systems with data. If requirements to the engineered system had been elicited from the technical specification document, it will be possible later to check the design decisions described in the CAD\CAM systems against the modeled requirements.

To do the semantic modeling we should identify and describe elements of the model "vocabulary" (classes or individuals) and then to describe the relationships between them. These elements may later become reference data items used for further semantic modeling during the engineering project. Identified reference data items are organized into the structures of two main types - classifiers and breakdowns, reflecting the logical and/or physical relations intrinsic to a knowledge domain. Besides basic classification and breakdown relationships, many other types of relationships can be added to reference data, turning a simple vocabulary-thesaurus into complex domain ontology.

If a document modeled contains requirements on the engineered system, requirements are modeled among other statements about reference data items as particular relationships between individuals and classes which are to be fulfilled at design decisions are made.

Specialized classifiers for classes and relationships used in the schema are registered as described in the paragraph **Elements of the Methodology reference data schema** .

Stage 1. **Identification of a modeled technical document**

Document modeling starts from the definition of the boundaries of the characterized text. It is a non-obvious choice: the main decision at this stage is the decision on the modeling of related documents.

Usually a technical document contains a lot of references to other technical documents – regulations, standards, technical studies and reports. A substantial part of identified reference data items (see the process **Description of Reference Classes**) can be described comprehensively only after a thorough reading of the external documents the characterized document is referencing to. Without such a detour the definition of many elements of reference data is possible only at the level «*is defined by the Regulation XXX*» without any disclosure of the exact meaning of the term. If this effort is prohibited by the modeling project – the definition by reference should be left in place. Of course such definitions can be properly unfolded later, when the referenced document is characterized itself.

Although such external documents are beyond the scope of the detailed modeling, reference data items referring to them should preserve the link to the source (i.e., should contain the reference outside the text characterized). Such sources should be registered in the structure model, as it is described in the manual **Document Structure Description**.

Stage 2. **Building a structural model of a document**

Process 2.1. **Structural modeling of a document**

Step 1. **Document item identification**

When modeling the structure of a document, the rules for document item identification depend heavily on the rules used during the document creation, and may vary for different organizations, industries, countries.

Identified items have to form a breakdown – a «part-whole» hierarchy. The characterized document as a whole is at the upper level of this hierarchy. At the lowest level are elementary text statements on the domain objects and their relationships, which are not decomposed further within the structural model. Between these are several levels of aggregation – element collections.

To create such a hierarchy the document is first separated into identifiable levels having some headers (and usually a number) – the document itself, its volumes, books, chapters, sections, subsection – if any.

Below the level of sections the document model may contain unnamed and/or unnumbered levels. Even below are levels of paragraphs and phrases, in tables, figures and diagrams. The way of identification for unnumbered levels and text blocks have to be agreed upon. For example, one can use a sequential numbering scheme in addition to the very last numbered level of document breakdown.

There is significant freedom in determination of a level at which decomposition stops. (Good guidelines for a particular situation can be obtained from requirements management methodologies and handbooks.) Ideally decomposition stops when we obtain simplest non-separable statements about knowledge domain objects in textual form, which are still sensible for humans, complete and interpretable. Depending on the document such statements may be present in paragraphs or phrases, in parts of a phrase, or in a table, or in a particular row in a table, in a diagram or in a figure. When it is unfeasible to break text into small elementary statements, it is recommended keep together statements related to the single subject.

At the lowest level we are interested in two types of statements – statements on “what is there” and statements on “what should be”.

In the first category are blocks of document text containing reference data – these are descriptions of domain entities (objects and relationships) and their “usual” relationships. In the technical specification document reference data statements are necessary for the definition of the statements of second type – requirements. Parts of the document containing only domain knowledge on reference data may be quite big. Elementary statements should satisfy the condition mentioned above – if possible, within one document item at the lowest level of the structural model only the descriptions related to a single subject or a single engineering discipline should be brought together.

For the statements in the second category (requirements to the engineered system) the breakdown is also completed when we obtain elementary non-separable requirements in textual form about «what should be». If possible, each elementary statement should contain exactly one essential requirement. If it is impossible, one elementary statement should combine only the requirements related to the single subject.

All items on all levels and every elementary statement of the document should be assigned unique identifiers. If the document itself (or its volumes or books) has a specific code in some external system of documents identification, such code can be used as a prefix in items’ identifiers. If possible, it is better not to use titles of chapters/sections/etc, but use their numbers instead. Hierarchical numbering is useful when it is necessary to create identifiers at the lowest levels of the document structure. The title of the document and all its part, chapter, sections and subsection titles should be used as the texts of corresponding intermediary level item, while at the lowest level it should be the text of the document itself (paragraph, phrase or part of a phrase).

Step 2. **Item subject identification**

Each elementary statement of the document should be linked to one (or probable several equally important) subject – the subject of the domain knowledge statement or a subject of a requirement. At upper and intermediate levels of document breakdown subjects are also desirable.

The subject should be individual object, relationship, class of objects or relationships, which is a subject of concern in this item. If some identified item or all items at some level deal with several equally important subjects, it is possible either directly address these subjects or address a common class, containing all those subjects as subclasses or parts. Such usage of the common class is a good practice for subjects on the upper and middle levels of breakdown.

An identified subject should be registered as a reference data item of the document model and should be characterized as described in the process **Reference Class Descriptions**.

The link between the document item and its subject is an important relationship between structural and semantic models of the document.

Step 3. **Classification of the document items**

Commonly used requirements management standards prescribe classification of the requirements by various classifiers, sometimes listed as «attributes of requirements» in corresponding data models. Sometimes particular set of classifiers is applied quite early, at the stage of the creation of a modelled document for humans.

In some cases it is decided to add one or more new classifications at the structural modeling stage, in addition to those used during the document creation.

All classes of these classifiers are included in the model and described in the core reference data description table as described in the process **Reference data item description**.

Manual 1. **Document structure description**

Document structure model, which reflects all decisions described above, is created in the TabLan data description table of the document structural model. Data description table of the structural model is located on the **Structure Model** tab of the model spreadsheet file.

The header of the **Structure Model** data description table:

Section	Item ID	Item Class	Item Text	Item of Whole	Item Subject	Classifier Class

The table is completed according to the following rules:

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

Item ID – a unique identifier of a document item, defined as described under Step 1 **Document element identification** above. This identifier should be unique within the document modeling project, and the best practice for its creation is a hierarchical numbering schema.
Mandatory.

Item Class – one of the following classes of the Methodology reference data schema (described in the section «Reference data schema elements» in core reference data description table):

- Document - for the document item as a whole;
- Element Collection - for items on intermediate levels of the structure hierarchy;
- Elementary Statement – for a non-separable elementary items of the document at the lowest level of the hierarchy.

Mandatory.

Item Text – a text string with the item's content:

- for a Document or an Element Collection – a header of a corresponding upper or intermediate level of the hierarchy, i.e. the title of the document, volume or book titles, chapter or section headers (without a number, which is preferably used as an Item ID cell). If a text header is absent (at a paragraph or even phrase level) the same number used as ID can be repeated.
- for an Elementary Statement – a text of the chapter, paragraph, or a particular phrase as decided according to description of steps above. If complex tables or figures are present, completion of this cell requires some serialization for these types of data or use of some system of external referencing, which is beyond of the scope of this Methodology.

Mandatory.

Item of Whole – a unique identifier of the document item on the previous level of structure hierarchy, that is the whole for the characterized item. For example, the identifier for the whole document is Item of Whole for a volume, the identifier of the chapter is Item of Whole for a section, etc. **Item of Whole** should be defined as an Item ID somewhere in the same **Structure Model** table.
Mandatory for the items classified as Element Collection and Elementary Statement.

Item Subject – a reference data item being the subject of the characterized document element. Entered exactly as it is described by English or Russian name in core reference data description table on **Class Definitions** tab.
Mandatory for the items classified as Elementary Statement.

Classifier Class – a class of additional classification for the characterized document item, selected from classes described in the core reference data description table. Entered exactly as it is described by English or Russian name in core reference data description table on **Class Definitions** tab.
Optional.

Any complete row characterizing a document item may be followed by zero or more rows with only the **Item Subject** or **Classifier Class** cells filled. In this case additional relationships are created between the item described in the last completely filled row and **Item Subject**s and/or **Classifier Class**es in additional rows, exactly as it has been done for a complete row.

The document structure model should include the descriptions of all the documents referenced in the document characterized or any other documents and reference data libraries which are used as sources of data in this Methodology. Therefore in addition to the characterized document and its parts, the **Structure Model** data description table should be supplemented with the data items of the Document class, representing all the external sources (see the stage **Identification of a modeled technical document**).

> *For example such documents can be ISO 15926 part 7, PCA reference data library, TechInvestLab.ru RD sandbox and a special dummy document entitled General Reference Data, which is used as a source for the reference data elements from non-specific sources.*

An alternative approach to build the document structure model can be taken, if the transformation of the textual document to some requirement management software format (IBM - Rational DOORS, for example) had been already carried out. In such case it is possible to map the data model of the software in question to the TabLan structure model (see the description of the ISO 15926 data model for TabLan data description tables in the accompanying document **«TabLan.15926 Data Mapping Specification»**). It is also possible to enhance TabLan document structure model to include other data present in the requirement management system. If an ISO 15926 adapter for such system already exists, the mapping is done as described in the methodology **«ISO 51926 Reference Data Engineering Methodology»** (http://techinvestlab.ru/RefDataEng).

Stage 3. **Building a semantic model of a document**

Process 3.1. **Reference data items description**

Semantic modeling of the characterized technical document starts with the identification and description of all entities necessary for the making of formal statements (including requirements) about a domain the document describes. Not only physical objects, but also all events, activities, signs, attributes along with their relationships will be considered "entities" here, as prescribed by ISO 15926 modeling approach to the world (which is based on 4D extensionalism). Entity identification is performed on the basis of characterized document textual data and (if agreed) on data from the documents referenced from it. Every entity identified is referenced to its source, which have to be an item of the document structural model described above. Necessary entities may also be extracted from reference data libraries used in the modeling project (the descriptions of all such elements should also be included in the data description table).

The entities that will be used in the formal statements of the semantic model become **reference data** of the document model, whereas the statement expressions themselves (described below in the processes **Complex relationship definition** and **Requirement definition**) become its **project data**.

Open-source TabLan extension for the .15926 Editor writes both reference and project data in one ISO 15926-8 RDF file and allows defining a namespace for reference and project data created. For a specific project the namespace and naming rules can be changed according to the rules of organization using the Methodology.

As a rule, during the requirements definition or system design stages of system's life-cycle major entities identified as subjects of technical document statements (or necessary for the making of such statements) are **classes of individuals**. In the ISO 15926 data model they will belong to the ClassOfIndividual entity type or its subtypes. Particular individuals – individual objects, events or actions – are practically absent at these stages of the life-cycle, as the design process is mainly focused on some generalized collections of phenomena, processes, objects, relationships. According to the ISO 15926 data model, documents and other objects of information nature are also modeled as classes. Therefore, the process description below is restricted to the identification and description of **reference classes of individuals (**ISO 15926, Part 2, sect. 4.8.3.1, page 36) and **reference classes of relationships** (ISO 15926, Part 2, sect. 4.8.3.3, page 37).

This Methodology uses a restricted set of relationships used to describe ontology of reference data entities (see the processes **Reference data classifier definition** and **Reference data breakdown definition**). Extending this set requires modification of this Methodology and changes to the open-source extension to the .15926 Editor.

Step 1. **Searching the reference data libraries**

It is likely that reference data entity identified in the document have close analog or is present as it is needed in some ISO 15926 reference data library (RDL). The list of external RDLs used in the project should be approved at the very beginning of the document characterization process, after consideration of their ownership, maintenance processes, availability to the modelers and potential users of the model. The list of libraries can be extended if necessary, but requires changes to the software tool configuration.

Open-source extension to the.15926 Editor can use as a source of external reference data either a local copy of the PCA reference data library or TechInvestLab.ru reference data sandbox available as SPARQL endpoint. The software can be modified for a specific project to use other reference data libraries in various formats.

If the required class has been found in an external RDL, it is always better to use it in document characterization, possibly providing a Russian translation of its name in the data description table for easier cross-referencing inside the table. The Russian name for an item from external RDL will be lost after TabLan table transformation to RDF/OWL.

Step 2. **Identification and naming of reference data**

Reference data identification requires assignment of a unique identifier in some namespace and definition of a unique meaningful name, preferable in Russian and in English.

Strict compliance to reference data naming rules requires the knowledge of the ISO 15926 Part 6 and is beyond the scope of this Methodology.

To facilitate further use the reference data item discovered in external reference data library in document characterization one should save in data description table its unique identifier and meaningful name from the source library.

Unique identifiers for newly created reference data items are assigned by the .15926 Editor during the TabLan tables' data conversion in the namespace defined in conversion settings. For a specific project the namespace and naming rules can be configured according to the needs of organization using this Methodology.

Step 3. **Reference data typing**

Every reference data item should be assigned one of the ISO 15926 Part 2 entity data types. Specific rules for the typing of a reference data require the knowledge of the ISO 15926 Part 2. Thorough description of the appropriate rules is beyond the scope of this Methodology. It is recommended to start identification of the entity type by the following guidelines:

Classes of material objects – **ClassOfFunctionalObject** or **ClassOfInanimatePhysicalObject**. Such classes are created to make statements about all the material entities – enterprises in general, machines, mechanisms, devices, facilities and their spare parts. They are also used

Classes of actions and phenomena – **ClassOfActivity**. Such classes are created to make statements about all the phenomena, processes or actions, persisting in time.

Classes of events –**ClassOfEvent**. Such classes are created to make statements about the events – marking changes in the state of objects, beginning and ending of activities, and similar.

Classes of documents – **ClassOfInformationObject**. Such classes are created to make statements about particular documents and their groups (starting from the project documentation sets and databases to particular reports, calculations, 2D documents and 3D models).

In order to make statements about the properties of entities identified in the characterized document, reference data also includes items belonging to the **ClassOfProperty** and **ClassOfIndirectProperty** types. It is often possible to find required classes in the external RDL (PCA is a good place to look for required items), as described in Step 1 above. See the details on the identification of such items in the manual **Reference data property description**.

Classes of relationships – **ClassOfRelationship** and its numerous subtypes. Such classes are created to make statements about various types of relationships between other identified entities – connections, assemblies, containment, temporal sequencing, cause-and-effect, etc.

After an identification of reference data items referred directly in the characterized document, it is recommended to add classification and/or specialization relationships as described in Step 4 below. This requires either identification of appropriate classes in external RDL, or their creation. Another source of additional reference data items is creation of complete classification and breakdown structures as described in the processes **Reference data classifier definition** and **Reference data breakdown definition**. All these processes lead to the identification and description of additional items, especially belonging to the **ClassOfClass** type and its numerous subtypes (**ClassOfClassOfIndividual, ClassOfClassOfRelationship** etc). It is important to remember that all reference data items used in all TabLan tables have to be identified and described according to this process!

.15926 Editor supports all correct ISO 15926 Part 2 type references without additional configuration.

Step 4. **Reference data classification**

Reference data items identified in the process of characterization of a technical document have to be included into the unified reference data taxonomy of ISO 15926. This taxonomy is constantly evolving as the ISO 15926 federated reference data libraries are developed. This inclusion is done by establishing classification (membership) and/or specialization (subclass-superclass) relationship between identified reference data item and another class from the modeled documents or from an external RDL.

The general good practice of ISO 15926 modeling is to have at least one classification for individual objects and at least one specialization for a class. As we've mentioned earlier, individuals rarely appear at early life-cycle stages this Methodology is mostly targeting.

It is recommended to continue the chain of such relationships until the classifier and/or superclass from external RDL is reached.

More details on the taxonomy building principles can be found in the ISO 15926 Part 6. The details are beyond the scope of this Methodology.

Many additional taxonomy trees necessary for the domain modeling can be created as described in the processes **Reference data classifier definition**.

Manual 2. **Core description of reference data**

All reference data items identified as outlined above are described in the TabLan core reference data description table. Core reference data description table is located on the **Class Definitions** tab of the model spreadsheet file.

The header of the **Class Definitions** data description table:

Section	URI	Russian Unique Class Name	English Unique Class Name	Part 2 Type	Superclass	Classifier Class	Source

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

URI – a unique identifier of the reference data item compliant with RDF recommendations and ISO 15926 Part 8 requirements. Should be entered for reference data items found in the external RDL (any item from external RDL have to be registered in the TabLan core reference data description table if it is used anywhere in the model).

> *TabLan open source extension of the .15926 Editor uses either a local copy of the PCA reference data library or TechInvestLab.ru reference data sandbox.*
> *- URI of PCA reference data item is entered in the form **rdl:RDS########**.*
> *- URI of TechInvestLab.ru sandbox data item is entered in the form **til:########**.*
> *The use of other reference data libraries for correct data visualization after data conversion requires changes to the TabLan extension of .15926 Editor.*

If the URI is not set – the item is created as project reference data and is assigned an URI in the project namespace defined in the Editor import settings.
Optional.

Russian Unique Class Name – unique meaningful Russian name identifier of the characterized item.

> *This name for an item from external RDL (with non-empty URI cell) will be lost after TabLan table transformation to RDF/OWL. It can be used for reference to this data item in the TabLan file though.*

Mandatory if English Unique Class Name is absent.

English Unique Class Name – unique meaningful English name identifier of the characterized item. If the item is found in an external RDL the name should be exactly as recorded in the library.
Mandatory if Russian Unique Class Name is absent.

Part 2 Type – an entity type of the characterized item as defined by ISO 15926 Part 2. The name should be written in the CamelCase format exactly as defined in the Part 7 of the standard.
Mandatory.

Superclass – a class for which the characterized item is a subclass. The name of the superclass should exactly match the English or Russian name in the corresponding cell of its description on the same **Class Definitions** tab.
Optional.

Classifier Class – a class for which the characterized item is a member. The name of the classifier should exactly match the English or Russian name in the corresponding cell of its description on the same **Class Definitions** tab.

Optional.

Source – a unique identifier (Item ID on the **Structure Model** tab) of an item in the structure model of the characterized document which is the information source for the characterized item. It can be the characterized document as a whole, any of its parts at any level of the structural hierarchy, any external document referenced in the given one, or any other source which is used to obtain reference data items (including external RDLs).
Mandatory.

Process 3.2. **Reference data items description** (continued)

Technical documents contain statements about properties that particular entities (objects, actions, phenomena or their classes) possess or should possess. According to the ISO 15926 data model modeling of a property is a two-step process. At the first step, the class of entities possessing that property is defined. At the second step, we claim that the entity in question belongs to the class of objects possessing that property or is related to it.

In this process we describe the general definition of the class of objects, actions or phenomena possessing some property.

The statements about the membership in these classes (about the actual or desired possession of these properties) – are described below in the processes **Complex relationship definition** and **Requirement definition** correspondingly.

More rigorous understanding of the peculiarities of property modeling requires the familiarity with the ISO 15926 Part 2.

Step 5. **Identification of a property possessing class**

Any mentioned in the characterized document class of objects, actions or phenomena possessing particular property should be first identified according to the Steps 1-4 of the first part of this process, and described in the core reference data description table on the **Class Definitions** tab of the model spreadsheet file. As a rule, the identification of a class possessing a property starts with the identification of a more general class, which is later specialized to possess that particular property.

If the characterized document refers to entities possessing more then one property, it is allowed to model a single class possessing all of them. The value for each property is described separately of course.

Classes possessing properties should be typed as **ClassOfIndividual** or its subtypes.

Step 6. **Identification of a property and of an indirect property**

To define a class possessing a property we should identify:

- The property type (in ISO 15926 Part 2 data model these are entities of **ClassOfProperty** type or more often one of its most used subtypes **SinglePropertyDimension** or **EnumeratedPropertySet**). The property type (“direct” property) is used when we are planning to make a statement about actual measurements of a corresponding property for the class described.

Temperature and Pressure are examples of a "direct" property used when we are making statements directly about some the measurements of a particular characteristic.

It is always recommended to search for a needed property in the external RDL approved for the use in the document modeling project (PCA RDL is a good place to find all required properties).

- And, if necessary, the indirect property type (in ISO 15926 Part 2 data model these are classes of relationship – entities of **ClassOfIndirectProperty** type). In ISO 15926 models indirect property is used when entities in question do not directly possess the property in a measurable way. Instead, indirect property is derived from some process, calculation or estimation, and does not directly apply.

  There can only be one temperature that a thing has (at a time), so a Maximum Allowable Working Temperature is not its temperature, but an indirect property derived from doing some tests or calculations to determine its value (as opposed to it being a current measurement).

  It is always recommended to search for a needed indirect property in the external RDL approved for the use in the document modeling project (PCA RDL contains many indirect property classes).

Direct and indirect properties should be also identified as reference data according to the Steps 1-4 of this process, and described in the core reference data description table on the **Class Definitions** tab of the model spreadsheet file according to the manual **Core description of reference data**. If desired direct or indirect property is found in an approved RDL, it should be referenced by its URI in the corresponding cell of this table.

Sometimes only some generalized class for required direct and/or indirect property can be found in the RDL. This class should be added to the model and after that it can be specialized by a necessary class and referenced as its superclasses.

Step 7. **Property range determination**

It is quite reasonable that, according to the ISO 15926 modeling rules, only individual (particular) objects, activities and phenomena can possess an exact single property value. All classes of objects, activities and phenomena used in the modeling of a life-cycle of a technical system can be characterized only by a range of any property. Technical documentation, however, often neglects this rule, and classes (or sets) of entities are described with «precise» property values without any note of the measurement errors or allowed tolerance. If it is impossible to deduce the confidence interval during the document characterization (internally or based on external sources), we have to assume that the value given is simultaneously an upper and lower bound of a property range in question.

Property values are used in TabLan modeling exactly as they appear in the text of the characterized document, with the numerical value and unit of measure put together as a name of a reference data item. This method leads to the creation of specific items in the model for each property value, which is not convenient if the model is further loaded to some CAD or PLM system. Further modeling steps are necessary for putting values and units of measure apart as separate items of the data model. These steps can be made later by the qualified data modeler, and are beyond the scope of this Methodology.

Manual 3. **Reference data property description**

All identified property values for the reference data items (classes) are described in the TabLan properties data description table (reference data items themselves should be described according to the manual **Core description of reference data**). Properties table is located on the **Class Properties** tab of the model spreadsheet file.

The header of the **Class Properties** data description table:

Section	Class Name	Property Name	Indirect Property Name	Property Min	Property Max	Source

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

Class Name – an English or Russian name of the characterized class (class possessing property or properties). The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Property Name – full English or Russian name of a property ("direct" property) the characterized class should possess or should be related to. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Indirect Property Name – full English or Russian name of an indirect property the characterized class should possess. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional.

Property Min – lower bound (minimum value) of the property range, indicated as a numerical value followed by a unit of measure.
Mandatory.

Property Max – upper bound (maximum value) of the property range, indicated as a numerical value followed by a unit of measure.
Mandatory.

Source – a unique identifier (Item ID on the **Structure Model** tab) of an item in the structure model of the characterized document which is the information source for the property described. It can be the characterized document as a whole, any of its parts at any level of the structural hierarchy, any external document referenced in the given one, or any other source which is used to obtain reference data items (including external RDLs).
Mandatory.

Process 3.3. R**eference data classifiers description**

Classifier is a simplest data structure used in information modeling, but its formal representation requires some attention. Let's introduce some concepts for further use.

The *classifier* is a set of classes related by *specialization* relationship between *superclass* and *subclass* (set-subset in the set theory language). (Classifier graph is not necessarily a tree; it can be any general acyclic graph.)

At the lowest level of a classifier are individual *members* of classes, which are connected with their "parents" by another relationship – *classification* relationship between class and its member (set-element in the set theory language). Class membership is defined outside of this process: membership statements can be recorded via **Classifier Class** cell in reference data description table on the **Class Definitions** tab of the model spreadsheet file.

Rather often the same classes or elements are included simultaneously into different specialization structures – to reflect different viewpoints or to organize multiple inheritances of properties. Multiple specializations and are characteristic feature of ISO 15926 modeling. To make distinguishing between various classifiers easier, it is useful to introduce for each modeled classifier a pair of special classes **<class of classifier relationships, class of classifier classes>**. These are used to collect together relationships and classes of a single classifier to make browsing, searches and other uses of data model easier at subsequent stages of system life-cycle.

The choice of classifiers (classes of individuals and classes of classes) is very important for each modeling project. This choice is often ambiguous, particularly at the level if class-class of class distinction. The discussion of 3-level modeling and its consequences is beyond the scope of this Methodology.

As described in the Step 4 of the process **Reference data item description**, it is recommended to identify a classifier and/or a superclass for a characterized class during the core description of reference data for this item, to include new items in the general taxonomy of ISO 15926. However the characterized document analyses may require the creation of different classifiers, dictated by the particular project peculiarities, system of standardization, coding system, etc. Therefore the creation of classifiers is separated into a separated process and supported with dedicated TabLan data description table.

Please remember that before a new classifier is modeled all of its classes should be defined according to the process **Reference data item description**.

Each step described below is performed for every classifier created in the model.

Step 1. **Classification relationship identification**

For each new classifier it is recommended to introduce a class of the classifier relationships. The class will contain only «class-subclass» relationships of this classifier. This is not a mandatory requirement, but later on it can be of great value for a verification and further usage of the data model.

The class of classifier relationships should be identified as reference data item according to the Steps 1-4 of the **Reference data item description** process, and described in the core reference data description table on the **Class Definitions** tab of the model spreadsheet file. Its entity type should be defined as **ClassOfSpecialization**. A single superclass for all relationships of all project classifiers also may be defined in the project reference data.

Step 2. **Class of classifier classes identification**

For the consolidating of already defined classes into a new classifier, it is recommended to introduce a class of classes of this classifier, i.e. the class whose only members will be the classes chosen for this classifier. This is not a mandatory requirement, but later on it can be of great value for a verification and usage of the data model.

Each class of classifier classes should be identified as reference data item according to the Steps 1-4 of the **Reference data item description** process, and described in the core reference data description table on the **Class Definitions** tab of the model spreadsheet file. Its entity type should be defined as **ClassOfClass** or its subtypes if appropriate. A single superclass for all classes of all project classifiers also may be defined in the project reference data.

Manual 4. **Classifier description**

All classifiers identified as outlined above are described in the TabLan classifiers description table. Classifiers description table is located on the **Classifier Definition** tab of the model spreadsheet file.

The header of the **Classifier Definition** data description table:

Section	Class Name	Superclass Name	Classifier Relationship Class	Class of Classifier Classes	Source

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

Class Name – an English or Russian name of the characterized class of the classifier described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Superclass Name – an English or Russian name of the superclass for the characterized class in the classifier described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional (should be left empty for the top element of the classifier).

Classifier Relationship Class – an English or Russian name of the class of relationships of the classifier described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional.

Class of Classifier Classes – an English or Russian name of the class of classifier classes for the classifier described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional.

Source – a unique identifier (Item ID on the **Structure Model** tab) of an item in the structure model of the characterized document which is the information source for the classification relationship

described. It can be the characterized document as a whole, any of its parts at any level of the structural hierarchy, any external document referenced in the given one, or any other source which is used to obtain reference data items or their relationships (including external RDLs).
Mandatory.

Process 3.4. **Reference data breakdowns description**

Decomposition into constituent parts is a traditional method of organization and analysis of engineering information. Hierarchical breakdown structures are results of such decompositions. System descriptions usually contain a lot of breakdown structures (called below simply "breakdowns"): functional, geometry, geography, work, document breakdowns, etc. In a breakdown reference data elements are connected by «part-whole» relationships. According to this Methodology, we are creating only classes of reference data for further use, therefore for classes these relationships are more correctly called "class of part – class of whole", meaning that members of a class of parts are parts of members of a class of wholes).

Identification of breakdown structures required for an engineering project is an essential part of the project preparation; therefore technical documents of the early design stages often either define some breakdowns, or inherit them from industry or discipline standards.

To make distinguishing between various breakdowns easier, it is useful to introduce for each modeled breakdown a pair of special classes **<class of breakdown relationships, class of breakdown classes>**. These are used to collect together relationships and classes of a single breakdown to make browsing, searches and other uses of data model easier at subsequent stages of system life-cycle.

Please remember that before a new breakdown is modeled all of its elements should be defined according to the process **Reference data item description**.

Each step described below is performed for every breakdown structure created in the model.

Step 1. **Breakdown relationship identification**

For each new breakdown it is recommended to introduce a class of the breakdown relationships. The class will contain only «part-whole» relationships of the classifier created. This is not a mandatory requirement, but later on it can be of great value for a verification and further usage of the data model.

The class of breakdown relationships should be identified as reference data item according to the Steps 1-4 of the **Reference data item description** process, and described in the core reference data description table on the **Class Definitions** tab of the model spreadsheet file. Its entity type should be defined as **ClassOfClassOfComposition**. A single superclass for all relationships of all project breakdowns also may be defined in the project reference data.

Step 2. **Class of breakdown classes identification**

For the consolidating of already defined classes into a breakdown structure, it is recommended to introduce a class of classes of this beakdown, i.e. the class whose only members will be the classes organized in this breakdown. This is not a mandatory requirement, but later on it can be of great value for a verification and usage of the data model.

Each class of breakdown classes should be identified as reference data item according to the Steps 1-4 of the **Reference data item description** process, and described in the core reference data description table on the **Class Definitions** tab of the model spreadsheet file. Its entity type should be defined as **ClassOfClass** or its subtypes if appropriate. A single superclass for all classes of all project breakdowns also may be defined in the project reference data.

Manual 5. **Breakdown description**

All breakdowns identified as outlined above are described in the TabLan breakdowns description table. Breakdowns description table is located on the **Breakdown Definition** tab of the model spreadsheet file.

The header of the **Breakdown Definition** data description table:

Section	Class Name	Whole Class Name	Breakdown Relationship Class	Class of Breakdown Classes	Source

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

Class Name – an English or Russian name of the characterized class of the breakdown described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Whole Class Name– an English or Russian name of the class of whole for which the characterized class is class of part in the breakdown described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional (should be left empty for the top element of the breakdown).

Breakdown Relationship Class –an English or Russian name of the class of relationships of the breakdown described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional.

Class of Breakdown Classes – an English or Russian name of the class of breakdown classes for the breakdown described. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Optional.

Source – a unique identifier (Item ID on the **Structure Model** tab) of an item in the structure model of the characterized document which is the information source for the breakdown relationship described. It can be the characterized document as a whole, any of its parts at any level of the structural hierarchy, any external document referenced in the given one, or any other source which is used to obtain reference data items or their relationships (including external RDLs).
Mandatory.

Process 3.5. **Complex relationships description**

In addition to information about basic classifiers and breakdowns, the characterized document may contain more complex statements about other relationships between reference data items.

The set of complex relationship kinds described below is defined in the open-source TabLan extension of the .15926 Editor. Extending this set requires extension of the Methodology and changes to the extension code.

Each complex relationship elicited from the characterized document is designed to bind two reference data items (classes), assigning them the roles to play – role 1 and role 2. This syntax and list of relationships are inspired by Gellish engineering language (http://sourceforge.net/apps/trac/gellish/).

Please remember that before any complex relationship is modeled its participating entities should be described in the model according to the process **Reference data item description**.

Step 1. **Relationship kind identification**

The following complex relationship kinds can be identified in the characterized document analysis:

a. **«is described by»**

Is used to make a statement that a reference data item in role 1 (any class of objects, persons or phenomena) is described by the document (an entity of **ClassOfInformationObject** type) in role 2.

b. **«participates in»**

Is used to make a statement that a reference data item in role 1 (any class of objects or persons) participates in an activity or phenomena (an entity of **ClassOfActivity** type) in role 2.

c. **«is a predecessor in time of»**

Is used to make a statement that a reference data item in role 1 (any class of objects or phenomena) precedes in time another reference data item (also any class of objects or phenomena) in role 2. This relationship kind is most often used to describe dependencies between activities, phenomena or events (entities of **ClassOfActivity** or **ClassOfEvent** types). But in some cases it is possible to use this relationship for objects.

d. **«has as part»**

Is used to make a statement that a reference data item in role 1 (any class of objects, persons or phenomena) has as a part some other reference data item (also any class of objects, persons or phenomena) in role 2. The relationship can be used if creation of a complete breakdown as described above is unreasonable. This relationship is most often used to describe material objects (entities of **ClassOfIndividual** type) or documents (entities of **ClassOfInformationObject** type). But in some cases it is recommended to use this relationship for actions also.

e. **«is disjoint»**

Is used to make a statement that a reference data item in role 1 (any class of objects, persons or phenomena) has no common elements (as a class, not as an assembly!) with the reference data element (also any class of objects, persons or phenomena) in role 2.

f. **«is performed by»**

Is used to make a statement that an activity represented by a reference data item in role 1 (an entity of **ClassOfActivity** type) is performed by the reference data item (an entity of **ClassOfIndividual** type) in role 2.

g. **«is related to»**

Is used to make a general statement that a reference data item in role 1 (any class of objects, persons or phenomena) is somehow related to the reference data item (also any class of objects, persons or phenomena) in role 2. This relationship kind is used to make statements about dependencies between objects, persons, activities or phenomena not falling into any relationship kind described above.

Step 2. **Cardinality determination**

During the analysis of the characterized document one might find statements about cardinalities of roles in identified relationships, or deduce them to achieve a more rigorous modeling.

Cardinalities are not taken into account for **is described by** and **is disjoint** relationship kinds.

Manual 6. **Complex relationship description**

Complex relationships identified as outlined above are described in the TabLan other relationships description table. Other relationships description table is located on the **Other Relationships** tab of the model spreadsheet file.

The header of the **Other Relationships** data description table:

Section	Role 1	Role 1 Cardinality	Relates to	Role 2	Role 2 Cardinality	Source

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

Role 1 – an English or Russian name of the class in the role 1 of the described relationship. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Role 1 Cardinality – cardinality of the role 1 in the relationship. Is written in the format **(#:#)**, where # is a natural number (including 0) or * in a second position for infinity (cardinality ***(*:*)*** should be simply omitted).
Optional.

Relates to – the kind of the described relationship. This column can contain only the strings identifying kinds supported by the software used for table conversion. The open-source TabLan extension of the .15926 Editor supports relationship kinds listed in the Step 1 **Relationship kind identification** above. These are identified with the following strings:

is described by
participates in
is a predecessor in time of
has as part
is disjoint
is performed by
is related to

Mandatory.

Role 2 – an English or Russian name of the class in the role 2 of the described relationship. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Role 2 Cardinality – cardinality of the role 2 in the relationship. Is written in the format **(#:#)**, where # is a natural number (including 0) or * in a second position for infinity (cardinality ***(*:*)*** should be simply omitted).
Optional.

Source – a unique identifier (Item ID on the **Structure Model** tab) of an item in the structure model of the characterized document which is the information source for the described relationship. It can be the characterized document as a whole, any of its parts at any level of the structural hierarchy, any external document referenced in the given one, or any other source which is used to obtain reference data items or their relationships (including external RDLs).
Mandatory.

Process 3.6. **Requirements description**

Technical documents at early stages of system lifecycle often contain requirements to the system of interest, its components, or performance. Thorough analysis of the characterized document helps to identify statements about objects or activities carrying the obligatory status, "what should be achieved as design decisions are made". Such statements should be identified, elicited and described in a semantic model of requirements suitable for machine processing. Requirements are semantically modeled as usual relationships between reference data items, additionally classified with a particular modality class, degree of "obligatoriness" or "necessity".

The set of requirement kinds described below is defined in the open-source TabLan extension of the .15926 Editor. Extending this set requires extension of the Methodology and changes to the extension.

Each requirement elicited from the characterized document is designed to bind two reference data items (classes), assigning them the roles to play – role 1 and role 2.

Please remember that before any requirement is modeled its participating entities should be described in the model according to the process **Reference data item description**.

Step 1. **Requirement kind identification**

The following requirement kinds can be identified in the characterized document analysis:

a. **«is classified as»**

Is used to require that a reference data item in role 1 (any class of objects, persons or phenomena) should belong (is a member) to a class (an entity of **ClassOfClass** type or its subtypes) in role 2.

b. **«is subclass of»**

Is used to require that a reference data item in role 1 (any class of objects, persons or phenomena) should be a subclass of a class (of objects, persons or phenomena) in role 2.

The most important application of this requirement kind is the following – it is used to express a requirement for particular properties of some object or phenomena.

c. **«complies to description in»**

Is used to require that a reference data item in role 1 (any class of objects, persons or phenomena) should comply with a description in a document (an entity of **ClassOfInformationObject** type) in role 2.

d. **«participates in»**

Is used to require that a reference data item in role 1 (any class of objects or persons) should participate in an activity or phenomena (an entity of **ClassOfActivity** type) in role 2.

e. **«is a predecessor in time of»**

Is used to require that a reference data item in role 1 (any class of objects or phenomena) should precede in time another reference data item (also any class of objects or phenomena) in role 2. This requirement kind is most often used to require dependencies between activities, phenomena or events (entities of **ClassOfActivity** or **ClassOfEvent** types). But in some cases it is possible to use this requirement for objects.

f. **«has as part»**

Is used to require that a reference data item in role 1 (any class of objects, persons or phenomena) should include as a part some other reference data item (also any class of objects, persons or phenomena) in role 2. This requirement is most often used for material objects (entities of **ClassOfIndividual** type) or documents (entities of **ClassOfInformationObject** type). But in some cases it is recommended to use this requirement for actions also.

g. **«is performed by»**

Is used to require that an activity represented by a reference data item in role 1 (an entity of **ClassOfActivity** type) should be performed by the reference data item (an entity of **ClassOfIndividual** type) in role 2.

h. **«is related to»**

Is used to make a general requirement that a reference data item in role 1 (any class of objects, persons or phenomena) should be somehow related to the reference data item (also any class of objects, persons or phenomena) in role 2. This requirement kind is used to require dependencies between objects, persons, activities or phenomena not falling into any requirement kind described above.

Step 2. Modality determination

Classification of a relationship by a degree of "necessity" is distinguishing requirements from other statements about reference data. In the semantic model items from a special modality classifier are used to declare that requirements belong to one of special classes of "necessity".

For each identified requirement it is necessary to decide what class of modality will contain that requirement.

The following classes are included in the modality classifier «**Requirement modality»**:

«Mandatory»
«Optional»
«In exceptional cases»
«Desirably»

The «**Requirement modality»** classifier and its classes are the elements of the Methodology reference data schema described in the section «Reference data schema elements» in core reference data description table. All classes in this classifier has **ClassOfAssertion** entity type. It is possible to expand this classifier by adding additional reference data items to this section.

Step 3. **Cardinality determination**

In the analysis of the requirements one might identify statements about cardinalities for roles in required relationships, or deduce them to achieve a more rigorous modeling.

Cardinalities are not taken into account for **is classified as, is subclass of** and **complies to description in** requirement kinds.

Manual 7. **Requirement description**

Requirements identified as outlined above are described in the TabLan requirement description table. Requirement description table is located on the **Requirements** tab of the model spreadsheet file.

The header of the **Requirements** data description table:

Section	Statement Classification	Role 1	Role 1 Cardinality	Relates to	Role 2	Role 2 Cardinality	Source

Section – a comment cell referring to the group of subsequent rows and used to increase the readability of big data description tables. All other cells in the row with non-empty **Section** cell are ignored.
Optional.

Statement Classification – an English or Russian name of the class from the modality classifier. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Role 1 – an English or Russian name of the class in the role 1 of the described requirement. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Role 1 Cardinality – cardinality of the role 1 in the requirement. Is written in the format **(#:#)**, where # is a natural number (including 0) or * in a second position for infinity (cardinality ***(*:*)*** should be simply omitted).
Optional.

Relates to – the kind of the described requirement. This column can contain only the strings identifying kinds supported by the software used for table conversion. The open-source TabLan extension of the .15926 Editor supports requirement kinds listed in the Step 1 **Requirement kind identification** above. These are identified with the following strings:

is classified as
is subclass of
complies to description in
participates in
is a predecessor in time of
has as part
is performed by
is related to

Mandatory.

Role 2 – an English or Russian name of the class in the role 2 of the described requirement. The name should exactly match the name of the item in core reference data description table on **Class Definitions** tab.
Mandatory.

Role 2 Cardinality – cardinality of the role 2 in the requirement. Is written in the format **(#:#)**, where # is a natural number (including 0) or * in a second position for infinity (cardinality ***(*:*)*** should be simply omitted).
Optional.

Source – a unique identifier (Item ID on the **Structure Model** tab) of an item in the structure model of the characterized document which is the information source for the described requirement. It can be the characterized document as a whole, any of its parts at any level of the structural hierarchy, or any external document referenced in the given one.
Mandatory.

Stage 4. **Use of a document model**

TabLan data description tables with a model of a characterized document are transformed into the ISO 15926 – compliant data model in RDF/OWL file. This model can be imported into other software (CAD\CAM, PLM or EAM, ERP, etc.) and used for project data organization and verification. Such import can be done either via ISO 15926 adapter (iRING Tools component, for example) or by transformation directly to target data model.

This Methodology describes only the usage of the TabLan model to obtain exchange files compliant with ISO 15926 part 8.

Process 4.1. **Transforming tables into ISO 15926 format**

Step 1. **Choice of a namespace**

To obtain a data model in RDF/OWL format compliant to ISO 15926 Part 8, one should select a **project namespace** that will hold identifiers of all new reference data items created in the modeling process.

Understanding the principles of namespace assignment assumes some familiarity with the requirements of ISO 15926 Part 8 and RDF specification. Detailed description of these standards and rules is beyond the scope of this Methodology.

Import settings in the .15926 Editor allows configuration of a project namespace. For a specific project the namespace and naming rules should be changed according to the rules of organization using the Methodology.

Step 2. **Translation into a machine-readable form and review**

TabLan tables' transformation into RDF/OWL is done by the open-source TabLan extension of the .15926 Editor (available at http://techinvestlab.ru/dot15926Editor). The Editor also does a visualization of ISO 15926 models. The software is developed by TechInvestLab.ru. Licenses, terms of distribution and usage can be found with software package.

TabLan data is linked to PCA and to TechInvestLab.ru sandbox reference data, and p7tpl templates are instanced during the conversion. The open-source TabLan extension of the .15926 Editor is designed to use a local copy of the PCA reference data library and enhanced local copy of the template definition file p7tpl.owl. It can connect to TechInvestLab.ru sandbox SPARQL query interface.

Special attributes are used to identify alternate Russian and English names of an entity. They are defined in the *http://techinvestlab.ru/meta#* namespace.

Reference data libraries, languages used for naming, language attribute mapping and other aspects of software configuration can be changed while configuring TabLan extension of the .15926 Editor for specific project.

Manual 8. **Data table transformation**

Download the .15926 Editor from http://techinvestlab.ru/dot15926Editor and install it on your computer as described in the documentation.

You will need **PCA-RDL.owl.zip** file downloaded from http://rds.posccaesar.org/downloads/PCA-RDL.owl.zip (the file is sometimes updated, use the newest version available) and unzipped to get **RDL.owl** file.

The file **p7tpl_enhanced.owl** for use with this Methodology is distributed together with this document. It contains some template definitions developed specifically for the Methodology data model and not found in original p7tpl.owl file.

In .15926 Editor menu command *Import - Setup TabLan dependencies…* opens form for catalog data import settings.

Select **p7tpl_enhanced.owl** as template definitions file.

Select **RDL.owl** as file with RDL used in import.

Fill in the namespace for new entities or leave default namespace in the field.

Then go to *Import – Import TabLan reference data from.xslx table…* menu command and choose TabLan data model *.xlsx* file. Wait while template set and RDL are loaded. Do not close RDL or template set data sources until import process ends, or import will fail.

Imported data source is marked with *Imported file:…* name in the Project panel. If you want to see all named entities in the model – search for an empty string in the search box at the top of the panel (just click at the box and hit *Enter*). You can unfold the data tree nodes and see unnamed relationships and template instances created during transformation.

You will find unrecognized entities (marked by ?) in some relationships. These are reference data items from TechInvestLab.ru sandbox. To see them properly add TechInvestLab.ru endpoint to the Project panel through *Common endpoints - Reference data from TechInvestLab sandbox…* menu command. Then you can select any unrecognized entity and use *Edit - Search endpoints for URI (F4)* menu command. Or simply open TechInvestLab.ru endpoint in a separate panel and search for an empty string in the search box at the top of the panel. After all sandbox entities are added to the view you can reload unrecognized entities in import results.

Full instructions for unrecognized data identification can be found in .15926 Editor documentation. Refer to this documentation to learn more about browsing and querying the model.

You have to use *Save as…* (*Ctrl+Shift+S*) command to save imported data to a new .rdf file. Or you can change something in the source table, save it and repeat import with *File - Reimport* command.

Full description of the process with some screenshots can be found in .15926 Editor documentation. Refer to this documentation to learn more about browsing and querying the model.

Process 4.2. **Import to other information systems**

The semantic model of characterized document in ISO 15926 Part 8 format could be imported into other software as classifiers, breakdowns or just as project data. Import requires mapping of the target information system data model to ISO 15926. General description of this process you can find in the «**ISO 15926 Reference Data Engineering Methodology**» prepared by TechInvestLab.ru (http://techinvestlab.ru/RefDataEng).

***