INTERNATIONAL STANDARD

ISO 14739-1

First edition
2014-12-15

Document management — 3D use of Product Representation Compact (PRC) format —

Part 1: PRC 10001

Gestion de documents — Utilisation en 3D du format compact de représentation de produit (PRC) —
Partie 1: PRC 10001

Reference number
ISO 14739-1:2014(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/TC 171, *Document management applications*, Subcommittee SC 2, *Application issues*.

Introduction

The data representations in PRC allows 3D design data, typically created in CAD and PLM systems, to be viewed and interrogated by visualization applications and to be integrated into complex documents.

This document specifies a wide range of data forms. The wide range is necessary to:

— Achieve a high fidelity, visually equivalent representation of 3D design data produced by an advanced CAD or PLM system without requiring the original application.

— Allow applications to compute high accuracy product shape measurements.

PRC is intended to complement native or open standard CAD and PLM formats as a compact, concise binary form for visualization and documentation. PRC is not intended as a data format for CAD interoperability or use in factory automation systems, e.g. automated manufacturing and inspection systems, which is addressed by the ISO 10303 standards.

Document management — 3D use of Product Representation Compact (PRC) format —

Part 1:
PRC 10001

1 Scope

This International Standard describes PRC 10001 of a product representation compact (PRC) file format for three dimensional (3D) content data. This format is designed to be included in PDF (ISO 32000) and other similar document formats for the purpose of 3D visualization and exchange. It can be used for creating, viewing, and distributing 3D data in document exchange workflows. It is optimized to store, load, and display various kinds of 3D data, especially that coming from computer aided design (CAD) systems.

This International Standard does not apply to:

- Method of electronic distribution
- Converting CAD system generated datasets to the PRC format
- Specific technical design, user interface, implementation, or operational details of rendering
- Required computer hardware and/or operating systems

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 12651:1999, *Electronic imaging — Vocabulary*

ISO 24517-1:2008, *Document management — Engineering document format using PDF — Part 1: Use of PDF 1.6 (PDF/E-1)*

ISO 32000, *Document management — Portable document format*

IEEE 754, *Floating-Point Arithmetic*

The OpenGL Graphics System, A Specification, Version 4.1 (Core Profile), July 25, 2010[1]

[1] Available at http://www.opengl.org/registry/doc/glspec41.core.20100725.pdf

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 32000-1, ISO 24517-1 and ISO 12651 and the following apply.

3.1
PRC File Writer
software application which writes a particular PRC file

3.2
PRC File Reader
software application which reads a particular PRC file

3.3
byte
group of eight bits processed as a single unit of data

4 Document syntax conventions

4.1 Conventions

The following conventions are used within this document to describe data within a PRC File.

Terms highlighted in bold within this document signify field names in the description of entity types. Entity types are denoted in *italic*.

A table with three columns is used to describe the data within a contiguous portion of the file.

The first column indicates the name of the field. Field names are not unique and can be considered to have a scope limited to the data class.

The second column describes the type of the data. This might be

— A simple data type such as a Boolean, UnsignedInteger, or Double

— A simple class of data such as PRC_TYPE_TOPO_Body or PtrTopology where the name of the class is used to define the data stored for that class

— An Array<data class>[<size>] which indicates an array of data of the specified class. An array has <size> elements. Elements of an array are referenced beginning at 0.

The third column indicates if the field is required or optional. If the field is optional a condition is described when the field is present. The field may also be described.

4.2 Example Structure

Table 1 — PRC file structure example

Name	Type	Value
flag1	*Boolean*	(Required) describe **flag1**
data_field1	*<data class1>*	(Optional; if **flag1** is TRUE) describe data_field1
data_field2	*<data class2>*	(Optional; if **flag1** is FALSE) describe data_field2
topological_body_data	*PRC_TYPE_TOPO_Body*	(Required) describe **topological_body_data**
data_type	*<enumerated type> or Integer*	(Required) describe **data_type**
data_of_type3	*<data class3>*	(Optional; if **data_type** is 1) describe **data_type3**
data_of_type4	*<data class4>*	(Optional; if **data_type** is 2) describe **data_type4**
Size	*UnsignedInteger*	(Required) describe **size**
array_of_type5	*Array<data class5>*[**size**]	(Required) describe the array of <data class5>

5 PRC file concepts

5.1 The PRC file

A PRC File is a sequential binary file, written in a way to make the file portable across machine architectures and operating systems.

PRC is optimized to store various kinds of 3D data, especially those coming from computer-aided design (CAD) systems. Most of the main constructs of CAD systems are supported within the PRC File Format:

— Assemblies and parts

— Trees of 3D entities (coordinate systems, wireframes, surfaces, and solids)

— Exact geometry representation for curves, surfaces

— Tessellated (triangulated) representation

— Markup data

PRC is meant to be multipurpose. There are two ways to store exact geometry and tessellation depending on the usage of the file and on the original information:

— Regular compression is used to directly represent CAD data without loss or transformation from the originating CAD system.

— High compression is used to store very small files, which have a specified physical tolerance from the originating shape. The tolerance is typically 0,001 mm for exact geometric data and 0,01 mm for tessellation data.

Each PRC File corresponds to a single model file (see 8.3.3) which defines the root product occurrences within the FileStructures of the PRC File. A PRC File is a collection of FileStructures which are independent from each other and can come from various authoring PRC File Writers. A FileStructure is the representation of an independent physical file denoting an independent 3D part, assembly, etc. within a PRC file. Hence, there is one header for each FileStructure with specific information and one global header for the model file which contains information about the PRC File Writer that assembled all these individual FileStructures into the final PRC File. Header sections gather primarily information on file version, FileStructure ids and offsets for reading / skipping sections.

Each FileStructure contains one or more product occurrences (see 8.3.10.2). The product occurrences denote the assembly hierarchy of the FileStructure. A product occurrence can have child nodes, which are also product occurrences. There is exactly one root product occurrence in each FileStructure. The root product occurrence is the only entry point to the FileStructure. (refer to Figure 1 below)

In parallel to the FileStructures, the model file also contains an array of root product occurrences that comprise the starting point for the entire assembly description. A product occurrence may refer to a corresponding part definition (see 8.3.11) which in turn contains:

— Geometrical data stored in a tree of representation items (see 8.6.3)

— An optional tree of part markups, grouped into annotation entities and views (see 8.7.2)

A product occurrence can contain:

— A tree of product markups

— Filters used to redefine the loading and presentation of data defined in the child product occurrences or in the part definition (see 8.3.12)

The representation items are defined through a combination of tessellation or topology and geometry data, which may be highly compressed. The markups are defined by tessellation data.

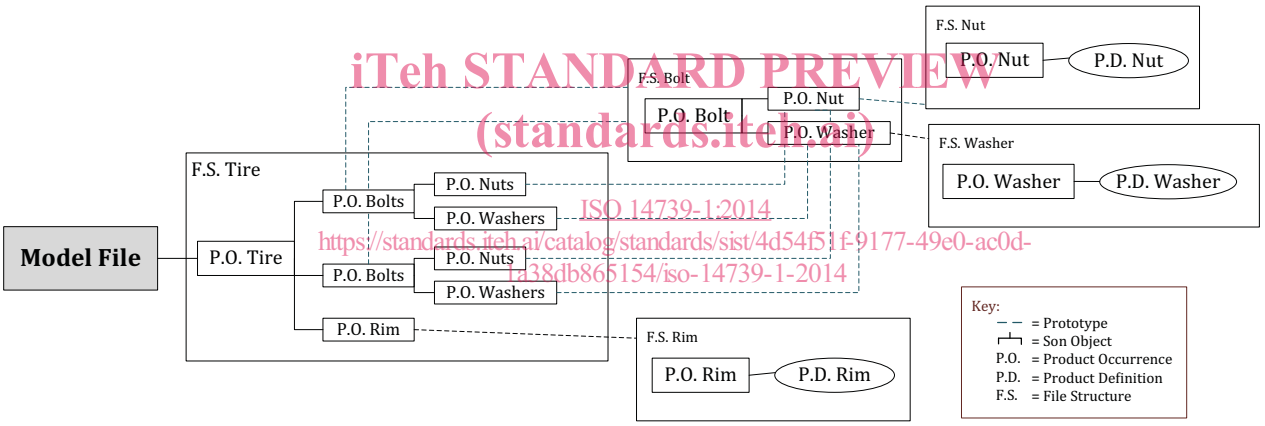

Figure 1 —Tessellation data

To optimize reading, a PRC File is arranged so that referenced entities are read before being referenced. Therefore, the FileStructures are ordered using parts, then subassemblies, and finally, the top (root) assembly.

A PRC File is composed of one header section, which starts with uncompressed data, one or more FileStructures, and one model file (*PRC_TYPE_ASM_ModelFile*) data section at the end, each individually compressed. Refer to 8.3.10 for more information about the PRC file structure.

Table 2 — PRC file structure

Section	Sub Sections	Compression	Description
FileHeader		Uncompressed	Defines originating author of the file; specifies start, and possibly end, location of other sections in file
File Structure 1	*FileStructureHeader*	Uncompressed	Identifiers of other File Structures referenced from within this FileStructure;
	Schema	Compressed	Description of changes between minimal version and authoring version of the FileStructure within the PRC File Format Specification
	Globals	Compressed	Referenced FileStructures and colors, line styles, and coordinate systems for each tree entity of the FileStructure
	Tree	Compressed	a description of the tree of items (product occurrences, part definitions, representation items, and markup)
	Tesselation	Compressed	All tessellated (triangulated) data in the leaf entities of the tree (representation items and markups).
	Geometry	Compressed	All exact geometry and topology data of the leaf entities of the tree(representation items)
	Extra Geometry	Compressed	Geometry summary data, which allow for partial loading of the FileStructure without loading the entire geometry
			Additional FileStructure sections in the PRC File
File Structure N		Compressed	Last FileStructure section in the PRC File
Schema	*ModelFile Schema*	Compressed	Description of changes between minimal version and authoring version of the ModelFile Section of the PRC File Format Specification
Model File Data	*PRC_TYPE_ASM_ModelFile*	Compressed	

5.2 Versioning

A file format version number is used to define the particular version of this international standard that a PRC File Reader or PRC File Writer conforms to.

Version numbers consist of the year modulo 2000 followed by three digits representing the day of the year. This international standard shall have the version 10001, corresponding to the 1st day of the year 2010.

The **current_version** is the maximum version number that a PRC File Reader or PRC File Writer conforms to.

Each FileStructure and the FileHeader are independent and can be copied around or written by PRC File Writers of different versions. Each of them have their own **authoring_version** and **minimal_version_for_read** version numbers. The **authoring_version** represents the version of the international standard that the PRC File Writer conformed to at the time the FileHeader or the FileStructure was written. The **minimal_version_for_read** represents the version of the international standard that a PRC File Reader shall conform to to successfully read the FileHeader or the FileStructure. If the **current_version** of the PRC File Reader is less than the **minimal_version_for_read**, then the PRC File Reader shall not continue to process the file and report an error.

If **minimal_version_for_read** is lower than **authoring_version**, the PRC File Writer shall write a schema description in the PRC File providing the differences between the two versions of this international standard. This description will enable a PRC File Reader to read and skip new information, since these new data cannot be interpreted as they are from a newer version.

A PRC File schema contains a description of new fields or new entity types added between the **minimal_version_for_read** and the **authoring_version**. See 6.3 for a description of a schema.

A PRC File Reader uses the information in the schema to read and skip new information:

— After reading each entity type, the schema information is queried and new data are skipped accordingly, following the tokens.

— Each time an entity type is read, if the type is unknown, the schema is searched and its data is skipped.

5.3 Unique identifiers

5.3.1 General

A PRC File reader/writer shall generate (writer) or interprete (reader) unique identifiers for information within the PRC File. Each FileStructure within a PRC File has an identifier (UUID) which uniquely identifies this particular FileStructure among all of the FileStructures within the PRC File. Within each FileStructure, unique identifiers for referenceable entities are generated using the order that they are first encountered in the FileStructure. Thus, the first referenceable entity in the FileStructure has the number 1 as its identifier. Subsequent identifiers of referenceable entities are incremented by 1.

5.3.2 File structure

A PRC FileStructure is identified by an identifier (see *UncompressedUniqueId*) which is unique among all of the FileStructures within a single PRC File.

The method to calculate a unique identifier for a given FileStructure is not part of this international standard.

This approach offers an advantage, for example, such as when there is some intent to repurpose FileStructures inside other PRC files without entirely rewriting them.

5.3.3 Base entities

The PRC format provides support for referencing entities. See Entity Types in 8.2.1 for a list of entity types whose entities are referenceable.

The purpose of using references on entities is to enable an interpreter to handle the same entity several times without any duplication of data, either by any other program, or by another structure in the same or another PRC file.

A referenceable entity is retrieved using the following:

— The UUID of the FileStructure.

— The entity's unique identifier (a non-zero unsigned 32-bit integer) within the FileStructure.

Just as the FileStructure UUID is unique among all of the FileStructures in the PRC File, the entity identifier is unique inside a FileStructure for all referenceable entities.

A PRC File Writer shall ensure that two referenceable entities inside the same FileStructure do not have the same identifier. The next available index within a FileStructure is the maximum value that has been assigned for identifiers to date, and is stored in the PRC File (see 8.3.4) so it is possible to safely add new entities to a FileStructure and assign them a unique identifier greater than this maximum value.

Data within the FileStructure tessellation and geometry sections are not accessible from outside the FileStructure using the approach for referenceable entities. These data are referenced only by data within the tree section (see 5.1 above) of the same FileStructure. However, it is possible (see 8.4.8) to reference topological data from outside the particular FileStructure which the topological entity lies in. This is restricted to faces in the current version of PRC but may be extended to other data in future versions of the format.

5.3.4 Other systems

In addition to the unique identifier mechanisms described above, this international standard provides for storage of identifiers from the originating system. The external identifiers and their persistence flag are stored as information in PRC strictly to support external workflows. These identifiers by themselves only exist to convey information and do not play any role in PRC.

5.4 Current data values

A PRC File reader and writer shall implement the concept of **CURRENT** for various data.

NOTE This enables smaller file sizes since duplicate data need not be written to the file. It also enables faster readers because some of the data is not being read. Default initial values are in parentheses.

— Current name (NULL)

— Current graphics

  — Index of layer (-1)

  — Index of line style (-1)

  — Behavior bit field (-1)

Current values shall be updated as they are encountered in the file. Values shall be reset when serialization is flushed at the end of every flate-compressed section (see 5.8 Compressed File Sections).

5.5 Userdata

The PRC File format provides a mechanism for a PRC File Writer to write private data within various sections of a PRC File. Such data shall consist of a bit stream of data containing the size of the bit stream followed by the specified number of bits.

NOTE Any PRC File Reader can read the bit stream and can even resave the private data, but it may not be able to interpret the data.

Each FileStructure within a PRC File may contain UserData. UserData are defined in conjunction with an application unique identifier (UUID) which allows for their interpretation. This application identifier

shall be stored in the FileStructurerHeading. By default, any user data within the FileStructure shall be assumed to be written by this one writer. However, UserData may contain streams from different writers. Therefore, different application UUIDs shall be stored accordingly with each of these other data. UserData are meant to be interpreted only by software which is aware of its meaning according to a given writers UUID. A conforming PRC File Reader should either ignore those UserData, or interpret them according to the writers UUID. PRC also allows applications to define special attributes which behave similarly to UserData,as discussed in 7.3.3.8.

Unique application identifiers are assigned through Adobe Systems, initially, and probably ISO in the long term.. Each company should have it's own unique application identifier, and if they want to share data with another company, they should share application identifiers.

5.6 Units

A conforming writer shall define the units used in a PRC File. This should be done at both the model file level (see 8.3.3) and at the product occurrence level (see 8.3.10). The unit may come from an actual CAD file and therefore be considered reliable to represent physical values in the data. If a unit is valid/reliable, the flag unit_from_CAD_file shall be set to TRUE. However, some formats do not contain units. Regardless, it is mandatory to define one unit at both the model file level and at the product occurrence level .

The unit which shall be used is the first valid unit in the ModelFile / ProductOccurrence chain. If a valid unit is defined at the ModelFile level, it will apply to all product occurrences. Once a valid unit is found, the remainder of the data shall be interpreted with respect of that unit, even for occurrences higher in the product occurrence hierarchy.

NOTE In other words, if a product occurrence having no valid unit has a child with a valid unit, it is assumed that the entire hierarchy of model file and product occurrence are to be interpreted and used according to this unit.

If no valid unit is found at either the ModelFile level or any ProductOccurrence level (i.e. unit_from_CAD_file is always set to FALSE), a conforming PRC File Reader shall clearly indicate that the unit defined is not valid for measurement purposes.

5.7 Tolerances

PRC distinguishes between several notions of tolerances:

— A first notion of tolerance represents the maximum deviation between compressed and original data, as introduced by the lossy compression of geometry or topology. When provided, this tolerance shall be a user-defined value which represents a physical length given with a unit.

— A second notion of tolerance is introduced by numerical uncertainty inherent in every 3D modelling system. This form of tolerance is non-dimensional. Its purpose is to perform consistent numerical operations. This tolerance value corresponds to coincidence (e.g. of two 3D vertices) and is generally defined in conjunction with a minimal value representing zero and a maximal value representing infinity. For instance, a system might define coincidence at 1e-3, zero as any value less than 1e-12 and infinity as any value greater than 1e6. Then, additional logic outside the modelling system should define the unit so that the numerical values can be interpreted by the computer as physical values (i.e. in a particular unit).

In PRC, the 3D modelling system corresponds to entities in tessellation section (8.8) and topology section (7.9). Hence the various tolerances stored within these sections are always numerical values with no unit. A conforming writer shall never store a tolerance which can be directly interpreted as a physical value.

NOTE For instance, brep_data_compressed_tolerance in 8.9.19 which represents the deviation introduced between original data and compressed one is stored without unit, even if it might be derived from a user interface

which takes those units into account. Then, outside this 3D modelling system, unit is defined in ProductOccurrences and ModelFile as discussed in previous chapter. The physical interpretation of those tolerances should then be done from both indications. For instance, if a tolerance in a topology section is stored as 0,001 and if the unit of the ProductOccurrence it belongs to is 1000 meters, then the physical tolerance on data is actually 1 meter.

5.8 Compressed file sections

Within a PRC File, all sections, except header sections, are individually compressed with a Flate method.

NOTE This form of compression is considered to be "lossless". It occurs systematically whatever the actual content of the PRC file, and even if it contains compressed geometry or tessellation.

5.9 Compressed geometry

Compression of geometry results in very small files. This compression is "lossy" in that the geometry is an approximation to geometry to within a specified tolerance (typically 0,001mm). See 8.9.21 and 8.9.20 for entities representing compressed geometry. Note that the methodology to determine an approximation of the original geometry (e.g. analytic recognition) is not part of PRC standard. Only the resulting entities and the method to store them are described in PRC.

5.10 Compressed tessellation

Compression of tessellation data results in very small files. This compression is "lossy" in that the tessellation data is an approximation to tessellation data to within a specified tolerance (typically 0,01mm). See 8.8.9 for an entity representing compressed tessellation. Tessellation data is not necessarily generated from or considered as an "approximation" of geometry; it is an alternative way to convey data. Note that the methodology to determine an approximation of the original tessellation (e.g. polygon decimation) is not part of PRC standard. Only the resulting entities and the method to store them are described in PRC.

6 PRC file contents

6.1 Fileheader

6.1.1 General

The Header contains the file version for the authoring version (PRC Format Specification) that the PRC File Writer is based on and the minimal version for read (PRC Format Specification) that a conforming PRC File Reader is based on that write/read the data outside of the individual FileStructures in the PRC File.

Each FileStructure has a identifier which is unique among all of the FileStructures within this PRC File.

Each application has a unique identifier which enables the interpretation of UserData. This identifier shall be set to 0000 or to a valid application identifier. 0000 indicates that the authoring application is not "registered" but (as discussed above) there can still be user data from another application in the file. Valid identifiers are distributed by Adobe Systems, Inc. upon request.

A valid PRC File shall contain at least one FileStructure.