PD ISO/PAS 8329:2024

BSI Standards Publication

Extended master connection file (χMCF) — Description of mechanical connections and joints in structural systems

Documents
Document Sharing Hub

Full document details: PD ISO/PAS 8329:2024

National foreword

This Published Document is the UK implementation of ISO/PAS 8329:2024.

The UK participation in its preparation was entrusted to Technical Committee AMT/4, Industrial data and manufacturing interfaces.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

This publication is not to be regarded as a British Standard.

© The British Standards Institution 2024
Published by BSI Standards Limited 2024

ISBN 978 0 539 25764 9

ICS 25.040.40; 25.040.40

Compliance with a Published Document cannot confer immunity from legal obligations.

This Published Document was published under the authority of the Standards Policy and Strategy Committee on 30 September 2024.

Amendments/corrigenda issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: PD ISO/PAS 8329:2024

Publicly Available Specification

ISO/PAS 8329

First edition
2024-08

Extended master connection file (χMCF) — Description of mechanical connections and joints in structural systems

Reference number
ISO/PAS 8329:2024(en)

© ISO 2024

Documents
Document Sharing Hub

Full document details: PD ISO/PAS 8329:2024

COPYRIGHT PROTECTED DOCUMENT

© ISO 2024

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Contents

Page

Foreword ..... **vi**
Introduction ..... **vii**
1 Scope **1**
2 Normative references **1**
3 Terms and definitions **1**
4 Design principles and basic features of χMCF **1**
4.1 General 1
4.2 Design principles 2
4.3 Idealization of joints 2
4.4 Reconstruction of joints from χMCF 3
4.5 Description of topology 3
4.6 χMCF in the development processes 4
5 Keywords of XML specification **5**
5.1 Keywords 5
6 Parts, properties and assemblies **7**
6.1 General 7
6.2 Parts 7
6.2.1 General 7
6.2.2 Part labels 7
6.2.3 Part instances 7
6.3 Properties 7
6.4 Assemblies 8
7 File structure of χMCF **8**
7.1 General 8
7.2 Elements containing general information 9
7.2.1 General 9
7.2.2 Date 9
7.2.3 Time 9
7.2.4 Version 10
7.2.5 Unit system 10
7.3 Application, user and process specific data 10
7.3.1 General 10
7.3.2 User specific data `<appdata/>` 11
7.3.3 Finite element specific data `<femdata/>` 12
7.4 Connection data `<connection_group/>` 13
7.4.1 General 13
7.4.2 Connected objects 14
7.4.3 Contacts and friction 19
7.4.4 Joints 21
7.5 Minimalistic example of a χMCF file 21
7.6 XML schema definition 22
8 Data common to any connection **22**
8.1 Indices and their properties 22
8.2 Connection referencing 23
8.2.1 Need for referencing 23
8.2.2 Attribute `label` 23
8.2.3 Attribute `ident` 23
8.3 Dimensions and coordinates 23
8.4 Attribute `quality_control` 23
8.5 Custom attributes list 23
8.6 Distinction between `<custom_attributes/>` and `<appdata/>` 27
8.6.1 General 27

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

8.6.2 Needs of different process roles, addressed by <custom_attributes/> and <appdata/> 28
8.6.3 Needs of different applications, addressed by <custom_attributes/> and <appdata/> 28
8.6.4 Different levels of <custom_attributes/> and <appdata/> within χMCF data model 28

9 0D connections 29
9.1 Generic definitions 29
9.1.1 Identification 29
9.1.2 Location 30
9.1.3 Direction 30
9.1.4 Type specification 31
9.2 Spot welds 31
9.2.1 General 31
9.2.2 Attribute diameter 32
9.2.3 Attribute technology 32
9.3 Robscans 33
9.4 Rivets 35
9.4.1 General 35
9.4.2 Blind rivets 37
9.4.3 Self-piercing rivets 39
9.4.4 Solid rivets 40
9.4.5 Swop rivets 43
9.4.6 Clinch rivet studs 44
9.5 Threaded connections — Bolts and screws 45
9.5.1 General 45
9.5.2 Contacts and friction 47
9.5.3 Definition of element <threaded_connection/> 50
9.5.4 Washer 52
9.5.5 Nut 53
9.5.6 Bolt 54
9.5.7 Screw 58
9.6 Gum drops 62
9.7 Clinches 62
9.8 Heat stakes / Thermal stakes 65
9.9 Clips / Snap joints 68
9.10 Nails 70
9.11 Rotation joints 73
9.11.1 General 73
9.11.2 ROTAV 74

10 1D connections 75
10.1 Generic definitions 75
10.1.1 Identification 75
10.1.2 Location 75
10.1.3 Intermittent connection lines 77
10.1.4 Type specification 84
10.2 Seam welds 84
10.2.1 Description and modelling parameters 84
10.2.2 Seam weld definition overview 84
10.2.3 Specific XML realization 86
10.2.4 Generic seam weld definition 86
10.2.5 Butt joint 95
10.2.6 Corner weld 98
10.2.7 Edge weld 104
10.2.8 I-weld 107
10.2.9 Overlap weld 110
10.2.10 Y-joint 116
10.2.11 K-joint 120

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

10.2.12 Cruciform joint ... 124
10.2.13 Flared joint ... 129
10.3 Adhesive lines ... 131
10.3.1 Element <adhesive_line/> ... 131
10.3.2 Element <loc_list/> ... 132
10.3.3 Element <appdata/> ... 132
10.3.4 Element <femdata/> ... 132
10.4 Hemming flanges ... 132
10.4.1 General ... 132
10.4.2 Element <hemming/> is placed within <connection_1d/> ... 134
10.4.3 Element <loc_list/> ... 135
10.4.4 Element <appdata/> ... 135
10.4.5 Element <femdata/> ... 135
10.4.6 Element <hemming/> ... 135
10.5 Sequence connections ... 137

11 2D Connections ... 139
11.1 Generic definitions ... 139
11.1.1 Identification ... 139
11.1.2 Connection face ... 140
11.1.3 Type specification ... 141
11.2 Adhesive faces ... 141

12 Future extensions ... 143
12.1 General ... 143
12.2 Additional parameters for spot and seam welds ... 143
12.3 Other relevant and new joint types ... 143

Annex A (informative) **Derivation of formulae used for regular intermittent welds ... 144**

Annex B (informative) **Federative use of χMCF with ISO 10303-242 ... 146**

Annex C (informative) **Background and context to this document ... 149**

Bibliography ... 150

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

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 document 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).

ISO draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO takes no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents. ISO shall not be held responsible for identifying any or all such patent rights.

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

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 4, *Industrial data*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Introduction

This document aims at describing mechanical connections or joints related to mechanical systems or structures. The demand for such a standard has grown from the observation that modern product lifestyle management (PLM) systems, while working well with part information (e.g. geometry, material, weight), are lacking a consistent handling of logical and process related connection information (e.g. parts being connected, orientation of point connections, assembly process parameter).

PLM workstreams need to include connection data to automate development processes and enable seamless data flows between engineering functions. χMCF is intended to be the "language" that is understood and used by the various tools to exchange connection data along the development chain.

The initial motivation to develop this document came from the automotive industry (see Annex C for background and context on this document). However, there is no element in this document that limits it to this industry. It is clearly targeted to support virtual development processes for mechanical systems or structures in any industrial area.

One design goal of χMCF is to support the widest possible range of development and manufacturing processes. This makes it very likely that xMCF and STEP (ISO 10303-242),[1] will be used together. Annex B investigates how this can be done in a way that benefits both standards.

Regardless of the respective industrial domain, complex technical systems (e.g. vehicles, planes, ships) typically consist of thousands of individual parts which are assembled by joints. Depending on the involved materials and the manufacturing processes, a wide range of joining types are used within an individual technical structure or system. Typical connection types are welds, bolt connections, adhesives, rivets, clips, etc. Efficient and reliable data management of such connection data is not only required for the actual design and verification process [computer-aided design (CAD) and computer-aided engineering (CAE)], but also for manufacturing planning and even cost estimation. Various design, material and manufacturing parameters are required to be managed for each connection.

Details for connections or joints grow and mature along the development process. At different development stages (e.g. concept phase, detailed design, verification, manufacturing planning) and engineering functions (e.g. CAD, CAE, manufacturing), data will be added and consumed. Therefore, a database for connection data is required. But also, the software tools adding or extracting data need to understand the data structure and use a common description language. χMCF, defined in this document, serves as this language.

The advantages are evident. Integrating dedicated connection data into the PLM structure and using a common language (χMCF) for data exchange avoids data conversions or re-generations and, therefore, decreases inconsistencies and flaws during system development.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Documents
Document Sharing Hub

Extended master connection file (χMCF) — Description of mechanical connections and joints in structural systems

1 Scope

This document specifies XML definitions that are used to describe data and information related to connections or joints in mechanical systems or structures.

The following is within the scope of this document:

— description and explanation of XML definitions for logical or process related data or other properties of a connection.

The following aspects are outside the scope of this document:

— geometry of fasteners or other parts,

— handling of χMCF data in

  — product data management (PDM) systems,

  — subscriber data management (SDM) systems, and

  — other data management systems.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

No terms and definitions are listed in this document.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp,

— IEC Electropedia: available at https://www.electropedia.org/.

4 Design principles and basic features of χMCF

4.1 General

The Extended Master Connection File (χMCF) is a container for connection information of complex structures. A complex structure consists of individual parts which are joined together. Connections establish a topology between the parts. Therefore, a database or container designed to gather connection information should be equipped with data structures which reflect this topology between the parts.

χMCF is intended to define an industry standard for the exchange of connection data between different CAx tools (e.g. CAD, CAE, CAM, CAT) along development process steps. Design principles for χMCF are required to keep the standard as lean as possible on one hand, but also enable use case dependent extensions.

Subclause 4.2 explains the design principles and basic features of χMCF.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

4.2 Design principles

The design of χMCF is guided by the following principles:

a) χMCF should be able to completely and unambiguously describe all relevant connections/joints that are in use in the automotive or other industries. Amongst others, this includes spot welds, seam welds, rivets, adhesives.

b) χMCF should be able to address all kinds of CAx processes.

c) χMCF contains only information relevant for connections. Hierarchical product structures, assembly sequences, part variants etc. are not the subject of χMCF. Such kind of information needs different methods for propagation. However, χMCF may refer to such "external" information, for example part codes. This principle provides the flexibility to use χMCF in any development process variant established at different companies.

d) χMCF has to be flexible and easy to extend to any future joint types and applications.

e) χMCF is based on the industry standard extensible markup language (XML).[2]

f) Connection data in χMCF must be unique.

g) The content of χMCF data may be incomplete to a certain extent. This addresses the fact that new data is created continuously and needs to be stored throughout the course of CAx processes, without changing its vessel.

h) χMCF follows the max-min principle. It contains information as much as necessary and, at the same time, as little as possible.

i) χMCF shall enable the reconstruction of connections at any certain stage of the involved processes without loss of data or risk of ambiguities.

j) Data in χMCF format shall be kept compact. Elements shall be reused, whenever possible.

k) χMCF offers containers which can be assigned to any certain connector, to a collection of connectors or even to the complete file. This allows incorporation of software or usage specific data before or without standardization.

l) χMCF forms a good candidate for long-term archival of connection data due to its simplicity and extendibility.

XML has been selected as a foundation since it is by itself an industry standard and human readable. XML facilitates efficient data structures which describe the connection topology of such complex structures like automobiles or planes.

4.3 Idealization of joints

Different types of joints have different characteristics. They can differ from each other by their geometrical shapes, mechanical properties like strengths for different loadings, manufacturing processes etc.

To allow for efficient description of joints, some simplifications and idealizations are necessary. The approach chosen by χMCF is to classify joints by their most basic and mandatory attribute, namely its geometrical dimensions. Thus, there are 0-, 1- and 2-dimensional joints in χMCF.

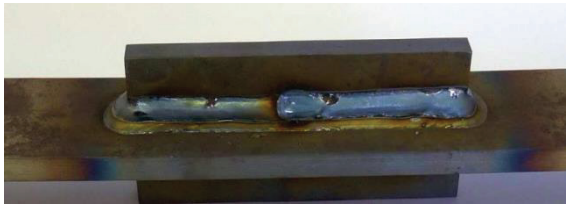


Figure 1 — Seam weld as 1-dimensional joint

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

A spot weld is treated as a 0-dimensional joint in χMCF. In this way, a (an idealized) spot weld is geometrically described by its coordinate vector *x* and its diameter *d* as an additional attribute. Besides spot welds, there are more joints which can be treated as 0-dimensional.

A seam weld is a typical representative of 1-dimensional joints, see Figure 1 above. It is characterized by a curve describing its spatial course and additional parameters (attributes) determining the sectional shape perpendicular to the curve.

Similarly, adhesive joints can be modelled as 2-dimensional surfaces.

4.4 Reconstruction of joints from χMCF

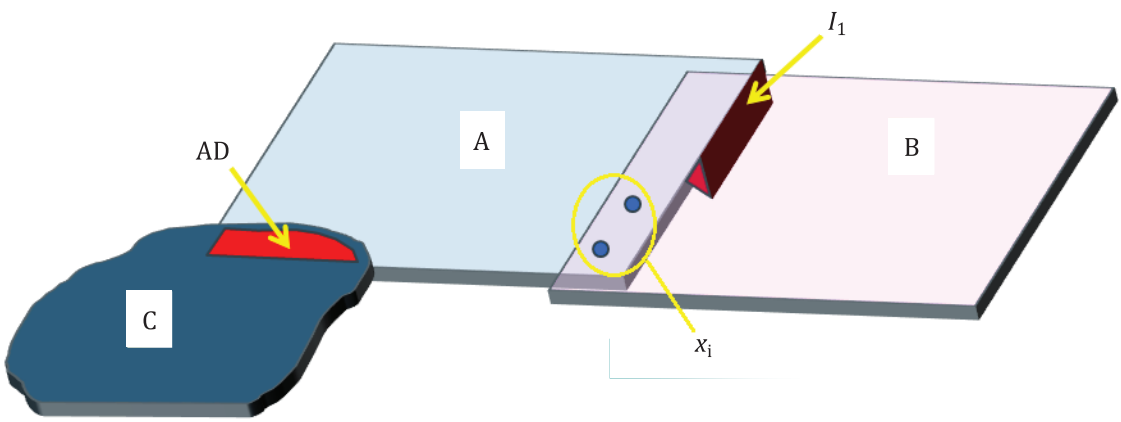
The reconstruction of joints from χMCF is an important use case. It is crucial that it is possible to reconstruct any joint in its idealized form uniquely by means of the introduced parameters and attributes. In case of a spot weld, a unique reconstruction is possible by the coordinate vector *x* and the diameter *d*, plus the sheet thicknesses which by themselves are not a constituent of χMCF (recall χMCF contains only information relevant to joints), but of the corresponding CAD or CAE model.

4.5 Description of topology

As mentioned before, a complex structure arises by connection of parts and sub-structures (assemblies). The connections introduce a topology between the individual components. The following example (see Figure 2) demonstrates the way how χMCF facilitates description of such topology:

— Part (or Assembly) A is joined to Part B by the seam weld 1 along the curve l_1 and the spot welds at positions x_i, and

— Part (or Assembly) A is connected to Part C by the adhesive AD in the area A, etc.



Key

A, B, C	parts
I_1	seam weld 1
x_i	spot welds
AD	adhesive

Figure 2 — Topological relations between parts and assemblies

This kind of topology is represented in χMCF by the element `<connection_group/>`. A `<connection_group/>` comprises all joints which connect the same parts (or assemblies).

Frequently, more than two parts are joined. A spot weld can, for instance, join three sheets, a screw even more. Such situations are covered, too.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

According to design principle c), overall product structure cannot be reproduced from χMCF. For example, any of the product structures shown in Figure 3 would equally fit to Figure 2:

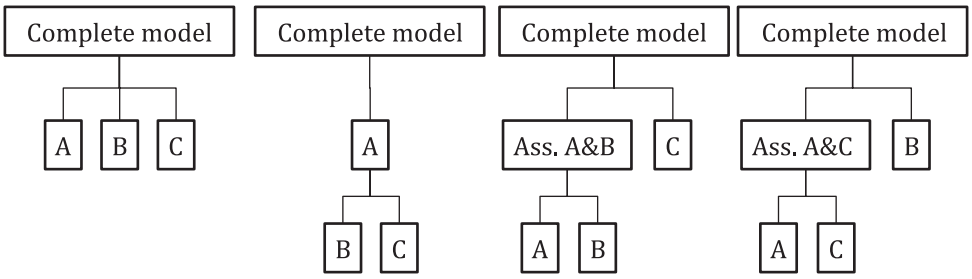


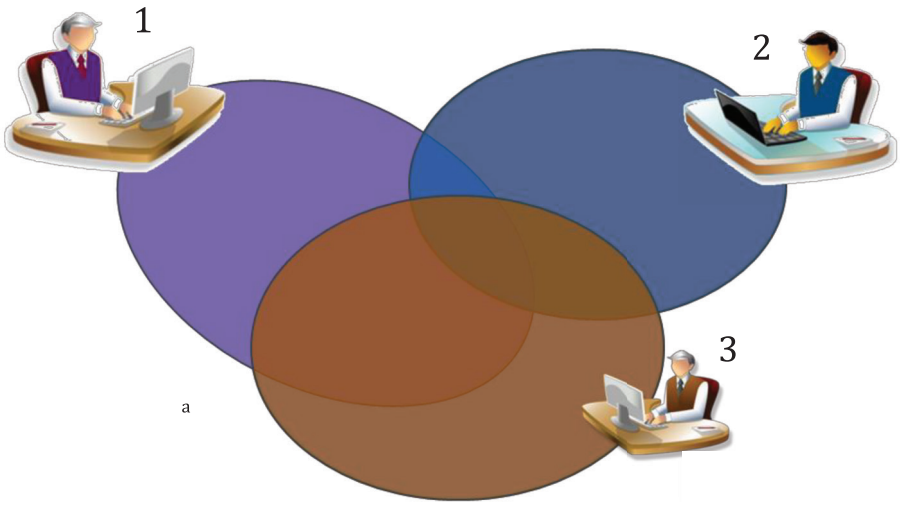
Figure 3 — Product structures fitting to previous figure

NOTE This list of four product structures shown in Figure 3 is not exhaustive.

4.6 χMCF in the development processes

A typical development process is a long chain involving many (maybe overlapping) single steps, e.g. design, construction, prototyping, simulation, testing, production planning (see Figure 4). Depending on the manufacturer considered, information of connections and joints arises at different stages of the process and comes from different parties (see Figure 5). An efficient handling and management of this information can only be guaranteed by a (common) database/container which contains the information uniquely. This shall be guaranteed by using χMCF.

Figure 4 — Development process



Key

1 design, construction

2 engineering

3 production planning

a χMCF.

Figure 5 — χMCF as a platform for connection data in the complete development process

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

A careful look at Figure 5 provides understanding on how the work with χMCF in a real process can be organized: χMCF is a structured set which can be divided into several overlapping subsets. Each subset contains a part of connection information which is of interest for a certain party, for instance simulation or planning. The intersection of all subsets contains information which is of interest for all the parties involved, e.g. coordinates and flange partners.

As mentioned before, the information contained in χMCF is not necessarily complete, at least not at an early stage of the development process. Rather its content grows while the process is advancing. Defining the individual joint and filling up the container thus builds up a continuous process. As shown in Figure 5, connection information can be created by any of the involved parties (e.g. design, construction, engineering, planning). The common situation is that each party contributes part of the information (e.g. geometrical, technological) defining a specific joint. Merging of the partial information leads to the complete characterization of the joint. Therefore, χMCF is an ideal tool to enable this dynamic process since filling up χMCF means merging information.

Figure 5 also illustrates that connection information (full or partial) is available to all involved parties once it is defined and stored in χMCF. Thus, unnecessary duplication of effort is avoided automatically. Typically, different teams work in different environments using different software tools. Provided all involved systems support χMCF, translation of data from one format to another will not be necessary anymore. This will save development cost and avoid loss of data caused by the translation.

Information contained in χMCF can be used to automate many tasks in a development process and will therefore facilitate efficiency gains:

— Automatic CAE model assembly
  Most FE preprocessors can mesh parts automatically in batch-meshing mode. An automated model assembly can be realized by the connection information contained in χMCF.

— Automatic Programming of Welding Robots
  Based on χMCF, welding robots can be programmed automatically.

An essential feature of χMCF is that it contains only information relevant to the joints. No data are included which are dependent on a specific development process. Therefore, χMCF can be implemented into any development process. Depending on the application, it is possible to use χMCF as a stand-alone database or integrate χMCF into an even more comprehensive database.

5 Keywords of XML specification

5.1 Keywords

As in any XML file, the carrier of information in an χMCF file is an element which can be equipped with some attributes and child elements. Elements and attributes are defined by their names (identifiers) and values (information).

By the XML standard, values assumed by elements can be distinguished by their types, e.g. boolean, float, double, string, date. The same applies to attributes. The user can determine how elements and attributes are used (optional, required or prohibited). If declared necessary, the frequency of occurrence of elements with a given name (number of siblings of identical names) can be restricted. In the XML schema, this is specified by the attributes `minOccurs` and `maxOccurs`.

In accordance with the XML Schema Definition Language version 1.1 Part 2 the following keywords are used in the current document to characterize the elements and attributes:

— Type,

— Value space,

— Default,

— Use,

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

— Multiplicity (corresponds to the attributes `minOccurs` and `maxOccurs` of the element `<xs:element/>` of the XML schema),

— Restrictions (corresponds to the element `restriction` of XML schema).

NOTE 1 Up to now, only versions 1.0[2] and 1.1[3] of XML exist, where 1.1 is not widely used. Therefore, most systems still create XML 1.0 files (for differences between both versions see http://www.w3.org/TR/xml11/#sec-xml11).

The type of the value of an element or attribute is specified by the key-word `Type`. The numerical ID of a property (attribute `pid`) of a `<part/>` element for instance is an integer, which is a built-in type of XML standard.

Examples for the most common types in XML are:

— `xs:string`,

— `xs:decimal`,

— `xs:integer`,

— `xs:float`,

— `xs:boolean`,

— `xs:date`,

— `xs:time`.

NOTE 2 The maximum number of decimal digits you can specify is 18.

However, only positive integers are usually used in this context. This means that the possible values of the ID (type integer) have to be restricted. To specify the values which are allowed for an element or an attribute, the key-word Value space is used. The Value space can be given as an enumeration (a finite set), or an explicitly defined set. For example, a positive integer is symbolized by > 0 whereas a float between 0,0 and 1,0 is given by [0,0, 1,0], according to mathematical notation.

Some elements and attributes obtain default values if they are not explicitly specified in the χMCF file. The default values to be adopted are defined by the keyword `Default`.

In this document, the special type “alphanumeric” is frequently used for labels of parts and assemblies, which deserves a careful discussion. In the CAD world, a label is synonymous with the name of a part, a geometric object etc. Not only letters “[A-Za-z]”, but also numbers “[0-9]” and other special characters such as “[-.$#±]” and more are used for labels. Sometimes, the first character is restricted to “[A-Za-z]”. Thus, it is difficult to give an exact definition for the type “alphanumeric” which would fit to the individual need. Fortunately, when using XML’s “encoding” attribute, even non-ASCII characters can be handled easily, e.g. Arabic, Chinese, Cyrillic, Greek, Hebrew. Nevertheless, labels should not start or end with white space.

The key-word `Use` specifies, whether an element or an attribute is optional, required or prohibited. The frequency of the occurrence of an element or attribute is defined by `Multiplicity`, that is in the form: `minOccurs` ≤ `Multiplicity` ≤ `maxOccurs`. By convention, when `Use` is optional, `minOccurs` is 0. Any additional restrictions imposed on an element, or an attribute are specified by the key-word `Restrictions`.

As explained above, the individual use of some elements or attributes may be optional. But some of them must be coherent (thus redundant in certain sense). For instance, the `label`, numerical ID of a property (`PID`), and alphanumerical name of a property (`pname`) of a part or an assembly represent the same part (except for e.g. tailored blanks) and one can use one or the other or both to identify a part.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

6 Parts, properties and assemblies

6.1 General

χMCF describes how parts, properties and assemblies are connected by joints in a pre-defined way. Therefore, a clear understanding is needed about what a part, property or assembly is, in our context.

6.2 Parts

6.2.1 General

Parts are logical groupings of 3D objects. Their objective is to provide a general nomenclature of the pieces which form a certain product. This nomenclature allows communications between all stakeholders of all involved processes.

Typically, it is assumed that parts do not disintegrate into several physical compounds.

Parts can be instantiated at different locations of a product, e.g. wheels in a car.

Parts can be mirrored on a symmetry plane of the model, e.g. front doors of a car.

Parts can contain other parts (sub-parts): A car, for example, is made of body in white, power train, doors etc. A door is made of an outer sheet, an inner sheet, a window with its mechanics, some crash reinforcements etc. The mechanics of a window are made of some guiding rails, an electric motor and so on.

Therefore, in the sense of graph theory, parts form a tree (if their instances are considered) or a directed, cycle free graph. Parts without sub-parts are called the “leaves” of this tree or graph.

If a part is mentioned in a list, not only its own content (e.g. finite elements) is addressed, but also all contents of its sub-parts and their children, down to the lowest level (leaves) of the part graph.

6.2.2 Part labels

A part is uniquely identified by its label, up to ditto-parts. Connectors within a connection group that refer to ditto parts shall be able to “detect” the “correct” part instance according to their respective geometrical location.

We assume that mirror parts have other part labels than their “base” parts.

NOTE In most CAx processes, parts have two string attributes: One label describing the name and usage of a part in a human readable form, and another one used for indexing this item in the OEM’s “part store”. The latter one typically consists of only a few characters (e.g. some 8 to 12), resembles more to a number than to a name, and therefore, is not human readable. In our context, the term “part label” refers to the latter one.

6.2.3 Part instances

Instances of parts, also known as ditto-parts, typically have the same label as their “base” parts. Stating their instance makes such parts uniquely distinguishable, without resort to their geometrical location. Stating an instance without a part is meaningless, however.

6.3 Properties

In CAE, properties are a concept for assigning physical behaviour to several finite elements. Therefore, any finite element can have at most one property. However, frequently there are elements without such properties (e.g. RBEs, masses). In most solvers, properties are uniquely identified by positive integers, so-called property IDs or short: `PID`s. Some other solvers identify properties by an alphanumerical name, short `pname`.

Even if finite elements of different parts have the same physical behaviour (e.g. left and right wing of a car), they usually have assigned different properties. This can be seen as a reminiscence to ancient times when parts had not been invented and properties were also used for administrative purposes.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

However, for χMCF, properties are just alternative, non-recursive means for addressing collections of elements.

One specific part often consists of one specific property, only. However, there are important exceptions:

— A tailored blank is a metal sheet which consists of several pieces of simple sheets joined together. The thicknesses and the materials of the individual sheets may both differ. Nevertheless, a tailored blank is one single part from the χMCF point of view. Since one property would not provide an identifier for the complete part, the part label must be used, or else an assembly of several properties.

— Sometimes, a cast part can be represented with finite element (FE) shell element formulation in its thin areas, whereas solid elements (with different properties) are used in other areas.

— Due to stamping processes for example, physical behaviour and thickness can vary even within one originally homogeneous sheet metal, requiring several properties for a correct simulation.

— Occasionally, CAD parts containing several subparts with their properties are aggregated to one single CAE part, consequently still containing several properties.

6.4 Assemblies

In many CAx systems, parts containing sub-parts are called assemblies. The notion distinguishes them from leaves of the part tree or graph.

However, in χMCF, an assembly is considered a set of parts, denoted by their part labels or properties. They do not need to have any special relation respective to the part graph. The opposite is true: χMCF-assemblies address situations where specifying a single property would not address enough, a high-level part would address way too many elements and medium-size parts would not make the job.

On the other hand, this does not happen too often. For example, if a seam weld crosses property boundaries, these properties usually belong to the same tailored blank, therefore, the same part. If there would be a physical gap between the properties, welding would be applied to a single sheet across this gap, which causes new questions to the welding process, as depicted in Figure 6:

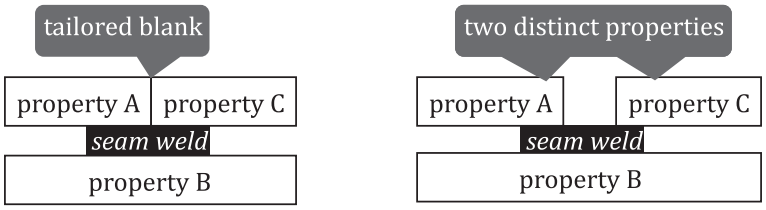


Figure 6 — Seam weld crossing tailored blank versus seam weld crossing physical gap

Even if there is a gap, due to geometrical proximity and usual assembly processes, it is very likely that properties A and C belong to the same part, just one level above, even if there is a gap in part of the graph.

7 File structure of χMCF

7.1 General

As mentioned, χMCF is built upon XML. This eases χMCF to offer a clear logical structure.

The root/document element of χMCF must be named `<xmcf/>`. The root element may contain the following types of child elements:

a) Comments following the usual XML standard; therefore, not further discussed here,

b) Elements containing general information,

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

c) Groups of connection specific elements `<connection_group/>` of arbitrary number,

d) Element `<appdata/>` containing specific data for individual applications,

e) Element `<femdata/>` containing finite element specific data.

7.2 Elements containing general information

7.2.1 General

χMCF has the following elements for general information:

— `<date/>` optional,

— `<version/>` mandatory,

— `<units/>` optional.

The root element `<xmcf/>` contains the following nested elemenths shown in Table 1:

Table 1 — Nested elements of element `<xmcf/>`

Nested elements	Multiplicity	Use	Constraint / Remarks
date	1	Optional	-
version	1	Required	-
units	1	Optional	-
appdata	1-*	Optional	See 7.3.2
femdata	1-*	Optional	See 7.3.3
connection_group	1-*	Optional	See 7.4

7.2.2 Date

The element `<date/>` of the format “yyyy-mm-dd” specifies the date on which the file was created. It follows the ISO 8601 series.[4]

EXAMPLE

```
<?xml version="1.0" encoding="UTF-8" ?>
<xmcf xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <date> 2023-04-13 </date>
    <version> 3.1.1 </version>
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    ...
</xmcf>
```

7.2.3 Time

The element `<time/>` of the format “hh:mm:ss±hh:mm” specifies the time on which the file was created. It follows the ISO 8601 series [4].

Time element may exist only if date element exists.

EXAMPLE

```
<?xml version="1.0" encoding="UTF-8" ?>
<xmcf xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <date> 2023-04-13 </date>
    <time> 15:34:05-01:00 </time>
    <version> 3.1.1 </version>
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

```
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    ...
</xmcf>
```

7.2.4 Version

The code of the χMCF standard version on which the current file is based is specified by the element `<version/>`.

The version code of χMCF files following this document is version 3.1.1.

EXAMPLE

```
<?xml version="1.0" encoding="UTF-8" ?>
<xmcf xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <date> 2023-04-13 </date>
    <time> 15:34:05-01:00 </time>
    <version> 3.1.1 </version>
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    ...
</xmcf>
```

7.2.5 Unit system

The unit system used by χMCF is based on the International System of Units (SI)[5] and is represented by the element `<units/>`. Both the base and the derived units are supported, including decimal prefixes.

Following non-SI units are allowed (in addition): Length [in] and [ft]; Mass [lb], see Reference [6].

No units need to be specified for dimensionless physical quantities, e.g. friction coefficients.

The XML-specification of `<units/>` is shown in Table 2:

Table 2 — XML-specification of `<units/>`

Attribute	Use	Value space	Default
length	Optional	“mm”, “m”, “in”, “ft"	“mm”
angle	Optional	“deg”, “rad”	"deg”
mass	Optional	“g”, “kg”, “t”, “lb”	“kg”
force	Optional	“kN”, “N”	“N”
time	Optional	“s”, “min”, “h”	“s”
torque	Optional	“Nm”	“Nm”
angular_speed	Optional	“rad/s”, “Hz”, “kHz”, “rpm”	“Hz”

EXAMPLE

```
<?xml version="1.0" encoding="UTF-8" ?>
<xmcf xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <date> 2023-04-13 </date>
    <version> 3.1.1 </version>
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    ...
</xmcf>
```

7.3 Application, user and process specific data

7.3.1 General

The user/application software can store additional information into a χMCF file. In this way, flexibility is created that allows easy integration of χMCF into an existing development process.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

The current χMCF definition allows two such data elements:

— `<appdata/>`
Contents must be documented by the corresponding application or user. It is not an official part of the χMCF standard.

— `<femdata/>`
Contents must be documented in FATXML[7] and therefore does not need to be described here.

7.3.2 User specific data `<appdata/>`

`<appdata/>` is suitable for any user/application specific information and can be placed on root level (directly within `<xmcf/>` element) and/or within any single connector (elements `<connection_0d/>`, `<connection_1d/>`, and `<connection_2d/>`). Additionally, it can be placed directly under the `<connection_group/>` element.

`<appdata/>` shall contain at least one nested element named after the application or user that is intended to interpret the data. In examples A and B, the associated application is MEDINA, so the nested element is `<MEDINA/>`.

Content of `<appdata/>` is regarded to be "private property" of the corresponding application. However, in terms of best practices, the following is recommended, but not required:

— to place application specific elements into a separate namespace;

— to provide an XML schema for its content.

The user shall be aware that different systems are likely to introduce the same physical parameter at the same time (inducted e.g. by a certain new emerging connecting method) but describe it in their own XML schemas with different element/attribute names.

In this situation, a preprocessor does not have any chance to detect these equivalent parameters. Therefore, it cannot prevent contradictions between different `<appdata/>` blocks of the same χMCF file.

EXAMPLE 1 `<appdata/>` for MEDINA at root level

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<xmcf xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xmlns:MEDINA="http://servicenet.t-systems.com/medina/xMCF"
xsi:schemaLocation="http://servicenet.t-systems.com/medina/xMCF   mcf_MEDINA.xsd"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <date> 2014-08-07 </date>
    <version> 3.1.1 </version>
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    <appdata>
        <MEDINA xmlns="http://servicenet.t-systems.com/medina/xMCF">
            <data_at_root>
                <version MEDINA="MEDINA 8.4.2 Maintenance Release (64 Bit)"/>
                ...
            </data_at_root>
        </MEDINA>
    </appdata>
    ...
</xmcf>
```

EXAMPLE 2 `<appdata/>` for MEDINA at connection level

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<xmcf xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xmlns:MEDINA="http://servicenet.t-systems.com/medina/xMCF"
xsi:schemaLocation="http://servicenet.t-systems.com/medina/xMCF mcf_MEDINA.xsd"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <date> 2014-08-07 </date>
    <version> 3.1.1 </version>
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    ...
    <connection_group id="1">
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

```
        <connected_to>
            ...
        </connected_to>
        <connection_list>
            <connection_1d>
                <loc_list>
                    ...
                </loc_list>
                <seamweld>
                    ...
                </seamweld>
                <appdata>
                    <MEDINA xmlns="http://servicenet.t-systems.com/medina/xMCF">
                        <data_at_connector>
                            ....
                        </data_at_connector>
                    </MEDINA>
                </appdata>
            </connection_1d>
        </connection_list>
    </connection_group>
</xmcf>
```

7.3.3 Finite element specific data `<femdata/>`

For numerical simulation by the finite element method, a joint can be discretized (realized) in different ways depending on the focus of the simulation (e.g. crash, fatigue). It is therefore often necessary to switch from one realization to another one. For this purpose, details of a specific realization can be of interest.

The optional `<femdata/>` information can be placed within any single connector (relevant elements are `<connection_0d/>`, `<connection_1d/>` and `<connection_2d/>`).

χMCF versions 3.1 or later allow to contain `<femdata/>` at root level, but this is not allowed in V 3.0 and below. `<femdata/>` is not allowed on `<connection_group/>` level in any case.

`<femdata/>` refers to FE entities that are related to the connector in which `<femdata/>` is placed. Its content and the referenced elements are specific to a particular solver.

Usually, this kind of referencing is done by solver specific entity IDs, which have no meaning outside the context of a specific finite element model. If, for example, element IDs in this model get renumbered, a χMCF file referencing such element IDs becomes detached and needs to be re-created.

In conclusion, a χMCF file containing `<femdata/>` always refers to one specific solver deck.

Solver names should be taken from the current FATXML version. Examples are the following:

— PAM-CRASH,

— LS-DYNA,

— RADIOSS,

— OPTISTRUCT,

— NASTRAN,

— PERMAS,

— ABAQUS.

Only `<entity/>` (see Table 3) is allowed as a nested element of the child element of `<femdata/>`. Its definition and documentation follow `<ENTITY/>`, the corresponding element in FATXML.[7]

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 3 — Nested elements of the child element of `<femdata/>`

Nested elements	Multiplicity	Use	Constraint / Remarks
entity	1-*	Required	Corresponds to element **<ENTITY/>**, defined in Reference [7].

For further definition of `ENTITY` see the document source website for FATXML.[7]

EXAMPLE <femdata/> within a <connection_0d/> element

```
<connection_0d>
    ...
    <femdata>
       <NASTRAN>
            <entity>
                <TYPE>
                    CQUAD
                </TYPE>
                <ID>
                    12345-12356
                </ID>
            </entity>
       </NASTRAN>
   </femdata>
   ...
</connection_0d>
```

Like FATXML, χMCF data can be embedded into solver decks by this means: Any receiving system can easily detect and remove discretization objects, created by a sending system, in order to substitute them by its own new discretization objects.

The `<femdata/>` element can be used versatile for different use cases – even for yet unknown ones. This makes it difficult to define exact semantics.

Specific agreements, for example between preprocessor and solver/postprocessor, may be made to support specific use cases.

7.4 Connection data `<connection_group/>`

7.4.1 General

`<connection_group/>` contains the topological information about the parts or assemblies involved (Clause 6), respectively. As explained in 4.5, joints are grouped together by the parts or assemblies which they commonly connect.

The topological relation (relation of neighbours) is defined by the child element `<connected_to/>` whereas all involved joints are listed in the child element `<connection_list/>` according to their types (see 4.3).

Each `<connection_group/>` is uniquely identified by a numeric identifier (`id`).

NOTE Therefore, χMCF files cannot be simply “pasted together” by use of a standard text editor.

XML-specification of `<connection_group/>` is shown in Table 4 and Table 5.

Table 4 — Attributes of element `<connection_group/>`

Attributes	Type	Use	Constraint / Remark
id	Integer	Required	unique within a χMCF file

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 5 — Nested elements of element `<connection_group/>`

Nested elements	Multiplicity	Use	Constraint
connected_to	1	Required	-
connection_list	1	Required	-
contact_list	1	Optional	-

An empty or missing `<connected_to/>` element means a connection according to geometric neighbourhood, alone. However, if `<connected_to/>` is present, it must be complete, i.e. no additional connection partners shall be searched. Searching for a geometric neighbourhood can yield different results, depending on the algorithm employed. To avoid ambiguities, no connections with missing `<connected_to/>` should reach the solver. Therefore, `<connected_to/>` should be filled by the pre-processor.

In addition to parts and properties, no other means (such as sets) for grouping objects are allowed.

7.4.2 Connected objects

7.4.2.1 General

The basic objects which can be jointed are parts and assemblies (see Clause 6) which appear as nested elements `<part/>` and `<assy/>` of `<connected_to/>`. The XML-specification of `<connected_to/>` is shown in Table 6.

Table 6 — Nested elements of `<connected_to/>`

Nested elements	Multiplicity	Use	Constraint
part	1 - *	Optional	-
assy	1 - *	Optional	-

7.4.2.2 Element `<part/>`

In χMCF, a part may refer to one CAx part or one CAE property.

A part is described by the element `<part/>` and a numeric `index`, a `label` (part code), a `pid` (property id) or `pname` (property name), all provided as attributes. However, if both attributes “label” and “pid” or “label” and “pname” are present, the label governs.

Although most solvers use numbers as identifiers, ABAQUS uses names as identifiers. To identify a property, only one of `pid` or `pname` is sufficient. If both identifiers are present, they must be equivalent in the sense that they both address the same collection of elements. Rationale for allowing presence of both identifiers is the case that the same mesh, and therefore, same properties are used in both, NASTRAN and ABAQUS. In this situation, it is recommended to have a χMCF file which contains both, PIDs and property names. On the solver side, this would cause no confusion since NASTRAN would ignore the property name and ABAQUS the PID. The responsibility to keep both primary keys unique and equivalent resides on the pre-processor side. Upon import of χMCF to a pre-processor, inconsistent property keys shall cause an error.

The `index` needs to be unique only within the parent element `<connected_to/>`. For specific connections, it is used as the matching index for the base sheet.

The attribute `index` of `<part/>` element is required only if the `part` element is used as a nested element under the `<connected_to/>` element. If the `<part/>` element is used within the element `<assy/>`, then `index` is not allowed as an attribute of the `<part/>` element. The XML-specification of `<part/>` is shown in Table 7.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 7 — Attributes of element `<part/>`

Attributes	Type	Value space	Use	Constraint
index	Integer	> 0	Required	Unique and required only within the parent element `<connected_to/>`
label	Alphanumeric	Alphanumeric	Optional	At least `label`, `pid`, or `pname` must exist.
pid	Integer	> 0	Optional	
pname	Alphanumeric	Alphanumeric	Optional	
instance	Alphanumeric	non-empty	Optional	`label` must exist if `instance` is used.

EXAMPLE 1 <part/> with required attributes only (pid or pname may be used alternatively to label)

```
<connected_to>
    <part index="1" label="PART_7000400"/>
</connected_to>
```

EXAMPLE 2 <part/> with optional use of label and pid

```
<connected_to>
    <part index="1" label="PART_7000400" pid="3202132"/>
</connected_to>
```

EXAMPLE 3 <part/> with pname to identify a part or property

```
<connected_to>
    <part index="1" pname="P3202132 Thin Shell Property"/>
</connected_to>
```

EXAMPLE 4 <part/> using a label and an instance to identify a part

```
<connected_to>
    <part index="1" label="PART_WHEEL_900" instance="4"/>
</connected_to>
```

7.4.2.3 Element `<assy/>`

An assembly represents a sub-structure consisting of at least two `<part/>` elements. It is described by the element `<assy/>` with only the mandatory attribute `index`. The XML specification of element `<assy/>` is shown in Table 8.

Table 8 — Attributes of element `<assy/>`

Attributes	Type	Use	Constraint
index	Integer	Required	Unique within the parent element

In the following examples, any `pid` attribute can be supplemented or replaced by a `pname` attribute.

EXAMPLE 1 Full definition of `<assy/>` element within `<connected_to/>`

```
<connected_to>
    <assy index="42">
        <part label="PART_7000400" pid="110013"/>
        <part label="PART_7000800" pid="110099"/>
    </assy>
</connected_to>
```

EXAMPLE 2 Full definition of the combined use of `<part/>` and `<assy/>` elements within `<connected_to/>`

```
<connected_to>
    <part index="1" label="PART_9004400" pid="3202132"/>
    <assy index="42">
        <part label="PART_7000400" pid="110013"/>
        <part label="PART_7000800" pid="110099"/>
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

```
    </assy>
</connected_to>
```

EXAMPLE 3 Usage of the attribute instance for a `<part/>` within an `<assy/>` element

```
<connected_to>
    <part index="1" label="PART_9004400" pid="3202132"/>
    <assy index="42">
        <part label="PART_7000400" instance="2" pid="110013"/>
        <part label="PART_7000800" pid="110099"/>
    </assy>
</connected_to>
```

EXAMPLE 4 Minimum definition for the combined use of `<part/>` and `<assy/>` elements within `<connected_to/>`

```
<connected_to>
    <part index="1" label="PART_9004400"/>
    <assy index="42">
        <part label="PART_7000400"/>
        <part label="PART_7000800"/>
    </assy>
</connected_to>
```

or

```
<connected_to>
    <part index="1" pid="3202132"/>
    <assy index="42">
        <part pid="110013"/>
        <part pid="110099"/>
    </assy>
</connected_to>
```

The body of an `<assy/>` element equals that of a `<connected_to/>` element. But the meaning is different: All parts within one `<assy/>` element are meant to constitute the same side/layer/partner of a flange, whereas all members of a `<connected_to/>` element are different sides/layers/partners of a flange.

Recursion, such as an `<assy/>` element nested within another `<assy/>` element, is not allowed.

7.4.2.4 Special topological situations

7.4.2.4.1 Stacking

The aim of the `<connection_group/>` element is to group up all the joints that connect the same parts.

Therefore, `<connected_to/>` contains each part connected in the joint only once. However, in some situations it can be important to explicitly define in which order some parts of the group are connected.

This includes the following scenarios:

— the stacking order of the connected parts can be important,

— some parts may be involved more than once in the same joint (self-connected joint), where either:

  — each part involved in a self-connected joint more than once is known individually, or

  — just the number of parts involved in a self-connected joint is known, or

  — a combination of the two sub-scenarios above.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

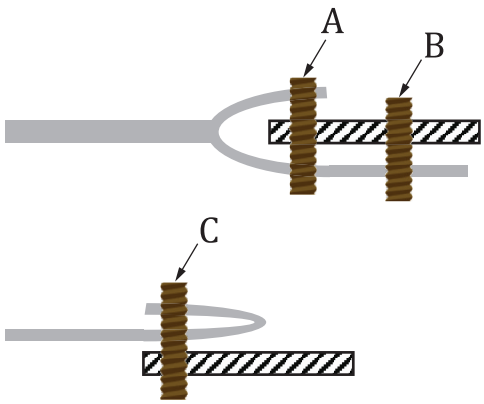


Figure 7 — Special stacking topologies

In Figure 7, all joints, A, B, C, exist within the same `<connection_group/>`, but each joint is connected in a different way.

```
<connection_group>
    <connected_to>
        <part index="1" label="PART_7000800"/>   <!-- grey part in figure -->
        <part index="2" label="PART_7000400"/>   <!-- hatched part in figure -->
    </connected_to>
</connection_group>
```

For joints A and C, the number of flanges connected is more than the number of parts in `<connected_to/>`. Between joints A and C, the flanges feature the same parts, but in a different order.

To store this information for each case, the `<stacking/>` element is used.

7.4.2.4.2 Element <stacking/>

`<stacking/>` may dictate the list of flanges/sheets involved in a joint, as well as their order. Alternatively, `<stacking/>` may indicate the number of flanges/sheets of a joint, without defining which parts are connected more than once. Table 9 shows the nesting element of `<stacking/>` and Table 10 its attribute.

Table 9 — Nested elements of `<stacking/>`

Nested Elements	Multiplicity	Use	Constraint
level	1 - *	Optional	-

Table 10 — Attributes of `<stacking/>`

Attributes	Type	Value space	Use	Constraints / Remarks
nr_levels	Integer	> 0	Optional	if `nr_levels` exists, no `<level/>` elements are allowed in `<stacking/>`. `nr_levels` has to be greater than the number of nested elements of `<connected_to/>`

The attribute `nr_levels` dictates the number of flanges/sheets connected by the joint.

The element `<level/>` within `<stacking/>` is specified as shown in Table 11.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 11 — Attributes of `<level/>`

Attributes	Type	Value space	Use	Constraint
order	Integer	> 0	Required	Unique only within the parent element `<stacking/>`
part_index	Integer		Required	

The attribute meanings are:

— `part_index`: The flange partner with this index (see 7.4.2.2). The part of the flange is referenced by the attribute `index` inside the element `<part/>` or `<assy/>` of the `<connected_to/>` element,

— `order`: indicates the position of a flange relative to other flanges.

The order of the levels in the stacking list is identified by the numerical value of their attribute `order`, in ascending order. Therefore, indices must be unique within one stacking list. The restriction that `nr_levels` must be greater than the number of nested elements of `<connected_to/>` implies that `nr_levels` attribute can only be used for self-connected joints.

EXAMPLE 1 The situations in Figure 7 can be described by using `<level/>` elements in order to explicitly define the stacking of the part flanges involved.

```
<connection_group>

    <connected_to>
         <part index="1" label="PART_7000800"/>          <!-- grey part in figure -->
         <part index="2" label="PART_7000400"/>          <!-- hatched part in figure  -->
    </connected_to>

   <connection_list>
        <connection_0d label="A">
              <stacking>
                    <level order="1" part_index="1"/>   <!-- grey part in figure -->
                    <level order="2" part_index="2"/>   <!-- hatched part in figure  -->
                    <level order="3" part_index="1"/>   <!-- grey part in figure -->
              </stacking>
              ...
        </connection_0d>

        <connection_0d label="B">
              <stacking>
                    <level order="1" part_index="2"/>   <!-- hatched part in figure  -->
                    <level order="2" part_index="1"/>   <!-- grey part in figure -->
              </stacking>
              ...
        </connection_0d>

        <connection_0d label="C">
              <stacking>
                    <level order="1" part_index="1"/>   <!-- grey part in figure -->
                    <level order="2" part_index="1"/>   <!-- grey part in figure -->
                    <level order="3" part_index="2"/>   <!-- hatched part in figure  -->
              </stacking>
              ...
        </connection_0d>
   </connection_list >

</connection_group>
```

EXAMPLE 2 The same situations can also be expressed using the nr_`levels` attribute, which simply states how many flanges of the <connected_to> parts are involved in each joint.

```
<connection_group>

    <connected_to>
         <part index="1" label="PART_7000800"/>   <!-- grey part in figure -->
         <part index="2" label="PART_7000400"/>   <!-- hatched part in figure  -->
    </connected_to>
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

```
    <connection_list>
        <connection_0d label="A">
            <stacking nr_levels="3"/>   <!-- "hatched", "grey" and one of "hatched"/"grey"
-->
            ...
        </connection_0d>

        <connection_0d label="B">
            ...                          <!-- "hatched", "grey" in any order -->
        </connection_0d>

        <connection_0d label="C">
            <stacking nr_levels="3"/>   <!-- "hatched", "grey" and one of "hatched"/"grey"
-->
            ...
        </connection_0d>
    </connection_list >

  </connection_group>
```

7.4.3 Contacts and friction

7.4.3.1 General

For many joint types, e.g. bolts and screws, friction between the jointed partners plays an important role for the manufacturing and the mechanical behaviour of the joints in service.

In general, friction is a property of pairs of materials in contact. Normally it can be assumed that the friction property, here simply characterized by the static and kinetic friction coefficients, is homogenous. Nevertheless, friction properties shall allow for local modification of an individual connection to enhance the service behaviour.

In χMCF, friction coefficients for any combination of joint partners defined in `<connected_to/>` can be specified by the element `<contact/>` which is nested in the element `<contact_list/>`. Each part in contact is given by the element `<partner/>`. The static and kinetic friction coefficients are defined by the element `<coefficients/>`.

The friction property between the head of a bolt to jointed parts is specified, where the joint is defined.

7.4.3.2 Element `<contact_list/>`

Relevant contacts, which are possible between the flange partners of a `<connection_group/>`, are collected in a `<contact_list/>`. The XML specification of `<contact_list/>` element is shown in Table 12.

Table 12 — Nested elements of element `<contact_list/>`

Nested elements	Multiplicity	Use	Constraint
contact	1 - *	Required	Any set (= non-ordered pair) of physical contact partners is not allowed to appear more than once within a `<contact_list/>`.

The element `<contact_list/>` does not allow for any attributes.

7.4.3.3 Element `<contact/>`

The features or coefficients of a physical contact between flange partners are described by an element `<contact/>`. The XML specification of the element `<contact/>` is shown in Table 13.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 13 — Nested elements of element `<contact/>`

Nested elements	Multiplicity	Use	Constraint
partner	2	Required	-
coefficients	1	Required	-

Ordering of `<contact/>` elements within a `<contact_list/>` is irrelevant, since it is assumed that features of a physical contact are invariant under permutation of the two involved materials.

The element `<contact/>` does not allow for any attributes.

7.4.3.4 Element `<partner/>`

Each joint partner involved in a contact is specified by the element `<partner/>`. Only the first level of parts/assemblies which are listed in `<connected_to/>`, is allowed. The XML specification of the `<partner/>` element is shown in Table 14.

Table 14 — Attributes of element `<partner/>`

Attributes	Type	Value space	Use	Constraints / Remarks
part_index	Integer		Required	

The attribute has following meaning:

— `part_index`: The flange partner with this index (see 7.4.2.2). The part of the flange is referenced by the attribute `index` inside the element `<part/>` or `<assy/>` of the `<connected_to/>` element.

The element `<partner/>` does not allow for any nested elements.

7.4.3.5 Element `<coefficients/>`

Static and kinetic friction coefficients are defined by the attributes `static_friction` and `kinetic_friction` of an element `<coefficients/>`, respectively.

Example

```
<connected_to>
    <part index="1" label="PART_9004400" pid="3202132"/>
    <assy index="42">
        <part label="PART_7000400" pid="110013"/>
        <part label="PART_7000800" pid="110099"/>
    </assy>
</connected_to>
<contact_list>
    <contact>
       <partner part_index="1"/>
       <partner part_index="42"/>
       <coefficients static_friction="0.3" kinetic_friction=".25"/>
    </contact>
</contact_list>
```

The element `<coefficients/>` does not allow for any nested elements.

7.4.3.6 Local contact properties

If necessary, local contact properties can be given within any element `<connection_0d/>` or `<connection_1d/>`, respectively (see 7.4.4). In case of conflict, a local `<contact_list/>` overrules the global one.

NOTE `<connection_2d/>` is not relevant for the currently known use cases and was therefore intentionally not included in the list.

XML-specification of `<coefficients/>` are shown in Table 15.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 15 — Attributes of element `<coefficients/>`

Attributes	Type	Value space	Use	Constraints / Remarks
static_friction	Floating point	[0, ∞[	Optional	-
kinetic_friction	Floating point	[0, ∞[	Optional	-

A `<connection_list/>` is not allowed to be empty, i.e. at least one connection must be defined.

7.4.4 Joints

All the joints which connect the same set of objects (order does not matter) described in the element `<connected_to/>` are listed in the element `<connection_list/>`. There should be only one connection group for any distinct set of objects in a χMCF file.

As discussed in 4.3, χMCF differs between 0-, 1- and 2-dimensional joints which will be specified in detail in the following clauses. Thus, an element `<connection_list/>` can comprise child elements `<connection_0d>`, `<connection_1d>` and `<connection_2d>` of arbitrary repetitions.

The XML-specification of `<connection_list/>` is given in Table 16:

Table 16 — Nested elements of element `<connection_list/>`

Nested elements	Multiplicity	Use	Constraint
connection_0d	*	optional	-
connection_1d	*	optional	-
connection_2d	*	optional	-

A `<connection_list/>` is not allowed to be empty, i.e. at least one connection must be defined.

7.5 Minimalistic example of a χMCF file

The following example shows how a χMCF file should look.

EXAMPLE

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<xmcf xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:MEDINA="http://servicenet.t-systems.com/medina/xMCF"
xsi:schemaLocation="http://servicenet.t-systems.com/medina/xMCF mcf_MEDINA.xsd"
xsi:noNamespaceSchemaLocation="xmcf_3_1_1.xsd">
    <!-- some comments -->
    <date> 2016-01-11 </date>
    <version> 3.1.1 </version>
    <units length="mm" angle="rad" mass="kg" force="N" time="s"/>
    <appdata> <!--appdata at root level -->
        <MEDINA xmlns="http://servicenet.t-systems.com/medina/xMCF">
            <data_at_root>
                <version MEDINA="MEDINA 8.4.2 Maintenance Release (64 Bit)"/>
                ...
            </data_at_root>
        </MEDINA>
    </appdata>
        ...
    <connection_group ...>
        <connected_to>
            <part index="1" label="PART_8000880" pid="20123213"/>
            <part index="2" label="PART_8100340" pid="90123213"/>
        </connected_to>
        <appdata> <!--appdata at connection_group level -->
            <MEDINA xmlns="http://servicenet.t-systems.com/medina/xMCF">
                <data_at_connection_group>
                    ...
                </data_at_connection_group>
            </MEDINA>
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

```
        </appdata>
        <connection_list>
            <connection_0d>
                <femdata>
                    <NASTRAN>
                        ...
                    </NASTRAN>
                </femdata>
               ...
            </connection_0d>
            <connection_1d>
                <loc_list>
                    ...
                </loc_list>
                <seamweld>
                    ...
                </seamweld>
                <appdata>
                    <MEDINA xmlns="http://servicenet.t-systems.com/medina/xMCF">
                        <data_at_connector>
                            ....
                        </data_at_connector>
                    </MEDINA>
                </appdata>
               ...
            </connection_1d>
            <connection_2d>
               ...
            </connection_2d>
            ...
        </connection_list >
    </connection_group>
    ...
</xmcf>
```

7.6 XML schema definition

The XML schema definition (XSD) of χMCF can be found in computer-interpretable form at the following URL: https://standards.iso.org/iso/pas/8329/ed-1/en/

8 Data common to any connection

8.1 Indices and their properties

χMCF provides several elements which are essentially ordered sets of the same data type (strings, integers, or decimals). More precisely they are like lists or vectors. For example, the `<loc_list/>` for the coordinate list of a seam weld or the `<string_list/>` in the `<custom_attributes/>`. Often the order of the elements in a set is essential. For instance, the coordinates in the `<loc_list/>` for a seam weld define the weld in the space uniquely by their values and their explicit order in the list.

The current XML standard allows that several child elements with an identical name share a common parent. However, it lacks a built-in mechanism to introduce a logical structure (like an order) in an XML-document. χMCF resolves this problem by introducing an index (attribute) in such cases. Indices may play a twofold role: to distinguish from each other and to ensure a unique arrangement in the list. Usually, an index can consist of strictly monotonically increasing natural numbers. In some cases, strictly monotonically increasing real numbers can also be appropriate. For example, the `<loc_list/>` for the coordinate list of a seam weld can be indicated both by real numbers like arc length of the line or any increasing integer series.

Depending on the context, the name of an index (attribute) is `index`, `v` or something else. They are always explicitly stated at the appropriate places in the text.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

8.2 Connection referencing

8.2.1 Need for referencing

Any connection should have a way of referring to it, since its shape and dimensions can vary along the design process. Typically, connections are referred to by their assigned ID or label.

8.2.2 Attribute `label`

Any connection should have an attribute called `label`, which identifies it throughout the entire CAx process, maybe even throughout the complete product lifecycle, including manufacturing. It is not necessary that these labels are unique: For instance, if a seam weld is split into different parts at a certain step in the process (if there are interfering holes in the structure, for example), its components shall keep the `label` attribute. A system downstream in the process (detached from any centralized naming authority) may create new connections with all the same label such as "0" or empty string.

The `label` may be composed of digits only, but it should not be confused with other IDs such as a finite element IDs. If desired, finite element IDs would have to be placed within some `<appdata/>` element.

8.2.3 Attribute `ident`

For systems or processes that use integers for referring to connections, the attribute `ident` is provided. In contrast to alphanumeric labels, integers are easy to generate and simple to shift when grouping is needed. This allows for unique identification, detached from a centralized naming authority, in case a connection is split, inserted, or duplicated.

`ident` can be used together with `label` as alternative ways of referring to a connection, bridging the gap between tools that work with integers only and tools that use labels only.

`ident` is a positive integer and unique within the χMCF file.

Example

```
<connection_list>
    <connection_0d label="SPOT_3490" ident="3490">
        <loc> ... </loc>
        <spotweld/>
    </connection_0d>
</connection_list>
```

8.3 Dimensions and coordinates

Connections can have three different dimensions: `<connection_0d/>`, `<connection_1d/>` and `<connection_2d/>`.

Any connection shall have coordinates. How many they are and how they are described depends on the connection's dimension. Details are described in the following subclauses.

8.4 Attribute `quality_control`

Some connections are more relevant than others, e.g. with respect to crash safety. Therefore, several levels of quality control are well established in manufacturing processes. For this reason, any connection can have an optional attribute `quality_control`. Since there is no general standard for such quality controls, χMCF cannot define a set of possible values for this attribute. Therefore, it must be of type `Alphanumeric`.

8.5 Custom attributes list

It was mentioned in 4.2 that only information relevant to connections should be contained in χMCF. The exceptions `<appdata/>` and `<femdata/>` were introduced in 7.3. These two elements essentially aim at specific needs of application software. The internal structure of `<appdata/>` itself is not standardized, can

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

be very complex, and depends on the specific software. The content can usually not be interpreted by other software systems.

Frequently, there are situations where a user of χMCF wishes to introduce supplementary information (attributes) to enrich the standard attributes defined by χMCF. In principle, the supplementary information could also be placed in an `<appdata/>` block, but with a substantial drawback, namely, its exchange between different commercial software tools will be difficult in case the tool specific internal structure is not documented.

With `<custom_attributes/>`, χMCF provides an element which is simple in handling and flexible enough to meet many requirements. All descendants of `<custom_attributes/>` are key-value-pairs, following the same pattern *key* ↔ *value(s)*, with supported *value-type* ∈ [*int*, *real*, *string*]^N, where *N* is a positive integer:

(*value-type*) *key* = {*value1, value2, …, valueN*}.

The case *N>1* is reminiscent of the vector or list templates from the STL of C++ (see Reference [8]) and is called "list" in χMCF.

In detail, the individual elements of `<custom_attributes/>` are of one of the following forms:

```
<int key="NameofIntValue"> value </int>
<int_list key="NameofIntListValue">
    <value index="1"> value1 </value>
    …
    <value index="N"> valueN </value>
</int_list>
<real key="NameofRealValue"> value </real>
<real_list key="NameofRealListValue">
    <value index="1"> value1 </value>
    …
    <value index="N"> valueN </value>
</real_list>
<string key="NameofStringValue"> value </string>
<string_list key="NameofStringListValue">
    <value index="1"> value1 </value>
    …
     <value index="N"> valueN </value>
</string_list>.
```

This means that the name of the element specifies the value-type while the value(s) is/are hold in one or several element(s) `<value/>`. A list is signified by the suffix "_list". All elements own the attribute `key`.

Often, the `<custom_attributes/>` element has an owner and is needed for a special purpose. For example, Mr. Brown needs for one and the same joint element an integer valued attribute named "priority" which should take on different values for two different applications "Fatigue" (1) and "Statics" (22). This can be specified in a `<custom_attributes_list/>` as follows:

EXAMPLE

```
<custom_attributes_list>
     <custom_attributes owner="Mr Brown" for="Fatigue">
            <int key="priority"> 1 </int>
     </custom_attributes>
     <custom_attributes owner="Mr Brown" for="Statics">
             <int key="priority"> 22 </int>
     </custom_attributes>
</custom_attributes_list>
```

In the above example, the `owner` is "Mr Brown" in both cases, while the applications can be distinguished by the attributes `for="Fatigue"` and `for="Statics"`, respectively.

The more general case that several `<custom_attributes/>` with different ownerships and for different purposes is considered by the element `<custom_attributes_list/>` with all the `<custom_attributes/>` elements as child-elements. No attributes are associated to the element `<custom_attributes_list/>`.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

The existence of a `<custom_attributes_list/>` inside a connection is optional. There can be at most one element inside each connection.

The `<custom_attributes_list/>` contains at least one of the following nested elements (see Table 17):

Table 17 — Nested elements of element `<custom_attributes_list/>`

Nested elements	Multiplicity	Use	Constraint
custom_attributes	1 - *	Required	-

At least one element `<custom_attributes/>` must be inside a `<custom_attributes_list/>`.

The XML specification of the `<custom_attributes/>` element is shown in Table 18:

Table 18 — Attributes of `<custom_attributes/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
owner	Alphanumeric	Alphanumeric	Required	Non-empty string
for	Alphanumeric	Alphanumeric	Optional	Non-empty string

The pair of attributes `owner` and `for` of each `<custom_attributes/>` element must be unique within each `<custom_attributes_list/>`.

The `<custom_attributes/>` element must contain at least one of the following nested elements (see Table 19):

Table 19 — Nested elements of element `<custom_attributes/>`

Nested elements	Multiplicity	Use	Constraints / Remarks
string	1 - *	optional	At least one of these nested elements is needed.
real	1 - *	optional	
int	1 - *	optional	
string_list	1 - *	optional	
real_list	1 - *	optional	
int_list	1 - *	optional	

The elements `<string/>`, `<real/>` and `<integer/>` must have the following data type assignments for their value, respectively:

— `<string/>`: alphanumeric value, which is covered by string data type in xsd, which can contain characters, line feeds, carriage returns, and tab characters.

If required to handle special characters such as line feeds or tabs, the normalized string data type should be used in the XSD. The normalized string data type is derived from the string data type. The normalized string data type also contains characters, but the XML processor will remove line feeds, carriage returns, and tab characters.

— `<real/>`: floating point value, which is covered by decimal data type in xsd, which can contain a numeric value. The maximum number of decimal digits you can specify is 18;

— `<integer/>`: integer value, which is covered by integer data type in xsd, which can contain a numeric value without a fractional component.

The XML specification of the `<string/>` element is shown in Table 20:

Table 20 — Attributes of `<string/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
key	Alphanumeric	Alphanumeric	Required	Non-empty string

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

The XML specification of the `<real/>` element is shown in Table 21:

Table 21 — Attributes of `<real/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
key	Alphanumeric	Alphanumeric	Required	Non-empty string

The XML specification of the `<integer/>` element is shown in Table 22:

Table 22 — Attributes of `<integer/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
Key	Alphanumeric	Alphanumeric	Required	Non-empty string

The XML specification of the `<string_list/>` element is shown in Table 23:

Table 23 — Attributes of `<string_list/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
key	Alphanumeric	Alphanumeric	Required	Non-empty string

`<string_list/>` has the nested element (see Table 24):

Table 24 — Nested elements of `<string_list/>` element

Nested elements	Type	Multiplicity	Use	Constraints / Remarks
value	**Alphanumeric**	**1 - ***	**required**	**-**

Where `<value/>` within `<string_list/>` is specified as (see Table 25):

Table 25 — Attributes of `<value/>` element inside `<string_list/>`

Attributes	Type	Value space	Use	Constraints / Remarks
index	Integer	>0	Required	unique within the parent element

The XML specification of the `<real_list/>` element is shown in Table 26:

Table 26 — Attributes of `<real_list/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
key	Alphanumeric	Alphanumeric	Required	Non-empty string

`<real_list/>` has the nested element as (see Table 27):

Table 27 — Nested element of `<real_list/>` element

Nested elements	Type	Multiplicity	Use	Constraints / Remarks
value	**Floating point**	**1 - ***	**Required**	**-**

Where `<value/>` within `<real_list/>` is specified as (see Table 28):

Table 28 — Attributes of `<value/>` element inside `<real_list/>`

Attributes	Type	Value space	Use	Constraints / Remarks
index	integer	>0	Required	unique within the parent element

The XML specification of the `<int_list/>` element is shown in Table 29:

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 29 — Attributes of `<int_list/>` element

Attributes	Type	Value space	Use	Constraints / Remarks
key	Alphanumeric	Alphanumeric	Required	Non-empty string

`<int_list/>` has the nested element (see Table 30):

Table 30 — Nested elements of `<int_list/>` element

Nested elements	Type	Multiplicity	Use	Constraints / Remarks
value	**integer**	**1 - ***	**Required**	**-**

Where `<value/>` within `<int_list/>` is specified as (see Table 31):

Table 31 — Attributes of `<value/>` element inside `<real_list/>`

Attributes	Type	Value space	Use	Constraints / Remarks
index	integer	>0	Required	unique within the parent element

Remarks:

a) Values of `key` must be unique within their common parent element,

b) The order of the values in the corresponding list is identified by the numerical value of their attribute `index`, in ascending order. Therefore, indices must be unique within one list,

c) In case of strings, the whitespaces deserve extra mention: To avoid mistakes, whitespaces are not to be used at beginning and end of a string.

EXAMPLE

```
<custom_attributes_list>
     <custom_attributes owner="DepartmentA" for="Fatigue">
            <int key="priority"> 1 </int>
            <string key="used S-N curve">Steel_225_ISO</string>
            <real key="fatigue_limit"> 223.1 </real>
     </custom_attributes>
     <custom_attributes owner="DepartmentA" for="Statics">
            <int key="priority"> 2 </int>
     </custom_attributes>
     <custom_attributes owner="DepartmentB">
            <string key="priority">high</string>
            <real_list key="direction vector">
                   <value index="1">10.3 </value>
                   <value index="2"> -2.1</value>
                   <value index="3">-1.5</value>
            </real_list>
            <string_list key="verifiedby" >
                   <value index="1">john</value>
                   <value index="2">Smith</value>
            </string_list>
     </custom_attributes>
</custom_attributes_list>
```

8.6 Distinction between `<custom_attributes/>` and `<appdata/>`

8.6.1 General

At first glance, `<custom_attributes/>` and `<appdata/>` seem to address similar purpose or even to be redundant. This is misleading, as the following subclauses show.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

8.6.2 Needs of different process roles, addressed by `<custom_attributes/>` and `<appdata/>`

In the context of χMCF, at least two different roles can be distinguished:

— the programmer of an application, and

— the engineer using this application.

The programmer needs to store extra data that are specific to the application.

The engineer needs to store additional data specific to the process in which the connections are involved.

As its name suggests, `<appdata/>` is used to store application-specific data, whose structure and purpose is known only by the application itself, respectively applies to this application alone. The software vendor can choose to standardize and publish the format of this data in order to allow other applications to port data to it or may choose to use `<appdata/>` as a private storage of internal state.

`<custom_attributes/>` represent OEM- or process-specific data, whose purpose is known by the engineers, but may not be known by the application.

Engineers choose which attributes they need to store and designate the corresponding data in `<custom_attributes/>`.

Applications store auxiliary data in `<appdata/>`. These data may be data that engineers do not need to know. `<appdata/>` may include information about the internal state of the application specific data model.

Engineers know the purpose and representation of `<custom_attributes/>`. The software may not know what each custom attribute represents, but it shall nevertheless be able to transport these data unchanged, or to offer a (generic) graphical user interface (GUI) for accessing it.

8.6.3 Needs of different applications, addressed by `<custom_attributes/>` and `<appdata/>`

`<appdata/>` may be used as means of intercommunication between different applications. In this case, the format of `<appdata/>` needs to be documented and published by the `<appdata/>` owner. However, it is not mandatory that information stored in `<appdata/>` is handled, maintained, or processed by third-party software. Therefore, `<appdata/>` should be considered as data that can be disregarded or thrown away by a third-party party software. Therefore, applications shall not rely on preservation of `<appdata/>`. Data corruption or crash must be avoided if data from `<appdata/>` gets lost.

The internal structure of `<custom_attributes_list/>` is completely standardized, whereas the internal structure of `<appdata/>` is arbitrary and can for optionally be described by a software-specific XML schema. Therefore, `<custom_attributes_list/>` cannot be used as flexible as `<appdata/>`, but its content is easier to be preserved across system boundaries.

8.6.4 Different levels of `<custom_attributes/>` and `<appdata/>` within χMCF data model

The elements `<appdata/>` and `<custom_attributes_list/>` have been introduced in 7.3.2 and 8.5, respectively. In this subclause, they are compared regarding their arrangement in the XML tree.

`<appdata/>` may be used either on different levels of a χMCF file:

— on root level (directly within the `<xmcf/>` element), or

— within any single connector (elements `<connection_0d/>`, `<connection_1d/>` and `<connection_2d/>`)

— or both.

In contrast to this, the element `<custom_attributes_list/>` can only be used within any single connector, but not at root level. This is justified below.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

In the usual scenario where multiple χMCF files, each containing connections of subsystems, are to be read and combined in an application, the application must deal with possible conflicts between data at root level (which applies to many connectors) and data at connector level (which applies to a single connector, only).

Each application should be able to resolve conflicts in its own `<appdata/>`, as the semantics of the data is known to the application.

On the other hand, the purpose of a `<custom_attributes/>` element assumed at root level would not be known to a random application. Conflicts with `<custom_attributes/>` elements at connector level would not even be identifiable with certainty. The application would therefore have to pass on the task of resolving `<custom_attributes/>` conflicts to the user. This is undesirable.

Because `<custom_attributes/>` may only be within a single connector, no conflicts can arise. Any application should be able to handle `<custom_attributes/>`, even if it does not understand the semantics. The reason for this is that the syntax is standardised and that connectors are self-contained objects with a clearly defined role and limited scope.

9 0D connections

9.1 Generic definitions

9.1.1 Identification

9.1.1.1 General

The possible XML attributes of the `<connection_0d/>` element, which describes a point connection (0D connection), are specified in Table 32:

Table 32 — Attributes of element `<connection_0d/>`

Attributes	Type	Use	Constraint
label	Alphanumeric	Optional	-
ident	Integer	Optional	positive, unique within a χMCF file
quality_control	Alphanumeric	Optional	See 8.4.

9.1.1.2 Attribute “label”

The label defines the human readable identification of a connection. It can contain a description of the connection or simply an index as an integer.

EXAMPLE 1 Minimum definition of a 0d connection without label

```
<connection_list>
    <connection_0d>
        <loc>
            ...
        </loc>
        <spotweld>
            ...
        </spotweld>
    </connection_0d>
</connection_list>
```

EXAMPLE 2 Definition of a 0d connection with label

```
<connection_list>
    <connection_0d label="SPOT_3490">
        <loc>
            ...
        </loc>
        <spotweld>
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

```
            ...
        </spotweld>
    </connection_0d>
</connection_list>
```

9.1.1.3 Attribute "ident"

The attribute `ident` provides an alternative identification to the connection. The value of `ident` is a positive integer and unique within the χMCF file.

EXAMPLE

```
<connection_list>
    <connection_0d ident="3490">
        <loc>
            ...
        </loc>
        <spotweld>
            ...
        </spotweld>
    </connection_0d>
</connection_list>
```

9.1.2 Location

The definition of the connection location of a point connection is described by the element `<loc/>`. This element is nested below the parent element `<connection_0d/>`. It contains three values specifying the x, y, and z coordinates of the location as text content (Table 33).

Table 33 — Text values of element `<loc/>`

Text	Type	Value space	Use	Constraint
x	Floating point	(−∞, +∞)	Required	-
y	Floating point	(−∞, +∞)	Required	-
z	Floating point	(−∞, +∞)	Required	-

EXAMPLE

```
<connection_0d>
    <loc> 2581.21 -708.408 31.6532 </loc>
    ...
</connection_0d>
```

9.1.3 Direction

The definition of connection directions, where applicable, is described by the elements `<normal_direction/>` and `<tangential_direction/>`. They both specify a direction vector. Lengths of both vectors are not relevant but must be > 0.

Their XML syntax is identical. Their names describe their semantics:

— Element `<normal_direction/>` denotes a direction of a local z axis.

— Element `<tangential_direction/>` denotes the direction of an axis tangential to the (base) part surface next to the point given in `<loc/>`, giving locale x axis. Its orthogonalization relative to `<normal_direction/>` is not allowed to vanish, that means both vectors are not allowed to be collinear.

If both elements are given, a right-handed coordinate system is uniquely defined:

— Origin is in `<loc/>`.

— z-axis is in direction of `<normal_direction/>`.

— x-axis is the orthogonalization of `<tangential_direction/>` relative to `<normal_direction/>`.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

— y-axis is given by cross product z-axis × x-axis.

The XML specification of the `<normal_direction/>` and `<tangential_direction/>` elements is shown in Table 34:

Table 34 — Attributes of elements `<normal_direction/>` & `<tangential_direction/>`

Attributes	Type	Value space	Use	Constraint
x	Floating point	(−∞, +∞)	Required	-
y	Floating point	(−∞, +∞)	Required	-
z	Floating point	(−∞, +∞)	Required	-

Both elements do not allow for any nested elements.

Unless otherwise stated, direction elements are optional. However, if omitted, importing systems may use a geometric search for determining `<normal_direction/>`, using a spherical characteristic, which may not be very reliable. In this situation, `<tangential_direction/>` can only be guessed, implying a random orientation of the connection (e.g. a robscan) in receiving system. Therefore, it is recommended for a receiving system to issue a warning, at least.

EXAMPLE

```
<normal_direction x="0.0" y="0.0" z="-1.0" />
<tangential_direction x="70.7" y="70.7" z="0.0" />
```

9.1.4 Type specification

Each connection should be assigned a type during its life cycle. The XML definitions of all 0D connections contain the following elements (see Table 35):

Table 35 — Nested elements of element `<connection_0d/>`

Nested elements	Multiplicity	Use	Constraint
clinch	1	Optional	-
clip	1	Optional	-
heat_stake	1	Optional	-
nail	1	Optional	-
gumdrop	1	Optional	-
rivet	1	Optional	-
robscan	1	Optional	-
rotation_joint	1	Optional	-
spotweld	1	Optional	-
threaded_connection	1	Optional	-
contact_list	1	Optional	See 7.4.3.6.
stacking	1	Optional	See 7.4.2.4

Up to one of the type elements (`clinch`, `clip`, `heat_stake`, `gumdrop`, `nail`, `rivet`, `robscan`, `spotweld`, or `threaded_connection`) may exist in `<connection_0d/>`. There is no default type.

9.2 Spot welds

9.2.1 General

A spot weld is denoted by an element `<spotweld/>`. This element is described completely by its attributes and nested elements (see Table 36).

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 36 — Nested elements of `<connection_0d/>` for `<spotweld/>`

Nested elements	Multiplicity	Use	Constraints / Remarks
spotweld	1	Optional	-
loc	1	Required	-
appdata	1	Optional	See 7.3.2.
femdata	1	Optional	See 7.3.3.
custom_attributes_list	1	Optional	See 8.5.

The XML specification of the `<spotweld/>` with element `diameter` is shown in Table 37:

Table 37 — Attributes of element `<spotweld/>`

Attributes	Type	Value space	Use	Constraint
diameter	Floating point	> 0,0	Optional	-
technology	Selection	resistance, laser, projection, friction	Optional	-

9.2.2 Attribute `diameter`

The diameter of a spotweld is specified by the attribute `diameter` for the child element of `<connection_0d/>`.

9.2.3 Attribute `technology`

The technology used to weld the connection can be specified for each of the spot welds of a connection separately.

This technology can be one of:

— Resistance welding,

— Laser welding,

— Projection welding,

— Friction welding.

The element `<spotweld/>` allows for following nested elements (see Table 38):

Table 38 — Nested elements of element `<spotweld/>`

Nested elements	Multiplicity	Use	Constraint
normal_direction	1	Optional	-
tangential_direction	1	Optional	-

EXAMPLE

```
<connection_0d label="SPOT_Left_Gh_2123921">
    <spotweld diameter="5.0"/>
    <loc> 1645.83 821.145 616.585 </loc>
    <appdata>
        ...
    </appdata>
</connection_0d>
```

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

9.3 Robscans

A Robscan is a pattern of arbitrary shape, drawn onto the flange partners by a laser optic. Such a shape has a length and width significantly larger than the diameter of the laser focus. The laser beam defines a local z-axis and is assumed to be perpendicular to the flange partners. However, the pattern can be rotated around this z-axis, and it can be mirrored at its x-axis. This is depicted in Figure 8:

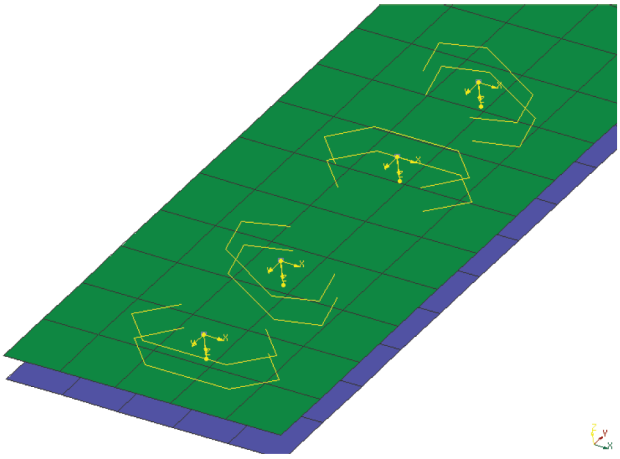


Figure 8 — Robscans with different rotation angles, two of them mirrored

The pattern of the bottom left Robscan is oriented with no rotation and no mirroring with respect to its own coordinate system (yellow). The next instance has 30° rotation. The two Robscans, top right in the figure, have a mirrored pattern; the uppermost having again 30° rotation.

There is a continuum of patterns for Robscans. Each one which shall be used at an assembly line needs to be verified (by simulation plus test) in advance, which is expensive. Some implications are:

— companies regard this information to be their own intellectual property,

— a pattern shall not simply be stretched etc. It would need a new validation,

— validated Robscan patterns are usually not part of distributions of FE processors.

However, subcontractors possibly need to know the position and bounding box of the Robscan. Therefore, χMCF definition shall contain some “abstract” data. FE processors may address the danger of inconsistency by taking both levels of information from the same configuration file. Therefore, it is at the responsibility of the companies’ admins to have consistent data in that file.

Since the exact shape of the Robscan pattern is third-party intellectual property, it cannot be part of the χMCF definition. It is referred to by just a string attribute “pattern”. Possible values of attribute “pattern” are not the subject of this document: In general, they are very original equipment manufacturer (OEM) specific. However, to provide a minimum amount of information, width and length of the pattern are given by attributes `pattern_width` and `pattern_length`.

A Robscan is denoted by an element `<robscan/>`. This element is described completely by its attributes and nested elements, see Table 39, Table 40, and Table 41.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 39 — Nested elements of `<connection_0d/>` for `<robscan/>`

Nested elements	Multiplicity	Use	Constraints / Remarks
robscan	1	Optional	-
loc	1	Required	-
appdata	1	Optional	See 7.3.2.
femdata	1	Optional	See 7.3.3.
custom_attributes_list	1	Optional	See 8.5.

The XML specification of the `<robscan/>` element is shown in the following Table 40:

Table 40 — Attributes of element `<robscan/>`

Attributes	Type	Value space	Use	Constraints / Remarks
base	Integer	> 0	Optional	-
pattern	Alphanumeric	Alphanumeric	Optional	Non-empty, if present.
gap	Floating point	>= 0,0	Optional	-
width	Floating point	> 0,0	Optional	-
pattern_width	Floating point	> 0,0	Optional	-
pattern_length	Floating point	> 0,0	Optional	-
mirrored	Boolean	"false" (default), "true"	Optional	-
orientation_angle	Floating point	[-180°, 180°]	Optional	According to the unit of angles, defined in element `<units/>`.
filler_material	Alphanumeric	Alphanumeric	Optional	-

All attributes of element `<robscan/>` are optional for import to CAD or CAE processors. However, specific FE solvers may declare some of them to be mandatory.

General defaults are:

— `false` – for Boolean values,

— `0` – for numeric values,

— `""` – for strings.

However, these defaults are not always useful for CAE:

— `gap`: this defines the gap between both flange partners (perpendicular to the surface);

— `width`: this is the width of the laser beam;

— Width and length of the pattern are given by attributes `pattern_width` and `pattern_length`;

— `mirrored`: this denotes, whether the pattern has to be mirrored along its length-axis x, i.e. local y coordinate has to be inverted;

— `orientation_angle`: this defines a rotation around z axis, following right-hands-rule. Angle is measured in the unit of angles, defined in element `<units/>`, within range [–180°, 180°]. –180° and +180° degree are regarded to be identical (applies also to equivalent ranges in another unit).

Both parameters, `mirrored` and `orientation_angle` address numerical optimization: An angle and a boolean allow to vary the Robscan placement easier than to calculate completely new orientation vectors.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 41 — Nested elements of element `<robscan/>`

Nested elements	Multiplicity	Use	Constraint
normal_direction	1	Optional	-
tangential_direction	1	Optional	-

Additional explanations for the directions are:

— Element `<normal_direction/>` denotes direction of laser beam, giving local z axis;

— Element `<tangential_direction/>` denotes laser moving direction, giving local x axis.

`<normal_direction/>` and `<tangential_direction/>` elements are described in 9.1.3.

EXAMPLE

```
  <connection_0d label="RSC_1272360">
      <loc> 507 1 0.8 </loc>
      <robscan base="1" pattern="KL_ST" gap="0.15" width="0.4" mirrored="false" pattern_
width="5" pattern_length="12" orientation_angle="0">
          <normal_direction x="0" y="0" z="-1"/> <!-- locale z axis -->
          <tangential_direction x="1" y="0" z="0"/> <!-- locale x axis -->
      </robscan>
      <appdata>
          ...
      </appdata>
  </connection_0d>
```

9.4 Rivets

9.4.1 General

There are many different types of rivets. If at some state of the model the specific type of rivet (e.g. blind rivet, self-piercing rivet) is not known, then a generic rivet element should be used to capture just the necessary information, like direction, length, and diameter.

A rivet is denoted by an element `<rivet/>`. This element is described completely by its attributes and nested elements, see Table 42 and Table 43.

Table 42 — Nested elements of `<connection_0d/>` **for** `<rivet/>`

Nested elements	Multiplicity	Use	Constraints / Remarks
rivet	1	Optional	-
loc	1	Required	-
appdata	1	Optional	See 7.3.2.
femdata	1	Optional	See 7.3.3.
custom_attributes_list	1	Optional	See 8.5.

The XML specification of the `<rivet/>` element is shown in Table 43 and rivet head types are shown in Figure 9:

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Table 43 — Attributes of element `<rivet/>`

Attributes	Type	Value space	Use	Constraints / Remarks
hardness	Floating point	> 0,0	Optional	-
shaft_diameter	Floating point	> 0,0	Optional	-
length	Floating point	> 0,0	Optional	-
head_diameter	Floating point	> 0,0	Optional	-
head_height	Floating point	≥ 0,0	Optional	If at least one of them is specified *head_height + sink_size > 0* is required.
head_type	Alphanumeric	Alphanumeric	Optional	-
sink_size	Floating point	≥ 0,0	Optional	-
strength_property_class	Alphanumeric	Alphanumeric	Optional	-
part_code	Alphanumeric	Alphanumeric	Optional	-

a) Dome **b) Large flange** **c) Countersunk**

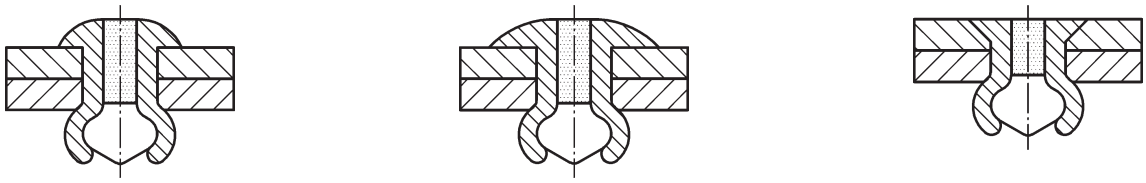


Figure 9 — Rivet head types (dome, large flange, countersunk)

The following list explains the attributes:

— `hardness`: Vickers hardness HV (see Reference [9]) of the rivet material. (Attribute hardness was moved from element `<self_piercing/>` to element `<rivet/>` with χMCF version 3.1.);

— `shaft_diameter`: the diameter of the shaft of the (unmounted) rivet;

— `length`: the overall length of the (unmounted) rivet itself;

— `head_diameter`: the diameter of the head of the (unmounted) rivet;

— `head_height`: the height of the head;

— `head_type`: the description of head type ("dome", "countersunk" or "large_flange");

— `sink_size`: the size of the head that is sunk;

— `strength_property_class`: Strength according to ISO, EN, BSW, DIN, SAE etc., e.g. Reference [10].

— `part_code`: the part code of the rivet, as used e.g. in a PDM system. It can be convenient to use the rivet norm (according to e.g. ISO, EN, BSW, DIN) as part code.

If possible, a rivet should know the direction of fixation, therefore, possess a nested element `<normal_direction/>`. However, this is not mandatory so that incomplete data can also be imported. The direction sense of `<normal_direction/>` (see 9.1.3) is from rivet head to foot.

A `<tangential_direction/>` can be provided for rivets that are not axis-symmetric and require a spatial orientation.

A `<rivet/>` is always placed into holes drilled before, whereas its subtype `<self_piercing/>` creates its own hole during placement.

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Specific subtypes of rivets are defined by adding nested elements, listed in following Table 44:

Table 44 — Nested elements of element `<rivet/>`

Nested elements	Multiplicity	Use	Constraint / Remarks
normal_direction	1	Optional	-
tangential_direction	1	Optional	-
blind self_piercing solid swop clinch_rivet_stud	1	Optional	Maximum one of the listed elements.

The subtypes are described in detail in the 9.4.2 to 9.4.6.

EXAMPLE 1 Example for a (axisymmetric) rivet connection that uses only the <normal_direction/>

```
<connection_0d label="RVT_2123921">
    ...
    <rivet shaft_diameter="5.0" head_diameter="8" length="3.5">
        <normal_direction x="0" y="0" z="3"/>
    </rivet>
    <loc> 1645.83 821.145 616.585 </loc>
    <appdata>
        ...
    </appdata>
</connection_0d>
```

EXAMPLE 2 Example for a rivet connection that requires also the `<tangential_direction/>`

```
<connection_0d label="RVT_2123922">
    ...
    <rivet shaft_diameter="5.0" head_diameter="8" length="3.5">
        <normal_direction x="0" y="0" z="3"/>
        <tangential_direction x="3" y="0" z="0"/>
    </rivet>
    <loc> 1645.83 -821.145 616.585 </loc>
    <appdata>
        ...
    </appdata>
</connection_0d>
```

9.4.2 Blind rivets

Blind rivets are one-sided rivets that require a pre-drilled hole. Blind rivets form their shape when the mandrel is pulled out from the rivet body. This action securely clamps the sheets together. A blind rivet is denoted by a nested element `<blind/>` within `<rivet/>`. This element is described completely by its attributes and those of `<rivet/>`.

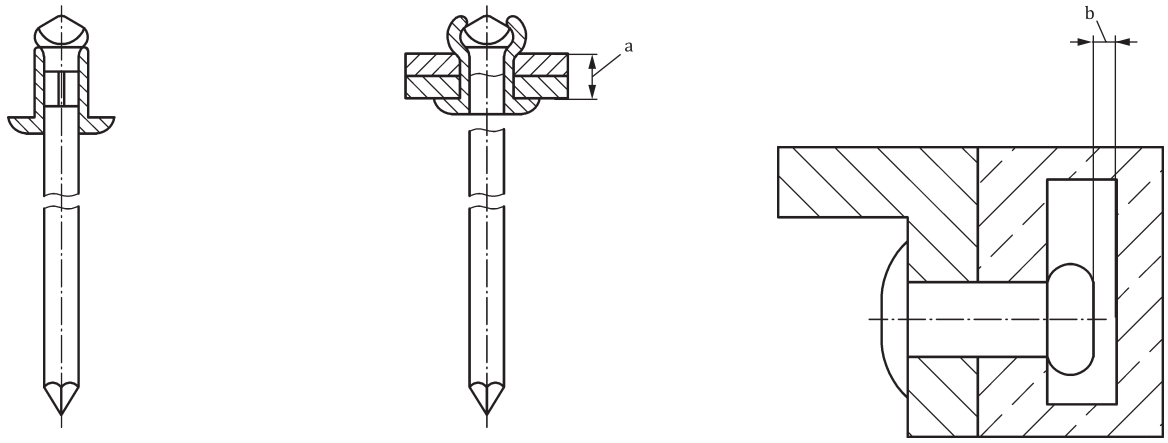
The XML specification of the `<blind/>` element is shown in Table 45:

Table 45 — Attributes of element `<blind/>`

Attributes	Type	Value space	Use	Constraint / Remarks
min_grip	Floating point	> 0,0	Optional	-
max_grip	Floating point	> 0,0	Optional	greater equal to min_grip
clearance	Floating point	> 0,0	Optional	-
material	Alphanumeric	Alphanumeric	Optional	material of the rivet body

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024



a) Before riveting **b) After riveting** **c) Application example**

a Grip.

b Clearance (blind side).

Figure 10 — Blind rivet — Key attributes

Figure 10 describes what the attributes of `<rivet/>` and `<blind/>` correspond to:

— `min_grip`, `max_grip`: These two attributes collectively describe the effective grip range of the rivet. A blind rivet is engineered so that it can be used for a specific range of material thickness for which it provides proper joining between connected parts. This can be called as the blind rivet's grip range.

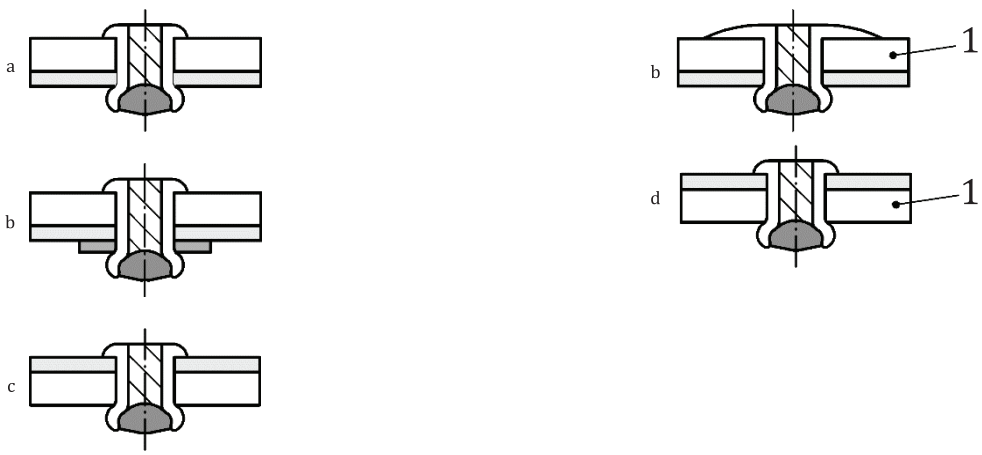
— `clearance`: The blind rivet needs some clearance on the blind side, which is the side of mandrel head, when inserted into the holes, before it is applied.

— `material`: This attribute defines the applied material of the blind rivet body. Generally, the applied rivet should be used with connected parts so that the connector rivet element has the same physical and mechanical properties as the components to be joined. Usual materials: Steel, Stainless Steel, Nickel Copper Alloy (Monel), Copper and several grades of Aluminium.

NOTE If material thickness changes in connected parts, this can lead to other size of blind rivet as joining element.

If a blind rivet is applied to join two components which have different mechanical properties, e.g. one of them is thinner or softer than the other, then the `<normal_direction/>` element will become more important to show the proper setting direction of the rivet as seen in Figure 11.

a) Thin / Thick assembly **b) Soft / Hard assembly**



© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024

Key

1 soft part

[a] Satisfactory.

[b] Good.

[c] Better.

[d] Poor.

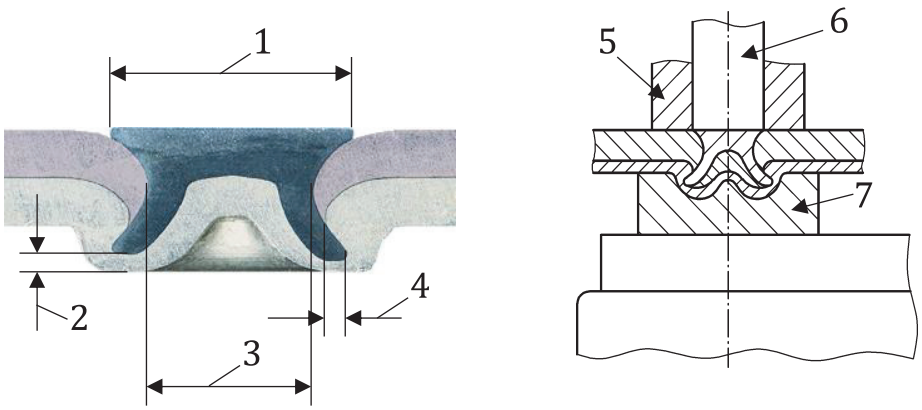
Figure 11 — Assembly recommendations for blind rivets

EXAMPLE

```
<connection_0d label="RVT_2123921">
    <rivet shaft_diameter="3.35" head_diameter="5.5" head_type="dome" length="4">
           <blind min_grip="3" max_grip="3.2" clearance="4.5" material="Steel"/>
           <normal_direction x="0.0" y="1.5" z="3.0"/>
    </rivet>
    <loc> 1645.83 821.145 616.585 </loc>
    <appdata>
           ...
    </appdata>
</connection_0d>
```

9.4.3 Self-piercing rivets

A self-piercing rivet is a special kind of rivet which does not need a pre-drilled hole. Originally a hollow cylinder with a cap on one end, it deforms together with the material it is pushed into as shown below in Figure 12:



Key

1 rivet head diameter

2 bottom thickness

3 rivet diameter

4 undercut

5 blank holder

6 punch

7 die

Figure 12 — Cross-section of a self-piercing rivet and riveting device

A wide range of such rivets is available. They can be used with different rivet dies to optimize the riveting process. Such combinations must be chosen in accordance with the materials of the flange partners. Details are third party intellectual property and therefore cannot be part of the χMCF definition. However, χMCF offers alphanumeric attributes for rivet and die parameters. Possible values of these attributes are

© ISO 2024 – All rights reserved

Documents
Document Sharing Hub
Full document details: PD ISO/PAS 8329:2024