INTERNATIONAL STANDARD

ISO/IEC 15909-2

First edition
2011-02-15

Systems and software engineering — High-level Petri nets —

Part 2: Transfer format

Ingénierie des systèmes et du logiciel — Réseaux de Petri de haut niveau —

Partie 2: Format de transfert

Reference number
ISO/IEC 15909-2:2011(E)

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

Published in Switzerland

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 15909-2 was prepared by Joint Technical Committe ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

ISO/IEC 15909 consists of the following parts, under the general title *Systems and software engineering – High-level Petri nets*:

— *Part 1: Concepts, definitions and graphical notation*

— *Part 2: Transfer format*

“Extensions” will form the subject of a future Part 3.

Introduction

ISO/IEC 15909 is concerned with defining a modelling language and its transfer format, known as *High-level Petri Nets*. ISO/IEC 15909-1 provides the mathematical definition of *High-level Petri Nets*, called the semantic model, the graphical form of the technique, known as *High-level Petri Net Graphs* (HLPNGs), and its mapping to the semantic model. It also introduces some common notational conventions for HLPNGs.

This part of ISO/IEC 15909 defines a transfer format for *High-level Petri Nets* in order to support the exchange of *High-level Petri Nets* among different tools. This format is called the *Petri Net Markup Language* (*PNML*). Since there are many different versions of *Petri nets* in addition to *High-level Petri Nets*, this part of ISO/IEC 15909 defines the *core concepts* of *Petri nets* along with an XML syntax, which can be used for exchanging any kind of *Petri net*. Based on this *PNML Core Model*, this part of ISO/IEC 15909 also defines the transfer syntax for the three versions of *Petri nets* that are defined in ISO/IEC 15909-1: *Place/Transition Nets*, *Symmetric Nets*[1], and *High-level Petri Nets*, where *Place/Transition Nets* and *Symmetric Nets* can be considered to be restricted versions of *High-level Petri Nets*. For *Place/Transition Nets*, this part of ISO/IEC 15909 introduces two different transfer formats: one is a format specifically tuned to *Place/Transition Nets*, the other is a format that represents *Place/Transition Nets* as a restricted version of *High-level Petri Nets* as defined in ISO/IEC 15909-1.

The basic level of conformance to this part of ISO/IEC 15909 is to the *PNML Core Model*. The other levels are according to the particular type of the *Petri net*; for *High-level Petri Nets* there are two levels of conformance: *textual conformance* ignores the exact syntax and structure of the *labels*; *structural conformance* requires that *labels* are given in the exact syntax as defined here. Since *Symmetric Nets* are designed for analysability, *textual conformance* does not make any sense for *Symmetric Nets*; therefore, there is only *structural conformance* for *Symmetric Nets*.

Note that this part of ISO/IEC 15909 introduces some concepts that are not defined in ISO/IEC 15909-1. These concepts are not related to the mathematical concepts of *Petri nets* and their semantics. They concern the graphical representation of nets and the structuring of large *Petri net* models. These concepts need to be defined, along with a transfer format for *Petri nets*, in order to ensure that the graphical appearance of a *Petri net* in different tools is similar.

This part of ISO/IEC 15909 is structured as follows: Clause 1 describes the scope, the areas of application and the intended audience of this part of ISO/IEC 15909. Clause 2 defines conformance. Clause 3 gives references that are essential for the correct interpretation of this International Standard. Clause 4 defines all terms relevant to this International Standard and includes a list of abbreviations. Clause 5 introduces the concepts of PNML using UML meta models. Clause 5.2 defines the *PNML Core Model*, which is the structure common to all versions of *Petri nets*. Clause 5.3 defines the particular concepts of the different *Petri net types*. Clause 6 provides the mapping of the syntactical concepts defined in this part of ISO/IEC 15909 to the concepts defined in ISO/IEC 15909-1. Clause 7 defines how the concepts of PNML as defined in Clause 5 are mapped to XML syntax.

Annex A defines the exact XML syntax for the *PNML Core Model* in terms of a RELAX NG grammar. Annex B defines the exact XML syntax for the different types of *Petri nets*. Annex C provides a small example for the syntax of a symmetric net. Annex D discusses a framework for implementing this International Standard and an API for accessing *Petri nets*, which is based on the UML models for the PNML meta models.

[1] Symmetric nets were first introduced as well-formed nets and are currently standardized as ISO/IEC 15909-1:2004/Amd. 1:2010.

Systems and software engineering – High-level Petri nets – Part 2: Transfer format

1 Scope

This part of ISO/IEC 15909 defines an XML-based transfer format for *Petri nets*, which are defined conceptually and mathematically in ISO/IEC 15909-1. This transfer format enables the exchange of *Petri nets* among different *Petri net* tools and among different parties. Moreover, this part of ISO/IEC 15909 defines some concepts and XML-based syntax for defining the detailed graphical appearance of *Petri nets*.

The focus of this part of ISO/IEC 15909 is on the transfer format for *Place/Transition Nets*, *High-level Petri Nets* and *Symmetric Nets*. The presentation, however, is structured in such a way that it is open for future extensions, so that other versions of *Petri nets* can be added later. The exact definition of this extension mechanism, called *Petri net type definition*, is not defined in this part of ISO/IEC 15909; it will be defined in ISO/IEC 15909-3.

The transfer format will be used to transfer specifications of systems developed in *High-level Petri Nets* between tools to facilitate the development of systems in teams.

This part of ISO/IEC 15909 is written as a reference for developers of *Petri net* tools. Moreover, it will be useful for researchers who define new versions and variants of *Petri nets*.

2 Conformance

There are different levels of conformance to this part of ISO/IEC 15909. All conformance levels impose additional conditions on valid XML documents.

2.1 PNML Documents

An XML document is conformant to the *PNML Core Model* if it meets the definitions of Clause 5.2 (concepts) and Clause 7.1 (their mapping to XML syntax) – such a document is called a *PNML Document* or a *Petri Net Document*. A Petri net tool is conformant to the *PNML Core Model* if it can import all *PNML Documents* and if it can export all *Petri nets* to a *PNML Document*.

The other levels of conformance concern the different *Petri Net Types*.

2.2 PNML Place/Transition Net Documents

A *PNML Document* is a conformant *Place/Transition Net* if it meets the additional restrictions of Clause 5.3.1 (concepts of P/T-nets) and Clause 7.2 (their mapping to XML syntax) – such a document is called a *PNML Place/Transition Net Document*. A Petri net tool is conformant to the *PNML Place/Transition Net* definition if it can import all *PNML Place/Transition Net Documents*, and if it can export all *Place/Transition Nets* to *PNML Place/Transition Net Documents*. Note that this transfer format is tuned to *Place/Transition Nets*. There is another format, which considers *Place/Transition Nets* as a restricted form of *High-level Petri Nets* (see Clause 2.5).

2.3 Textually conformant PNML High-level Petri Net Documents

For *High-level Petri Nets*, there are two different levels of conformance. The first level requires the existence of textual labels as defined in Clause 5.3.2 and 5.3.11 (concepts) and Clause 7.3 (mapping to XML syntax). But,

it does not require the existence of the structural parts of the *annotations*; it only requires that the textual parts of the *annotations* exist, but the text is not required to be in a specific syntax and, therefore, the meaning of it cannot be transferred to other tools. Such a *PNML Document* is called a *textually conformant PNML High-level Petri Net Document*. A Petri net tool is conformant to the *textual PNML High-level Petri Net* definition if it can import all *textually conformant PNML High-level Petri Net Documents*, and if it can export all *High-level Petri Nets* to a *textually conformant PNML High-level Petri Net Document*.

2.4 Structurally conformant PNML High-level Petri Net Documents

The second level of *High-level Petri Net* conformance requires that all *annotations* obey the rules defined in Clause 5.3.2 and 5.3.11 (concepts) and Clause 7.3 (mapping to XML syntax). Such a *PNML Document* is called a *structurally conformant PNML High-level Petri Net Document*. A Petri net tool is conformant to the *structural PNML High-level Petri Net* definition if it can import all *structurally conformant PNML High-level Petri Net Documents*, and if it can export all *High-level Petri Nets* to a *structurally conformant PNML High-level Petri Net Document*.

2.5 Place/Transition Net Document in High-level Notation

A *structurally conformant PNML High-level Petri Net Document* that uses only the single sort *dot* and only the *arc annotations* of *Place/Transition Nets* is a *conformant Place/Transition Net Document in High-level Notation*.

2.6 Symmetric Net Documents

Finally, there is conformance to *Symmetric Nets*, which is a restricted version of *High-level Petri Nets*. A *Symmetric Net Document* is a *structurally conformant PNML High-level Petri Net Document* if it contains only the concepts defined in Clause 5.3.10. A Petri net tool is conformant to the *Symmetric Net* definition if it can import all *Symmetric Net Documents*, and if it can export all *Symmetric Nets* to *Symmetric Net Documents*.

3 Normative references

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

ISO/IEC 15444 (all parts), *Information technology – JPEG 2000 image coding system*

ISO/IEC 15909-1, *Systems and software engineering – High-level Petri nets – Part 1: Concepts, definitions and graphical notation*

ISO/IEC 15948, *Information technology – Computer graphics and image processing – Portable Network Graphics (PNG): Functional specification*

ISO/IEC 19757-2:2008, *Information technology – Document Schema Definition Language (DSDL) – Part 2: Regular-grammar-based validation – RELAX NG*

CSS, *Cascading Style Sheets, level 2 revision 1, CSS 2.1 Specification; W3C Candidate Recommendation, 25 February 2004*

OCL 2.0, *Object Constraint Language, OMG Available Specification, Version 2.0. OMG formal/06-05-01, May 2006*

UML 2.1, *OMG Unified Modeling Language (OMG UML): Superstructure, V2.1.2 OMG Available Specification, November 2007*

XML 1.1, *Extensible Markup Language (XML) 1.1 (Second Edition); W3C Recommendation, 29 September 2006*

XML Schema Datatypes: *XML Schema Part 2: Datatypes (Second Edition); W3C Recommendation, 28 October 2004*

4 Terms, definitions and abbreviations

4.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 15909-1 and the following apply.

4.1.1
annotation
label (4.1.5) represented as text near to the **object** (4.1.7) it is associated with

4.1.2
attribute
label (4.1.5) that governs the form or shape of the **object** (4.1.7) it is associated with, which, in contrast to an **annotation** (4.1.1), is typically not shown as text

4.1.3
graphical information
information defining the graphical appearance of **objects** (4.1.7) and **labels** (4.1.5) of a **net graph**, which can be the position, size, line colour, fill colour, font or line width

4.1.4
global label
label (4.1.5) associated with the **net graph** itself, rather than with an **object** (4.1.7) of a **net graph**

4.1.5
label
information associated with the **net graph** or one of its **object** (4.1.7)

4.1.6
meta model
model defining the concepts and their relations for some modelling notation

NOTE The meta models in this part of ISO/IEC 15909 are mainly UML class diagrams.

4.1.7
object (of a net graph)
arc, **node**, **reference node** (4.1.14), or **page** (4.1.8) of a **net graph**

4.1.8
page
structuring mechanism used to split a large **net graph** into smaller parts, which are also the units of the net to be printed

4.1.9
PNML Core Model
meta model (4.1.6) defining the basic concepts and structure of **net graph** models that are common to all versions of **Petri nets**

4.1.10
PNML Document (Petri Net Document)
XML document that contains one or more **net graphs**

4.1.11
PNML High-level Net Document
PNML Document (4.1.10) that contains one or more **net graphs**, where all **net graphs** conform to **High-level Petri Nets**

4.1.12
PNML Place/Transition Net Document
PNML Document (4.1.10) that contains one or more **net graphs**, where all **net graphs** conform to **Place/Transition Nets**

4.1.13
PNML Symmetric Net Document
PNML Document (4.1.10) that contains one or more **net graphs**, where all **net graphs** conform to **Symmetric Nets**

4.1.14
reference node
node of a **Petri net** that is a representative of another **node**, possibly defined on another **page** (4.1.8) of the **net graph**

4.1.15
reference place
reference node (4.1.14) that represents a **place** and refers to either another **reference place** (4.1.15) or to a **place**

4.1.16
reference transition
reference node (4.1.14) that represents a **transition** and refers to either another **reference transition** (4.1.16) or to a **transition**

4.1.17
source node
node associated with the start of an **arc**

4.1.18
target node
node associated with the end of an **arc**

4.1.19
tool specific information
information associated with an **object** (4.1.7) of a **net graph** or with the **net graph** itself that is specific to a particular tool and is not meant to be used by other tools

4.2 Abbreviations

4.2.1
CSS
Cascading Stylesheets

4.2.2
CSS2
Cascading Stylesheets level 2

4.2.3
OCL
Object Constraint Language

4.2.4
P/T Net
Place/Transition Net

4.2.5
PNML
Petri Net Markup Language

4.2.6
RELAX NG
Regular Language description for XML/ New Generation

4.2.7
SN
Symmetric Net

4.2.8
UML
Unified Modeling Language

4.2.9
XML
eXtensible Markup Language

5 Concepts

A *Petri net* can be considered to be a labelled graph, where the particular *labels* depend on the particular type of *Petri net*. The concepts for *High-level Petri Net Graphs* are defined in part 1 of ISO/IEC 15909 and are represented in terms of a meta model in UML notation in this clause. The mapping of these concepts to XML syntax is defined in Clause 7.

Some concepts, however, have not been defined in part 1 because the main concern of part 1 was on the semantics of *High-level Petri Nets*. Part 2 of ISO/IEC 15909 defines concepts for defining the graphical appearance of *Petri nets* in order to enable similar graphical representation in different tools. Moreover, part 2 defines the concept of *pages* in order to be able to split the graphical representation of a *Petri net* into smaller units for printing and viewing them.

Note that the concepts and definitions presented in part 1 of ISO/IEC 15909 are semantic in nature, whereas part 2 is syntactic in nature. For example, part 1 refers to *types*, which are sets, but does not mandate any concrete syntax for transferring such sets. Likewise part 1 uses *functions*, but does not mandate a syntax for transferring them. In order to transfer *High-level Petri Nets*, this part of ISO/IEC 15909 needs to introduce a syntax for *types* and *functions*. To this end, it uses the concepts of *sorts* and *operators*. These are syntactic symbols for denoting *types* and *functions*, and are assigned a fixed interpretation, which is defined along with the *sorts* and *operators*. Therefore, this part of ISO/IEC 15909 often refers to *sorts* and *operators*, where part 1 refers to their semantical counterpart, *types* and *functions*. By the fixed interpretation of *sorts* and *operators*, the respective *types* and *functions* are also defined.

5.1 General Principles

This part of ISO/IEC 15909 defines a transfer format for *Place/Transition Nets*, *High-level Petri Net Graphs*, and *Symmetric Nets* as defined in part 1 of ISO/IEC 15909. This transfer format has been designed to be extensible and to be open for future variants of *Petri nets* and possibly for other use, such as the transfer of results associated with the analysis of Petri nets, e. g., reachability graphs. In order to obtain this flexibility, the transfer format considers a *Petri net* as a labelled directed graph, where all type specific information of the net is represented in *labels*. A *label* may be associated with a *node*, an *arc*, a *page* or the *net* itself. This basic structure is captured in the *PNML Core Model*, which is defined in Clause 5.2.

The *PNML Core Model* is presented using UML class diagrams. Note that these UML diagrams do not define the concrete XML syntax for *PNML Documents*; rather, they define the abstract syntax only. The mapping of the *PNML Core Model* elements (i. e., the abstract syntax) to the concrete XML syntax will be defined in Clause 7; Annex A defines the exact XML syntax in terms of a RELAX NG grammar.

The *PNML Core Model* imposes no restrictions on *labels*. Therefore, the *PNML Core Model* can represent any kind of *Petri net*. Due to this generality of the *PNML Core Model*, there can be *PNML Documents* that do not correspond to a *Petri net* at all. For example, there could be *labels* from two different and even incompatible versions of *Petri nets* within the same *PNML Document*. For a concrete version of *Petri nets*, the legal *labels* are defined by extending the *PNML Core Model* with another meta model that exactly defines the legal *labels* of this type.

Technically, the *PNML Core Model* is a UML package, and there are additional UML packages for the different *Petri net types* that extend the *PNML Core Model* package. This part of ISO/IEC 15909 defines a package for *Place/Transition Nets*, a package for *Symmetric Nets*, and a package for *High-level Petri Net Graphs*, where the package for *High-level Petri Net Graphs* extends the package for *Symmetric Nets*. Therefore, every *Symmetric Net* is also a *High-level Petri Net Graph*. The representation of these concepts in XML syntax is defined in Clause 7.2; Annex B defines the exact XML syntax of the concepts for the different types in terms of RELAX NG grammars.

Figure 1 shows the different packages and how they are related. The package *PNML Core Model* defines the basic structure of *Petri nets* or *net graphs*; this structure is extended by the package for each type. The *PNML Core Model* is defined in Clause 5.2, the package PT-Net is defined in Clause 5.3.1, the package SymmetricNet is defined in Clause 5.3.10 and the package HLPNG is defined in Clause 5.3.11. Clause 7 shows how the concepts defined in these packages are represented in concrete XML syntax.

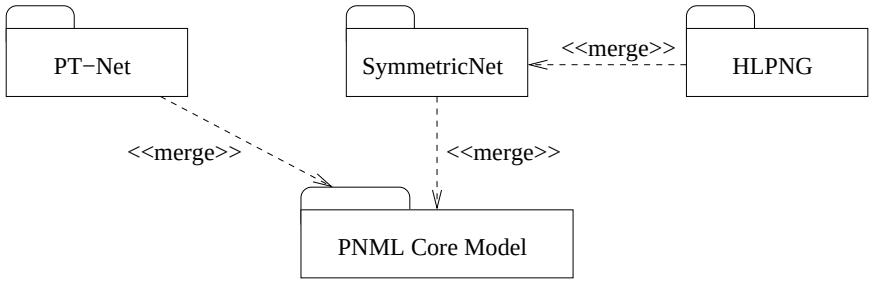

Figure 1: Overview of the UML packages of PNML

5.2 PNML Core Model

Figures 2 and 3 show the *PNML Core Model* as a UML class diagram. The diagram of Fig. 2 focuses on the conceptual parts, whereas the diagram of Fig. 3 focuses on the parts concerning the graphical representation. Note that the data type *String* is imported from a separate package *XMLSchemaDataTypes*, which is discussed in Clause 5.2.6.

The concepts defined in the *PNML Core Model* are discussed in the following clauses.

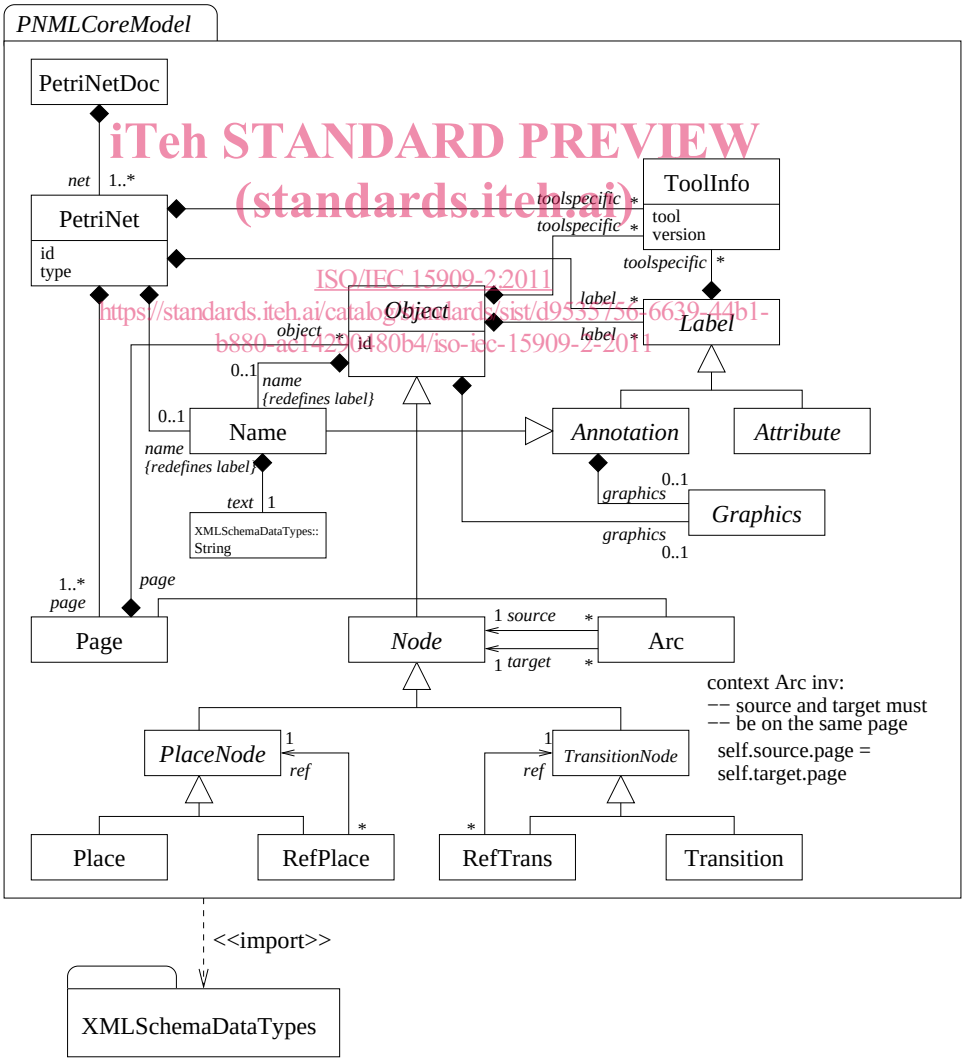

Figure 2: The *PNML Core Model* package: concepts

5.2.1 Petri Net Documents, Petri Nets, and Objects

A document that meets the requirements of the *PNML* Core Model is called a *Petri Net Document* (PetriNetDoc) or a *PNML Document*. It contains one or more *Petri Nets* (PetriNet). Each *Petri Net* has a unique identifier and a *type*. The *type* is a name referring to the package with its definition; an example for such a package name is `http://www.pnml.org/version-2009/grammar/ptnet` for the package defining *P/T Nets*. The URL for the definition of PNML itself is `http://www.pnml.org/version-2009/grammar/pnml`.

A *Petri net* consists of one or more top-level *pages* that in turn consist of several *objects*. These *objects*, basically, represent the graph structure of the *Petri net*. Each *object* within a *Petri net document* has a unique *identifier*, which can be used for referring to this *object*. Moreover, each *object* may be equipped with graphical information defining its position, size, colour, shape and other attributes on its graphical appearance. The precise graphical information that can be provided for an *object* depends on the particular type of *object* (see Clause 5.2.4 and Fig. 3).

The main *objects* of a *Petri net* are *places*, *transitions* and *arcs*. For convenience, a *place* or a *transition* is generalized to a *node*. For reasons explained in Clause 5.2.2 below, this generalization is via *place nodes* and via *transition nodes*. *Nodes* of a *Petri net* can be connected by *arcs*.

Note that it is legal to have an *arc* from a *place* to a *place* or from a *transition* to a *transition* according to the *PNML Core Model*. The reason is that there are versions of *Petri nets* that support such *arcs*. If a *Petri net type* does not support such *arcs*, this restriction will be defined in the particular package defining this type.

5.2.2 Pages and Reference Nodes

Three other kinds of *objects* are used for structuring a *Petri net*: *pages*, *reference places*, and *reference transitions*. As mentioned above, a *page* may contain other *objects*; since a *page* is an *object* itself, a *page* may even contain other *pages*, which defines a hierarchy of subpages.

This part of ISO/IEC 15909 requires that an *arc* must connect *nodes* on the same *page* only. The reason for this requirement is that *arcs* connecting *nodes* on different *pages* cannot be drawn graphically on a single *page*. In the *PNML Core Model* of Fig. 2, this requirement is captured by the OCL expression (OCL 2.0, OMG) next to the class for *arcs*.

In order to connect *nodes* on different *pages* by an *arc*, a representative of one of the two *nodes* is drawn on the same *page* as the other *node*. Then, this representative may be connected with the other *node* by an *arc*. This representative is called a *reference node*, because it has a reference to the *node* it represents. Note that a *reference place* must refer to a *place* or a *reference place*, and a *reference transition* must refer to a *reference transition* or a *transition*. Moreover, cyclic references among *reference nodes* are not allowed. Though this requirement cannot be expressed in UML or OCL notation, this requirement is mandatory.

The concepts of *pages* and *reference nodes* are not defined in part 1 of ISO/IEC 15909. The reason for introducing them in part 2 is that these concepts are needed for graphical and structuring purposes. Semantically, they do not carry any meaning. The meaning of these concepts is defined by merging each *reference node* with the *node* it ultimately refers to and by simply ignoring the *pages*. This procedure is called *flattening* of the page structure.

5.2.3 Labels

In order to assign further meaning to an *object*, each *object* may have *labels*. Typically, a *label* represents the name of a *node*, the initial marking of a *place*, the *transition condition*, or some *arc annotation*. In addition, the *Petri net* itself or its *pages* may have some *labels*, which are called *global labels*. For example, the package HLPNG defines *declarations* as *global labels* of a *High-level Petri Net*, which are used for defining *variables*, and user-defined *sorts* and *operators*.

This part of ISO/IEC 15909 distinguishes two kinds of *labels*: *annotations* and *attributes*. An *annotation* comprises information that is typically displayed as text next to the corresponding *object*. Examples of *annotations* are names, initial markings, arc annotations, transition conditions, and timing or stochastic information.

In contrast, an *attribute* is, typically, not displayed as text next to the corresponding *object*. Rather, an *attribute* has an effect on the shape or colour of the corresponding *object*. For example, an *attribute* such as arc type could have domain { `normal`, `read`[2], `inhibitor`, `reset` }. This part of ISO/IEC 15909, however, does not mandate the effect that an *attribute* has on the graphical appearance of a net *object*.

Note that the classes for *label*, *annotation* and *attribute* are abstract in the *PNML Core Model*, which means that the *PNML Core Model* does not define concrete *labels*, *annotations*, and *attributes*. The only concrete *label* defined in the *PNML Core Model* is the *name*, which is a *label* that can be used for any *object* within any *Petri net type*. Note that, this way, any *object* – including *nodes*, *pages*, even the *net* itself as well as *arcs* – can have a *name*. The value of a *name* is a *String*, which is imported from the separate package *XMLSchemaDataTypes* (see Clause 5.2.6 for more information). The other concrete *labels* will be defined in the packages for the concrete *Petri net types* (see Clause 5.3).

In order to support the exchange of information among tools that have different textual representation for the same concepts (i. e. if they have different concrete syntax), there are two ways for representing the information within an annotation: *textually* in some concrete syntax and *structurally* as an abstract syntax tree (see Clauses 5.3.11 and 7.1.2 for details).

Note that *reference nodes* may have *labels*, but these *labels* do not have any effect on the semantics of the net and can be completely ignored semantically. Since the labels of a *reference node* do not have an effect on the semantics, it is easy to obtain a semantically equivalent *Petri net* without *pages* by merging every *reference node* to the *node* it directly or indirectly refers to. This is called *flattening* of the *Petri net*. But, the labels of a *reference node* can have an effect on the graphical appearance or can give some additional information to the user.

5.2.4 Graphical Information

In addition to the *Petri net* concepts, information concerning the graphical appearance can be associated with each *object* and each *annotation*. For a *node*, this information includes its position; for an *arc*, it includes a list of positions that define intermediate points of the *arc*; for an *object's annotation*, it includes its relative position with respect to the corresponding *object*; and for an *annotation* of a *page*, the position is absolute. There can be further information concerning the size, colour and shape of nodes or arcs, or concerning the colour, font and font size of labels. Note that this information can be used for automatically transforming a *Petri Net* into *Scalable Vector Graphics* (SVG) by XSLT transformations (see [8] for more details). This transformation, however, is not defined in this part of ISO/IEC 15909.

Figure 3 shows the different graphical information that can be attached to the different types of *objects* and the different attributes. Note that this still belongs to the *PNML Core Model*; it is shown in a different figure only for better understandability. Table 1 gives an overview of the meaning and the domain of the attributes of the different graphical features. The exact XML syntax is defined in Clause 7.1.3.

A *position* element defines the absolute position for *nodes* and *pages*. For an *annotation* an *offset* element defines its relative position to the *object* it is attached to – if it is a *global annotation*, the *offset* also defines the absolute position on that *page*. Each absolute or relative position consists of a pair of Cartesian coordinates (x, y), where the units are points (pt). As for many graphical tools, the x-axis runs from left to right and the y-axis from top to bottom. And the reference point for the position of an *object* is its centre.

For an *arc*, the (possibly empty) sequence of *positions* defines its intermediate points (bend points). Note that the positions of the *start point* and the *end point* of an *arc* are not given explicitly for the arc. These positions are determined from the position of the source and the target *node* of the *arc* and the direction of the respective segment of the *arc*. They are defined by the line starting or ending at the reference point of the respective node in the direction of the arc and the intersection with the border of that node. Altogether, the *arc* is displayed as a path from the *start point* on the border of the *source node* to the *end point* on the border of the *target node* via the intermediate points. Depending on the value of the attribute *shape* of element *line*, the path is displayed as a broken *line* (polyline) or as a quadratic Bezier *curve*. In the case of a Bezier curve the intermediate positions alternately are the line connectors or Bezier control points. The reference point of an arc is the middle of the arc, which is the middle of the middle segment of the arc, if there are an odd number of segments; it is the middle point, if there are an even number of segments.

[2] Sometimes, a read arc is called a test arc.