

# 中华人民共和国国家标准

GB/T 15969.10—2023/IEC 61131-10:2019

## 可编程序控制器 第 10 部分:PLC 的 XML 开放交互格式

Programmable Controllers—Part 10: PLC open XML exchange format

(IEC 61131-10:2019, IDT)

2023-08-06 发布

2024-03-01 实施

国家市场监督管理总局  
国家标准化管理委员会 发布

目次

前言 ..... III

引言 ..... IV

1 范围 ..... 1

2 规范性引用文件 ..... 2

3 术语、定义和缩略语..... 3

4 模式概念概述 ..... 3

5 约定 ..... 7

6 主模式元素“Project” ..... 8

7 抽象复合类型..... 16

8 命名空间声明..... 25

9 用户定义数据类型声明..... 26

10 程序组织单元声明 ..... 29

11 变量声明 ..... 40

12 行为描述 ..... 44

13 图形行为描述 ..... 46

14 资源声明 ..... 66

15 其他 ..... 67

附录 A(规范性) XML 交互格式模式定义 ..... 69

附录 B(资料性) 推荐方案 ..... 122

附录 C(资料性) XML 文档示例 ..... 143

参考文献..... 205

## 前 言

本文件按照 GB/T 1.1—2020《标准化工作导则 第 1 部分:标准化文件的结构和起草规则》的规定起草。

本文件是 GB/T 15969《可编程序控制器》的第 10 部分。GB/T 15969 已发布了以下部分:

- 第 1 部分:通用信息;
- 第 2 部分:设备要求和测试;
- 第 3 部分:编程语言;
- 第 4 部分:用户导则;
- 第 5 部分:通信;
- 第 6 部分:功能安全;
- 第 7 部分:模糊控制编程;
- 第 8 部分:编程语言的应用和实现导则;
- 第 9 部分:用于小型传感器和执行器的单点数字通信接口(SDCI);
- 第 10 部分:PLC 的 XML 开放交互格式。

本文件等同采用 IEC 61131-10:2019《可编程序控制器 第 10 部分:PLC 的 XML 开放交互格式》。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中国机械工业联合会提出。

本文件由全国工业过程测量控制和自动化标准化技术委员会(SAC/TC 124)归口。

本文件起草单位:杭州电子科技大学、北京机械工业自动化研究所有限公司、江苏长江智能制造研究院有限责任公司、清华大学、北京和利时系统工程公司、北京航空航天大学、华中科技大学、浙江中控研究院有限公司、杭州和利时自动化有限公司、杭州义益钛迪信息技术有限公司、深圳市嘉合劲威电子科技有限公司、广东美的智能科技有限公司、深圳市朗宇芯科技有限公司、三门三友科技股份有限公司、深圳市世椿智能装备股份有限公司、重庆中科摇橹船信息科技有限公司、深圳市顾美科技有限公司。

本文件主要起草人:邬惠峰、王凯、孙丹枫、黄双喜、刘新、陶永、尹作重、孙洁香、薛靖婉、陈佰平、朱毅明、陈冰、方垒、何志伟、王天林、潘艳飞、王琳、胡俊杰、朱静强、王孙骏、段春艳、陈晖、杨大胜、王亮、乔森、吴俊义、时军伟、柳胜耀、齐文博、王侃、曾小虎。

## 引 言

可编程序控制器自 1969 年问世以来,已在工业自动化的各个领域广泛使用,并成为工业自动化系统的重要支柱。可编程序控制器技术的发展十分迅速。首先,已由单一机型发展为整套系列(微型、小型、中型、大型、特大型),且通信、联网、运算、自适应控制等功能大大增强;其次,应用范围亦由逻辑控制扩展到运动控制、过程控制、批量控制、配方控制等;再次,控制范围亦由单机扩展到整个车间以至全厂范围和无线电远程控制,从而覆盖了一部分由分布式控制系统(DCS)、数控系统(NC)、机器人控制系统的应用领域。总之,可编程序控制器已不断朝纵向和横向集成扩展。

GB/T 15969 也随着可编程序控制器技术的发展不断地修改和扩充,对 PLC 的软件、硬件、外围设备及其相关指南进行了系统的规范,拟由 10 部分构成。

- 第 1 部分:通用信息。目的在于建立可编程序控制器及其相关外围设备的相关定义,并确认与可编程序控制器及其外围设备的选择和应用相关的主要特性。
- 第 2 部分:设备要求和测试。目的在于对可编程序控制器(PLC)及其外围设备的设备要求和相关试验进行规定。
- 第 3 部分:编程语言。目的在于为可编程序控制器最常用的编程语言定义其语义与语法。
- 第 4 部分:用户导则。目的在于为 PLC 最终用户提供 GB/T 15969 的通用信息,并帮助他们按 GB/T 15969 系列标准选择 PLC 装置及技术规范。
- 第 5 部分:通信。目的在于定义可编程序控制器与其他电子系统间的通信。
- 第 6 部分:功能安全。对可编程序控制器及其相关外围设备的功能安全要求进行了规定,目的在于建立和描述功能安全可编程序控制器的安全生命周期要素,并与 IEC 61508-1~3 中确定的一般安全生命周期保持一致。
- 第 7 部分:模糊控制编程。对模糊控制的编程语言以及将模糊控制应用集成到 GB/T 15969.3 编程语言中的基本方法进行了规定,目的在于为实现不同编程系统之间交互移植模糊控制程序提供基础。
- 第 8 部分:编程语言的应用和实现导则。目的在于帮助用户使用 GB/T 15969.3 中所定义的编程语言进行软件开发。
- 第 9 部分:用于小型传感器和执行器的单点数字通信接口(SDCI)。对用于 PLC 主站和设备单点数字通信接口(SDCI)的通信服务和协议进行了规范,目的在于将 GB/T 15969.2 中的数字 IO 接口向 SDCI 扩展。
- 第 10 部分:PLC 的 XML 开放交互格式。目的在于通过定义 IEC 61131-3 的 XML 交互格式,实现 IEC 61131-3 程序甚至是整个 IEC 61131-3 工程在不同开发环境中的交换。

IEC 61131-3 定义了编程语言,但只是整体解决方案的一部分,其他部分还包括用于模拟、调试、版本控制、文档、网络等的工具。为了解决 IEC 61131-3 的程序或整个工程在不同软件平台之间的交互,本文件定义了一种基于 XML 的独立于解决方案的可扩展标记语言(extensive markup language, XML)交换格式。除了文本和程序逻辑信息,它还提供了图形表示信息的描述能力,例如功能块的位置和大小以及它们之间的连接方式。如果一个 IEC 61131-3 工程存储在这个标准的 XML 交换格式中,它可独立于一个特殊的开发环境而被重用。因此,它可被支持该标准 XML 交换格式的任何其他开发环境修改和维护。



可编程序控制器  
第 10 部分:PLC 的 XML 开放交互格式

1 范围

1.1 概要

本文件规定了 IEC 61131-3 工程导入导出时的 XML 交互格式,使得在符合 IEC 61131-3 的开发环境中开发的一个完整的 IEC 61131-3 工程能够在不同的开发环境之间进行转移,转移的内容可包括配置元素、数据类型及使用下列语言编写的程序组织单元(Program Organization Units, POUs):

- 文本语言,指令表(Instruction List, IL),
- 文本语言,结构化文本(Structured Text, ST),
- 图形语言,梯形图(Ladder Diagram, LD),
- 图形语言,功能块图(Function Block Diagram, FBD),
- 顺序功能图(Sequential Function Chart, SFC)。

交互格式由相应的 XML 模式(XML schema)进行详细说明。XML 模式是以.xsd 为扩展名的独立文件,并作为本文件的一部分。该模式的规范包含在附录 A 中。附录 B 给出了扩展的推荐方案。附录 C 给出了 XML 文档示例。假定本文件的读者熟悉 XML 技术。

图 1 提供了一个 XML 交互格式用法示例,不同的工具可生成或应用基于 XML 的 IEC 61131-3 信息。

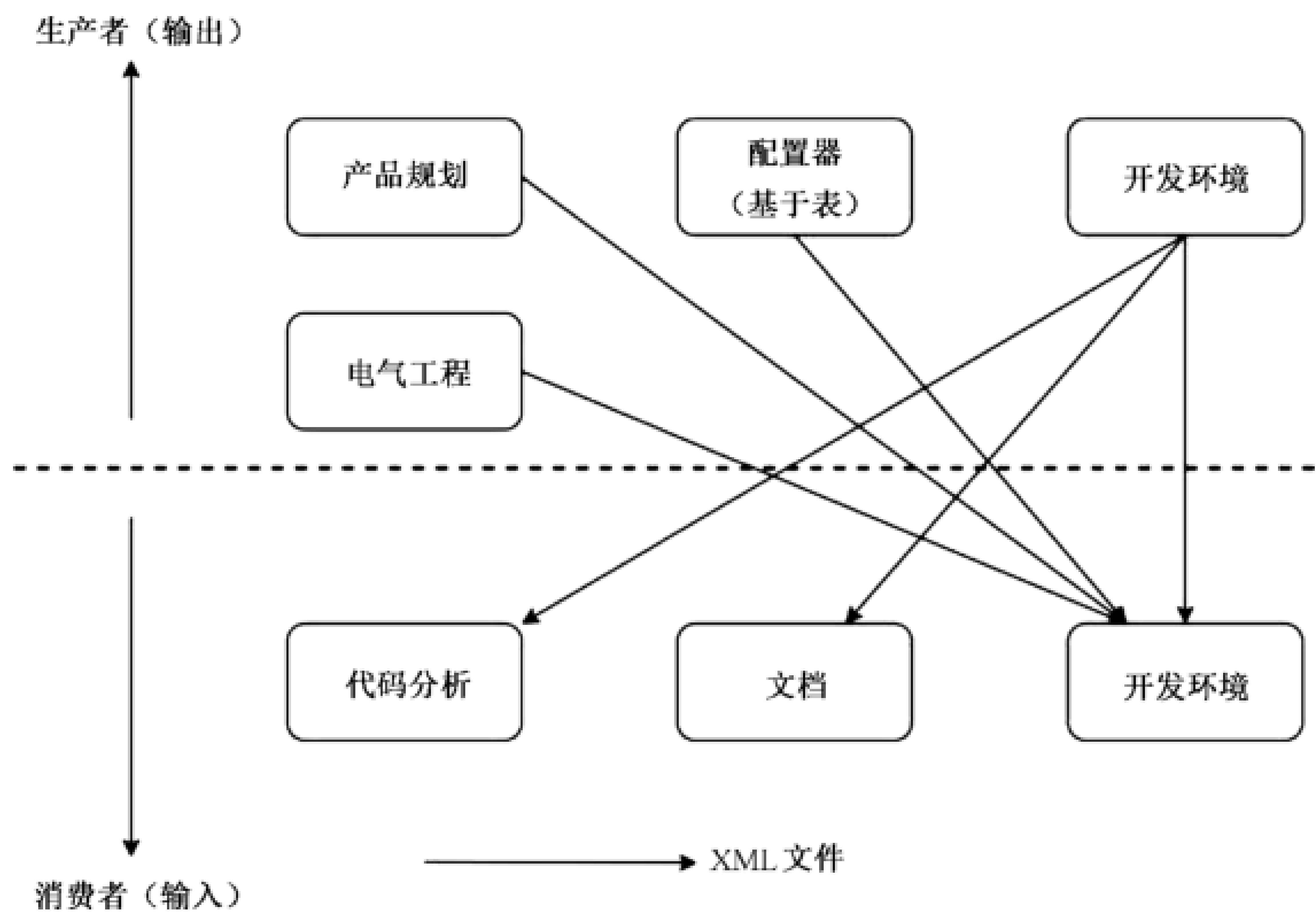


图 1 XML 交互格式用法概述(示例)

对 XML 交互格式的应用不仅可支持工程在开发环境之间简单地导出和导入,工程所有的相关信息都可被导出,例如可能需要包含图形控件的坐标信息。导入工具宜具有过滤功能,即能够选择部分信息导入至目标环境。供应商相关的信息和属性可能需要包含在导出文件中并在适用时被选择性地导

入,该信息不应影响到程序的逻辑部分。过滤工作宜在导入时进行,供应商应确保其 XML 模式文件的扩展方式,以保证在导入时忽略供应商信息不会影响到 IEC 61131-3 工程的功能。在本文件中定义的 XML 交互格式之外,供应商的特定属性和信息可通过供应商特定的 XML 模式文件进行添加。

本文件所描述的格式用于 IEC 61131-3 工程的导入和导出,此类 IEC 61131-3 工程可以是尚在开发中的不完整的工程。

关于不同编程系统之间图形语言结构的交换,其重点在于具有可选显式图形的逻辑信息的交换。

## 1.2 实现相关的参数

本文件并未提出满足兼容功能的方法和要求[例如被所有编程和调试工具(Programming and Debugging Tools, PADTs)支持的功能子集]。本文件支持 IEC 61131-3 中定义的所有特性的交互,此外,很多与实现相关的特性可通过 AddData 机制表达。

在一些用例中,程序不是从一个 PADT 转移到另一个 PADT,就是被创建用于不同的 PADT。在这两种情况下,这些 PADT 的功能集以及它们中与实现相关的参数设置都有可能不同。如果需要支持/考虑多个 PADT,则程序的功能应限制在所有 PADT 都支持的子集内。这些功能可通过 IEC 61131-3 中有关 PADT 的属性表来确定,例如:

- a) 支持的数据类型和标准功能,
- b) 抢占式或非抢占式调度,
- c) 包含或不包含终点扫描的 SFC 等。

对其他功能以及与实现相关的参数设置的确定更加复杂,例如:

- a) 每个 POU 的最大代码量或变量数,
- b) 标识符的最大长度(变量名长度),
- c) STRING 和 WSTRING 变量的默认长度或最大长度的大小,
- d) SFC 计算所有转换条件还是仅前驱步为活动状态的转换条件,
- e) TIME, DATE, TOD, DT 数据类型的范围和精度,
- f) (POU 在)PLC 的运行性能,
- g) 图形网络的执行顺序等。

对于具有多个 PADT 的用例,需考虑这些方面的差异。在某些情况下,宜仅使用所涉及的全部 PADT 都支持的功能,而在另一些情况下,可能需要在导入 PADT 后手动更改和测试程序。

本文件并未规定关于 PADT 的兼容功能的要求,而是为符合 IEC 61131-3 的程序定义了交互格式。

## 2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中,注日期的引用文件,仅该日期对应的版本适用于本文件;不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

IEC 61131-1 可编程序控制器 第 1 部分:通用信息(Programmable controllers—Part 1: General information)

注: GB/T 15969.1—2007 可编程序控制器 第 1 部分:通用信息(IEC 61131-1:2003, IDT)

IEC 61131-3 可编程序控制器 第 3 部分:编程语言(Programmable controllers—Part 3: Programming languages)

注: GB/T 15969.3—2017 可编程序控制器 第 3 部分:编程语言(IEC 61131-3:2013, IDT)

### 3 术语、定义和缩略语

#### 3.1 术语和定义

IEC 61131-3 界定的以及下列术语和定义适用于本文件。

##### 3.1.1

**IEC 61131-3 工程 IEC 61131-3 project**

符合 IEC 61131-3 的信息集合,包括 POU 类型和实例、数据类型、全局变量实例、配置和资源。

##### 3.1.2

**抽象复合类型 abstract complex type**

XML 模式定义中“abstract”属性为 true 的“complexType”元素。

##### 3.1.3

**具体复合类型 concrete complex type**

XML 模式定义中的“complexType”元素,该元素通常扩展了抽象复合类型而其自身不是抽象的。

##### 3.1.4

**简单类型 simple type**

XML 模式定义中的“simpleType”元素。

#### 3.2 缩略语

下列缩略语适用于本文件。

FBD:功能块图(Function Block Diagram)

IL:指令表(Instruction List)

LD:梯形图(Ladder Diagram)

PLC:可编程逻辑控制器(Programmable Logic Controller)

POU:程序组织单元(Program Organization Unit)

PADT:编程和调试工具(Programming and Debugging Tool)

SFC:顺序功能图(Sequential Function Chart)

ST:结构化文本(Structured Text)

URI:统一资源标识符(Uniform Resource Identifier)

URL:统一资源定位器(Uniform Resource Locator)

XML:扩展标记语言(Extensible Markup Language)

### 4 模式概念概述

#### 4.1 模式版本

模式文件中包含一个版本属性,其值的第一位数字与本文件的版本号相同,第二位数字保留为主版本微小改动后的子版本号。模式文件的文件名中也反映了版本号。

此外,“schemaVersion”属性指明了 XML 文件创建时所使用的模式文件的版本号。

#### 4.2 命名规范

在所有已定义的模式中(包括推荐模式),使用以下命名约定:

- a) 属性名应以小写字母开头；
- b) 类型名及元素名应以大写字母开头；
- c) 名称包含多个单词时,除了第一个单词外,其他的单词首字母应大写,不使用下划线。

4.3 图形化语言坐标系

本文件支持 IEC 61131-3 工程的交换,包括以一种图形化语言(即 FBD、LD 和 SFC 元素的图形化表达)编程的 POU。本文件定义了一个虚拟坐标系,以对图形元素(例如功能块、触点、线圈和步)的排列布局进行描述,这使得导入系统可(但不需要)将导入的代码尽可能地按照导出系统的呈现形式进行布局。因此,图形化语言的每个元素可将其相对位置和大小存储为 XML 属性。另外,子元素(例如功能块调用的形式参数)可存储相对于其父元素的相对位置。此外,两个元素之间的连接可存储为一组相对位置以描述路径布局。连接描述请参见 13.6。

相对位置和大小是可选属性且只能用于图形描述,相对位置不表示执行顺序。

虚拟坐标系有 X 和 Y 两个轴,坐标系原点(0,0)应在图形主体的左上角。X 轴的正向向右,Y 轴的正向向下。图形元素的相对位置标识了此元素左上角的位置。因此,子元素与其父元素之间的相对位置可能具有正值或负值。

为了帮助导入系统缩放图形,导出系统可在“ContentHeader”的“CoordinateInfo”中添加虚拟坐标系的使用信息,如图 2 所示。

- 对 FBD——X 和 Y 坐标均采用两个引脚之间的最小距离；
- 对 LD——采用“线圈”大小(X 为宽度,Y 为高度)；
- 对 SFC——采用“转换”的大小(X 为宽度,Y 为高度)。

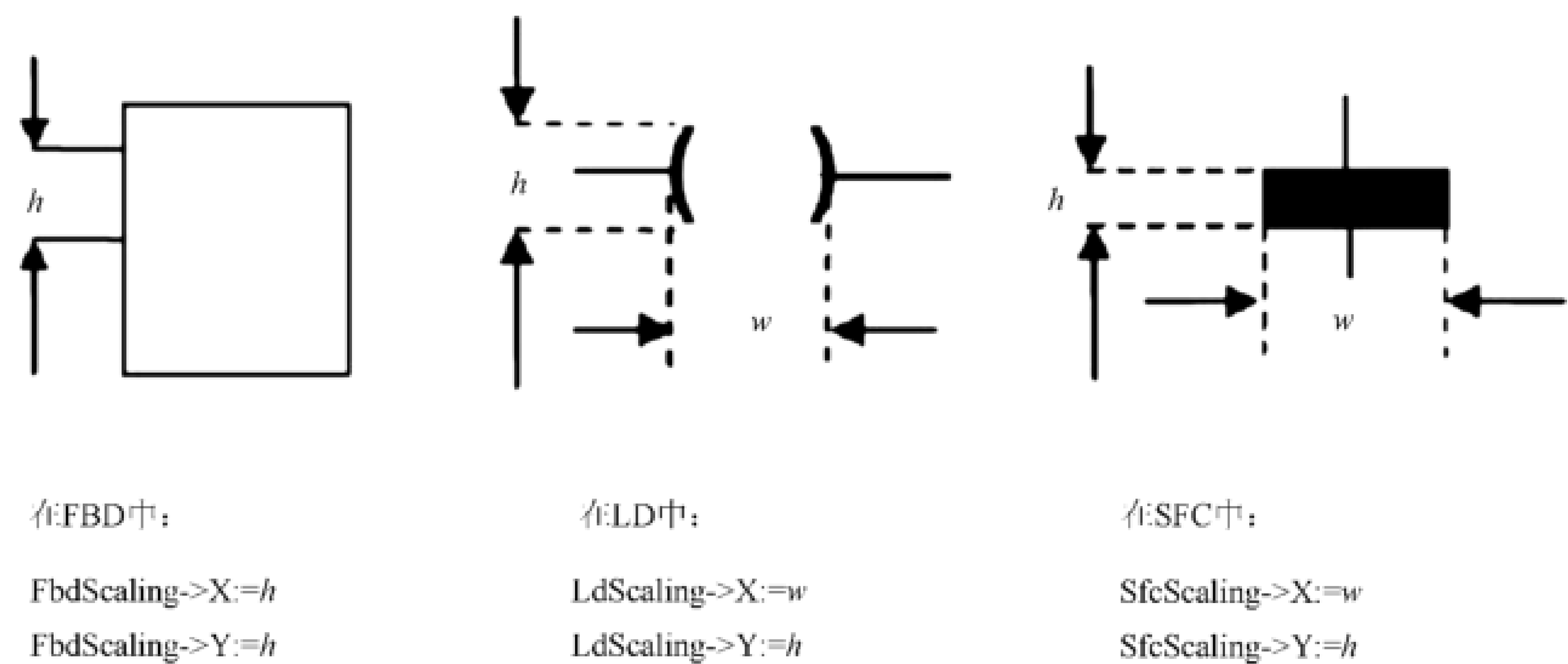


图 2 坐标信息到坐标系的映射

通过这些给定的信息,可实现坐标变换。例如,在 FbdScaling=10 的系统 A 中导出一个相对位置为 P(30,20)的功能块调用,之后在 FbdScaling=5 的系统 B 中导入时,P 会被转换到相对位置(15,10),如图 3 所示。

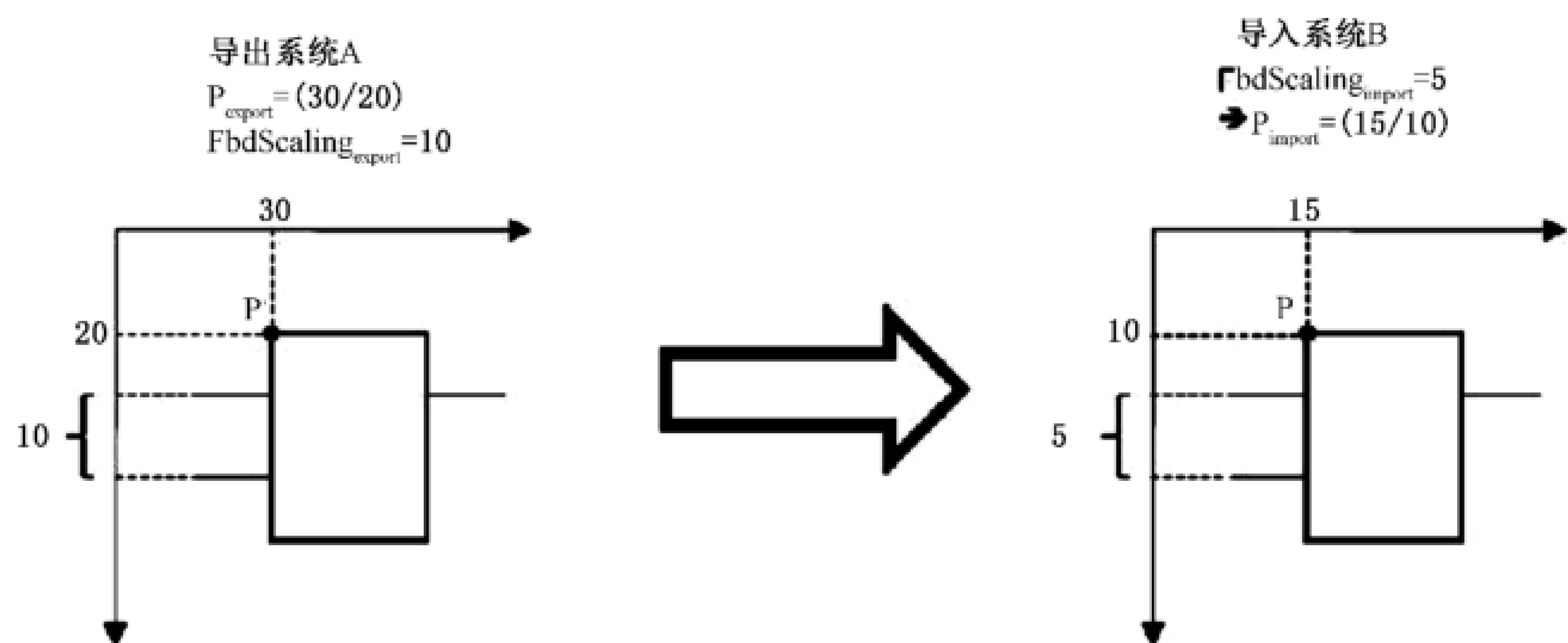


图 3 使用比例信息对坐标进行转换

图形对象的“RelPosition”元素表示对象的锚点位置。“RelPosition”为“XyDecimalValue”类型，具有属性“x”和“y”。对象的锚点应为对象矩形块的左上角，对象矩形块仅包含对象的主体，标号（实例名、线圈名）之类的附加元素或取反符号不应成为矩形块大小的考虑因素。对象矩形块的大小应由图形对象的“Size”元素指定（同为“XyDecimalValue”类型），其中宽度对应于“x”，高度对应于“y”。图形元素的示例如图 4 所示。

“RelPosition”始终指定相对于上级图形对象的相对位置，该图形对象的类型派生于“GraphicalObjectBase”或“BehaviourRepresentationBase”。

- 在 FBD 中，图形对象的“RelPosition”和上级元素“FbdNetwork”相关。
- 在 LD 中，图形对象的“RelPosition”和上级元素“Rung”相关。
- “FbdNetwork”和“Rung”的“RelPosition”分别和它们的上级元素“FBD”和“LD”相关。换句话说，这是绝对位置。
- 在 SFC 中，图形对象的“RelPosition”与上级元素“SFC”相关。换句话说，这是绝对位置。
- “ConnectionPointIn”和“ConnectionPointOut”的“RelPosition”与上级图形元素相关。
- “Connection”的“RelPosition”和上级元素“ConnectionPointIn”相关。

对象	例 1	例 2
步		
转换		
选择分支/合并		

图 4 对象锚点和对象矩形块示例

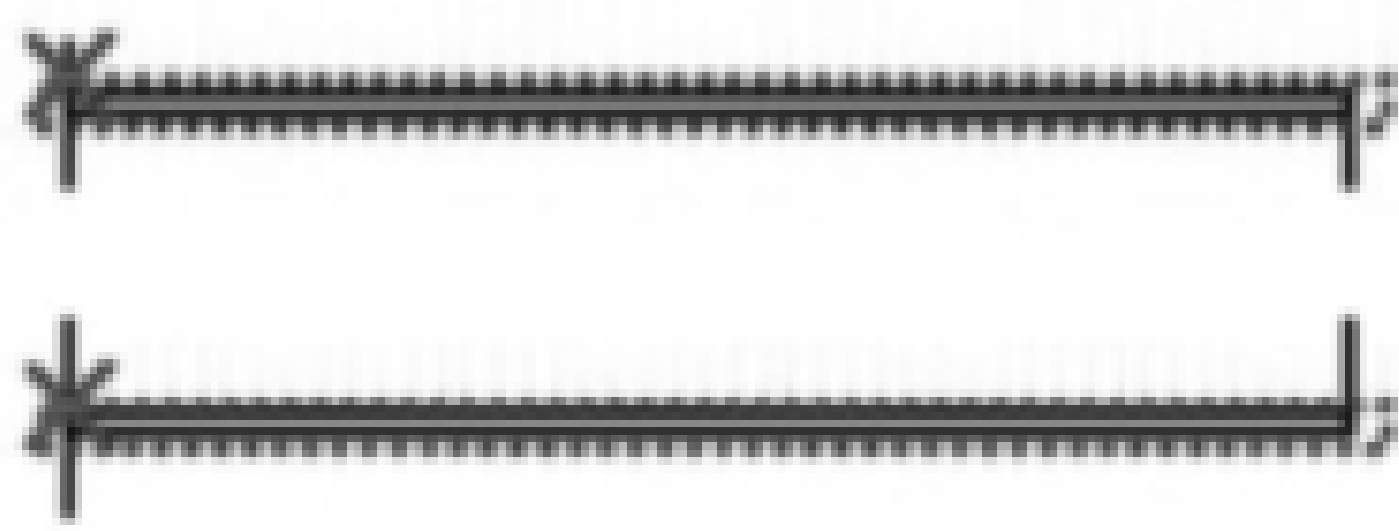
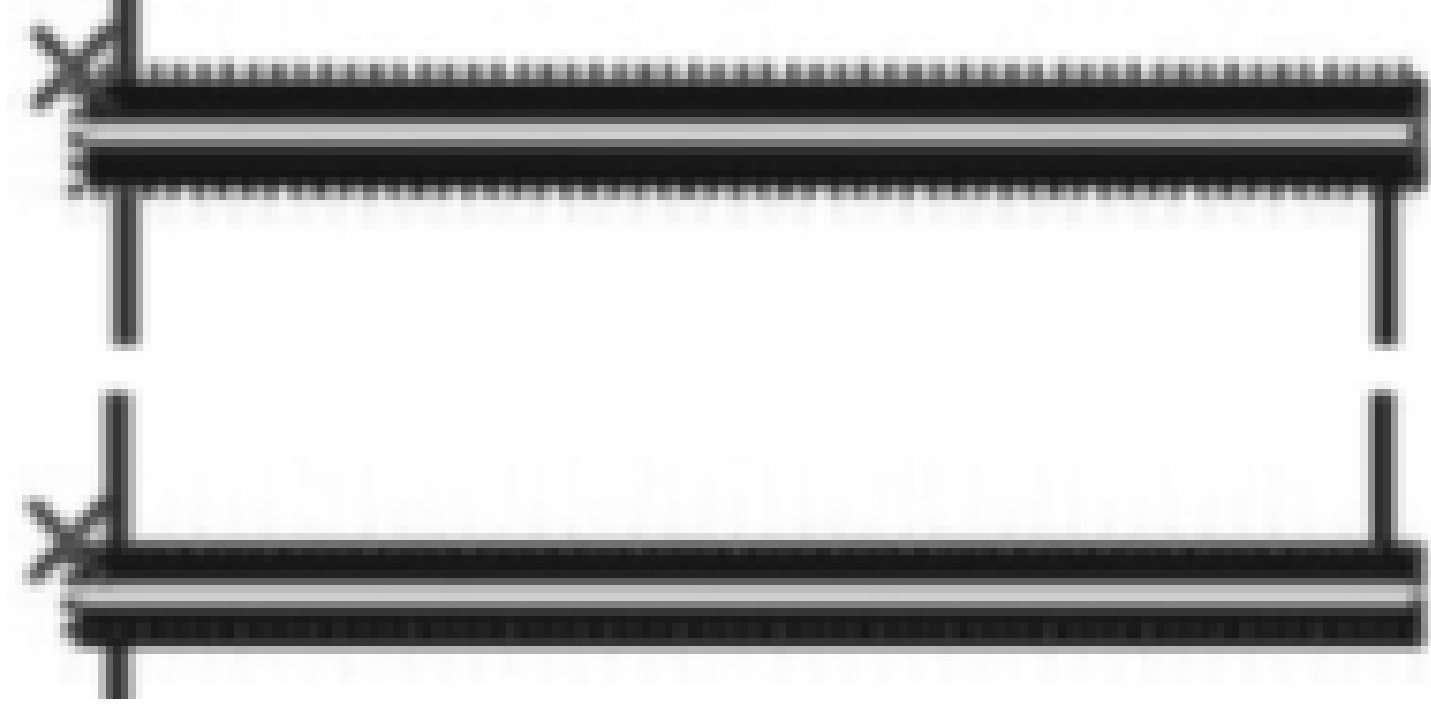



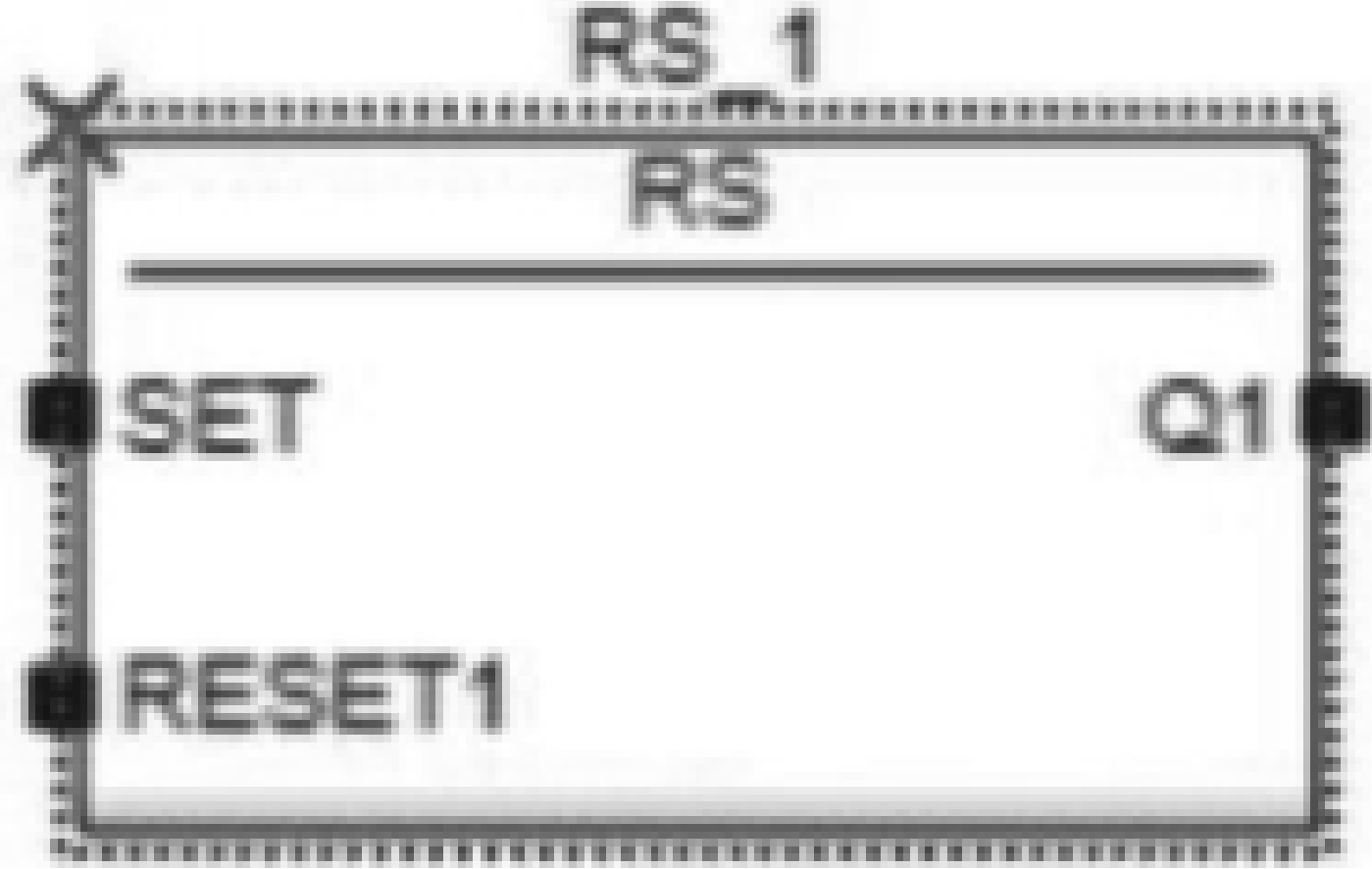

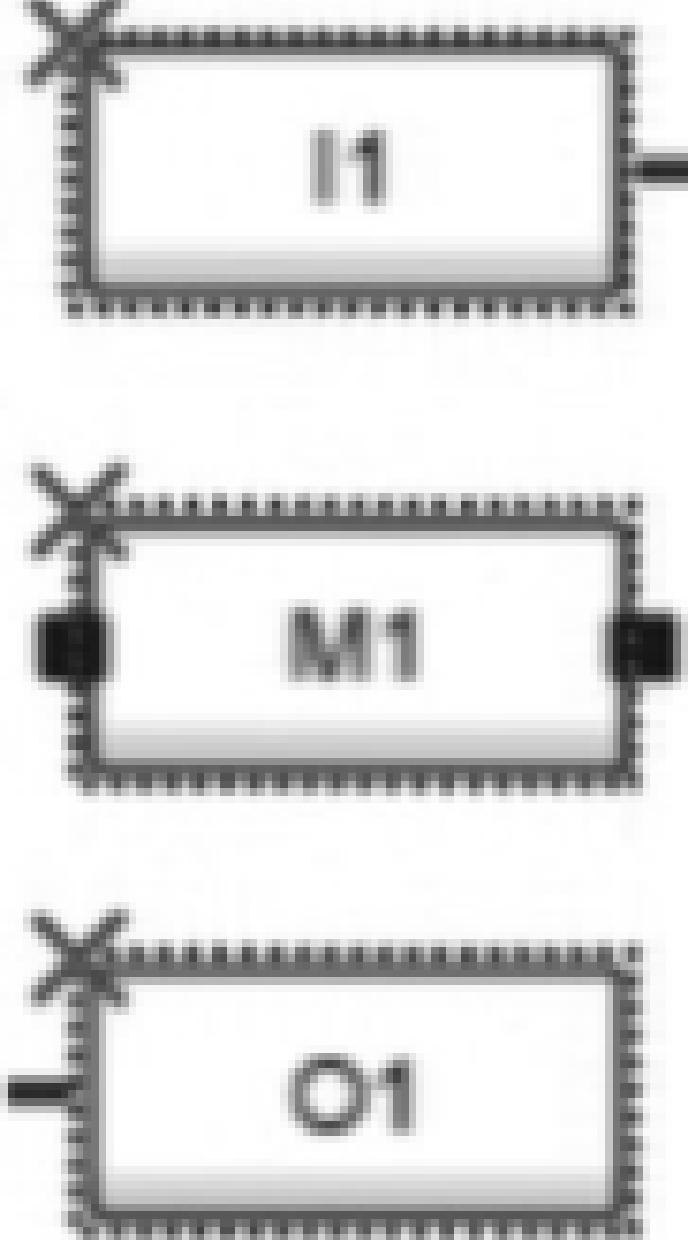
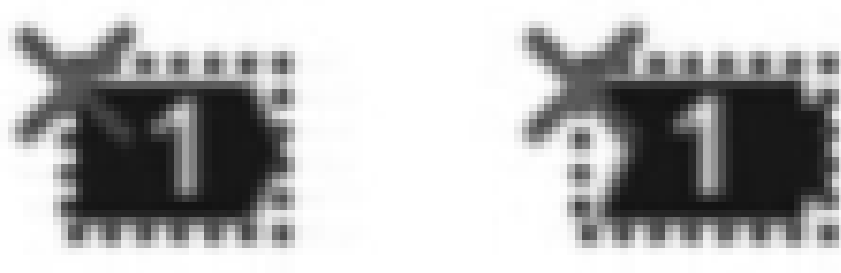

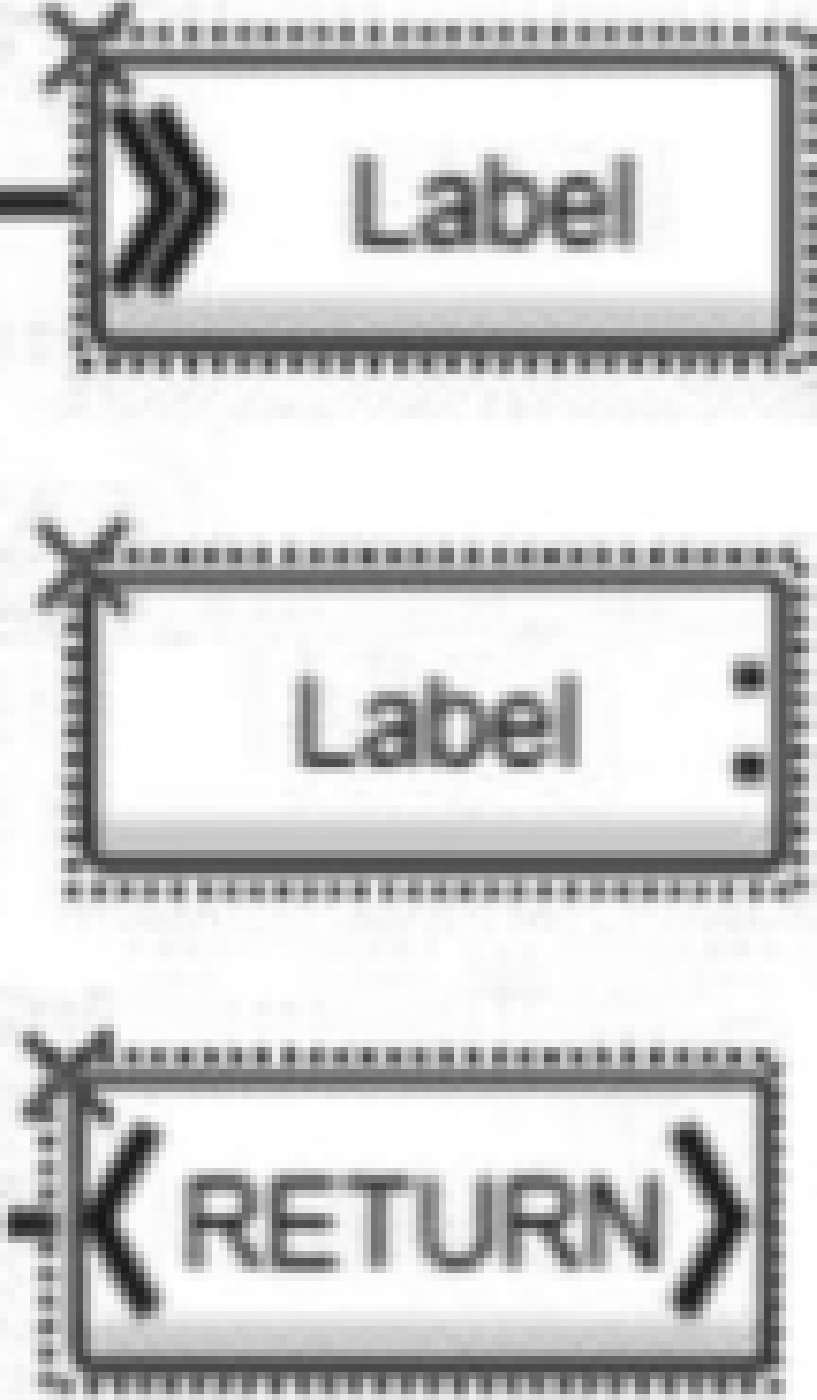
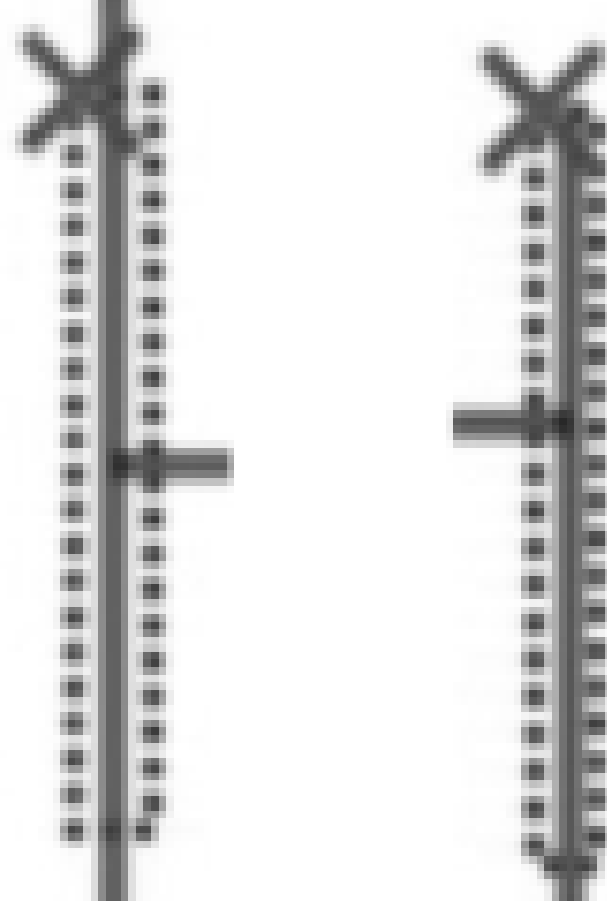

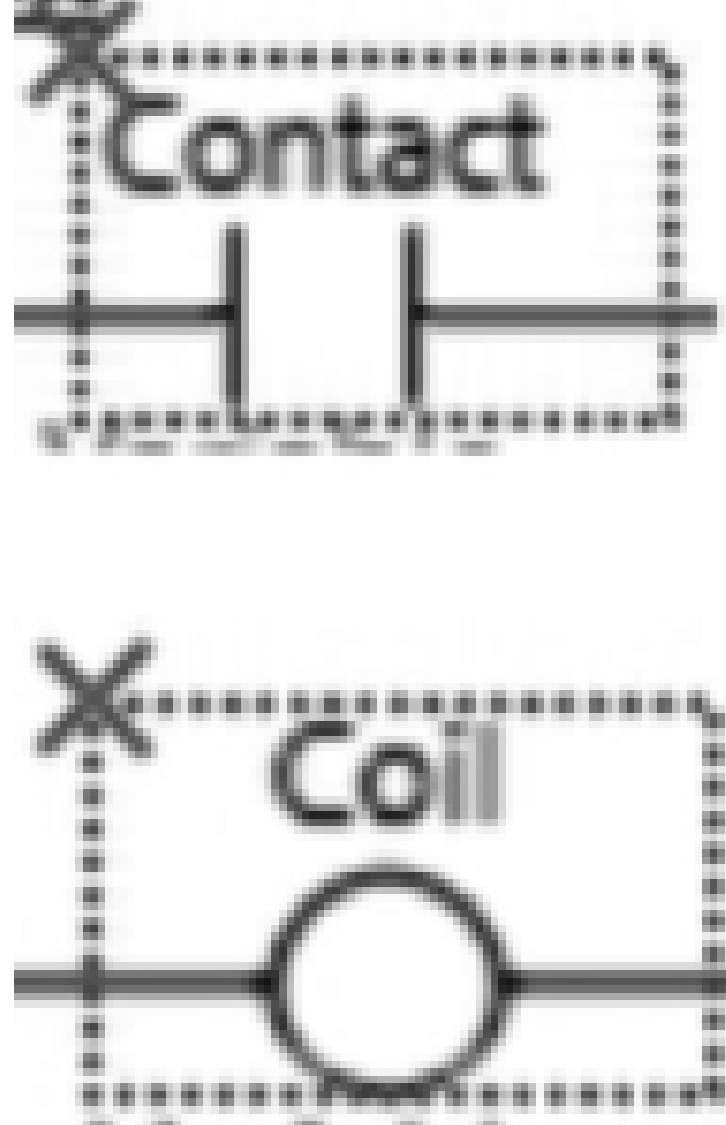
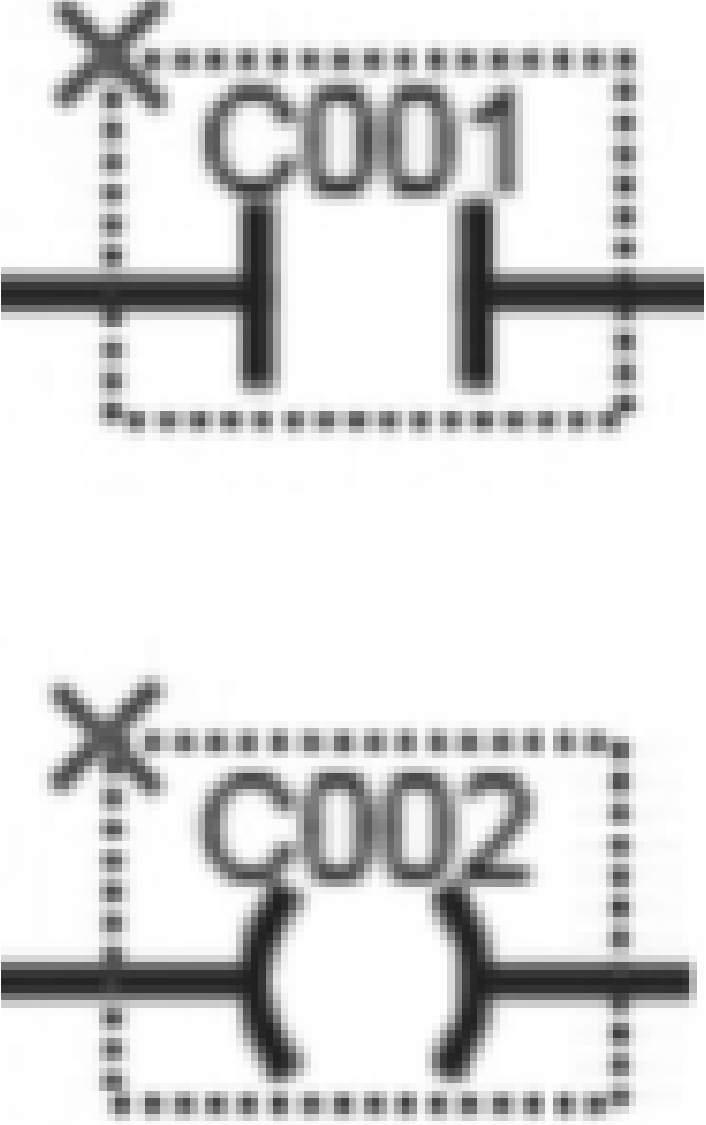
对象	例 1	例 2
同步分支/合并		
执行块		
块		
输入/输入输出/输出变量		
连接源端、连接末端		
跳转、标签、返回		
左/右电源母线		
触点、线圈		

图 4 对象锚点和对象矩形块示例（续）

## 4.4 模式扩展概念

模式文件是根据 IEC 61131-3 中定义的语言元素和功能创建的。但是,使用模式的 PADT 可支持额外的功能或需要在导出的文件中存储附加信息。该模式支持两种扩展机制,允许对交互格式中的信息进行重组,这超出了 IEC 61131-3 的范围。

第一种扩展机制是通过“AddData”元素,该元素可被嵌套在其他元素中。它的作用是扩展其他元素以增加其信息,但这对于元素在工具之间的交换不是关键的。B.1 列出了使用“AddData”来实现的常用扩展的建议,在设计新扩展时可将其作为指南。

应用“AddData”机制的 PADT 还需提供定义相应扩展的模式。所有附加的模式应在 6.3 中所描述的“AddDataInfo”元素中列出。有关“AddData”内容的详细信息见 15.2。

第二种扩展机制是抽象复合类型。模式中使用的所有抽象复合类型将在第 7 章中进行介绍。抽象复合类型允许对模式进行复杂扩展,可以是独立的而不依赖于已有元素。此外,IEC 61131-3 并未将 PLC 的编程语言限定于 IL、ST、FBD、LD 和 SFC 的范围。在此模式中,可通过为元素指定抽象复合类型来扩展以下元素:

- 用户定义数据类型(通过扩展“TypeSpecBase”);
- 主体、活动等的行为描述(通过扩展“BehaviourRepresentationBase”);
- 图形化对象类型(通过扩展“CommonObjectBase”“FbdObjectBase”“LdObjectBase”或“SfcObjectBase”);
- SFC 中转换条件的图形化描述(通过扩展“NetworkBase”);
- 任务类型(通过扩展“TaskBase”);
- 文本(通过扩展“TextBase”)。

新的元素类型可从抽象复合类型派生,并将其用作指定抽象复合类型的元素类型的替换,此类扩展的示例包括供应商特定的编程语言或任务类型。与“AddData”一样,此扩展的模式也提供给需要理解该扩展的其他工具。B.2 列出了使用抽象复合类型来实现的常用扩展的建议,在设计新扩展时可将其作为指南。

## 5 约定

### 5.1 通用要求

IEC 61131-1 中定义的 PADT 在完全或部分遵守本文件时应:

- a) 提供一个功能特性的子集,并提供如下定义的相应供应商合规声明;
- b) 不需要为了实现任何功能而包含替代或附加模式元素;
- c) 提供说明供应商特定依赖项的文档,包括本文件中明确指定的实现依赖项,以及本文件中未明确定义的限制参数,例如最大长度、数量、大小和取值范围;
- d) 提供说明在实现过程中所有可检测和报告的错误的文档;
- e) 不使用本文件中定义的模式元素名称来实现定义与本文件中功能不同的功能特性。

### 5.2 功能表

本文件复用了 IEC 61131-3 中定义的功能表,IEC 61131-3 中所有的功能表都以通用方式用于特殊目的。第一列包含“功能编号”,第二列包含“功能描述”,随后几列可能包含示例或更多信息。该表结构用于供应商的合规声明中。

为了达到本文件的目的,供应商有两种方法来重用 IEC 61131-3 特性表:

- a) 在 PADT 功能表添加 2 列(XML 导出,XML 导入);
- b) 为导出/导入功能创建单独的功能表列表。

5.3 供应商合规声明

供应商可定义 IEC 61131-3 功能表中所列出功能的任意一致性子集,并应在“供应商合规声明”中对其进行声明。

供应商合规声明应包含在系统随附的文档中,或由系统本身产生。

供应商合规声明的格式应提供以下信息:

- 常规信息,包括供应商名称和地址,产品名称、版本及发行日期;
- 对于已实现的每个功能提供相应功能表的编号、功能编号和适用的编程语言。

可选内容:特性表的标题、副标题、特性描述、示例和供应商说明等。

未实现的表和功能可省略。

IEC 61131-3 提供了供应商合规声明的示例,可对其进行修改以用于本文件。

6 主模式元素“Project”

6.1 概述

主模式元素“Project”表示 IEC 61131-3 工程。其内容如图 5 所示。

属性“schemaVersion”表示 XML 文档创建时所采用的模式版本。

“FileHeader”“ContentHeader”“Types”和“Instances”将分别在 6.2、6.3、6.4 和 6.5 中介绍。

“Documentation”所属类型“TextBase”将在 15.3 中介绍。

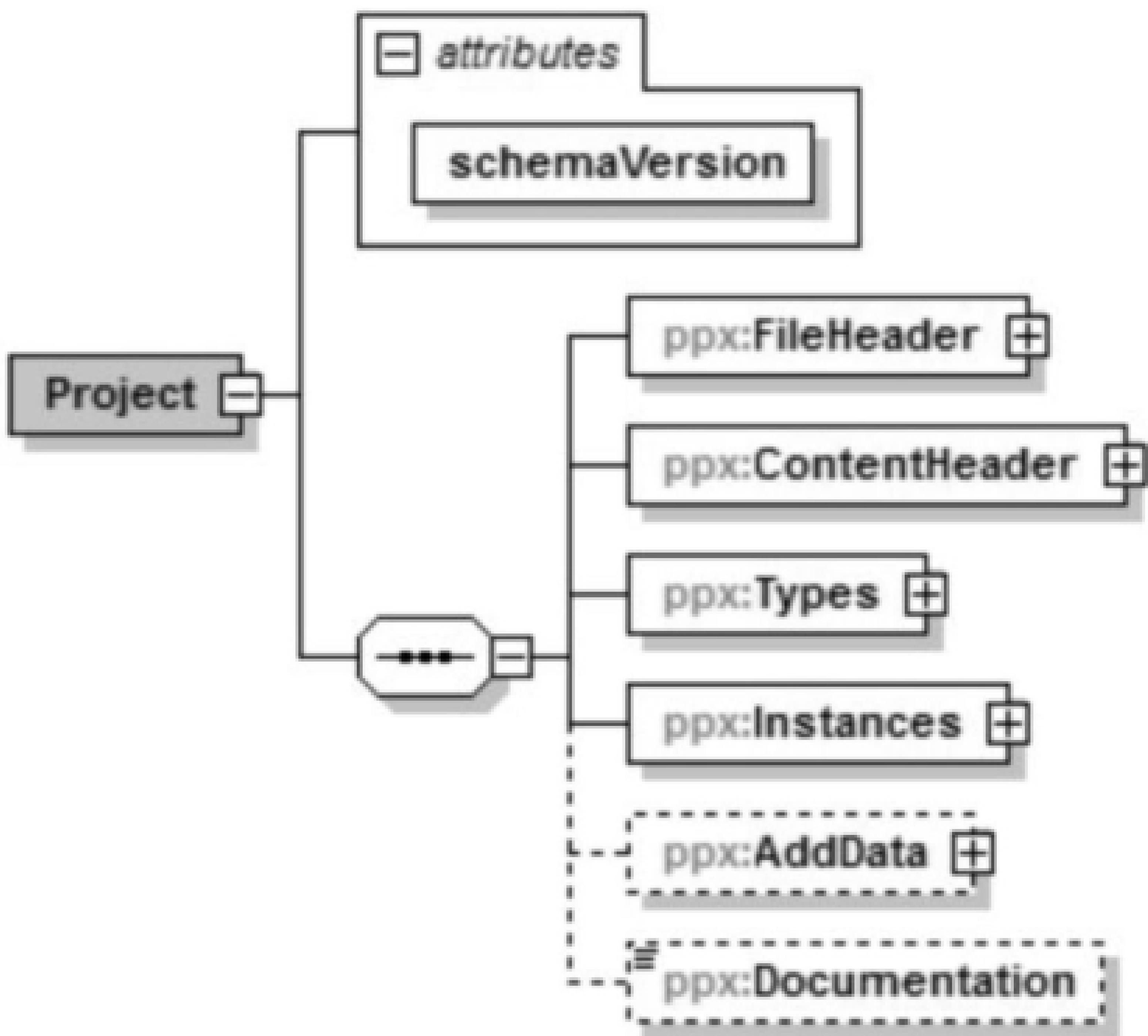


图 5 主模式元素“Project”

注: 本文件中的图片可包含由实线或虚线构成的矩形。实线表示必需的信息,虚线表示可选的信息。以图 5 为例,ppx:FileHeader(实线矩形)是必需信息,而 ppx:AddData(虚线矩形)是可选信息。

这些图是使用 XML 工程开发环境构建的,参见参考文献中列出的条目。



6.2 文件标头“FileHeader”

“FileHeader”表示有关 XML 文件导出工具的信息。其内容如图 6 所示。

属性“companyName”和“companyURL”描述了导出工具的供应商信息,“productName”“productVersion”和“productRelease”描述了工具本身。

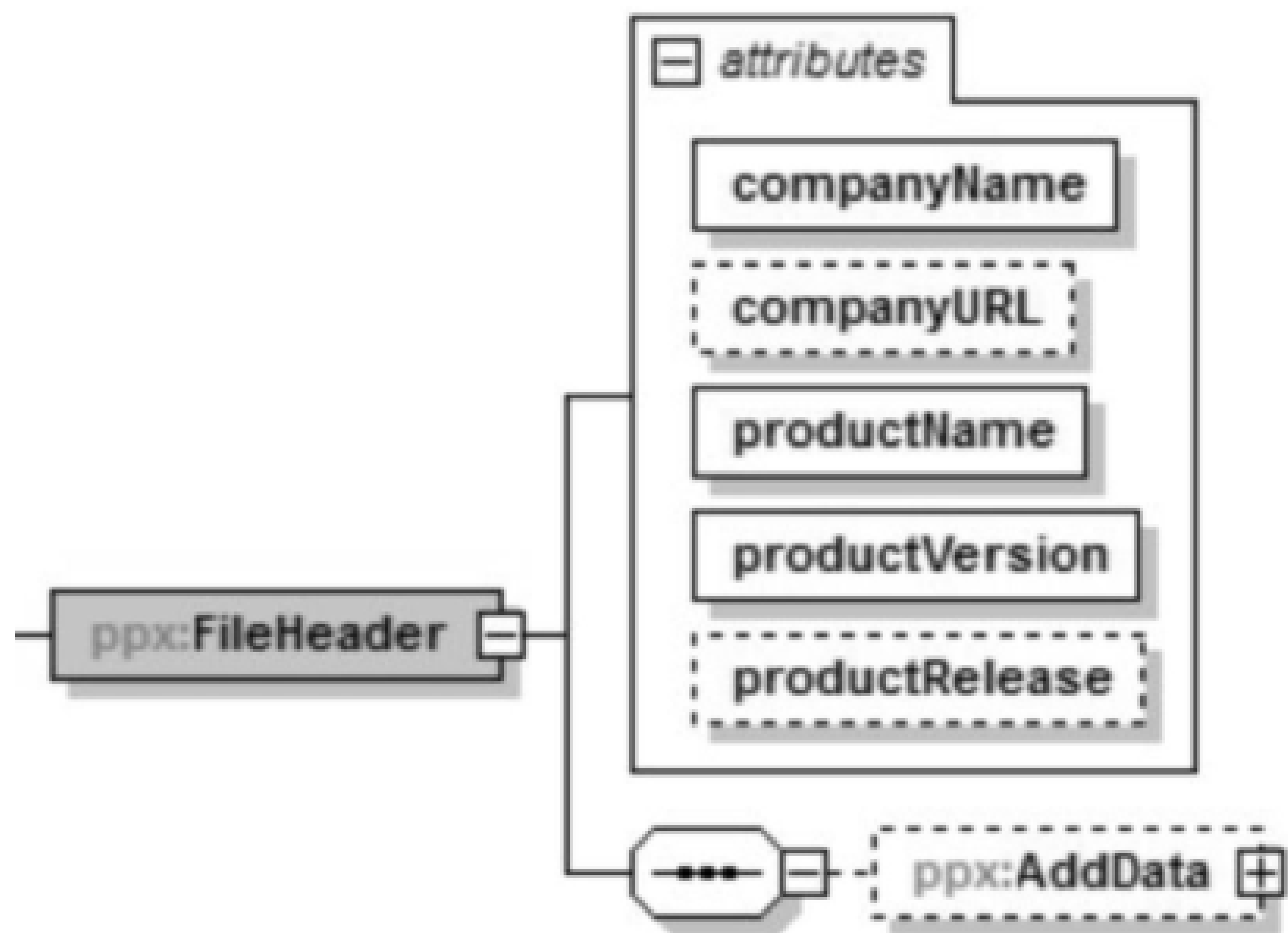


图 6 元素“FileHeader”

6.3 内容标头“ContentHeader”

“ContentHeader”表示 XML 文件内容的概述信息。其内容如图 7 所示。

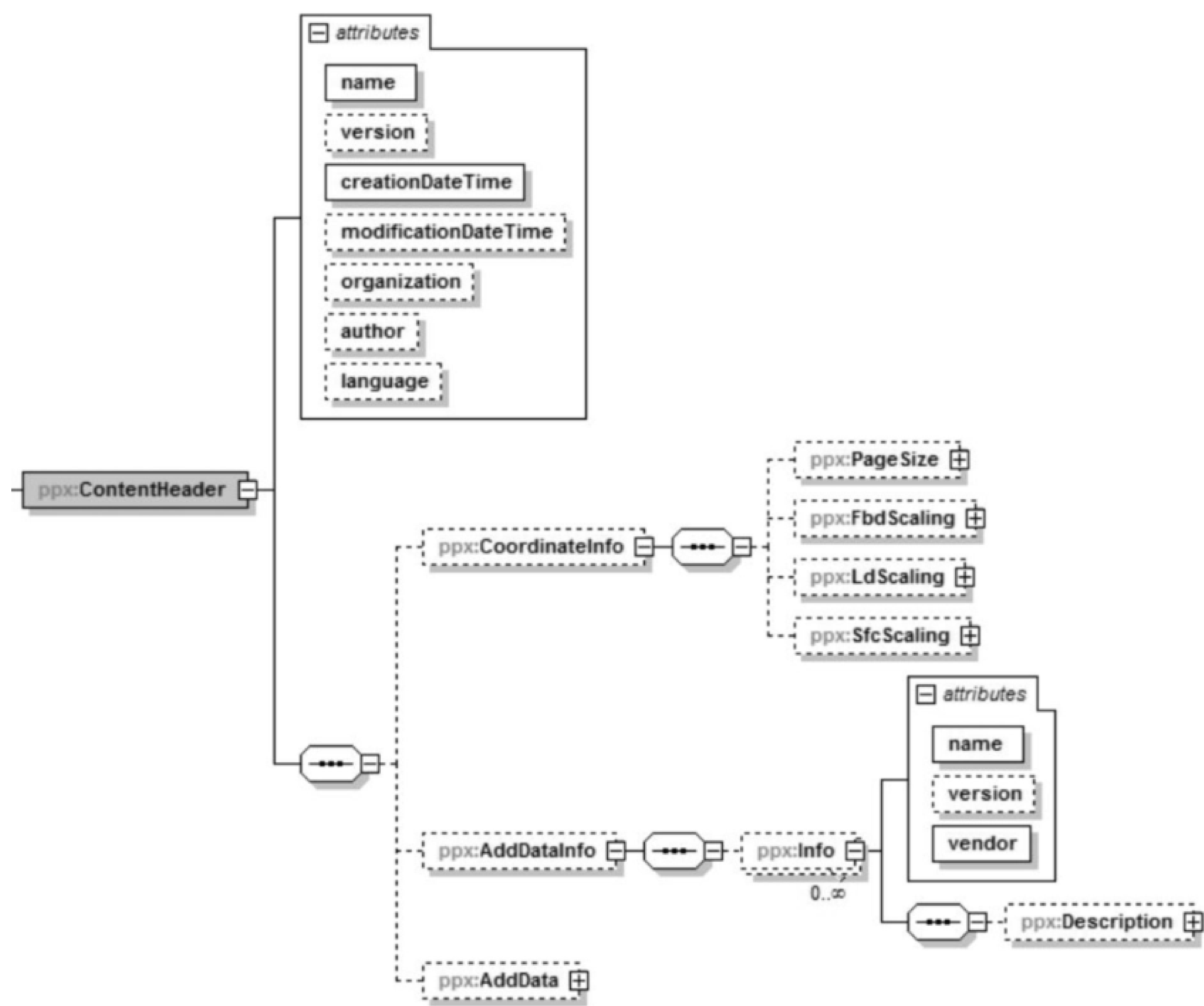


图 7 元素“ContentHeader”

该元素提供了“name”(在导出时设置为工程名称)“version”“creationDateTime”“modificationDa-teTime”“organization”“author”和“language”(工程定义时使用的语言)属性。

“FbdScaling”“LdScaling”和“SfcScaling”所属类型“XyDecimalValue”将在 15.1 中介绍。缩放 (scaling)概念已在 4.3 中介绍。

“AddDataInfo”表示相应“AddData”内容的唯一标识。属性“name”应包含供应商域名(URI)以确 保其唯一性,并应在生成的 XML 文档中进一步使用以引用相应的 AddData 模式。因此,属性“name” 在生成的 XML 文档中应是唯一的。属性“vendor”应包含域 URL。通过使用属性“name”,导入工具可 处理“AddData”。供应商应明确导入工具的行为,防止出现名称不能被导入工具识别的情况,特别是考 虑到稍后需要从该工具导出时。属性“version”表示模式版本。属性“Description”是依据 [http:// www.w3.org/1999/xhtml](http://www.w3.org/1999/xhtml) 规范的格式化文本,其中包含文件中使用的每个模式的简要说明。

6.4 类型“Types”

“Types”表示 IEC 61131-3 中定义的所有类型相关的元素。其内容如图 8 所示。

“GlobalNamespace”基于抽象复合类型“TextualObjectBase”,该类型将在 7.5.2 中介绍。

“NameSpaceDecl”的所属类型“NameSpaceDecl”将在第 8 章中介绍。“DataTypeDecl”所属类型

“UserDefinedTypeDecl”将在 9.1 中介绍。“PouDecl”组的内容将在 10.1 中介绍。

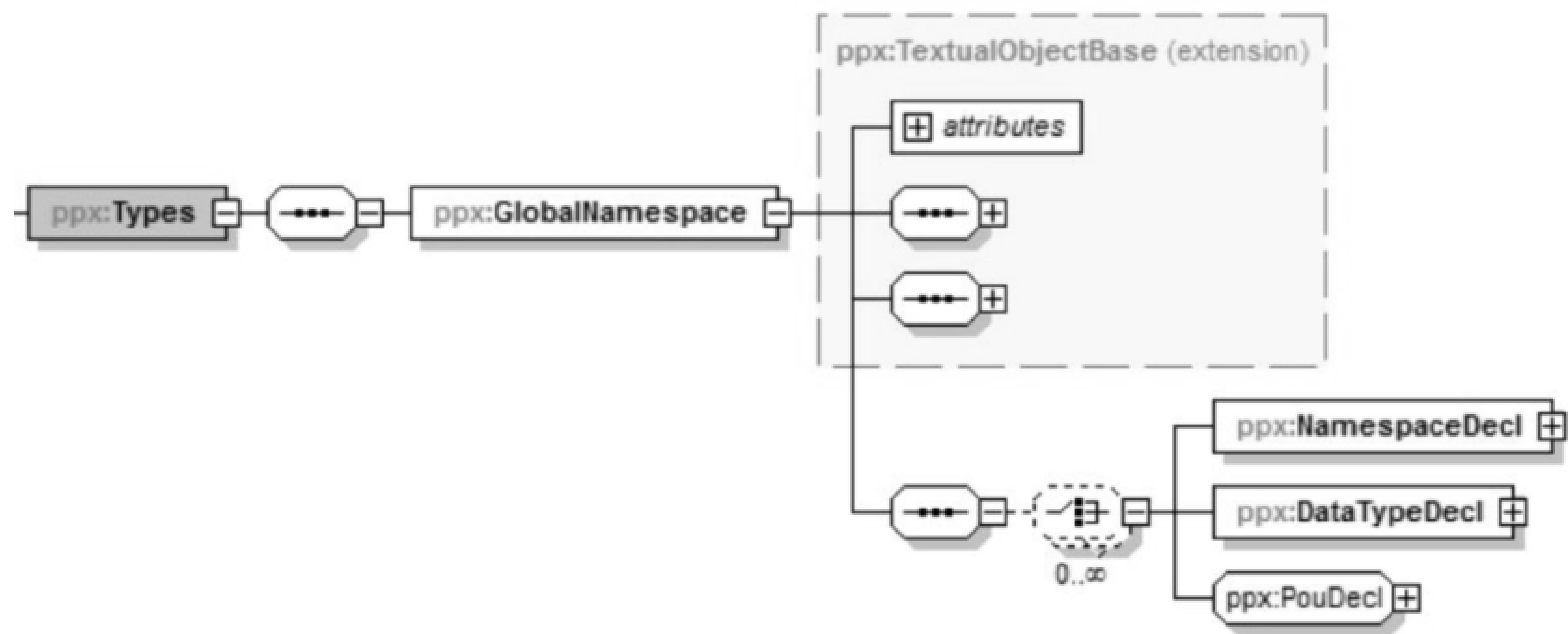


图 8 元素“Types”

6.5 实例“Instances”

6.5.1 配置“Configuration”

“Instances”表示 IEC 61131-3 中定义的所有实例相关的元素。其内容如图 9 所示。

“Configuration”基于抽象复合类型“TextualObjectBase”，该类型将在 7.5.2 中介绍。

“Resource”“GlobalVars”(IEC 61131-3: global variables) 所属类型“VarList”“AccessVars”和“ConfigVars”将分别在 6.5.2、11.1、6.5.3 和 6.5.4 中介绍。

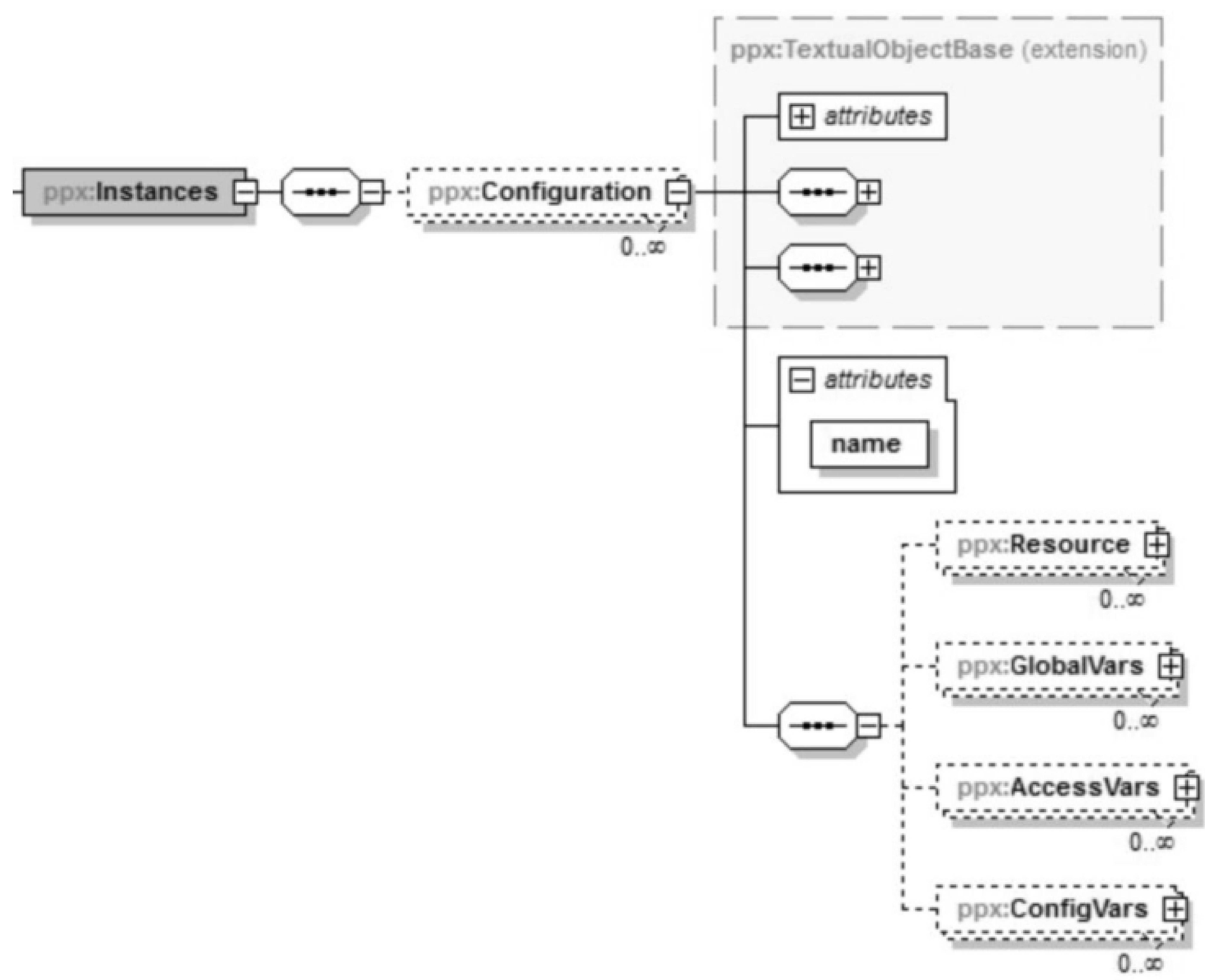


图 9 元素“Instances”

6.5.2 资源“Resource”

“Resource”表示 IEC 61131-3 中定义的资源。其内容如图 10 所示。

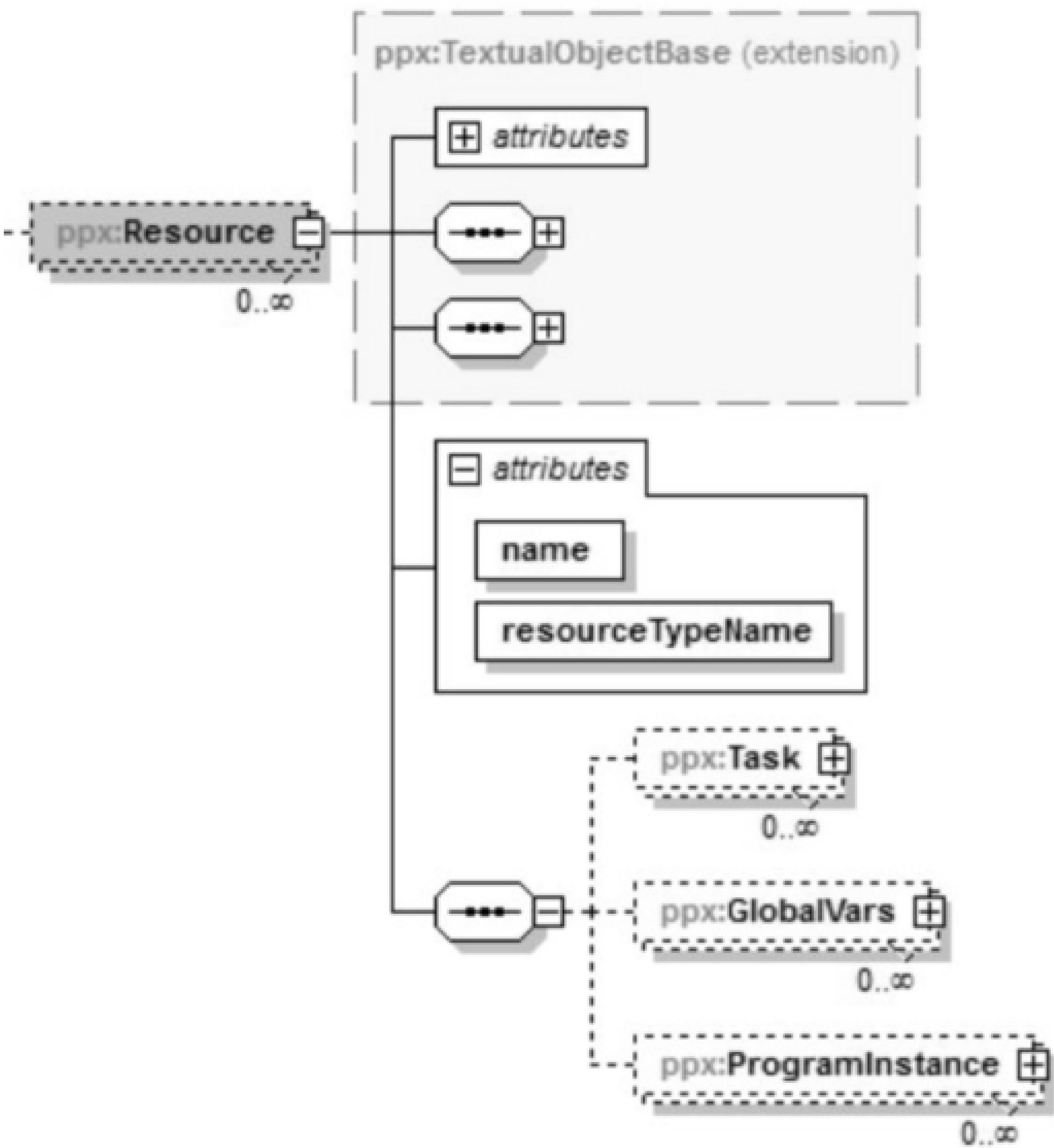


图 10 元素“Resource”

“Resource”基于抽象复合类型“TextualObjectBase”，该类型将在 7.5.2 中介绍。“Resource”通过属性“name”进行标识。此外，“Resource”还提供了“resourceTypeName”属性。

“Resource”包含“TaskBase”类型的“Task”、“VarList”类型的“GlobalVars”（IEC 61131-3：全局变量）和“TextualObjectBase”类型的“ProgramInstance”，“TaskBase”和“VarList”类型将分别在 7.5.4 和 11.1 中介绍。

“ProgramInstance”表示 IEC 61131-3 中定义的程序实例。其内容如图 11 所示。

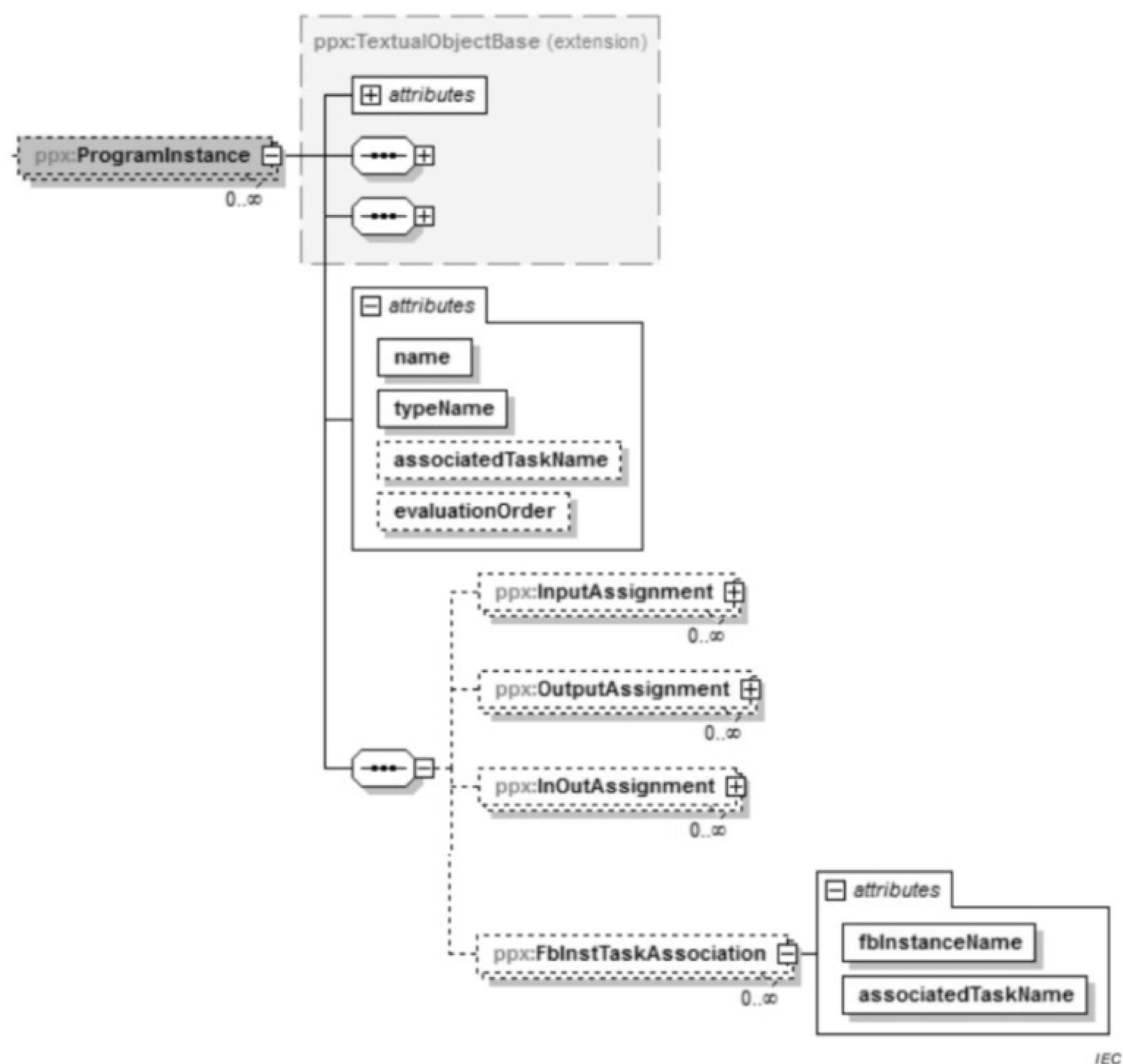


图 11 元素“ProgramInstance”

“ProgramInstance”基于抽象复合类型“TextualObjectBase”，该类型将在 7.5.2 中介绍。“ProgramInstance”通过属性“name”进行标识。程序实例的名称在每个资源中应是唯一的。

属性“typeName”表示该实例的程序类型名。

属性“associatedTaskName”表示与该程序实例相关联的任务的名称。在省略此属性的情况下，应关联 IEC 61131-3 中定义的系统默认任务（具有最低优先级的循环任务）。在此优先级内，将根据“evaluationOrder”对其进行调用。

属性“evaluationOrder”表示任务内的执行顺序，具有较小“evaluationOrder”值的程序实例应被较早地执行，其值不能为负值。由于在 W3C 的 XML 1.0 规范中同级元素的出现顺序无意义，因此提供了该属性。强烈建议导出工具为每个任务的每个“ProgramInstance”元素的属性赋予递增的编号。

“InputAssignment”“OutputAssignment”和“InOutputAssignment”所属类型“ParameterAssignment”将在 14.2 中介绍。

“FbInstanceTaskAssociation”表示功能块实例与任务的显式关联。属性“fbInstanceName”和“associatedTaskName”为字符串类型。

6.5.3 存储变量“AccessVars”

“AccessVars”表示 IEC 61131-3 中定义的存取变量。其内容如图 12 所示。

“AccessVars”基于抽象复合类型“TextualObjectBase”，该类型将在 7.5.2 中介绍。其中包含的“AccessVariable”也是基于“TextualObjectBase”的。

该元素提供了“alias”“instancePathAndName”和“direction”属性。

“Type”所属类型“TypeRef”将在 11.5 中介绍。

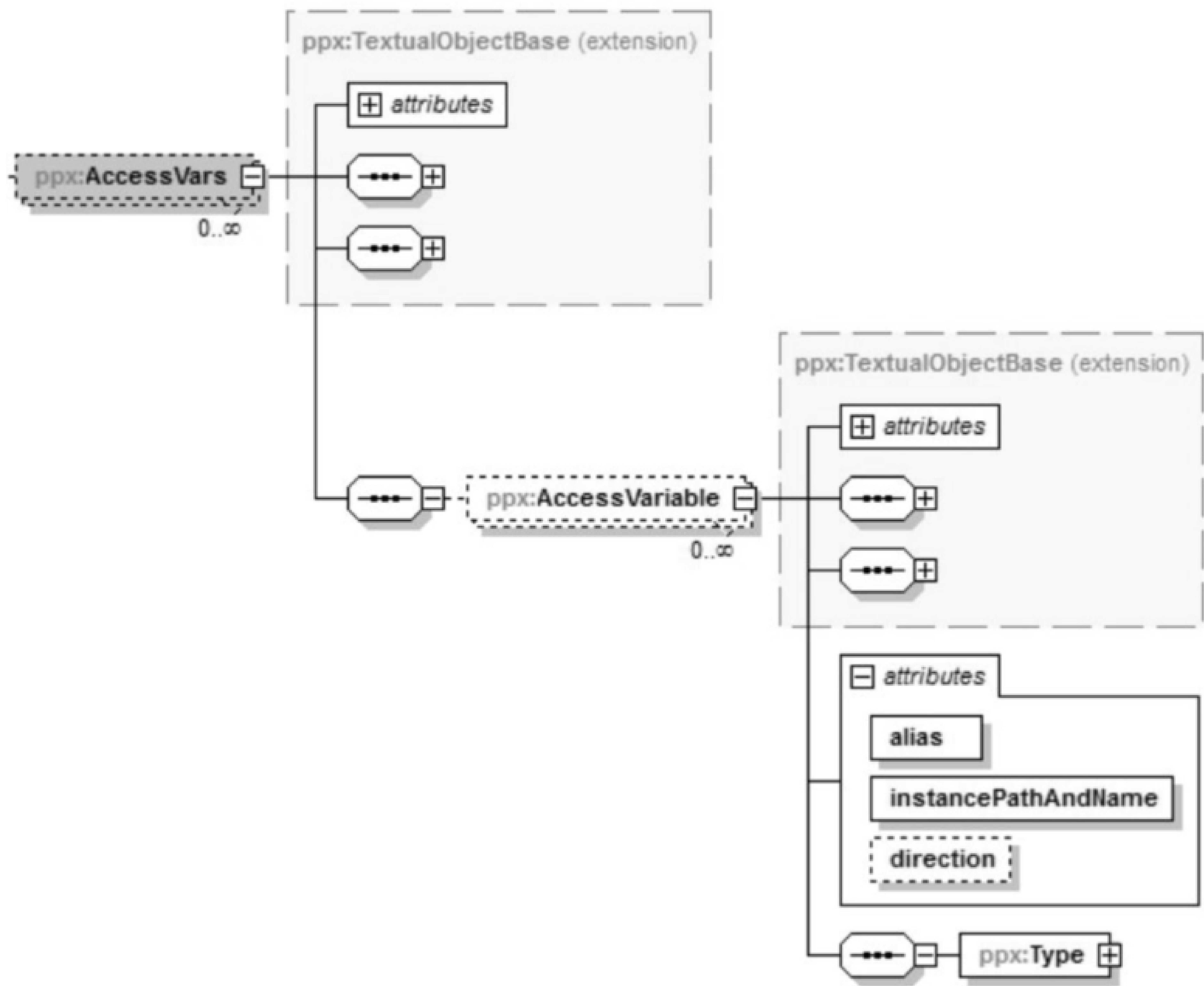


图 12 元素“AccessVars”

6.5.4 配置变量“ConfigVars”

“ConfigVars”表示 IEC 61131-3 中定义的配置变量。其内容如图 13 所示。

“ConfigVars”基于抽象复合类型“TextualObjectBase”，该类型将在 7.5.2 中介绍。其中包含的“ConfigVariable”也是基于“TextualObjectBase”的。

该元素提供了“instancePathAndName”属性。

“Type”所属类型“TypeRef”“InitialValue”所属类型“Value”以及“Address”所属类型“FixedAddressExpression”将分别在 11.5、11.6 和 11.8 中介绍。

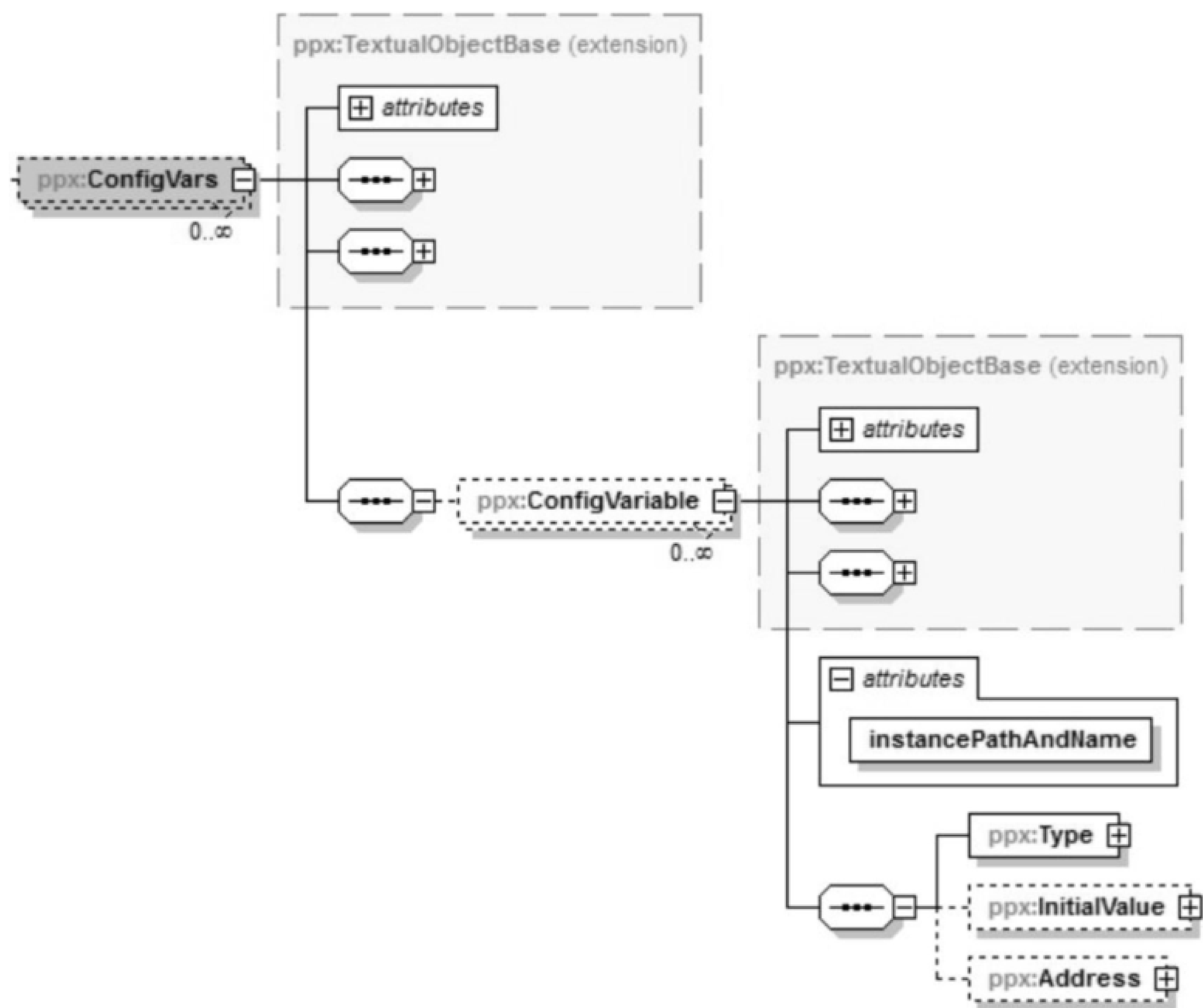


图 13 元素“ConfigVars”

7 抽象复合类型

7.1 抽象复合类型的意义

模式中提供了抽象复合类型,用于以下目的之一。

- a) 避免在派生复合类型中多次定义相似的属性和子元素。为此,特定复合类型中的公共属性和子元素都以抽象复合类型定义。抽象类型由其相关的具体类型进行扩展。
- b) 用于实现供应商特定的扩展。为此,在标准的模式文件中,抽象复合类型被指定为元素的类型。这允许将基于抽象复合类型扩展的具体复合类型应用于元素类型。在这种情况下,抽象复合类型本身甚至可能不包含任何属性或子元素。通过在供应商特定的扩展模式中添加扩展的复合类型,可将其用作元素的类型。

7.2 数据类型规范的抽象复合类型

7.2.1 概述

此类的抽象复合类型与具体复合类型之间的扩展关系如图 14 所示。



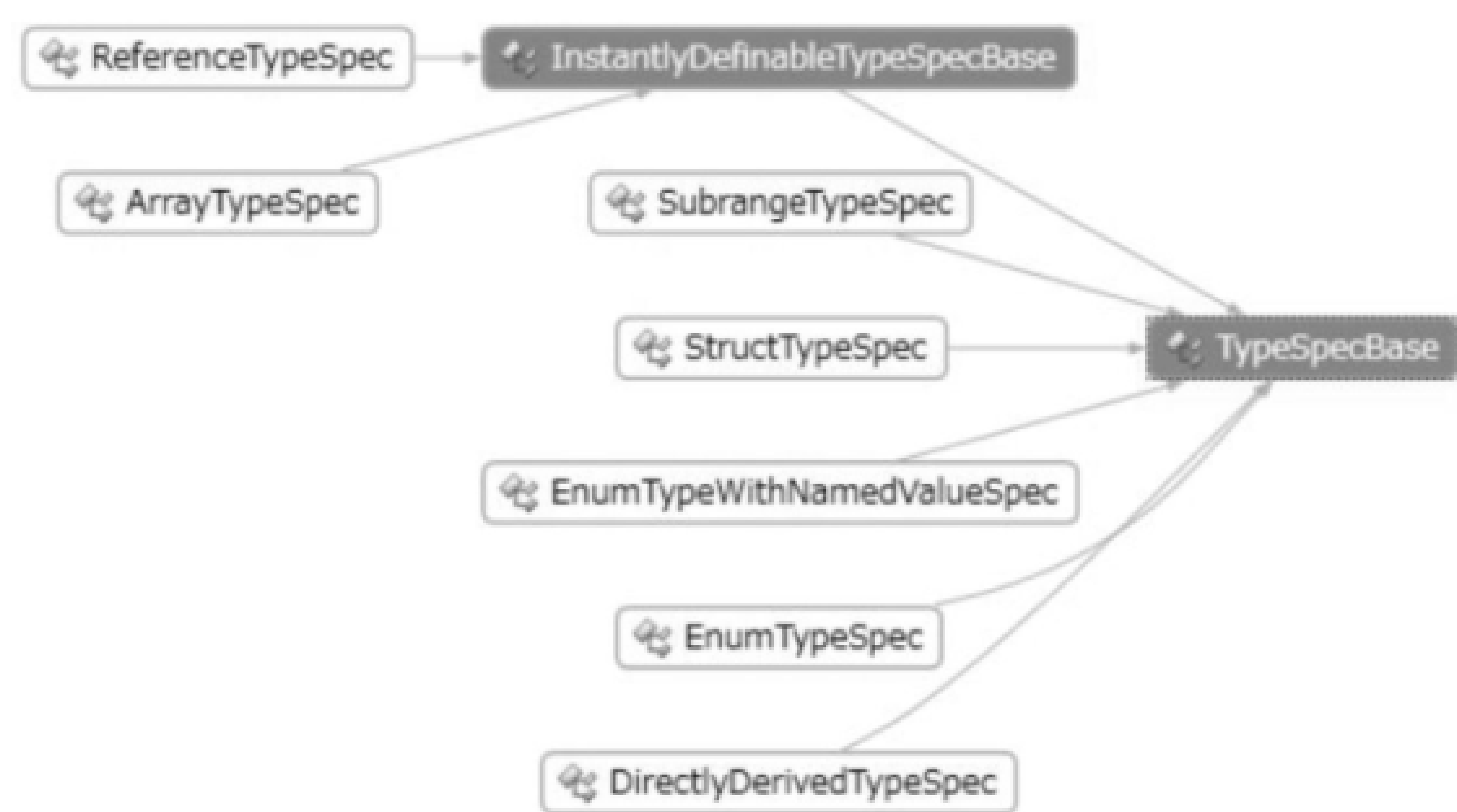


图 14 数据类型声明中复合类型之间的扩展关系

7.2.2 “TypeSpecBase”

抽象复合类型“TypeSpecBase”用于供应商特定的用户定义数据类型扩展。它应被指定为表示用户定义数据类型规范的复合类型的扩展基础,不包含具体内容。

7.2.3 “InstantlyDefinableTypeSpecBase”

抽象复合类型“InstantlyDefinableTypeSpecBase”也用于供应商特定的用户定义数据类型扩展。它应被指定为表示可通过变量声明定义的用户定义数据类型声明的复合类型的扩展基础。它基于 7.2.2 中介绍的抽象复合类型“TypeSpecBase”,但不包含具体内容。

7.3 行为描述抽象类型

7.3.1 概述

此类的抽象复合类型与具体复合类型之间的扩展关系如图 15 所示。

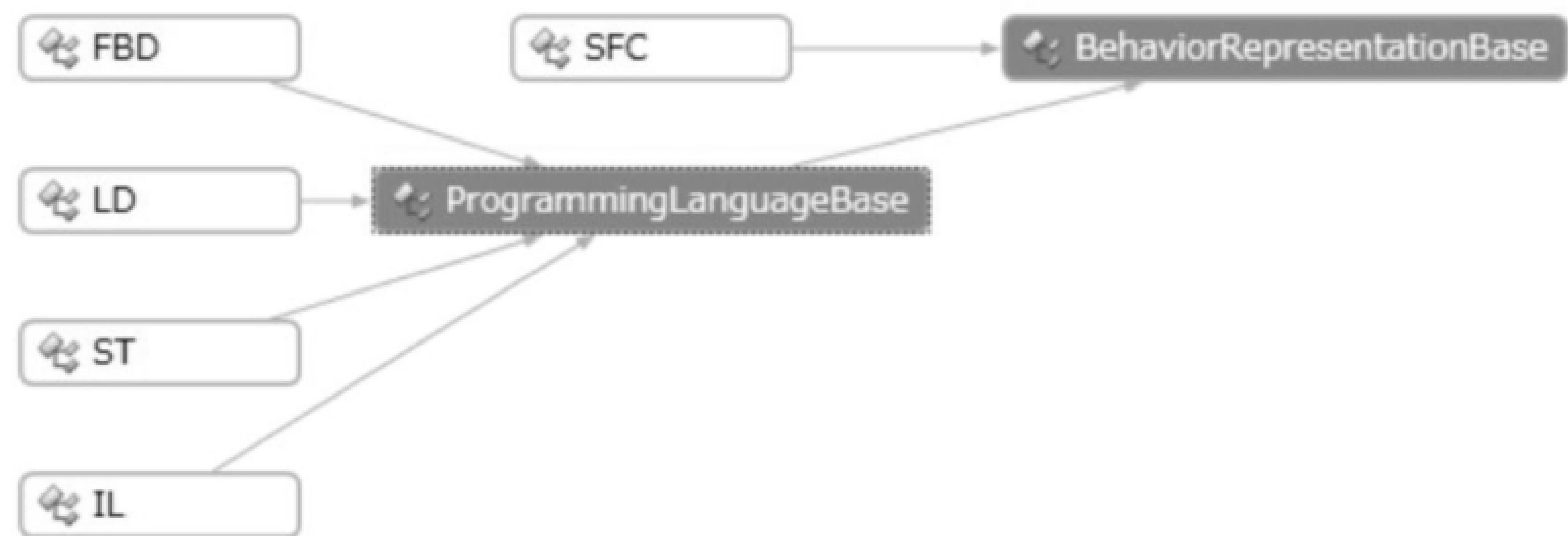


图 15 行为描述中复合类型之间的扩展关系

7.3.2 行为表示基类“BehaviorRepresentationBase”

抽象复合类型“BehaviorRepresentationBase”用于供应商特定的行为描述扩展。它应被指定为表示需要内部状态(例如 SFC)的高层次行为描述的复合类型的扩展基础,不包含任何内容。从该类型导出的行为描述不适用于功能或方法。

7.3.3 编程语言基类“ProgrammingLanguageBase”

抽象复合类型“ProgrammingLanguageBase”也用于供应商特定的行为描述扩展。它应被指定为表示图形化语言或文本化语言(例如 FBD、LD、IL、ST)的复合类型的扩展基础。它扩展了 7.3.2 中介绍的“BehaviorRepresentationBase”,但不包含额外内容。“ProgrammingLanguageBase”可用于功能和方法。

7.4 图形对象的抽象复合类型

7.4.1 概述

此类的抽象复合类型与具体复合类型之间的扩展关系如图 16 所示。

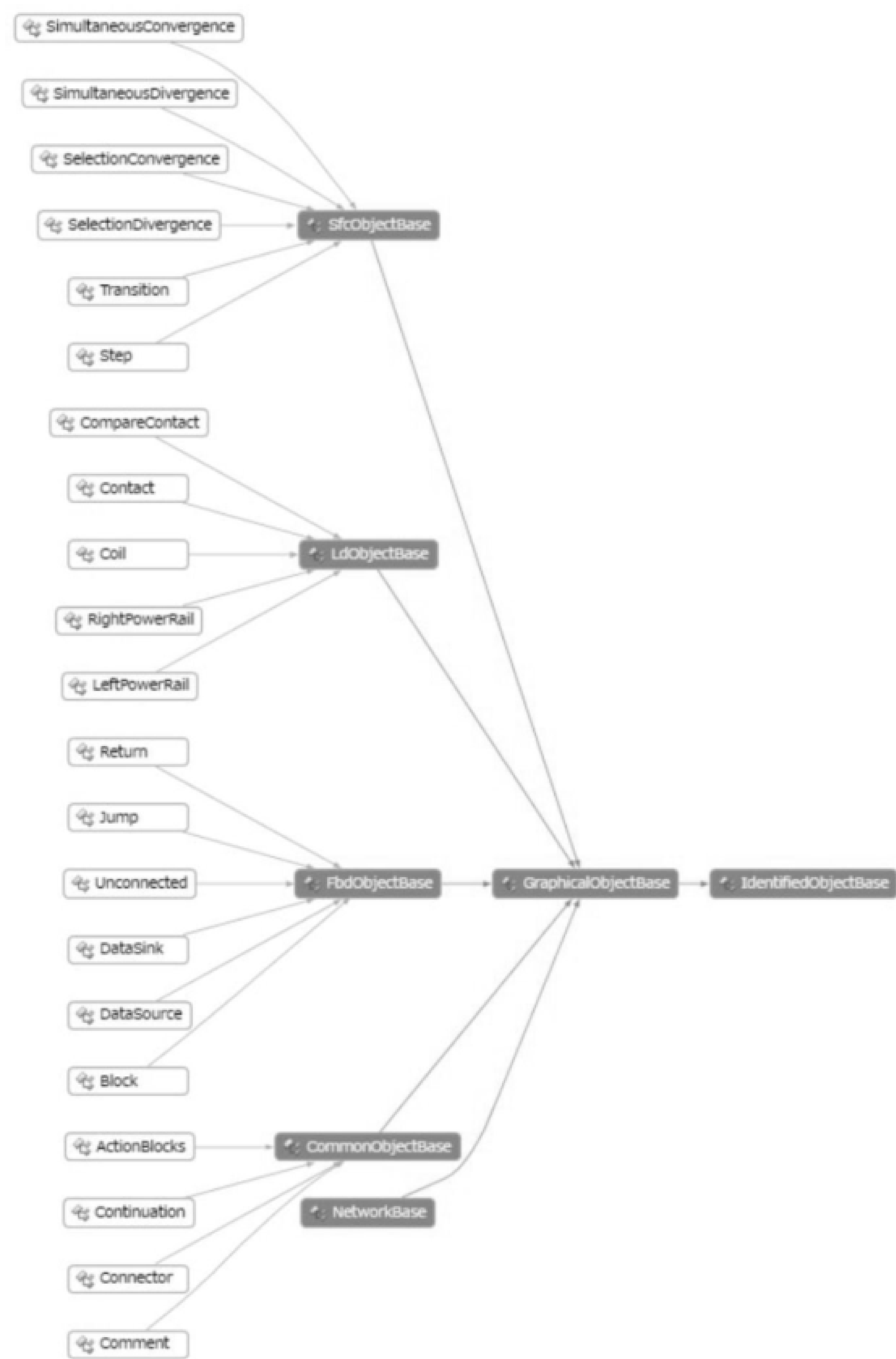


图 16 图形对象中复合类型之间的扩展关系

7.4.2 标识对象基类“IdentifiedObjectBase”

“IdentifiedObjectBase”是抽象复合类型,包含了通过“globalId”属性标识的对象的通用数据。其内容如图 17 所示。

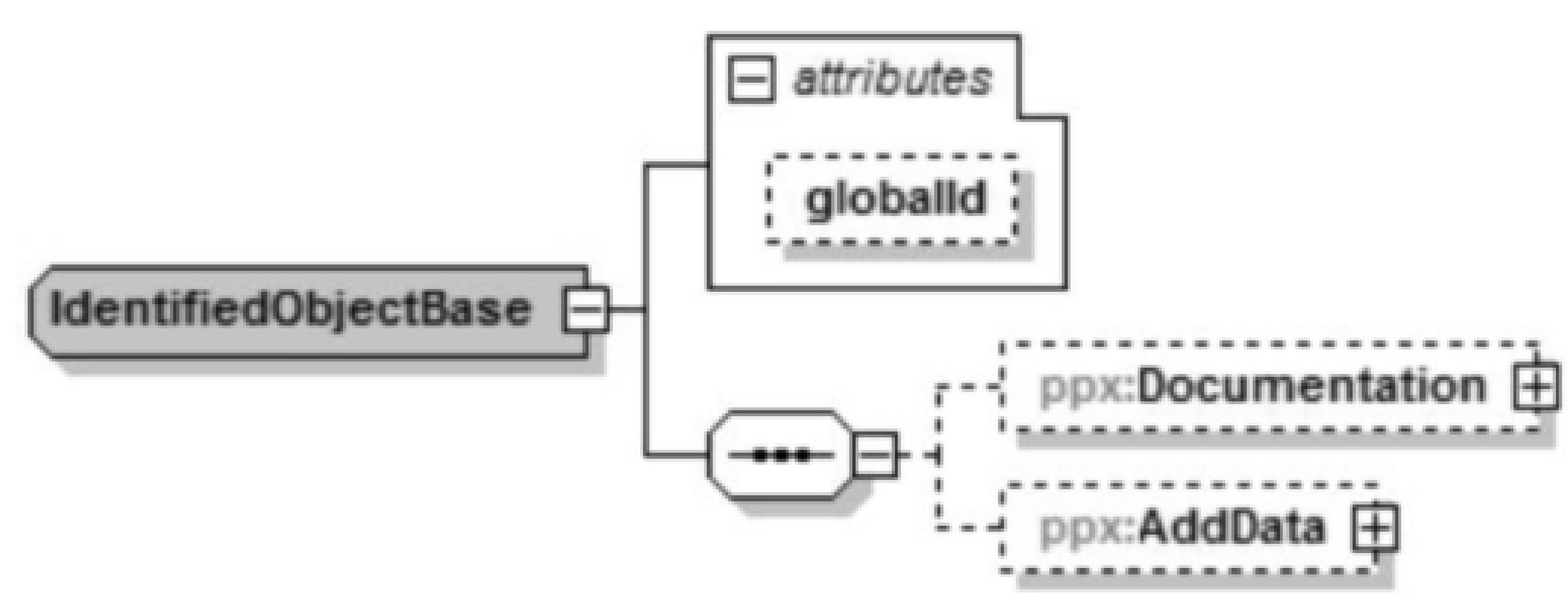


图 17 复合类型 “IdentifiedObjectBase”

属性“globalId”的主要目的是能够从外部明确地标识 XML 文档中的元素。  
元素“Documentation”表示对象的可选文本文件,其所属复合类型“TextBase”将在 15.3 中介绍。

7.4.3 图形对象基类“GraphicalObjectBase”

“GraphicalObjectBase”是抽象复合类型,包含了图形对象的通用数据,例如相对位置和大小。其内容如图 18 所示。

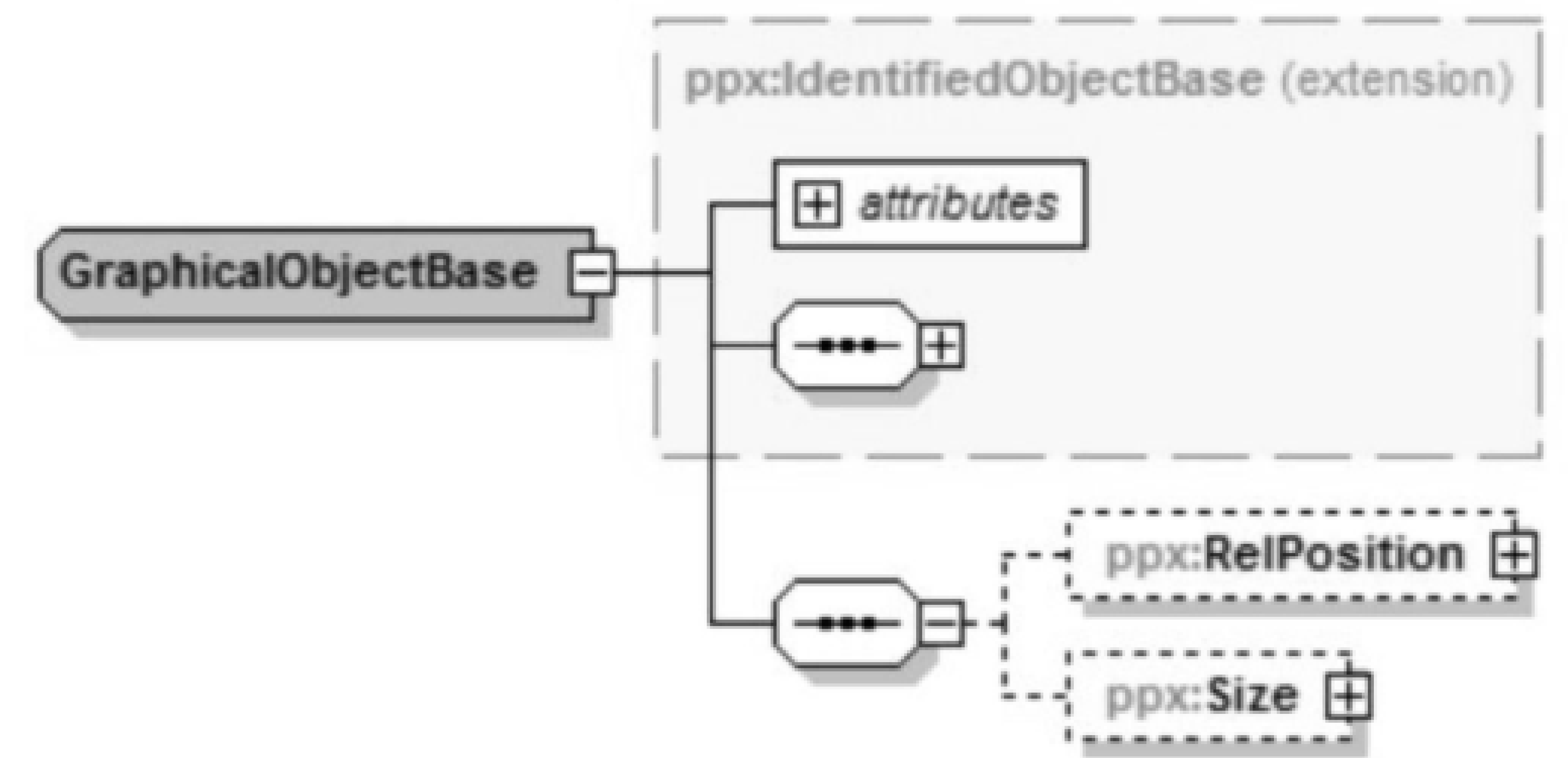


图 18 复合类型 “GraphicalObjectBase”

“GraphicalObjectBase”基于抽象类型“IdentifiedObjectBase”,该类型已在 7.4.2 中介绍。  
元素“RelPosition”表示图形对象在父图形对象内锚点的相对位置,其所属复合类型“XyDecimal-Value”将在 15.1 中介绍。  
元素“Size”表示图形对象的大小,其所属复合类型“XyDecimalValue”将在 15.1 中介绍。

7.4.4 通用对象基类“CommonObjectBase”

“CommonObjectBase”是抽象复合类型,作为表示 FBD、LD 和 SFC 中图形对象复合类型的扩展基础。该复合类型也可用于供应商特定的图形对象扩展。通过添加由此抽象复合类型扩展而来的复合类型,供应商不仅可使用标准图形对象(例如“Connector”“Continuation”等),还可使其特定的图形对象在 FBD、LD 和 SFC 中通用。其内容如图 19 所示。

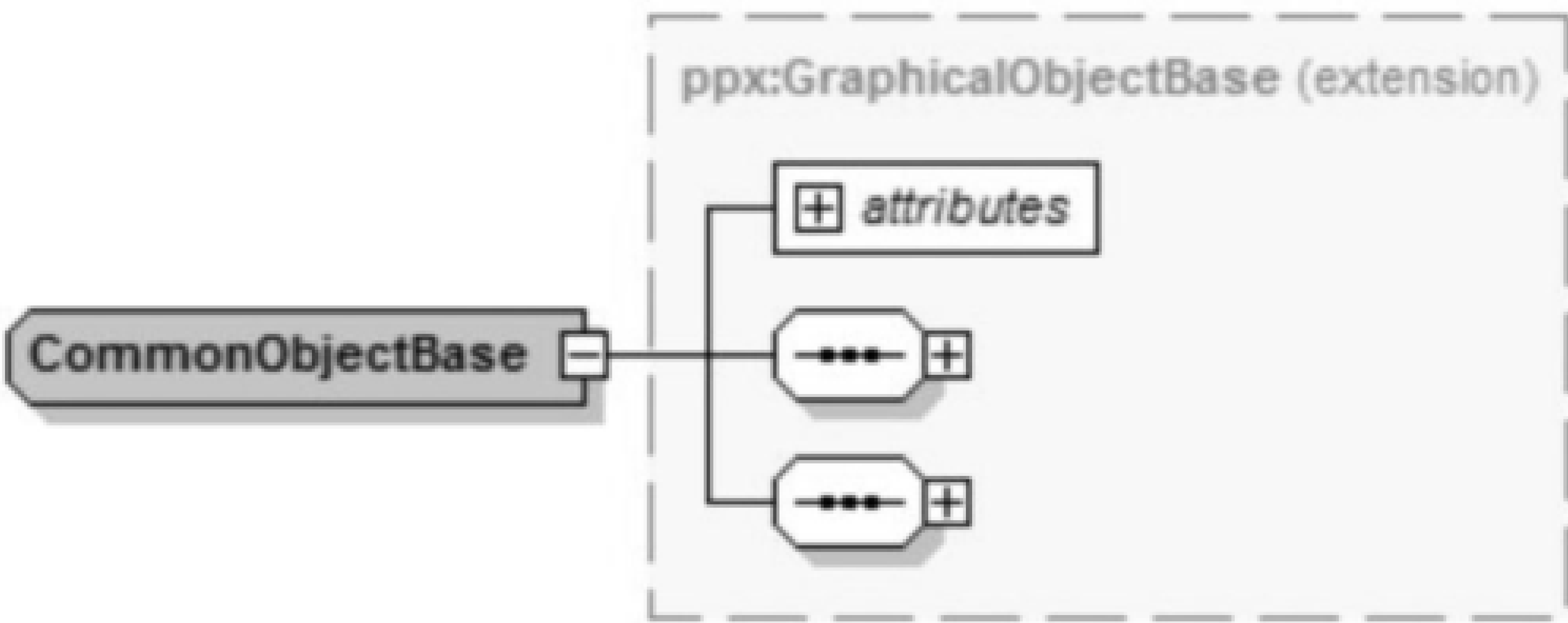


图 19 复合类型“CommonObjectBase”

该复合类型扩展了 7.4.3 中介绍的基本复合类型“GraphicalObjectBase”,但其自身不包含特定的内容。

7.4.5 FBD 对象基类“FbdObjectBase”

“FbdObjectBase”是抽象复合类型,作为描述 FBD 和 LD 图形对象复合类型的扩展基础。该复合类型也可用于供应商特定的图形对象扩展。通过添加由此抽象类型扩展而来的复合类型,供应商不仅可使用标准图形对象(例如“Block”“DataSource”“DataSink”等),还可使其特定的图形对象在 FBD 和 LD 中通用。其内容如图 20 所示。

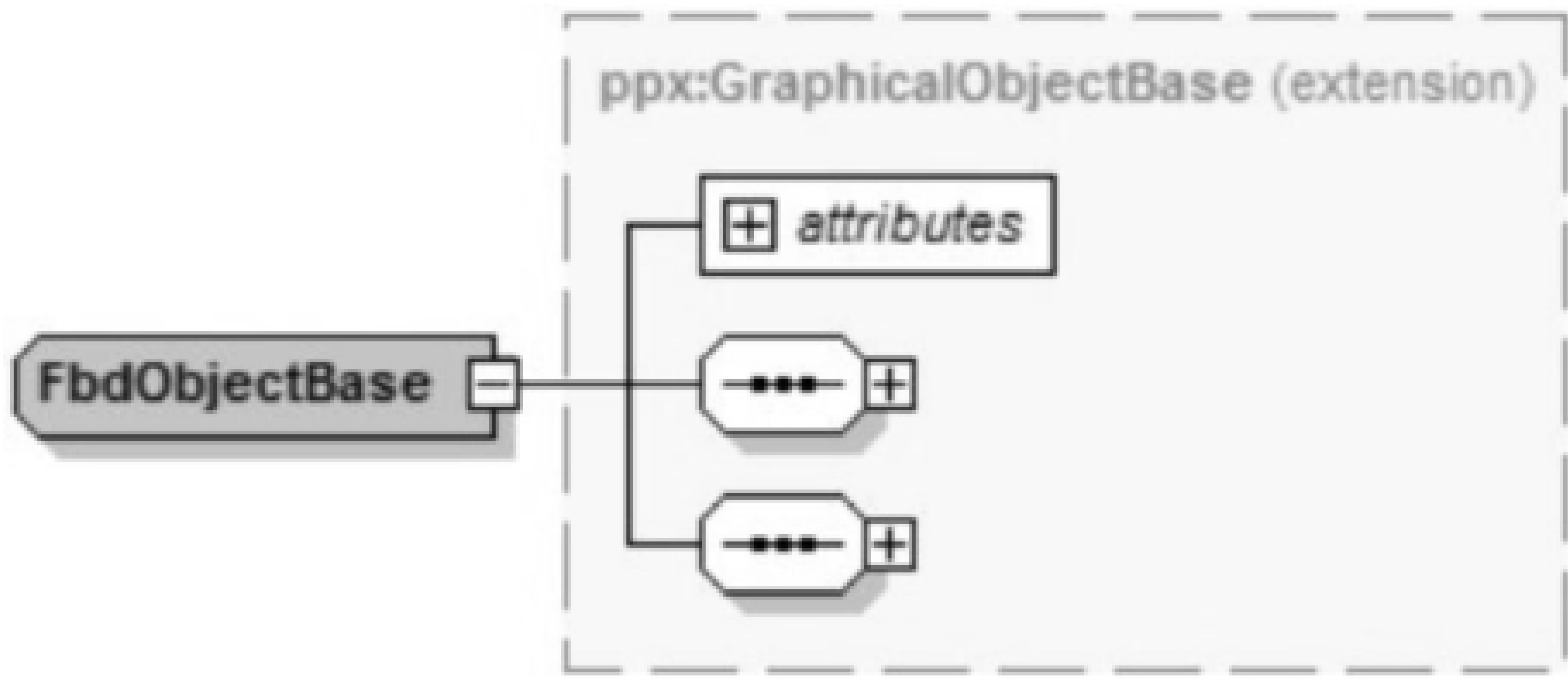


图 20 复合类型“FbdObjectBase”

该复合类型扩展了 7.4.3 中介绍的基本复合类型“GraphicalObjectBase”,但其自身不包含特定的内容。

7.4.6 LD 对象基类“LdObjectBase”

“LdObjectBase”是抽象复合类型,作为描述 LD 图形对象复合类型的扩展基础。该复合类型也可用于供应商特定的图形对象扩展。通过添加由此抽象类型扩展而来的复合类型,供应商不仅可使用标准图形对象(例如“Coil”“Contact”等),还可使其特定的图形对象在 LD 中可用。其内容如图 21 所示。

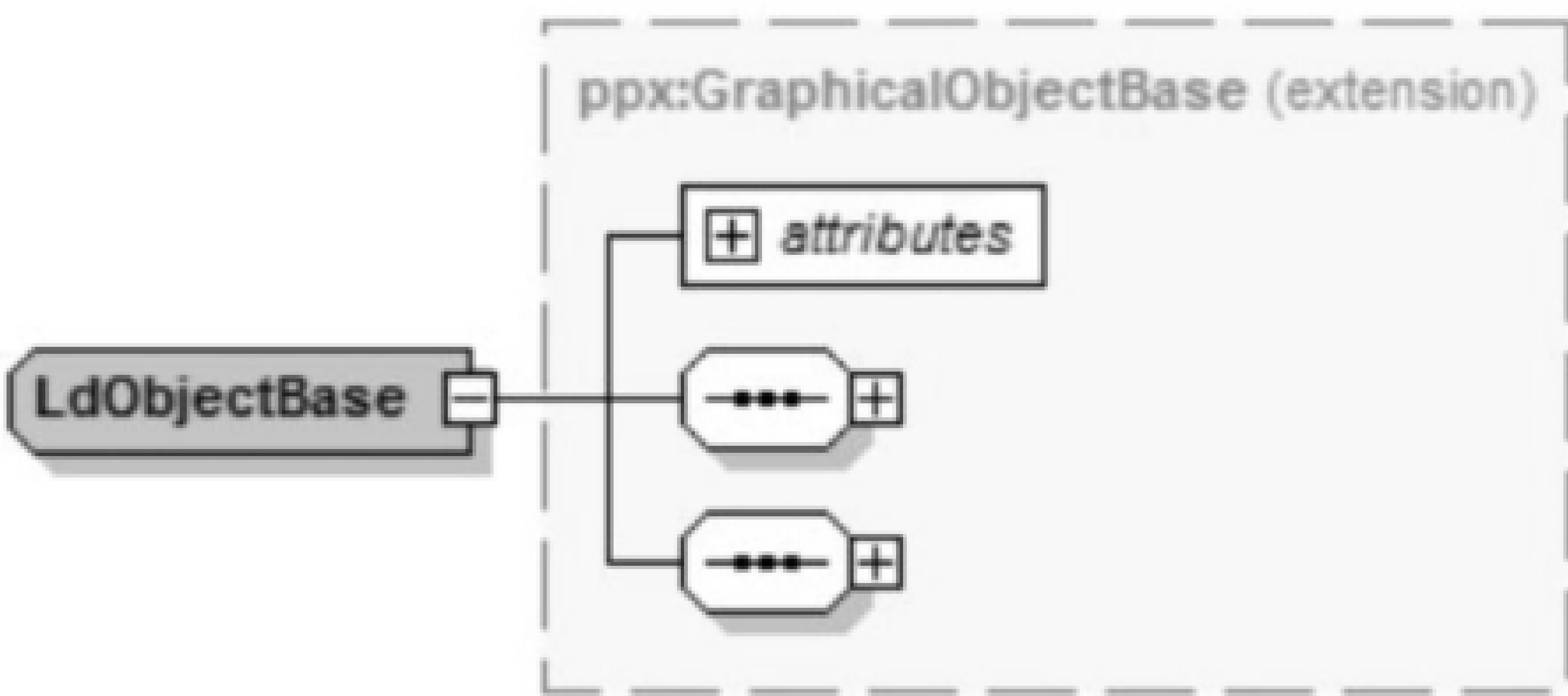


图 21 复合类型“LdObjectBase”

该复合类型扩展了 7.4.3 中介绍的基本复合类型“GraphicalObjectBase”，但其自身不包含特定的内容。

7.4.7 SFC 对象基类“SfcObjectBase”

“SfcObjectBase”是抽象复合类型，作为描述 SFC 图形对象复合类型的扩展基础。该复合类型也可用于供应商特定的图形对象扩展。通过添加由此抽象类型扩展而来的复合类型，供应商不仅可使用标准图形对象（例如“Step”“Transition”等），还可使其特定的图形对象在 SFC 中可用。其内容如图 22 所示。

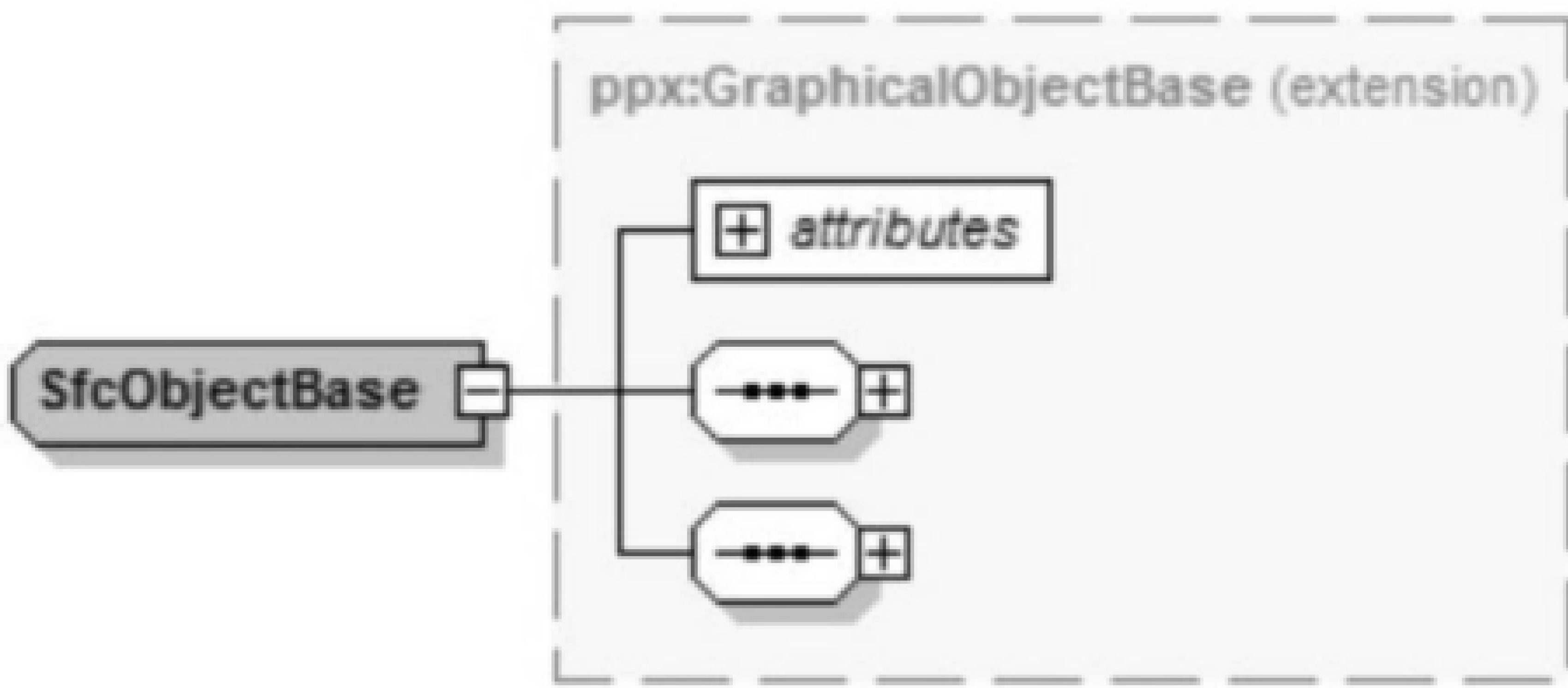


图 22 复合类型“SfcObjectBase”

该复合类型扩展了 7.4.3 中介绍的基本复合类型“GraphicalObjectBase”，但其自身不包含特定的内容。

7.4.8 网络基类“NetworkBase”

“NetworkBase”是抽象复合类型，包含了 LD 和 FBD 网络通用属性和子元素。它作为复合类型“LD”中元素“Rung”和复合类型“FBD”中元素“FbdNetWork”的扩展基础。其内容如图 23 所示。

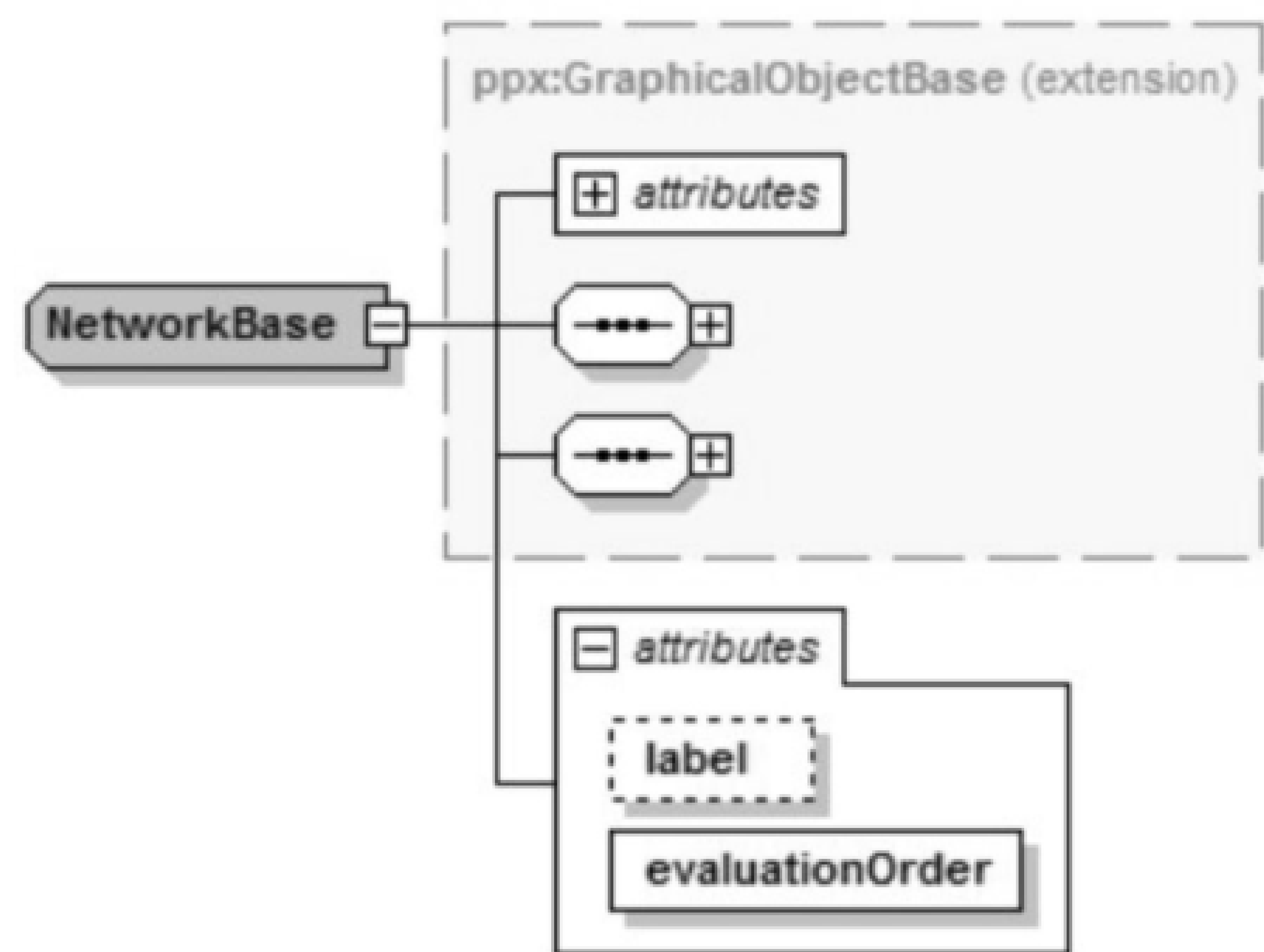


图 23 复合类型“NetworkBase”

该复合类型扩展了 7.4.3 中介绍的基本复合类型“GraphicalObjectBase”。

可选属性“label”表示网络的标号,可作为执行控制元素“jump”的目标。

必需属性“evaluationOrder”表示 LD 或 FBD 中每个网络的执行顺序,在 IEC 61131-3 中十分重要。具有较小“evaluationOrder”值的网络应被更早执行,其值不能为负值。由于在 W3C 的 XML 1.0 规范中同级元素的出现顺序无意义,因此提供了“evaluationOrder”属性。强烈建议导出工具为每个“Rung”元素(LD)和每个“FbdNetWork”元素(LD、FBD)的属性赋予递增的编号。对于 SFC,每个“GraphicalExpression”的“evaluationOrder”值应为“0”。

7.5 文本构造的抽象复合类型

7.5.1 概述

此类的抽象复合类型与具体复合类型之间的扩展关系如图 24 所示。

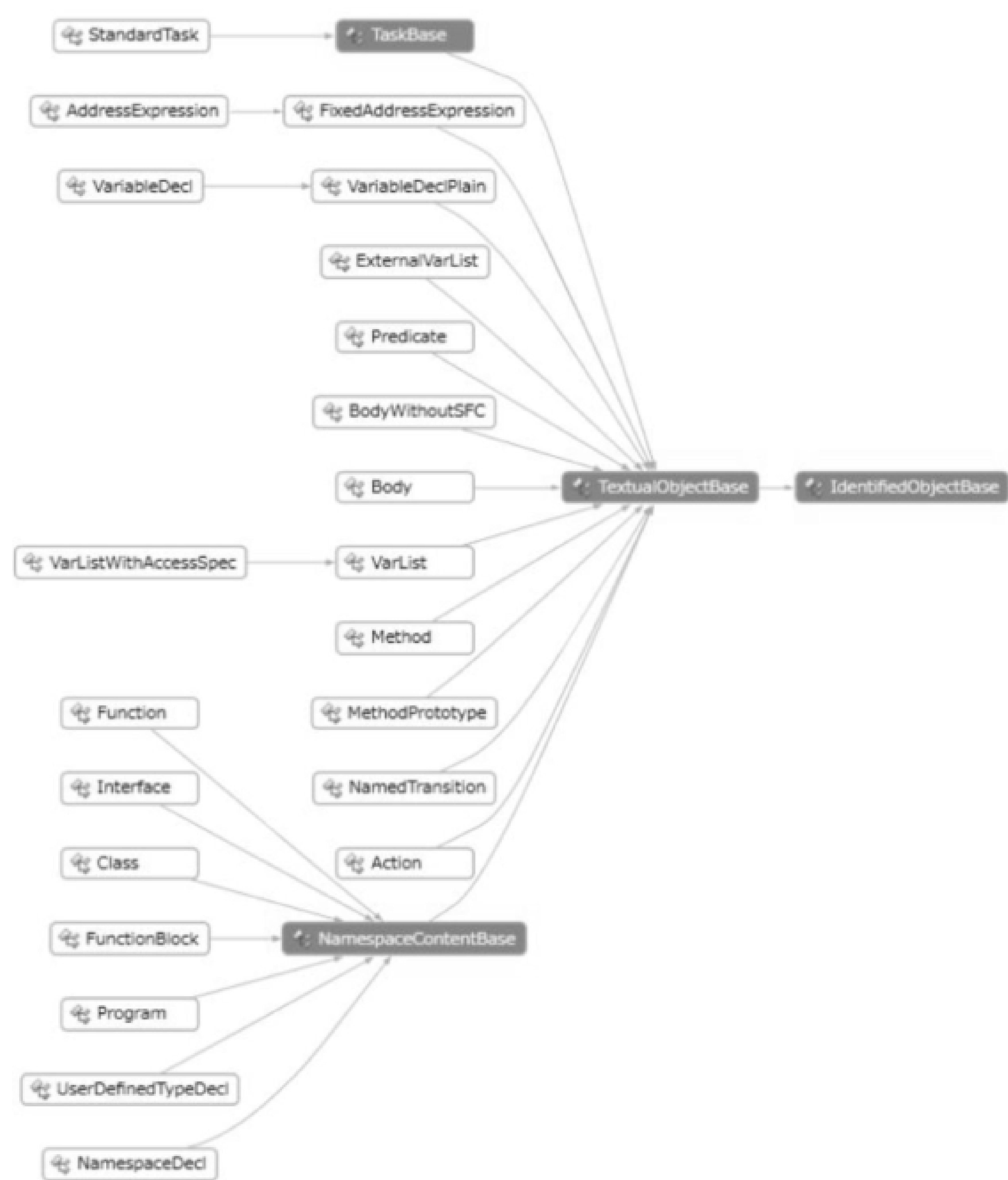


图 24 文本对象中复合类型之间的扩展关系

7.5.2 文本对象基类“TextualObjectBase”

“TextualObjectBase”是抽象复合类型,包含了文本构造所声明的所有文本对象的通用数据。其内容如图 25 所示。

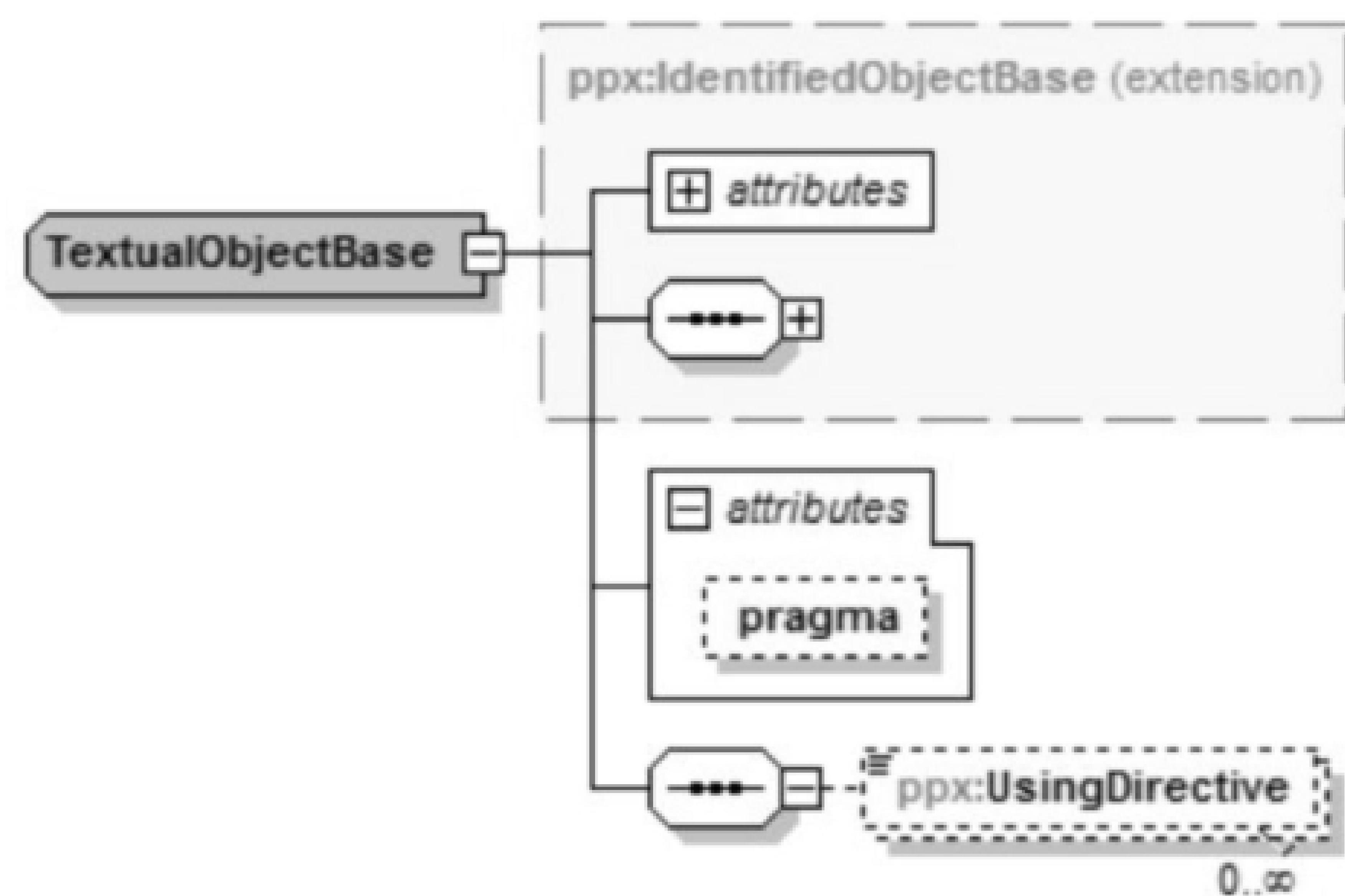


图 25 复合类型“TextualObjectBase”

“TextualObjectBase”基于 7.4.2 中介绍的抽象类型“IdentifiedObjectBase”。

属性“pragma”为 string 类型。为了在同一个字符串中保留多个注释,每个注释应嵌入到大括号中,如 IEC 61131-3 中所定义。

元素“UsingDirective”为 string 类型,表示使用带有“USING”关键字的命名空间指令,如 IEC 61131-3中所定义。

7.5.3 命名空间内容基类“NamespaceContentBase”

“NamespaceContentBase”是抽象复合类型,包含了命名空间内容的通用数据,例如数据类型和 POU。其内容如图 26 所示。

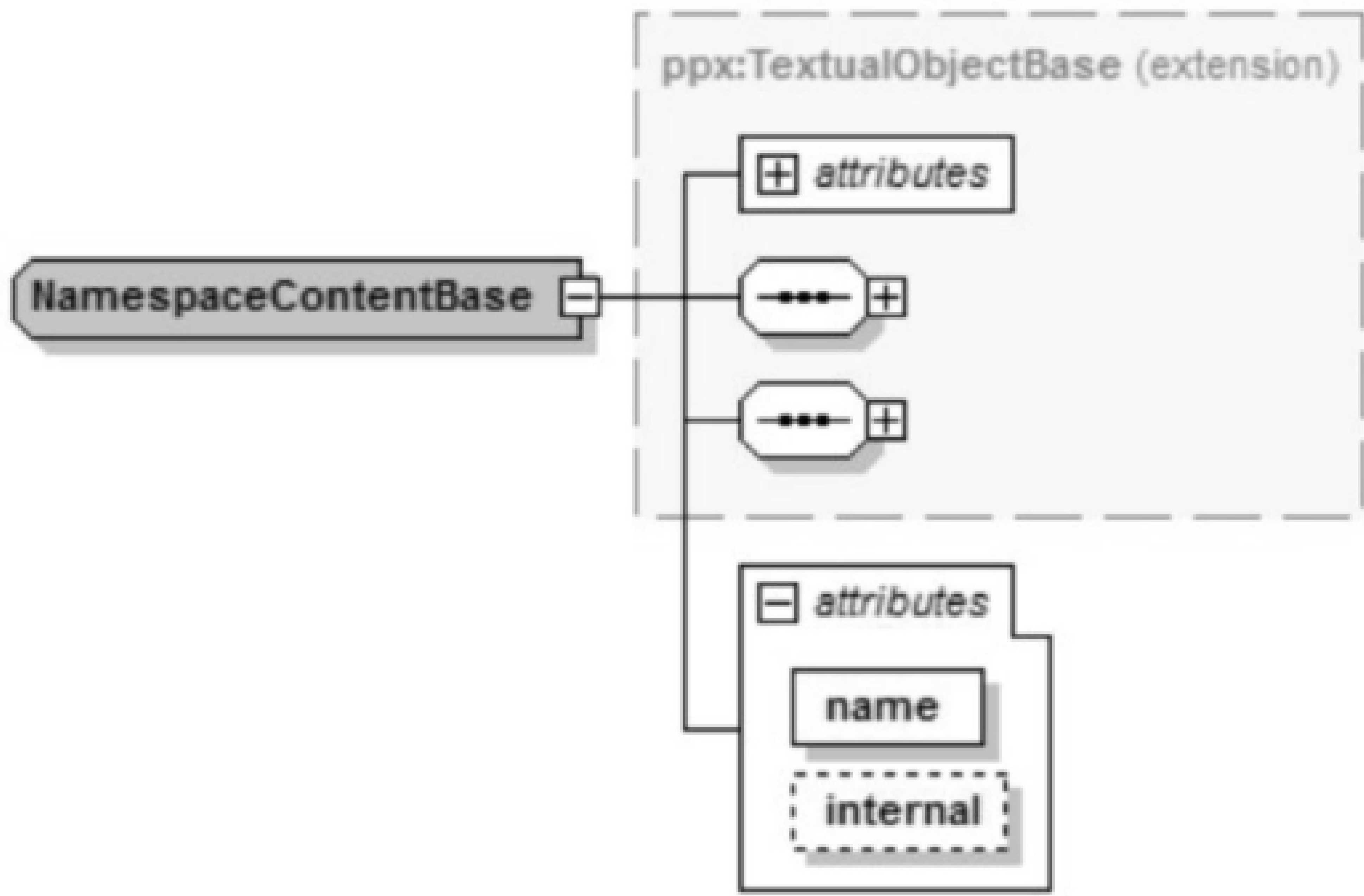


图 26 复合类型“NamespaceContentBase”

“NamespaceContentBase”基于 7.5.2 中介绍的抽象类型“TextualObjectBase”。

必需属性“name”表示对象的名称,例如用户定义的数据类型或 POU 定义。可选属性“internal”表示具有关键字“INTERNAL”的内部访问说明符,如 IEC 61131-3 中所定义。

7.5.4 任务基类“TaskBase”

“TaskBase”是抽象复合类型,作为 IEC 61131-3 中定义的任务复合类型的扩展基础。它用于提供供应商特定的任务扩展。通过添加由此抽象复合类型扩展而来的复合类型,供应商不仅可使用由此抽象复合类型扩展的“StandardTask”,还可定义其特定的任务。其内容如图 27 所示。



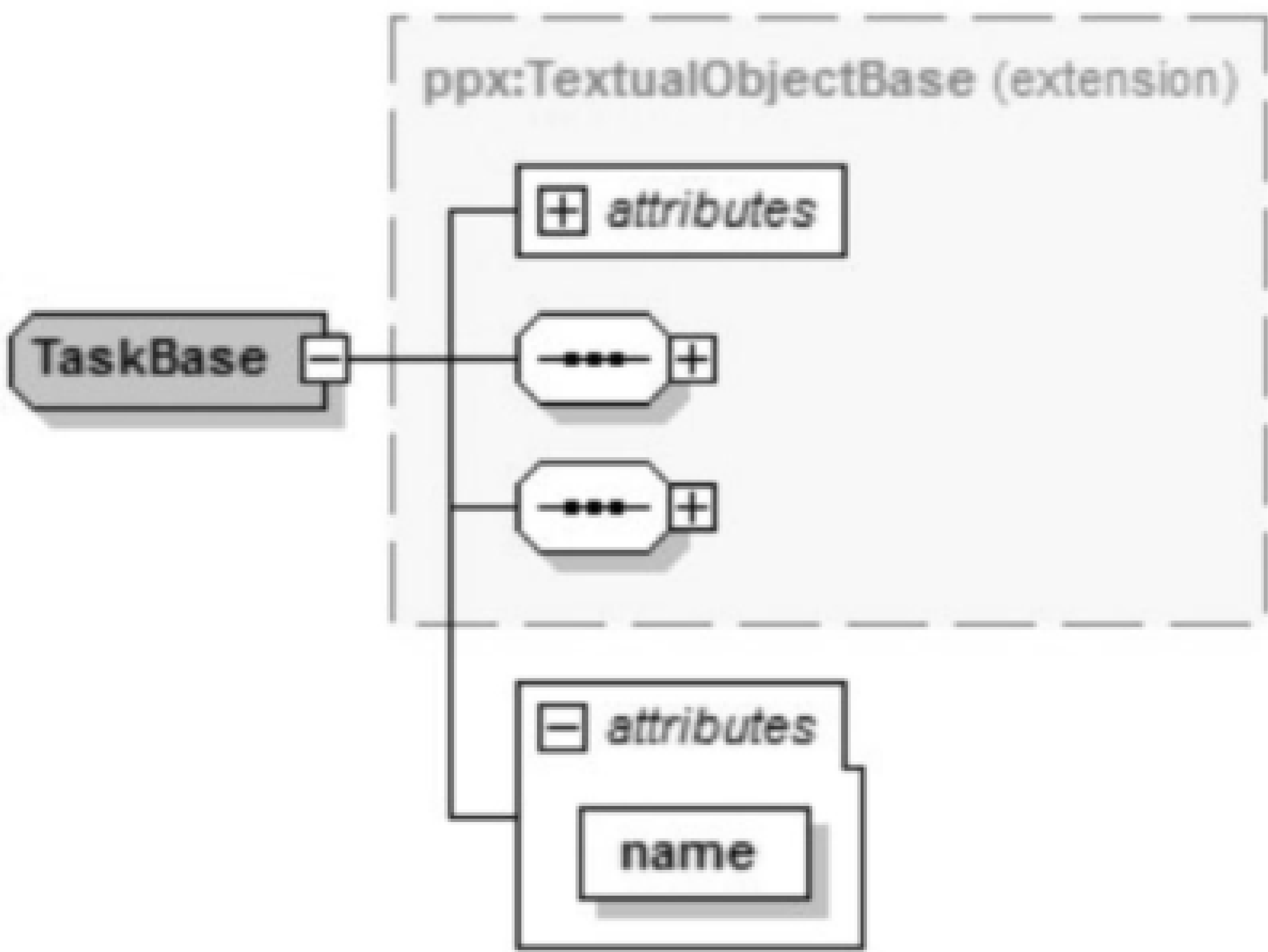


图 27 复合类型“TaskBase”

“TaskBase”基于 7.5.2 中介绍的抽象复合类型“TextualObjectBase”。  
必需属性“name”表示任务的名称,可由程序实例声明引用,如 IEC 61131-3 中所定义。

8 命名空间声明

复合类型“NamespaceDecl”表示 IEC 61131-3 中定义的命名空间。其内容如图 28 所示。

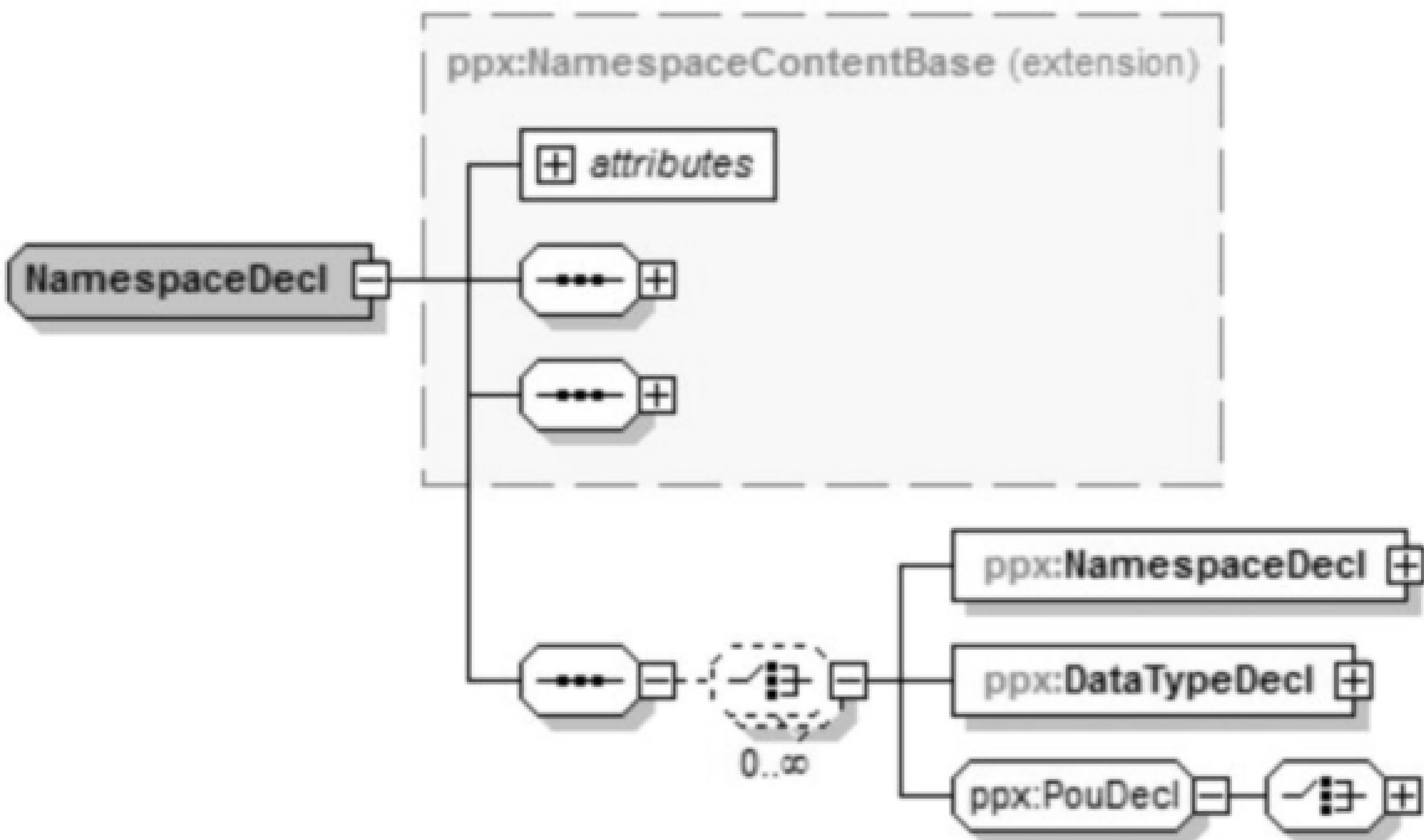


图 28 复合类型“NamespaceDecl”

“NamespaceDecl”基于 7.5.3 中介绍的抽象类型“NamespaceContentBase”。  
命名空间可递归定义,因此元素“NamespaceDecl”可包含在其自身中。  
“DataTypeDecl”所属类型“UserDefinedTypeDecl”将在 9.1 中介绍。“PouDecl”组的内容将在 10.1 中介绍。

9 用户定义数据类型声明

9.1 用户定义数据类型“UserDefinedTypeDecl”

复合类型“UserDefinedTypeDecl”表示 IEC 61131-3 中定义的所有用户定义数据类型相关的元素。其内容如图 29 所示。

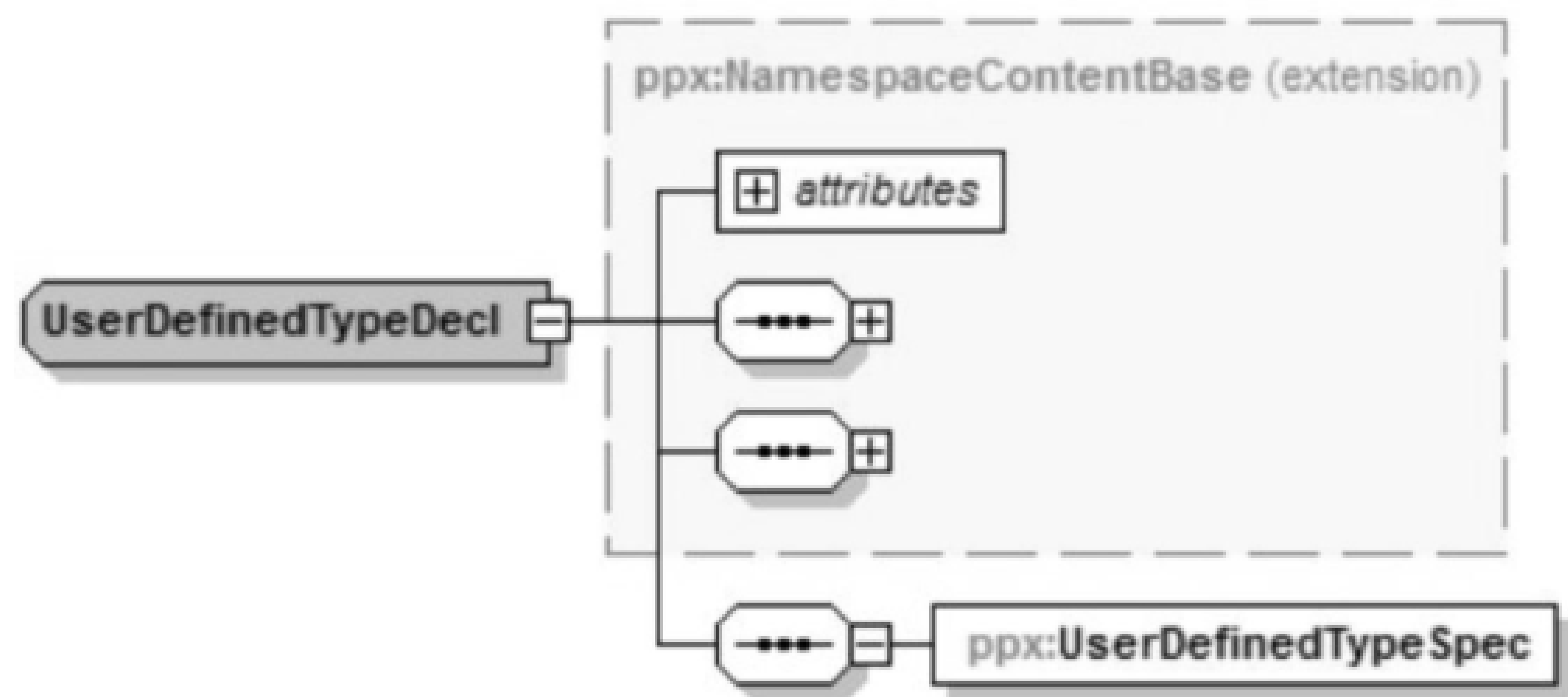


图 29 复合类型“UserDefinedTypeDecl”

“UserDefinedTypeDecl”继承了 7.5.3 中介绍的“NamespaceContentBase”类型的所有元素。  
“UserDefinedTypeSpec”所属抽象复合类型“TypeSpecBase”已在 7.2.2 中介绍。该元素应包含一个派生具体的用户定义数据类型。4.4 中介绍了如何使用抽象类型。  
IEC 61131-3 中定义的可能派生类型将分别在 9.2、9.3、9.4、9.5、9.6、9.7、9.8 和 9.9 中介绍。

9.2 数组数据类型“ArrayTypeSpec”

复合类型“ArrayTypeSpec”表示 IEC 61131-3 中定义的数组数据类型。其内容如图 30 所示。

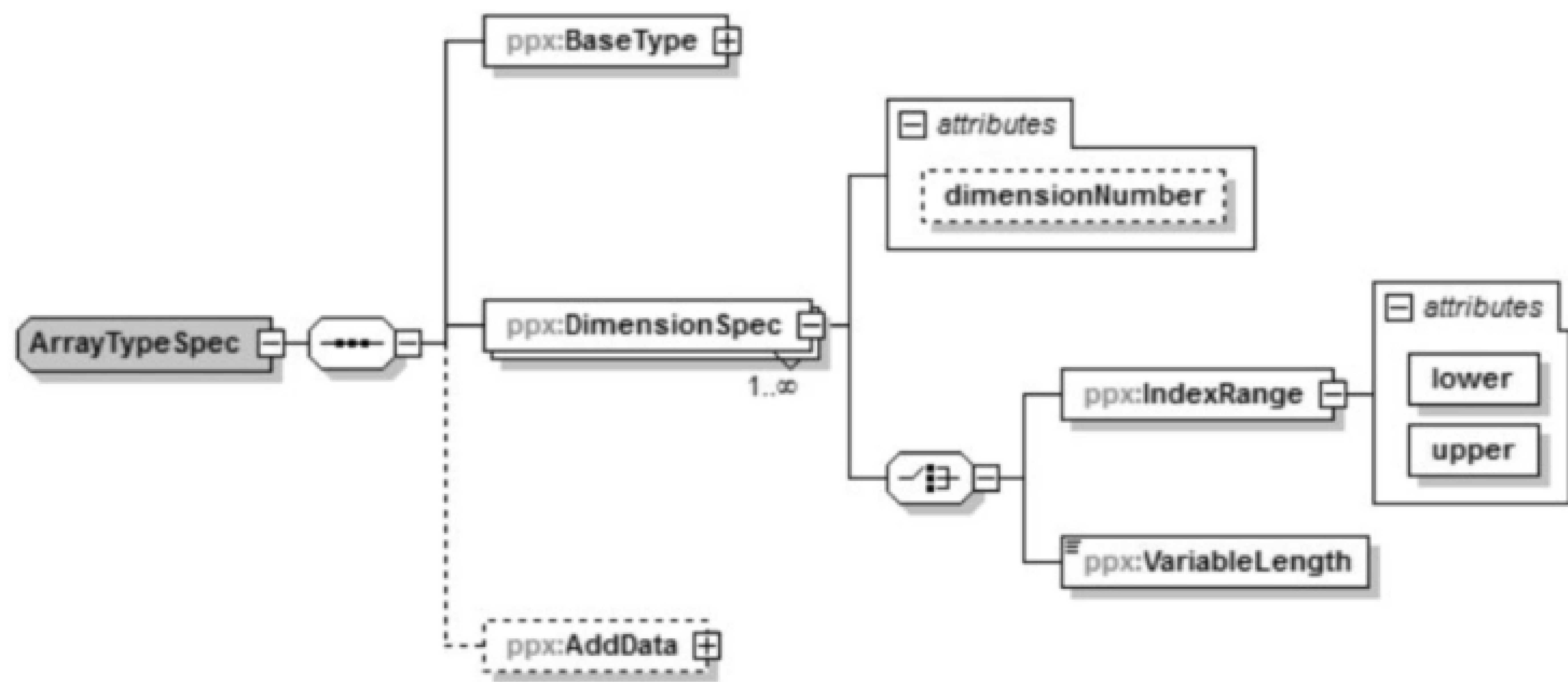


图 30 复合类型“ArrayTypeSpec”

“BaseType”表示数组的基础类型名。“BaseType”所属类型“TypeRef”将在 11.5 中介绍。  
“DimensionSpec”用于指定数组数据类型的一个或多个维度，数组实际的维数为属性“dimension-

Number”的值。

元素“IndexRange”的属性值用于指定维度的上下边界。如果数组是 IEC 61131-3 中定义的变长数组,则应使用元素“VariableLength”。

9.3 直接派生数据类型“DirectlyDerivedTypeSpec”

复合类型“DirectlyDerivedTypeSpec”表示 IEC 61131-3 中定义的直接派生数据类型。其内容如图 31所示。

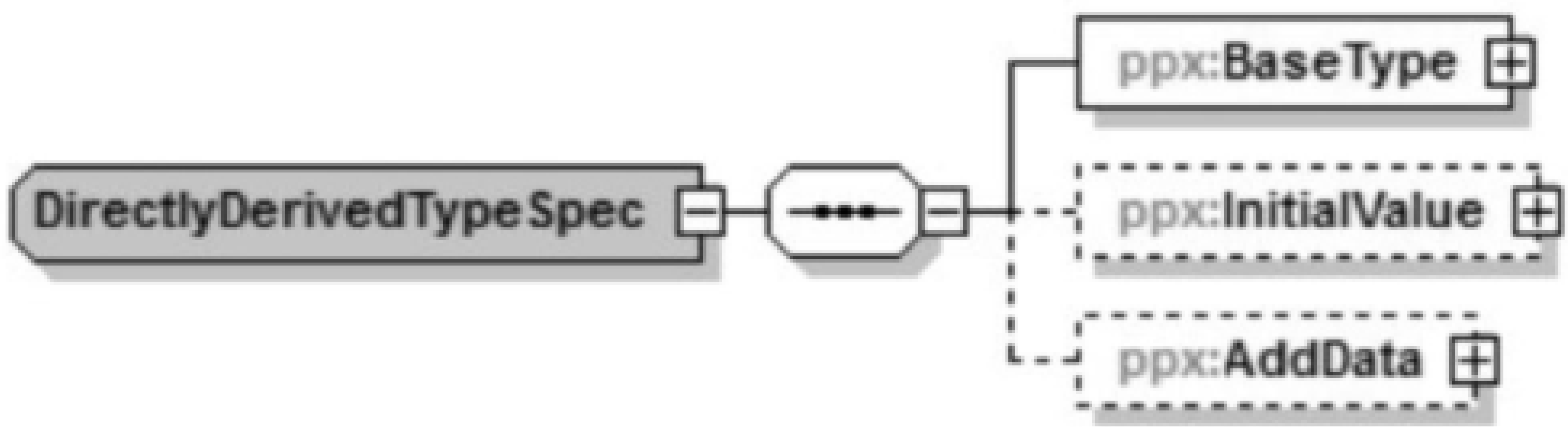


图 31 复合类型“DirectlyDerivedTypeSpec”

“BaseType”所属类型“TypeRef”和“InitialValue”所属类型“Value”将分别在 11.5 和 11.6 中介绍。

9.4 枚举数据类型“EnumTypeSpec”

复合类型“EnumTypeSpec”表示 IEC 61131-3 中定义的枚举数据类型。其内容如图 32 所示。

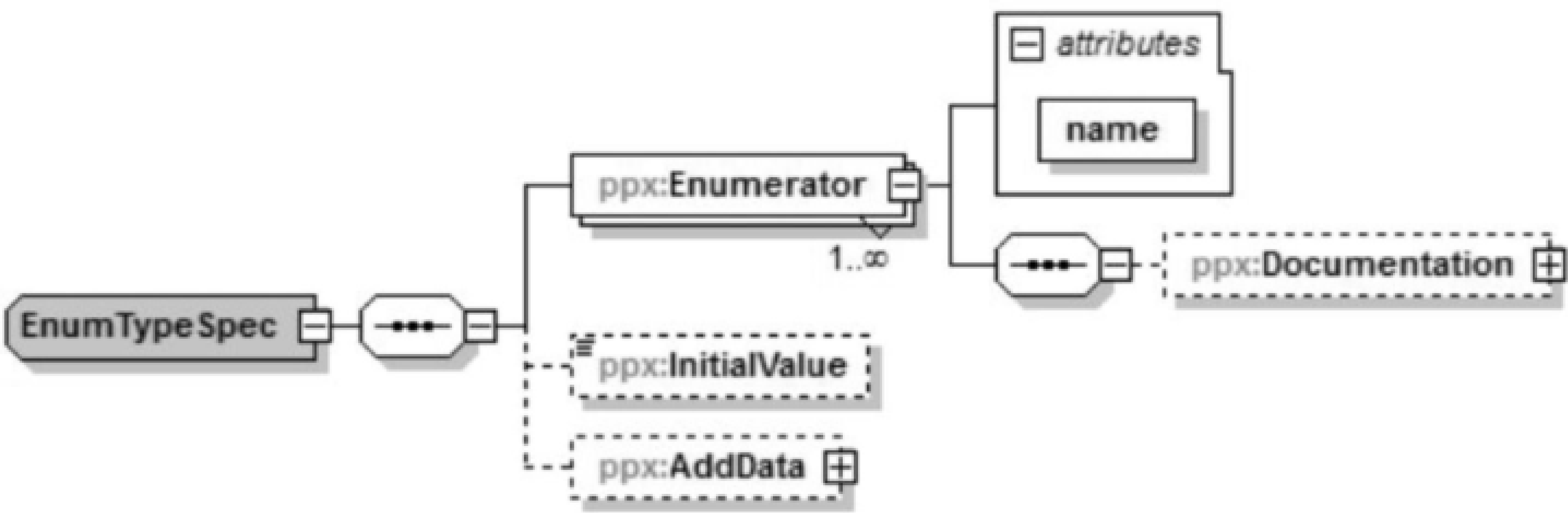


图 32 复合类型“EnumTypeSpec”

枚举列表中的每个元素都需要一个“Enumerator”元素。属性“name”存储了相关的标识符。“Documentation”所属类型“TextBase”将在 15.3 中介绍。

“InitialValue”为 string 类型,表示枚举类型的初始值,该值应为此类型的枚举值之一。

9.5 带名称值的枚举类型“EnumTypeWithNamedValueSpec”

复合类型“EnumTypeWithNamedValueSpec”表示 IEC 61131-3 中定义的带名称值的数据类型。其内容如图 33 所示。

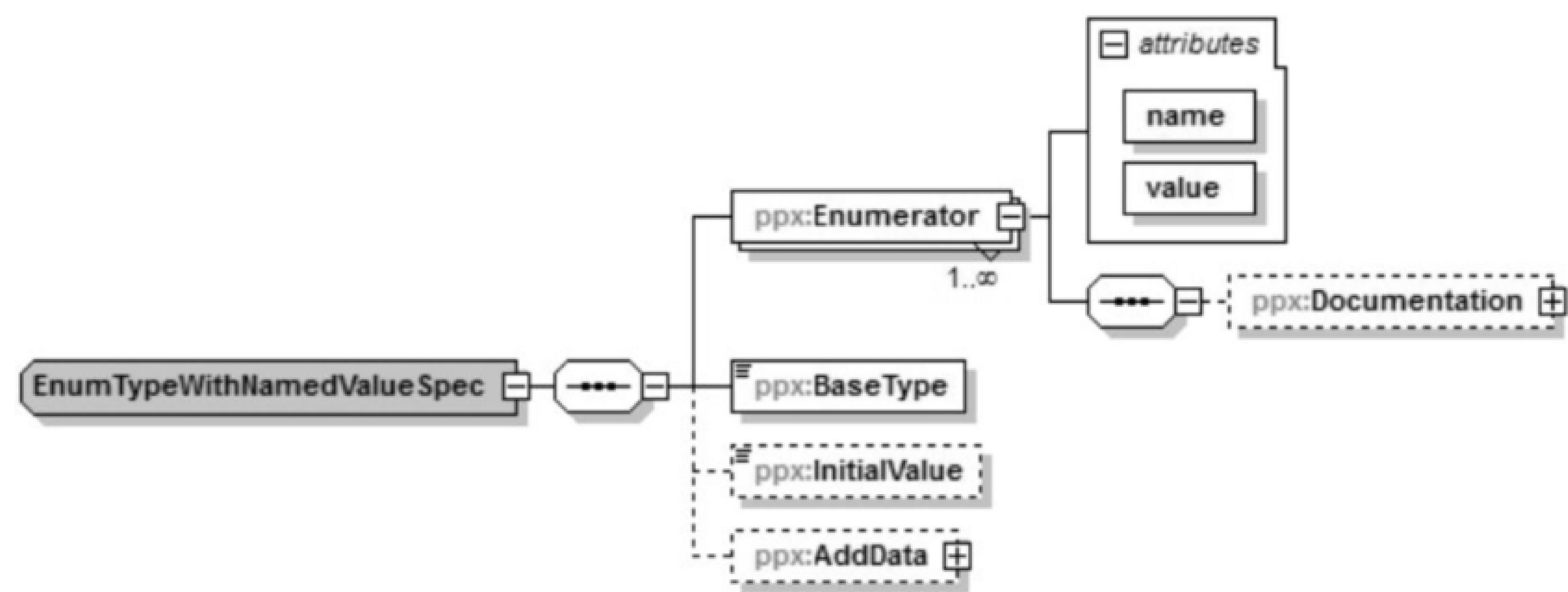


图 33 复合类型“EnumTypeWithNamedValueSpec”

枚举列表的每个元素都需要一个“Enumerator”元素。属性“name”存储了相关的标识符。此外，属性“value”存储了与标志符相关联的枚举值。“Documentation”所属类型“TextBase”将在 15.3 中介绍。  
“BaseType”所属类型“ElementaryType”将在 9.9 中介绍。“InitialValue”为 string 类型。

9.6 结构化数据类型“StructTypeSpec”

复合类型“StructTypeSpec”表示 IEC 61131-3 中定义的结构化数据类型。其内容如图 34 所示。

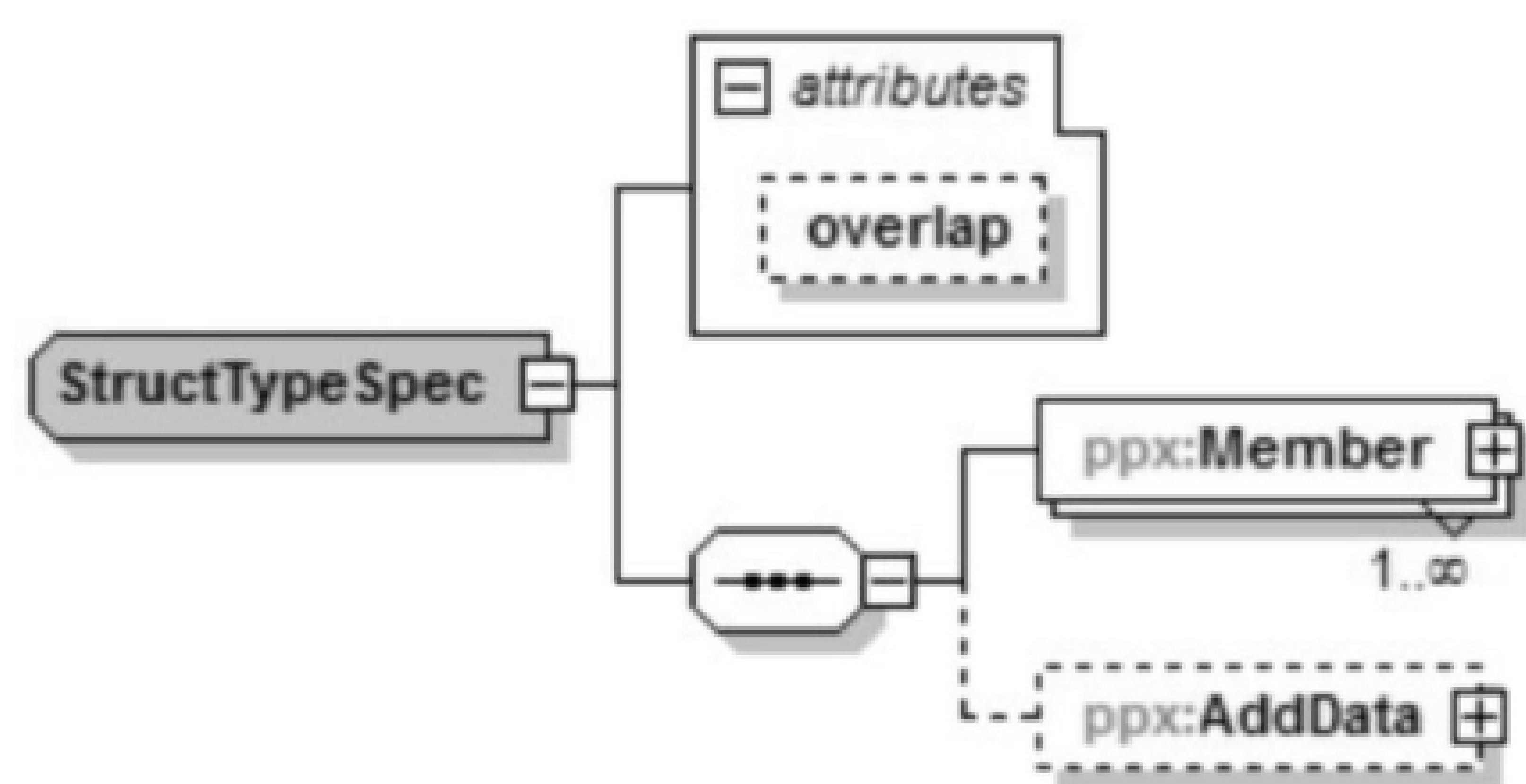


图 34 复合类型“StructTypeSpec”

属性“overlap”映射到结构化数据类型声明中的 OVERLAP 关键字。“Member”所属类型“VariableDecl”将在 11.3 中介绍。

9.7 子范围数据类型“SubrangeTypeSpec”

复合类型“SubrangeTypeSpec”表示 IEC 61131-3 中定义的限定子范围的数据类型。其内容如图 35 所示。

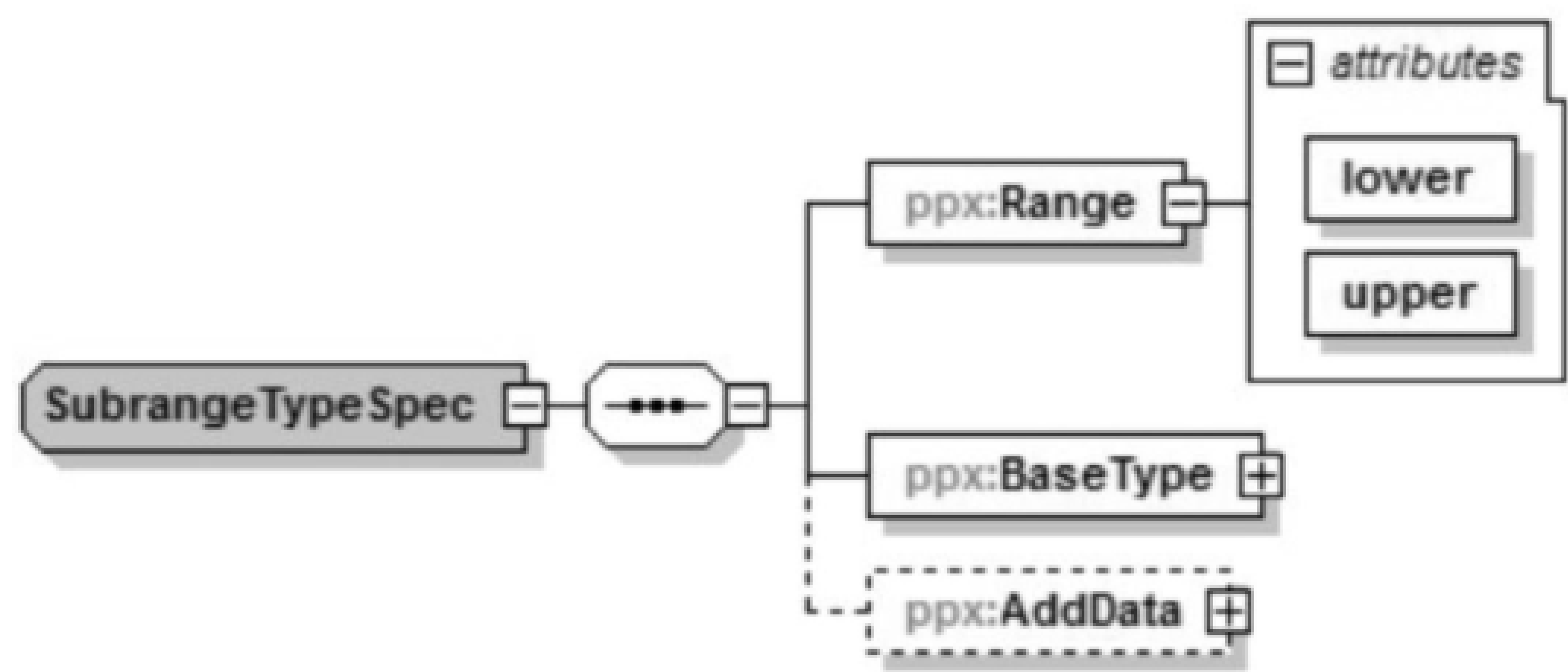


图 35 复合类型“SubrangeTypeSpec”

元素“Range”的属性“lower”和“upper”用于指定子范围数据类型的上下限。  
“BaseType”所属类型“TypeRef”将在 11.5 中介绍。

9.8 引用类型“ReferenceTypeSpec”

复合类型“ReferenceTypeSpec”表示 IEC 61131-3 中定义的引用类型。其内容如图 36 所示。

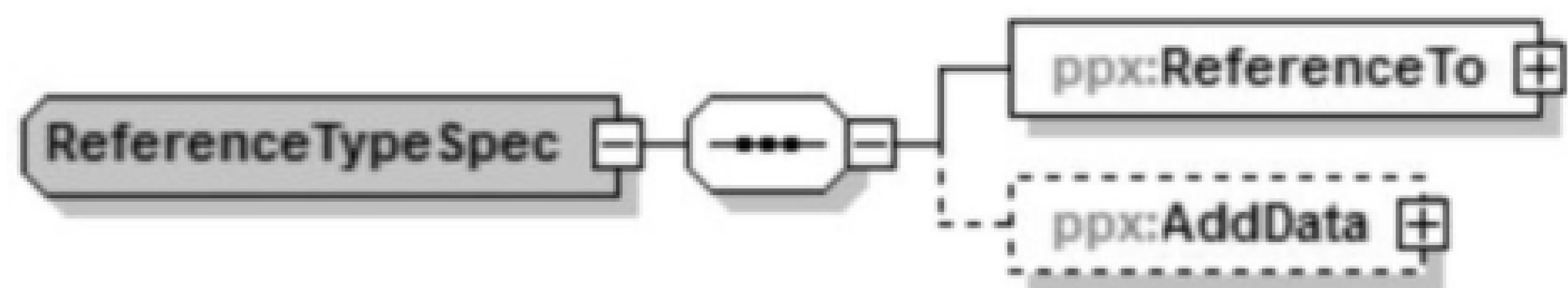


图 36 复合类型 “ReferenceTypeSpec”

“ReferenceTo”所属类型“TypeRef”将在 11.5 中介绍。

9.9 基本数据类型“ElementaryType”

简单类型“ElementaryType”表示 IEC 61131-3 中定义的所有基本数据类型。“ElementaryType”为 string 类型并且其值严格遵守 IEC 61131-3 对基本数据类型名称的定义。

10 程序组织单元声明

10.1 POU 声明“PouDecl”

“PouDecl”组表示在 IEC 61131-3 中定义的不同 POU 之间的选择。其内容如图 37 所示。

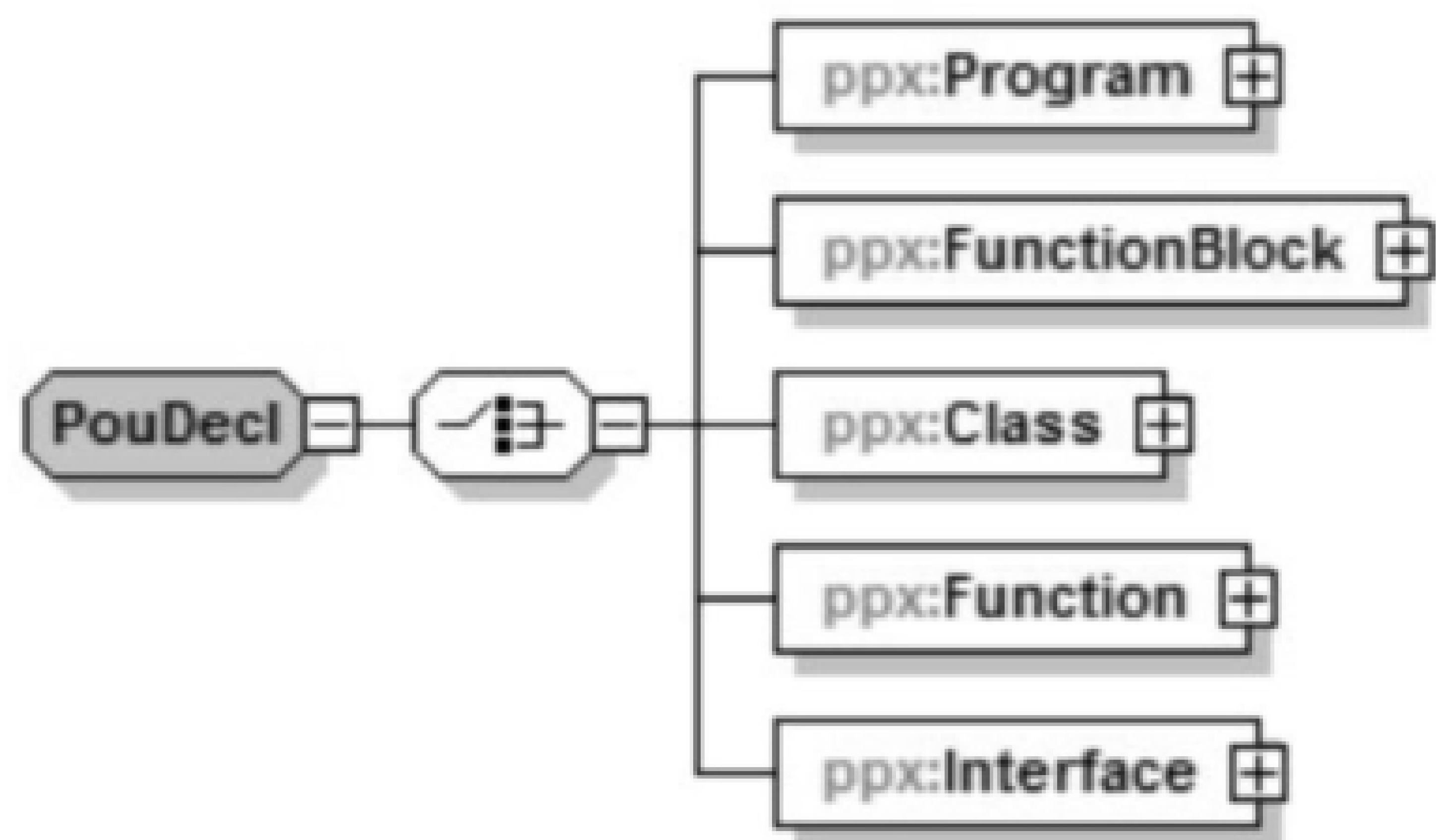


图 37 复合类型“PouDecl”

“Program”所属类型“Program”、“FunctionBlock”所属类型“FunctionBlock”、“Class”所属类型“Class”、“Function”所属类型“Function”和“Interface”所属类型“Interface”将分别在 10.2、10.3、10.4、10.5 和 10.6 中介绍。

10.2 程序“Program”

复合类型“Program”表示 IEC 61131-3 中定义的程序。其内容如图 38 所示。

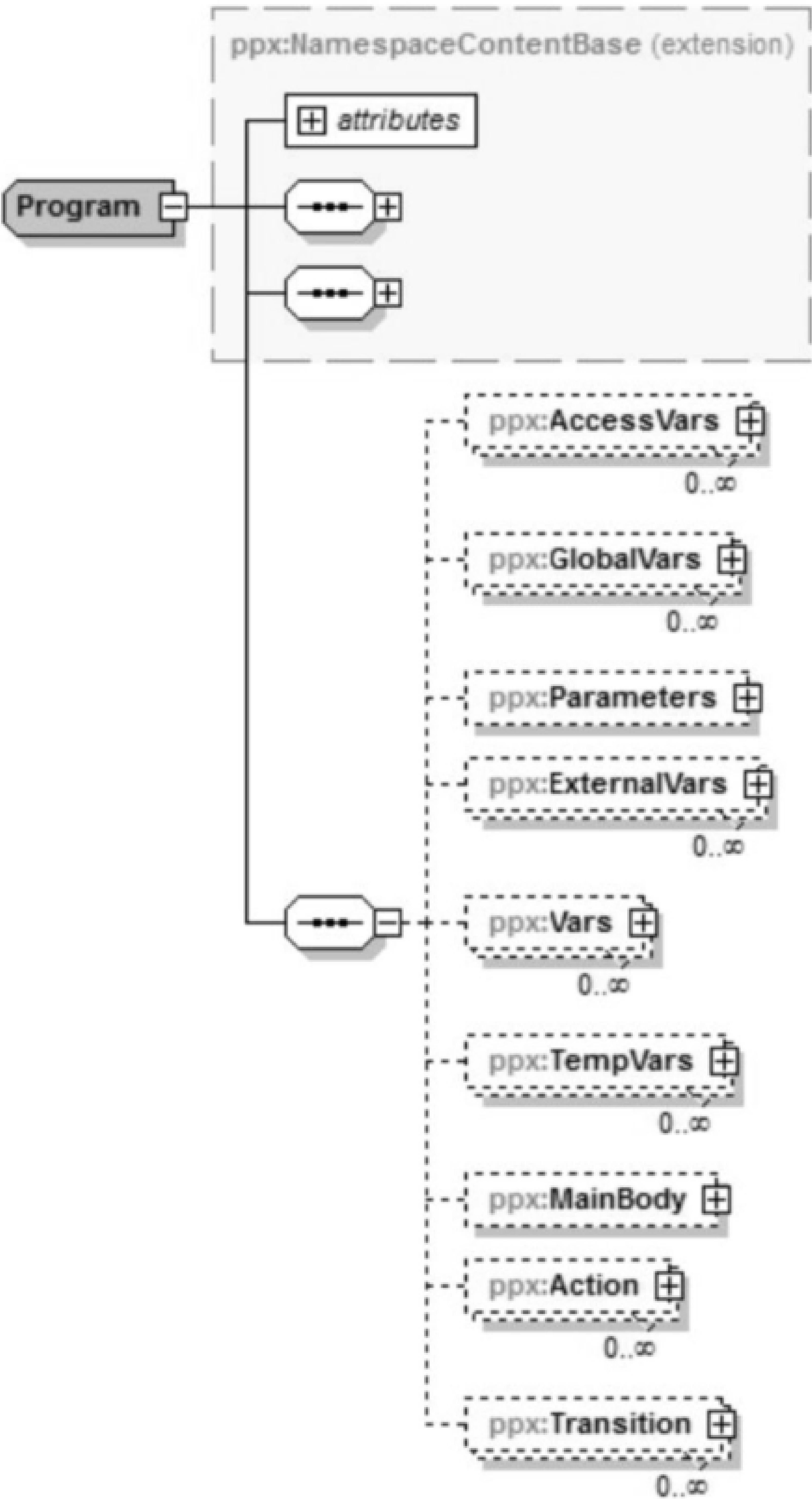


图 38 复合类型“Program”

该复合类型扩展了 7.5.3 中介绍的基本复合类型“NamespaceContentBase”。

“AccessVars”和“GlobalVars”所属类型“VarList”将在 11.1 中介绍。

“Parameters”所属类型“ParameterSet”将在 10.11 中介绍。

“ExternalVars”所属类型“ExternalVarList”和“Vars”所属类型“VarListWithAccessSpec”将在 11.2 中介绍。

“TempVars”所属类型“VarList”将在 11.1 中介绍。

“MainBody”所属类型“Body”将在 10.14 中介绍。

“Action”所属类型“Action”和“Transition”所属类型“NamedTransition”将分别在 10.7 和 10.8 中介绍。

10.3 功能块“FunctionBlock”

复合类型“FunctionBlock”表示 IEC 61131-3 中定义的功能块。其内容如图 39 所示。

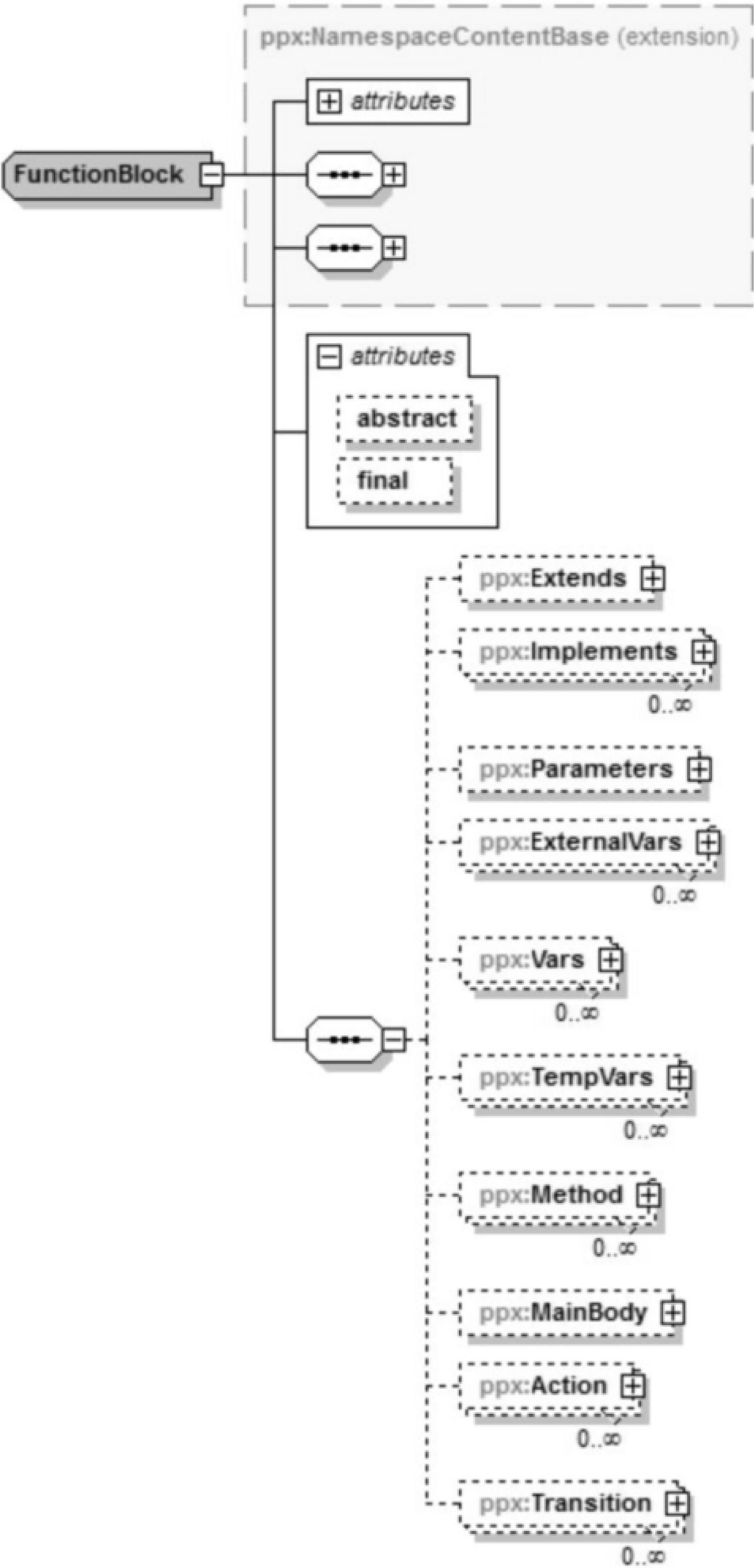


图 39 复合类型“FunctionBlock”

该复合类型扩展了 7.5.3 中介绍的基本复合类型“NamespaceContentBase”。

属性“abstract”和“final”表示是否使用了 IEC 61131-3 中定义的关键字 ABSTRACT 和 FINAL 定义功能块。

“Extends”包含基类或功能块的名称(如适用),“Implements”包含已实现的接口(如适用),二者都属于将在 11.5 中介绍的“TypeRef”类型。

“Parameters”所属类型“ParameterSet”将在 10.11 中介绍。

“ExternalVars”所属类型“ExternalVarList”将在 11.2 中介绍。

“Vars”所属类型“VarListWithAccessSpec”将在 10.12 中介绍。

“TempVars”所属类型“VarList”将在 11.1 中介绍。

“Method”所属类型“Method”将在 10.10 中介绍。

“MainBody”所属类型“Body”将在 10.14 中介绍。

“Action”所属类型“Action”和“Transition”所属类型“NamedTransition”将分别在 10.7 和 10.8 中介绍。

10.4 类“Class”

复合类型“Class”表示 IEC 61131-3 中定义的类。其内容如图 40 所示。

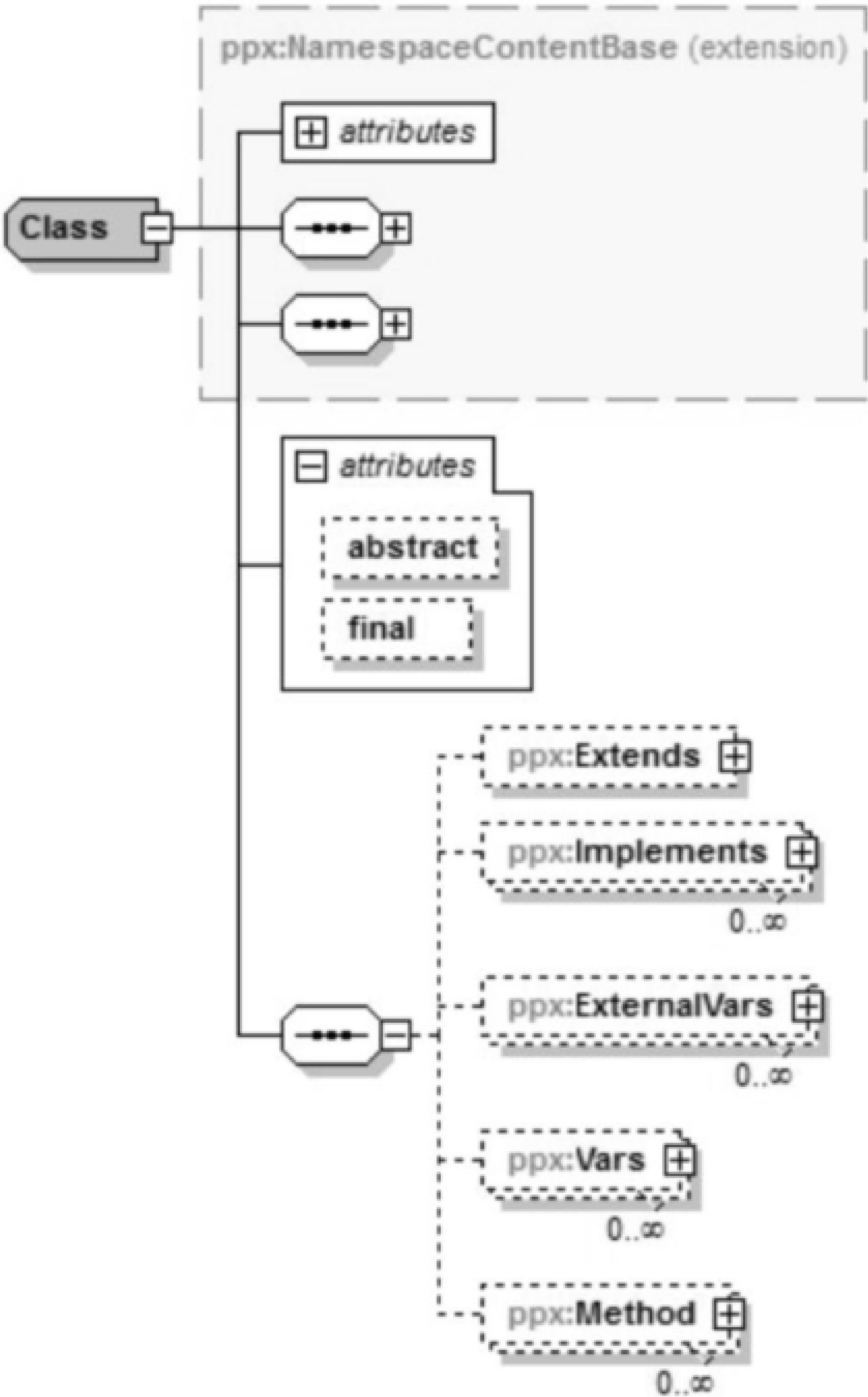


图 40 复合类型“Class”



该复合类型扩展了 7.5.3 中介绍的基本复合类型“NamespaceContentBase”。

属性“abstract”和“final”表示是否使用了 IEC 61131-3 中定义的关键字 ABSTRACT 和 FINAL 定义类。

“Extends”包含基类的名称(如适用),“Implements”包含已实现的接口(如适用),二者都属于将在 11.5 中介绍的“TypeRef”类型。

“ExternalVars”所属类型“ExternalVarList”将在 11.2 中介绍。

“Vars”所属类型“VarListWithAccessSpec”将在 10.12 中介绍。

“Method”所属类型“Method”将在 10.10 中介绍。

10.5 功能“Function”

复合类型“Function”表示 IEC 61131-3 中定义的功能。其内容如图 41 所示。

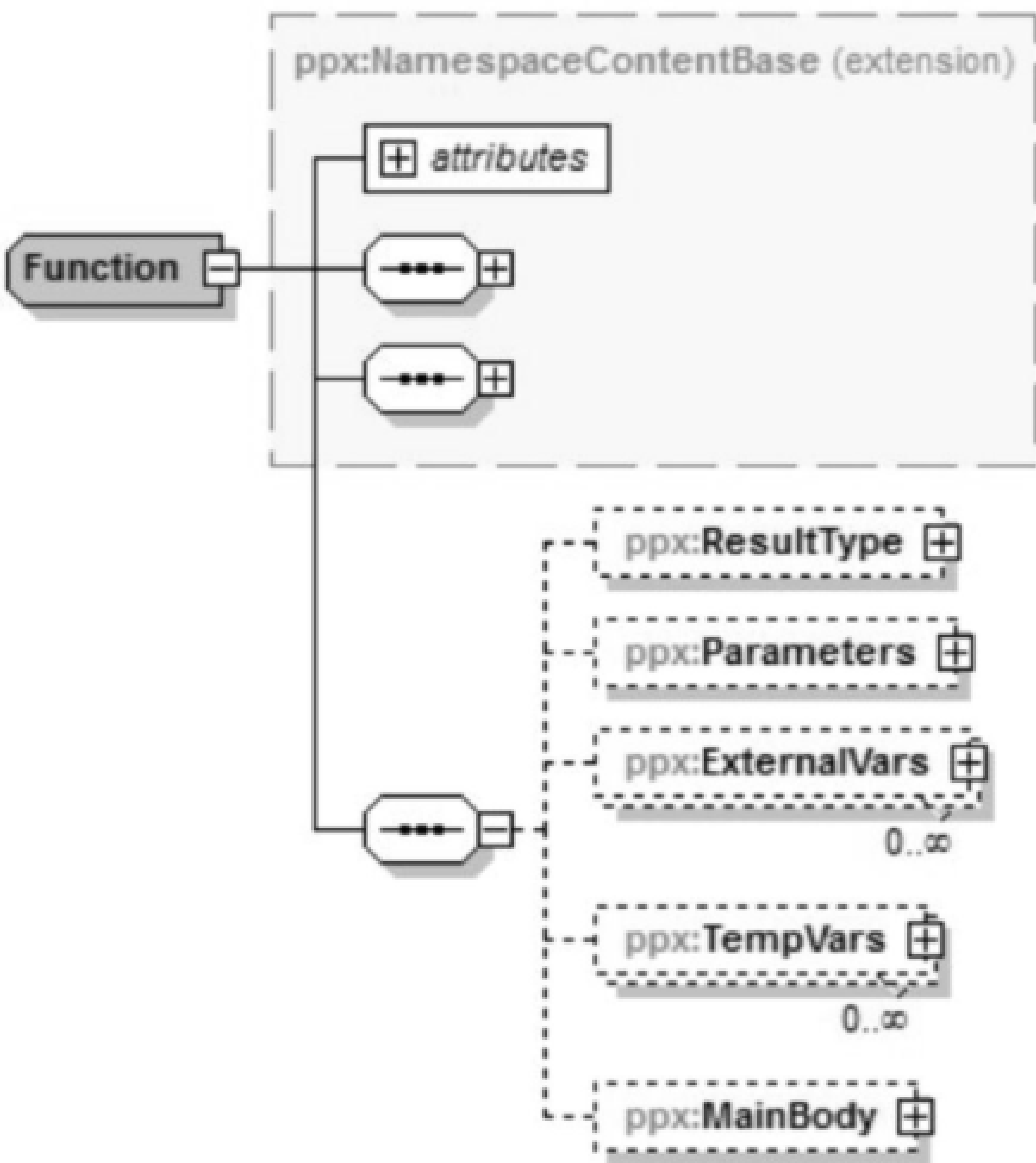


图 41 复合类型“Function”

该复合类型扩展了 7.5.3 中介绍的基本复合类型“NamespaceContentBase”。

“ResultType”包含功能所要传递的数据类型的名称。“ResultType”所属类型“TypeRef”将在 11.5 中介绍。

“Parameters”所属类型“ParameterSet”将在 10.11 中介绍。

“ExternalVars”所属类型“ExternalVarList”将在 11.2 中介绍。

“TempVars”所属类型“VarList”将在 11.1 中介绍。

“MainBody”所属类型“Body”将在 10.14 中介绍。

10.6 接口“Interface”

复合类型“Interface”表示 IEC 61131-3 中定义的接口。其内容如图 42 所示。

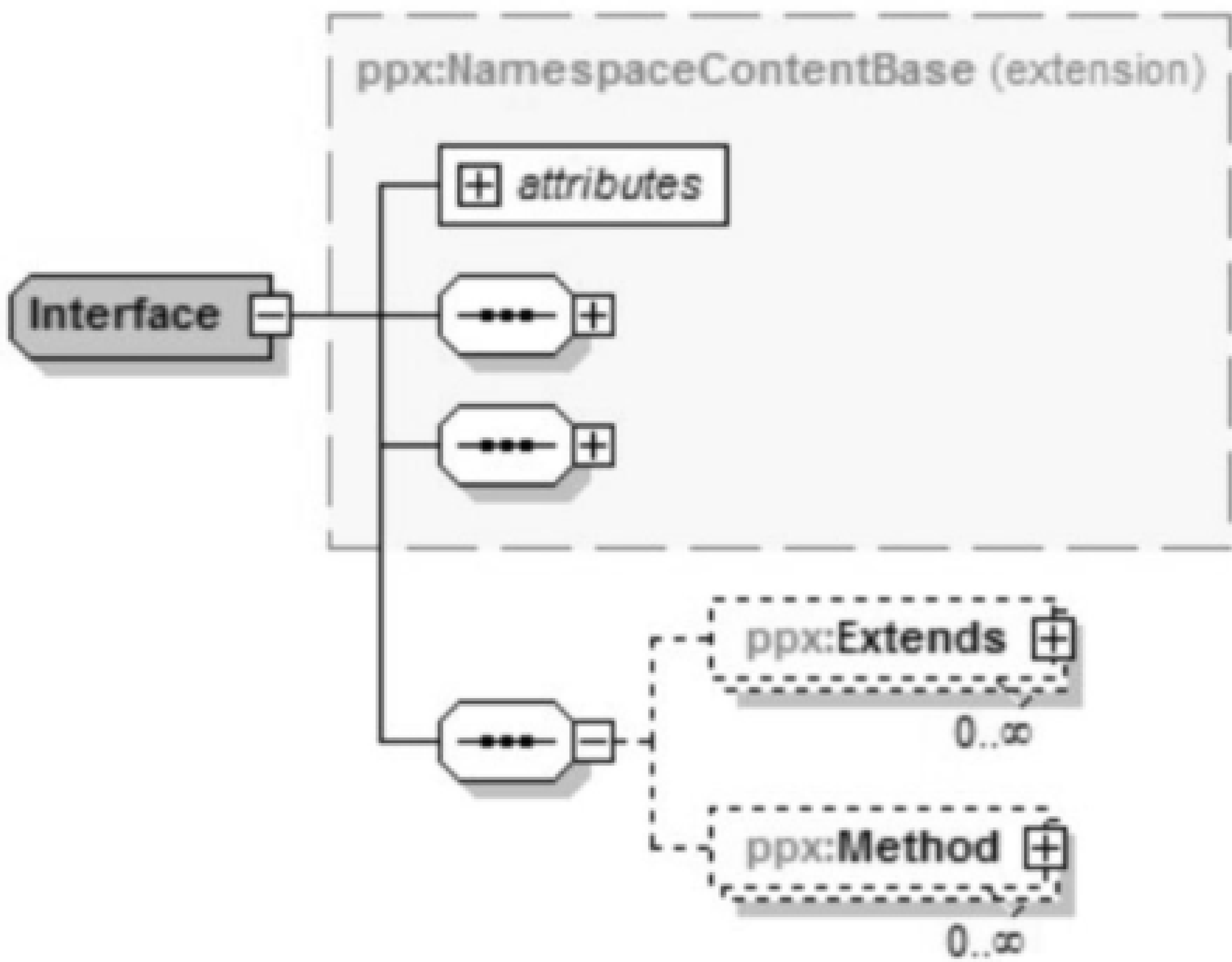


图 42 复合类型“Interface”

该复合类型扩展了 7.5.3 中介绍的基本复合类型“NamespaceContentBase”。

“Extends”包含基接口的名称(如适用),其所属类型“TypeRef”类型将在 11.5 中介绍。

“Method”所属类型“MethodPrototype”将在 10.9 中介绍。

10.7 动作“Action”

复合类型“Action”表示 IEC 61131-3 中定义的动作。其内容如图 43 所示。

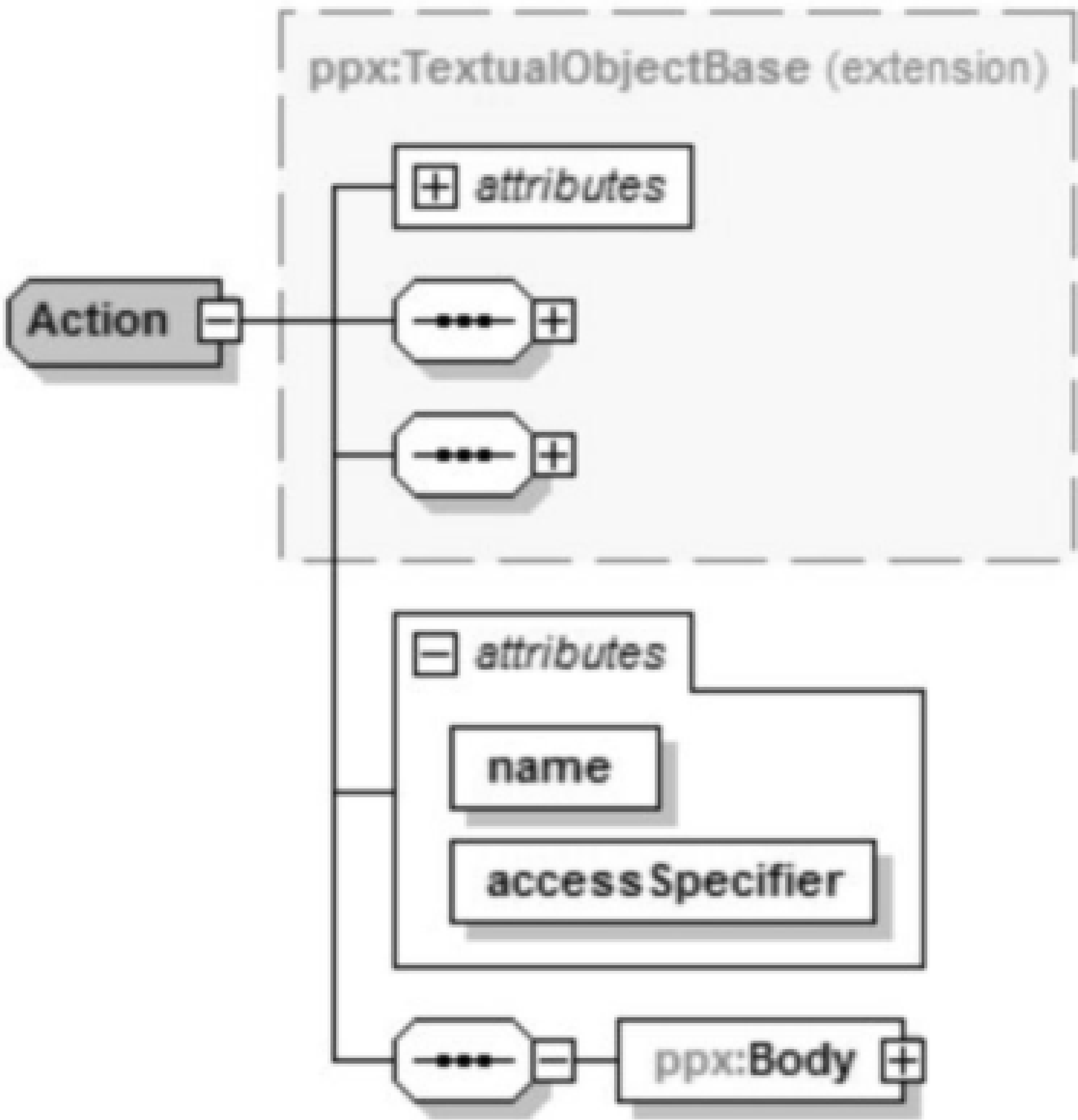


图 43 复合类型“Action”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。

属性“name”为 string 类型。

“Body”所属类型“Body”将在 10.14 中介绍。

10.8 带名称转换“NamedTransition”

复合类型“NamedTransition”表示 IEC 61131-3 中定义的有名称的转换。其内容如图 44 所示。

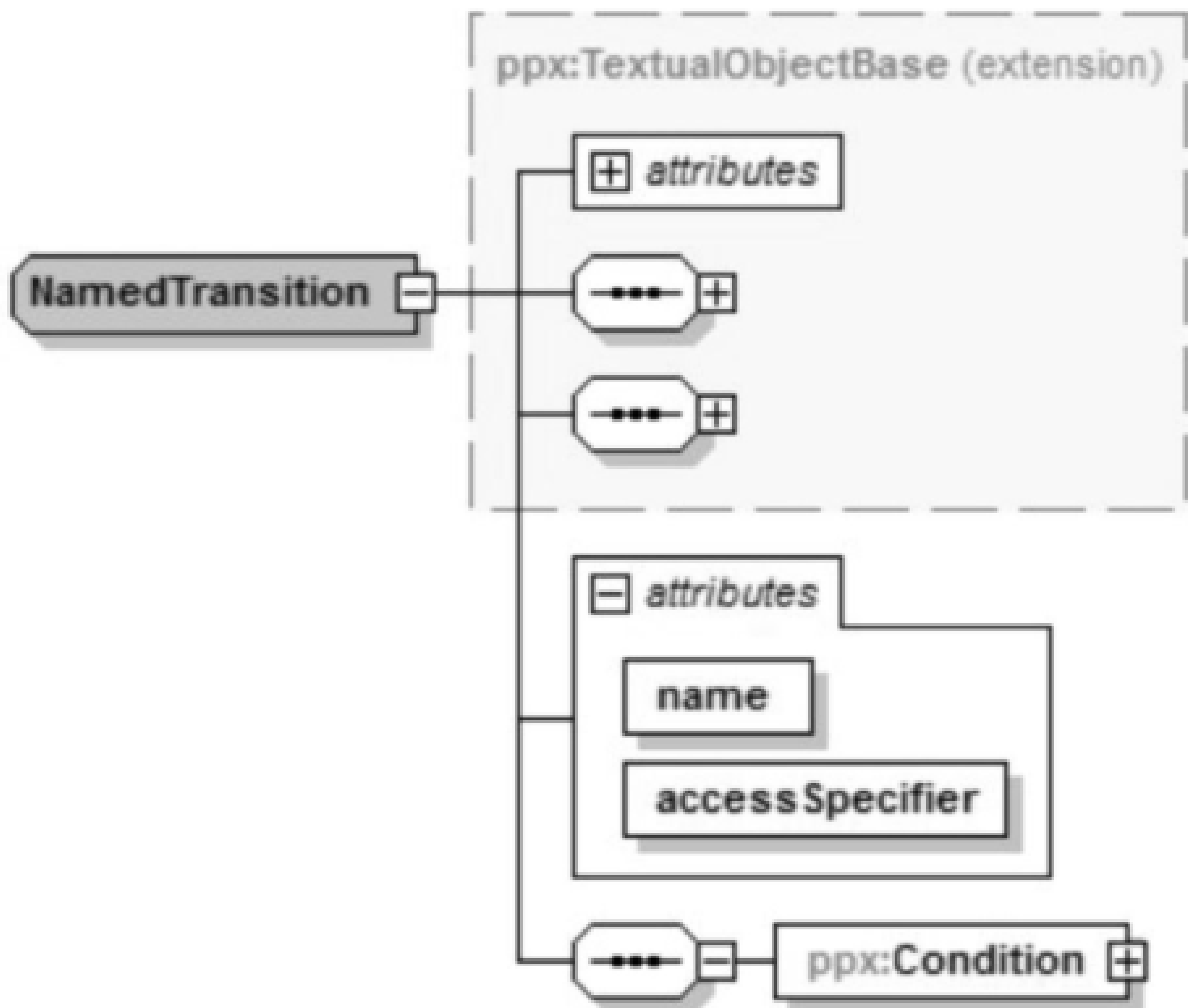


图 44 复合类型“NamedTransition”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。

属性“name”为 string 类型。

“Condition”所属类型“Predicate”将在 10.16 中介绍。

10.9 方法原型“MethodPrototype”

复合类型“MethodPrototype”表示 IEC 61131-3 中定义的方法原型。该 XML 元素可在 XML 元素“Interface”的上下文中使用,以定义方法的名称及其参数。其内容如图 45 所示。

注: IEC 61131-3 也将术语“方法原型”应用于抽象方法,其名称及参数分别在基类和基功能块中定义。在本文件中,能通过分别在“Class”或“FunctionBlock”的“Method”元素中使用 XML 属性“abstract”来表示此构造。

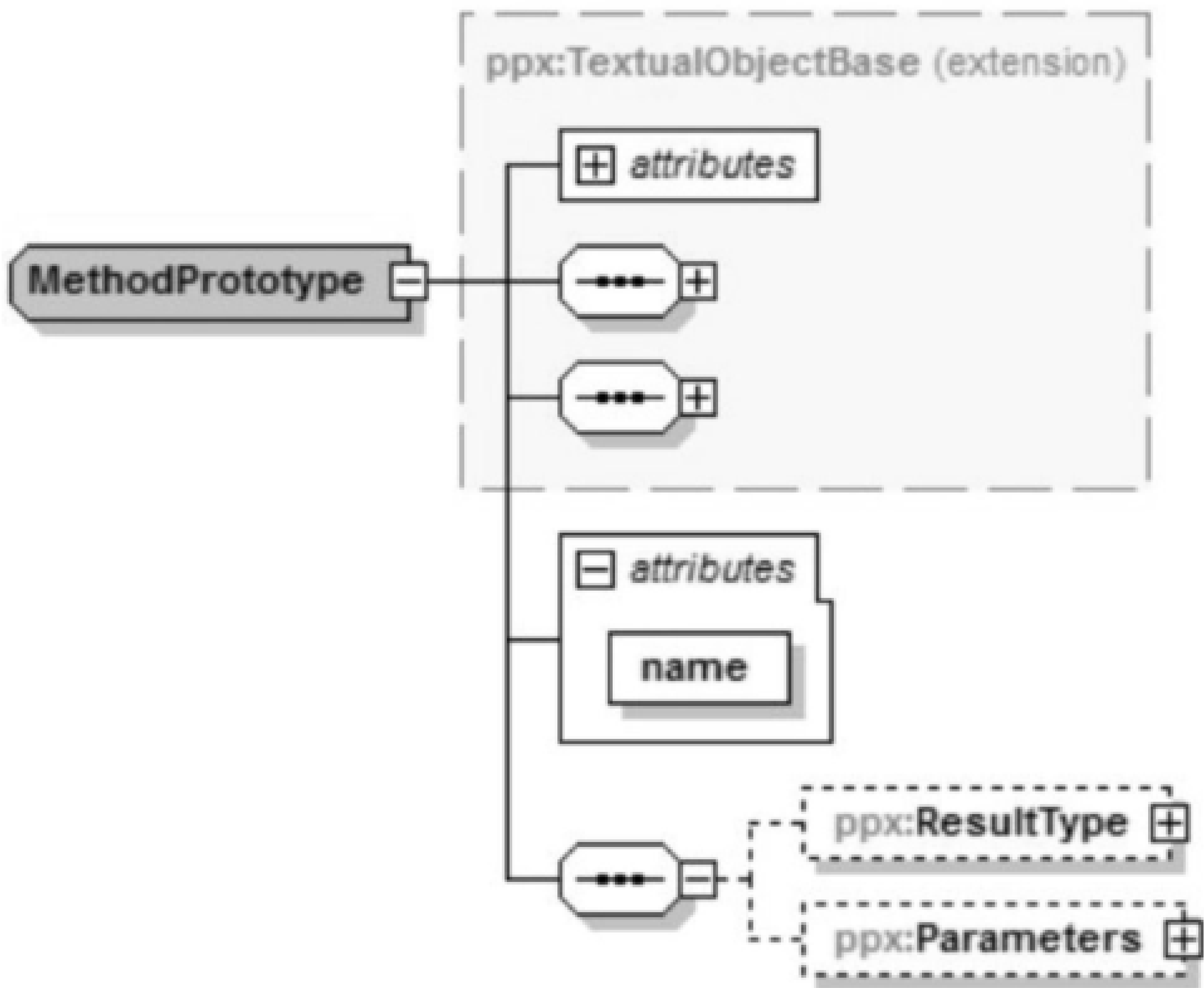


图 45 复合类型“MethodPrototype”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。  
属性“name”为 string 类型。  
“ResultType”包含方法所要传递的数据类型的名称,其所属类型“TypeRef”将在 11.5 中介绍。  
“Parameters”所属类型“ParameterSet”将在 10.11 中介绍。

10.10 方法“Method”

复合类型“Method”表示 IEC 61131-3 中定义的方法。其内容如图 46 所示。

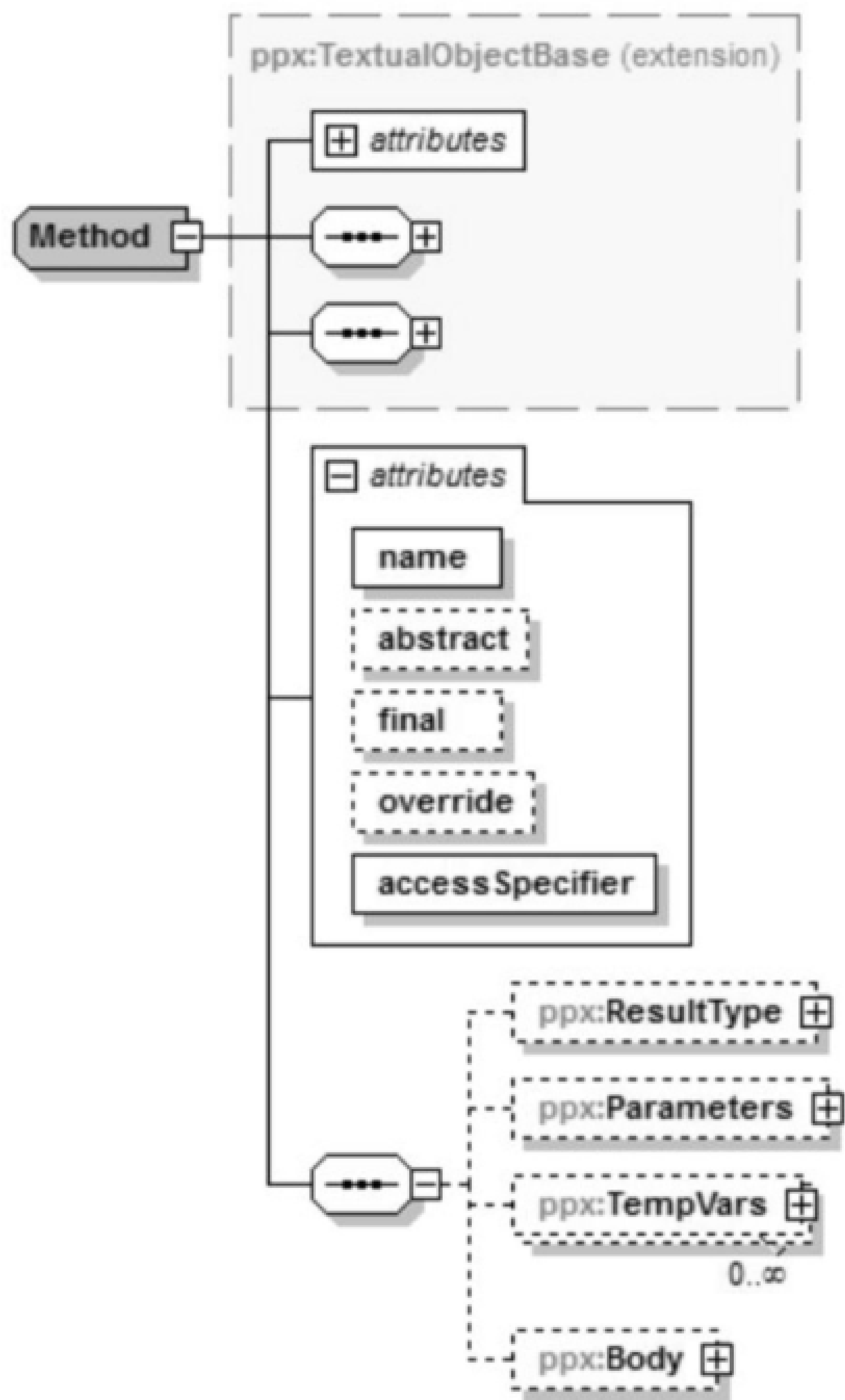


图 46 复合类型“Method”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。  
属性“name”为 string 类型。  
属性“abstract”“final”和“override”表示是否使用了 IEC 61131-3 中定义的关键字 ABSTRACT、FINAL 和 OVERRIDE 定义方法。若“abstract”为“true”,则“Body”应省略。  
属性“accessSpecifier”所属类型“AccessSpecifiers”将在 10.13 中介绍。  
“ResultType”包含方法所要传递的数据类型的名称。“ResultType”所属类型“TypeRef”将在 11.5 中介绍。  
“Parameters”所属类型“ParameterSet”将在 10.11 中介绍。  
“TempVars”所属类型“VarList”将在 11.1 中介绍。

“Body”所属类型“Body”将在 10.14 中介绍。

### 10.11 参数集“ParameterSet”

复合类型“ParameterSet”表示 IEC 61131-3 中定义的用于功能、功能块、程序、方法原型或方法的参数,使用关键字 VAR\_INPUT、VAR\_OUTPUT 和 VAR\_IN\_OUT 定义。其内容如图 47 所示。

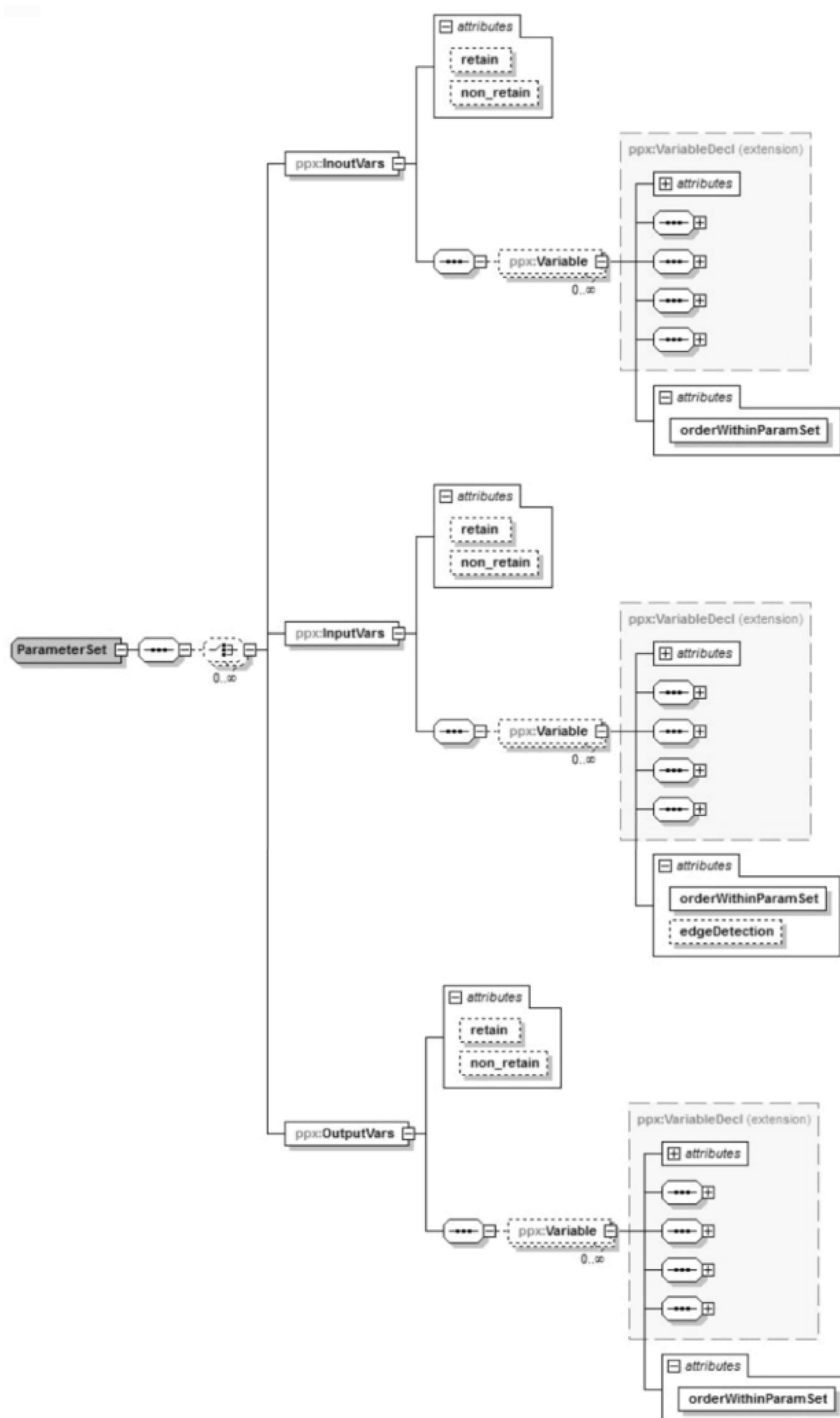


图 47 复合类型“ParameterSet”

“InputVars”和“OutputVars”具有布尔型属性“retain”和“non\_retain”。根据 IEC 61131-3, RETAIN 和 NON\_RETAIN 仅可用于功能块和程序的参数。

“InOutVars”“InputVars”和“OutputVars”都是“Variable”。“Variable”通过属性“orderWithinParamSet”扩展了将在 11.3 中介绍的“VariableDecl”类型,该属性指定了参数的顺序,此顺序在文本非正式调用中十分重要。每个“Variable”的此属性值在所属的“ParameterSet”中应唯一,并且应从 1 开始并递增 1。此外,在“InputVars”中,“Variable”通过“EdgeModifierType”类型的“edgeDetection”属性进行了扩展,该类型将在 15.5 中介绍。

10.12 带访问限定的变量列表“VarListWithAccessSpec”

复合类型“VarListWithAccessSpec”表示 IEC 61131-3 中定义的功能块、程序或类中的变量,可用访问说明符定义。其内容如图 48 所示。

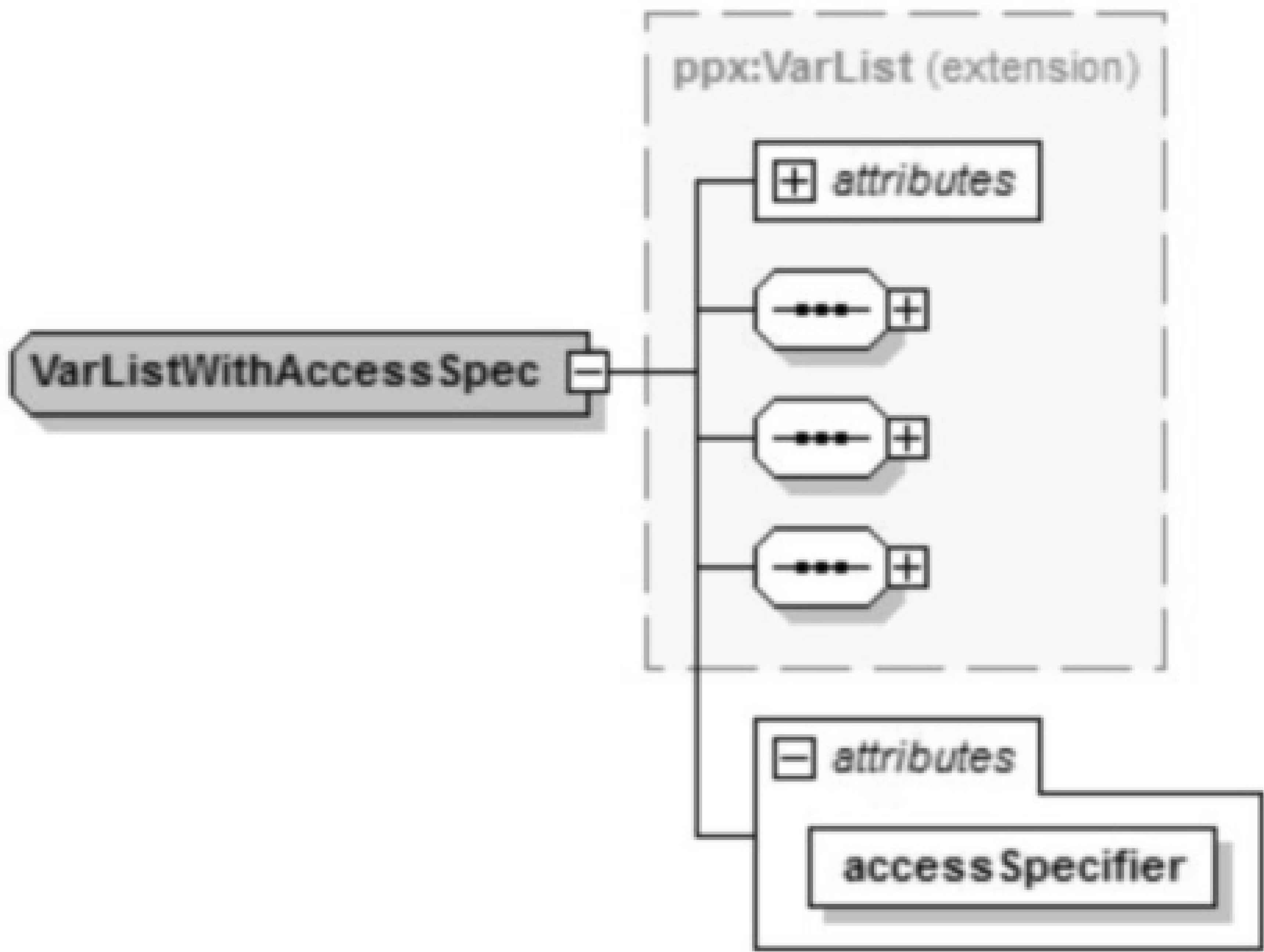


图 48 复合类型“VarListWithAccessSpec”

该复合类型扩展了将在 11.1 中介绍的基本复合类型“VarList”。  
属性“accessSpecifier”所属类型“AccessSpecifiers”将在 10.13 中介绍。

10.13 访问修饰符“AccessSpecifiers”

简单类型“AccessSpecifiers”为 string 类型,其值仅能为“private”“protected”“internal”和“public”。

10.14 主体“Body”

复合类型“Body”表示功能块、程序或动作中的代码主体。其内容如图 49 所示。

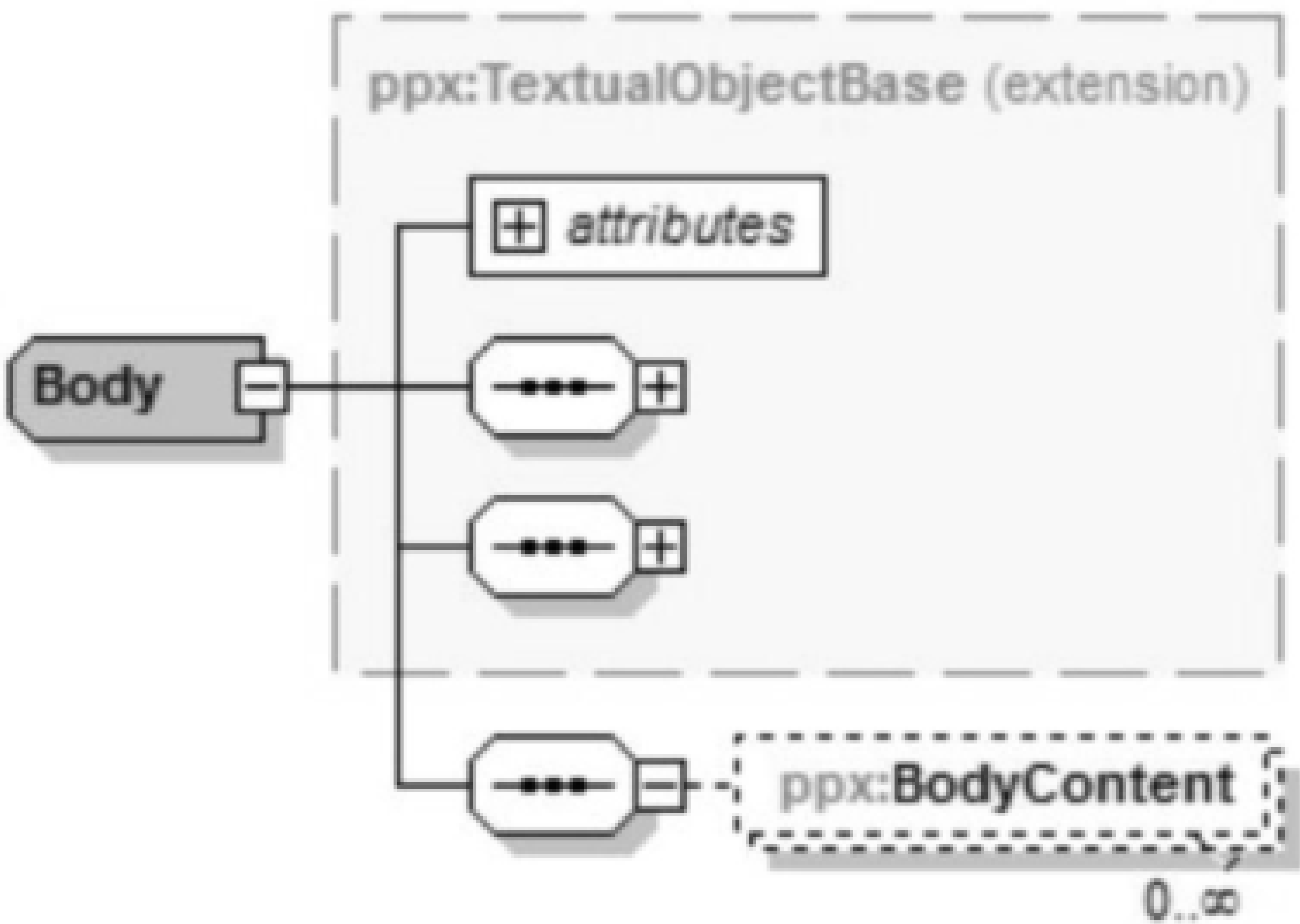


图 49 复合类型“Body”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。  
“BodyContent”所属类型“BehaviorRepresentationBase”已在 7.3.2 中介绍。

10.15 不带 SFC 的主体“BodyWithoutSFC”

复合类型“BodyWithoutSFC”表示功能或方法中的代码主体。其内容如图 50 所示。

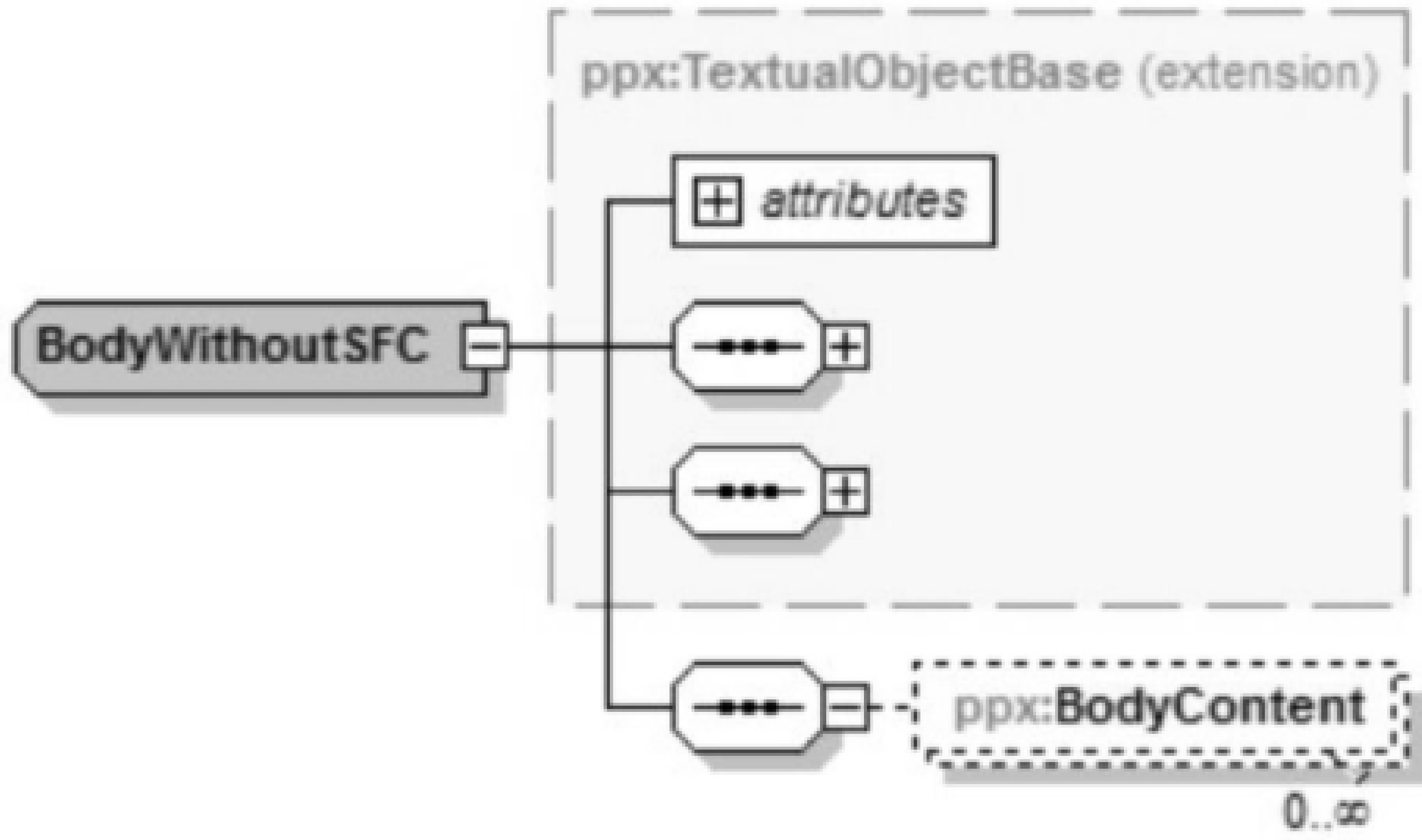


图 50 复合类型“BodyWithoutSFC”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。  
“BodyContent”所属类型“ProgrammingLanguageBase”已在 7.3.3 中介绍。

10.16 谓词“Predicate”

复合类型“Predicate”表示有名称或无名称转换的转换条件。其内容如图 51 所示。

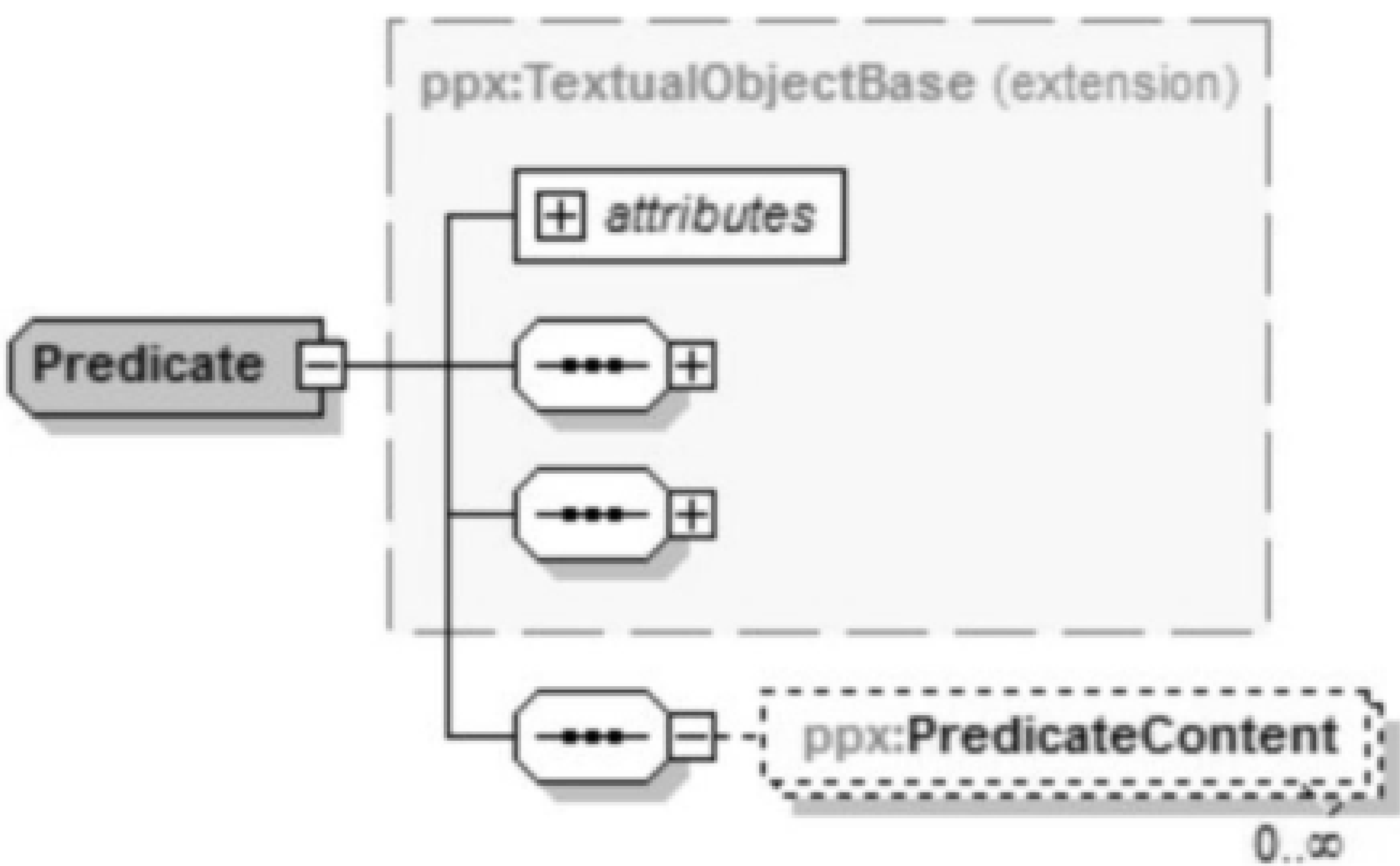


图 51 复合类型“Predicate”

该复合类型扩展了 7.5.2 中介绍的基本复合类型“TextualObjectBase”。

“PredicateContent”所属类型“ProgrammingLanguageBase”已在 7.3.3 中介绍。

11 变量声明

11.1 变量列表“VarList”

复合类型“VarList”表示 IEC 61131-3 中定义的变量列表。其内容如图 52 所示。

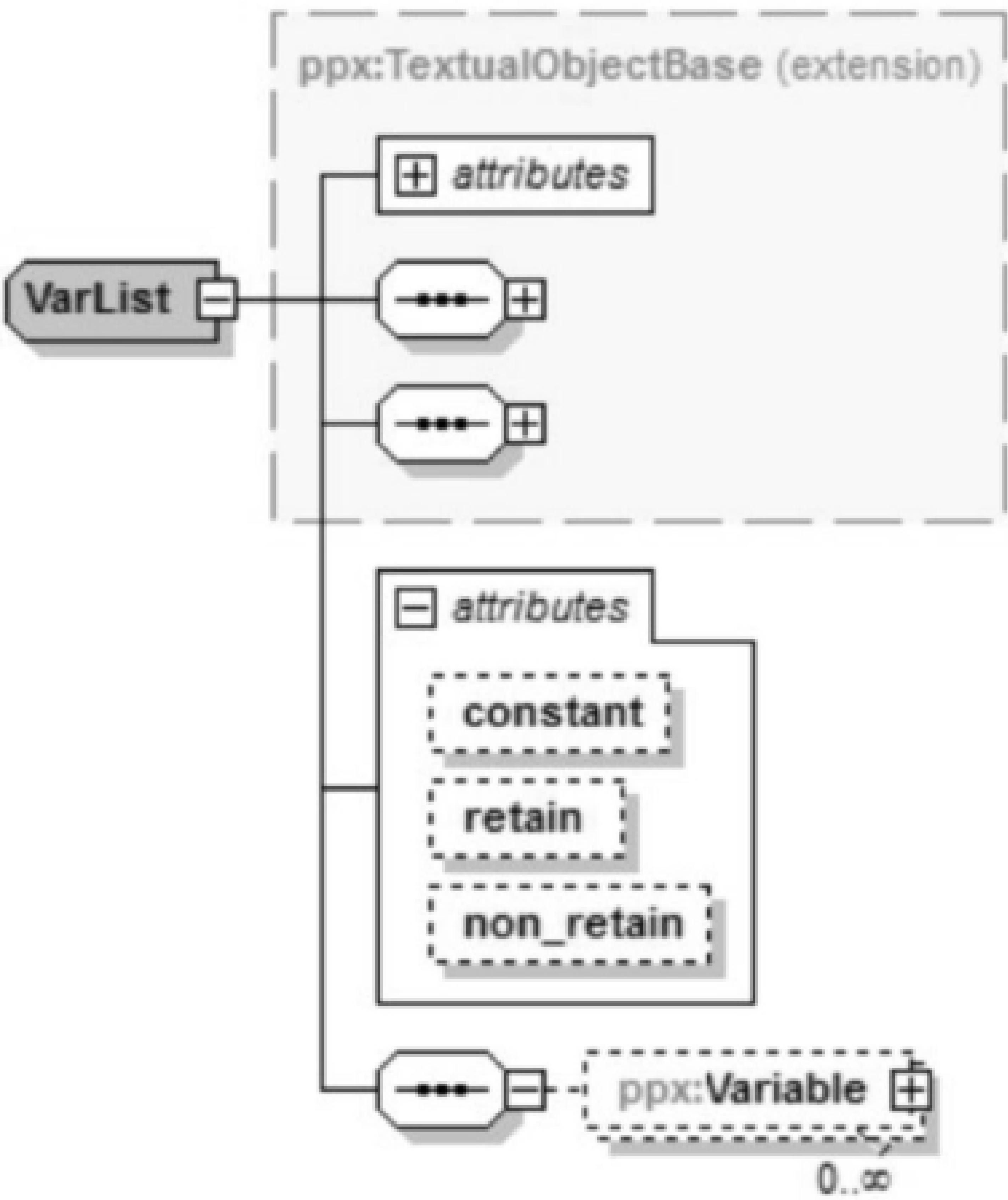


图 52 复合类型“VarList”

“VarList”基于抽象复合类型“TextualObjectBase”，该类型已在 7.5.2 中介绍。

该元素提供了“constant”“retain”和“non\_retain”属性。

“Variable”所属类型“VariableDecl”将在 11.3 中介绍。



11.2 外部变量列表“ExternalVarList”

复合类型“ExternalVarList”表示 IEC 61131-3 中定义的外部变量列表。其内容如图 53 所示。

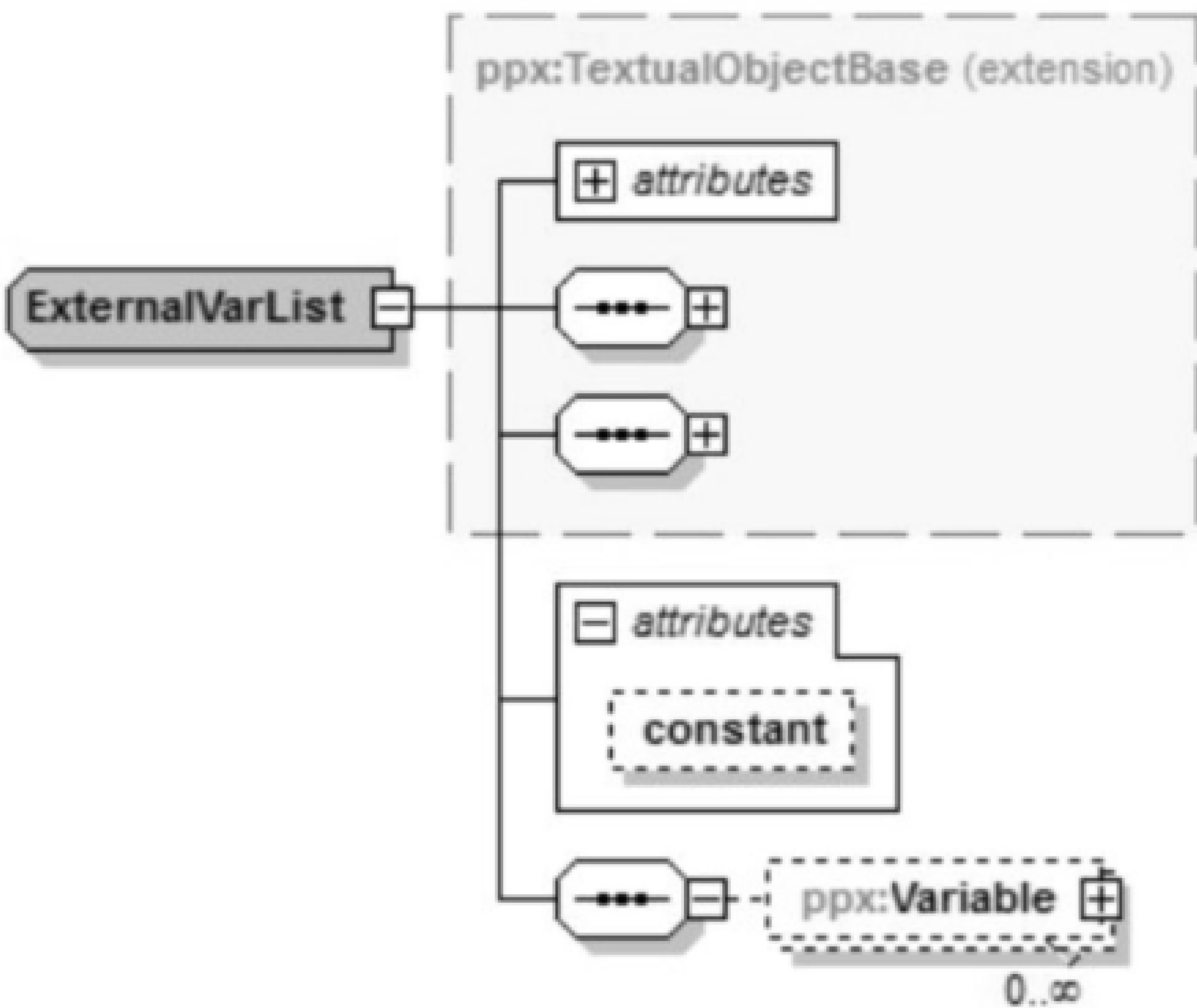


图 53 复合类型“ExternalVarList”

“ExternalVarList”基于抽象类型“TextualObjectBase”，该类型已在 7.5.2 中介绍。该元素提供了“constant”属性。

“Variable”所属类型“VariableDecl”将在 11.3 中介绍。

11.3 变量声明“VariableDecl”

复合类型“VariableDecl”表示 IEC 61131-3 中定义的变量属性。其内容如图 54 所示。

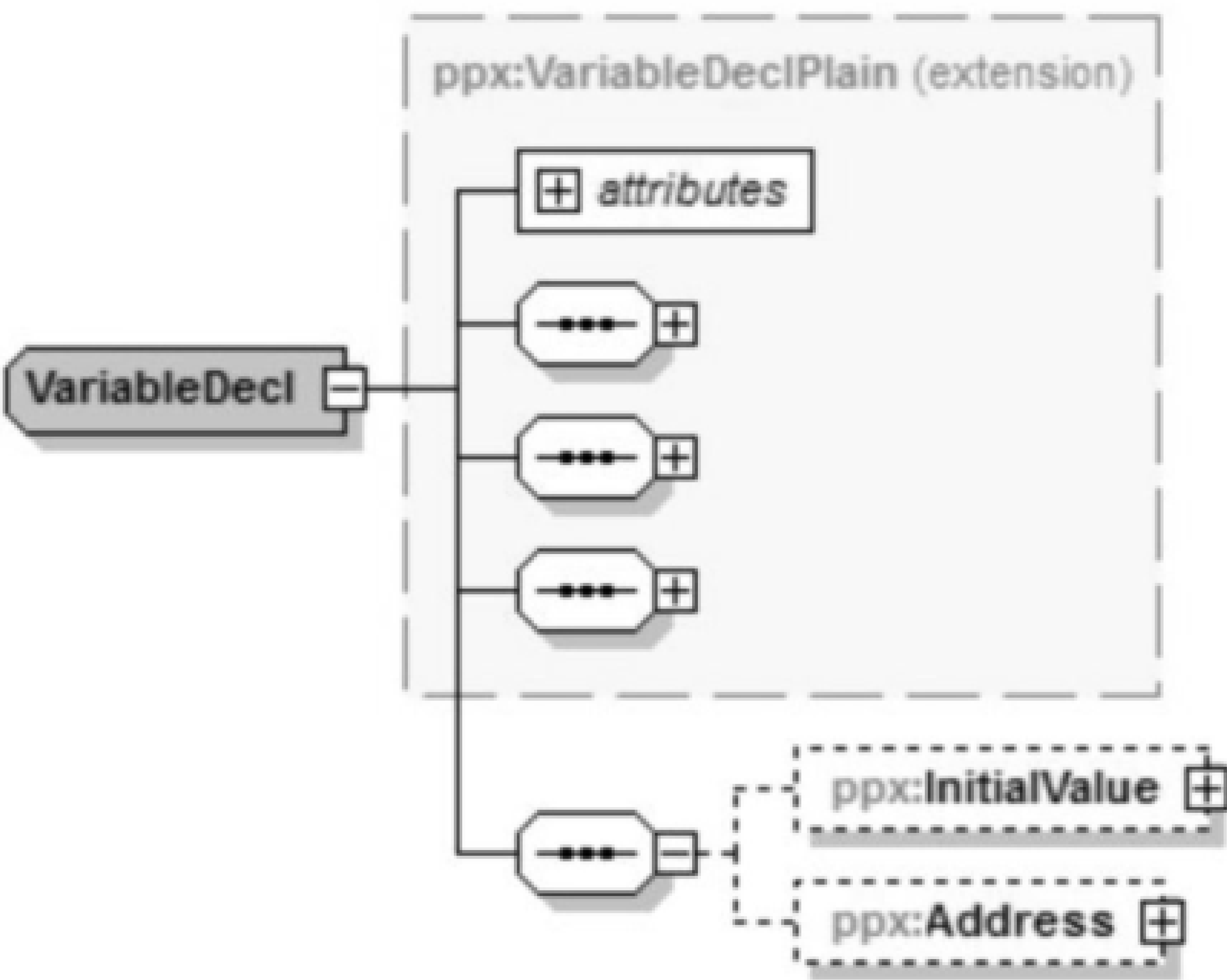


图 54 复合类型“VariableDecl”

“VariableDecl”基于“VariableDeclPlain”类型，该类型将在 11.4 中介绍。

“InitialValue”所属类型“Value”和“Address”所属类型“AddressExpression”将分别在 11.6 和 11.7

中介绍。

11.4 变量简单声明“VariableDeclPlain”

复合类型“VariableDeclPlain”表示 IEC 61131-3 中定义的变量简单声明。其内容如图 55 所示。

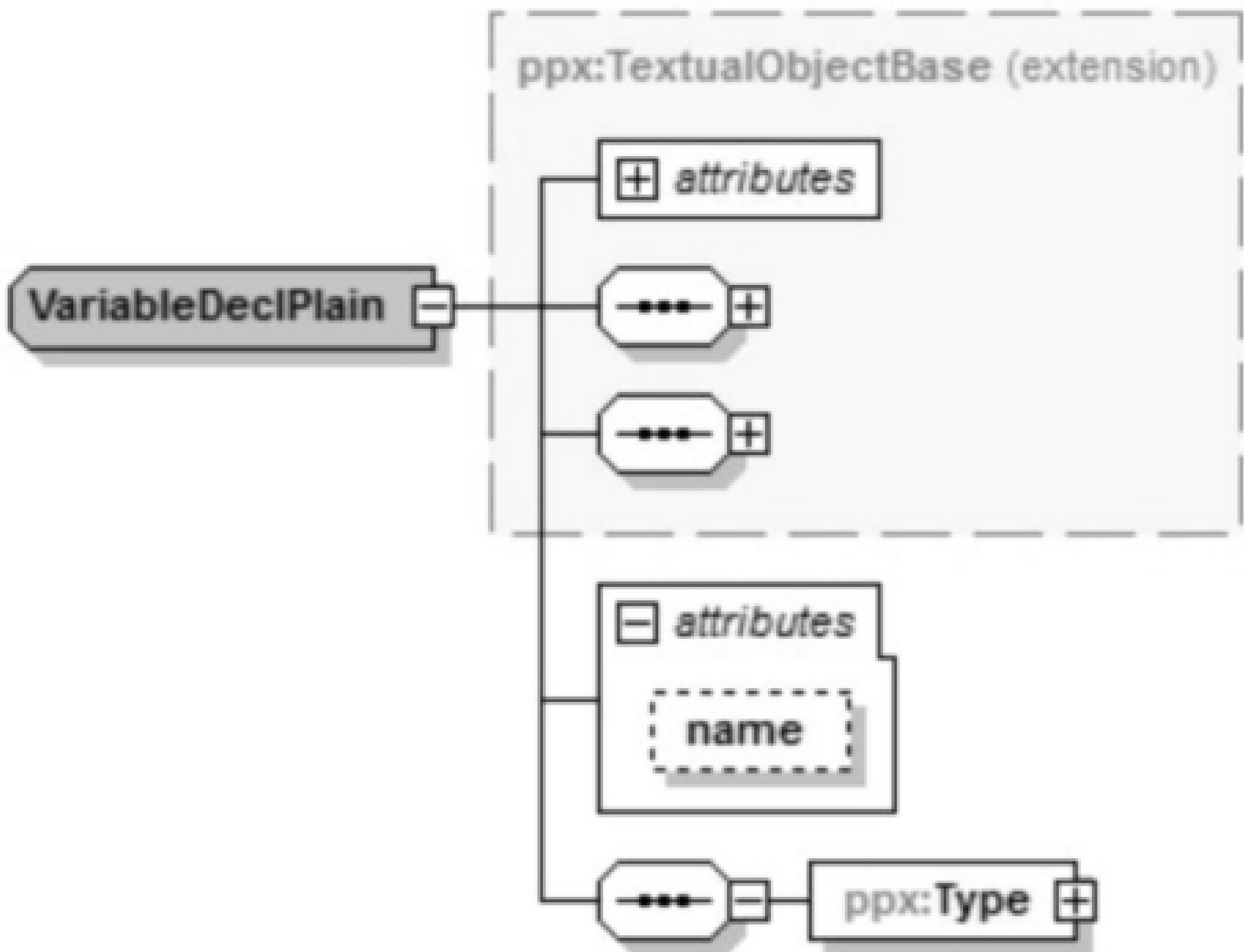


图 55 复合类型“VariableDeclPlain”

“VariableDeclPlain”基于抽象类型“TextualObjectBase”，该类型已在 7.5.2 中介绍。

强制属性“name”包含变量的名称。如果变量仅通过其地址指定（例如 IEC 61131-3 中的示例：“VAR AT%IB12 : REAL; END\_VAR”，不宜使用此方法），地址说明符也应用作其名称。

“Type”所属类型“TypeRef”将在 11.5 中介绍。

11.5 类型引用“TypeRef”

复合类型“TypeRef”表示 IEC 61131-3 中定义的变量类型声明。其内容如图 56 所示。

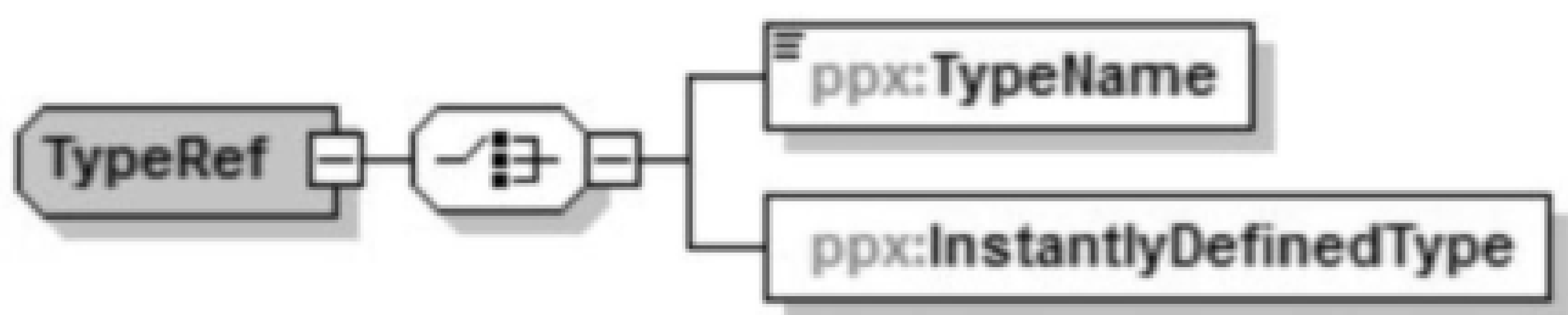


图 56 复合类型“TypeRef”

“TypeRef”可选择引用一个已定义类型的名称（string 类型的“TypeName”），或者定义一个匿名类型（7.2.3 中介绍的“InstantlyDefinableTypeSpecBase”类型的“InstantlyDefinedType”）。

11.6 值“Value”

复合类型“Value”表示变量的初始值。其内容如图 57 所示。

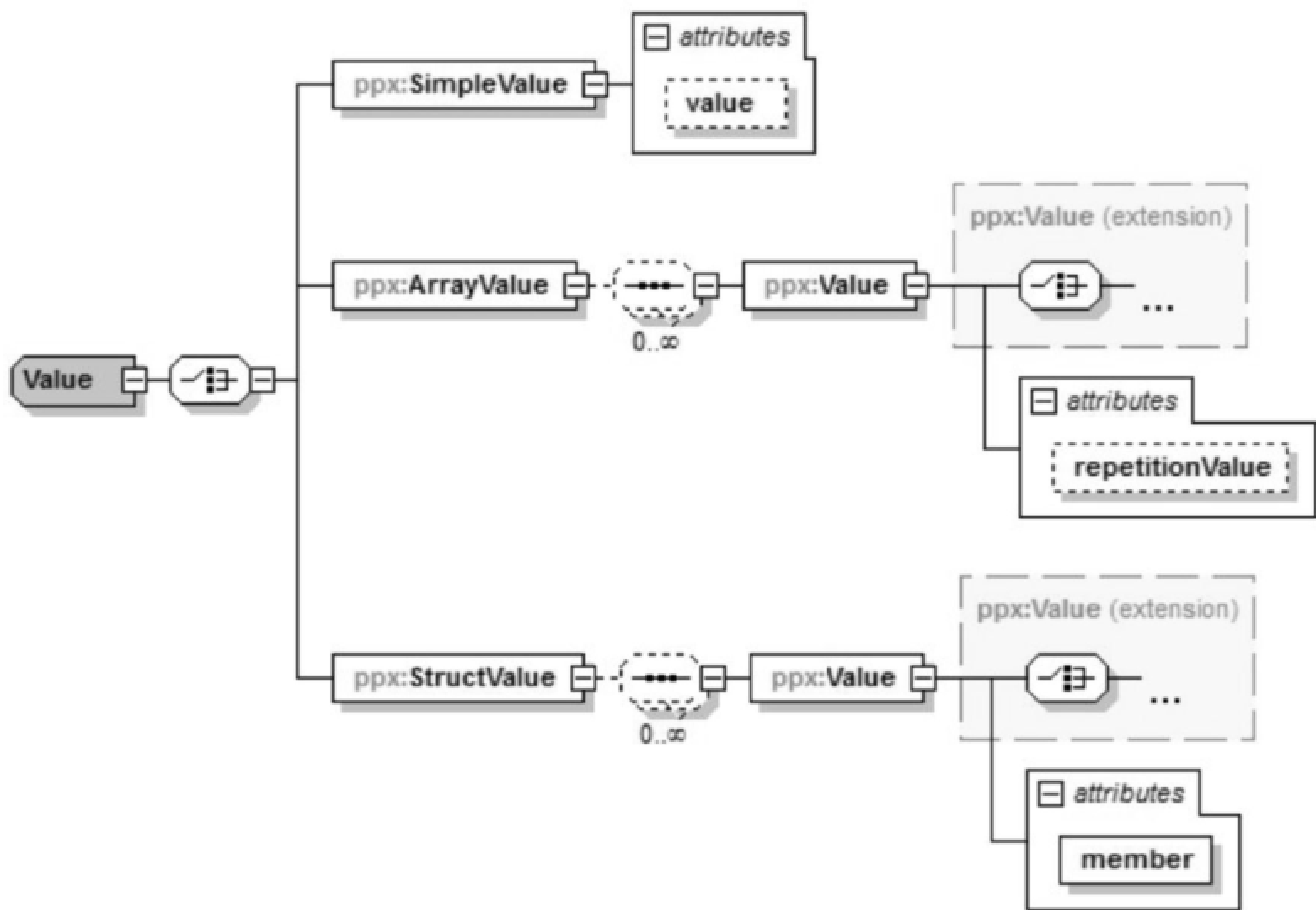


图 57 复合类型“Value”

“SimpleValue”提供了“value”属性，“ArrayValue”提供了“repetitionValue”属性，“StructValue”提供了“member”属性。值的初始化可递归定义，因此“Value”元素可包含在其自身中。

11.7 地址表达式“AddressExpression”

复合类型“AddressExpression”表示 IEC 61131-3 中定义的部分表达变量。其内容如图 58 所示。

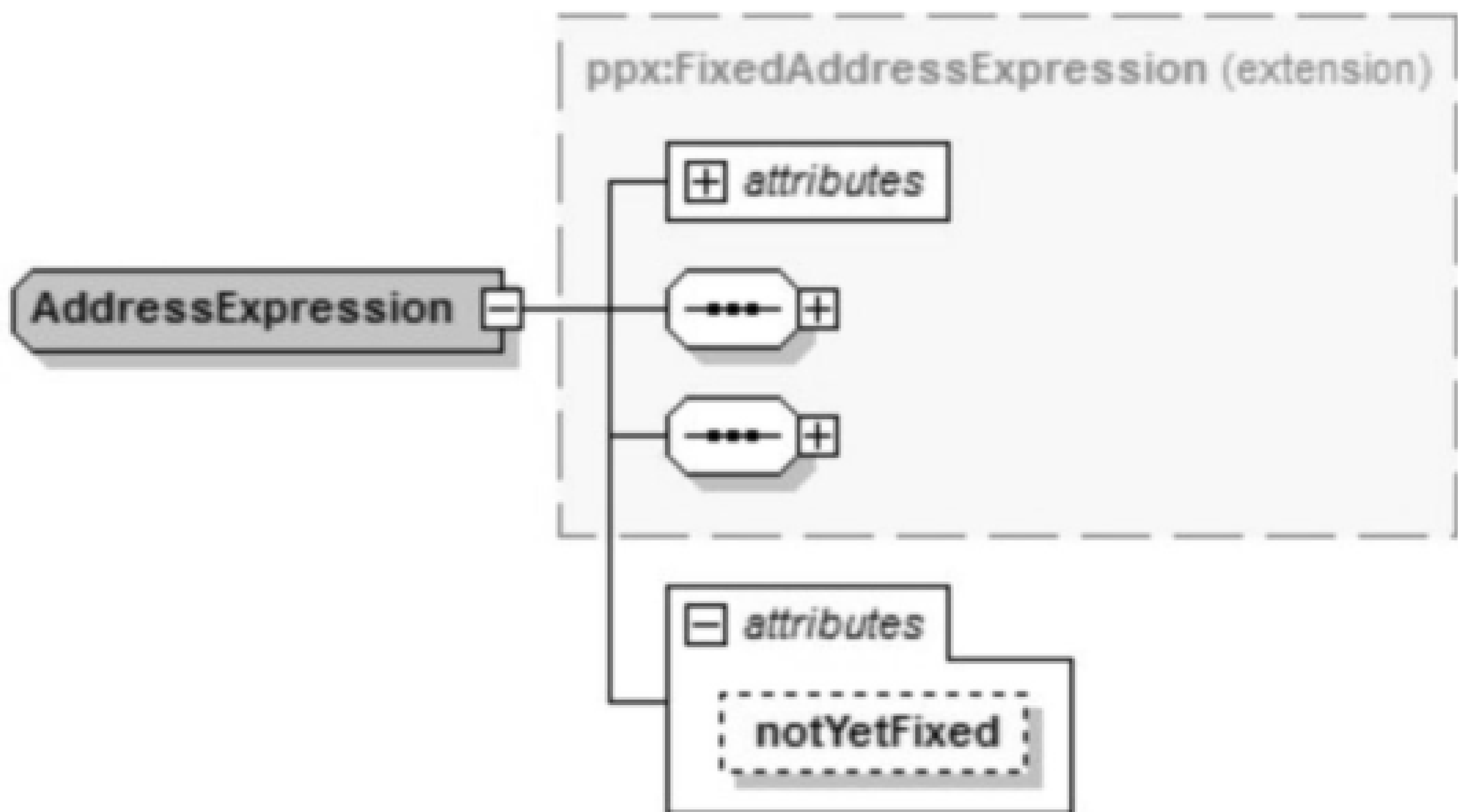


图 58 复合类型“AddressExpression”

“AddressExpression”基于抽象类型“FixedAddressExpression”，该类型将在 11.8 中介绍。该元素提供了“not YetFixed”属性。

11.8 直接表达变量“FixedAddressExpression”

复合类型“FixedAddressExpression”表示 IEC 61131-3 中定义的直接表达变量。其内容如图 59 所示。

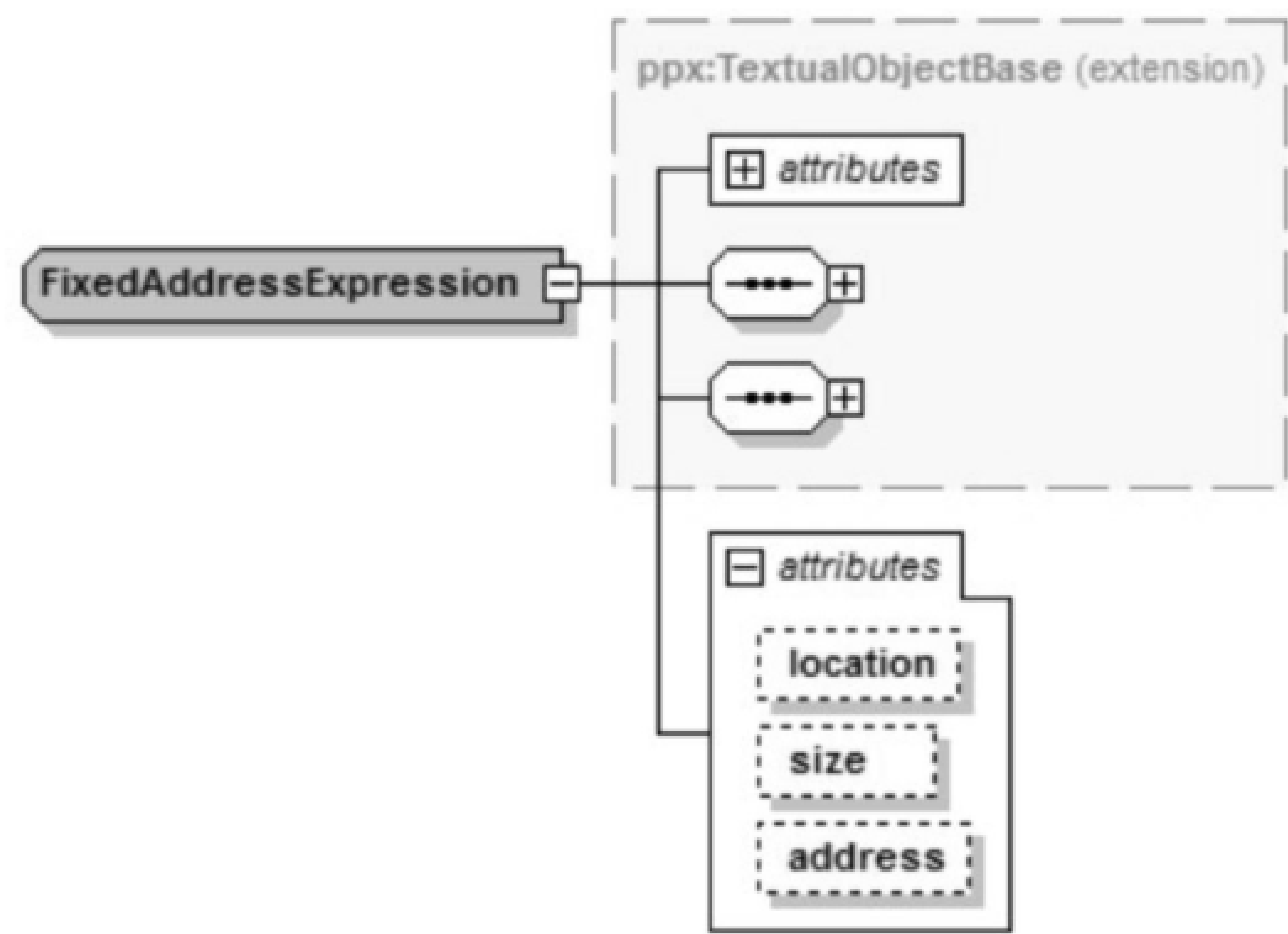


图 59 复合类型“FixedAddressExpression”

“FixedAddressExpression”基于抽象类型“TextualObjectBase”，该类型已在 7.5.2 中介绍。该元素提供了“location”“size”和“address”属性。

12 行为描述

12.1 指令表“IL”

复合类型“IL”表示指令表中的一段代码。其内容如图 60 所示。

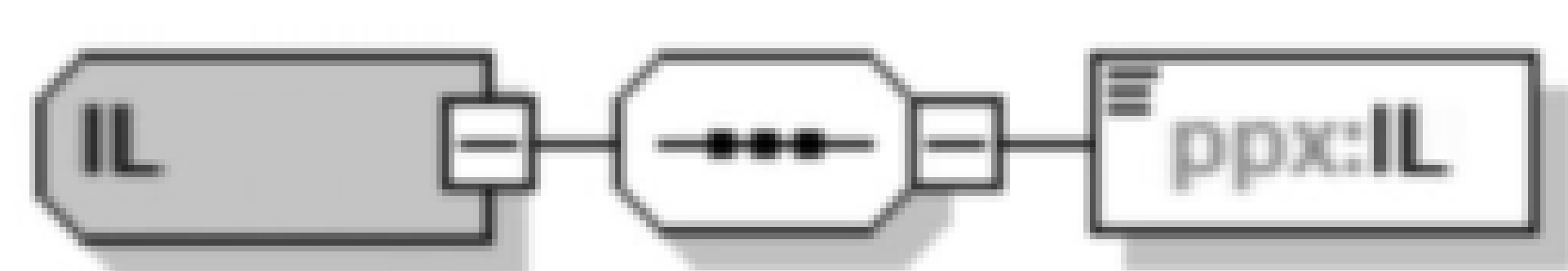


图 60 复合类型“IL”

代码本身为 string 类型。

12.2 结构化文本“ST”

复合类型“ST”表示结构化文本中的一段代码。其内容如图 61 所示。

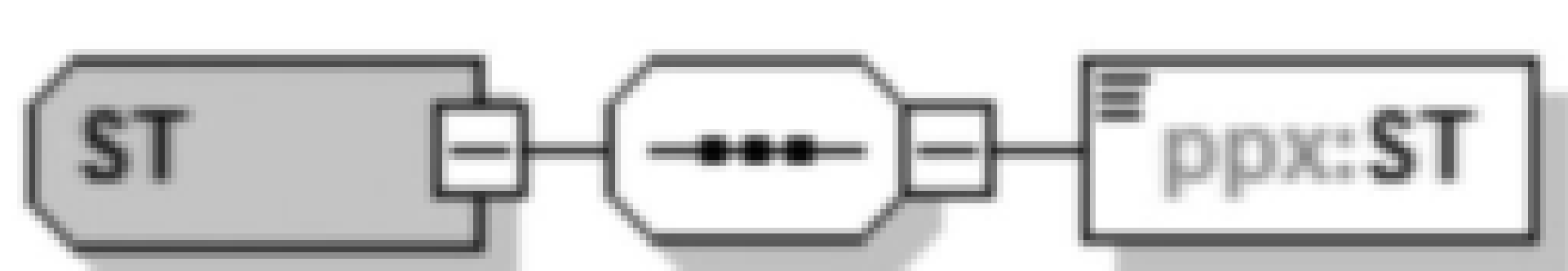


图 61 复合类型“ST”

代码本身为 string 类型。

12.3 功能块图“FBD”

复合类型“FBD”表示功能块图中的一段代码。其内容如图 62 所示。

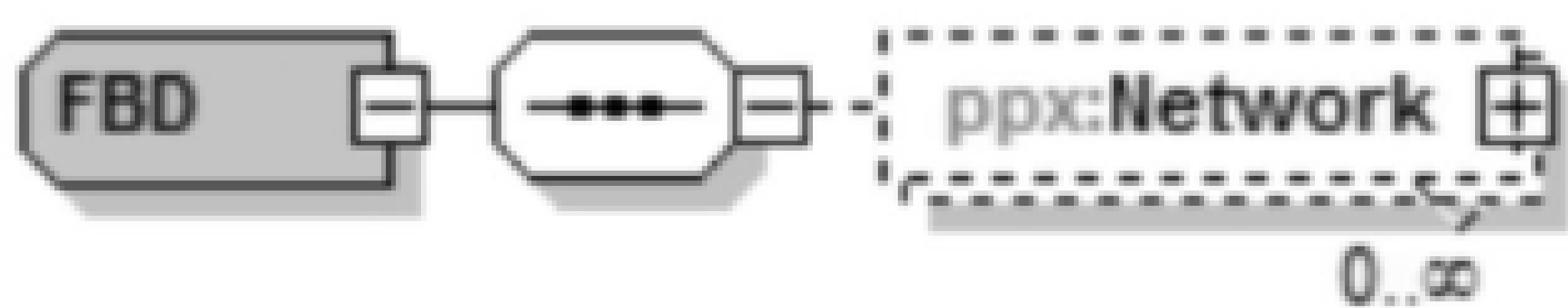


图 62 复合类型“FBD”

FBD 由一系列的网路组织而成。“Network”所属类型为“FbdNetwork”，该类型将在 12.4 介绍。

12.4 FBD 网络“FbdNetwork”

复合类型“FbdNetwork”表示 FBD 中的单个网络。其内容如图 63 所示。

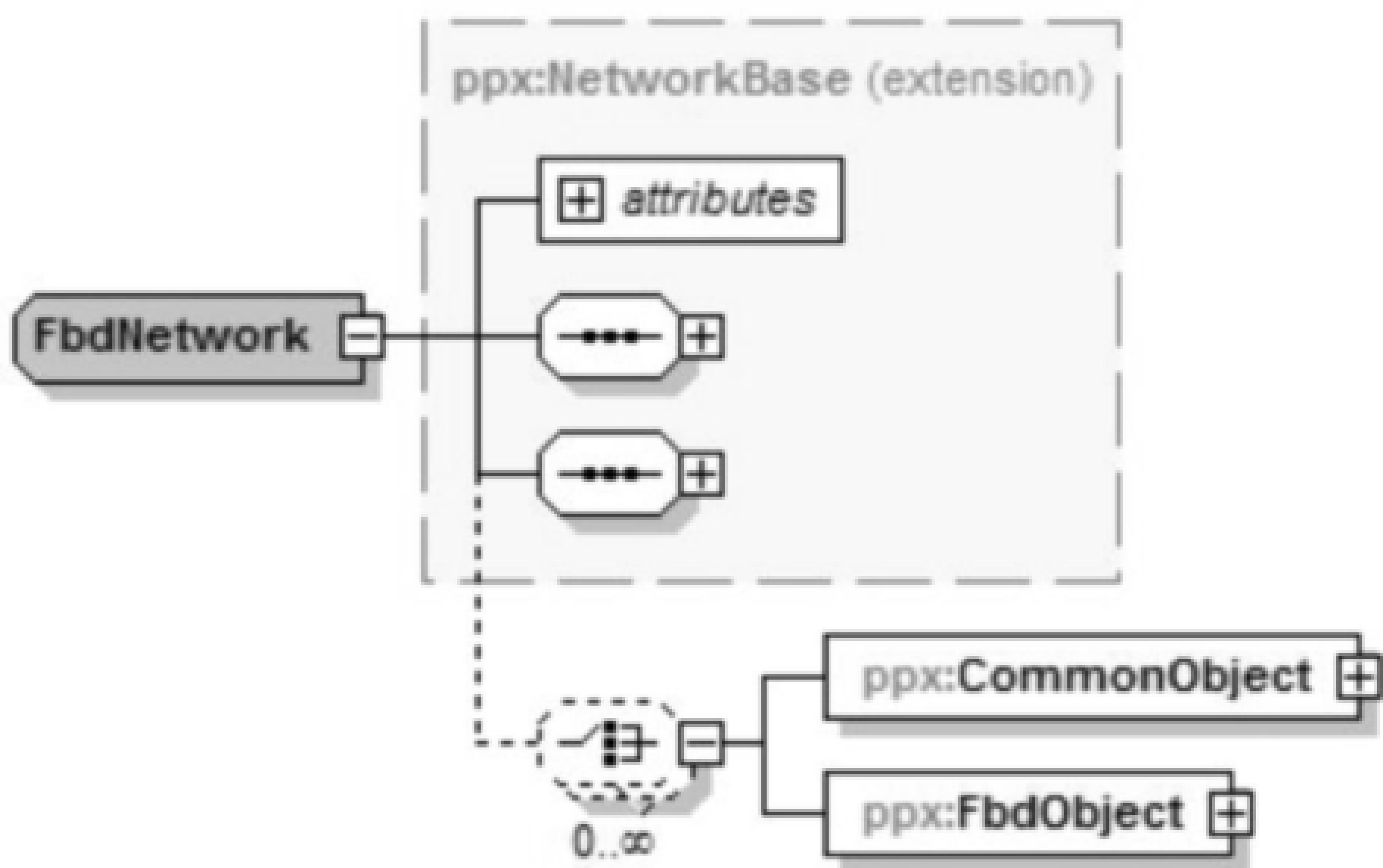


图 63 复合类型“FbdNetwork”

“FbdNetwork”扩展了 7.4.8 中介绍的“NetworkBase”类型。每个“FbdNetwork”表示一个 FBD 网络,包含零个或多个“CommonObjectBase”类型(已在 7.4.4 中介绍)的“CommonObject”或“FbdObject-Base”类型(已在 7.4.5 中介绍)的“FbdObject”。此外,也可在“FbdNetwork”中使用 7.4 中介绍的抽象复合类型的派生类型。

12.5 梯形图“LD”

复合类型“LD”表示梯形图中的一段代码。其内容如图 64 所示。

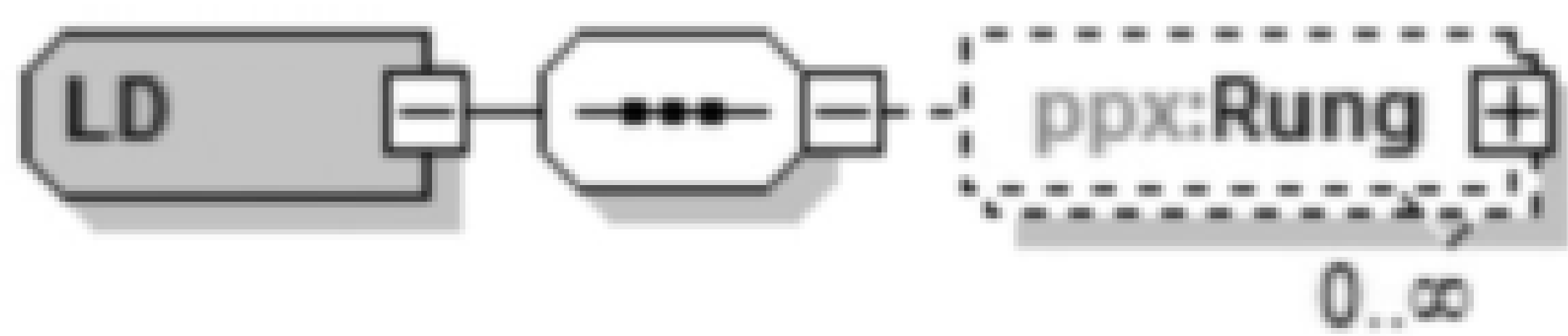


图 64 复合类型“LD”

LD 代码由一系列梯级组织而成。“Rung”所属类型为“LadderRung”，该类型将在 12.6 中介绍。

12.6 梯级“LadderRung”

复合类型“LadderRung”表示 LD 中的梯级。其内容如图 65 所示。

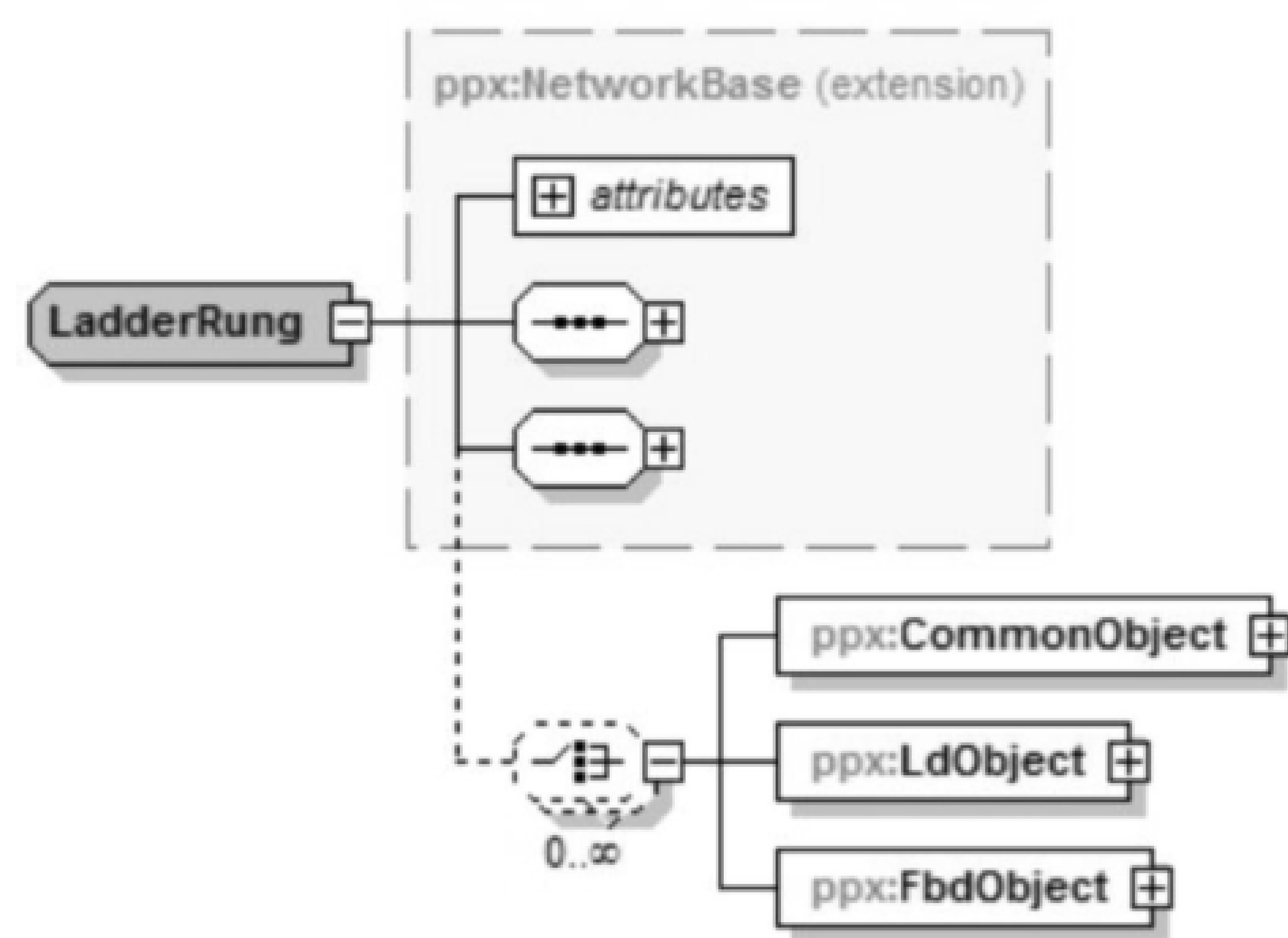


图 65 复合类型“LadderRung”

“LadderRung”扩展了 7.4.8 中介绍的“NetworkBase”类型。每个“Rung”表示一个 LD 网络,包含零个或多个“CommonObjectBase”类型(已在 7.4.4 中介绍)的“CommonObject”、“LdObjectBase”类型(已在 7.4.6 中介绍)的“LdObject”或“FbdObjectBase”类型(已在 7.4.5 介绍)的“FbdObject”。此外,也可在“Rung”中使用 7.4 中介绍的抽象复合类型的派生类型。

12.7 顺序功能图“SFC”

复合类型“SFC”表示顺序功能图。其内容如图 66 所示。

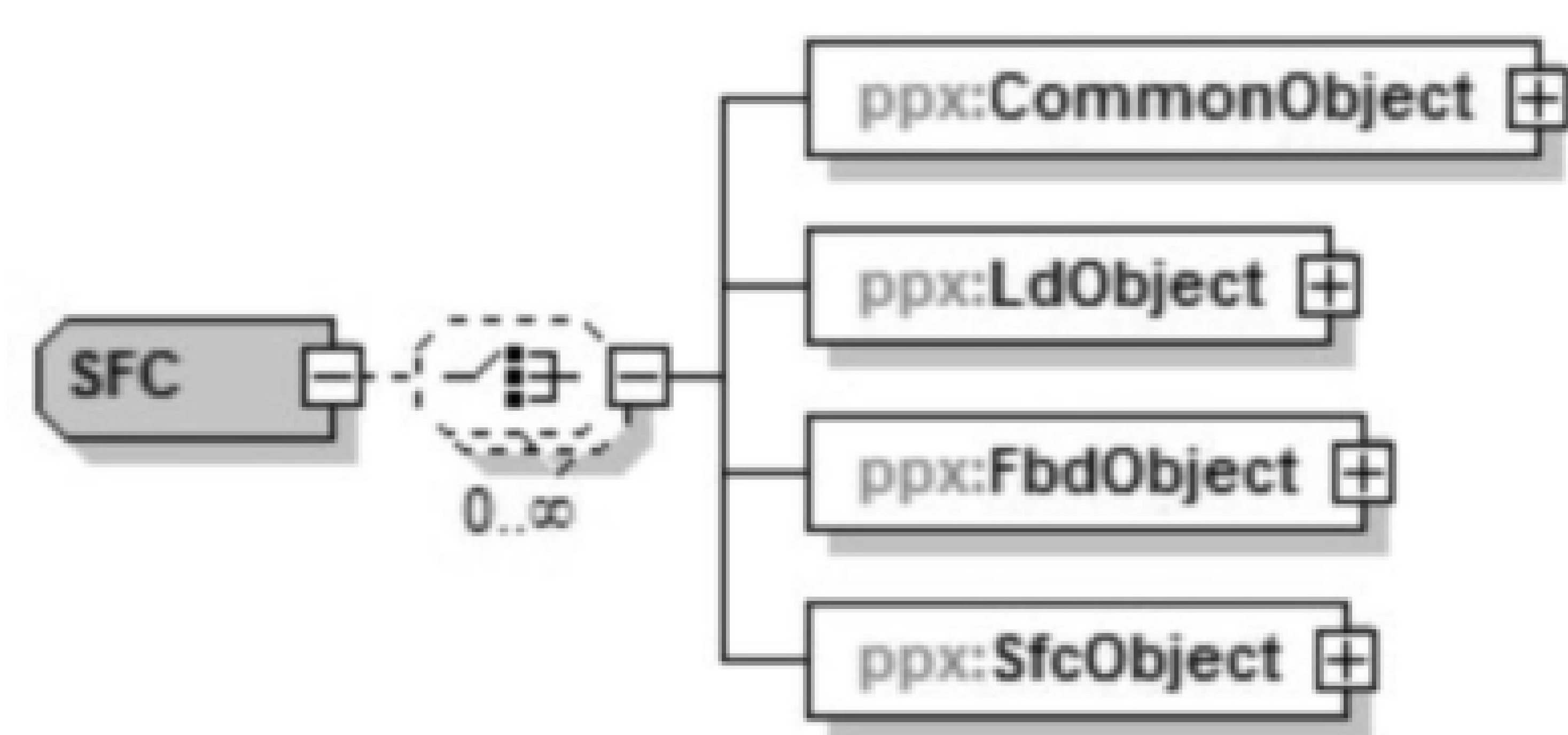


图 66 复合类型“SFC”

SFC 包含零个或多个“CommonObjectBase”“FbdObjectBase”“SfcObjectBase”类型的对象,这些类型已分别在 7.4.4、7.4.5 和 7.4.6 中介绍。此外,也可在“SFC”中使用 7.4 中介绍的抽象复合类型的派生类型。

13 图形行为描述

13.1 概述

通用元素(13.2)可用于 FBD、LD 和 SFC 网络。

FBD 元素(13.3)也可用于 LD 和 SFC 网络。

LD 元素(13.4)也可用于 SFC 网络。

第 13 章使用的连接概念将在 13.6.1 中介绍。

13.2 通用元素

13.2.1 注释“Comment”

复合类型“Comment”表示图形化语言中的注释。其内容如图 67 所示。

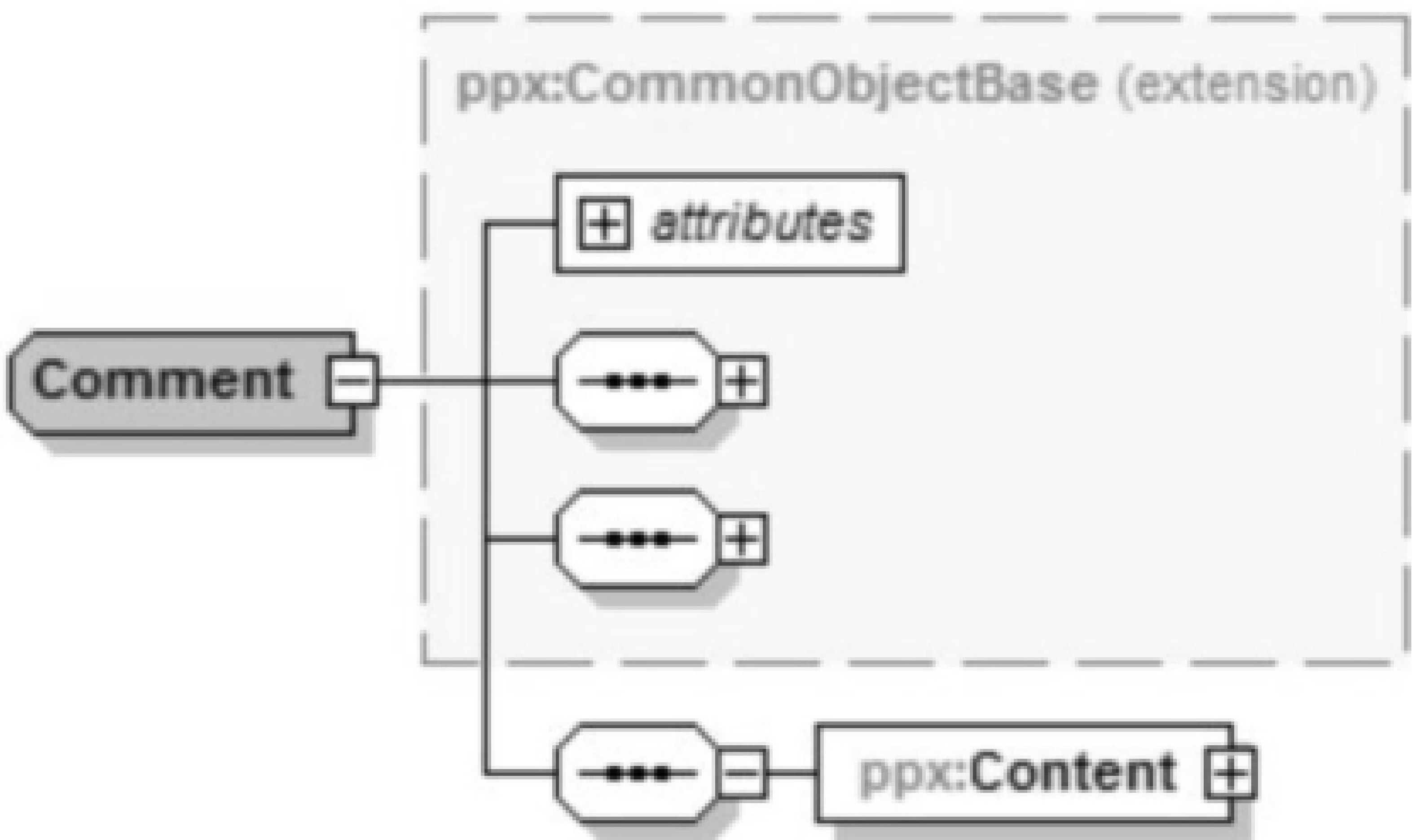


图 67 复合类型“Comment”

“Comment”扩展了 7.4.4 中介绍的基本复合类型“CommonObjectBase”。  
元素“Content”表示对象的文本内容,其所属类型“TextBase”将在 15.3 中介绍。

13.2.2 连接源端“Connector”

复合类型“Connector”表示 IEC 61131-3 中定义的图形化描述中使用标号的线扩展源端。其内容如图 68 所示。

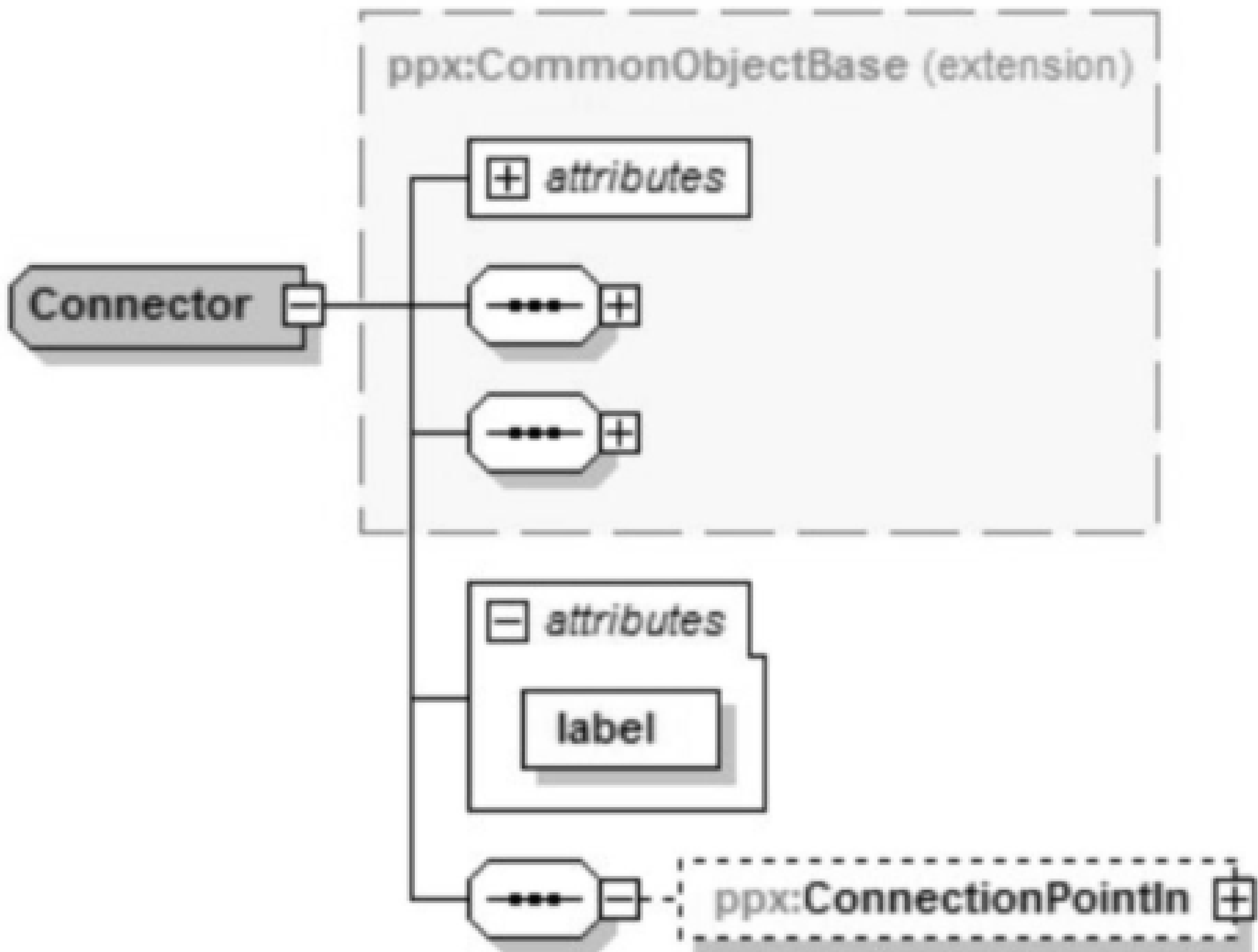


图 68 复合类型“Connector”

“Connector”扩展了 7.4.4 中介绍的基本复合类型“CommonObjectBase”。  
必需属性“label”表示 IEC 61131-3 中定义的“Connector”标号。“label”在其所在的主体内应唯一,它可匹配一个或多个“Continuation”元素。

元素“ConnectionPointIn”表示传入线的图形连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

13.2.3 连接末端“Continuation”

复合类型“Continuation”表示 IEC 61131-3 中定义的图形化描述中使用标号的线扩展目的端。其内容如图 69 所示。

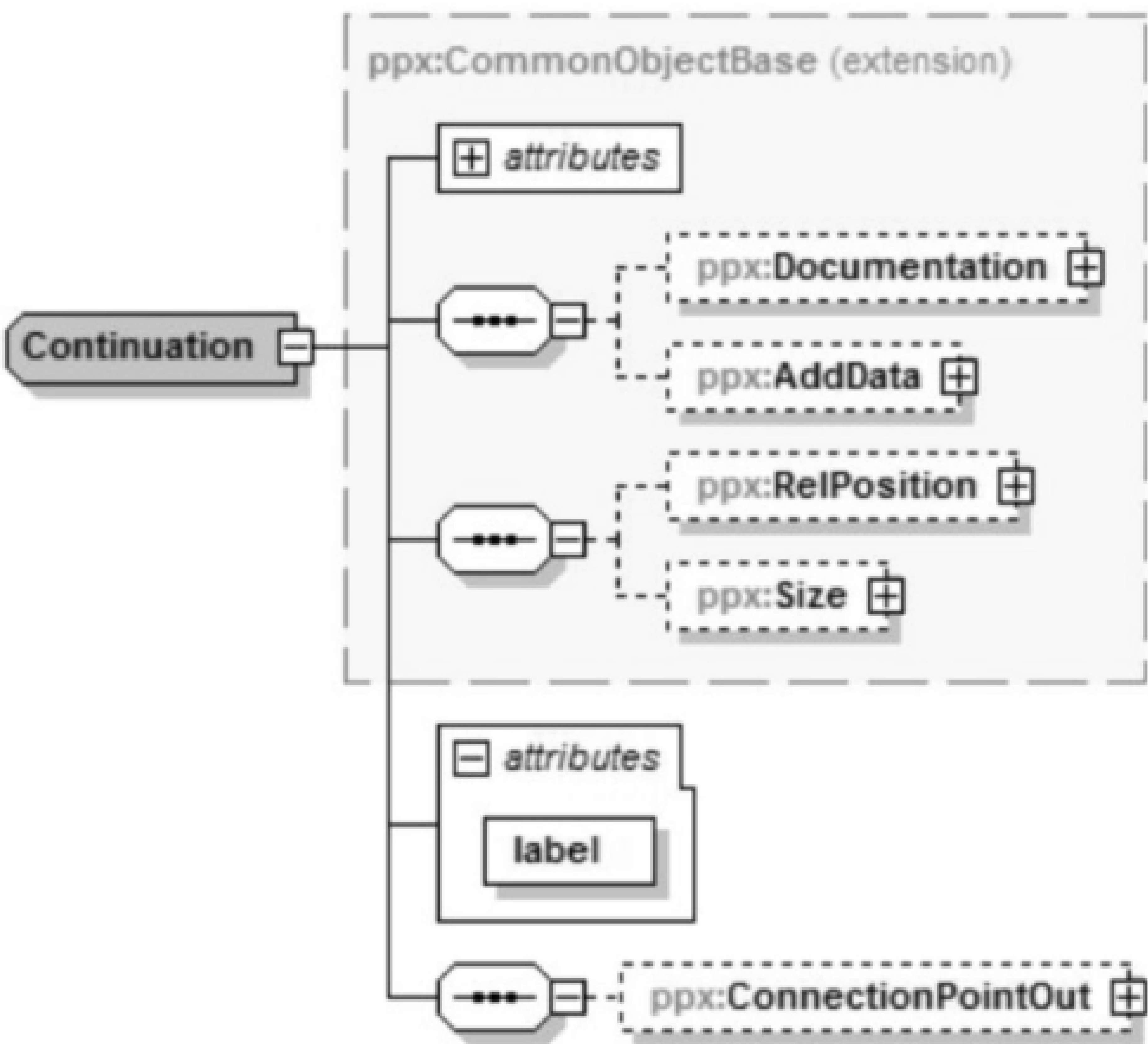


图 69 复合类型“Continuation”

“Continuation”扩展了 7.4.4 中介绍的基本复合类型“CommonObjectBase”。

必需属性“label”表示 IEC 61131-3 中定义的“continuation”标号。“label”在其所在的主体内应唯一,它可匹配一个或多个“Connector”元素。

元素“ConnectionPointOut”表示传出线的图形连接点,其所属类型“ConnectionPointOut”将在 13.6.5 中介绍。

13.2.4 执行块“ActionBlocks”

复合类型“ActionBlocks”表示 IEC 61131-3 中定义的可与 LD 或 FBD 中的梯级或线相关联的“执行块”或“级联执行块”。其内容如图 70 所示。

“ActionBlocks”扩展了 7.4.4 中介绍的基本复合类型“CommonObjectBase”。

元素“ConnectionPointIn”表示传入线的图形连接点,可与 LD 中的梯级或 FBD 中的布尔类型线进行连线,如 IEC 61131-3 中所定义。其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

元素“ActionBlock”表示每个具体动作的执行。这里可指定多个“ActionBlock”来表示“级联执行块(concatenated action block)”,如 IEC 61131-3 中所定义。元素“ActionBlock”为匿名复合类型,基于 7.4.3 中介绍的抽象复合类型“GraphicalObjectBase”,并增加了以下内容。



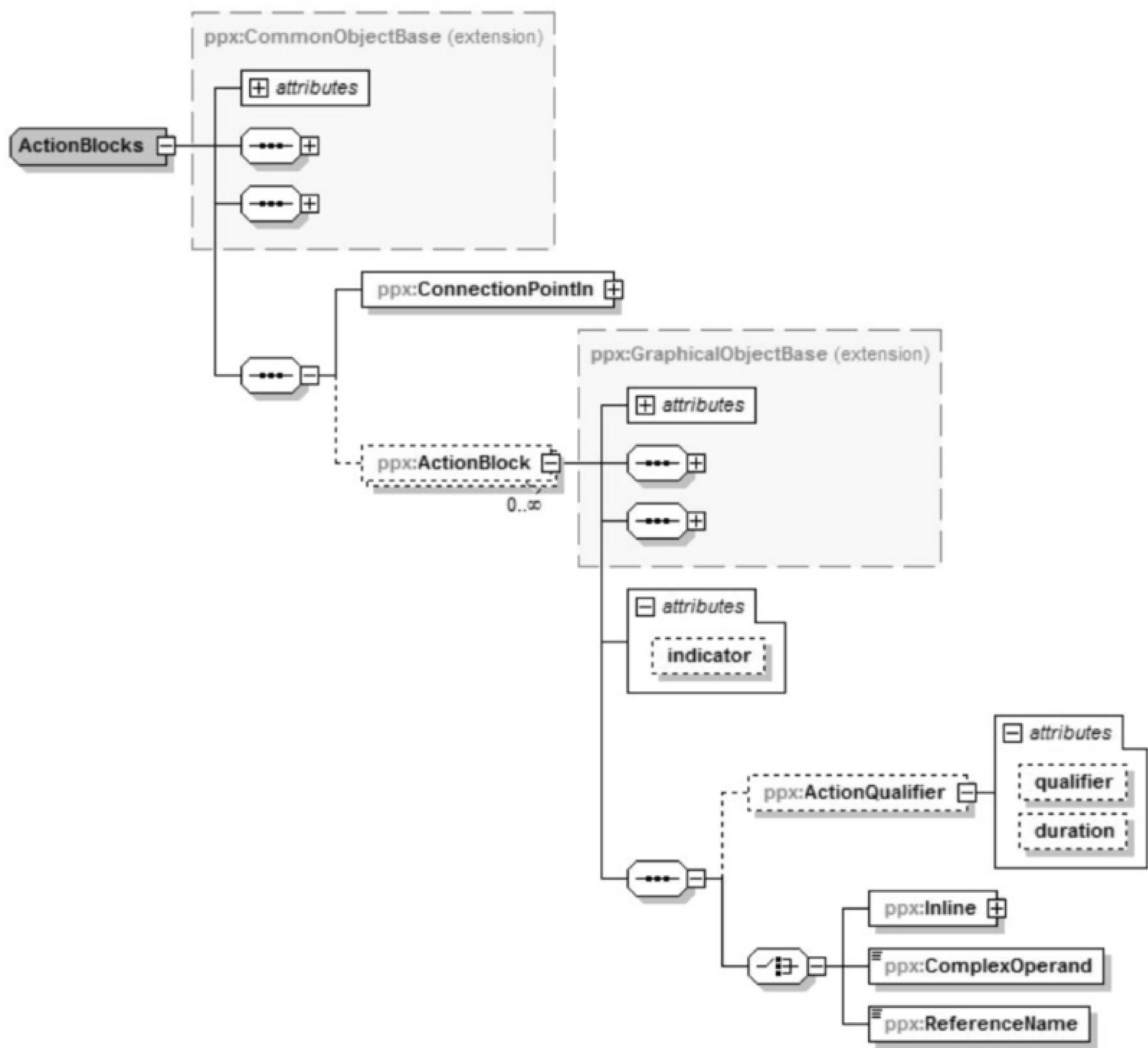


图 70 复合类型“ActionBlocks”

- 可选属性“indicator”表示执行块的指示器变量名称,如 IEC 61131-3 中所定义。
- 元素“ActionQualifier”表示由以下属性构成的动作限定符。可选属性“qualifier”表示动作限定符类型,例如“P1”“N”“P0”“R”“S”“L”“D”“P”“DS”“DL”“SD”或“SL”。可选属性“duration”表示 TIME 或 LTIME 常量表达式中的持续时间,对某些类型的动作限定符而言是必需的。
- 应选择“Inline”“ComplexOperand”或“ReferenceName”元素来表示动作主体。
- 元素“Inline”表示在执行块中实现的动作主体,其所属类型“Body”已在 10.14 中介绍。
- 元素“ComplexOperand”表示一个布尔变量名。
- 元素“ReferenceName”表示在同一 POU 中声明的可访问动作的名称或布尔变量的名称,其所属类型为 string。

13.3 FBD 元素

13.3.1 块“Block”

复合类型“Block”表示功能调用、功能块实例调用、功能块方法调用或类实例方法调用的图形化描述。其内容如图 71 所示。

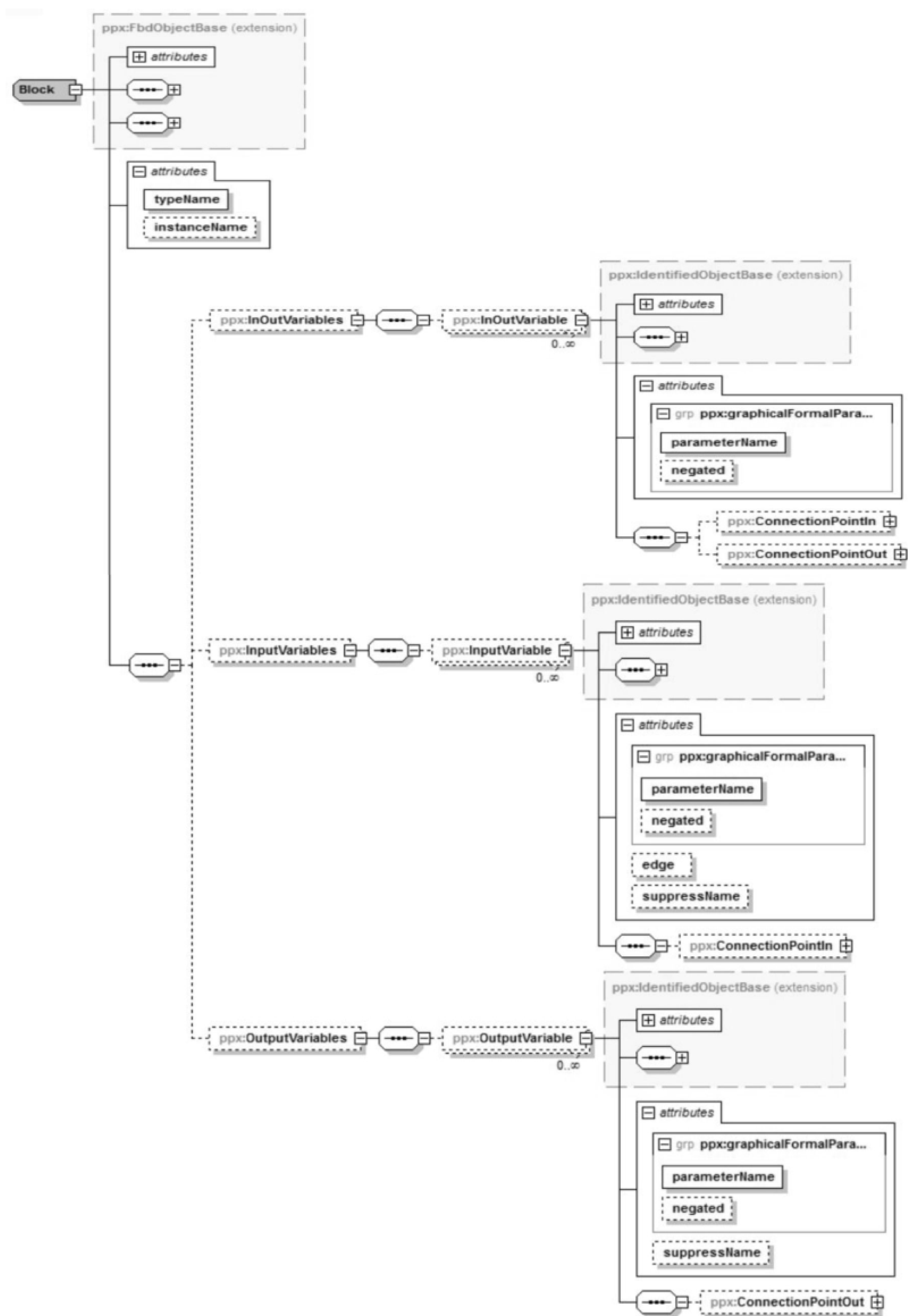


图 71 复合类型“Block”

“Block”扩展了 7.4.5 中介绍的基本复合类型“FbdObjectBase”。

必需属性“typeName”表示要在矩形块顶部居中显示的类型名称。可选属性“instanceName”表示要在矩形块上方显示的实例名称。对于功能,则不使用此属性。

元素“InOutVariables”表示块中输入-输出变量的图形化表示,由一组表示每个输入-输出变量规范的“InOutVariable”元素组成。元素“InOutVariable”为匿名复合类型,扩展了 7.4.2 中介绍的“IdentifiedObjectBase”,并增加了以下内容。

- 属性组“graphicalFormalParameterCommon”表示块中输入变量、输出变量和输入-输出变量图形化描述的公共属性。详细内容将在 13.3.2 中介绍。
- 元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。
- 元素“ConnectionPointOut”表示可连接右侧线的连接点,其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

元素“InputVariables”表示块中输入变量的图形化表示,由一组表示每个输入变量规范的“InputVariable”元素组成。元素“InputVariable”为匿名复合类型,扩展了 7.4.2 中介绍的“IdentifiedObjectBase”,并增加了以下内容。

- 属性组“graphicalFormalParameterCommon”表示块中输入变量、输出变量和输入-输出变量图形化描述的公共属性。详细内容将在 13.3.2 中介绍。
- 可选属性“edge”表示 IEC 61131-3 中定义的边缘修饰符。此属性仅用于形式参数的图形化描述,是否应用上升沿或下降沿应在相应功能块类型的变量声明中定义。
- 可选属性“suppressName”表示没有名称的形式参数。IEC 61131-3 为 MUL 或 AND 等标准功能定义了此类参数。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属类型“ConnectionPointIn”将在 13.6.2 中介绍。

元素“OutputVariables”表示块中输出变量的图形化表示,由一组表示每个输出变量规范的“OutputVariable”元素组成。元素“OutputVariables”为匿名复合类型,扩展了 7.4.2 中介绍的“IdentifiedObjectBase”,并增加了以下内容。

- 属性组“graphicalFormalParameterCommon”表示块中输入变量、输出变量和输入-输出变量的图形化描述的公共属性。详细内容将在 13.3.2 中介绍。
- 元素“ConnectionPointOut”表示可连接右侧线的连接点,其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

由于在 W3C 的 XML 1.0 规范中同级元素的出现顺序无意义,因此模式未明确定义“Block”的子元素“InOutVariable”“InputVariable”和“OutputVariable”的顺序。如有必要,导出系统可使用分配给每个变量的“ConnectionPointIn”或“ConnectionPointOut”中的“RelPosition”元素,以便通过图形布局隐式地表明顺序。导入系统可使用此布局,也可根据导入工具的内部规则排列变量。

### 13.3.2 图形化参数公共属性“graphicalFormalParameterCommon”

属性组“graphicalFormalParameterCommon”表示块中输入变量、输出变量和输入-输出变量图形化表示的公共属性。

必需属性“parameterName”表示输入变量、输出变量或输入-输出变量的名称。

可选属性“negated”表示 IEC 61131-3 中定义的反相布尔型输入。

13.3.3 数据源“DataSource”

复合类型“DataSource”表示将指定变量赋值给 LD 或 FBD 中行的图形化表示。其内容如图 72 所示。

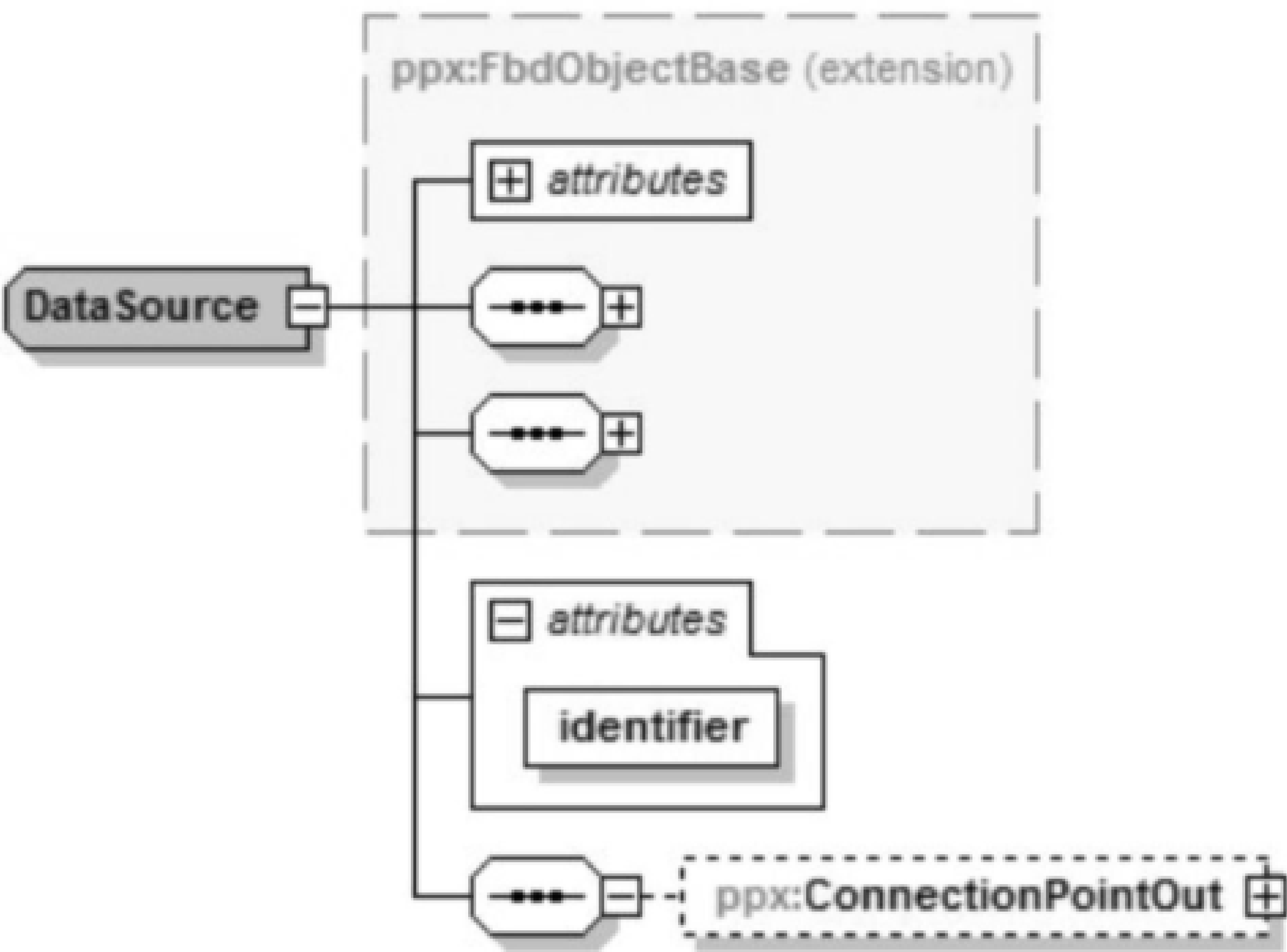


图 72 复合类型“DataSource”

“DataSource”扩展了 7.4.5 中介绍的基本复合类型“FbdObjectBase”。

必需属性“Identifier”表示变量名称或文本表达式。

元素“ConnectionPointOut”表示可连接右侧线的连接点,其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

13.3.4 数据落点“DataSink”

复合类型“DataSink”表示将 LD 或 FBD 行赋值给指定变量的图形化表示。其内容如图 73 所示。

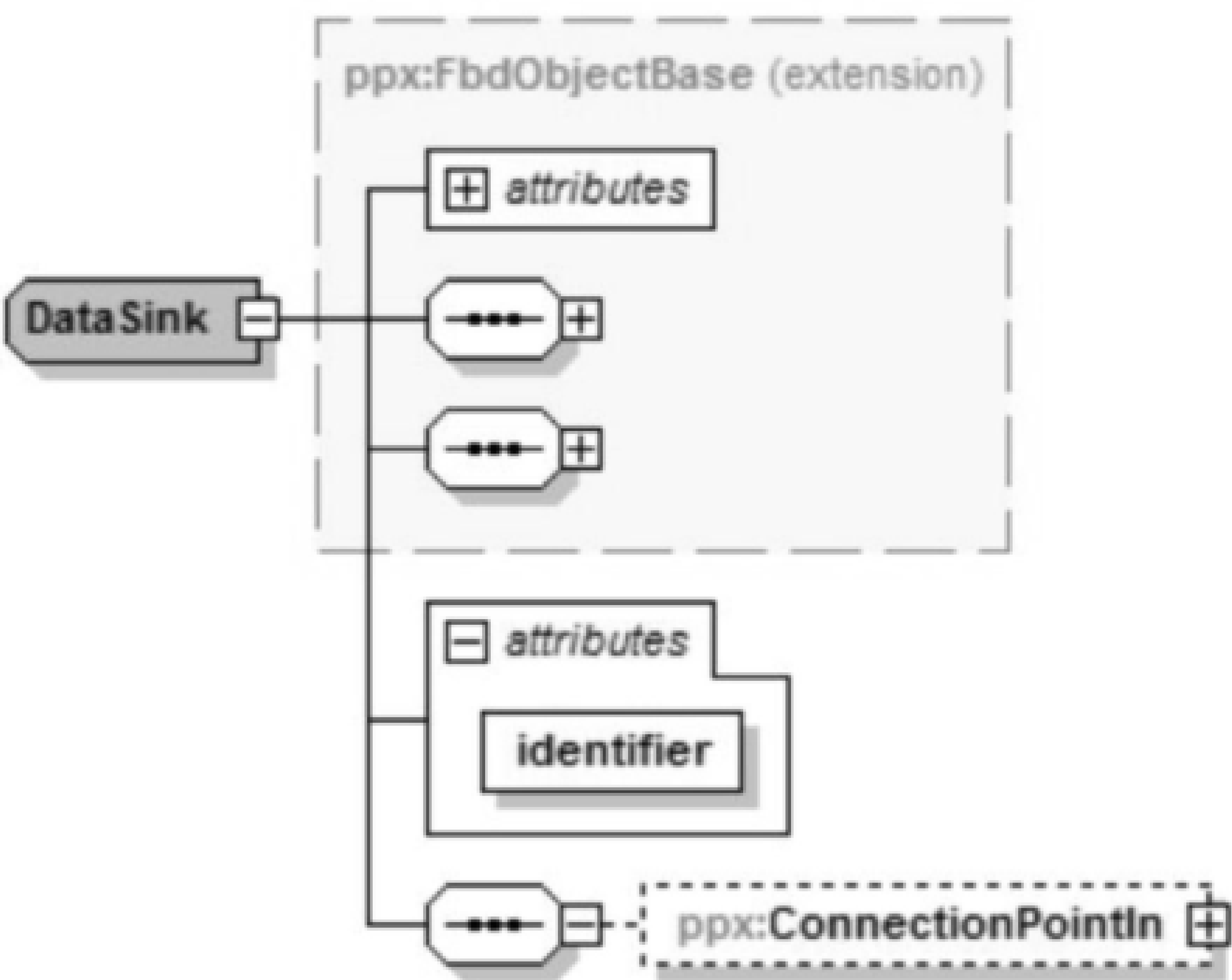


图 73 复合类型“DataSink”

“DataSink”扩展了 7.4.5 中介绍的基本复合类型“FbdObjectBase”。

必需属性“Identifier”表示变量名称。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

13.3.5 未连接“Unconnected”

复合类型“Unconnected”表示没有连接到任何行的变量或字段,特用于未完成项目。其内容如图 74所示。

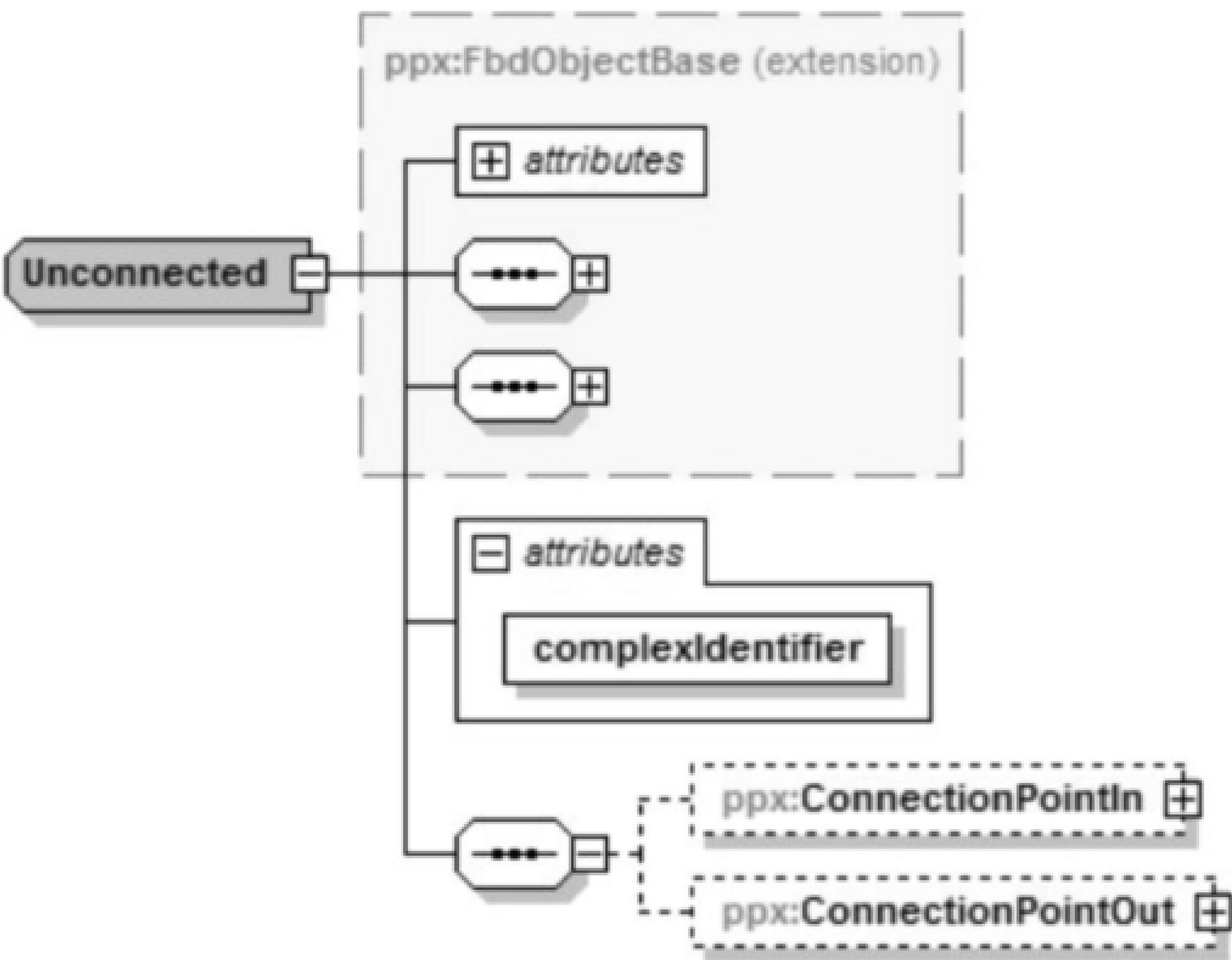


图 74 复合类型“Unconnected”

“Unconnected”扩展了 7.4.5 中介绍的基本复合类型“FbdObjectBase”。

必需属性“Identifier”表示变量名称。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

元素“ConnectionPointOut”表示可连接右侧线的连接点,其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

13.3.6 跳转“Jump”

复合类型“Jump”表示 IEC 61131-3 中定义的 LD 或 FBD 中图形化的执行控制元素“jump”。其内容如图 75 所示。

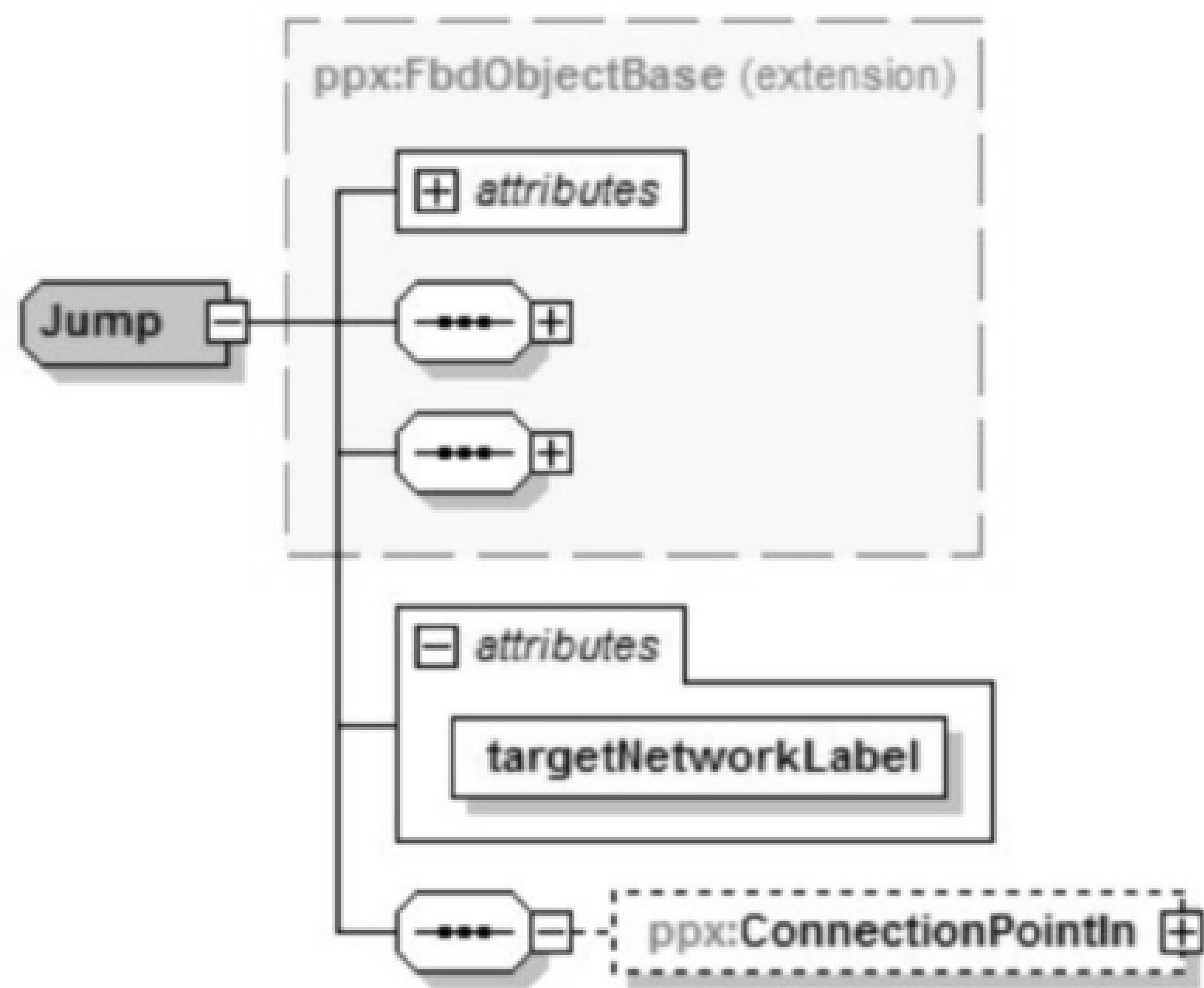


图 75 复合类型“Jump”

“Jump”扩展了 7.4.5 中介绍的基本复合类型“FbdObjectBase”。

必需属性“targetNetworkLabel”表示该跳转的目标网络的标号。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

13.3.7 返回“Return”

复合类型“Return”表示 IEC 61131-3 中定义的 LD 或 FBD 中图形化的执行控制元素“return”。其内容如图 76 所示。

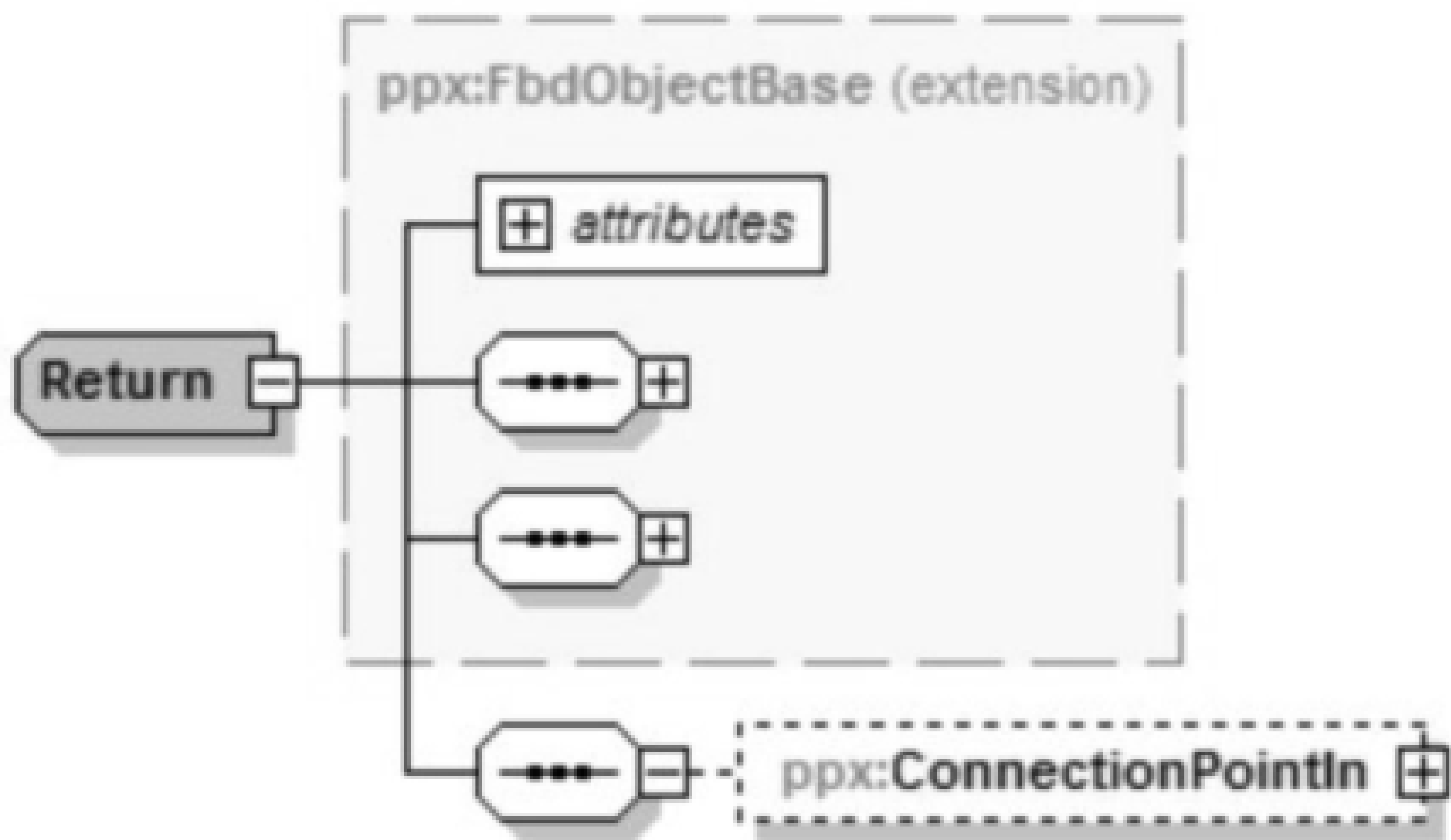


图 76 复合类型“Return”

“Return”扩展了 7.4.5 中介绍的基本复合类型“FbdObjectBase”。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

13.4 LD 元素

13.4.1 左侧电源母线“LeftPowerRail”

复合类型“LeftPowerRail”表示 LD 中图形化的左侧电源母线。其内容如图 77 所示。

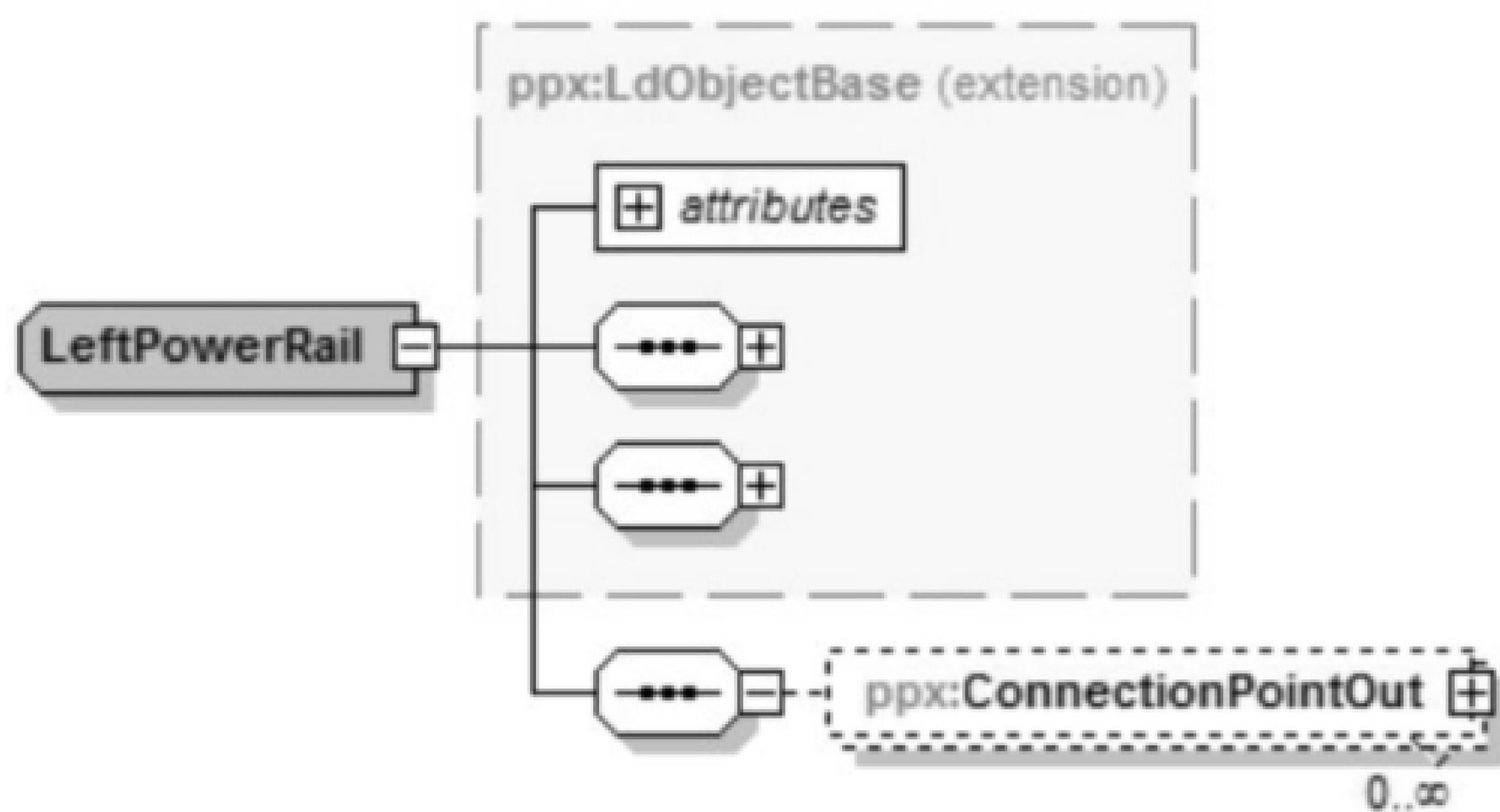


图 77 复合类型“LeftPowerRail”

“LeftPowerRail”扩展了 7.4.6 中介绍的基本复合类型“LdObjectBase”。

元素“ConnectionPointOut”表示可连接右侧线的连接点,其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

13.4.2 右侧电源母线“RightPowerRail”

复合类型“RightPowerRail”表示 LD 中图形化的右侧电源母线。其内容如图 78 所示。

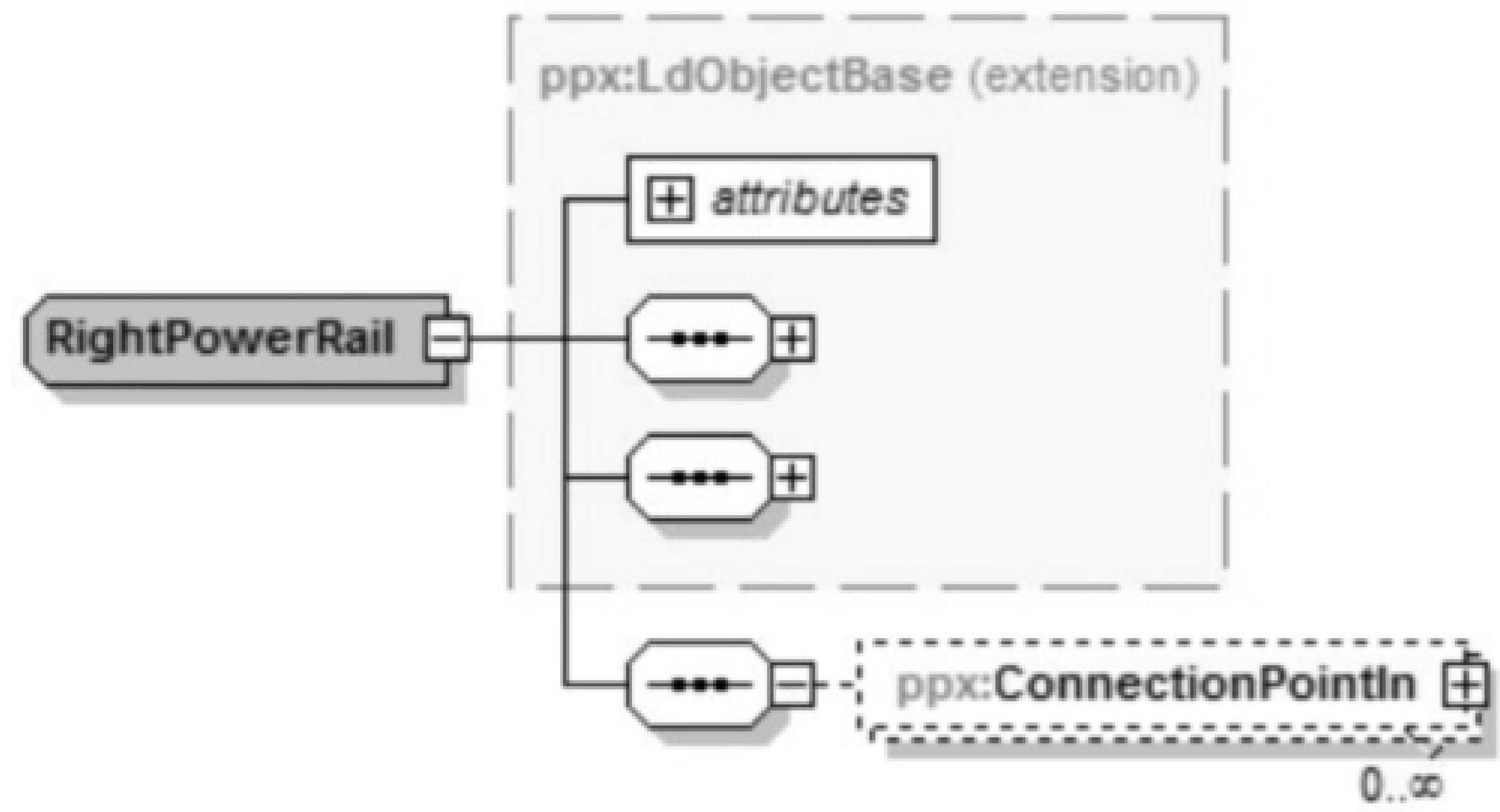


图 78 复合类型“RightPowerRail”

“RightPowerRail”扩展了 7.4.6 中介绍的基本复合类型“LdObjectBase”。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

13.4.3 线圈“Coil”

复合类型“Coil”表示 IEC 61131-3 中定义的 LD 中图形化的线圈。其内容如图 79 所示。

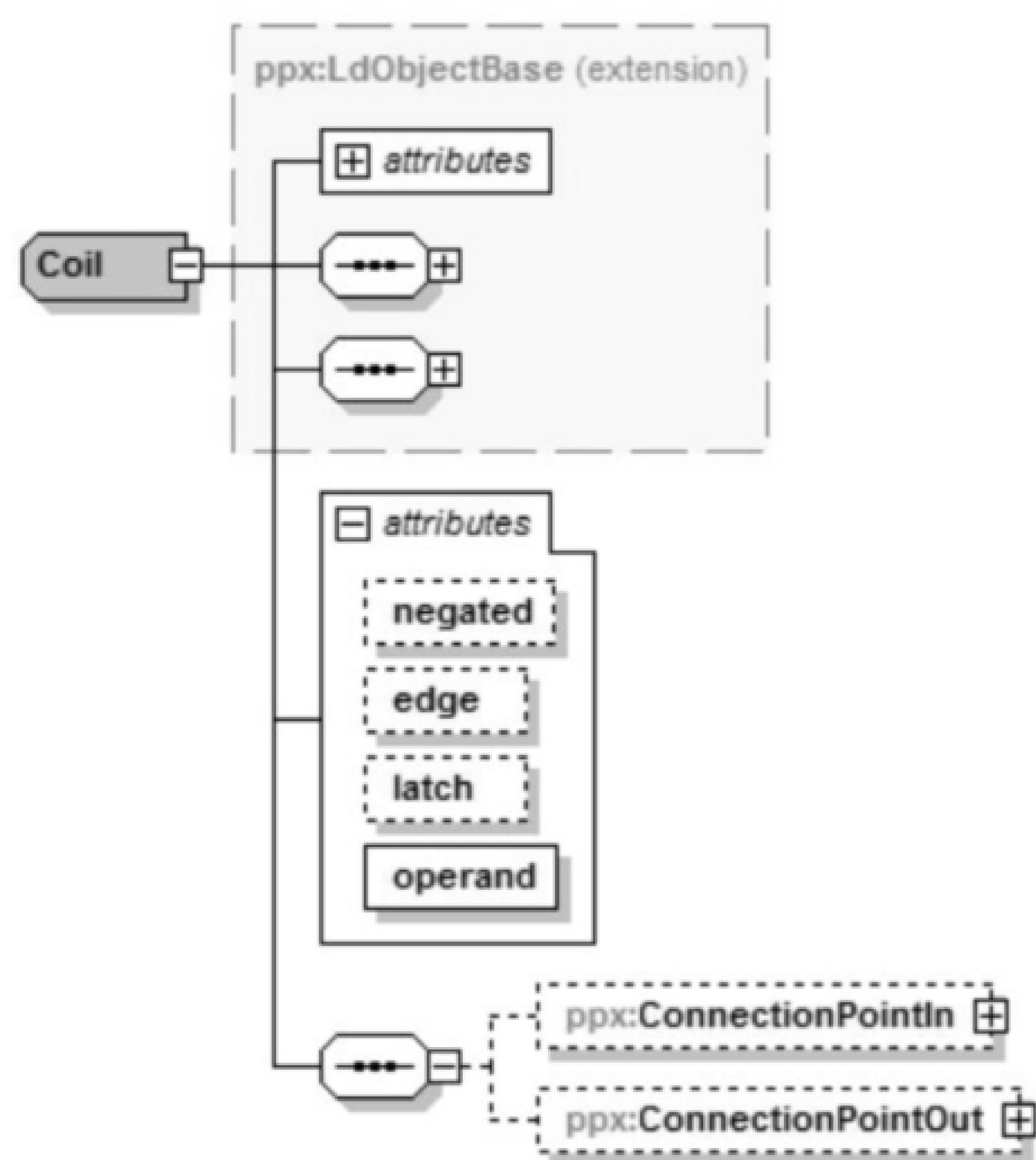


图 79 复合类型“Coil”

“Coil”扩展了 7.4.6 中介绍的基本复合类型“LdObjectBase”。

可选属性“negated”表示该线圈是否为取反线圈。可选属性“edge”表示该线圈是否为跳变触发线圈,当其值为“none”时为非跳变触发线圈,值为“rising”时为正跳变触发线圈,值为“falling”时为负跳变触发线圈。可选属性“latch”表示该线圈是否为锁存线圈,当其值为“none”时为非锁存线圈,值为“set”时为置位线圈,值为“reset”时线圈是复位线圈。必需属性“operand”表示变量名称、地址的直接表达式或线圈的文本表示。

元素“ConnectionPointIn”表示可连接左侧线的连接点,其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

元素“ConnectionPointOut”表示可连接右侧线的连接点,其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

13.4.4 触点“Contact”

复合类型“Contact”表示 IEC 61131-3 中定义的 LD 中图形化的触点。其内容如图 80 所示。





图 80 复合类型“Contact”

“Contact”扩展了 7.4.6 中介绍的基本复合类型“LdObjectBase”。

可选属性“negated”表示该触点是否为常闭触点。可选属性“edge”表示该线圈是否为跳变触点，当其值为“none”时为非跳变触点，值为“rising”时为正跳变触点，值为“falling”时为负跳变触点。必需属性“operand”表示变量名称、地址的直接表达式或触点的文本表示。

元素“ConnectionPointIn”表示可连接左侧线的连接点，其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

元素“ConnectionPointOut”表示可连接右侧线的连接点，其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

13.4.5 比较触点“CompareContact”

复合类型“CompareContact”表示 IEC 61131-3 中定义的 LD 中图形化的比较触点。其内容如图 81 所示。

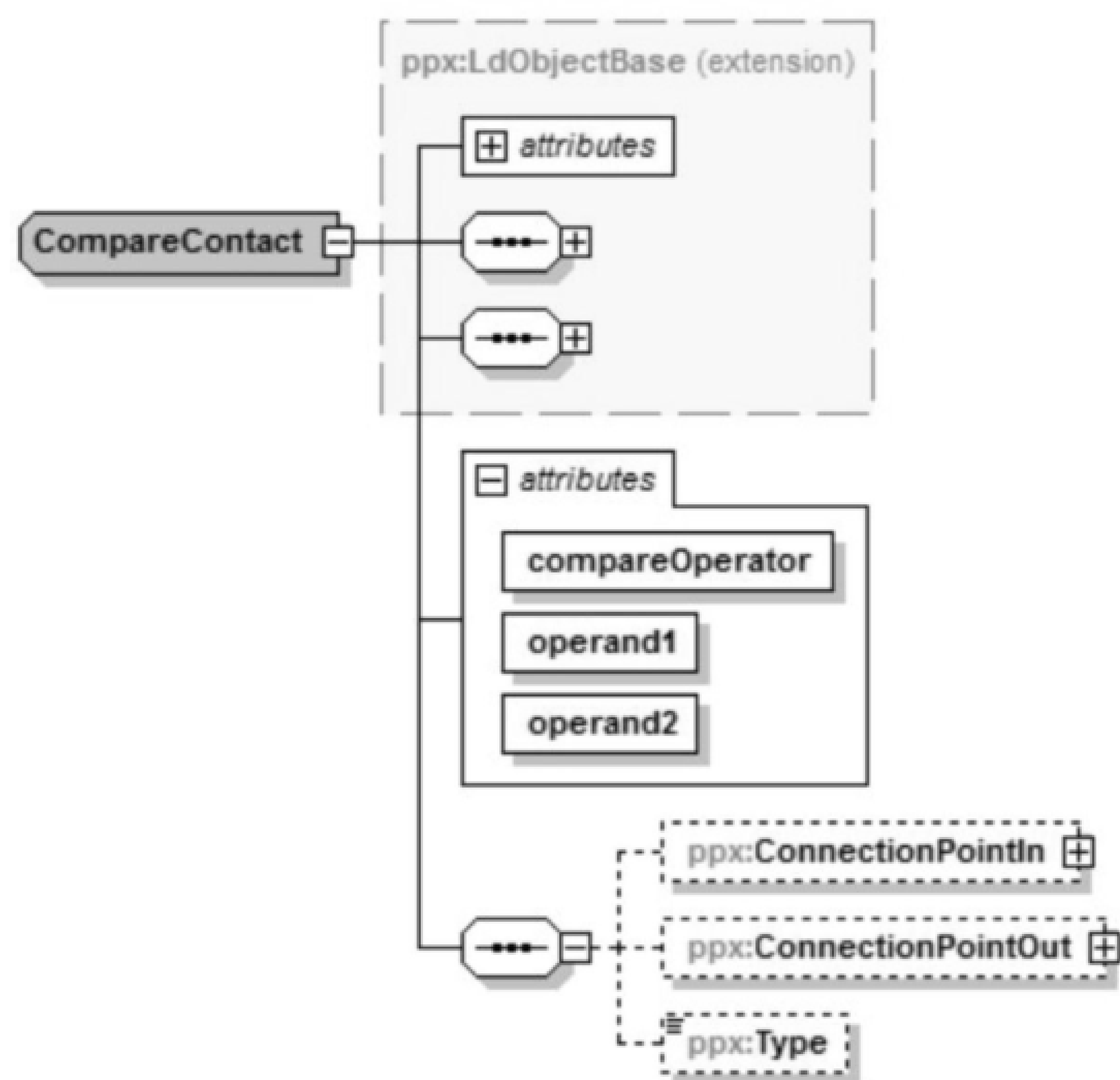


图 81 复合类型“CompareContact”

“CompareContact”扩展了 7.4.6 中介绍的基本复合类型“LdObjectBase”。

必需属性“compareOperator”表示该比较触点中的比较运算符。必需属性“operand1”表示要与另一个操作数进行比较的操作数，为 string 类型。必需属性“operand2”表示要与“operand1”进行比较的另一个操作数，为 string 类型。

元素“ConnectionPointIn”表示可连接左侧线的连接点，其所属复合类型“ConnectionPointIn”将在 13.6.2 中介绍。

元素“ConnectionPointOut”表示可连接右侧线的连接点，其所属复合类型“ConnectionPointOut”将在 13.6.5 中介绍。

元素“Type”表示“operand1”和“operand2”的数据类型，其所属复合类型“ElementaryType”已在 9.9 中介绍。两个操作数应具有相同的 IEC 61131-3 中定义的基本数据类型。

13.5 SFC 元素

13.5.1 步“Step”

复合类型“Step”表示 IEC 61131-3 中定义的步。其内容如图 82 所示。

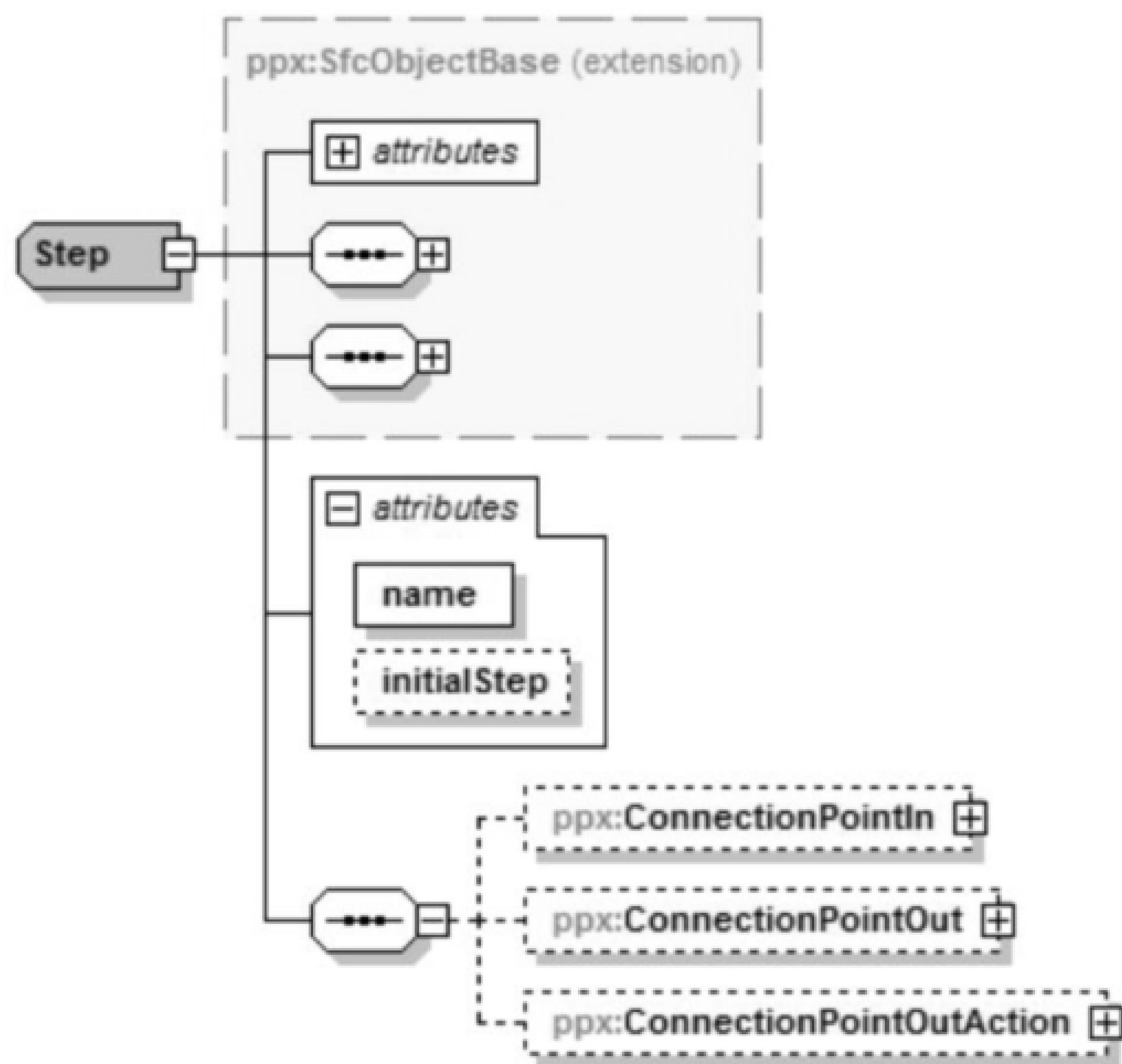


图 82 复合类型“Step”

“Step”基于抽象类型“SfcObjectBase”，该类型已在 7.4.7 中介绍。

该元素提供了“name”和“initialStep”属性。

“ConnectionPointIn”(其所属类型“ConnectionPointIn”将在 13.6.2 介绍)和“ConnectionPointOut”(其所属类型“ConnectionPointOut”将在 13.6.5 中介绍)表示 SFC 转换的前继和后继、分支和合并连接。“ConnectionPointOutAction”(其所属类型“ConnectionPointOutAction”将在 13.6.5 中介绍)表示与关联执行块的连接。

13.5.2 转换“Transition”

复合类型“Transition”表示 IEC 61131-3 中定义的转换。其内容如图 83 所示。



图 83 复合类型“Transition”

“Transition”基于抽象类型“SfcObjectBase”，该类型已在 7.4.7 中介绍。

“ConnectionPointIn”所属类型“ConnectionPointIn”和“ConnectionPointOut”所属类型“ConnectionPointOut”将分别在 13.6.2 和 13.6.5 中介绍。这些元素表明了 SFC 步的前继和后继、分支和合并的连接。

“Condition”具有下列元素之一。

- 具有属性“name”的“Reference”，该属性链接到一个已命名转换。
- 用于直接在 SFC 图中定义 LD 或 FBD 网络的“GraphicalPredicate”，其所属类型“ConnectionPointIn”将在 13.6.2 中介绍。“ConnectionPointIn”用于描述 SFC 图形内部的图形化条件。“GraphicalExpression”所属类型“NetworkBase”已在 7.4.8 中介绍。由于在“Condition”中重用了“NetworkBase”，导出工具应指定一个“evaluationOrder”。导入工具应忽略此“evaluationOrder”并根据 SFC 执行规则计算转换条件。
- 声明 ST 或 IL 中转换条件表达式的“TextualPredicate”，其所属类型“Predicate”已在 10.16 中介绍。

13.5.3 选择分支“SelectionDivergence”

复合类型“SelectionDivergence”表示 IEC 61131-3 中定义的选择分支。其内容如图 84 所示。

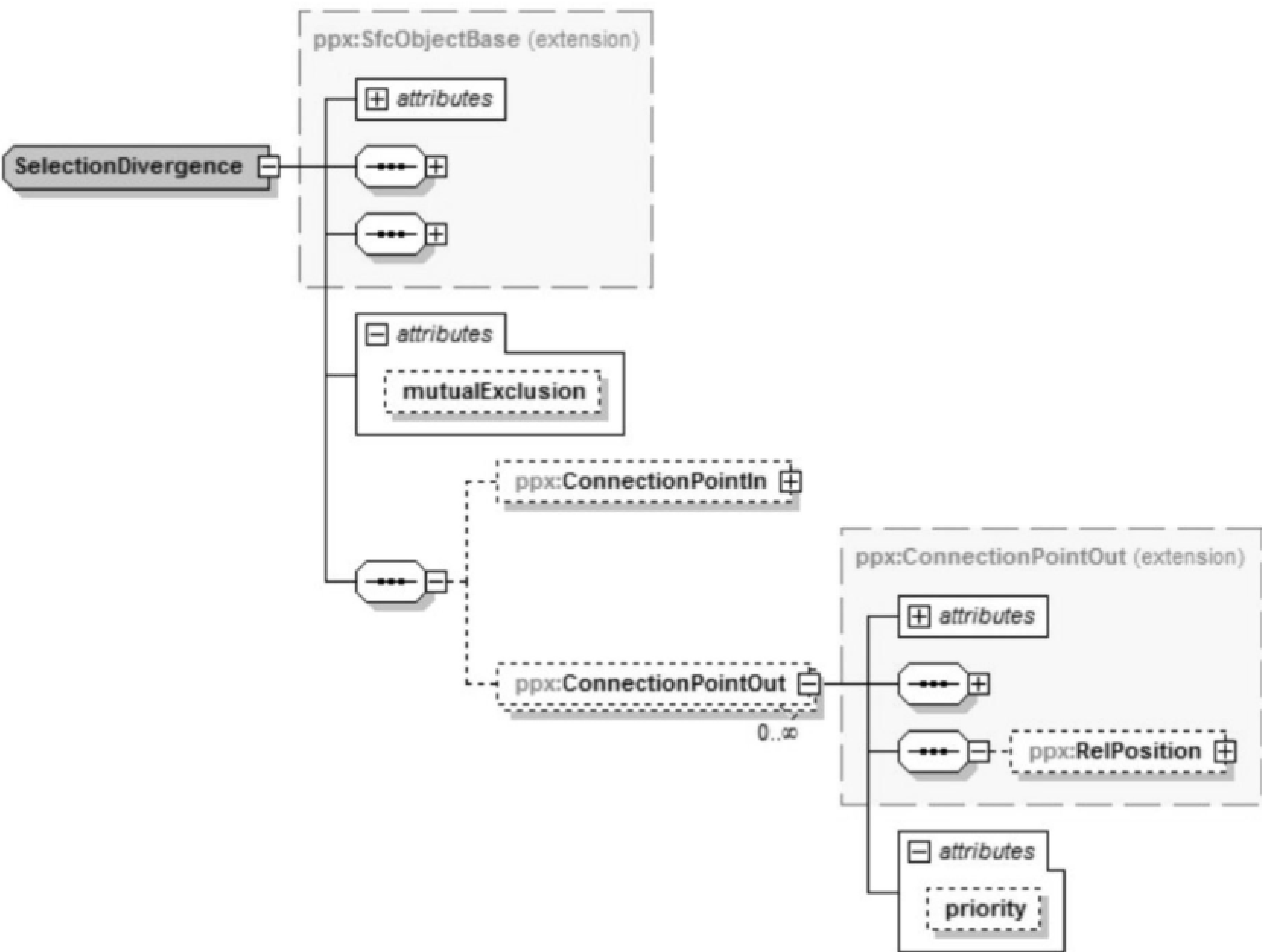


图 84 复合类型“SelectionDivergence”

“SelectionDivergence”基于抽象类型“SfcObjectBase”，该类型已在 7.4.7 中介绍。

“ConnectionPointIn”所属类型“ConnectionPointIn”和“ConnectionPointOut”所属类型“ConnectionPointOut”将分别在 13.6.2 和 13.6.5 中介绍。

IEC 61131-3 提供了 3 种策略来确定不同分支的优先级：

- a) 从左至右优先，
- b) 分支编号，
- c) 互斥。

在“SelectionDivergence”的每个“ConnectionPointOut”中，本文件定义了属性“priority”，以上任何策略都会用到该属性。在从左至右优先的策略中，属性值取决于相对位置。“Transition”所连接到的“ConnectionPointOut”的“priority”属性值越小，将会越早被执行，其值为不能为负值。

在互斥策略中，应设置布尔属性“mutualExclusion”，并应省略属性“priority”。

13.5.4 选择合并“SelectionConvergence”

复合类型“SelectionConvergence”表示 IEC 61131-3 中定义的选择合并。其内容如图 85 所示。

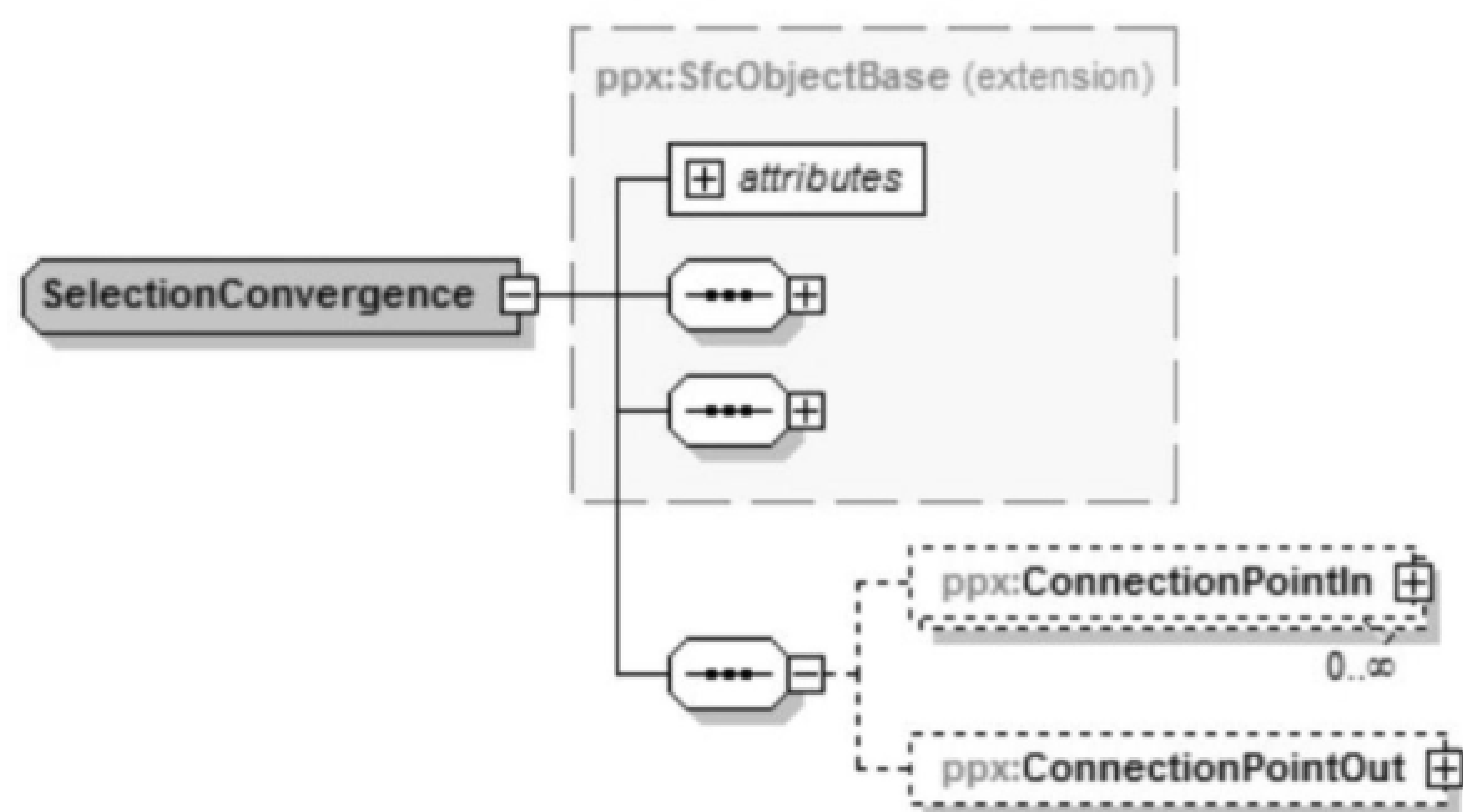


图 85 复合类型“SelectionConvergence”

“SelectionConvergence”基于抽象类型“SfcObjectBase”,该类型已在 6.10 中介绍。  
“ConnectionPointIn”所属类型“ConnectionPointIn”和“ConnectionPointOut”所属类型“ConnectionPointOut”将分别在 13.6.2 和 13.6.5 中介绍。

13.5.5 同步分支“SimultaneousDivergence”

复合类型“SimultaneousDivergence”表示 IEC 61131-3 中定义的同步分支。其内容如图 86 所示。

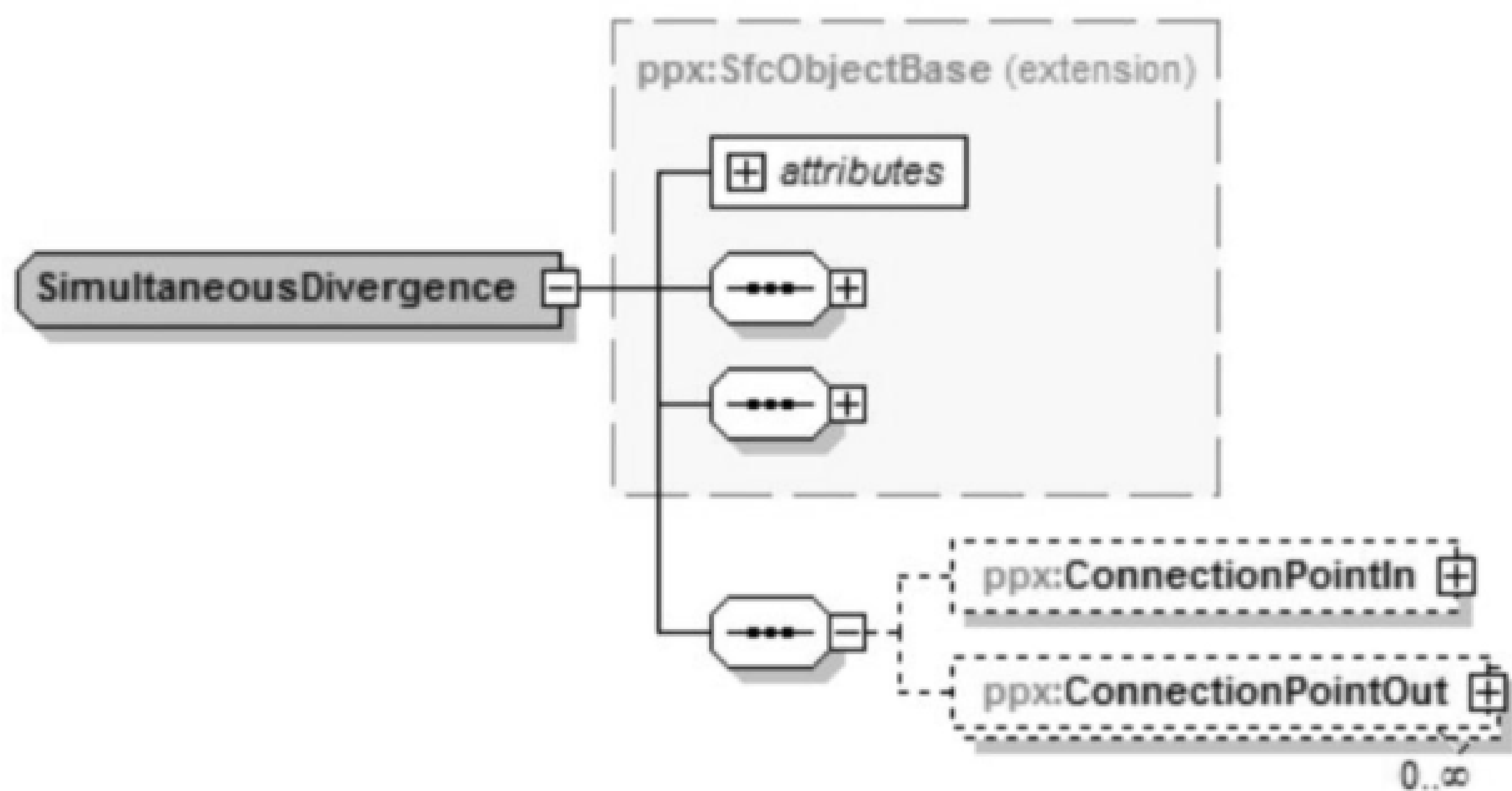


图 86 复合类型“SimultaneousDivergence”

“SimultaneousDivergence”基于抽象类型“SfcObjectBase”,该类型已在 7.4.7 中介绍。  
“ConnectionPointIn”所属类型“ConnectionPointIn”和“ConnectionPointOut”所属类型“ConnectionPointOut”将分别在 13.6.2 和 13.6.5 中介绍。

13.5.6 同步合并“SimultaneousConvergence”

复合类型“SimultaneousConvergence”表示 IEC 61131-3 中定义的同步合并。其内容如图 87 所示。

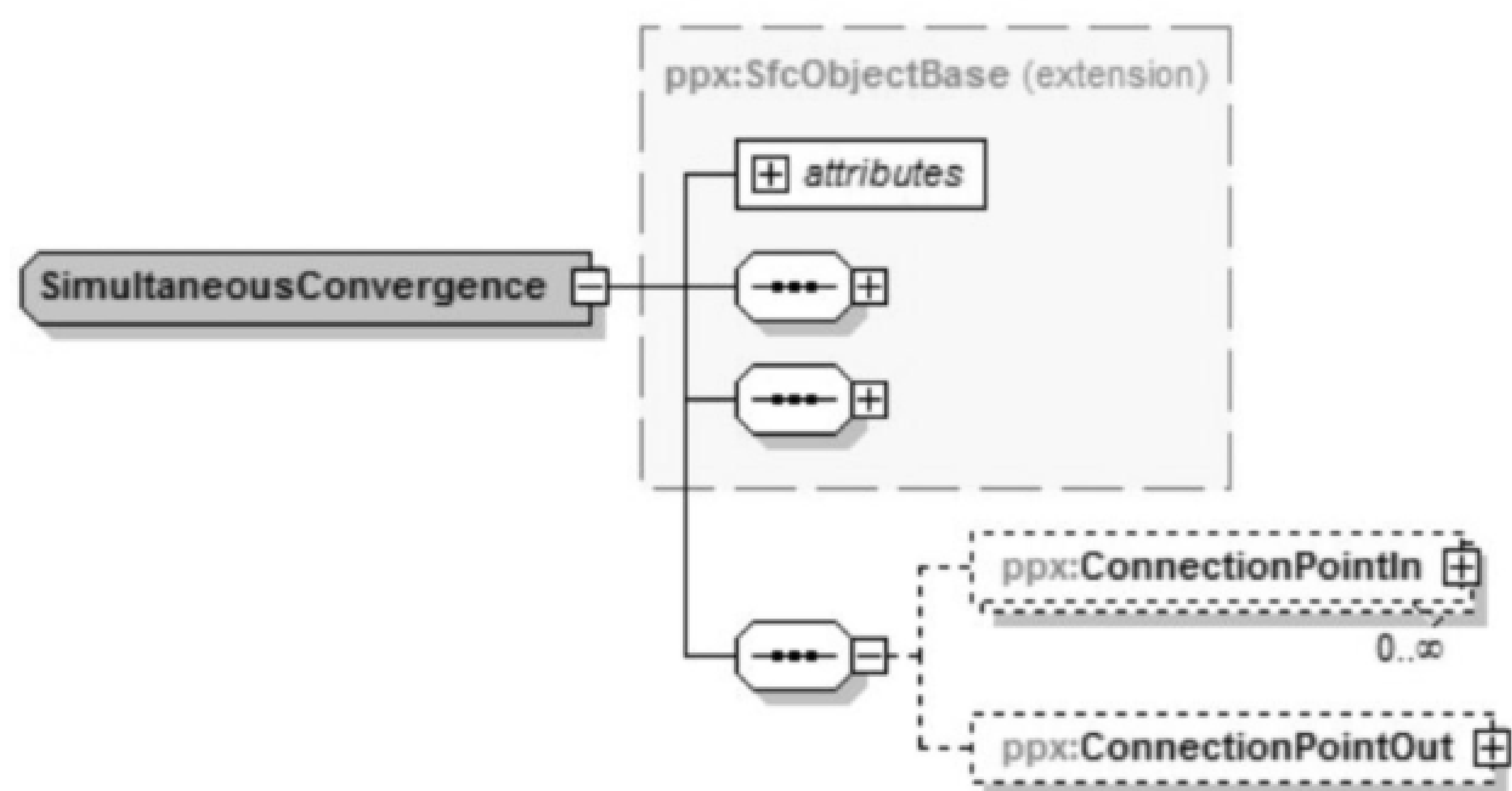


图 87 复合类型“SimultaneousConvergence”

“SimultaneousConvergence”基于抽象类型“SfcObjectBase”，该类型已在 7.4.7 中介绍。

“ConnectionPointIn”所属类型“ConnectionPointIn”和“ConnectionPointOut”所属类型“ConnectionPointOut”将分别在 13.6.2 和 13.6.5 中介绍。

13.6 连接

13.6.1 概述

“Connection”表示两个图形对象(源,目标)之间的连接。源对象具有“ConnectionPointOut”,而目标对象具有“ConnectionPointIn”。连接本身始终在“ConnectionPointIn”中指定。如果多个源对象连接到同一个目标对象(例如 LD 中若干个触点连接到同一个线圈),则在目标对象的“ConnectionPointIn”中指定“Connection”集合。

图形对象之间的连接通过 ID 引用概念实现。每个“ConnectionPointOut”都有一个“connectionPointOutId”属性。“ConnectionPointIn”中的“Connection”具有一个“refConnectionPointOutId”属性,该属性指向相应的“ConnectionPointOut”中的“connectionPointOutId”。通过这种方法,实现了从源对象到目标对象的有向联系。

13.6.2 连接入点“ConnectionPointIn”

复合类型“ConnectionPointIn”表示实现前继-后继关系所需要的传入连接点。其内容如图 88 所示。

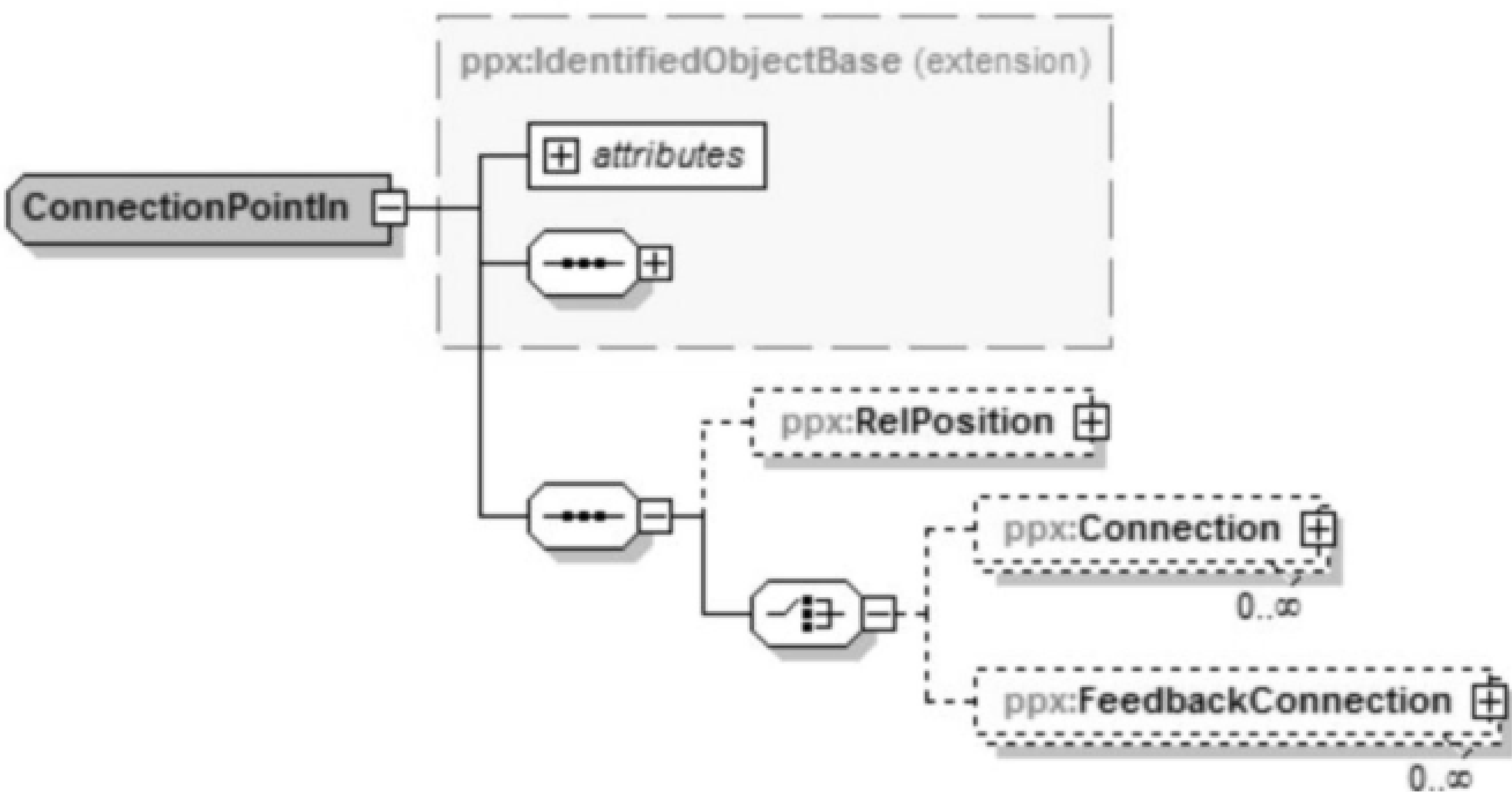


图 88 复合类型“ConnectionPointIn”

“ConnectionPointIn”基于抽象类型“IdentifiedObjectBase”，该类型已在 7.4.2 中介绍。

“RelPosition”所属类型“XyDecimalValue”、“Connection”所属类型“Connection”和“FeedbackConnection”所属类型“FeedbackConnection”将分别在 15.1、13.6.3 和 13.6.4 中介绍。

13.6.3 连接“Connection”

复合类型“Connection”表示连接路径。其内容如图 89 所示。

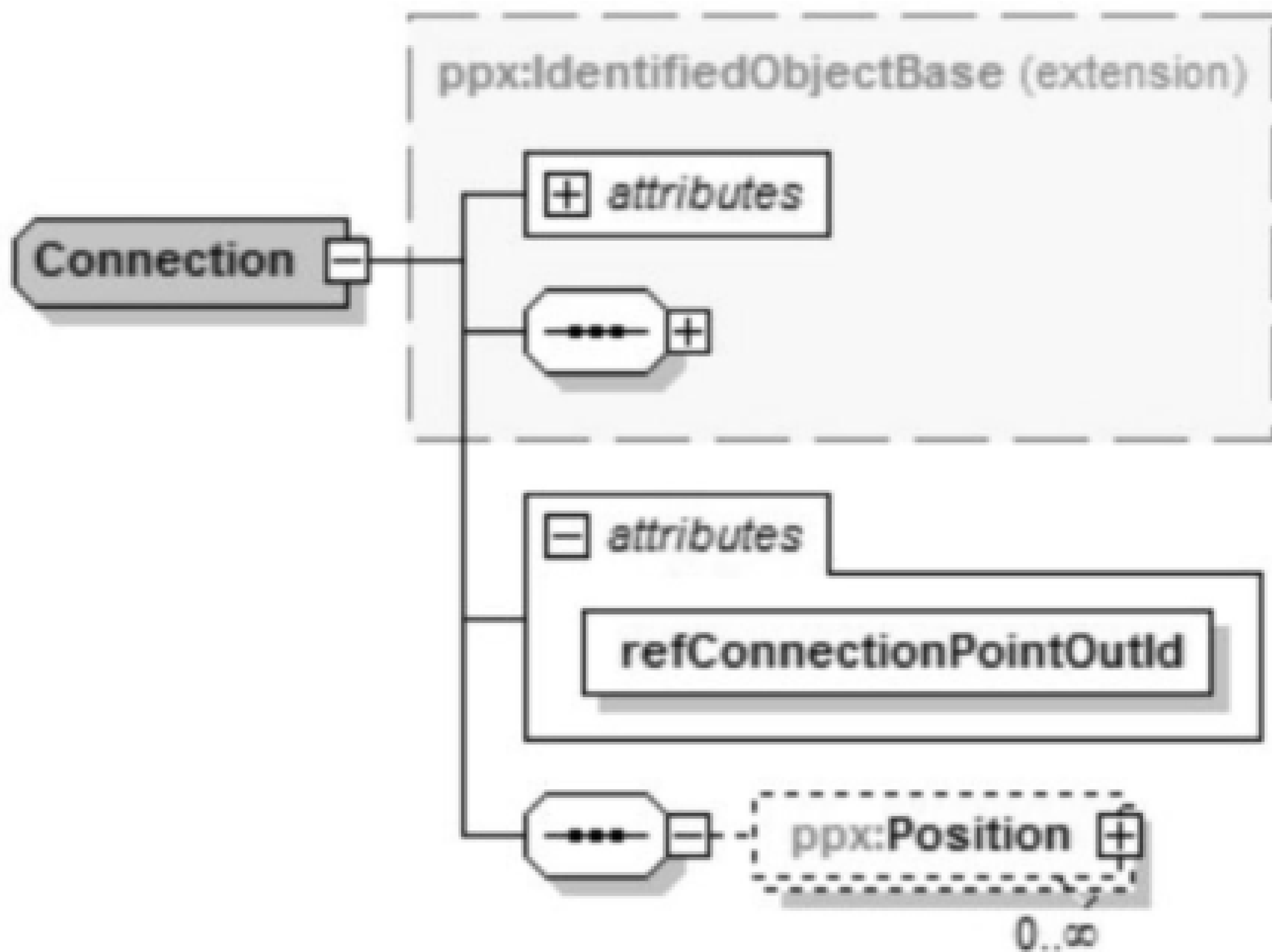


图 89 复合类型“Connection”

“Connection”基于抽象类型“IdentifiedObjectBase”，该类型已在 7.4.2 中介绍。

该元素提供了“refConnectionPointOutId”属性。

“Position”所属类型“XyDecimalValue”将在 15.1 中介绍。

13.6.4 反馈连接“FeedbackConnection”

复合类型“FeedbackConnection”表示 IEC 61131-3 中定义的反馈路径。其内容如图 90 所示。



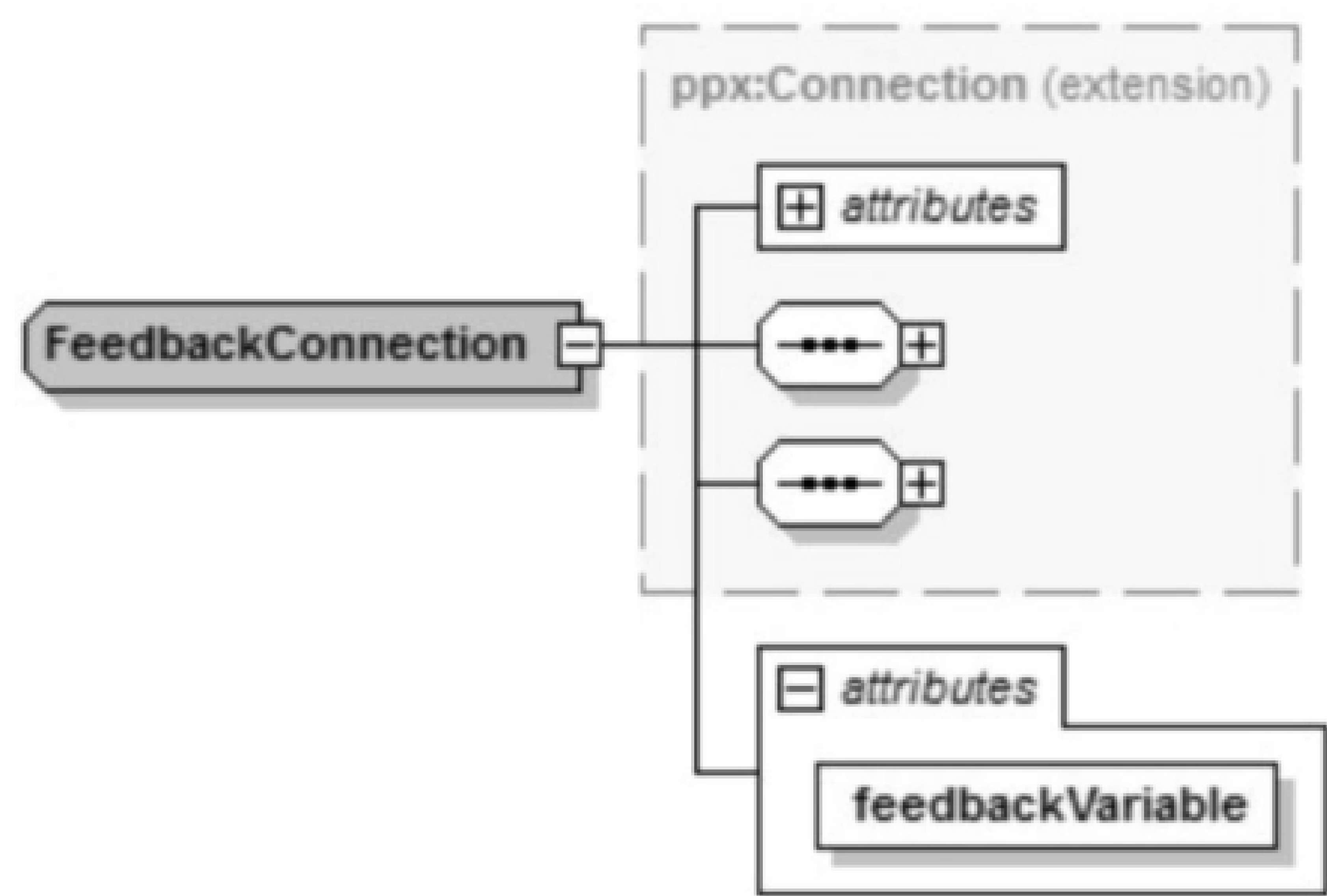


图 90 复合类型“FeedbackConnection”

“FeedbackConnection”基于抽象类型“Connection”，该类型已在 13.6.3 中介绍。该元素提供了“feedbackVariable”属性。

13.6.5 连接出点“ConnectionPointOut”

复合类型“ConnectionPointOut”表示实现前继-后继关系所需要的传出连接点。其内容如图 91 所示。

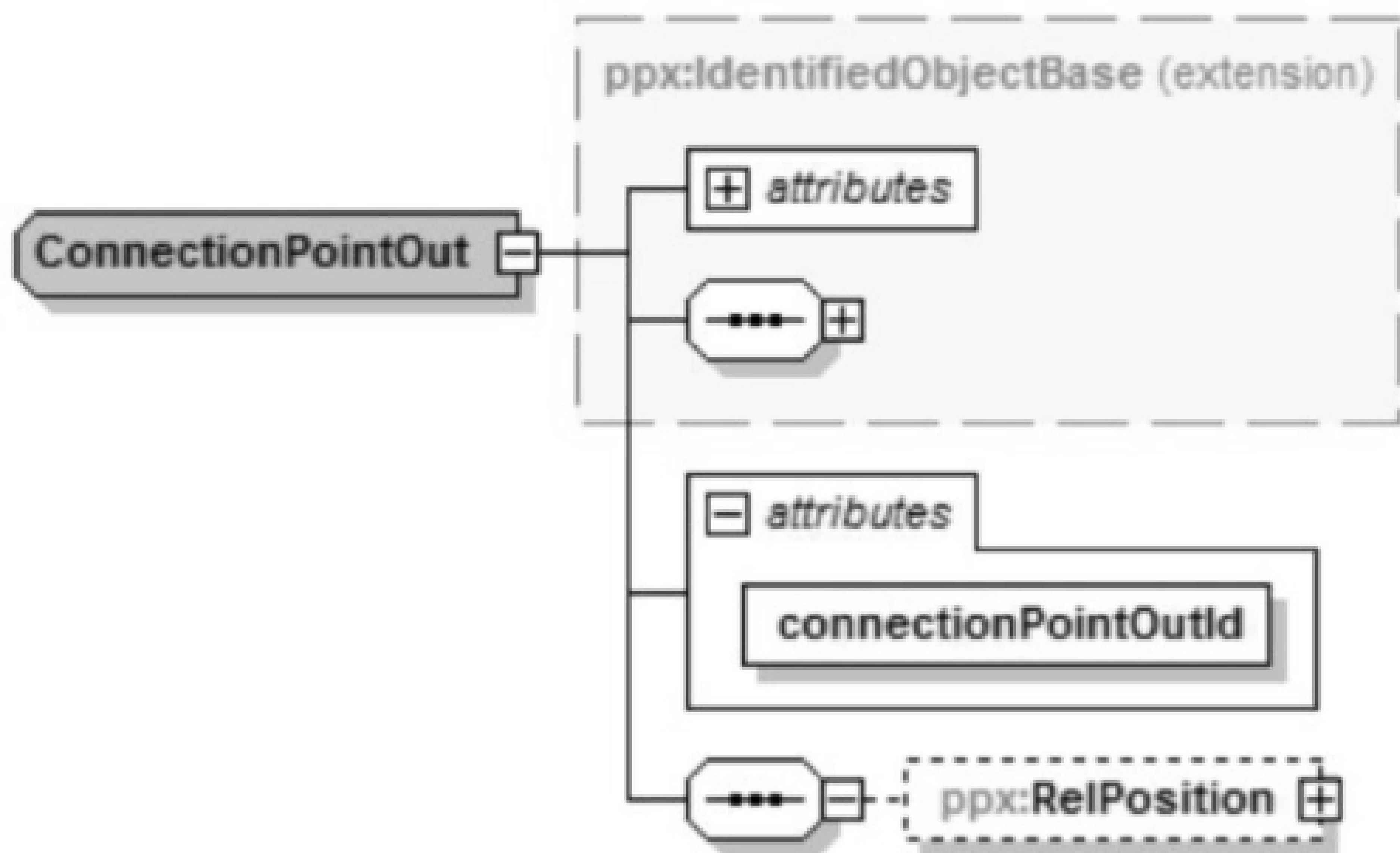


图 91 复合类型“ConnectionPointOut”

“ConnectionPointOut”基于抽象类型“IdentifiedObjectBase”，该类型已在 7.4.2 中介绍。属性“connectionPointOutId”用于通过另一个元素的“refConnectionPointOutId”属性引用此“ConnectionPointOut”元素，因此“connectionPointOutId”应是唯一的。对于 LD 或 FBD，则在每个“Ladder-Rung”或“FbdNetwork”中应是唯一的；对于 SFC，则在“SFC”中应是唯一的。“RelPosition”所属类型“XyDecimalValue”将在 15.1 中介绍。

14 资源声明

14.1 任务“StandardTask”

复合类型“StandardTask”表示 IEC 61131-3 中定义的任务。其内容如图 92 所示。

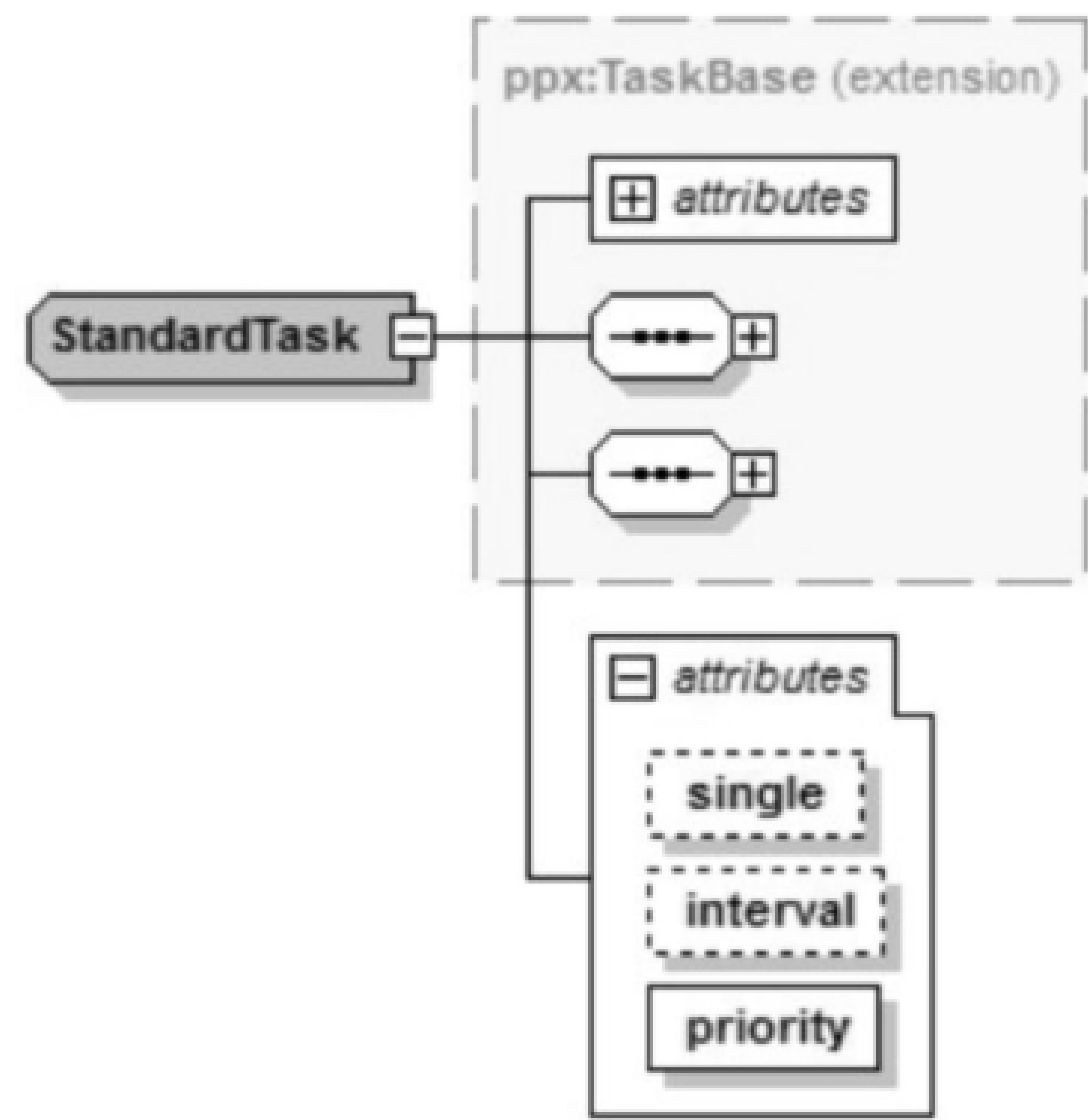


图 92 复合类型“StandardTask”

“StandardTask”基于抽象复合类型“TaskBase”，该类型已在 7.5.4 中介绍。可通过属性“interval”和“single”定义相应的周期性任务和边缘触发任务。可通过属性“priority”设置任务的调度优先级。

14.2 参数赋值“ParameterAssignment”

复合类型“ParameterAssignment”表示将实际值赋值给资源中使用的程序的形式参数，适用于所有的输入和输出参数。其内容如图 93 所示。

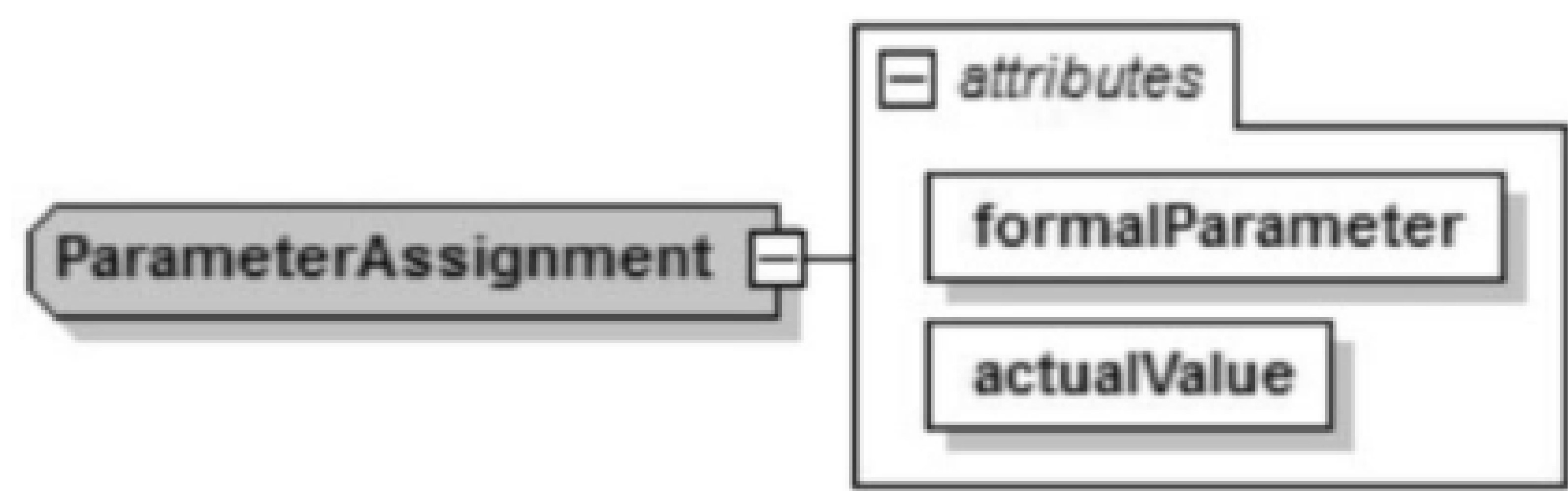


图 93 复合类型“ParameterAssignment”

属性“formalParameter”和“actualValue”为 string 类型。

15 其他

15.1 XY 数值“XyDecimalValue”

复合类型“XyDecimalValue”表示图形元素的位置或大小。其内容如图 94 所示。

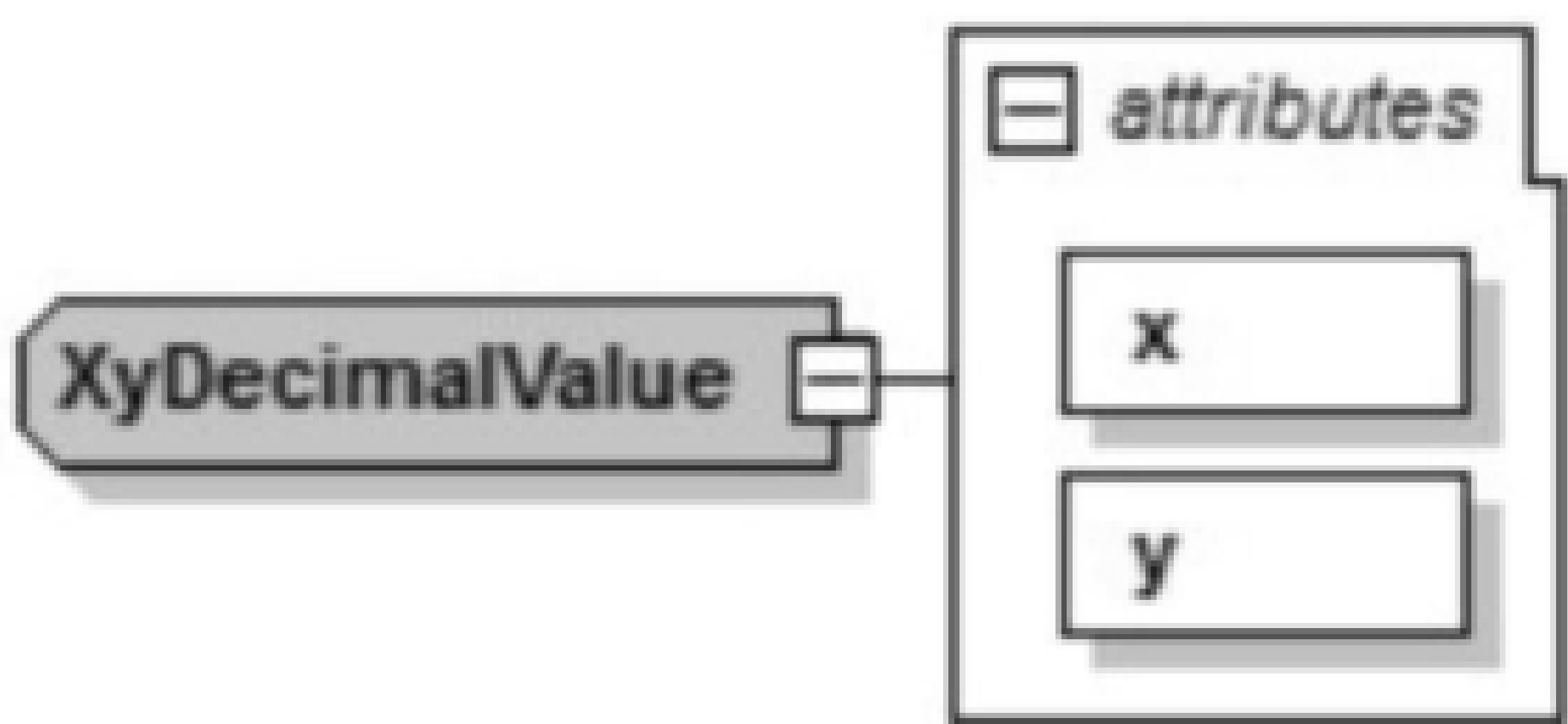


图 94 复合类型“XyDecimalValue”

属性“x”和“y”的值以小数的形式表示，详细内容已在 4.3 中介绍。

15.2 附加数据“AddData”

复合类型“AddData”可用于供应商特定的扩展，模式的扩展机制已在 4.4 中介绍。其内容如图 95 所示。

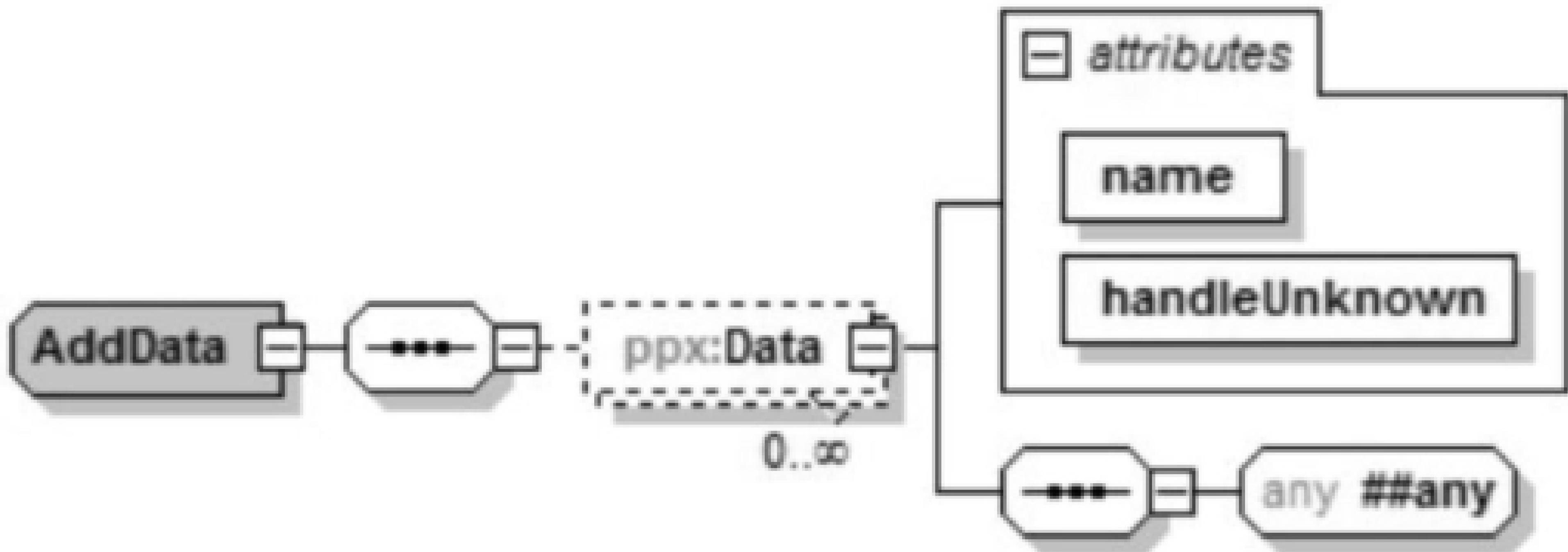


图 95 复合类型“AddData”

“Data”通过属性“name”进行标识，该属性引用“ContentHeader”中的“AddDataInfo”。通过使用“##any”，任何命名空间中的元素都可作为“Data”元素的内容。

属性“handleUnknown”的值可控制导入时的 PADT 如何处理“AddData”元素中的未知内容，有以下三种选项：

- 保留内容并将其包含在项目的下一次导出中；
- 丢弃内容；
- 由 PADT 根据其具体实现来决定如何处理该元素。

15.3 文本基类“TextBase”

“TextBase”是包含一个字符串的抽象复合类型。其内容如图 96 所示。



图 96 复合类型“TextBase”

15.4 简单文本“SimpleText”

复合类型“SimpleText”基于抽象复合类型“TaskBase”，该类型已在 15.3 中介绍。其内容如图 97 所示。



图 97 复合类型“SimpleText”

15.5 边沿触发修饰符“EdgeModifierType”

简单类型“EdgeModifierType”为 string 类型，其值为“none”“falling”或“rising”。

附 录 A  
(规范性)  
XML 交互格式模式定义

<CODE BEGINS>

```
<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xhtml="
http://www.w3.org/1999/xhtml" targetNamespace="www.iec.ch/public/TC65SC65BWG7TF10" elementFormDefault="qualified" attribute-
FormDefault="unqualified" version="1.0">
  <!--Main schema element "Project"-->
  <xsd:element name="Project">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="FileHeader">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="companyName" type="xsd:string" use="required"/>
            <xsd:attribute name="companyURL" type="xsd:anyURI" use="optional"/>
            <xsd:attribute name="productName" type="xsd:string" use="required"/>
            <xsd:attribute name="productVersion" type="xsd:string" use="required"/>
            <xsd:attribute name="productRelease" type="xsd:string" use="optional"/>
          </xsd:complexType>
        </xsd:element>
```

```

<xsd:element name="ContentHeader">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="CoordinateInfo" minOccurs="0">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="FbdScaling" type="ppx:XyDecimalValue" minOccurs="0"/>
            <xsd:element name="LdScaling" type="ppx:XyDecimalValue" minOccurs="0"/>
            <xsd:element name="SfcScaling" type="ppx:XyDecimalValue" minOccurs="0"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="AddDataInfo" minOccurs="0">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Info" minOccurs="0" maxOccurs="unbounded">
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="Description" type="ppx:TextBase" minOccurs="0"/>
                </xsd:sequence>
                <xsd:attribute name="name" type="xsd:anyURI" use="required"/>
                <xsd:attribute name="version" type="xsd:decimal"/>
                <xsd:attribute name="vendor" type="xsd:anyURI" use="required"/>
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

```

```

    </xsd:complexType>
  </xsd:element>
  <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string" use="required"/>
<xsd:attribute name="version" type="xsd:string" use="optional"/>
<xsd:attribute name="creationDateTime" type="xsd:dateTime" use="required"/>
<xsd:attribute name="modificationDateTime" type="xsd:dateTime" use="optional"/>
<xsd:attribute name="organization" type="xsd:string" use="optional"/>
<xsd:attribute name="author" type="xsd:string" use="optional"/>
<xsd:attribute name="language" type="xsd:language" use="optional"/>
</xsd:complexType>
</xsd:element>
<xsd:element name="Types">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="GlobalNamespace">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:TextualObjectBase">
              <xsd:sequence>
                <xsd:choice minOccurs="0" maxOccurs="unbounded">
                  <xsd:element name="NamespaceDecl" type="ppx:NamespaceDecl"/>
                  <xsd:element name="DataTypeDecl" type="ppx:UserDefinedTypeDecl"/>
                  <xsd:group ref="ppx:PouDecl"/>
                </xsd:choice>
              </xsd:sequence>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

```

```

        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="Instances">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Configuration" minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:TextualObjectBase">
              <xsd:sequence>
                <xsd:element name="Resource" minOccurs="0" maxOccurs="unbounded">
                  <xsd:complexType>
                    <xsd:complexContent>
                      <xsd:extension base="ppx:TextualObjectBase">
                        <xsd:sequence>
                          <xsd:element name="GlobalVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
                          <xsd:element name="Task" type="ppx:TaskBase" minOccurs="0" maxOccurs="unbounded"/>
                          <xsd:element name="ProgramInstance" minOccurs="0" maxOccurs="unbounded">
                            <xsd:complexType>
                              <xsd:complexContent>

```



```

maxOccurs="unbounded"/>
0" maxOccurs="unbounded"/>
maxOccurs="unbounded"/>

<xsd:extension base="ppx:TextualObjectBase">
  <xsd:sequence>
    <xsd:element name="InputAssignment" type="ppx:ParameterAssignment" minOccurs="0"
    maxOccurs="unbounded"/>
    <xsd:element name="OutputAssignment" type="ppx:ParameterAssignment" minOccurs="
    0" maxOccurs="unbounded"/>
    <xsd:element name="InOutAssignment" type="ppx:ParameterAssignment" minOccurs="0"
    maxOccurs="unbounded"/>
    <xsd:element name="FbInstTaskAssociation" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:attribute name="fbInstanceName" type="xsd:string" use="required"/>
        <xsd:attribute name="associatedTaskName" type="xsd:string" use=
        "required"/>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required"/>
  <xsd:attribute name="typeName" type="xsd:string" use="required"/>
  <xsd:attribute name="associatedTaskName" type="xsd:string" use="optional"/>
  <xsd:attribute name="evaluationOrder" type="xsd:unsignedLong" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>

```

```

        <xsd:attribute name="name" type="xsd:string" use="required"/>
        <xsd:attribute name="resourceTypeName" type="xsd:string" use="required"/>
    </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="GlobalVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="AccessVars" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="ppx:TextualObjectBase">
                <xsd:sequence>
                    <xsd:element name="AccessVariable" minOccurs="0" maxOccurs="unbounded">
                        <xsd:complexType>
                            <xsd:complexContent>
                                <xsd:extension base="ppx:TextualObjectBase">
                                    <xsd:sequence>
                                        <xsd:element name="Type" type="ppx:TypeRef"/>
                                    </xsd:sequence>
                                    <xsd:attribute name="alias" type="xsd:string" use="required"/>
                                    <xsd:attribute name="instancePathAndName" type="xsd:string" use="required"/>
                                    <xsd:attribute name="direction" use="optional">
                                        <xsd:simpleType>
                                            <xsd:restriction base="xsd:NMTOKEN">
                                                <xsd:enumeration value="readOnly"/>

```

```

        <xsd:enumeration value="readWrite"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="ConfigVars" minOccurs="0" maxOccurs="unbounded">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="ppx:TextualObjectBase">
                <xsd:sequence>
                    <xsd:element name="ConfigVariable" minOccurs="0" maxOccurs="unbounded">
                        <xsd:complexType>
                            <xsd:complexContent>
                                <xsd:extension base="ppx:TextualObjectBase">
                                    <xsd:sequence>
                                        <xsd:element name="Type" type="ppx:TypeRef"/>
                                        <xsd:element name="InitialValue" type="ppx:Value" minOccurs="0"/>

```

```

        <xsd:element name="Address" type="ppx:FixedAddressExpression" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="instancePathAndName" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
<xsd:element name="Documentation" type="ppx:TextBase" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="schemaVersion" type="xsd:decimal" use="required"/>
</xsd:complexType>

```

```

</xsd:element>
<!--Abstract types-->
<xsd:complexType name="TypeSpecBase" abstract="true"/>
<xsd:complexType name="InstantlyDefinableTypeSpecBase" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:TypeSpecBase"/>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="BehaviourRepresentationBase" abstract="true"/>
<xsd:complexType name="ProgrammingLanguageBase" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:BehaviourRepresentationBase"/>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IdentifiedObjectBase" abstract="true">
  <xsd:sequence>
    <xsd:element name="Documentation" type="ppx:TextBase" minOccurs="0"/>
    <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="globalId" type="xsd:ID" use="optional"/>
</xsd:complexType>
<xsd:complexType name="GraphicalObjectBase" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:IdentifiedObjectBase">
      <xsd:sequence>
        <xsd:element name="RelPosition" type="ppx:XyDecimalValue" minOccurs="0"/>

```

```

        <xsd:element name="Size" type="ppx:XyDecimalValue" minOccurs="0"/>
    </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CommonObjectBase" abstract="true">
    <xsd:complexContent>
        <xsd:extension base="ppx:GraphicalObjectBase"/>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FbdObjectBase" abstract="true">
    <xsd:complexContent>
        <xsd:extension base="ppx:GraphicalObjectBase"/>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="LdObjectBase" abstract="true">
    <xsd:complexContent>
        <xsd:extension base="ppx:GraphicalObjectBase"/>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SfcObjectBase" abstract="true">
    <xsd:complexContent>
        <xsd:extension base="ppx:GraphicalObjectBase"/>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="NetworkBase" abstract="true">

```

```

<xsd:complexContent>
  <xsd:extension base="ppx:GraphicalObjectBase">
    <xsd:attribute name="label" type="xsd:string" use="optional"/>
    <xsd:attribute name="evaluationOrder" type="xsd:unsignedLong" use="required"/>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="TextualObjectBase" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:IdentifiedObjectBase">
      <xsd:sequence>
        <xsd:element name="UsingDirective" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="pragma" type="xsd:string" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="NamespaceContentBase" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:attribute name="name" type="xsd:string" use="required"/>
      <xsd:attribute name="internal" type="xsd:boolean" use="optional" default="false"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

<xsd:complexType name="TaskBase" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--Namespace declaration-->
<xsd:complexType name="NamespaceDecl">
  <xsd:complexContent>
    <xsd:extension base="ppx:NamespaceContentBase">
      <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="NamespaceDecl" type="ppx:NamespaceDecl"/>
          <xsd:element name="DataTypeDecl" type="ppx:UserDefinedTypeDecl"/>
          <xsd:group ref="ppx:PouDecl"/>
        </xsd:choice>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--User-defined data type declaration-->
<xsd:complexType name="UserDefinedTypeDecl">
  <xsd:complexContent>
    <xsd:extension base="ppx:NamespaceContentBase">
      <xsd:sequence>

```



```

    <xsd:element name="UserDefinedTypeSpec" type="ppx:TypeSpecBase"/>
  </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ArrayTypeSpec">
  <xsd:complexContent>
    <xsd:extension base="ppx:InstantlyDefinableTypeSpecBase">
      <xsd:sequence>
        <xsd:element name="BaseType" type="ppx:TypeRef"/>
        <xsd:element name="DimensionSpec" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:choice>
              <xsd:element name="IndexRange">
                <xsd:complexType>
                  <xsd:attribute name="lower" type="xsd:string" use="required"/>
                  <xsd:attribute name="upper" type="xsd:string" use="required"/>
                </xsd:complexType>
              </xsd:element>
              <xsd:element name="VariableLength" type="xsd:boolean" fixed="true"/>
            </xsd:choice>
            <xsd:attribute name="dimensionNumber" type="xsd:unsignedInt" use="required"/>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DirectlyDerivedTypeSpec">
  <xsd:complexContent>
    <xsd:extension base="ppx:TypeSpecBase">
      <xsd:sequence>
        <xsd:element name="BaseType" type="ppx:TypeRef"/>
        <xsd:element name="InitialValue" type="ppx:Value" minOccurs="0"/>
        <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="EnumTypeSpec">
  <xsd:complexContent>
    <xsd:extension base="ppx:TypeSpecBase">
      <xsd:sequence>
        <xsd:element name="Enumerator" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="Documentation" type="ppx:TextBase" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:string" use="required"/>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

        <xsd:element name="InitialValue" type="xsd:string" minOccurs="0"/>
        <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
    </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="EnumTypeWithNamedValueSpec">
    <xsd:complexContent>
        <xsd:extension base="ppx:TypeSpecBase">
            <xsd:sequence>
                <xsd:element name="Enumerator" maxOccurs="unbounded">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="Documentation" type="ppx:TextBase" minOccurs="0"/>
                        </xsd:sequence>
                        <xsd:attribute name="name" type="xsd:string" use="required"/>
                        <xsd:attribute name="value" type="xsd:string" use="required"/>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="BaseType" type="ppx:ElementaryType"/>
                <xsd:element name="InitialValue" type="xsd:string" minOccurs="0"/>
                <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

```

```

<xsd:complexType name="StructTypeSpec">
  <xsd:complexContent>
    <xsd:extension base="ppx:TypeSpecBase">
      <xsd:sequence>
        <xsd:element name="Member" type="ppx:VariableDecl" minOccurs="1" maxOccurs="unbounded"/>
        <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="overlap" type="xsd:boolean" use="optional" default="false"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SubrangeTypeSpec">
  <xsd:complexContent>
    <xsd:extension base="ppx:TypeSpecBase">
      <xsd:sequence>
        <xsd:element name="Range">
          <xsd:complexType>
            <xsd:attribute name="lower" type="xsd:string" use="required"/>
            <xsd:attribute name="upper" type="xsd:string" use="required"/>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="BaseType" type="ppx:TypeRef"/>
        <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>

```

```

</xsd:complexType>
<xsd:complexType name="ReferenceTypeSpec">
  <xsd:complexContent>
    <xsd:extension base="ppx:InstantlyDefinableTypeSpecBase">
      <xsd:sequence>
        <xsd:element name="ReferenceTo" type="ppx:TypeRef"/>
        <xsd:element name="AddData" type="ppx:AddData" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:simpleType name="ElementaryType">
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="BOOL"/>
        <xsd:enumeration value="BYTE"/>
        <xsd:enumeration value="WORD"/>
        <xsd:enumeration value="DWORD"/>
        <xsd:enumeration value="LWORD"/>
        <xsd:enumeration value="SINT"/>
        <xsd:enumeration value="INT"/>
        <xsd:enumeration value="DINT"/>
        <xsd:enumeration value="LINT"/>
        <xsd:enumeration value="USINT"/>
        <xsd:enumeration value="UINT"/>

```

```

    <xsd:enumeration value="UDINT"/>
    <xsd:enumeration value="ULINT"/>
    <xsd:enumeration value="REAL"/>
    <xsd:enumeration value="LREAL"/>
    <xsd:enumeration value="TIME"/>
    <xsd:enumeration value="LTIME"/>
    <xsd:enumeration value="DT"/>
    <xsd:enumeration value="LDT"/>
    <xsd:enumeration value="TOD"/>
    <xsd:enumeration value="LTOD"/>
    <xsd:enumeration value="STRING"/>
    <xsd:enumeration value="WSTRING"/>
    <xsd:enumeration value="CHAR"/>
    <xsd:enumeration value="WCHAR"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="STRING\[ [A-Za-z0-9_]+\]" />
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="WSTRING\[ [A-Za-z0-9_]+\]" />
  </xsd:restriction>
</xsd:simpleType>

```

```

    </xsd:union>
</xsd:simpleType>
<!--POU declaration-->
<xsd:group name="PouDecl">
    <xsd:choice>
        <xsd:element name="Program" type="ppx:Program"/>
        <xsd:element name="FunctionBlock" type="ppx:FunctionBlock"/>
        <xsd:element name="Class" type="ppx:Class"/>
        <xsd:element name="Function" type="ppx:Function"/>
        <xsd:element name="Interface" type="ppx:Interface"/>
    </xsd:choice>
</xsd:group>
<xsd:complexType name="Program">
    <xsd:complexContent>
        <xsd:extension base="ppx:NamespaceContentBase">
            <xsd:sequence>
                <xsd:element name="AccessVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element name="GlobalVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0" maxOccurs="1"/>
                <xsd:element name="ExternalVars" type="ppx:ExternalVarList" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element name="Vars" type="ppx:VarListWithAccessSpec" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element name="TempVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element name="MainBody" type="ppx:Body" minOccurs="0" maxOccurs="1"/>
                <xsd:element name="Action" type="ppx:Action" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element name="Transition" type="ppx:NamedTransition" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>

```

```

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FunctionBlock">
  <xsd:complexContent>
    <xsd:extension base="ppx:NamespaceContentBase">
      <xsd:sequence>
        <xsd:element name="Extends" type="ppx:TypeRef" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="Implements" type="ppx:TypeRef" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="ExternalVars" type="ppx:ExternalVarList" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="Vars" type="ppx:VarListWithAccessSpec" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="TempVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="Method" type="ppx:Method" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="MainBody" type="ppx:Body" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="Action" type="ppx:Action" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="Transition" type="ppx:NamedTransition" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="abstract" type="xsd:boolean" use="optional" default="false"/>
      <xsd:attribute name="final" type="xsd:boolean" use="optional" default="false"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Class">
  <xsd:complexContent>
    <xsd:extension base="ppx:NamespaceContentBase">

```



```

<xsd:sequence>
  <xsd:element name="Extends" type="ppx:TypeRef" minOccurs="0" maxOccurs="1"/>
  <xsd:element name="Implements" type="ppx:TypeRef" minOccurs="0" maxOccurs="unbounded"/>
  <xsd:element name="ExternalVars" type="ppx:ExternalVarList" minOccurs="0" maxOccurs="unbounded"/>
  <xsd:element name="Vars" type="ppx:VarListWithAccessSpec" minOccurs="0" maxOccurs="unbounded"/>
  <xsd:element name="Method" type="ppx:Method" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="abstract" type="xsd:boolean" use="optional" default="false"/>
<xsd:attribute name="final" type="xsd:boolean" use="optional" default="false"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Function">
  <xsd:complexContent>
    <xsd:extension base="ppx:NamespaceContentBase">
      <xsd:sequence>
        <xsd:element name="ResultType" type="ppx:TypeRef" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="ExternalVars" type="ppx:ExternalVarList" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="TempVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="MainBody" type="ppx:BodyWithoutSFC" minOccurs="0" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Interface">

```

```

<xsd:complexContent>
  <xsd:extension base="ppx:NamespaceContentBase">
    <xsd:sequence>
      <xsd:element name="Extends" type="ppx:TypeRef" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="Method" type="ppx:MethodPrototype" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Action">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="Body" type="ppx:Body"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="NamedTransition">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="Condition" type="ppx:Predicate"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodPrototype">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="ResultType" type="ppx:TypeRef" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Method">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="ResultType" type="ppx:TypeRef" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="Parameters" type="ppx:ParameterSet" minOccurs="0"/>
        <xsd:element name="TempVars" type="ppx:VarList" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="Body" type="ppx:BodyWithoutSFC" minOccurs="0" maxOccurs="1"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
      <xsd:attribute name="abstract" type="xsd:boolean" use="optional" default="false"/>
      <xsd:attribute name="final" type="xsd:boolean" use="optional" default="false"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

    <xsd:attribute name="override" type="xsd:boolean" use="optional" default="false"/>
    <xsd:attribute name="accessSpecifier" type="ppx:AccessSpecifiers" use="required"/>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ParameterSet">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="InoutVars">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded">
              <xsd:complexType>
                <xsd:complexContent>
                  <xsd:extension base="ppx:VariableDecl">
                    <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/>
                  </xsd:extension>
                </xsd:complexContent>
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:choice>
  </xsd:sequence>
  <xsd:element name="InputVars">
    <xsd:complexType>
      <xsd:sequence>

```

```

<xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="ppx:VariableDecl">
        <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/>
        <xsd:attribute name="edgeDetection" type="ppx:EdgeModifierType" use="optional" default="none"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="retain" type="xsd:boolean" use="optional" default="false"/>
<xsd:attribute name="non_retain" type="xsd:boolean" use="optional" default="false"/>
</xsd:complexType>
</xsd:element>
<xsd:element name="OutputVars">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Variable" minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:VariableDecl">
              <xsd:attribute name="orderWithinParamSet" type="xsd:unsignedInt" use="required"/>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

```

```

        </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="retain" type="xsd:boolean" use="optional" default="false"/>
    <xsd:attribute name="non_retain" type="xsd:boolean" use="optional" default="false"/>
</xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="VarListWithAccessSpec">
    <xsd:complexContent>
        <xsd:extension base="ppx:VarList">
            <xsd:attribute name="accessSpecifier" type="ppx:AccessSpecifiers" use="required"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:simpleType name="AccessSpecifiers">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="private"/>
        <xsd:enumeration value="protected"/>
        <xsd:enumeration value="internal"/>
        <xsd:enumeration value="public"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="Body">
    <xsd:complexContent>

```

```

    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="BodyContent" type="ppx:BehaviourRepresentationBase" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="BodyWithoutSFC">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="BodyContent" type="ppx:ProgrammingLanguageBase" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Predicate">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="PredicateContent" type="ppx:ProgrammingLanguageBase"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--Variable declaration-->

```

```

    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="BodyContent" type="ppx:BehaviourRepresentationBase" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="BodyWithoutSFC">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="BodyContent" type="ppx:ProgrammingLanguageBase" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Predicate">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="PredicateContent" type="ppx:ProgrammingLanguageBase"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--Variable declaration-->

```



```

    <xsd:element name="InitialValue" type="ppx:Value" minOccurs="0"/>
    <xsd:element name="Address" type="ppx:AddressExpression" minOccurs="0"/>
  </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="VariableDeclPlain">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:sequence>
        <xsd:element name="Type" type="ppx:TypeRef"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="TypeRef">
  <xsd:choice>
    <xsd:element name="TypeName" type="xsd:string"/>
    <xsd:element name="InstantlyDefinedType" type="ppx:InstantlyDefinableTypeSpecBase"/>
  </xsd:choice>
</xsd:complexType>
<xsd:complexType name="Value">
  <xsd:choice>
    <xsd:element name="SimpleValue">
      <xsd:complexType>

```

```

    <xsd:attribute name="value" type="xsd:string" use="optional"/>
  </xsd:complexType>
</xsd:element>
<xsd:element name="ArrayValue">
  <xsd:complexType>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="Value">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:Value">
              <xsd:attribute name="repetitionValue" type="xsd:string" use="optional" default="1"/>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="StructValue">
  <xsd:complexType>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="Value">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:Value">
              <xsd:attribute name="member" type="xsd:string" use="required"/>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

```

```

        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:complexType>
<xsd:complexType name="AddressExpression">
  <xsd:complexContent>
    <xsd:extension base="ppx:FixedAddressExpression">
      <xsd:attribute name="notYetFixed" type="xsd:boolean" use="optional" default="false"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FixedAddressExpression">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextualObjectBase">
      <xsd:attribute name="location" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:NMTOKEN">
            <xsd:enumeration value="I"/>
            <xsd:enumeration value="Q"/>
            <xsd:enumeration value="M"/>
          </xsd:restriction>

```

```

    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="size" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
        <xsd:enumeration value="X"/>
        <xsd:enumeration value="B"/>
        <xsd:enumeration value="W"/>
        <xsd:enumeration value="D"/>
        <xsd:enumeration value="L"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="address" type="xsd:string" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!--Behaviour representation-->
<xsd:complexType name="IL">
  <xsd:complexContent>
    <xsd:extension base="ppx:ProgrammingLanguageBase">
      <xsd:sequence>
        <xsd:element name="IL" type="xsd:string"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>

```

```

</xsd:complexType>
<xsd:complexType name="ST">
  <xsd:complexContent>
    <xsd:extension base="ppx:ProgrammingLanguageBase">
      <xsd:sequence>
        <xsd:element name="ST" type="xsd:string"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FBD">
  <xsd:complexContent>
    <xsd:extension base="ppx:ProgrammingLanguageBase">
      <xsd:sequence>
        <xsd:element name="Network" type="ppx:FbdNetwork" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FbdNetwork">
  <xsd:complexContent>
    <xsd:extension base="ppx:NetworkBase">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="CommonObject" type="ppx:CommonObjectBase"/>
        <xsd:element name="FbdObject" type="ppx:FbdObjectBase"/>
      </xsd:choice>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="LD">
  <xsd:complexContent>
    <xsd:extension base="ppx:ProgrammingLanguageBase">
      <xsd:sequence>
        <xsd:element name="Rung" type="ppx:LadderRung" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="LadderRung">
  <xsd:complexContent>
    <xsd:extension base="ppx:NetworkBase">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="CommonObject" type="ppx:CommonObjectBase"/>
        <xsd:element name="LdObject" type="ppx:LdObjectBase"/>
        <xsd:element name="FbdObject" type="ppx:FbdObjectBase"/>
      </xsd:choice>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SFC">
  <xsd:complexContent>
    <xsd:extension base="ppx:BehaviourRepresentationBase">

```

```

    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="CommonObject" type="ppx:CommonObjectBase"/>
      <xsd:element name="LdObject" type="ppx:LdObjectBase"/>
      <xsd:element name="FbdObject" type="ppx:FbdObjectBase"/>
      <xsd:element name="SfcObject" type="ppx:SfcObjectBase"/>
    </xsd:choice>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!--Graphical behaviour representation - Common elements-->
<xsd:complexType name="Comment">
  <xsd:complexContent>
    <xsd:extension base="ppx:CommonObjectBase">
      <xsd:sequence>
        <xsd:element name="Content" type="ppx:TextBase"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Connector">
  <xsd:complexContent>
    <xsd:extension base="ppx:CommonObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="label" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Continuation">
  <xsd:complexContent>
    <xsd:extension base="ppx:CommonObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="label" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ActionBlocks">
  <xsd:complexContent>
    <xsd:extension base="ppx:CommonObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn"/>
        <xsd:element name="ActionBlock" minOccurs="0" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:complexContent>
              <xsd:extension base="ppx:GraphicalObjectBase">
                <xsd:sequence>
                  <xsd:element name="ActionQualifier" minOccurs="0">
                    <xsd:complexType>
                      <xsd:attribute name="qualifier" use="optional" default="N">

```



```

    <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN">
        <xsd:enumeration value="P1"/>
        <xsd:enumeration value="N"/>
        <xsd:enumeration value="P0"/>
        <xsd:enumeration value="R"/>
        <xsd:enumeration value="S"/>
        <xsd:enumeration value="L"/>
        <xsd:enumeration value="D"/>
        <xsd:enumeration value="P"/>
        <xsd:enumeration value="DS"/>
        <xsd:enumeration value="DL"/>
        <xsd:enumeration value="SD"/>
        <xsd:enumeration value="SL"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="duration" type="xsd:string" use="optional"/>
</xsd:complexType>
</xsd:element>
<xsd:choice>
  <xsd:element name="Inline" type="ppx:Body"/>
  <xsd:element name="ComplexOperand" type="xsd:string"/>
  <xsd:element name="ReferenceName" type="xsd:string"/>
</xsd:choice>
</xsd:sequence>

```

```

        <xsd:attribute name="indicator" type="xsd:string" use="optional"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!--Graphical behaviour representation - FBD elements-->
<xsd:complexType name="Block">
  <xsd:complexContent>
    <xsd:extension base="ppx:FbdObjectBase">
      <xsd:sequence>
        <xsd:element name="InOutVariables" minOccurs="0">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="InOutVariable" minOccurs="0" maxOccurs="unbounded">
                <xsd:complexType>
                  <xsd:complexContent>
                    <xsd:extension base="ppx:IdentifiedObjectBase">
                      <xsd:sequence>
                        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
                        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
                      </xsd:sequence>
                    <xsd:attributeGroup ref="ppx:graphicalFormalParameterCommon"/>

```

```

        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="InputVariables" minOccurs="0">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="InputVariable" minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:IdentifiedObjectBase">
              <xsd:sequence>
                <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
              </xsd:sequence>
              <xsd:attributeGroup ref="ppx:graphicalFormalParameterCommon"/>
              <xsd:attribute name="edge" type="ppx:EdgeModifierType" use="optional" default="none"/>
              <xsd:attribute name="suppressName" type="xsd:boolean" use="optional" default="false"/>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>

```

```

</xsd:element>
<xsd:element name="OutputVariables" minOccurs="0">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="OutputVariable" minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="ppx:IdentifiedObjectBase">
              <xsd:sequence>
                <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
              </xsd:sequence>
              <xsd:attributeGroup ref="ppx:graphicalFormalParameterCommon"/>
              <xsd:attribute name="suppressName" type="xsd:boolean" use="optional" default="false"/>
            </xsd:extension>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="typeName" type="xsd:string" use="required"/>
<xsd:attribute name="instanceName" type="xsd:string" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

```

```

<xsd:attributeGroup name="graphicalFormalParameterCommon">
  <xsd:attribute name="parameterName" type="xsd:string" use="required"/>
  <xsd:attribute name="negated" type="xsd:boolean" use="optional" default="false"/>
</xsd:attributeGroup>
<xsd:complexType name="DataSource">
  <xsd:complexContent>
    <xsd:extension base="ppx:FbdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="identifier" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DataSink">
  <xsd:complexContent>
    <xsd:extension base="ppx:FbdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="identifier" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Unconnected">
  <xsd:complexContent>

```

```

    <xsd:extension base="ppx:FbdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="complexIdentifier" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Jump">
  <xsd:complexContent>
    <xsd:extension base="ppx:FbdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="targetNetworkLabel" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Return">
  <xsd:complexContent>
    <xsd:extension base="ppx:FbdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>

```

```

    </xsd:complexContent>
</xsd:complexType>
<!--Graphical behaviour representation - LD elements-->
<xsd:complexType name="LeftPowerRail">
  <xsd:complexContent>
    <xsd:extension base="ppx:LdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="RightPowerRail">
  <xsd:complexContent>
    <xsd:extension base="ppx:LdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Coil">
  <xsd:complexContent>
    <xsd:extension base="ppx:LdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>

```

```

    <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="negated" type="xsd:boolean" use="optional" default="false"/>
  <xsd:attribute name="edge" type="ppx:EdgeModifierType" use="optional" default="none"/>
  <xsd:attribute name="latch" use="optional" default="none">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="none"/>
        <xsd:enumeration value="set"/>
        <xsd:enumeration value="reset"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="operand" type="xsd:string" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Contact">
  <xsd:complexContent>
    <xsd:extension base="ppx:LdObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="negated" type="xsd:boolean" use="optional" default="false"/>
      <xsd:attribute name="edge" type="ppx:EdgeModifierType" use="optional" default="none"/>

```



```

        <xsd:attribute name="operand" type="xsd:string" use="required"/>
    </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CompareContact">
    <xsd:complexContent>
        <xsd:extension base="ppx:LdObjectBase">
            <xsd:sequence>
                <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
                <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
                <xsd:element name="Type" type="ppx:ElementaryType" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="compareOperator" use="required">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:enumeration value=">"/>
                        <xsd:enumeration value=">="/>
                        <xsd:enumeration value="="/>
                        <xsd:enumeration value="<="/>
                        <xsd:enumeration value="<"/>
                        <xsd:enumeration value="<>"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:attribute>
            <xsd:attribute name="operand1" type="xsd:string" use="required"/>
            <xsd:attribute name="operand2" type="xsd:string" use="required"/>

```

```

    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--Graphical behaviour representation - SFC elements-->
<xsd:complexType name="Step">
  <xsd:complexContent>
    <xsd:extension base="ppx:SfcObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
        <xsd:element name="ConnectionPointOutAction" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required"/>
      <xsd:attribute name="initialStep" type="xsd:boolean" use="optional" default="false"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Transition">
  <xsd:complexContent>
    <xsd:extension base="ppx:SfcObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
        <xsd:element name="Condition" minOccurs="0">
          <xsd:complexType>
            <xsd:choice>

```

```

    <xsd:element name="Reference">
      <xsd:complexType>
        <xsd:attribute name="name" type="xsd:string" use="required"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="GraphicalPredicate">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn"/>
          <xsd:element name="GraphicalExpression" type="ppx:NetworkBase"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="TextualPredicate" type="ppx:Predicate"/>
  </xsd:choice>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SelectionDivergence">
  <xsd:complexContent>
    <xsd:extension base="ppx:SfcObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>

```

```

    <xsd:element name="ConnectionPointOut" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:complexContent>
          <xsd:extension base="ppx:ConnectionPointOut">
            <xsd:attribute name="priority" type="xsd:unsignedLong" use="optional"/>
          </xsd:extension>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="mutualExclusion" type="xsd:boolean" use="optional" default="false"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SelectionConvergence">
  <xsd:complexContent>
    <xsd:extension base="ppx:SfcObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SimultaneousDivergence">
  <xsd:complexContent>

```

```

<xsd:extension base="ppx:SfcObjectBase">
  <xsd:sequence>
    <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
    <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SimultaneousConvergence">
  <xsd:complexContent>
    <xsd:extension base="ppx:SfcObjectBase">
      <xsd:sequence>
        <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--Graphical behaviour representation - Connections-->
<xsd:complexType name="ConnectionPointIn">
  <xsd:complexContent>
    <xsd:extension base="ppx:IdentifiedObjectBase">
      <xsd:sequence>
        <xsd:element name="RelPosition" type="ppx:XyDecimalValue" minOccurs="0"/>
        <xsd:choice>
          <xsd:element name="Connection" type="ppx:Connection" minOccurs="0" maxOccurs="unbounded"/>

```

```

        <xsd:element name="FeedbackConnection" type="ppx:FeedbackConnection" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:choice>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Connection">
    <xsd:complexContent>
        <xsd:extension base="ppx:IdentifiedObjectBase">
            <xsd:sequence>
                <xsd:element name="RelPosition" type="ppx:XyDecimalValue" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
            <xsd:attribute name="refConnectionPointOutId" type="xsd:unsignedLong" use="required"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FeedbackConnection">
    <xsd:complexContent>
        <xsd:extension base="ppx:Connection">
            <xsd:attribute name="feedbackVariable" type="xsd:string" use="required"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ConnectionPointOut">
    <xsd:complexContent>
        <xsd:extension base="ppx:IdentifiedObjectBase">

```

```

    <xsd:sequence>
      <xsd:element name="RelPosition" type="ppx:XyDecimalValue" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="connectionPointOutId" type="xsd:unsignedLong" use="required"/>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!--Resource declaration -->
<xsd:complexType name="StandardTask">
  <xsd:complexContent>
    <xsd:extension base="ppx:TaskBase">
      <xsd:attribute name="single" type="xsd:string" use="optional"/>
      <xsd:attribute name="interval" type="xsd:string" use="optional"/>
      <xsd:attribute name="priority" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:integer">
            <xsd:minInclusive value="0"/>
            <xsd:maxInclusive value="65535"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ParameterAssignment">
  <xsd:attribute name="formalParameter" type="xsd:string" use="required"/>

```

```

    <xsd:attribute name="actualValue" type="xsd:string" use="required"/>
</xsd:complexType>
<!-- Miscellaneous -->
<xsd:complexType name="XyDecimalValue">
    <xsd:attribute name="x" type="xsd:decimal" use="required"/>
    <xsd:attribute name="y" type="xsd:decimal" use="required"/>
</xsd:complexType>
<xsd:complexType name="AddData">
    <xsd:sequence>
        <xsd:element name="Data" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:any namespace="#" #any" processContents="lax"/>
                </xsd:sequence>
                <xsd:attribute name="name" type="xsd:anyURI" use="required"/>
                <xsd:attribute name="handleUnknown" use="required">
                    <xsd:simpleType>
                        <xsd:restriction base="xsd:NMTOKEN">
                            <xsd:enumeration value="preserve"/>
                            <xsd:enumeration value="discard"/>
                            <xsd:enumeration value="implementation"/>
                        </xsd:restriction>
                    </xsd:simpleType>
                </xsd:attribute>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

```



```
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="TextBase" abstract="true"/>
<xsd:complexType name="SimpleText" mixed="true">
  <xsd:complexContent>
    <xsd:extension base="ppx:TextBase"/>
  </xsd:complexContent>
</xsd:complexType>
<xsd:simpleType name="EdgeModifierType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="none"/>
    <xsd:enumeration value="falling"/>
    <xsd:enumeration value="rising"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:schema>

<CODE ENDS>
```

## 附录 B

(资料性)

### 推荐方案

#### B.1 概述

如果只有标准模式用于 PLC 工程中,那么 PLC 工程的 XML 文件中只需要指明其引用的附录 A 中定义的标准模式文件,如图 B.1 所示。



图 B.1 仅包含标准模式

如果 PLC 工作中使用了“AddData”扩展机制,PLC 工程的 XML 文件宜引用附录 B.2 中定义的模式,如图 B.2 所示。

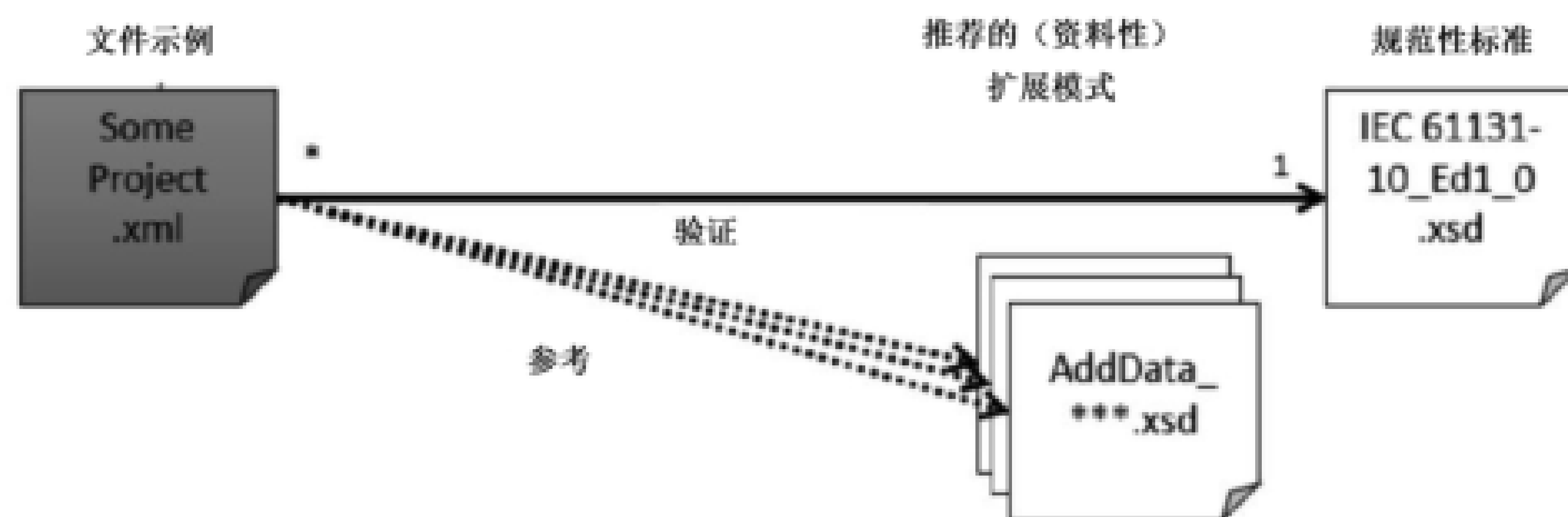


图 B.2 供应商相关的扩展“AddData”

如果 PLC 工作中使用了供应商相关的语言扩展,那么供应商宜定义他们自己的模式文件并在其中导入附录 A 定义的标准模式。PLC 工程的 XML 文件宜引用该供应商的模式文件(如图 B.3 所示)。供应商可在他们的模式文件中导入 B.3 中推荐的模式并在 XML 文件中使用。

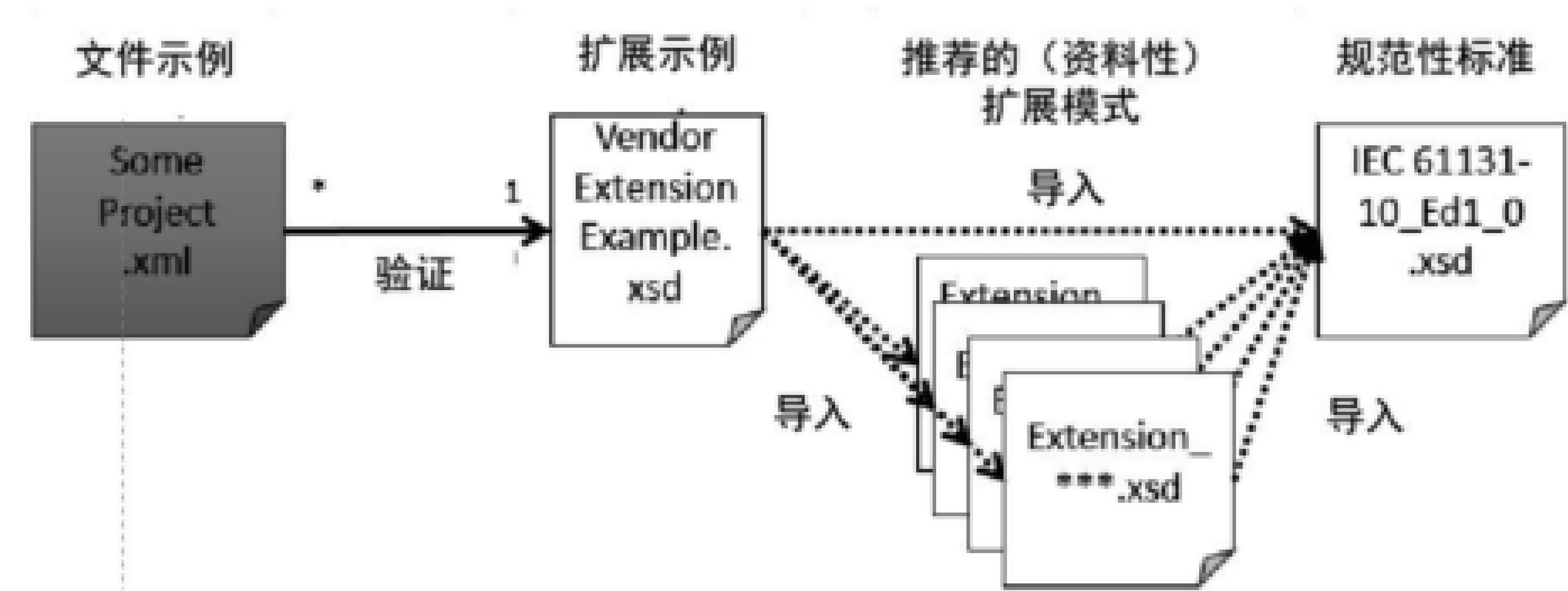


图 B.3 供应商相关的扩展(抽象组合类型)

B.2 AddData 推荐模式

AddData\_AnchoredComment.xsd

```
<CODE BEGINS>

<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:import namespace="www.iec.ch/public/TC65SC65BWG7TF10" schemaLocation="IEC61131_10_Ed1_0.xsd" />
  <xsd:element name="AnchoredComment">
    <xsd:annotation>
      <xsd:documentation>
        AnchoredComment
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
</xsd:schema>
```

=====

'AnchoredComment' is a comment which is located to a graphic element (FBD, LD or SFC):

The 'AnchoredComment' object is placed at a position relative to its parent.

The offset of the position is given as relative coordinates, thus it may have negative values.

The comment is just a comment, it has no semantical information to the program code.

Expressing AnchoredComment via "addData" by the following mechanism:

Each graphic element (derived from 'GraphicalObjectBase') may be extended by 'AnchoredComment' child node

```
</xsd:documentation>
```

```
</xsd:annotation>
```

```
<xsd:complexType>
```

```
<xsd:sequence>
```

```
<xsd:element name="RelPosition" type="ppx:XyDecimalValue" minOccurs="1" maxOccurs="1"/>
```

```
<xsd:element name="Size" type="ppx:XyDecimalValue" minOccurs="0" maxOccurs="1"/>
```

```
<xsd:element name="Content" type="ppx:TextBase" minOccurs="1" maxOccurs="1"/>
```

```
</xsd:sequence>
```

```
</xsd:complexType>
```

```
</xsd:element>
```

```
</xsd:schema>
```

```
<CODE ENDS>
```

## AddData\_EvaluationPriority.xsd

<CODE BEGINS>

```
<? xml version="1.0" encoding="utf-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:element name="EvaluationPriority">
    <xsd:annotation>
      <xsd:documentation>
        EvaluationPriority
        =====
        'EvaluationPriority' defines a priority among different 'ppx:Block' elements.
        It defines a rule in which order 'ppx:Block' shall be evaluated (executed).
        The rule is applicable only for those blocks with an evaluation priority specified.
        The scope of this rule is in FBD a 'ppx:FbdNetwork' and in LD a 'ppx:LadderRung'.
        A block with a smaller number shall be evaluated earlier within its superior network.
        Within a network each priority shall be unique, i.e. two blocks shall not have the same priority value.

        Expressing EvaluationPriority via "addData" by the following mechanism:
        Each 'ppx:Block' may be extended by 'EvaluationPriority' child node.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:attribute name="priorityInNetwork" type="xsd:unsignedLong" />
    </xsd:complexType>
```

</xsd:element>

</xsd:schema>

<CODE ENDS>

## AddData\_HideFormalParameter.xsd

<CODE BEGINS>

```
<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:element name="HiddenFormalParameter">
    <xsd:annotation>
      <xsd:documentation>Hide a FormalParameter (do not display formal parameter of a FB invocation) via "addData".
        Use 'hiddenFormalParameter' as a child node of 'inputVariables'->'variable'->'addData'.
        The attribute 'hideable' defines whether this formal parameter can be hidden or not.
        The attribute 'hide' defines whether this formal parameter (at this FU/FB invocation) is hidden.
        The connection of the formal parameter with a variable (or literal) is expressed via 'connectionPointIn' anyhow, although
        in case of 'hide=true' the connected variable or literal is not displayed, too.
        Note: the attribute 'suppressName' (defined by IEC 6113-10) of the node type 'variable' defines only whether the name of the formal pa-
        rameter is displayed or not.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:attribute name="hideable" type="xsd:boolean" default="false"/>
      <xsd:attribute name="hide" type="xsd:boolean" use="optional"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

<CODE ENDS>

## AddData\_Worksheet.xsd

128

&lt;CODE BEGINS&gt;

&lt;? xml version="1.0" encoding="UTF-8"? &gt;

&lt;xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"

elementFormDefault="qualified" attributeFormDefault="unqualified"&gt;

&lt;xsd:element name="Worksheet"&gt;

&lt;xsd:annotation&gt;

&lt;xsd:documentation&gt;

Worksheet

=====

Worksheet is a logical unit to structure the code within a POU;

One or more networks (or ST statements) build a worksheet.

One or more worksheets build the main body of a POU.

One or more worksheets build a method.

Each worksheet has a name (unique within this body).

Expressing worksheet via "addData" by the following mechanism;

Each element "ST", "IL", "FBD", "LD" and "SFC" is extended by 'worksheet' child node

&lt;/xsd:documentation&gt;

&lt;/xsd:annotation&gt;

&lt;xsd:complexType&gt;

&lt;xsd:attribute name="worksheetName" type="xsd:string" use="required"/&gt;

<xsd:attribute name="evaluationOrder" type="xsd:decimal" use="required"/> <!-- Evaluation/Execution of lowest number first, starting with 0. -->



```

    </xsd:complexType>
  </xsd:element>
</xsd:schema>

```

<CODE ENDS>

### B.3 抽象复合类型的推荐方案

<CODE BEGINS>

```

<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10"
  xmlns:rxt="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  targetNamespace="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:import namespace="www.iec.ch/public/TC65SC65BWG7TF10" schemaLocation="IEC61131_10_Ed1_0.xsd"/>
  <xsd:annotation>
    <xsd:documentation>This is a recommendation for using xml markup to format text.</xsd:documentation>
  </xsd:annotation>

  <xsd:simpleType name="Colour">
    <xsd:annotation>
      <xsd:documentation>
        a color using sRGB; #RRGGBB as Hex values There are also 16 widely known color names with their sRGB values; Black = #000000
        Green = #008000 Silver = #C0C0C0 Lime = #00FF00 Gray = #808080 Olive = #808000 White = #FFFFFF Yellow = #FFFF00 Maroon
        = #800000 Navy = #000080 Red = #FF0000 Blue = #0000FF Purple = #800080 Teal = #008080 Fuchsia= #FF00FF Aqua = #00FFFF

```

```

    </xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[A-Za-z]+|#[0-9A-Fa-f]{3}|#[0-9A-Fa-f]{6}"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:element name="font">
  <xsd:annotation>
    <xsd:documentation>
      change font style locally
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup">
        <xsd:attribute name="size" type="xsd:string" use="optional"/>
        <xsd:attribute name="color" type="rxt:Colour" use="optional"/>
        <xsd:attribute name="face" type="xsd:string" use="optional"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="strike">
  <xsd:annotation>

```

```

    <xsd:documentation>
      strike-through
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

```

```

<xsd:element name="italic">
  <xsd:annotation>
    <xsd:documentation>
      italic font
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

```

```

<xsd:element name="bold">
  <xsd:annotation>

```

```
<xsd:documentation>
  bold font
</xsd:documentation>
</xsd:annotation>
<xsd:complexType mixed="true">
  <xsd:complexContent>
    <xsd:extension base="rxt:textmarkup"/>
  </xsd:complexContent>
</xsd:complexType>
</xsd:element>

<xsd:element name="big">
  <xsd:annotation>
    <xsd:documentation>
      bigger font
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="small">
  <xsd:annotation>
```

```

    <xsd:documentation>
      smaller font
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="underline">
  <xsd:annotation>
    <xsd:documentation>
      underline
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:group name="textstyle">
  <xsd:choice>

```

```

    <xsd:element ref="rxt:italic"/>
    <xsd:element ref="rxt:bold"/>
    <xsd:element ref="rxt:underline"/>
    <xsd:element ref="rxt:strike"/>
    <xsd:element ref="rxt:big"/>
    <xsd:element ref="rxt:small"/>
    <xsd:element ref="rxt;font"/>
  </xsd:choice>
</xsd:group>

<xsd:complexType name="textmarkup" mixed="true">
  <xsd:annotation>
    <xsd:documentation>
      Use textstyle elements for markup of text inside of headline and paragraph elements
    </xsd:documentation>
  </xsd:annotation>
  <xsd:choice minOccurs="0" maxOccurs="unbounded">
    <xsd:group ref="rxt:textstyle"/>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="FormattedMarkupText">
  <xsd:complexContent mixed="true">
    <xsd:extension base="ppx:TextBase">
      <xsd:all>
        <xsd:element name="par">

```

```

<xsd:annotation>
  <xsd:documentation>
    paragraph
  </xsd:documentation>
</xsd:annotation>
<xsd:complexType mixed="true">
  <xsd:complexContent>
    <xsd:extension base="rxt:textmarkup"/>
  </xsd:complexContent>
</xsd:complexType>
</xsd:element>

<xsd:element name="headline">
  <xsd:annotation>
    <xsd:documentation>
      headline
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="rxt:textmarkup"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element ref="rxt:italic"/>

```

```
<xsd:element ref="rxt:bold"/>
<xsd:element ref="rxt:underline"/>
<xsd:element ref="rxt:strike"/>
<xsd:element ref="rxt:big"/>
<xsd:element ref="rxt:small"/>
<xsd:element ref="rxt;font"/>
</xsd:all>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>

<CODE ENDS>
```



<CODE BEGINS>

```
<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10"
  xmlns:rxt="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  targetNamespace="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:import namespace="www.iec.ch/public/TC65SC65BWG7TF10" schemaLocation="IEC61131_10_Ed1_0.xsd" />
  <xsd:complexType name="ScriptBlock">
    <xsd:annotation>
      <xsd:documentation>
        <![CDATA[
          "Script block" is a kind of graphical block object where short textual script can be programed instantly.
        ]]>
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="ppx:FbdObjectBase">
        <xsd:sequence>
          <xsd:element name="Script" type="ppx:ProgrammingLanguageBase"/>
          <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
          <xsd:element name="ConnectionPointOut" type="ppx:ConnectionPointOut" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

```

</xsd:complexType>

138

</xsd:schema>

<CODE ENDS>

<CODE BEGINS>

```
<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10"
  xmlns:rxt="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  targetNamespace="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:import namespace="www.iec.ch/public/TC65SC65BWG7TF10" schemaLocation="IEC61131_10_Ed1_0.xsd" />
  <xsd:element name="JumpStep">
    <xsd:annotation>
      <xsd:documentation>
        "JumpStep" is an element which can be used instead of a "Step" as a successor of a "Transition" or "SelectionConvergence".
        "JumpStep" does not have an associated actionBlock assigned nor a successor ("Transition" or "SelectionDivergence").
        Instead "JumpStep" contains the name of another existing step in the SFC chart.
        So "JumpStep" can be regarded as a SFC-connector which connects a transition or selection convergence to another step.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="ppx:SfcObjectBase">
          <xsd:sequence>
            <xsd:element name="ConnectionPointIn" type="ppx:ConnectionPointIn" minOccurs="0"/>
          </xsd:sequence>
          <xsd:attribute name="targetStepName" type="xsd:string" use="required"/>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

```
</xsd:complexContent>  
</xsd:complexType>  
</xsd:element>  
</xsd:schema>  
  
<CODE ENDS>
```

<CODE BEGINS>

```
<? xml version="1.0" encoding="UTF-8"? >
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ppx="www.iec.ch/public/TC65SC65BWG7TF10"
  xmlns:rxt="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  targetNamespace="www.iec.ch/public/TC65SC65BWG7TF10/Recommendation"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xsd:import namespace="www.iec.ch/public/TC65SC65BWG7TF10" schemaLocation="IEC61131_10_Ed1_0.xsd"/>
  <xsd:complexType name="NamedEventTask">
    <xsd:annotation>
      <xsd:documentation>
        Describe a system task which is executed upon system events like start/stop PLC.
        This implementation extends the abstract type "ppx:TaskBase".
        The system event is identified by its "triggerName".
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="ppx:TaskBase" >
        <xsd:attribute name="triggerName" use="required">
          <xsd:annotation>
            <xsd:documentation>Implementers may add/delete names of system-defined events.</xsd:documentation>
          </xsd:annotation>
          <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
              <xsd:enumeration value="plcColdStart"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:schema>
```

```
<xsd:enumeration value="plcWarmStart"/>
<xsd:enumeration value="plcHotStart"/>
<xsd:enumeration value="plcStop"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>

<CODE ENDS>
```

附录 C  
(资料性)  
XML 文档示例

符合 IEC 61131-3 的示例程序。

VAR\_GLOBAL

Q8\_1 AT %Q8.1; BOOL;

Q8\_2 AT %Q8.2; BOOL;

Q8\_3 AT %Q8.3; BOOL;

Q8\_4 AT %Q8.4; BOOL;

Q8\_5 AT %Q8.5; BOOL;

Q8\_6 AT %Q8.6; BOOL;

END\_VAR

PROGRAM Main

VAR

IL\_FB\_Instance; IL\_FB;

ST\_FB\_Instance; ST\_FB;

LD\_FB\_Instance; LD\_FB;

FBD\_FB\_Instance; FBD\_FB;

SFC\_FB\_Instance; SFC\_FB;

SFC\_FB2\_Instance; SFC\_FB2;

END\_VAR

IL\_FB\_Instance(Output => Q8\_1);

```
ST_FB_Instance(Output => Q8_2);  
LD_FB_Instance(Output => Q8_3);  
FBD_FB_Instance(Output => Q8_4);  
SFC_FB_Instance(Output => Q8_5);  
SFC_FB2_Instance(Output => Q8_6);  
END_PROGRAM
```

```
FUNCTION_BLOCK IL_FB
```

```
VAR_OUTPUT
```

```
Output; Bool;
```

```
END_VAR
```

```
VAR
```

```
S1; Bool;= TRUE;
```

```
S2; Bool;
```

```
S1TON; TON;
```

```
S2TON; TON;
```

```
END_VAR
```

```
// S1 -> S2
```

```
CAL S1TON(IN:= S1, PT:= T#1S);
```

```
LD S1TON.Q;
```

```
R S1;
```

```
S S2;
```



```
// S2 -> S1  
CAL S2TON(IN,= S2, PT,= T#1S);  
LD S2TON.Q;  
R S2;  
S S1;  
  
// Output  
LD S1;  
ST Output;  
END_FUNCTION_BLOCK
```

```
// S2 -> S1  
CAL S2TON(IN,= S2, PT,= T#1S);  
LD S2TON.Q;  
R S2;  
S S1;  
  
// Output  
LD S1;  
ST Output;  
END_FUNCTION_BLOCK
```

```
// Output  
Output:= S1;  
END_FUNCTION_BLOCK
```

FUNCTION\_BLOCK LD\_FB

148

VAR\_OUTPUT

Output: Bool;

END\_VAR

VAR

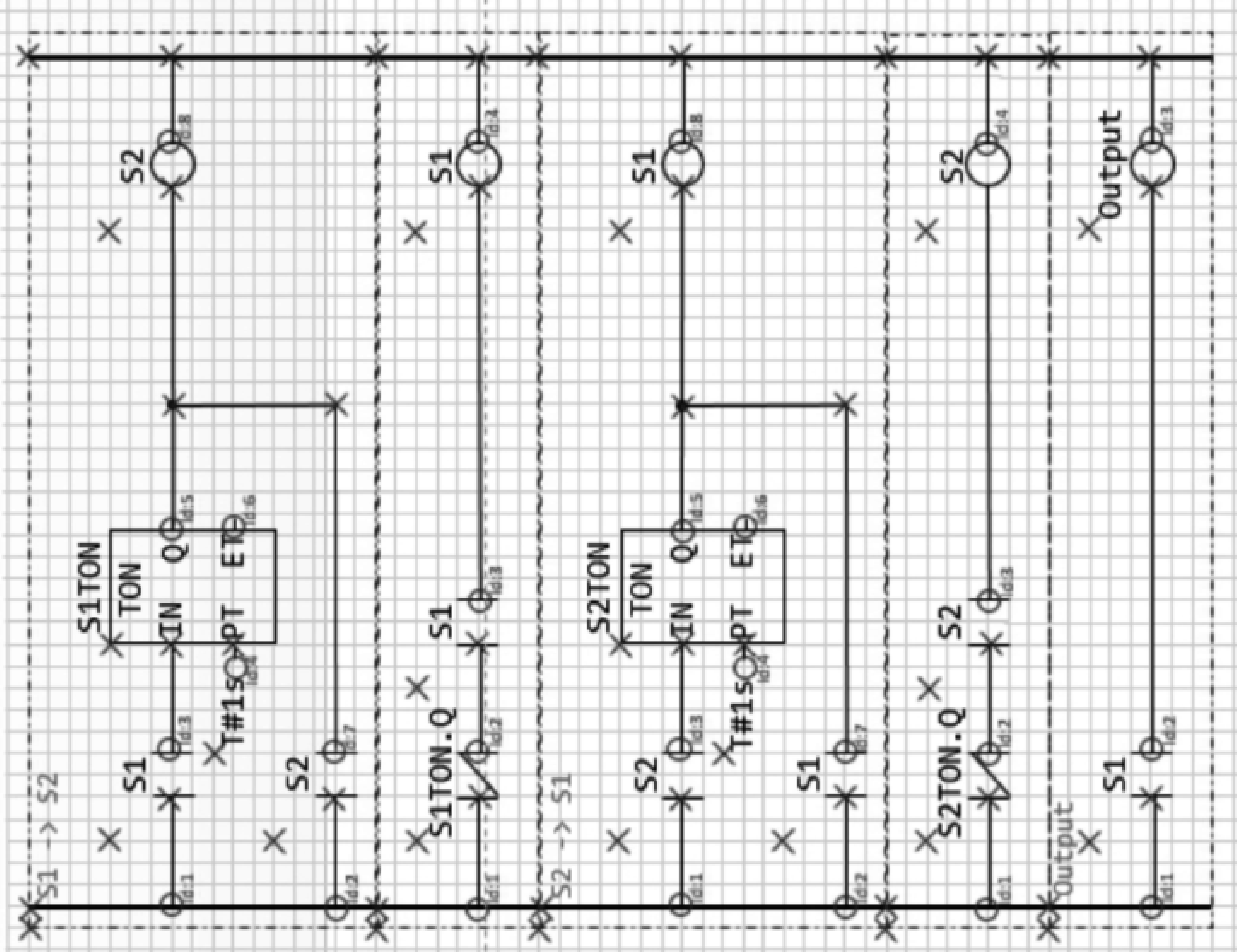
S1: Bool:= TRUE;

S2: Bool;

S1TON: TON;

S2TON: TON;

END\_VAR



END\_FUNCTION\_BLOCK

150

FUNCTION\_BLOCK FBD\_FB

VAR\_OUTPUT

Output: Bool;

END\_VAR

VAR

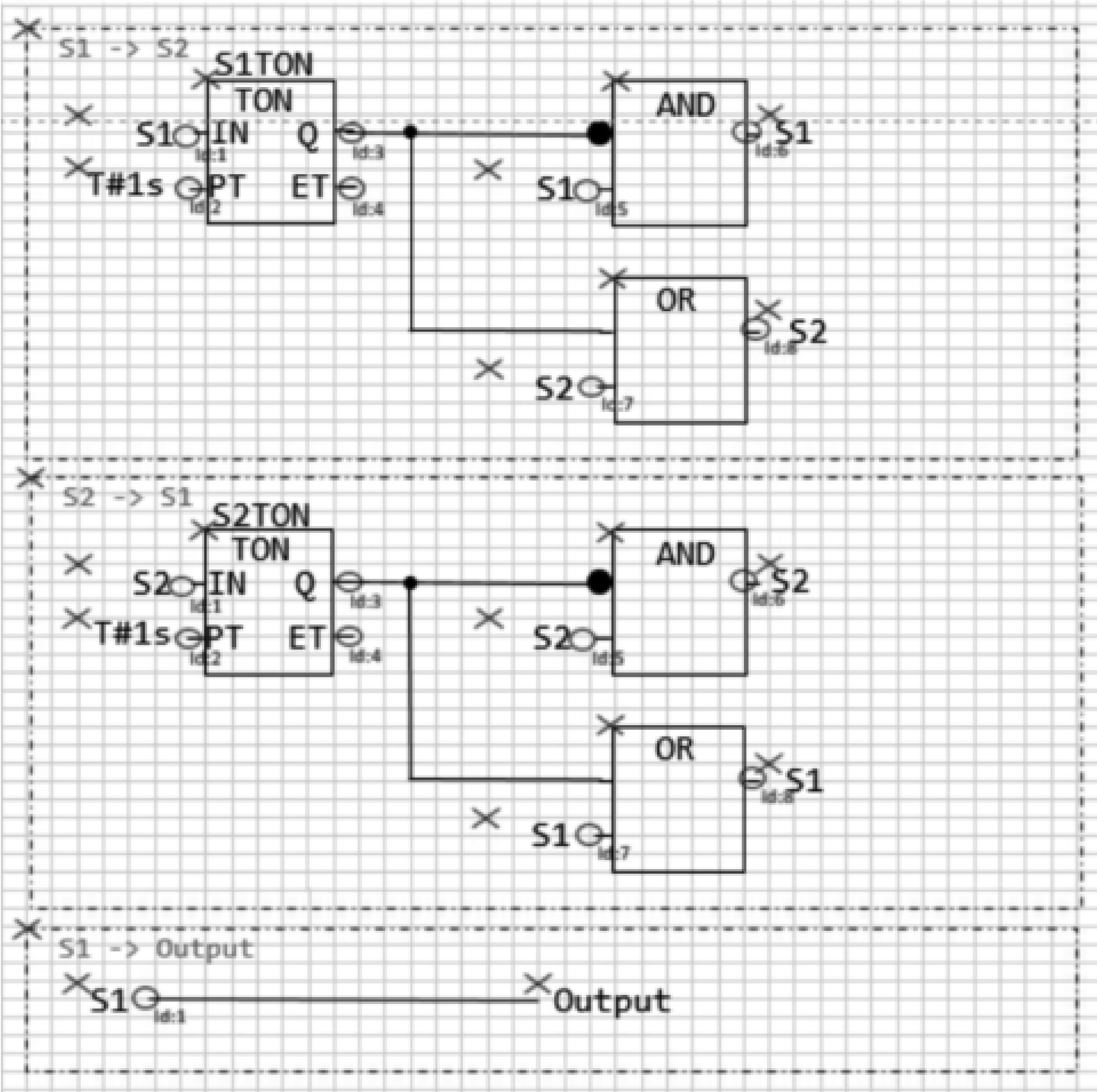
S1: Bool:= TRUE;

S2: Bool;

S1TON: TON;

S2TON: TON;

END\_VAR



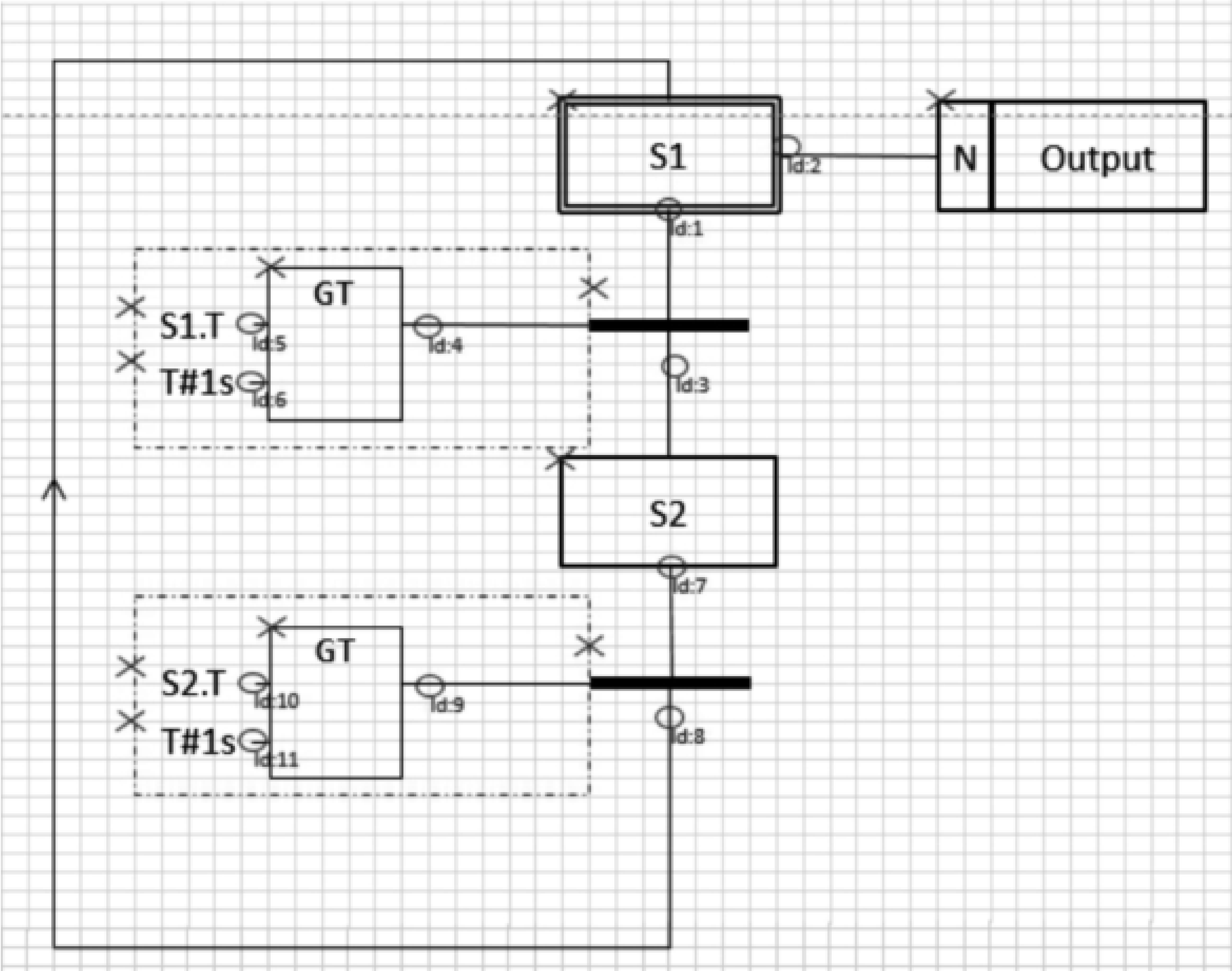
END\_FUNCTION\_BLOCK

FUNCTION\_BLOCK SFC\_FB

VAR\_OUTPUT

Output; Bool;  
END\_VAR

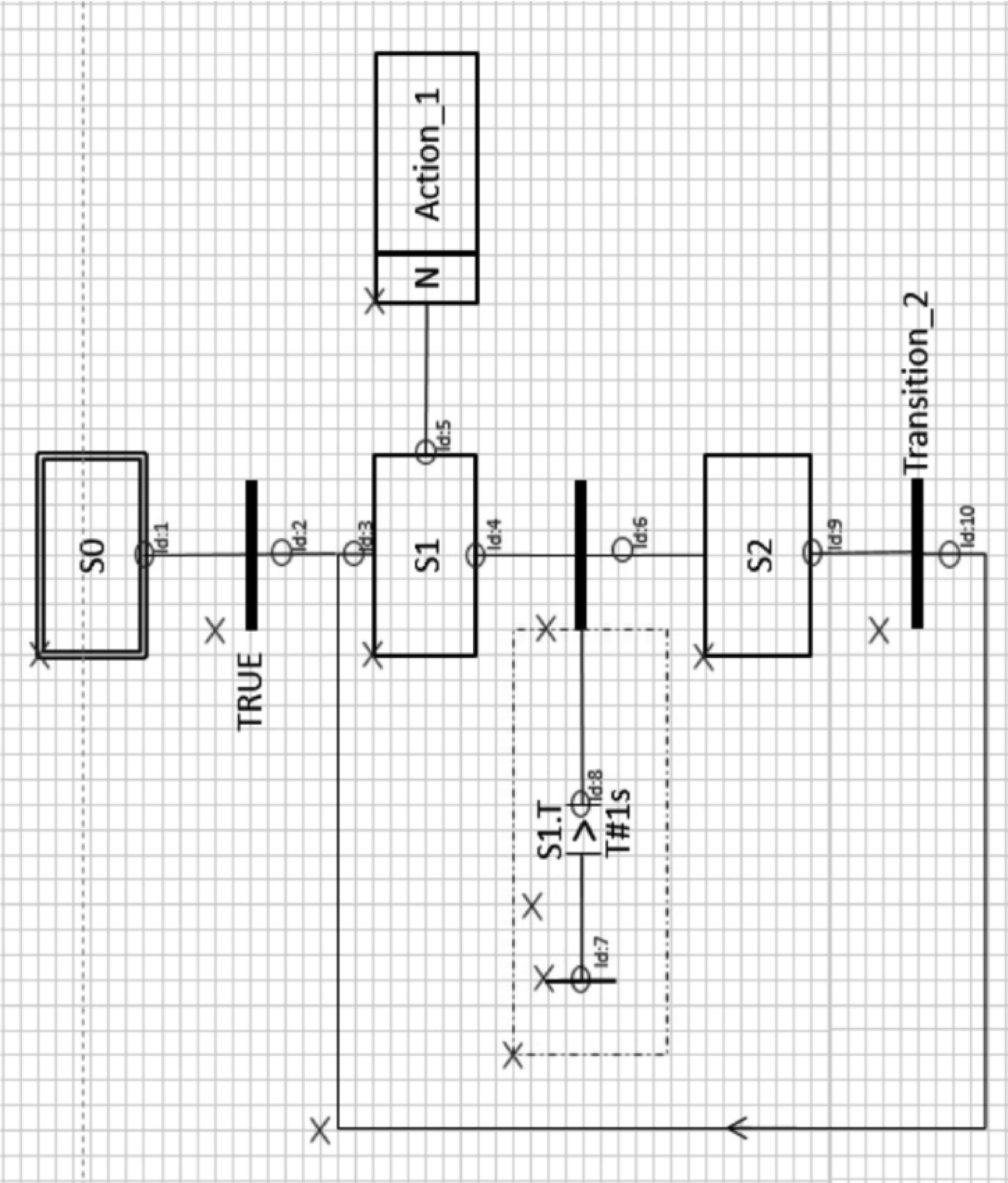
152



END\_FUNCTION\_BLOCK



```
FUNCTION_BLOCK SFC_FB2
VAR_OUTPUT
    Output: Bool;
END_VAR
```



END\_FUNCTION\_BLOCK

## 符合本文件的 XML 文档示例

&lt;CODE BEGINS&gt;

```
<? xml version="1.0" encoding="utf-8"? >
<Project xmlns="www.iec.ch/public/TC65SC65BWG7TF10"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="www.iec.ch/public/TC65SC65BWG7TF10 IEC61131_10_Ed1_0.xsd"
  schemaVersion="1.0">
  <FileHeader companyName="Some Company" companyURL="http://www.someCompany.com"
    productName="Some PADT" productVersion="Edition1" productRelease="2018"/>
  <ContentHeader name="IEC_61131-10_Document_Example" version="1.0"
    creationDateTime="2017-06-14T09:00:00Z" modificationDateTime="2018-11-04T11:56:00Z"
    organization="Some Organization" author="TF10" language="En">
    <CoordinateInfo>
      <FbdScaling x="3" y="3" />
      <LdScaling x="4" y="6" />
      <SfcScaling x="6" y="4" />
    </CoordinateInfo>
    <AddDataInfo/>
  </ContentHeader>
  <Types>
    <GlobalNamespace>
      <NamespaceDecl name="AnnexC">
        <Program name="Main">
          <ExternalVars>
```

```

<Variable name="Q8_1"> <Type> <TypeName>BOOL</TypeName> </Type> </Variable>
<Variable name="Q8_2"> <Type> <TypeName>BOOL</TypeName> </Type> </Variable>
<Variable name="Q8_3"> <Type> <TypeName>BOOL</TypeName> </Type> </Variable>
<Variable name="Q8_4"> <Type> <TypeName>BOOL</TypeName> </Type> </Variable>
<Variable name="Q8_5">
  <Type>
    <TypeName>BOOL</TypeName>
  </Type>
</Variable>
<Variable name="Q8_6">
  <Type>
    <TypeName>BOOL</TypeName>
  </Type>
</Variable>
</ExternalVars>
<Vars accessSpecifier="private">
  <Variable name="IL_FB_Instance"> <Type> <TypeName>IL_FB</TypeName> </Type> </Variable>
  <Variable name="ST_FB_Instance"> <Type> <TypeName>ST_FB</TypeName> </Type> </Variable>
  <Variable name="LD_FB_Instance"> <Type> <TypeName>LD_FB</TypeName> </Type> </Variable>
  <Variable name="FBD_FB_Instance"> <Type> <TypeName>FBD_FB</TypeName> </Type> </Variable>
  <Variable name="SFC_FB_Instance">
    <Type>
      <TypeName>SFC_FB</TypeName>
    </Type>
  </Variable>
  <Variable name="SFC_FB2_Instance">

```

```

        <Type>
            <TypeName>SFC_FB2</TypeName>
        </Type>
    </Variable>
</Vars>

    <MainBody>
        <BodyContent xsi:type="ST">
            <ST>
                <![CDATA[
IL_FB_Instance(Output => Q8_1);
ST_FB_Instance(Output => Q8_2);
LD_FB_Instance(Output => Q8_3);
FBD_FB_Instance(Output => Q8_4);
SFC_FB_Instance(Output => Q8_5);
SFC_FB2_Instance(Output => Q8_6);

                ]]>
            </ST>
        </BodyContent>
    </MainBody>
</Program>
<FunctionBlock name="IL_FB">
    <Parameters>
        <OutputVars>
            <Variable name="Output" orderWithinParamSet="1"> <Type> <TypeName>
                BOOL</TypeName> </Type> </Variable>
        </OutputVars>
    </Parameters>
</FunctionBlock>

```

```

    </Parameters>
    <Vars accessSpecifier="private">
        <Variable name="S1">    <Type>    <TypeName>BOOL</TypeName>    </Type>    <InitialValue>    <SimpleValue
value="TRUE"/>    </InitialValue>    </Variable>
        <Variable name="S2">    <Type>    <TypeName>BOOL</TypeName>    </Type>    </Variable>
        <Variable name="S1TON">    <Type>    <TypeName>TON</TypeName>    </Type>    </Variable>
        <Variable name="S2TON">    <Type>    <TypeName>TON</TypeName>    </Type>    </Variable>
    </Vars>
    <MainBody>
        <BodyContent xsi:type="IL">
            <IL>
                <![CDATA[
// S1 -> S2
CAL  S1TON(IN := S1, PT := T#1S)
LD  S1TON.Q
R    S1
S    S2

// S2 -> S1
CAL  S2TON(IN:= S2, PT := T#1s);
LD  S2TON.Q
R    S2
S    S1

// Output
LD  Output

```

ST S1

```

    ]]>
  </IL>
</BodyContent>
</MainBody>
</FunctionBlock>
<FunctionBlock name="ST_FB">
  <Parameters>
    <OutputVars>
      <Variable name="Output" orderWithinParamSet="1"> <Type> <TypeName>BOOL</TypeName> </Type> </
Variable>
    </OutputVars>
  </Parameters>
  <Vars accessSpecifier="private">
    <Variable name="S1"> <Type> <TypeName>BOOL</TypeName> </Type> <InitialValue> <SimpleValue
value="TRUE"/> </InitialValue> </Variable>
    <Variable name="S2"> <Type> <TypeName>BOOL</TypeName> </Type> </Variable>
    <Variable name="S1TON"> <Type> <TypeName>TON</TypeName> </Type> </Variable>
    <Variable name="S2TON"> <Type> <TypeName>TON</TypeName> </Type> </Variable>
  </Vars>
  <MainBody>
    <BodyContent xsi:type="ST">
      <ST>
        <![CDATA[
159 // S1 -> S2
      SITON(IN := S1, PT := T#1S);

```

```

IF S1TON.Q THEN
160   S1 := FALSE;
      S2 := TRUE;
END_IF

// S2 -> S1
S2TON(IN:= S2, PT := T#1s);
IF S2TON.Q THEN
      S2 := FALSE;
      S1 := TRUE;
END_IF

// Output
Output := S1;
]]>
</ST>
</BodyContent>
</MainBody>
</FunctionBlock>
<FunctionBlock name="LD_FB">
  <Parameters>
    <OutputVars>
      <Variable name="Output" orderWithinParamSet="1">
        <Type>
          <TypeName>BOOL</TypeName>
        </Type>

```



```

    </Variable>
  </OutputVars>
</Parameters>
<Vars accessSpecifier="private">
  <Variable name="S1">
    <Type>
      <TypeName>BOOL</TypeName>
    </Type>
    <InitialValue>
      <SimpleValue value="TRUE"/>
    </InitialValue>
  </Variable>
  <Variable name="S2">
    <Type>
      <TypeName>BOOL</TypeName>
    </Type>
  </Variable>
  <Variable name="S1TON">
    <Type>
      <TypeName>TON</TypeName>
    </Type>
  </Variable>
  <Variable name="S2TON">
    <Type>
      <TypeName>TON</TypeName>
    </Type>
  </Variable>

```

```

    </Variable>
</Vars>
<MainBody>
  <BodyContent xsi:type="LD">
    <Rung evaluationOrder="1">
      <RelPosition x="1" y="1"/>
      <CommonObject xsi:type="Comment">
        <RelPosition x="1" y="0"/>
        <Content xsi:type="SimpleText">S1 -> S2</Content>
      </CommonObject>
      <LdObject xsi:type="LeftPowerRail">
        <RelPosition x="1" y="0"/>
        <ConnectionPointOut connectionPointOutId="1">
          <RelPosition x="0" y="7"/>
        </ConnectionPointOut>
        <ConnectionPointOut connectionPointOutId="2">
          <RelPosition x="0" y="15"/>
        </ConnectionPointOut>
      </LdObject>
      <LdObject xsi:type="Contact" operand="S1">
        <RelPosition x="4" y="4"/>
        <ConnectionPointIn>
          <RelPosition x="2" y="3"/>
          <Connection refConnectionPointOutId="1"/>
        </ConnectionPointIn>
        <ConnectionPointOut connectionPointOutId="3">

```

```

    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<FbdObject xsi:type="DataSource" identifier="T # 1s">
  <RelPosition x="8" y="9"/>
  <ConnectionPointOut connectionPointOutId="4">
    <RelPosition x="4,2" y="1"/>
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="TON" instanceName="S1 TON">
  <RelPosition x="13" y="4"/>
  <InputVariables>
    <InputVariable parameterName="IN">
      <ConnectionPointIn>
        <RelPosition x="0" y="3"/>
        <Connection refConnectionPointOutId="3"/>
      </ConnectionPointIn>
    </InputVariable>
    <InputVariable parameterName="PT">
      <ConnectionPointIn>
        <RelPosition x="0" y="6"/>
        <Connection refConnectionPointOutId="4"/>
      </ConnectionPointIn>
    </InputVariable>
  </InputVariables>
  <OutputVariables>

```

```

    <OutputVariable parameterName="Q">
      <ConnectionPointOut connectionPointOutId="5">
        <RelPosition x="5" y="3"/>
      </ConnectionPointOut>
    </OutputVariable>
    <OutputVariable parameterName="ET">
      <ConnectionPointOut connectionPointOutId="6">
        <RelPosition x="5" y="6"/>
      </ConnectionPointOut>
    </OutputVariable>
  </OutputVariables>
</FbdObject>
<LdObject xsi:type="Contact" operand="S2">
  <RelPosition x="4" y="12"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="2"></Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="7">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Coil" operand="S2">
  <RelPosition x="32" y="4"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>

```

```

    <Connection refConnectionPointOutId="5"/>
    <Connection refConnectionPointOutId="7">
      <RelPosition x="-10" y="0"/>
      <RelPosition x="-10" y="8"/>
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="8">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="RightPowerRail">
  <RelPosition x="40" y="0"/>
  <ConnectionPointIn>
    <RelPosition x="0" y="7"/>
    <Connection refConnectionPointOutId="8"/>
  </ConnectionPointIn>
</LdObject>
</Rung>
<Rung evaluationOrder="2">
  <RelPosition x="1" y="18"/>
  <LdObject xsi:type="LeftPowerRail">
    <RelPosition x="1" y="0"/>
    <ConnectionPointOut connectionPointOutId="1">
      <RelPosition x="0" y="5"/>
    </ConnectionPointOut>
  </LdObject>

```

```

<LdObject xsi:type="Contact" negated="true" operand="S1TON.Q">
  <RelPosition x="4" y="2"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="1"/>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="2">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Contact" operand="S1">
  <RelPosition x="11" y="2"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="2"/>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="3">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Coil" operand="S1">
  <RelPosition x="32" y="2"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="3"/>
  </ConnectionPointIn>

```

```

    <ConnectionPointOut connectionPointOutId="4">
      <RelPosition x="4" y="3"/>
    </ConnectionPointOut>
  </LdObject>
  <LdObject xsi:type="RightPowerRail">
    <RelPosition x="40" y="0"/>
    <ConnectionPointIn>
      <RelPosition x="0" y="5"/>
      <Connection refConnectionPointOutId="4"/>
    </ConnectionPointIn>
  </LdObject>
</Rung>
<Rung evaluationOrder="3">
  <RelPosition x="1" y="26"/>
  <CommonObject xsi:type="Comment">
    <RelPosition x="1" y="0"/>
    <Content xsi:type="SimpleText">S2 -> S1</Content>
  </CommonObject>
  <LdObject xsi:type="LeftPowerRail">
    <RelPosition x="1" y="0"/>
    <ConnectionPointOut connectionPointOutId="1">
      <RelPosition x="0" y="7"/>
    </ConnectionPointOut>
    <ConnectionPointOut connectionPointOutId="2">
      <RelPosition x="0" y="15"/>
    </ConnectionPointOut>

```

```

</LdObject>
<LdObject xsi:type="Contact" operand="S2">
  <RelPosition x="4" y="4"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="1"/>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="3">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<FbdObject xsi:type="DataSource" identifier="T # 1s">
  <RelPosition x="8" y="9"/>
  <ConnectionPointOut connectionPointOutId="4">
    <RelPosition x="4.2" y="1"/>
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="TON" instanceName="S2TON">
  <RelPosition x="13" y="4"/>
  <InputVariables>
    <InputVariable parameterName="IN">
      <ConnectionPointIn>
        <RelPosition x="0" y="3"/>
        <Connection refConnectionPointOutId="3"/>
      </ConnectionPointIn>
    </InputVariable>
  </InputVariables>
</FbdObject>

```



```

    <InputVariable parameterName="PT">
      <ConnectionPointIn>
        <RelPosition x="0" y="6"/>
        <Connection refConnectionPointOutId="4"/>
      </ConnectionPointIn>
    </InputVariable>
  </InputVariables>
  <OutputVariables>
    <OutputVariable parameterName="Q">
      <ConnectionPointOut connectionPointOutId="5">
        <RelPosition x="5" y="3"/>
      </ConnectionPointOut>
    </OutputVariable>
    <OutputVariable parameterName="ET">
      <ConnectionPointOut connectionPointOutId="6">
        <RelPosition x="5" y="6"/>
      </ConnectionPointOut>
    </OutputVariable>
  </OutputVariables>
</FbdObject>
<LdObject xsi:type="Contact" operand="S1">
  <RelPosition x="4" y="12"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="2"></Connection>
  </ConnectionPointIn>

```

```
<ConnectionPointOut connectionPointOutId="7">
  <RelPosition x="4" y="3"/>
</ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Coil" operand="S1">
  <RelPosition x="32" y="4"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="5"/>
    <Connection refConnectionPointOutId="7">
      <RelPosition x="-10" y="0"/>
      <RelPosition x="-10" y="8"/>
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="8">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="RightPowerRail">
  <RelPosition x="40" y="0"/>
  <ConnectionPointIn>
    <RelPosition x="0" y="7"/>
    <Connection refConnectionPointOutId="8"/>
  </ConnectionPointIn>
</LdObject>
</Rung>
```

```

<Rung evaluationOrder="4">
  <RelPosition x="1" y="43"/>
  <LdObject xsi:type="LeftPowerRail">
    <RelPosition x="1" y="0"/>
    <ConnectionPointOut connectionPointOutId="1">
      <RelPosition x="0" y="5"/>
    </ConnectionPointOut>
  </LdObject>
  <LdObject xsi:type="Contact" negated="true" operand="S2TON.Q">
    <RelPosition x="4" y="2"/>
    <ConnectionPointIn>
      <RelPosition x="2" y="3"/>
      <Connection refConnectionPointOutId="1"/>
    </ConnectionPointIn>
    <ConnectionPointOut connectionPointOutId="2">
      <RelPosition x="4" y="3"/>
    </ConnectionPointOut>
  </LdObject>
  <LdObject xsi:type="Contact" operand="S2">
    <RelPosition x="11" y="2"/>
    <ConnectionPointIn>
      <RelPosition x="2" y="3"/>
      <Connection refConnectionPointOutId="2"/>
    </ConnectionPointIn>
    <ConnectionPointOut connectionPointOutId="3">
      <RelPosition x="4" y="3"/>
    </ConnectionPointOut>
  </LdObject>

```

```

    </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Coil" operand="S2">
    <RelPosition x="32" y="2"/>
    <ConnectionPointIn>
        <RelPosition x="2" y="3"/>
        <Connection refConnectionPointOutId="3"/>
    </ConnectionPointIn>
    <ConnectionPointOut connectionPointOutId="4">
        <RelPosition x="4" y="3"/>
    </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="RightPowerRail">
    <RelPosition x="40" y="0"/>
    <ConnectionPointIn>
        <RelPosition x="0" y="5"/>
        <Connection refConnectionPointOutId="4"/>
    </ConnectionPointIn>
</LdObject>
</Rung>
<Rung evaluationOrder="5">
    <RelPosition x="1" y="51"/>
    <CommonObject xsi:type="Comment">
        <RelPosition x="1" y="0"/>
        <Content xsi:type="SimpleText">Output</Content>
    </CommonObject>

```

```

<LdObject xsi:type="LeftPowerRail">
  <RelPosition x="1" y="0"/>
  <ConnectionPointOut connectionPointOutId="1">
    <RelPosition x="0" y="5"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Contact" operand="S1">
  <RelPosition x="4" y="2"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="1"/>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="2">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>
<LdObject xsi:type="Coil" operand="Output">
  <RelPosition x="32" y="2"/>
  <ConnectionPointIn>
    <RelPosition x="2" y="3"/>
    <Connection refConnectionPointOutId="2"/>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="3">
    <RelPosition x="4" y="3"/>
  </ConnectionPointOut>
</LdObject>

```

```
<LdObject xsi:type="RightPowerRail">
  <RelPosition x="40" y="0"/>
  <ConnectionPointIn>
    <RelPosition x="0" y="5"/>
    <Connection refConnectionPointOutId="3"/>
  </ConnectionPointIn>
</LdObject>
</Rung>
</BodyContent>
</MainBody>
</FunctionBlock>
<FunctionBlock name="FBD_FB">
  <Parameters>
    <OutputVars>
      <Variable name="Output" orderWithinParamSet="1"> <Type> <TypeName>BOOL</TypeName> </Type> </
Variable>
    </OutputVars>
  </Parameters>
  <Vars accessSpecifier="private">
    <Variable name="S1"> <Type> <TypeName>BOOL</TypeName> </Type> <InitialValue> <SimpleValue
value="TRUE"/> </InitialValue> </Variable>
    <Variable name="S2"> <Type> <TypeName>BOOL</TypeName> </Type> </Variable>
    <Variable name="S1TON"> <Type> <TypeName>TON</TypeName> </Type> </Variable>
    <Variable name="S2TON"> <Type> <TypeName>TON</TypeName> </Type> </Variable>
  </Vars>
</MainBody>
```

```

<BodyContent xsi:type="FBD">
  <Network xsi:type="FbdNetwork" evaluationOrder="1">
    <RelPosition x="1" y="1" />
    <Size x="41" y="23" />
    <CommonObject xsi:type="Comment">
      <RelPosition x="0" y="0" />
      <Content xsi:type="SimpleText">S1 -> S2</Content>
    </CommonObject>
    <FbdObject xsi:type="DataSource" identifier="S1">
      <RelPosition x="2" y="5" />
      <Size x="4" y="2" />
      <ConnectionPointOut connectionPointOutId="1">
        <RelPosition x="4" y="1" />
      </ConnectionPointOut>
    </FbdObject>
    <FbdObject xsi:type="DataSource" identifier="T # 1s">
      <RelPosition x="2" y="8" />
      <Size x="4" y="2" />
      <ConnectionPointOut connectionPointOutId="2">
        <RelPosition x="4" y="1" />
      </ConnectionPointOut>
    </FbdObject>
    <FbdObject xsi:type="Block" typeName="TON" instanceName="S1 TON">
      <RelPosition x="7" y="2" />
      <Size x="5" y="8" />
      <InputVariables>

```

```

<InputVariable parameterName="IN">
  <ConnectionPointIn>
    <RelPosition x="-1" y="3" />
    <Connection refConnectionPointOutId="1">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</InputVariable>
<InputVariable parameterName="PT">
  <ConnectionPointIn>
    <RelPosition x="-1" y="6" />
    <Connection refConnectionPointOutId="2">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</InputVariable>
</InputVariables>
<OutputVariables>
  <OutputVariable parameterName="Q">
    <ConnectionPointOut connectionPointOutId="3">
      <RelPosition x="6" y="3" />
    </ConnectionPointOut>
  </OutputVariable>
  <OutputVariable parameterName="ET">
    <ConnectionPointOut connectionPointOutId="4">
      <! -- Although there is nothing connected to 'S1TON.ET' there is still a connection point given. -->
    </ConnectionPointOut>
  </OutputVariable>
</OutputVariables>

```



```

        <RelPosition x="6" y="6" />
    </ConnectionPointOut>
</OutputVariable>
</OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S1">
    <RelPosition x="18" y="8" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="5">
        <RelPosition x="4" y="1" />
    </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="AND">
    <RelPosition x="23" y="3" />
    <Size x="5" y="8" />
    <InputVariables>
        <InputVariable parameterName="IN1" suppressName="true" negated="true">
            <ConnectionPointIn>
                <RelPosition x="-1" y="3" />
                <Connection refConnectionPointOutId="3">
                    <RelPosition x="-7" y="0" />

```

is given to indicate

a supporting point which is identical to the upper edge of the connection from 'S1TON.Q' to the 'OR' func-

tion. -->

```

</Connection>

```

```

    </ConnectionPointIn>
  </InputVariable>
  <InputVariable parameterName="IN2" suppressName="true">
    <ConnectionPointIn>
      <RelPosition x="-1" y="6" />
      <Connection refConnectionPointOutId="5">
        <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
      </Connection>
    </ConnectionPointIn>
  </InputVariable>
</InputVariables>
<OutputVariables>
  <OutputVariable parameterName="OUT" suppressName="true">
    <ConnectionPointOut connectionPointOutId="6">
      <RelPosition x="6" y="3" />
    </ConnectionPointOut>
  </OutputVariable>
</OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSink" identifier="S1">
  <RelPosition x="29" y="6" />
  <Size x="4" y="2" />
  <ConnectionPointIn>
    <RelPosition x="0" y="1" />
    <Connection refConnectionPointOutId="6">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>

```

```

    </Connection>
  </ConnectionPointIn>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S2">
  <RelPosition x="18" y="19" />
  <Size x="4" y="2" />
  <ConnectionPointOut connectionPointOutId="7">
    <RelPosition x="4" y="1" />
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="OR">
  <RelPosition x="23" y="14" />
  <Size x="5" y="8" />
  <InputVariables>
    <InputVariable parameterName="IN1" suppressName="true">
      <ConnectionPointIn>
        <RelPosition x="-1" y="3" />
        <Connection refConnectionPointOutId="3">
          <RelPosition x="-7" y="0" /> <!-- Positions are relative to superior 'ConnectionPointIn'. -->
          <RelPosition x="-7" y="-11" />
        </Connection>
      </ConnectionPointIn>
    </InputVariable>
    <InputVariable parameterName="IN2" suppressName="true">
      <ConnectionPointIn>
        <RelPosition x="-1" y="6" />

```

```

    <Connection refConnectionPointOutId="7">
      <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</InputVariable>
</InputVariables>
<OutputVariables>
  <OutputVariable parameterName="OUT" suppressName="true">
    <ConnectionPointOut connectionPointOutId="8">
      <RelPosition x="6" y="3" />
    </ConnectionPointOut>
  </OutputVariable>
</OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSink" identifier="S2">
  <RelPosition x="29" y="16" />
  <Size x="4" y="2" />
  <ConnectionPointIn>
    <RelPosition x="0" y="1" />
    <Connection refConnectionPointOutId="8">
      <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</FbdObject>
</Network>
<Network xsi:type="FbdNetwork" evaluationOrder="2">

```

```

<RelPosition x="1" y="26" />
<Size x="41" y="23" />
<CommonObject xsi:type="Comment">
  <RelPosition x="0" y="0" />
  <Content xsi:type="SimpleText">S2 -> S1</Content>
</CommonObject>
<FbdObject xsi:type="DataSource" identifier="S2">
  <RelPosition x="2" y="5" />
  <Size x="4" y="2" />
  <ConnectionPointOut connectionPointOutId="1">
    <RelPosition x="4" y="1" />
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="T # 1s">
  <RelPosition x="2" y="8" />
  <Size x="4" y="2" />
  <ConnectionPointOut connectionPointOutId="2">
    <RelPosition x="4" y="1" />
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="TON" instanceName="S2TON">
  <RelPosition x="7" y="2" />
  <Size x="5" y="8" />
  <InputVariables>
    <InputVariable parameterName="IN">
      <ConnectionPointIn>

```

```

    <RelPosition x="-1" y="3" />
    <Connection refConnectionPointOutId="1">
        <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
</ConnectionPointIn>
</InputVariable>
<InputVariable parameterName="PT">
    <ConnectionPointIn>
        <RelPosition x="-1" y="6" />
        <Connection refConnectionPointOutId="2">
            <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
        </Connection>
    </ConnectionPointIn>
</InputVariable>
</InputVariables>
<OutputVariables>
    <OutputVariable parameterName="Q">
        <ConnectionPointOut connectionPointOutId="3">
            <RelPosition x="6" y="3" />
        </ConnectionPointOut>
    </OutputVariable>
    <OutputVariable parameterName="ET">
        <ConnectionPointOut connectionPointOutId="4">
            <! -- Although there is nothing connected to 'S2TON.ET' there is still a connection point given. -->
            <RelPosition x="6" y="6" />
        </ConnectionPointOut>
    </OutputVariable>
</OutputVariables>

```

```

        </OutputVariable>
    </OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S2">
    <RelPosition x="18" y="8" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="5">
        <RelPosition x="4" y="1" />
    </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="AND">
    <RelPosition x="23" y="3" />
    <Size x="5" y="8" />
    <InputVariables>
        <InputVariable parameterName="IN1" suppressName="true" negated="true">
            <ConnectionPointIn>
                <RelPosition x="-1" y="3" />
                <Connection refConnectionPointOutId="3">
                    <RelPosition x="-7" y="0" />

```

is given to indicate

a supporting point which is identical to the upper edge of the connection from 'S2TON.Q' to the 'OR' func-

tion. -->

```

            </Connection>
        </ConnectionPointIn>
    </InputVariable>

```

```

<InputVariable parameterName="IN2" suppressName="true">
  <ConnectionPointIn>
    <RelPosition x="-1" y="6" />
    <Connection refConnectionPointOutId="5">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</InputVariable>
</InputVariables>
<OutputVariables>
  <OutputVariable parameterName="OUT" suppressName="true">
    <ConnectionPointOut connectionPointOutId="6">
      <RelPosition x="6" y="3" />
    </ConnectionPointOut>
  </OutputVariable>
</OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSink" identifier="S2">
  <RelPosition x="29" y="6" />
  <Size x="4" y="2" />
  <ConnectionPointIn>
    <RelPosition x="0" y="1" />
    <Connection refConnectionPointOutId="6">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>

```



```

</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S1">
  <RelPosition x="18" y="19" />
  <Size x="4" y="2" />
  <ConnectionPointOut connectionPointOutId="7">
    <RelPosition x="4" y="1" />
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="Block" typeName="OR">
  <RelPosition x="23" y="14" />
  <Size x="5" y="8" />
  <InputVariables>
    <InputVariable parameterName="IN1" suppressName="true">
      <ConnectionPointIn>
        <RelPosition x="-1" y="3" />
        <Connection refConnectionPointOutId="3">
          <RelPosition x="-7" y="0" /> <!-- Positions are relative to superior 'ConnectionPointIn'. -->
          <RelPosition x="-7" y="-11" />
        </Connection>
      </ConnectionPointIn>
    </InputVariable>
    <InputVariable parameterName="IN2" suppressName="true">
      <ConnectionPointIn>
        <RelPosition x="-1" y="6" />
        <Connection refConnectionPointOutId="7">
          <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->

```

```

        </Connection>
    </ConnectionPointIn>
</InputVariable>
</InputVariables>
<OutputVariables>
    <OutputVariable parameterName="OUT" suppressName="true">
        <ConnectionPointOut connectionPointOutId="8">
            <RelPosition x="6" y="3" />
        </ConnectionPointOut>
    </OutputVariable>
</OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSink" identifier="S1">
    <RelPosition x="29" y="16" />
    <Size x="4" y="2" />
    <ConnectionPointIn>
        <RelPosition x="0" y="1" />
        <Connection refConnectionPointOutId="8">
            <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->
        </Connection>
    </ConnectionPointIn>
</FbdObject>
</Network>
<Network xsi:type="FbdNetwork" evaluationOrder="3">
    <RelPosition x="1" y="51" />
    <Size x="41" y="8" />

```

```

<CommonObject xsi:type="Comment">
  <RelPosition x="0" y="0" />
  <Content xsi:type="SimpleText">S1 -> OUTPUT</Content>
</CommonObject>
<FbdObject xsi:type="DataSource" identifier="S1">
  <RelPosition x="2" y="3" />
  <Size x="4" y="2" />
  <ConnectionPointOut connectionPointOutId="1">
    <RelPosition x="4" y="1" />
  </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="DataSink" identifier="Output">
  <RelPosition x="20" y="3" />
  <Size x="6" y="2" />
  <ConnectionPointIn>
    <RelPosition x="0" y="1" />
    <Connection refConnectionPointOutId="1">
      <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</FbdObject>
</Network>
</BodyContent>
</MainBody>
</FunctionBlock>
<FunctionBlock name="SFC_FB">

```

```

<Parameters>
  <OutputVars>
    <Variable name="Output" orderWithinParamSet="1">
      <Documentation xsi:type="SimpleText">
        Output flickers: for 1 second it is true and for the next second it is false
      </Documentation>
      <Type> <TypeName>BOOL</TypeName> </Type>
    </Variable>
  </OutputVars>
</Parameters>
<MainBody>
  <BodyContent xsi:type="SFC">
    <! -- SFC-->
    <SfcObject xsi:type="Step" name="S1" initialStep="true">
      <RelPosition x="21" y="5" />
      <Size x="8" y="6" />
      <ConnectionPointIn>
        <RelPosition x="4" y="0" />
        <Connection refConnectionPointOutId="8">
          <! -- # # # to do: correct the insert coordinates of th connection line. -->
          <RelPosition x="0" y="-2"/> <! -- Positions are relative to superior 'ConnectionPointIn'. -->
          <RelPosition x="-23" y="-2"/>
          <RelPosition x="-23" y="45"/>
          <RelPosition x="0" y="45"/>
        </Connection>
      </ConnectionPointIn>
    </SfcObject>
  </BodyContent>
</MainBody>

```

```

    <ConnectionPointOut connectionPointOutId="1">
      <RelPosition x="4" y="6" />
    </ConnectionPointOut>
    <ConnectionPointOutAction connectionPointOutId="2">
      <RelPosition x="8" y="3" />
    </ConnectionPointOutAction>
  </SfcObject>
  <CommonObject xsi:type="ActionBlocks">
    <RelPosition x="35" y="5" />
    <Size x="10" y="6" />
    <ConnectionPointIn>
      <RelPosition x="0" y="3" />
      <Connection refConnectionPointOutId="2">
        <! - Since this is connection is a straight horizontal line with known start and end point, no 'RelPosition' is necessary.
      ->
      </Connection>
    </ConnectionPointIn>
    <ActionBlock>
      <ActionQualifier qualifier="N" />
      <ComplexOperand>Output</ComplexOperand>
    </ActionBlock>
  </CommonObject>
  <SfcObject xsi:type="Transition">
    <RelPosition x="22" y="15" />
    <Size x="6" y="4" />
    <ConnectionPointIn>

```

```

    <RelPosition x="3" y="0" />
    <Connection refConnectionPointOutId="1">
      <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="3">
    <RelPosition x="3" y="4" />
  </ConnectionPointOut>
  <Condition>
    <GraphicalPredicate>
      <ConnectionPointIn>
        <RelPosition x="0" y="2" />
        <Connection refConnectionPointOutId="4">
          <!-- Since this connection is a straight horizontal line with known start and end point, no 'RelPosition' is necessary.
-->

          </Connection>
        </ConnectionPointIn>
        <GraphicalExpression xsi:type="FbdNetwork" evaluationOrder="0">
          <RelPosition x="-17" y="-2"/> <!-- Position is relative to superior 'Transition'. -->
          <Size x="17" y="10"/>
          <FbdObject xsi:type="Block" typeName="GT">
            <RelPosition x="5" y="1" />
            <Size x="5" y="8" />
            <InputVariables>
              <InputVariable parameterName="IN1" suppressName="true">
                <ConnectionPointIn>

```

```

    <RelPosition x="-1" y="3" />
    <Connection refConnectionPointOutId="5">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</InputVariable>
<InputVariable parameterName="IN2" suppressName="true">
  <ConnectionPointIn>
    <RelPosition x="-1" y="6" />
    <Connection refConnectionPointOutId="6">
      <! -- Since start and end point have the same position, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
</InputVariable>
</InputVariables>
<OutputVariables>
  <OutputVariable parameterName="OUT" suppressName="true">
    <ConnectionPointOut connectionPointOutId="4">
      <RelPosition x="6" y="3" />
    </ConnectionPointOut>
  </OutputVariable>
</OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S1.T">
  <RelPosition x="0" y="3" />
  <Size x="4" y="2" />

```

```

    <ConnectionPointOut connectionPointOutId="5">
      <RelPosition x="4" y="1" />
    </ConnectionPointOut>
  </FbdObject>
  <FbdObject xsi:type="DataSource" identifier="T # 1s">
    <RelPosition x="0" y="6" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="6">
      <RelPosition x="4" y="1" />
    </ConnectionPointOut>
  </FbdObject>
</GraphicalExpression>
</GraphicalPredicate>
</Condition>
</SfcObject>
<SfcObject xsi:type="Step" name="S2">
  <RelPosition x="21" y="24" />
  <Size x="8" y="6" />
  <ConnectionPointIn>
    <RelPosition x="4" y="0" />
    <Connection refConnectionPointOutId="3">
      <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="7">
    <RelPosition x="3" y="4" />

```



```

    </ConnectionPointOut>
</SfcObject>
<SfcObject xsi:type="Transition">
  <RelPosition x="165" y="224" />
  <Size x="6" y="4" />
  <ConnectionPointIn>
    <RelPosition x="3" y="0" />
    <Connection refConnectionPointOutId="10">
      <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="8">
    <RelPosition x="3" y="4" />
  </ConnectionPointOut>
  <Condition>
    <GraphicalPredicate>
      <ConnectionPointIn>
        <RelPosition x="0" y="2" />
        <Connection refConnectionPointOutId="9">
          <!-- Since this connection is a straight horizontal line with known start and end point, no 'RelPosition' is necessary. -->
        </Connection>
      </ConnectionPointIn>
      <GraphicalExpression xsi:type="FbdNetwork" evaluationOrder="0">
        <RelPosition x="-17" y="-2"/> <!-- Position is relative to superior 'Transition', -->
        <Size x="17" y="10"/>
      </GraphicalExpression>
    </Condition>
  </SfcObject>
-->

```

```

<FbdObject xsi:type="Block" typeName="GT">
  <RelPosition x="5" y="1" />
  <Size x="5" y="8" />
  <InputVariables>
    <InputVariable parameterName="IN1" suppressName="true">
      <ConnectionPointIn>
        <RelPosition x="-1" y="3" />
        <Connection refConnectionPointOutId="10">
          <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->
        </Connection>
      </ConnectionPointIn>
    </InputVariable>
    <InputVariable parameterName="IN2" suppressName="true">
      <ConnectionPointIn>
        <RelPosition x="-1" y="6" />
        <Connection refConnectionPointOutId="11">
          <!-- Since start and end point have the same position, no 'RelPosition' is necessary. -->
        </Connection>
      </ConnectionPointIn>
    </InputVariable>
  </InputVariables>
  <OutputVariables>
    <OutputVariable parameterName="OUT" suppressName="true">
      <ConnectionPointOut connectionPointOutId="9">
        <RelPosition x="6" y="3" />
      </ConnectionPointOut>
    </OutputVariable>
  </OutputVariables>
</FbdObject>

```

```

        </OutputVariable>
    </OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S2.T">
    <RelPosition x="0" y="3" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="10">
        <RelPosition x="4" y="1" />
    </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="T # 1s">
    <RelPosition x="0" y="6" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="11">
        <RelPosition x="4" y="1" />
    </ConnectionPointOut>
</FbdObject>
</GraphicalExpression>
</GraphicalPredicate>
</Condition>
</SfcObject>
</BodyContent>
</MainBody>
</FunctionBlock>
<! - ===== Optional examples: =====>

```

```

        </OutputVariable>
    </OutputVariables>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="S2.T">
    <RelPosition x="0" y="3" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="10">
        <RelPosition x="4" y="1" />
    </ConnectionPointOut>
</FbdObject>
<FbdObject xsi:type="DataSource" identifier="T # 1s">
    <RelPosition x="0" y="6" />
    <Size x="4" y="2" />
    <ConnectionPointOut connectionPointOutId="11">
        <RelPosition x="4" y="1" />
    </ConnectionPointOut>
</FbdObject>
</GraphicalExpression>
</GraphicalPredicate>
</Condition>
</SfcObject>
</BodyContent>
</MainBody>
</FunctionBlock>
<! - ===== Optional examples: =====>

```

```

<SfcObject xsi:type="Transition">
  <RelPosition x="22" y="14" />
  <Size x="6" y="4" />
  <ConnectionPointIn>
    <RelPosition x="3" y="0" />
    <Connection refConnectionPointOutId="1">
      <! -- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="2">
    <RelPosition x="3" y="4" />
  </ConnectionPointOut>
  <Condition>
    <TextualPredicate>
      <PredicateContent xsi:type="ST">
        <ST>TRUE</ST>
      </PredicateContent>
    </TextualPredicate>
  </Condition>
</SfcObject>
<SfcObject xsi:type="SelectionConvergence">
  <RelPosition x="2" y="20" />
  <Size x="23" y="2" />
  <ConnectionPointIn>
    <RelPosition x="0" y="2" />
    <Connection refConnectionPointOutId="10">

```

```

    <RelPosition x="0" y="37" /> <!-- Positions are relative to superior 'ConnectionPointIn'. -->
    <RelPosition x="23" y="37" />
  </Connection>
</ConnectionPointIn>
<ConnectionPointIn>
  <RelPosition x="23" y="0" />
  <Connection refConnectionPointOutId="2">
    <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
  </Connection>
</ConnectionPointIn>
<ConnectionPointOut connectionPointOutId="3">
  <RelPosition x="23" y="2" />
</ConnectionPointOut>
</SfcObject>
<SfcObject xsi:type="Step" name="S1">
  <RelPosition x="21" y="23" />
  <Size x="8" y="6" />
  <ConnectionPointIn>
    <RelPosition x="4" y="0" />
    <Connection refConnectionPointOutId="3">
      <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
    </Connection>
  </ConnectionPointIn>
  <ConnectionPointOut connectionPointOutId="4">
    <RelPosition x="4" y="6" />
  </ConnectionPointOut>

```

```

    <ConnectionPointOutAction connectionPointOutId="5">
      <RelPosition x="8" y="3" />
    </ConnectionPointOutAction>
  </SfcObject>
  <CommonObject xsi:type="ActionBlocks">
    <RelPosition x="35" y="23" />
    <Size x="10" y="6" />
    <ConnectionPointIn>
      <RelPosition x="0" y="3" />
      <Connection refConnectionPointOutId="5">
        <! -- Since this connection is a straight horizontal line with known start and end point, no 'RelPosition' is necessary. -->
      </Connection>
    </ConnectionPointIn>
    <ActionBlock>
      <ActionQualifier qualifier="N" />
      <ReferenceName>Action_1</ReferenceName>
    </ActionBlock>
  </CommonObject>
  <SfcObject xsi:type="Transition">
    <RelPosition x="22" y="33" />
    <Size x="6" y="4" />
    <ConnectionPointIn>
      <RelPosition x="3" y="0" />
      <Connection refConnectionPointOutId="4">
        <! -- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
      </Connection>
    </ConnectionPointIn>
  </SfcObject>

```

```

</ConnectionPointIn>
<ConnectionPointOut connectionPointOutId="6">
  <RelPosition x="3" y="4" />
</ConnectionPointOut>
<Condition>
  <GraphicalPredicate>
    <ConnectionPointIn>
      <RelPosition x="0" y="2" />
      <Connection refConnectionPointOutId="8">
        <! -- Since this connection is a straight horizontal line with known start and end point, no 'RelPosition' is necessary.
-->

      </Connection>
    </ConnectionPointIn>
    <GraphicalExpression xsi:type="LadderRung" evaluationOrder="0">
      <RelPosition x="-17" y="-4"/> <! -- Position is relative to superior 'Transition'. -->
      <Size x="17" y="9"/>
      <LdObject xsi:type="LeftPowerRail">
        <RelPosition x="3" y="2"/>
        <ConnectionPointOut connectionPointOutId="7">
          <RelPosition x="0" y="2"/>
        </ConnectionPointOut>
      </LdObject>
      <LdObject xsi:type="CompareContact" compareOperator=">" operand1="S1,T" operand2="T # 1s">
        <RelPosition x="8" y="1"/>
        <ConnectionPointIn>
          <RelPosition x="2" y="3"/>

```



```

        <Connection refConnectionPointOutId="7"/>
    </ConnectionPointIn>
    <ConnectionPointOut connectionPointOutId="8">
        <RelPosition x="4" y="3"/>
    </ConnectionPointOut>
    <Type>TIME</Type>
</LdObject>
</GraphicalExpression>
</GraphicalPredicate>
</Condition>
</SfcObject>
<SfcObject xsi:type="Step" name="S2">
    <RelPosition x="21" y="42" />
    <Size x="8" y="6" />
    <ConnectionPointIn>
        <RelPosition x="4" y="0" />
        <Connection refConnectionPointOutId="6">
            <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary. -->
        </Connection>
    </ConnectionPointIn>
    <ConnectionPointOut connectionPointOutId="9">
        <RelPosition x="4" y="6" />
    </ConnectionPointOut>
</SfcObject>
<SfcObject xsi:type="Transition">
    <RelPosition x="22" y="52" />

```

```

    <Size x="6" y="4" />
    <ConnectionPointIn>
      <RelPosition x="3" y="0" />
      <Connection refConnectionPointOutId="9">
        <!-- Since this connection is a straight vertical line with known start and end point, no 'RelPosition' is necessary, -->
      </Connection>
    </ConnectionPointIn>
    <ConnectionPointOut connectionPointOutId="10">
      <RelPosition x="3" y="4" />
    </ConnectionPointOut>
    <Condition>
      <Reference name="Transition_2" />
    </Condition>
  </SfcObject>
</BodyContent>
</MainBody>
<Action name="Action_1">
  <Body>
    <BodyContent xsi:type="ST">
      <ST>
        <!-- [CDATA[
        Output := Action_1.Q;
        ]]>      <!-- Usage of ".Q" requires SFC with "final scan" logic in order to reset "Output" when Action becomes in-
active, -->

      </ST>
    </BodyContent>
  </Body>
</Action>

```

```
</Body>
</Action>
<Transition name="Transition_2">
  <Condition>
    <PredicateContent xsi:type="ST">
      <ST>
        <![CDATA[
          Transition_2 := S2.T > T # 1s;
        ]]>
      </ST>
    </PredicateContent>
  </Condition>
</Transition>
</FunctionBlock>
</NamespaceDecl>
</GlobalNamespace>
</Types>
<Instances>
  <Configuration name="MyConfiguration">
    <Resource name="CPU01" resourceTypeName="SomeResourceType">
      <GlobalVars>
        <Variable name="Q8_1"> <Type> <TypeName>BOOL</TypeName> </Type> <Address location="Q" size="X"
address="8.1"> </Address> </Variable>
        <Variable name="Q8_2"> <Type> <TypeName>BOOL</TypeName> </Type> <Address location="Q" size="X"
address="8.2"> </Address> </Variable>
        <Variable name="Q8_3"> <Type> <TypeName>BOOL</TypeName> </Type> <Address location="Q" size="X"
```

```
address="8.3"> </Address> </Variable>
    <Variable name="Q8_4"> <Type> <TypeName>BOOL</TypeName> </Type> <Address location="Q" size="X"
address="8.4"> </Address> </Variable>
    <Variable name="Q8_5"> <Type> <TypeName>BOOL</TypeName> </Type> <Address location="Q" size="X"
address="8.5"> </Address> </Variable>
    <Variable name="Q8_6">
        <Type>
            <TypeName>BOOL</TypeName>
        </Type>
        <Address location="Q" size="X" address="8.6"> </Address>
    </Variable>
</GlobalVars>
<Task xsi:type="StandardTask" name="T100" interval="100000" priority="1" />
<ProgramInstance name="PGI1" typeName="Main" associatedTaskName="T100" />
</Resource>
</Configuration>
</Instances>
</Project>

<CODE ENDS>
```

### 参 考 文 献

- [1] Extensible Markup Language (XML), W3C Recommendation, World Wide Web Consortium, <http://www.w3.org/TR/xml/> [viewed 2018-01-23]
  - [2] Visual Studio, Microsoft Corporation, <https://www.visualstudio.com/>
  - [3] XML Formats for IEC61131-3, Version 2.01-Official Release, Technical Paper, PLCopen Technical Committee 6, PLCopen, [http://www.plcopen.org/pages/tc6\\_xml/](http://www.plcopen.org/pages/tc6_xml/)
  - [4] XMLSpy XML Editor, Altova, Inc, <https://www.altova.com/xmlspy.html> [viewed 2018-01-23]
  - [5] <http://www.w3.org/1999/xhtml>, XHTML namespace
-