ICS 35.060;25.040.01
L 74

中华人民共和国国家标准

GB/T 39003.1—2020/IEC 62714-1:2018

工业自动化系统工程用工程数据交换格式 自动化标记语言 第1部分:架构和通用要求

Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 1:Architecture and general requirements

(IEC 62714-1:2018,IDT)

2020-09-29 发布　　2021-04-01 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

GB/T 39003《工业自动化系统工程用工程数据交换格式　自动化标记语言》分为以下4个部分：

——第1部分：架构和通用要求；

——第2部分：角色类库；

——第3部分：几何学和运动学；

——第4部分：逻辑。

本部分为GB/T 39003的第1部分。

本部分按照GB/T 1.1—2009给出的规则起草。

本部分使用翻译法等同采用IEC 62714-1:2018《工业自动化系统工程用工程数据交换格式　自动化标记语言　第1部分：架构和通用要求》。

本部分做了下列编辑性修改：

——为了增加可读性，本部分增加了附录NA“本部分使用的惯用词语中英文对照”；

——删除参考文献IEC 62714-2、IEC 62714-3、IEC 62714-4，因为与规范性引用文件重复。

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这些专利的责任。

本部分由中国机械工业联合会提出。

本部分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。

本部分起草单位：上海市计量测试技术研究院、上海佐竹冷热控制技术有限公司、杭州电子科技大学、广州致讯信息科技有限责任公司、机械工业仪器仪表综合技术经济研究所、西南大学、中国科学院沈阳自动化研究所、中国计量大学、云南省计量测试技术研究院、大金空调(上海)有限公司、上海交通大学、上海科学院、上海工业自动化仪表研究院有限公司、上海市在线检测与控制技术重点实验室。

本部分主要起草人：邵力、余国瑞、陈曦、杜军、吴卿、王裴劼、肖天雷、徐文劼、柳晓菁、祁虔、刘阳、孙坚、饶杰、陈杰、陈江平、楼志斌、肖红练。

引　言

IEC 62714 是针对自动化工程领域的数据交换解决方案。

IEC 62714 中定义的数据交换格式(自动化标记语言,AML)是一种基于可扩展标记语言(XML)架构的数据格式,它被用于支持异构工程工具之间的数据交换。

AML 旨在建立不同领域的工程工具之间的联系,例如机械装备工程、电气设计、过程工程、过程控制工程、人机界面开发、PLC 编程和机器人编程等。

AML 遵循面向对象的方法存储工程信息,并且允许用封装有不同方面内容的数据对象对工厂的物理和逻辑组成部分进行建模。一个对象可包含其他子对象,也可隶属于一个更大的组合或聚合。工厂自动化项目中一个典型对象包含的信息包括拓扑、几何学、运动学以及逻辑,而逻辑涵盖了序列、行为和控制。因此,面向对象的数据结构、几何学、运动学和逻辑就成为了工程领域中数据交换的一个重要焦点。

AML 整合现有用来在不同领域内存储和交换工程信号的工业数据格式。这些数据格式按照各自的规范独立实施,并不属于 AML 的分支。

AML 的核心是连接不同数据格式的顶层数据格式 CAEX。因此 AML 有其固有的分布式文档架构。

图 1 描述了 AML 的基本架构以及拓扑、几何学、运动学和逻辑信息分布。

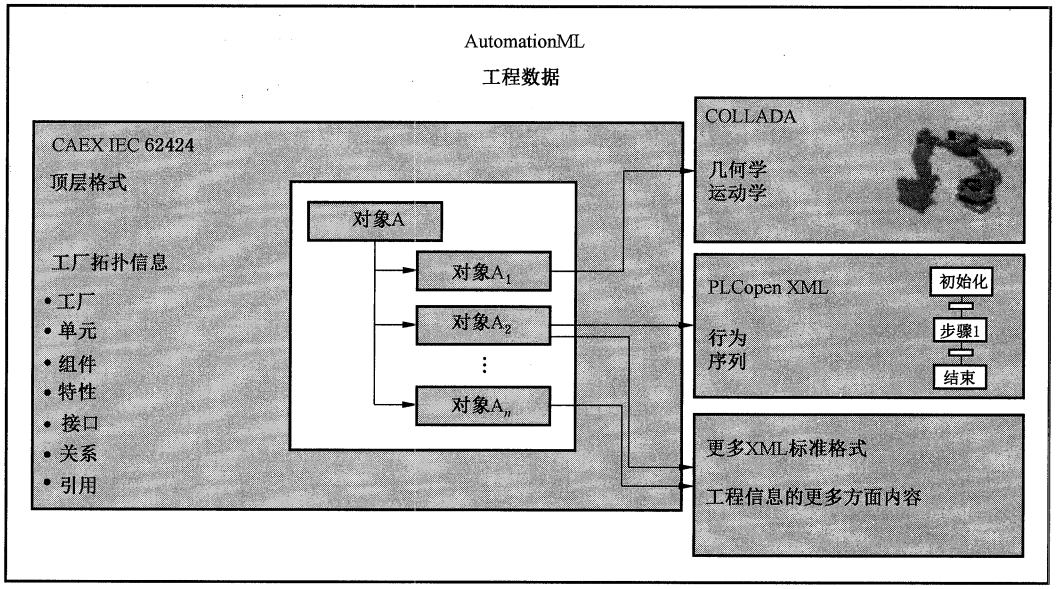


图 1　工程数据交换格式 AML 总览

由于 AML 包含不同的方面,IEC 62714 由针对不同方面的几个部分组成:

——第 1 部分:架构和通用要求,该部分规定了 AML 的架构、工程数据的建模、类、实例、关系、引用、分层结构、AML 基础库和扩展 AML 概念。它是现有和未来所有其他部分的基础,并且为引用其他子格式提供了机制。

——第 2 部分:角色类库,该部分会规定额外的 AML 库。

——第 3 部分:几何学和运动学,该部分会描述几何和运动信息的建模。

——第 4 部分:逻辑,该部分会描述与逻辑、序列、行为和控制相关的信息的建模。

为了将更多的数据标准与 AML 联系起来，以后可能会增加更多的部分。

鉴于没有更多部分描述了对更多标准的整合，因此应着重关注有限的一组子数据格式。否则任何数据格式都将被使用，从而导致无法进行数据交换。

附录 A 给出了 AML 的资料性信息、用例和示例。

附录 B 给出了本部分定义的 AML 基础库的 XML 表达实例。

工业自动化系统工程用工程数据交换格式 自动化标记语言 第1部分:架构和通用要求

1 范围

GB/T 39003的本部分规范了自动化标记语言的架构和通用要求,以便对在工业自动化和控制系统工具之间交换的工程信息进行建模。相关工具的导出/导入应用也可参照本部分的规定。

本部分没有定义数据交换过程的细节以及导出/导入工具的使用要求。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是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。

ISO/PAS 17506 工业自动化系统与集成 工业数据三维可视化用COLLADA数字资产模式规范(Industrial automation systems and integration—COLLADA digital asset schema specification for 3D visualization of industrial data)

ISO/IEC 29500-2 信息技术 文件描述和处理语言 办公开放式XML文件格式 第2部分:开放式打包协议(Information technology—Document description and processing languages—Office Open XML File Formats—Part 2:Open Packaging Conventions)

IEC 62424:2016 过程控制工程的表示法 P&I图表以及P&ID工具和PCE-CAE工具之间数据交换的要求(Representation of process control engineering—Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools)

IEC 62714(所有部分) 工业自动化系统工程的工程数据交换格式 自动化标记语言(Engineering data exchange format for use in industrial automation systems engineering—Automation Markup Language)

IETF RFC 2046 多用途因特网邮件扩展(MIME) 第2部分:媒体类型(Multipurpose Internet Mail Extensions (MIME)—Part Two:Media Types)[查看2017-11-13].可从以下网址获得:(http://www.ietf.org)

IETF RFC 4122 通用唯一标识符(UUID)URN命名空间(A Universally Unique Identifier (UUID) URN Namespace)[查看2017-11-13].可从以下网址获得:(http://www.ietf.org)

IETF RFC 5646 标识语言标签(Tags for Identifying Languages)[查看2017-11-13].可从以下网址获得:(http://www.ietf.org)

COLLADA 1.4.1:2008年3月,COLLADA-数字资产架构发布1.4.1[查看2017-11-13],可从以下网址获得:(http://www.khronos.org/files/collada_spec_1_4.pdf)

PLC开放XML 2.0:2008年12月3日以及PLC开放XML 2.0.1:2009年5月8日,IEC 61131-3的XML格式[查看2017-11-13],可从以下网址获得:(http://www.plcopen.org)

3 术语和定义、缩略语

3.1 术语和定义

下列术语和定义适用于本文件。

3.1.1

自动化标记语言 automation markup language;AML

符合 IEC 62714 的基于 XML 的工厂工程数据交换格式。

3.1.2

自动化对象 automation object

自动化系统中的物理或逻辑实体。

注：一个自动化对象是：一个自动化组件、一个阀门或一个信号。

3.1.3

AML 对象 AML object

具有一个或多个与一个 AML 角色类有关联的 CAEX RoleRequirements 的一个自动化对象的数据表达。

注：AML 对象是自动化标记语言的核心元素。它们代表了一个实例，并且可包括管理元素、属性、接口、关系和引用。

3.1.4

AML 类 AML class

预定义的 AML 对象类型，可以是 AML 系统单元类，AML 接口类，AML 角色类或 AML 属性类型。

注 1：AML 类存储于 AML 库中，AML 类的类型是 SystemUnitClass，InterfaceClass，RoleClass 或 AttributeType。

注 2：AML 类定义了可再使用的解决方案，其特征是属性、接口和聚合对象。

注 3：AML 类能被多重实例化。

注 4：AML 类是用户自定义的或标准的 AML 类。

3.1.5

AML 属性 AML attribute

属于 AML 对象并与 AML 类或 AML AttributeType 中定义的属性相关的 CAEX 属性。

注：AML 属性被描述为一个符合 IEC 62424:2016 中 A.2.4 相关规定的 XML 元素。

3.1.6

AML 文档 AML document

遵循 IEC 62714(所有部分)的 AML CAEX 文档及所有被引用的子文档。

注 1：AML 文档可被存储为文件、字符串或数据流。

注 2：AML 文档包含 AML 对象和/或用户自定义对象。

注 3：AML 文档可由以一个 AML CAEX 文档作为根文件的多个文件构成。

3.1.7

AML 文件 AML file

遵循本部分的 CAEX 文件。其扩展名为.aml，且不包括任何被引用的子文件。

3.1.8

AML 接口 AML interface

与 AML 接口类有关系的单一连接点。

注：接口允许按 CAEX 内部链接的定义描述对象之间的关系，例如信号接口、设备接口或电源接口。

3.1.9

AML 库　AML library

包含 AML 类的库。

3.1.10

AML 端口　AML Port

与标准 AML 接口类 Port 有直接或间接关系的 AML 接口，并允许指定嵌套接口。

注：端口属于一个父 AML 对象，它描述了该对象中各类复杂的接口。在更高的抽象级别中，端口之间能够实现互联。

3.1.11

AML 组　AML Group

与标准 AML 角色类 Group 有直接或间接关系的 AML 对象，并提供 AML 对象的特定视图。

3.1.12

AML 面　AML Facet

与标准 AML 角色类 Facet 具有直接或间接关系的 AML 对象，并提供一个 AML 对象的 AML 属性或接口的特定视图。

3.1.13

计算机辅助工程数据交换格式　computer aided engineering data exchange format;CAEX

中性的基于 XML 的数据格式。

注：CAEX 是一种中性的数据格式，符合 IEC 62424:2016 中第 7 章、附录 A 和附录 C 的规定。

3.1.14

副本-实例关系　copy-instance-relation

实例与其相关类之间的一种关系，该实例通过复制类的数据结构被创建。

注：任何一个实例会获得一个具备其所属 AML 类所有属性和特性的副本。对于该类的修改不会自动导致该实例的变化。在该实例内部，类的特性是独立的。在知道所属 AML 类的情况下，可能存在多个副本。

3.1.15

通用唯一标识符　universal unique identifier;UUID

AML 对象具有的唯一标识符。

3.1.16

全局唯一标识符　global unique identifier;GUID

通用唯一标识符的应用形式。

注 1：实际 GUID 示例：{AC76BA86-7AD7-1033-7B44-A70000000000}。

注 2：在 IEC 62714(所有部分)中，GUID 通常以简短的形式表示，如“GUID1”“GUID2”等，以代替实际使用的 GUID，并增加文本的可读性。

3.1.17

继承关系　inheritance relation

两个 AML 类之间的一种关系。

注：子类继承父类所有的属性与特征。

3.1.18

实例　instance

一个独立的物理或逻辑元素的数据表达形式。

注：实例能通过如聚合对象或属性得到扩展。

3.1.19

拓扑结构　topology

一个系统的分层结构，通常由对象树表示。

注：该概念包含多重分层结构、交叉结构和对象网络。

3.1.20

工厂拓扑结构　plant topology

一个工厂的分层结构，通常由对象树表示。

3.1.21

发布　publish

通过在外部文档中对数据结构进行建模以便在 CAEX 中使用。

注：允许在独立外部文档数据结构之间的关系做出定义。

3.1.22

关系　relation

CAEX 对象之间的联系。

注：关系的举例为父子关系和类-实例关系。

3.1.23

链接　link

CAEX 外部接口类型对象之间的连接。

注：一个链接按照 CAEX 内部链接方式建模。

3.1.24

引用　reference

CAEX 内部元素和外部存储信息之间的关联。

3.2　缩略语

表 1 列出的缩略语适用于本文件。

表 1　缩略语

缩略词	中文解释	英文全称
AML	自动化标记语言	Automation Markup Language
CAE	计算机辅助工程	Computer Aided Engineering
CAEX	计算机辅助工程数据交换格式	Computer Aided Engineering Exchange
COLLADA	协同设计行为	Collaborative Design Activity
GUID	全局唯一标识符	Global Unique Identifier
HMI	人机接口	Human Machine Interface
ID	标识符	Identifier
MES	制造执行系统	Manufacturing Execution System
PLC	可编程逻辑控制器	Programmable Logic Controller
URL	统一资源定位符	Uniform Resource Locator
URI	统一资源标识符	Uniform Resource Identifier
UUID	通用唯一标识符	Universal Unique Identifier
XML	可扩展标记语言	Extensible Markup Language

4　一致性

为声明对 AML 的支持以及与本部分的一致性，应满足本部分第 5 章、第 6 章、第 7 章和第 8 章中的要求。

5 AML 架构规范

5.1 概述

AML 的核心是顶层数据结构 CAEX。它是一种中性的数据结构，符合 IEC 62424:2016 中第 7 章、附录 A 和附录 C 的规定，并将拓扑、几何学、运动学、行为和序列信息等工程元素的已有数据格式进行互相关联。所以，AML 的一个基本特征是关于上述工程元素的固有的分布式文档架构。

图例仅具有解释性，其表达内容不作为标准规范。

5.2 通用 AML 架构

下列规定适用于通用 AML 架构：

工厂拓扑信息：工厂拓扑是工厂工程信息中的顶层数据结构，应根据 IEC 62424:2016 中第 7 章、附录 A 和附录 C 的规定，通过数据格式 CAEX 进行建模。CAEX 的语义扩展内容可单独描述。多层次和跨层次结构应根据 IEC 62424:2016 中 A.2.8.7 的规定通过镜像对象进行应用。

注 1：根据 IEC 62424:2016 中 A.2.8.7 规定，一个与另一个 AML 对象有关系的 AML 对象被称为“镜像对象”，而与之相关的 AML 对象被称为“主对象”。镜像对象被认为与主对象完全相同，从而实现将一个对象实例放入不同的工厂层次结构中，并允许通过交叉结构对复杂对象网络建模。

注 2：IEC 62714(所有部分)没有在语法上修改 CAEX 数据格式。本部分 A.1.2 和 IEC 62424:2016 的附录 D 给出了工厂拓扑的资料性概述和额外示例。

引用和关系信息：引用和关系应按照 5.5 和 5.6 的规定存储。外部存储信息之间的关系应按照 CAEX 的方法存储。在必要时，相关链接涉及的对象应作为 CAEX 外部接口在 CAEX 工厂拓扑描述中发布，它们应派生自 6.3 中定义的 AML 标准接口类。

注 3：引用描绘了 CAEX 对象和外部存储信息之间的链接。A.1.7 给出相关资料性概述。IEC 62714 的补充部分描述了引用和接口的发布。

注 4：关系描绘了 CAEX 对象之间的关联。

几何学和运动学信息：几何学和运动学的相关信息应使用 COLLADA™2 的数据格式存储。需要在顶层格式中互相关联的 COLLADA 接口应作为 CAEX 外部接口发布。

注 5：IEC 62714(所有部分)没有在语法上修改 COLLADA 的数据格式。A.1.3 给出如何引用 COLLADA 的实例。具体细节会在 IEC 62714 的第 3 部分中规定。

注 6：借助于不同对象的 COLLADA 几何学信息，能自动得到一个完整的场景。这些文件可能从 CAEX 引用，也可能通过 CAEX 链接机制互相链接。

逻辑信息：逻辑信息应按照 PLCopen XML 数据格式存储。需要在顶层数据格式中互相关联逻辑项，例如变量或信号，应作为 CAEX 外部接口发布。所有在顶层数据格式中发布的 PLCopen XML 条目应在 PLCopen XML 中具有唯一的 ID。

注 7：逻辑信息描述了动作的序列以及对象的内在行为，例如 I/O 连接和逻辑变量。IEC 62714 没有修改 PLCopen XML 的格式。A.1.4 给出如何引用逻辑信息的资料性概述。具体细节会在 IEC 62714 的第 4 部分中规定。

引用其他数据格式：IEC 62714 在将来可通过增加更多的部分得到扩展。这些部分将规定更多采用 AML 引用机制的 XML 数据格式的整合。更多细节可能在 IEC 62714 的额外部分中进行定义。

AML 数据结构不提供对约束、属性值、关系、引用和包含数据语义正确性的一致性检验：这些应由源工具或目标工具，或者相应的导入/导出应用完成。AML 对其文档仅允许语法的证明，而不是对应的架构。

5.3 AML 文档版本

IEC 62714 是基于以下文档格式：

——CAEX，版本号 3.0；

——PLCopenXML 2.0 和 2.0.1；

——ISO/PAS 17506 中规定的 COLLADA 1.5.0 和 COLLADA 1.4.1；

——本部分及 IEC 62714 的其他部分规定的 AML 标准库。

AML 集成了 CAEX，因此 AML 是更高一级标准。

注 1：与 AML 对象实例相关的版本信息的标准化内容在 8.7 中定义。工具特定元信息的存储在 5.4 中定义。

因此，以下规定适用：

——每个 AML CAEX 文档应根据 IEC 62424:2016 的 A.2.2.3 将 IEC 62714(所有部分)所遵循的 AML 版本存储在 CAEX 元素“SuperiorStandardVersion”中。

——该元素的值应为“AutomationML 2.10”以符合 IEC 62714(所有部分)。

——每一个被引用的 CAEX 文档应遵循根元素的 AML 版本。明确禁止混用不同 AML 版本的文档。

——每一个被引用的外部文档应遵循上层 AML 版本规范中规定的指定架构版本。明确禁止混用在 AML 版本规范之外的外部文档版本。

图 2 给出了遵循 AML2.10 版本的 CAEX 文档的 XML 文本。

```
<CAEXFile xmlns="http://www.dke.de/CAEX" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
SchemaVersion="3.0" FileName="AutomationML2.10BaseLibraries" xsi:schemaLocation="
http://www.dke.de/CAEX CAEX_ClassModel_V.3.0.xsd">
  <SuperiorStandardVersion>AutomationML 2.10</SuperiorStandardVersion>
```

图 2 AML 文档版本信息

——每一个 AML 标准库和用户自定义 AML 库应使用 CAEX 元素“Version”定义其版本号。本部分没有定义版本号数值的句法。

——在必要时，CAEX 类应使用 CAEX 元素“Version”定义其版本号。本部分没有定义 AML 库中类的版本号的句法和语义。

——不同版本的相同库禁止存储在同一个 AML 文件。

注 2：这确保了 AML 文件中 AML 库名称的唯一性。

——一个 AML 文档的创建器应确保只有版本兼容的类和外部文档被引用。

5.4 AML 源工具的元信息

当将用户自定义的数据从源工具转移到目标工具时，需要把源工具的信息直接存储到 AML 文档中。

因此，以下规定适用：

——根据 IEC 62424:2016，任何 AML 文档应提供编写该文档的源工具信息。

——在一个数据交换工具链中，所有参与的工具应用同样的方式将该信息存储于 CAEX 文档中。于是该文档可包含数据交换工具链中多种工具的信息。一种工具可移除其他工具的编写信息，这能阻碍与其他工具间的迭代式数据交换。因此，不建议移除其他工具的编写信息。

——为了识别每个 AML 对象实例(InternalElement，ExternalInterface)的源工具，本部分建议根据 IEC 62424:2016 的 A.2.2.7 使用可选的 CAEX 元素 SourceObjectInformation 及其属性 OriginID 和 SourceObjID。

图 3 给出了所需文档来源信息所需的 XML 文本。该示例显示了本部分提供的标准库的源信息。

```
<SourceDocumentInformation OriginID="IEC SC65E WG 9" OriginName="IEC SC65E WG 9"
OriginVersion="2.10.0" LastWritingDateTime="2016-08-25T09:58:00.0Z" OriginProjectID="Automation
Markup Language Standard Library" OriginRelease="2.10.0" OriginVendor="IEC" OriginVendorURL="
www.iec.ch" OriginProjectTitle="Automation Markup Language Standard Libraries"/>
```

图 3 AML 源工具信息的 XML 文本

5.5 AML 关系规范

5.5.1 概述

鉴于对象的重要性，有必要建立一个对象与对象之间联系的方法。本部分引入了两种存储这些联系信息的方法：关系和引用。5.5 对关系进行规范，5.6 对引用进行规范。A.1.7 给出关于关系和引用的资料性概述。

5.5.2 类-实例关系

实例的特征是由一个唯一标识符和一组参数进行表达的。以下规定适用于类-实例关系：

——AML 对象应作为 CAEX 实例分层结构或系统单元类中的 CAEX 内部元素进行建模；

——AML 对象可是一个单例，与任何系统单元类都不存在关系；

注 1：然而一个 AML 对象与标准 AML 角色类存在关系。

注 2：与 AML 基础角色不存在关系的实例可能存在，并且是一个用户自定义的对象。它们不是 AML 对象。

——根据 IEC 62424:2016 的 A.2.2.7，源类的改变应产生一个不同名称的类的新版本。在新类中，旧版本类的完整路径应存储在 CAEX 标签“OldVersion”中。另外，在旧类中，到新版本的路径宜存储在 CAEX 标签“NewVersion”中。

注 3：该规定支持在类的不同版本间进行改动追踪。

5.5.3 实例-实例关系

实例-实例关系是两个任意 AML 对象的接口之间的关系。

以下规定适用于实例-实例关系：

——外部接口应直接或间接地派生自 AML 标准接口类之一；

注 1：AML 标准接口类库在 6.3 中定义。AML 接口类定义了接口和链接的语义信息。一个在接口和一个没有引用的接口类之间的链接是没有语义的。

——COLLADA 文档可互相链接。对应的 COLLADA 接口可以是包含有效 URI 的任一元素。如果这些节点需要在 CAEX 中互相链接，它们应通过增加一个对应对象的外部接口在 CAEX 中发布。该外部接口应派生自 AML 标准接口类“COLLADAInterface”或它的某一派生类；

注 2：标准接口类“COLLADAInterface”在 6.3.7 中规定，其细节在 IEC 62714 第 3 部分中规定。

——PLCopen XML 文档可通过对应的 PLCopen XML 接口互相链接。如果 PLCopen XML 元素需要在 CAEX 中互相链接，它们应通过增加一个对应对象的外部接口在 CAEX 中发布。该外部接口应派生自 AML 标准接口类“PLCopenXMLInterface”或它的某一派生类；

注 3：标准接口类“PLCopenXMLInterface”在 6.3.8 中规定，其细节在 IEC 62714 第 4 部分中规定。

图 4a)给出了包含一个机器人“Rob1”和一个 PLC“PLC1”的示例，它们各自都有一个互相连接的信号接口。图 4b)描绘了该实例的对象分层结构。

a) 关系表示示例——框图　　b) 关系表示示例——对象树

图 4 关系表示示例——框图、对象树

图 5 和图 6 给出了上述示例的 AML 表达，和本示例中实例分层结构的完整 XML 文本，包含所有

的内部元素，包括工位“Station”、机器人“Rob1”、PLC“PLC1”和主板“Board01”，以及它们各自的接口。

注 4：本示例(见图 4)中的路径字符串被略写为“/../”以增加阅读性。

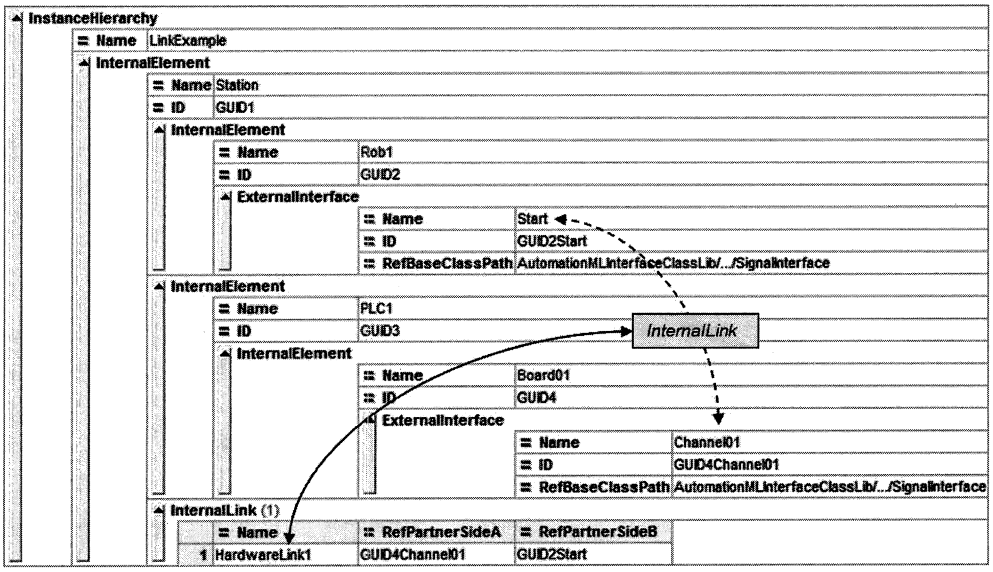


图 5　对象“PLC1”和“Rob1”之间关系示例

```
<InstanceHierarchy Name="LinkExample">
  <InternalElement Name="Station" ID="GUID1">
    <InternalElement Name="Rob1" ID="GUID2">
      <ExternalInterface Name="Start" ID="GUID2Start" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface"/>
    </InternalElement>
    <InternalElement Name="PLC1" ID="GUID3">
      <InternalElement Name="Board01" ID="GUID4">
        <ExternalInterface Name="Channel01" ID="GUID4Channel01" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface"/>
      </InternalElement>
    </InternalElement>
    <InternalLink Name="HardwareLink1" RefPartnerSideA="GUID4Channel01" RefPartnerSideB="GUID2Start"/>
  </InternalElement>
</InstanceHierarchy>
```

图 6　对象“PLC1”和“Rob1”之间关系示例的 XML 文本

5.5.4　对象识别

本部分建议根据 RFC 4122 通过 GUID 识别 InternalElements 和 ExternalInterfaces。为了比较两个标识符，以下规定适用：

——如果 InternalElement 或 ExternalInterface 的标识符是一个 GUID，那么当所包含的数值相同时，两个标识符应是相同的。允许使用方括号、括号、破折号或空格，但应无关紧要。

没有括号的示例：48d23207-09e0-4104-82fb-344007d2b7f5

有括号的示例：{ 48d23207-09e0-4104-82fb-344007d2b7f5 }

5.6　AML 文档引用规范

5.6.1　概述

文档的引用可用作一个 AML 对象和一个能包含几何学、运动学或序列信息的外部文档的链接。引用方法基于 AML 标准接口“ExternalDataConnector”或它的某一派生类。

5.6.2　引用 COLLADA 文档

引用 COLLADA 文档应基于 AML 标准接口类“COLLADAInterface”或它的某一派生类。该类在

6.3.7 中规定。具体细节在 IEC 62714 第 3 部分中规定。

5.6.3 引用 PLCopen XML 文档

引用 PLCopen XML 文档应基于 AML 标准接口类“PLCopenXMLInterface”或它的某一派生类。该类在 6.3.8 中规定。具体细节在 IEC 62714 第 4 部分中规定。

5.6.4 在 IEC 62714(所有部分)范围内引用其他文档

未来对 IEC 62714 的扩展将增加更多的接口类型,用来引用额外的文档类型。它们将在 IEC 62714 各自部分中单独规定,而不在该系列的本部分的范围内。以下规定适用于这些扩展:

——当 IEC 62714 中增加额外文档类型时,它们应使用额外的接口类进行建模。

——这些额外接口应作为 AML 接口类库的扩展进行建模,并应直接或间接地派生自标准接口类“外部数据连接器”。

——应使用标准接口类提供的同一个标准属性进行引用信息的存储。

5.6.5 在 IEC 62714(所有部分)范围外引用文件

如 AML 文件需引用不属于 IEC 62714(所有部分)范围的外部文件(如手册,指令或特定的工程结果,例如本地控制程序),则一定要遵循下列规定:

——一个超出 IEC 62714 范围的文档应通过一个 CAEX InternalElement 建模,该 CAE InternalElement 与 6.4.12 中定义的 RoleClass“ExternalData”直接或间接相关。引用的 RoleClass 应指定文档的内容。包含内容类型的多个角色能分配给一个文档。

注 1:每个文件能包含几种类型的内容,例如,物料清单和用户手册。

注 2:如果需要,每个文档能引用若干文件,例如,如果它分裂在不同的文件。

——如果一个文件有语言特定,则它应包含类型为“DocLang”的 CAEX 属性。如果一个文件包含多种语言,则它应该包含一个未分类的属性列表,如 8.8 所述,属性类型为“DocLang”。

——每个文档应包含一个或多个 ExternalInterfaces,它们应直接或间接地派生自接口类“ExternalDataReference”。

——这个 ExternalInterface 应通过从 AML 标准接口类“ExternalDataConnector”继承的“refURI”类型的预定义的 CAEX 属性来模拟 URI 至外部文档,并且应额外通过 AML 标准接口类“ExternalDataReference”继承的“MIMEType”类型的预定义的 CAEX 属性“MIMEType”来模拟文件类型。

附录 A 的 A.1.5 提供了更多的信息和示例。

5.6.6 将 CAEX 属性引用到外部文件中的条目

如果一个 CAEX 属性一定要与一个外部文件中的一个相关条目相关联[例如,一个外部 XML 文件或一个超出 IEC 62714(所有部分)范围的 Excel 文件],则适用下列规定:

——一个 CAEX 属性和一个外部文件中的一个条目之间的每个引用都应按照 5.6.5 中规定,由 CAEX ExternalInterface 建模。

——对于一个 CAEX 属性和一个外部文件中的一个条目之间的每个引用,此 CAEX ExternalInterface 将模拟具有嵌套属性的“AssociatedExternalValue”类型的一个附加属性。

——第一个嵌套属性应镜像 CAEX 属性。这意味着以“/”分隔的属性父对象的 GUID 和属性的名称在 CAEX 属性“RefAttributeType”中建模。这个属性的名称应是不相关的,但在其同类中是不同的。

——“refURI”类型的第二个嵌套属性应引用该外部文档中的该条目。这个引用应是 5.6.5 中定义

的父 InternalElement 引用的外部文档的相同文档或子文档。这个引用的语法超出了 IEC 62714(所有部分)的范围,并且需要一个可引用的外部文档元素。

——"Direction"类型的第三个嵌套属性将模拟信息流的方向。如果外部属性被 CAEX 属性使用(则外部条目占先),那么属性值应为"In";如果 CAEX 属性占先,并且外部条目使用了 CAEX 属性的值,那么属性值应为"Out"。不准许出现"InOut"值。

A.1.6 提供了更多的信息和示例。

6 AML 基础库

6.1 概述

本章定义了用于对 AML 核心概念建模的 AML 基础类和 AML 基础库。所有涉及的属性都是 AML 标准库的一部分,在不需要时可从实例分层结构中移除。

注:特定域的库是属于 IEC 62714 其他部分的范围。

6.2 通则

以下规定适用于 AML 基础库:

——所有 AML 对象应直接或间接地与角色类"AutomationMLBaseRole"相联系;

——所有接口应直接或间接地与一个标准接口类"AutomationMLBaseInterface"相联系。

6.3 AML 接口类库——AutomationMLInterfaceClassLib

6.3.1 概述

以下涉及的 AML 接口类库的建模符合 IEC 62424:2016 中第 7 章、附录 A 和附录 C 的要求。IEC 62714(所有部分)采用了 CAEX 接口的概念。AML 库的用户自定义扩展在 7.4 中规定。

每一个接口应直接或间接地派生自表 2 中给出的标准 AML 接口类库。6.3.2~6.3.11 对接口类的细节进行规定。

表 2 AML 接口类库中的接口类

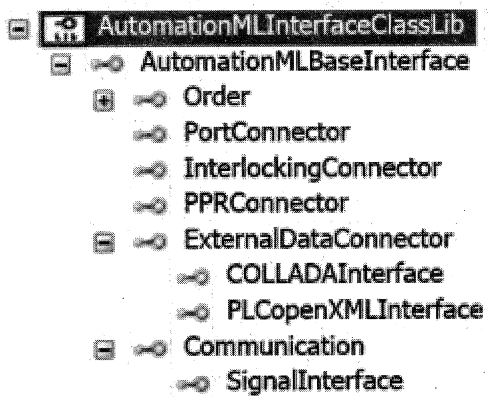
AML 接口类库	接口类	描述
	AutomationMLBaseInterface	抽象的接口类型
	Order	描述顺序的接口
	Port	描述和内部链接端口的接口
	PPRConnector	连接内部链接产品、资源或过程的连接器
	ExternalDataConnector	通用连接器至外部数据的接口
	COLLADAInterface	COLLDA 文档的接口
	PLCopenXMLInterface	PCLopen XML 文档的接口
	ExternalDataReference	在 IEC 62714(所有部分)以外范围的外部文档的接口
	Communication	通用通信接口
	SignalInterface	通用信号接口

图 7 给出了表格视图，图 8 给出了标准 AML 接口类库的 XML 文本。6.3.2～6.3.11 给出了关于这些类的细节信息。

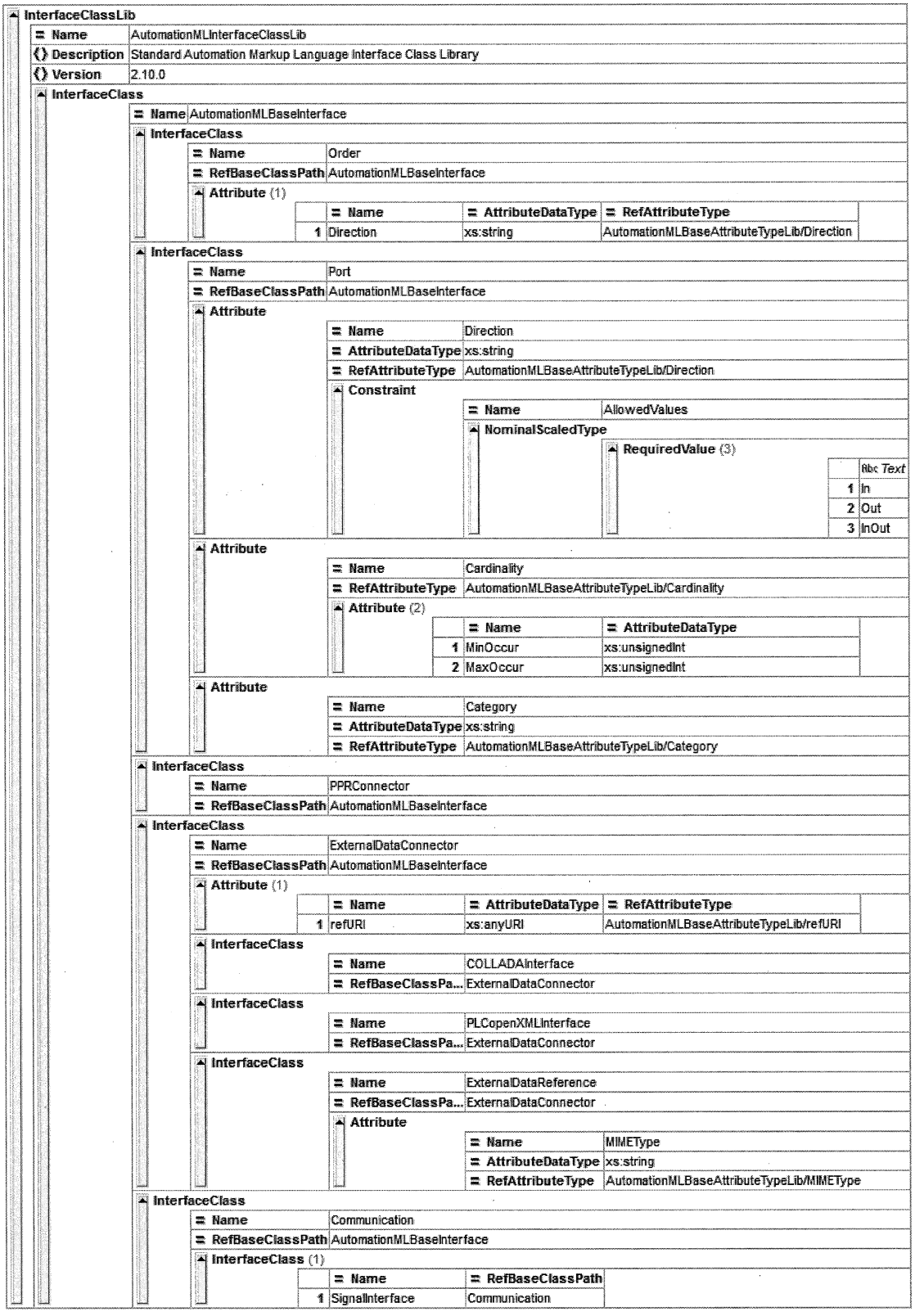


图 7　AML 基础接口类库——表格视图

```
<InterfaceClassLib Name="AutomationMLInterfaceClassLib">
  <Description>Standard Automation Markup Language Interface Class Library</Description>
  <Version>2.10.0</Version>
  <InterfaceClass Name="AutomationMLBaseInterface">
    <InterfaceClass Name="Order" RefBaseClassPath="AutomationMLBaseInterface">
      <Attribute Name="Direction" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction"/>
    </InterfaceClass>
    <InterfaceClass Name="Port" RefBaseClassPath="AutomationMLBaseInterface">
      <Attribute Name="Direction" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction">
        <Constraint Name="AllowedValues">
          <NominalScaledType>
            <RequiredValue>In</RequiredValue>
            <RequiredValue>Out</RequiredValue>
            <RequiredValue>InOut</RequiredValue>
          </NominalScaledType>
        </Constraint>
      </Attribute>
      <Attribute Name="Cardinality" RefAttributeType="AutomationMLBaseAttributeTypeLib/Cardinality">
        <Attribute Name="MinOccur" AttributeDataType="xs:unsignedInt"/>
        <Attribute Name="MaxOccur" AttributeDataType="xs:unsignedInt"/>
      </Attribute>
      <Attribute Name="Category" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Category"/>
    </InterfaceClass>
    <InterfaceClass Name="PPRConnector" RefBaseClassPath="AutomationMLBaseInterface"/>
    <InterfaceClass Name="ExternalDataConnector" RefBaseClassPath="AutomationMLBaseInterface">
      <Attribute Name="refURI" AttributeDataType="xs:anyURI" RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI"/>
      <InterfaceClass Name="COLLADAInterface" RefBaseClassPath="ExternalDataConnector"/>
      <InterfaceClass Name="PLCopenXMLInterface" RefBaseClassPath="ExternalDataConnector"/>
      <InterfaceClass Name="ExternalDataReference" RefBaseClassPath="ExternalDataConnector">
        <Attribute Name="MIMEType" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/MIMEType"/>
      </InterfaceClass>
    </InterfaceClass>
    <InterfaceClass Name="Communication" RefBaseClassPath="AutomationMLBaseInterface">
      <InterfaceClass Name="SignalInterface" RefBaseClassPath="Communication"/>
    </InterfaceClass>
  </InterfaceClass>
</InterfaceClassLib>
```

图 8　AML 基础接口类库——XML 表述

6.3.2　接口类 AutomationMLBaseInterface

表 3 规定了接口类“AutomationMLBaseInterface”。

表 3　接口类 AutomationMLBaseInterface

类名	AutomationMLBaseInterface
描述	接口类“AutomationMLBaseInterface”是一个基础的抽象接口类型，应作父类用于所有 AML 接口类的描述中
父类	无
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
属性	无

6.3.3　接口类 Order

表 4 规定了接口类“Order”。

表 4　接口类 Order

类名	Order
描述	接口类“Order”是一个抽象类，应用于描述顺序，例如前驱、后继
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
属性	名称：Direction RefAttributeType：AutomationMLBaseAttributeTypeLib/Direction 语义：见 6.5.2

6.3.4　接口类 Port

表 5 规定了接口类“Port”。

表 5　AML 接口 Port 的可选属性

类名	Port
描述	接口类“Port”是一个包含许多内部接口的接口类型，它通过这种方式可以描述复杂接口。AML 接口 Port 应引用这个接口类。详细说明和示例在 8.2 中规定
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Port
属性	名称：Direction RefAttribute 类型：AutomationMLBaseAttributeTypeLib/Direction 语义：见 6.5.2
	名称：Cardinality RefAttribute 类型：AutomationMLBaseAttributeTypeLib/Cardinality 语义：见 6.5.2
	名称：Category RefAttributeType：AutomationMLBaseAttributeTypeLib/Category 语义：见 6.5.2

6.3.5　接口类 PPRConnector

表 6 规定了接口类“PPRConnector”。

表 6　接口类 PPRConnector

类名	PPRConnector
描述	接口类“PPRConnector”应用于为资源、产品和过程提供一种关系。A.2.5 给出了更多的信息
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
属性	无

6.3.6 接口类 ExternalDataConnector

表 7 规定了接口类“ExternalDataConnector”。

表 7 接口类 ExternalDataConnector

类名	ExternalDataConnector
描述	接口类“ExternalDataConnector”是一个基础的接口抽象类型，应用于描述引用了外部文档的连接器接口。“COLLADAInterface”类和“PLCopenXMLInterface”类派生自该类。所有现有的和将来的连接器类都应直接或间接地派生自该类
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector
属性	名称：refURI RefAttributeType：AutomationMLBaseAttributeTypeLib/refURI 语义：见 6.5.2

6.3.7 接口类 COLLADAInterface

表 8 规定了接口类“COLLADAInterface”。IEC 62714-3 规定了其细节。

表 8 接口类 COLLADAInterface

类名	COLLADAInterface
描述	接口类“COLLADAInterface”应用于引用外部 COLLADA 文档，及发布在外部 COLLADA 文档内部定义的接口。IEC 62714-3 规定其细节
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/COLLADAInterface
属性	无

6.3.8 接口类 PLCopenXMLInterface

表 9 规定了接口类“PLCopenXMLInterface”。IEC 62714-4 规定了其细节。

表 9 接口类 PLCopenXMLInterface

类名	PLCopenXMLInterface
描述	接口类“PLCopenXMLInterface”应用于引用外部 PLCopen XML 文档，及发布在 PLCopen XML 逻辑描述内部定义的信号或变量。IEC 62714-4 中规定其细节
父类	AutomationMLBaseInterface/ExternalDataConnector
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/PLCopenXMLInterface
属性	无

6.3.9 接口类 ExternalDataReference

表 10 规定了接口类“ExternalDataReference”。5.6.4 规定了其细节。

表 10 接口类 Communication

类名	ExternalDataReferface
描述	接口类“ExternalDataReference”应用于引用 AML 范围外的外部文档。5.6.5 规定其细节
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/ExternalDataReference
属性	名称:MIMEType RefAttributeType:AutomationMLBaseAttributeTypeLib/MIMEType 语义:见 6.5.2

6.3.10 接口类 Communication

表 11 规定了接口类“Communication”。

表 11 接口类 Communication

类名	Communication
描述	接口类“Communication”是一个抽象接口类型,应用于与通信相关的接口描述。更多的通信相关类应直接或间接地派生自该类
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication
属性	无

6.3.11 接口类 SignalInterface

表 12 规定了接口类“SignalInterface”。

表 12 接口类 SignalInterface

类名	SignalInterface
描述	接口类“SignalInterface”应用于对信号进行建模。该接口类型是可以配置的,允许数字、模拟输入、输出信号,以及可配置的输入-输出信号的描述。图 4 给出了一个示例
父类	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication
元素引用路径	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication/SignalInterface
属性	无

6.4 AML 基础角色类库——AutomationMLBaseRoleClassLib

6.4.1 概述

本条定义了 AML 核心概念建模时所需要的标准角色类的 AML 基础库。一个角色是一个描述抽象功能却未定义底层技术实现的类。

一个角色能通过 CAEX SupportedRoleClass(es)的方式与一个系统单元类(SystemUniteClass)。这表明类能够支持引用的角色。支持多个 SupportedRoleClass。映射对象(MappingObject)提供角色属性和接口与 SystemUniteClass 的属性和接口之间的映射。一旦 SystemUniteClass 被实例化,相关的内部元素(InternalElement)保持该信息。然而,这并不说明实例在其范围内行使了所有角色。

CAEX RoleRequirements 为 CAEX InternalElement 实际的或请求的角色建模。支持 RoleRequirements。实际的角色被引用,关于角色的各自的实例的要求在 RoleRequirement 内部进行建模。如果需要,MappingObject 允许在角色类和 InternalElement 之间映射属性和接口。

当一个角色类与一个 AML 对象相联系的时候,该 AML 对象获得了语义。IEC 62714 第 2 部分会描述更多的扩展库。

所有涉及的属性都是 AML 标准库的一部分,在不需要时可以从实例分层结构中移除。

任何 AML 对象和用户自定义的角色类应与该 AML 库中的一个角色存在直接或间接引用关系。如果一个角色过于具体,应引用下一个父类。图 9～图 11 给出了标准基础角色类所对应的对象树、XML 表格和 XML 文本。6.4.2～6.4.12 给出每个角色类的细节。

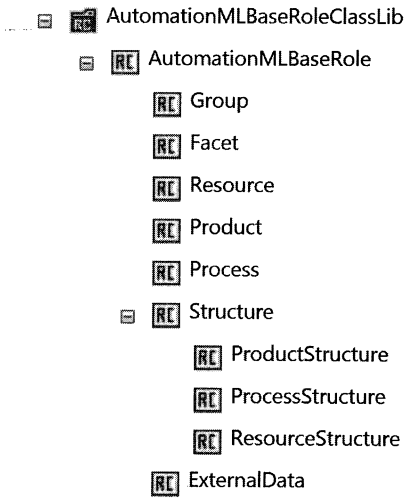


图 9 AML 基础角色类库——对象树

RoleClassLib

Name	AutomationMLBaseRoleClassLib
Description	Automation Markup Language base role class library
Version	2.10.0

RoleClass

Name	AutomationMLBaseRole

RoleClass

Name	Group
RefBaseClassPath	AutomationMLBaseRole

Attribute

Name	AssociatedFacet
AttributeDataType	xs:string
RefAttributeType	AutomationMLBaseAttributeTypeLib/AssociatedFacet

RoleClass

Name	Facet
RefBaseClassPath	AutomationMLBaseRole

RoleClass

Name	Resource
RefBaseClassPath	AutomationMLBaseRole

RoleClass

Name	Product
RefBaseClassPath	AutomationMLBaseRole

RoleClass

Name	Process
RefBaseClassPath	AutomationMLBaseRole

RoleClass

Name	Structure
RefBaseClassPath	AutomationMLBaseRole

RoleClass (3)

	Name	RefBaseClassPath
1	ProductStructure	Structure
2	ProcessStructure	Structure
3	ResourceStructure	Structure

RoleClass

Name	ExternalData
RefBaseClassPath	AutomationMLBaseRole

图 10　AML 基础角色类库——XML 表格视图

```
<RoleClassLib Name="AutomationMLBaseRoleClassLib">
  <Description>Automation Markup Language base role class library</Description>
  <Version>2.10.0</Version>
  <RoleClass Name="AutomationMLBaseRole">
    <RoleClass Name="Group" RefBaseClassPath="AutomationMLBaseRole">
      <Attribute Name="AssociatedFacet" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/AssociatedFacet"/>
    </RoleClass>
    <RoleClass Name="Facet" RefBaseClassPath="AutomationMLBaseRole"/>
    <RoleClass Name="Resource" RefBaseClassPath="AutomationMLBaseRole"/>
    <RoleClass Name="Product" RefBaseClassPath="AutomationMLBaseRole"/>
    <RoleClass Name="Process" RefBaseClassPath="AutomationMLBaseRole"/>
    <RoleClass Name="Structure" RefBaseClassPath="AutomationMLBaseRole">
      <RoleClass Name="ProductStructure" RefBaseClassPath="Structure"/>
      <RoleClass Name="ProcessStructure" RefBaseClassPath="Structure"/>
      <RoleClass Name="ResourceStructure" RefBaseClassPath="Structure"/>
    </RoleClass>
    <RoleClass Name="ExternalData" RefBaseClassPath="AutomationMLBaseRole"/>
  </RoleClass>
</RoleClassLib>
```

图 11　AML 基础角色类库——XML 文本

6.4.2　角色类 AutomationMLBaseRole

表 13 规定了角色类“AutomationMLBaseRole”。

表 13　角色类 AutomationMLBaseRole

类名	AutomationMLBaseRole
描述	角色类“AutomationMLBaseRole”是一个基础的抽象角色类型，是所有标准或用户自定义角色类的基础类
父类	无
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
属性	无

6.4.3　角色类 Group

表 14 规定了角色类“Group”。

表 14　角色类 Group

类名	Group
描述	角色类“Group”是对象的一种角色类型。该类对象用于对属于特定工程视图的镜像对象集合进行分组。AML 的组对象应引用该角色类。8.4 给出具体细节和示例
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
属性	名称：AssociatedFacet RefAttributeType：AutomationMLBaseAttributeTypeLib/AssociatedFacet 语义：见 6.5.2

6.4.4　角色类 Facet

表 15 规定了角色类“Facet”。

表 15　角色类 Facet

类名	Facet
描述	角色类“Facet”是对象的一种角色类型。该类对象是一个 AML 对象接口或属性的子视图。AML 的面对象应引用该角色类。8.3 给出具体细节和示例
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Facet
属性	无

6.4.5　角色类 Resource

表 16 规定了角色类“Resource”。

表 16　角色类 Resource

类名	Resource
描述	角色类“Resource”是一种基本的抽象角色类型，以及所有 AML 资源角色的基础类。它描述了工厂、设备和其他生产资源。AML 资源对象应直接或间接地引用该角色。A.2.5 给出示例
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Resource
属性	无

此外，AML 资源对象在必要时应包括 CAEX 外部接口“PPRConnector”，用于与产品和过程创建关系（见 6.3.5）。

6.4.6　角色类 Product

表 17 规定了角色类“Product”。

表 17　角色类 Product

类名	Product
描述	角色类“Product”是一种基本的抽象角色类型，以及所有 AML 产品角色的基础类。它描述了产品、产品配件或在工厂中加工的产品相关材料。AML 产品对象应直接或间接地引用该角色。A.2.5 给出示例
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product
属性	无

此外，AML 产品对象在必要时应包括 CAEX 外部接口“PPRConnector”，用于与资源和过程创建关系（见 6.3.5）。

6.4.7　角色类 Process

表 18 规定了角色类“Process”。

表 18　角色类 Process

类名	Process
描述	角色类“Process”是一种基本的抽象角色类型，以及所有 AML 过程角色的基础类。它描述了生产相关的过程。AML 过程对象应直接或间接地引用该角色。A.2.5 给出示例
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
属性	无

6.4.8 角色类 Structure

表 19 规定了角色类“Structure”。

表 19 角色类 Structure

类名	Structure
描述	角色类“Strucuture”是用于表达工厂分层结构中的结构元素的一种基础抽象角色类型，这些元素包括文件夹、工作场所或生产线等。AML 结构对象应直接或间接地引用该角色
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Structure
属性	无

6.4.9 角色类 ProductStructure

表 20 规定了角色类“ProductStructure”。

表 20 角色类 ProductStructure

类名	ProductStructure
描述	角色类“ProductStructure”是一种面向产品的对象分层结构的抽象角色类型。AML 产品结构对象应直接或间接地引用该角色
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Strucuture
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Strucuture/ProductStructure
属性	无

6.4.10 角色类 ProcessStructure

表 21 规定了角色类“ProcessStructure”。

表 21 角色类 ProcessStructure

类名	ProcessStructure
描述	角色类“ProcessStructure” 是一种面向过程的对象分层结构的抽象角色类型。AML 过程结构对象应直接或间接地引用该角色
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Strucuture
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Strucuture/ProcessStructure
属性	无

6.4.11 角色类 ResourceStructure

表 22 规定了角色类“ResourceStructure”。

表 22　角色类 ResourceStructure

类名	ResourceStructure
描述	角色类“ResourceStructure”是一种面向资源的对象分层结构的抽象角色类型。AML 资源结构对象应直接或间接地引用该角色
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Strucuture
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Strucuture/ResourceStructure
属性	无

6.4.12　角色类 ExternalData

表 23 规定了角色类“ExternalData”。

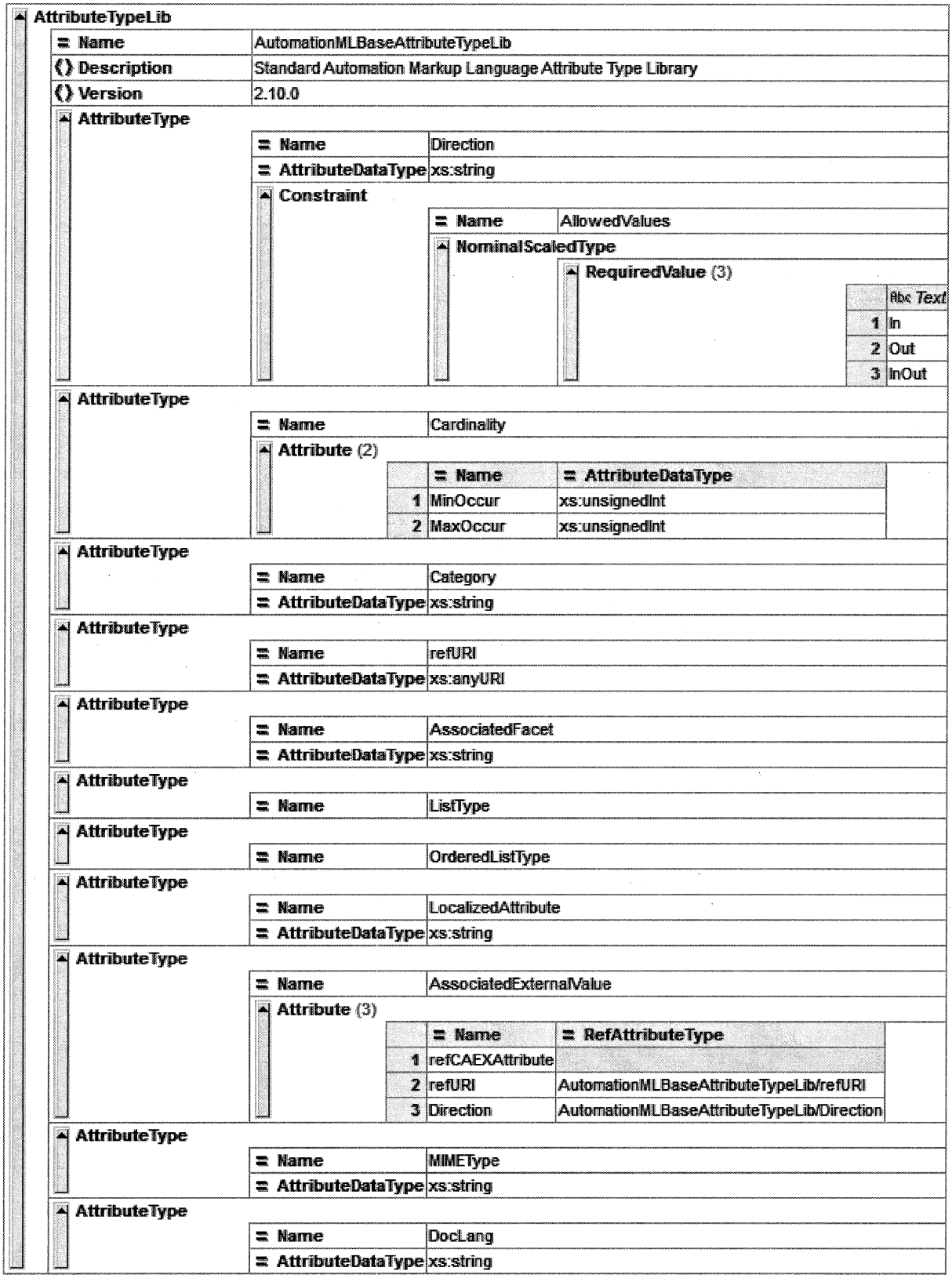
表 23　角色类 ExternalData

类名	ExternalData
描述	角色类“External”是一种文档类型的抽象角色类型和所有文档类型角色的基本类。它描述了不同的文档类型。AML 文档对象应直接或间接引用本角色。A.1.5 给出细节和示例
父类	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
元素引用路径	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/ExternalData
属性	无

6.5　AML 基本属性类型库

6.5.1　概述

6.5 定义了 6.3 和 6.4 定义的标准 AML 类的 AML 基本属性类型。图 12 和图 13 表示了作为 XML 表和 XML 内容的标准基本属性类型库。6.5.2 给出每个属性类型的细节。



- **AttributeTypeLib**
  - **Name**: AutomationMLBaseAttributeTypeLib
  - **Description**: Standard Automation Markup Language Attribute Type Library
  - **Version**: 2.10.0
  - **AttributeType**
    - **Name**: Direction
    - **AttributeDataType**: xs:string
    - **Constraint**
      - **Name**: AllowedValues
      - **NominalScaledType**
        - **RequiredValue** (3)

	Text
1	In
2	Out
3	InOut

  - **AttributeType**
    - **Name**: Cardinality
    - **Attribute** (2)

	Name	AttributeDataType
1	MinOccur	xs:unsignedInt
2	MaxOccur	xs:unsignedInt

  - **AttributeType**
    - **Name**: Category
    - **AttributeDataType**: xs:string
  - **AttributeType**
    - **Name**: refURI
    - **AttributeDataType**: xs:anyURI
  - **AttributeType**
    - **Name**: AssociatedFacet
    - **AttributeDataType**: xs:string
  - **AttributeType**
    - **Name**: ListType
  - **AttributeType**
    - **Name**: OrderedListType
  - **AttributeType**
    - **Name**: LocalizedAttribute
    - **AttributeDataType**: xs:string
  - **AttributeType**
    - **Name**: AssociatedExternalValue
    - **Attribute** (3)

	Name	RefAttributeType
1	refCAEXAttribute	
2	refURI	AutomationMLBaseAttributeTypeLib/refURI
3	Direction	AutomationMLBaseAttributeTypeLib/Direction

  - **AttributeType**
    - **Name**: MIMEType
    - **AttributeDataType**: xs:string
  - **AttributeType**
    - **Name**: DocLang
    - **AttributeDataType**: xs:string

图 12　AML 基本属性类型库

```
<AttributeTypeLib Name="AutomationMLBaseAttributeTypeLib">
  <Description>Standard Automation Markup Language Attribute Type Library</Description>
  <Version>2.10.0</Version>
  <AttributeType Name="Direction" AttributeDataType="xs:string">
    <Constraint Name="AllowedValues">
      <NominalScaledType>
        <RequiredValue>In</RequiredValue>
        <RequiredValue>Out</RequiredValue>
        <RequiredValue>InOut</RequiredValue>
      </NominalScaledType>
    </Constraint>
  </AttributeType>
  <AttributeType Name="Cardinality">
    <Attribute Name="MinOccur" AttributeDataType="xs:unsignedInt"/>
    <Attribute Name="MaxOccur" AttributeDataType="xs:unsignedInt"/>
  </AttributeType>
  <AttributeType Name="Category" AttributeDataType="xs:string"/>
  <AttributeType Name="refURI" AttributeDataType="xs:anyURI"/>
  <AttributeType Name="AssociatedFacet" AttributeDataType="xs:string"/>
  <AttributeType Name="ListType"/>
  <AttributeType Name="OrderedListType"/>
  <AttributeType Name="LocalizedAttribute" AttributeDataType="xs:string"/>
  <AttributeType Name="AssociatedExternalValue">
    <Attribute Name="refCAEXAttribute"/>
    <Attribute Name="refURI" RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI"/>
    <Attribute Name="Direction" RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction"/>
  </AttributeType>
  <AttributeType Name="MIMEType" AttributeDataType="xs:string"/>
  <AttributeType Name="DocLang" AttributeDataType="xs:string"/>
</AttributeTypeLib>
```

图 13　AML 基本属性类型库的 XML 代码

6.5.2　AutomationMLBaseAttributeTypeLib 的属性

表 24 规定了 AutomationMLBaseAttributeTypeLib 的属性类型。

表 24　AutomationMLBaseAttributeTypeLib 的属性类型

属性名称	语　义
Direction	本属性应用于描述 CAEX 接口的方向，例如，一个信号或者一个接口的方向。允许的值“In”、“out”或“InOut”。 CAEX 接口使用本属性遵守下列规范： ——带方向“In”的接口应仅与带方向“Out”或“InOut”的接口相连接。 ——带方向“Out”的接口应仅与带方向“In”或“InOut”的接口相连接。 本信息能用于例如证明连接的有效性。 示例： ——Direction=“Out”（例如一个插头） ——Direction=“In”（例如一个插座） ——Direction=“InOut” **注**：这些连接的有效性不在 IEC 62714 的范围内，但是一个工具功能性。 属性数据类型：xs:string 路径：automationMLBaseAttributeTypeLib/Direction

表 24（续）

属性名称	语　义
Cardinality	本属性属于 CAEX 外部接口，应使用于描述允许的连接接口的最大和最小数。 本属性 Cardinality 本身是一个复杂属性，不应有一个值。相应的子属性在表 25 中描述。 属性数据类型：因为本属性没有值，所以本属性没有属性数据类型。 路径：automationMLBaseAttributeTypeLib/Cardinality
Category	本属性属于 CAEX 外部接口，描述了接口的类别。本属性的值是用户自定义的。只有具有相同类别值的接口类才允许连接。本部分不预先定义类别值。 示例：Category="MaterialFlow" 属性数据类型：xs:string 路径：automationMLBaseAttributeTypeLib/Category
refURI	本属性应用于存储一个外部文件的路径。 属性数据类型：xs:anyURI 路径：automationMLBaseAttributeTypeLib/refURI
AssociatedFacet	本属性"AssociatedFacet"应用于定义相关面的名称。面概念在 8.3 中描述。 示例：AssociatedFacet="PLCFacet" 属性数据类型：xs:string 路径：automationMLBaseAttributeTypeLib/AssociatedFacet
ListType	本属性"ListType"应用于包含一个未排序属性清单的属性。本概念在 A.2.7 中描述。 属性数据类型：empty 路径：automationMLBaseAttributeTypeLib/ListType
OrderedListType	本属性"OrderedListType"应用于包含一个未排序属性清单的属性。本概念在 A.2.7 中描述。 属性数据类型：empty 路径：automationMLBaseAttributeTypeLib/ListType
LocalizedAttribute	本属性"LocalizedAttribute"应用于描述父类属性的替代语言的子属性。按照 RFC5646，本语言应作为属性名称使用。本概念在 A.2.6 中描述
AssociatedValue	本属性"AssociatedValue"包含允许互联 CAEX 属性到一个外部文件项的嵌套属性。本概念在 A.1.6 中描述。 属性"AssociatedValue"本身是一个复杂属性，不应有一个值。对应的子属性在表 26 中描述。 属性数据类型：因为本属性没有值，所以本属性没有属性数据类型。 父类：automationMLBaseAttributeTypeLib 路径：automationMLBaseAttributeTypeLib/AssociatedValue
MIMEType	MIMEType 描述了引用文件的 MIME 类型。 本属性应有符合 RFC 2046 的数值。 示例： MIMEType="application/pdf"——意思是本文件的文件类型是 pdf。 MIMEType="application/xml"——意思是本文件的文件类型是 xml。 属性数据类型：xs;string 父类：automationMLBaseAttributeTypeLib 路径：automationMLBaseAttributeTypeLib/MIMEType

表 24（续）

属性名称	语　义
DocLang	DocLang 描述了引用文件的语言。 本属性应有符合 RFC 5646 的数值。 示例：DocLang="fr-FR""en-US""de-DE"。意思是本文件是法语、美式英语或德式英语，在法国、美国或德国有效。 属性数据类型：xs:string 父类：automationMLBaseAttributeTypeLib 路径：automationMLBaseAttributeTypeLib/DocLang

表 25　属性"Cardinality"的子属性

属性	属性数据类型	描述	示例
MinOccur	xs:unsignedInt	MinOccur 的值描述连接的或者从相应接口类来的最小可能数值。 属性应有一个大于或等于 0 的值	MinOccur=1 意思是本端口宜与最少 1 个其他端口相连接
MaxOccur	xs:unsignedInt	MaxOccur 的值描述连接的或者从相应接口类来的最大可能数值。 属性应有一个大于或等于 MinOccur 或者 0 的值，取大者	MaxOccur=3 意思是本端口宜与最多 3 个其他端口相连接

表 26　属性"AssociatedValue"的子属性

属性	属性数据类型	描述	示例
refCAEXAttribute	〈GUID/attName〉	本属性与 CAEX 属性镜像，宜与一个外部文件的其他项互联。本属性没有值	refCAEXAttribute=GUID1/Temperature
refURI	AutomationMLBase-AttributeTypeLib/refURI	见表 24 中的 refURI	
Direction	AutoationMLBase-AttributeTypeLib/Direction	见表 24 中的 Direction	

7 用户自定义数据模型

7.1 概述

本章描述了用户自定义数据在 AML 中的建模。对一个特定的用户自定义数据进行建模是 AML 的一个核心概念。用户自定义数据指 IEC 62714(所有部分)中没有定义的 CAEX 属性、接口类和角色类。AML 顶层数据格式 CAEX 提供了用户自定义数据建模的机制。

用户自定义数据的交换可能需要用户特定协议或功能。这些内容不属于 IEC 62714(所有部分)的范围。5.4 中描述的源工程工具的特定元信息支持这些功能。

AML 允许通过角色、特性集概念或标准 CAEX 映射定义用户自定义数据和标准数据间的关系。这些概念简化了用户自定义类和属性的自动化解释。

7.2 用户自定义属性

所有在 IEC 62714(所有部分)中定义的属性被称为 AML 属性。所有未在 IEC 62714(所有部分)中定义的属性被称为用户自定义属性。AML 属性和用户自定义属性通过同样的方式以 CAEX 属性存储。

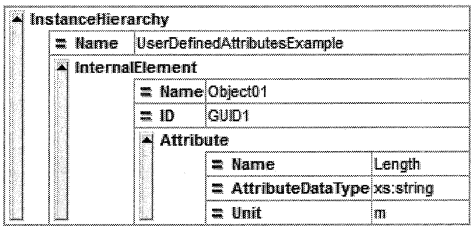
以下规定适用于用户自定义属性:

——CAEX 属性应根据 IEC 62424:2016 中 A.2.4 关于 CAEX 属性的定义存储在 AML 中。

——用户自定义属性在需要时应采用同样的单位制。本部分未定义单位制。

——建议使用 ISO 80000-1 定义的国际单位制。建议使用 IEC 60027 定义的信息技术单位。

图 14 给出了一个带有用户自定义属性“Length”的用户自定义对象“Object01”的示例。




```
<InstanceHierarchy Name="UserDefinedAttributesExample">
  <InternalElement Name="Object01" ID="GUID1">
    <Attribute Name="Length" AttributeDataType="xs:string" Unit="m"/>
  </InternalElement>
</InstanceHierarchy>
```

图 14 用户自定义属性示例——表格视图、XML 文本

7.3 用户自定义接口类 InterfaceClass

以下规定适用于用户自定义接口类:

——用户自定义接口类应根据 IEC 62424:2016 中 A.2.5 关于 CAEX 接口类的定义进行存储;

——属性类库推荐给特定模型用户、特定公司、特定区域、特定国家等,或者关于属性语义的规范性国际标准,相关的现在或未来的属性标准。这是储存认同的属性语义和保持语言和命名习惯独立性的基础;

示例:图 15 和图 16 解释了用户自定义的属性类的定义。

——属性类库“MyAttributeTypeLibrary”包含属性类“Height”“Width”和“Length”;

——内部元素“Station”有 3 个属性引用了用户自定义的属性类型。

注:属性类库允许不同于属性类的对象属性命名。这可以使语言和命名习惯保持独立性。

InstanceHierarchy

= Name	MyHierarchy

InternalElement

= Name	Station
= ID	GUID1

InternalElement

= Name	Robot_1
= ID	GUID2

Attribute (3)

	= Name	= RefAttributeType
1	Höhe	MyAttributeTypeLibrary/Height
2	Breite	MyAttributeTypeLibrary/Width
3	Länge	MyAttributeTypeLibrary/Length

AttributeTypeLib

= Name	MyAttributeTypeLibrary

AttributeType (3)

	= Name	= Attr...	= Unit
1	Height	xs:double	m
2	Width	xs:double	m
3	Length	xs:double	m

图 15　用户自定义属性类型示例——表格视图

```
<InstanceHierarchy Name="MyHierarchy">
  <InternalElement Name="Station" ID="GUID1">
    <InternalElement Name="Robot_1" ID="GUID2">
      <Attribute Name="Höhe" RefAttributeType="MyAttributeTypeLibrary/Height"/>
      <Attribute Name="Breite" RefAttributeType="MyAttributeTypeLibrary/Width"/>
      <Attribute Name="Länge" RefAttributeType="MyAttributeTypeLibrary/Length"/>
    </InternalElement>
  </InternalElement>
</InstanceHierarchy>

<AttributeTypeLib Name="MyAttributeTypeLibrary">
  <AttributeType Name="Height" AttributeDataType="xs:double" Unit="m"/>
  <AttributeType Name="Width" AttributeDataType="xs:double" Unit="m"/>
  <AttributeType Name="Length" AttributeDataType="xs:double" Unit="m"/>
</AttributeTypeLib>
```

图 16　用户自定义属性类型示例——XML 文本

7.4　用于自定义接口类 InterfaceClasses

所有在 IEC 62714(所有部分)中定义的接口类是标准 AML 接口类。所有未在 IEC 62714(所有部分)中定义的接口类是用户自定义接口类。

以下规定适用于用户自定义接口类：

——所有用户自定义的接口类应根据 IEC 62424:2016 中 A.2.6 关于 CAEX 接口类的定义进行存储；

注：用户自定义接口类和标准 AML 接口类以同样方式作为 CAEX 接口类存储。

——为了保证用户自定义接口类算数上的可解释性，它们应派生自 AML 接口类。

图 17 和图 18 给出一个派生自 AML 接口类“SignalInterface”的用户自定义类“MyDigitalInput”的示例。接口类“MyDigitalInput”和标准 AML 接口类“SignalInterface”之间的继承关系允许用户自定义类作为数字输入接口自动识别。适当设置用户自定义属性。在本示例中，用户自定义属性不在本部分范围内。

示例：本示例使用一个简化的路径符号以增加可读性。在实际应用中，提供完整路径。

InterfaceClassLib				
= Name	UserDefinedClassLib			
InterfaceClass				
= Name	MyDigitalInput			
= RefBaseClassPath	AutomationMLInterfaceClassLib/.../SignalInterface			
{} Version	1.0			
Attribute (3)				
		= Name	= AttributeDataType	{} Value
	1	Type	xs:string	Digital
	2	Direction	xs:string	In
	3	Enabled	xs:boolean	true

图 17　用户自定义接口类库中用户自定义接口类的示例——表格视图

```
<InterfaceClassLib Name="UserDefinedClassLib">
  <InterfaceClass Name="MyDigitalInput" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface">
    <Version>1.0</Version>
    <Attribute Name="Type" AttributeDataType="xs:string">
      <Value>Digital</Value>
    </Attribute>
    <Attribute Name="Direction" AttributeDataType="xs:string">
      <Value>In</Value>
    </Attribute>
    <Attribute Name="Enabled" AttributeDataType="xs:boolean">
      <Value>true</Value>
    </Attribute>
  </InterfaceClass>
```

图 18　用户自定义接口类库中用户自定义接口类的示例——XML 文本

7.5　用户自定义角色类 RoleClass

所有在 IEC 62714(所有部分)中定义的角色类被称为 AML 角色类。所有未在 IEC 62714(所有部分)中定义的角色类被称为用户自定义角色类。

以下规定适用于用户自定义角色类：

——CAEX 角色类应根据 IEC 62424:2016 中 A.2.7 关于 CAEX 角色类的定义进行存储；

注 1：AML 角色类和用户自定义角色类应以同样方式作为 CAEX 角色类存储。

——为了保证用户自定义角色类语义上的可解释性，它们应派生自 AML 角色类。

注 2：这是为了保证该类语义在算法上的可解释性。

图 19 和图 20 给出了一个派生自 AML 角色类资源“Resource”的用户自定义类围栏“Fence”。“Fence”和“Resource”之间的继承关系允许将该用户自定义类解释为一种资源。

RoleClassLib			
= Name	UserDefinedRoleClassLib		
{} Version	1.0		
RoleClass (1)			
		= Name	= RefBaseClassPath
	1	Fence	AutomationMLRoleClassLib/AutomationMLBaseRole/Resource

图 19　用户自定义角色类库中用户自定义角色类的示例——表格视图

```
<RoleClassLib Name="UserDefinedRoleClassLib">
  <Version>1.0</Version>
  <RoleClass Name="Fence" RefBaseClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource"/>
</RoleClassLib>
```

图 20　用户自定义角色类库中用户自定义角色类的示例——XML 文本

7.6 用户自定义系统单元类 SystemUnitClass

IEC 62714 未定义系统单元类，因此所有的系统单元类都是用户自定义的。

以下规定适用于用户自定义系统单元类：

——用户自定义系统单元类应根据 IEC 62424:2016 中 A.2.3 的 CAEX 系统单元类定义存储在 AML 中。

——用户自定义系统单元类宜对应一个 AML 角色类，并在应用时使用 AML 属性。因此，用户自定义系统单元类也许对应于一个标准 AML 角色类，一个用户自定义角色类或者没有角色类。

推荐将用户自定义系统单元类对应一个角色类，以便于自动解释。

示例：图 21 和图 22 通过两种不同示例解释了一个用户自定义系统单元类定义。

——系统单元类“Robot1234”是一个支持 AML 标准角色类库中角色“Resource”的用户自定义类。因此，该类可以被自动解释为“Resource”。

——系统单元类“SpecialRobot1234”是一个派生自“Robot1234”的新用户自定义类。因此该类也是“资源”。

SystemUnitClassLib		
= Name	UserDefinedSystemUnitClassLib	
SystemUnitClass		
= Name	Robot1234	
= ID	GUID1	
() Version	1.0	
SupportedRoleClass		
= RefRoleClassPath	AutomationMLRoleClassLib/AutomationMLBaseRole/Resource	
SystemUnitClass		
= Name	SpecialRobot1234	
= RefBaseClassPath	UserDefinedSystemUnitClassLib/Robot1234	

图 21 不同用户自定义系统单元类的示例

```
<SystemUnitClassLib Name="UserDefinedSystemUnitClassLib">
  <SystemUnitClass Name="Robot1234" ID="GUID1">
    <Version>1.0</Version>
    <SupportedRoleClass RefRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource"/>
  </SystemUnitClass>
  <SystemUnitClass Name="SpecialRobot1234" RefBaseClassPath="UserDefinedSystemUnitClassLib/Robot1234"/>
</SystemUnitClassLib>
```

图 22 不同用户自定义系统单元类的示例——XML 文本

7.7 用户自定义实例分层结构 InstanceHierarchy

CAEX 的实例分层结构用来存储独立或项目相关的工程信息。它们构成了 AML 顶层格式的中心，并包含所有独立的数据对象，包括特性、接口、关系和引用。

以下规定适用于用户自定义实例分层结构：

——本部分不限制分层结构的深度；

——本部分不限制分层结构的建筑规则；

——本部分不定义分层结构的命名规则；

——同一实例分层结构中的每个 AML 对象应直接或间接地对应一个 AML 角色类，以规定其抽象类型。

图 23 描述了一个项目分层结构的示例。其中包含了一条带有工位“S001”的生产线“L001”。工位中包含两个机器人“R0010_D”和“R0020_D”、一条传送带“RF010”和一个 PLC“P001”。

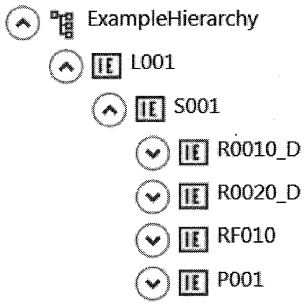


图 23　用户自定义实例分层结构示例——对象树

图 24 给出了该结构的 XML 文本。根据 IEC 62424:2016 中 A.2.9 的要求，每一个对象都与一个角色类相联系。

```
<InstanceHierarchy Name="ExampleHierarchy">
  <InternalElement Name="L001" ID="GUID1">
    <InternalElement Name="S001" ID="GUID2">
      <InternalElement Name="R0010_D" ID="GUID3" RefBaseSystemUnitPath="ABBRobotLibrary/ABB_Robot_1234">
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource/Mechatronic/Robot"/>
      </InternalElement>
      <InternalElement Name="R0020_D" ID="GUID4" RefBaseSystemUnitPath="ABBRobotLibrary/ABB_Robot_1234">
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource/Mechatronic/Robot"/>
      </InternalElement>
      <InternalElement Name="RF010" ID="GUID5">
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource/Mechatronic/Transport/Rollerbed"/>
      </InternalElement>
      <InternalElement Name="P001" ID="GUID6">
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource/ControlEquipment/ControlHardware/PLC"/>
      </InternalElement>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource/Structure/Station"/>
    </InternalElement>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource/Structure/Line"/>
  </InternalElement>
</InstanceHierarchy>
```

图 24　用户自定义实例分层结构的 XML 文本

8　扩展 AML 概念

8.1　概述

本部分为特定工程方面的建模定义了扩展概念。A.2 给出了资料性概述和示例。

8.2　AML 端口对象 Port

一个 AML 端口是一个组合了一定数量接口的 AML 对象。A.2.2 给出了端口概念的资料性概述和示例。

以下规定适用于 AML 端口：

——AML 端口应描述为一个与 6.3.4 所定义的角色类端口“Port”相联系的 CAEX 内部元素；

——端口对象的所有 CAEX 外部接口应直接或间接地派生自 6.3 定义的 AML 接口类。

8.3　AML 面对象 Facet

面是一个 AML 对象，它提供了父 AML 对象属性或接口的子视图。这个概念用于存储不同的配置设定，如 HMI 或 PLC 相关的数据，还允许多个工程控制步骤的自动化。因此，本部分定义了 AML

角色类面“Facet”(见 6.4.4)。A.2.3 给出了有关面概念的资料性概述和示例。

以下规定适用于 AML 面:

——AML 面对象应描述为一个与 6.4.4 所定义的角色类面“Facet”相联系的 CAEX 内部元素。

——AML 面对象也许在实例分层结构或者系统单元类的任意位置建模。

——面通过唯一 ID 进行识别。他们的名称只是显示名称。

——内部元素或者系统单元类可有任意数量的面对象。

——面对象应只包含镜像属性或接口。

——面可以有任意数量的面属性或面接口。

——每个面属性应按照 IEC 62424:2016 的 A.2.8.7 与父对象已存在的属性相镜像。父对象内的子属性和其他附属属性的镜像是有可能的。

——不准许存在不是父对象一部分的面属性。

——每个面接口应按照 IEC 62424:2016 的 A.2.8.7 与父对象已存在的接口相镜像。父对象内的内部接口的镜像是可能的。

——不准许存在不是父对象一部分的面接口。

——面不应包含新的子对象、属性或接口。

——面对象不应是嵌套的。

——面对象不应改变现存的属性或接口。

8.4 AML 组对象 Group

AML 组概念允许将结构信息从实例信息中分离。A.2.4 给出了有关组概念的资料性概述和示例。

以下规定适用于 AML 组对象:

——AML 组对象应描述为一个与 6.4.3 所定义的角色类组“Group”相联系的 CAEX 内部元素。

——一个 AML 组可以建模在实例分层结构或系统单元类的任意位置,应是一个实例分层结构、内部元素或者系统单元类的子对象。

——组通过唯一 ID 进行识别。他们的名称只是显示名称。

——一个内部元素或者系统单元类可有任意数量的组对象。

——一个 AML 组对象应只包含镜像对象和/或其他的组对象。

注 1:因此,组对象是能嵌套的。

——AML 组不应被用于描述工厂分层结构。

——一个 AML 组对象可将额外信息存储为属性、接口或端口,用于描述组的特定信息。

注 2:这些额外的属性、接口或端口与组内包含的镜像对象的属性、接口或端口不等。

——不准许为镜像对象增加额外信息,或改变镜像的现有属性、接口或端口。

——属性“AssociatedFacet”在使用时应有一个提供某现有面有效名称的值。

8.5 AML 顶层数据至不同文档的分离

根据 IEC 62424:2016 中 A.2.12 的要求,CAEX 明确支持将工程数据分布至不同的文件中,并提供了利用 CAEX 元素“ExternalReference”和 CAEX 中对应的别名概念来引用外部 CAEX 文件的方法。

8.6 国际化,AML 多语言表达

AML 多语言表达概念允许用不同语言存储文本表达作为一个属性结构。一个包括示例的关于多语言表达概念的概述见 A.2.6。

以下规定适用于 AML 多语言表达:

——多语言文本属性应作为 CAEX 属性进行建模,属性值应表示默认值。如果没有要求指定语言

或者没有对要求的语言进行建模，可使用默认值。

——属性的每个语言表达应作为嵌套属性进行建模。每个子属性应引用属性类型“LocalizedAttribute”。

——每个子属性的名称应是符合 RFC5646 的表达语言。例如“en-US”“de-DE”或者“fr-FR”等。语言属性值应是相关语言的文本。

附加的关于多语言表达的规范性要求见 6.5.2。

8.7 AML 对象版本信息

关于存储独立 AML 对象(对象实例)的版本和修订信息，应使用 IEC 62424:2016 中 A.2.2.2 要求的标准版本和修订域。

关于存储 AML 相关的版本信息和与 AML 库相关的版本信息，见 5.3。

关于存储工具特定的元信息，见 5.4。

8.8 结构化属性清单或队列

在许多应用中，需要对清单进行存储。例如支持的频率的清单。AML 允许对清单属性或者队列进行建模和存储。下列规定适用于对清单的建模：

——清单是同类项的序列。例如，所有项应具有相同的数据类型。

——清单作为清单根节点的 CAEX 属性进行建模。

——如果是无序清单，清单属性应引用属性类型“ListType”。

——如果是有序清单，清单属性应引用属性类型“OrderedListType”。

——属性数据类型、值、默认值和清单属性单元应是空的。

——清单项应作为清单属性的 CAEX 子属性进行建模。

——所有子属性应具有相同数据类型。

——如果是有序清单，子属性的名称应表示为一个整型数字“1”“2”等。为了较好的可读性，前导零可以添加。例如“0001”。整型数字应表示每个清单项的顺序索引。

注：术语整型不表示一个数据类型。

——如果是无序列表，子属性的名称在同级别中应是唯一的。

——子属性可以安排清单属性。允许队列建模。在这个示例中，所有子属性应引用属性类型“ListType”或者“OrderedListType”。

A.2.7 给出关于清单、队列的更多信息，通过示例解释了如何建模。

8.9 AML 容器

为了传递包含多个 AML 和其他文件的 AML 项目，本文件支持 OPC 作为容器格式。按照 OPC 定义，下列规定适用：

——AML 容器应作为按照 ISO/IEC 29500-2 提供数据压缩的 OPC 档案存储。

——AML 容器应是完全独立的或者与环境相关的。完全独立的 AML 容器应包括只是本地互联的所有相关文件。环境相关的 AML 容器可以包含连接 URIs 到 AML 容器外部的公开文件。

——AML 容器文件应有文件扩展名“.amlx”。

——按照 OPC 协议，本文件定义下列关系类型。

- AML 根文件。

  AML 根文件是一个作为指向 AML 容器入口的 AML 文件。

  关系类型：http://schemas.automationml.org/container/relationship/RootDocument

  Mime 类型：“model/vnd.automationml+xml”

- AML 库文件：
  AML 库文件是一个包含任意个或者多个角色类库、接口类库、系统单元类库和属性类库等类型元素的 AML 库。与 AML 根文件类似，AML 库文件是一个作为指向 AML 容器入口的 AML 文件。AML 容器可仅包含库文件。
  关系类型：http://schemas.automationml.org/container/relationship/Library
  Mime 类型："model/vnd.automationml+xml"
- COLLADA 文件：
  关系类型：http://schemas.automationml.org/container/relationship/Collada
  Mime 类型："model/vnd.collada+xml"
- PLCopenXML 文件：
  关系类型：http://schemas.automationml.org/container/relationship/PLCopenXML
  Mime 类型："model/vnd.plcopen+xml"
- 任意内容：
  关系类型：http://schemas.automationml.org/container/relationship/AnyContent
  Mime 类型："application/x-any"或者符合 RFC2046 的用户自定义。
- CAEX 架构：
  关系类型：http://schemas.automationml.org/container/relationship/CAEXSchema
  Mime 类型："text/xml"
- PLCopenXML 架构：
  关系类型：http://schemas.automationml.org/container/relationship/PLCopenXMLSchema
  Mime 类型："text/xml"
- COLLADA 架构：
  关系类型：http://schemas.automationml.org/container/relationship/ColladaSchema
  Mime 类型："text/xml"

附 录 A
（资料性附录）
自动化标记语言总体介绍

A.1 自动化标记语言通用概念

A.1.1 自动化标记语言架构

自动化标记语言(AML)是一种基于可扩展标记语言(XML)架构的数据格式，用于实现独立于供应商的工厂工程信息交换。AML 旨在建立不同领域已有异构工具之间的联系，例如机械工程、电气设计、过程工程、过程控制工程、HMI 开发、PLC 编程以及机器人编程等。

AML 遵循面向对象的方法存储工程信息，并且支持以封装不同方面的数据对象来对实际工厂组件建模。一个对象可包含其他子对象，也可隶属于一个更大的组合或聚合。对象可描述一个信号、一个 PLC、一个容器、一个控制阀、一个机器人、一个不同细节层面的制造单元，或者一个完整的生产现场、生产线或工厂。工厂自动化中典型对象包含的信息包括拓扑、几何学、运动学以及逻辑，而逻辑涵盖了序列、行为和控制。

AML 整合现有用来在不同领域内存储和交换工程信号的工业数据格式。这些数据格式按照各自的规范独立实施，并不属于 AML 的分支。

AML 的核心是顶层数据格式 CAEX。CAEX 利用 AML 来互连不同的数据格式。因此 AML 有其固有的分布式文档架构。

图 A.1 描述了 AML 的基本架构以及拓扑、几何学、运动学和逻辑信息分布。

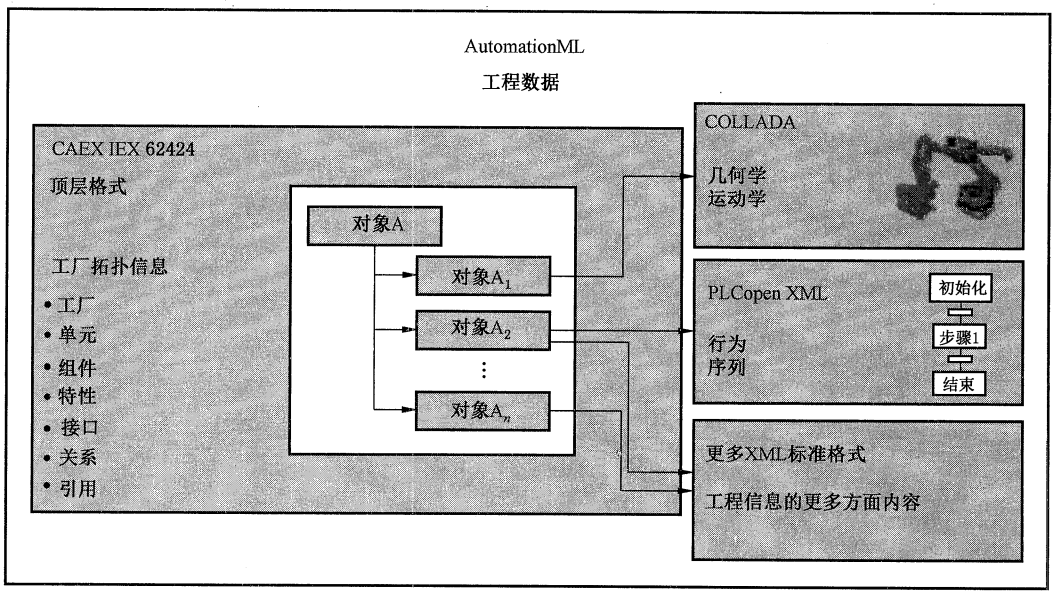


图 A.1 AML 通用架构

分布式文档概念的主要优势在于使用已验证和已有的数据格式，通过数据在不同文档之间的分布降低处理大批量信息的难度，同时简化可单独存储、交换和访问的 AML 库文件的使用。最后，不同细节层面如几何变量可单独存储。AML 主要定义了引用数据格式和工程对象之间的联系。

结合一些简单例子，A.1.1 对 AML 可存储及交换的信息做了概述：

——**工厂拓扑信息**:工厂拓扑将工厂描述为一个独立工厂对象的分层结构,该工厂对象由独立数据对象代表。这个对象结构在一定的细节层面上建模(如机器人、夹持器,但不是轴或者连接点);这些对象包含了特性以及在其层次结构中与其他对象的关系。工厂拓扑作为顶层数据结构,是按照符合 IEC 62424:2016 第 7 章、附录 A 及附录 C 要求的 CAEX 数据格式进行存储的。作为 IEC 62424 的延伸,AML 定义了从 CAEX 对象到储存在 CAEX 之外的外部文档中信息的引用。A.1.2 给出了如何用 AML 对工程拓扑信息建模的示例。

——**几何学与运动学信息**:一个单独工厂对象的几何学信息由它的几何表示构成。运动学信息描述了三维实体的物理连接和对象之间的依赖关系。几何学信息和运动学信息均采用 COLLADA 文件格式存储。此外,COLLADA 文档包括几何学信息与运动学信息耦合的定义。为满足今后的互相链接的需要,COLLADA 接口可在顶层格式内发布为 CAEX 外部接口。根据不同对象的 COLLADA 几何学信息,可自动得到一个完整的场景。这些文件可引用自 CAEX,并可通过 CAEX 链接机制互联。A.1.3 给出了一个简短的示例。详细信息在 IEC 62714-3 规定。

——**逻辑信息**:逻辑信息描述了动作序列以及包括 I/O 连接和逻辑变量的对象行为。序列被描述和存储在外部 PLCopen XML 文档中。变量或信号可发布为 CAEX 外部接口。这些文档既可从 CAEX 外部被引用,也可在 CAEX 内部互相链接。A.1.4 对主要概念做了简短介绍。详细信息在 IEC 62714-4 规定。

——**引用与关系信息**:AML 对引用和关系做了区分。引用描述了从 CAEX 对象到外部存储信息的链接。关系描述了 CAEX 对象之间的联系。此外,相同的方法被用来存储外部文档中信息之间的关联。为此,有必要通过 CAEX 外部接口在 CAEX 工厂拓扑中发布相关链接涉及的对象。引用 COLLADA 和 PLCopen XML 文档的详细信息在 IEC 62714-3 和 IEC 62714-4 中提供。A.1.7 给出了 AML 中引用和关系建模的资料性概述。5.5 和 5.6 给出了标准化规定。

——**引用其他数据格式**:IEC 62714 在将来可通过增加更多的部分得到扩展。这些部分将规定更多采用 AML 引用机制的数据格式的整合。

工程信息的交换还需要特定的扩展概念。A.2 解释了这些概念,第 8 章给出了标准化规定。

注:本部分中,路径有时以简化形式给出,如以“AutomationMLInterfaceClassLib/…/Port”来代替“AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Port”,这有助于增加本部分可读性。在实际 XML 文档中,所有路径依据本部分规定存储。

A.1.2 工厂拓扑信息的建模

在 AML 中,通常以封装工程信息不同方面的数据对象来对实际工厂组件进行建模。为此,有必要对数据对象进行结构化。结构化数据对象的一个既定方法是一个描述工厂拓扑(见 3.1.19)的对象分层结构。

AML 采用 IEC 62424:2016 中 A.2.8.2 所规定的顶层数据格式 CAEX 的概念来存储分层的工厂结构。图 A.2 给出了一条包括数个不同层次对象的生产线的工厂拓扑示例。

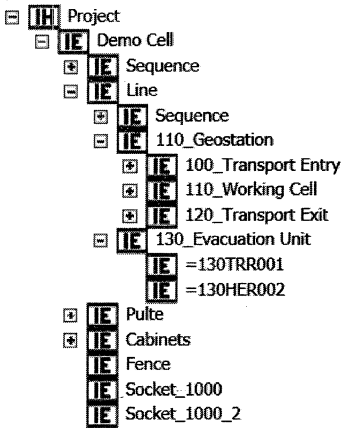


图 A.2 AML 工厂拓扑

多层次、交叉结构和复杂对象网络可采用标准 CAEX 概念建模。然而，面向对象的方法高效的关键是包含预定义和已证实解决方案的库的运用。因此，CAEX 为接口、角色、系统单元和实例分层结构这些对 AML 很重要的元素提供了许多不同的数据类型：

——**接口类和接口类库**：接口用于定义 AML 对象之间的关系。6.3 规定了 AML 标准接口类库，其包含一组用于描述通用自动化系统的抽象接口类。这些类包括语法和语义定义，并用于用户自定义对象接口的规范。7.3 规定了用户自定义接口类的建模。

——**角色类和角色类库**：角色类用于定义元素的抽象特征，并对用户自定义系统单元类或内部元素进行自动的语义解释。6.4 规定了 AML 基础角色类库，其包含一组用于描述通用自动化系统的抽象角色类。7.5 规定了用户自定义角色类的建模。更多的角色库在 IEC 62714-2 定义。

——**系统单元和系统单元类库**：系统单元类库用于对供应商特定的组件和内部结构进行类建模。7.6 规定了系统单元类定义的架构规则。AML 没有预先定义特定系统单元类库或系统单元类。

——**属性类型和属性类型库**：属性类型库用于对供应商特定的属性类型或标准属性类型进行建模。由属性类型派生的属性即使使用不同的名字，仍会继承其语义。这是语言独立性的基础。属性类型库能被用来作为语义库的基础并用于供应商特定属性的自动解释。

——**实例和实例层次**：实例分层结构存储了实际项目的拓扑数据，是 AML 的核心。实例分层结构由 AML 对象实例组成。7.7 规定了如何利用类型的实例分层结构存储工程信息。

工厂拓扑建模的一个重要方面是对象识别。不同工程工具使用不同概念识别对象，例如唯一的名称、唯一的标识符或唯一的路径。有些工具允许在其生命周期内改变标识符，而有些不准许。在单一的工具中，这种方式正常工作，但这不可能适用于在不同工具之间交换对象的情况。因此，5.5 规定了强制性对象识别的概念，它实现了利用个体对象识别的概念在不同工程工具之间进行数据交换。

A.1.3 几何学信息与运动学信息的引用

几何学信息与运动学信息按照 COLLADA 数据格式存储于单独的文档中。因此，几何学信息与运动学信息的建模也分为两个部分。一方面，本部分中对应对象按照 CAEX 的建模并不包含任何几何学信息或运动学信息，另一方面，应该提供一个包含几何学信息和运动学信息的 COLLADA 文档。于是，CAEX 对象存储了对 COLLADA 文档的引用，这些内容在 IEC 62714-3 中描述。

图 A.3 给出了一个包含对象“110RB_200”的 AML 文档示例，它引用了一个包含对应几何学信息和运动学信息的外部 COLLADA 文档。

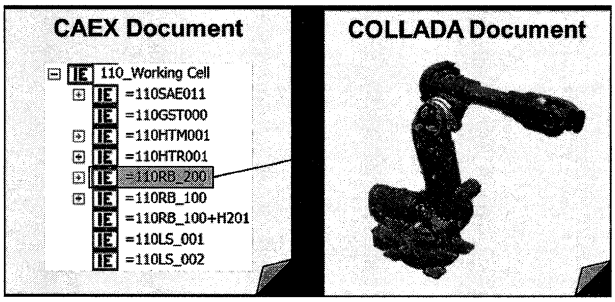


图 A.3　CAEX 文档对 COLLADA 文档的引用

引用通过派生自标准 AML 接口类“COLLADAInterface”的 CAEX 接口实现。5.7 和 6.3.7 对其进行了规定。IEC 62714-3 给出其细节。

A.1.4　逻辑信息的引用

逻辑信息按照 PLCopen XML 数据格式存储于单独的文档中。因此，逻辑信息的建模分为两个部分。一方面，本部分中对应对象按照 CAEX 的建模且不包含任何逻辑信息，另一方面，应该提供一个包含逻辑信息的 PLCopen XML 文档，相关内容会在 IEC 62714-4 中描述。于是，CAEX 对象存储了对 PLCopen XML 文档的引用。图 A.4 给出了一个包含对象“110_Working Cell”的 AML 文档示例，它引用了一个包含对应逻辑信息的外部 PLCopen XML 文档。

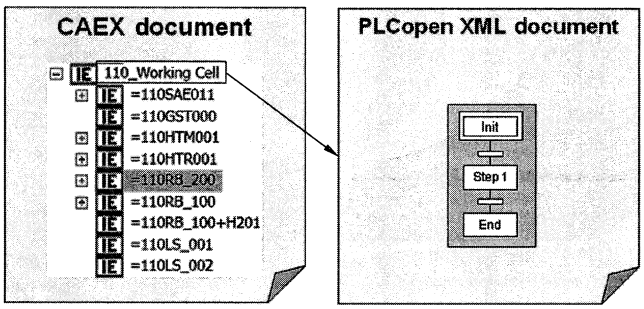


图 A.4　CAEX 文档对 PLCopen XML 文档的引用

A.1.5　IEC 62714(所有部分)范围外的文档引用

本部分对不在 IEC 62714(所有部分)范围之内的外部文档引用机制进行定义[例如：文档、pdf 文件、Excel 工作表、IEC 62714(所有部分)之外的 XML 文件等]。图 A.5 和图 A.6 演示了如何建模。

在本示例中，对象“TemperatureSensor”需要引用一个外部文件“example.xml”。为对该引用进行建模，内部元素“TemperatureSensor”有一个代表外部文档“ExternalData”类型的嵌套内部元素“ParameterDocument”。它的属性“DocLang”是建模文档的语言。内部元素“ParameterDocument”包含一个外部数据引用类型的 CAEX 外部接口。它的属性 refURI 和 MIMEType 定义了文档的 URI 和文档类型，在本案例中 MIMEType 是“application/xml”。5.6.5 给出了其标准化规定。

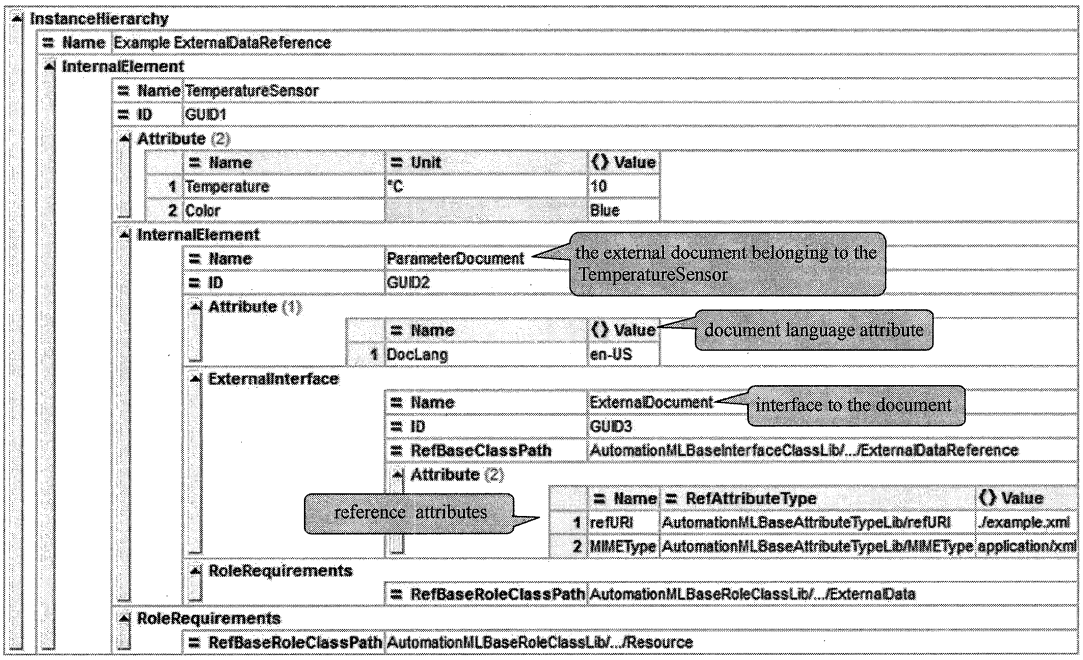


图 A.5　引用外部文档示例

```
<InstanceHierarchy Name="Example ExternalDataReference">
  <InternalElement Name="TemperatureSensor" ID="GUID1">
    <Attribute Name="Temperature" Unit="°C">
      <Value>10</Value>
    </Attribute>
    <Attribute Name="Color">
      <Value>Blue</Value>
    </Attribute>
    <InternalElement Name="ParameterDocument" ID="GUID2">
      <Attribute Name="DocLang">
        <Value>en-US</Value>
      </Attribute>
      <ExternalInterface Name="ExternalDocument" ID="GUID3" RefBaseClassPath="AutomationMLBaseInterfaceClassLib/.../ExternalDataReference">
        <Attribute Name="refURI" RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI">
          <Value>./example.xml</Value>
        </Attribute>
        <Attribute Name="MIMEType" RefAttributeType="AutomationMLBaseAttributeTypeLib/MIMEType">
          <Value>application/xml</Value>
        </Attribute>
      </ExternalInterface>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../ExternalData"/>
    </InternalElement>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.6　引用外部文档示例的 XML 文本

A.1.6　CAEX 属性和外部文档属性的链接

图 A.7 和图 A.8 演示了在一个 CAEX 属性和一个外部文档项之间如何进行链接建模。本示例是 A.1.5 中示例的扩展：属性“Temperature”需要被引用为外部文档“example.xml”中的一项。

除了 A.1.5/图 A.5 的示例外，外部接口“ExternaDocument”建模了类型“AssociatedExternalValue”的进一步属性。该属性包含 3 个嵌套属性：

——属性“refCAEXAttribute”是对所需 CAEX 属性的镜像。该属性没有值。

——属性“refURI”引用了外部属性。它引用了与外部接口的 refURI 属性相同的外部文档“example.xml”。

——属性“Direction”提供了引用的用法说明，在本示例中，外部文档值是主导，CAEX 是外部值的消费者。

5.6.6 给出了其标准化规定。

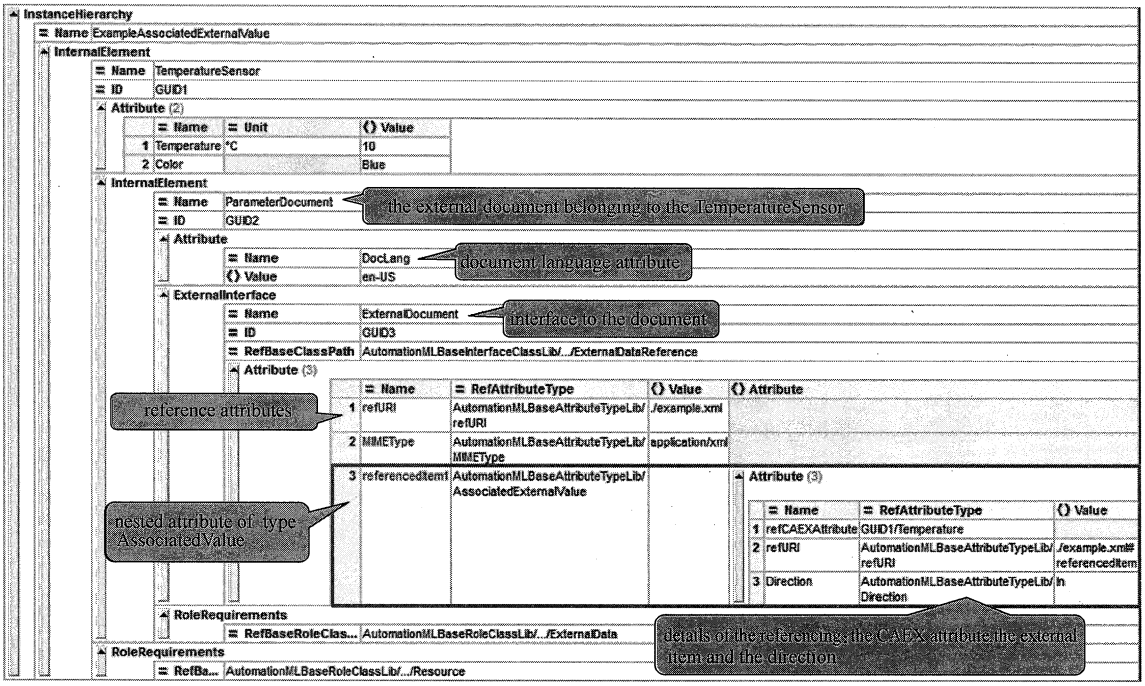


图 A.7　CAEX 属性引用到外部文档项的示例

```
<InstanceHierarchy Name="ExampleAssociatedExternalValue">
  <InternalElement Name="TemperatureSensor" ID="GUID1">
    <Attribute Name="Temperature" Unit="°C">
      <Value>10</Value>
    </Attribute>
    <Attribute Name="Color">
      <Value>Blue</Value>
    </Attribute>
    <InternalElement Name="ParameterDocument" ID="GUID2">
      <Attribute Name="DocLang">
        <Value>en-US</Value>
      </Attribute>
      <ExternalInterface Name="ExternalDocument" ID="GUID3" RefBaseClassPath="AutomationMLBaseInterfaceClassLib/.../ExternalDataReference">
        <Attribute Name="refURI" RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI">
          <Value>./example.xml</Value>
        </Attribute>
        <Attribute Name="MIMEType" RefAttributeType="AutomationMLBaseAttributeTypeLib/MIMEType">
          <Value>application/xml</Value>
        </Attribute>
        <Attribute Name="referencedItem1" RefAttributeType="AutomationMLBaseAttributeTypeLib/AssociatedExternalValue">
          <Value/>
          <Attribute Name="refCAEXAttribute" RefAttributeType="GUID1/Temperature"/>
          <Attribute Name="refURI" RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI">
            <Value>./example.xml#referencedItem</Value>
          </Attribute>
          <Attribute Name="Direction" RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction">
            <Value>In</Value>
          </Attribute>
        </Attribute>
      </ExternalInterface>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../ExternalData"/>
    </InternalElement>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.8　CAEX 属性引用到外部文档项示例的 XML 文本

A.1.7 关系的建模

对象建模使得有必要定义一种将这些对象互相联系起来的机制。将这些对象与外部存储数据进行链接需要另外的机制。

一种关系表达了两个或更多对象之间的联系。这种依赖性是出于自然属性的，包括物理依赖性和逻辑依赖性。根据 IEC 62424:2016 中 A.2.8，CAEX 支持以下关系：

——父子关系：

- AML 对象之间的父子关系；
- AML 类之间的父子关系。

——继承关系：

- 系统单元类之间的继承关系；
- 角色类之间的继承关系；
- 接口类之间的继承关系；
- 属性类型 4 之间的继承关系。

——类-实例关系(见 5.5.2)：

- 一个系统单元类和它的一个实例之间的关系；
- 一个角色类和它的一个实例之间的关系；
- 一个接口类和它的一个实例之间的关系；
- 一个属性类型和一个属性之间的关系。

——实例-实例关系(见 5.5.3)：

- AML 对象之间的关系；
- 已发布的外部存储数据之间的关系。

图 A.9 给出了 AML 支持的上述关系类型的示例。

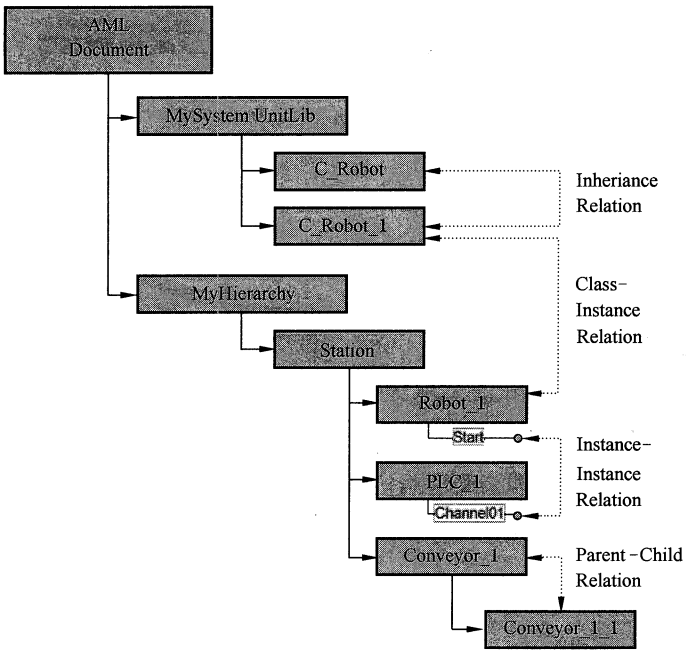


图 A.9 AML 中的关系

图 A.10 以表格视图给出了上述示例的 AML 模型。需要注意是，这里使用占位符“/…/”以减少路径信息，增加可读性。图 A.11 给出了 AML 库相应的 XML 文本。

图 A.12 给出了实例分层结构的 XML 文本。

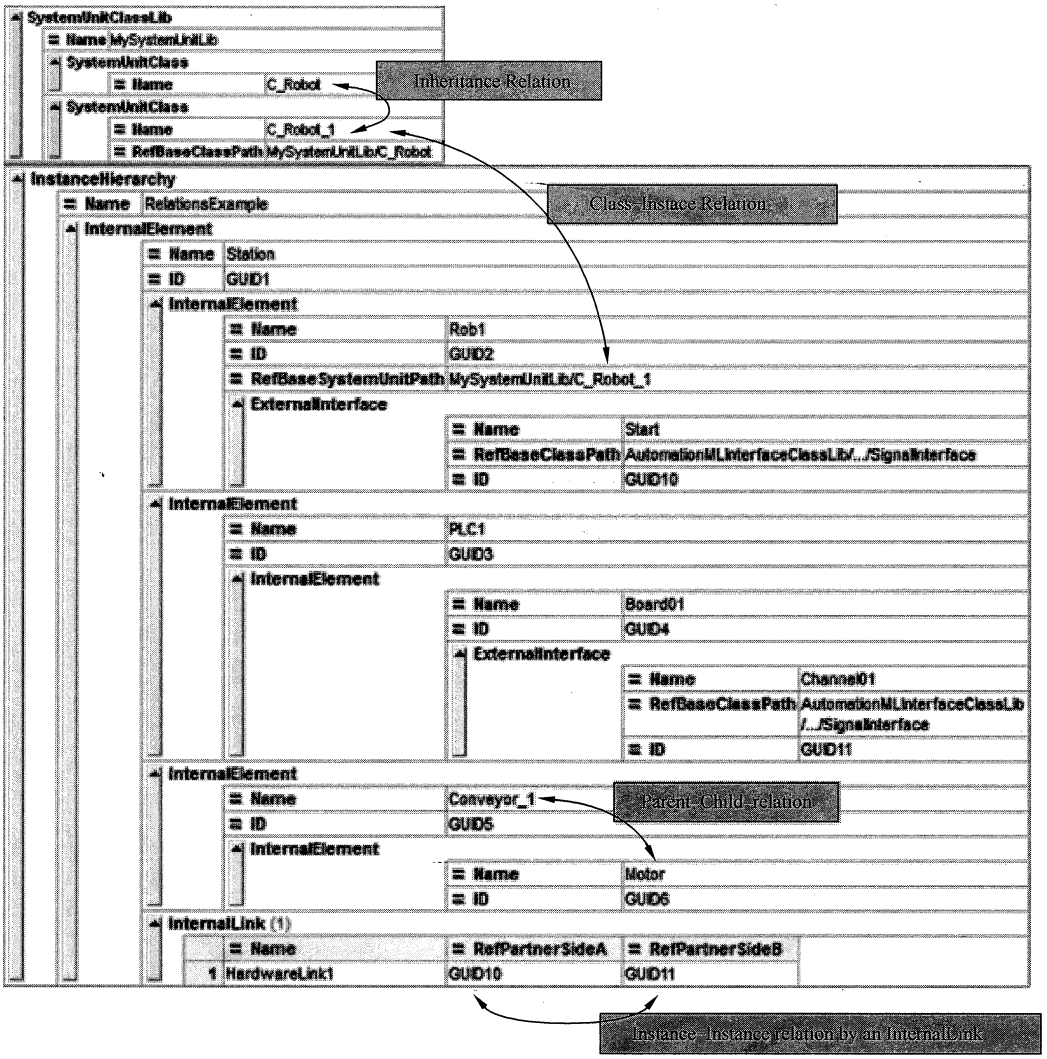


图 A.10 关系示例的 XML 描述

```
<SystemUnitClassLib Name="MySystemUnitLib">
    <SystemUnitClass Name="C_Robot"/>
    <SystemUnitClass Name="C_Robot_1" RefBaseClassPath="C_Robot"/>
</SystemUnitClassLib>
```

图 A.11 关系示例中系统单元类库的 XML 文本

```
<InstanceHierarchy Name="RelationsExample">
  <InternalElement Name="Station" ID="GUID1">
    <InternalElement Name="Rob1" ID="GUID2" RefBaseSystemUnitPath="MySystemUnitLib/C_Robot_1">
      <ExternalInterface Name="Start" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface" ID="GUID10"/>
    </InternalElement>
    <InternalElement Name="PLC1" ID="GUID3">
      <InternalElement Name="Board01" ID="GUID4">
        <ExternalInterface Name="Channel01" RefBaseClassPath="AutomationMLInterfaceClassLib/.../SignalInterface" ID="GUID11"/>
      </InternalElement>
    </InternalElement>
    <InternalElement Name="Conveyor_1" ID="GUID5">
      <InternalElement Name="Motor" ID="GUID6"/>
    </InternalElement>
    <InternalLink Name="HardwareLink1" RefPartnerSideA="GUID10" RefPartnerSideB="GUID11"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.12　关系示例中实例分层结构的 XML 文本

A.2　扩展 AML 概念与示例

A.2.1　概述

AML 对一些特定工程方面的建模定义了扩展概念，如 AML 端口概念、AML 面概念、AML 组概念。表 A.1 给出了这些概念的概述。

表 A.1　AML 主要扩展概念概述

概念	描　述
AML 端口	AML 端口概念允许对复杂接口进行高层次描述。AML 端口由一组同属的 AML 接口集组成，能被理解为类似插头或插座
AML 面	AML 面允许存储 AML 对象属性和接口的子集，能被看作工程数据的视图
AML 组	AML 组允许存储 AML 对象子集的不同视图，能用来根据不同工程工具对工厂树中的对象进行筛选
过程-产品-资源	过程-产品-资源概念允许基于以过程为中心、以产品为中心或者以资源为中心的视图，以及它们之间的关系对工程数据进行高层次的结构化
AML 多语言表达	AML 多语言表达允许存储不同的语言文字
属性列表和属性数组	AML 允许属性列表或属性数组建模

A.2.2　AML 端口 Port 概念

A.2.2.1　概念描述

AML 端口是一个组合了一定数量接口的 AML 对象（见图 A.13）。一个端口对象属于一个父 AML 对象，并描述父对象的复杂接口。端口可在一个更高的抽象层次上彼此互联，而不是在单一接口之间链接。AML 端口在描述插头、插座或其他可以直接互联的接口组时很有用。因此，AML 定义了 AML 角色类端口“Port”（见 6.3.4）。8.2 给出了其标准化规定。

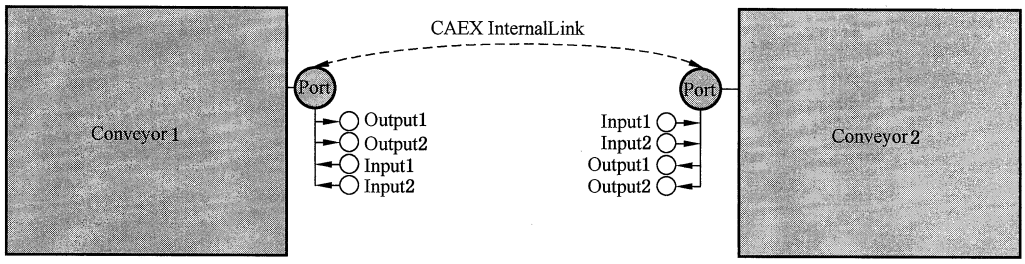


图 A.13 端口概念

A.2.2.2 示例

图 A.14 给出了 AML 端口概念的示例。对象工位“Station”由传送带“Conveyor1”和“Conveyor2”两个子对象组成。两个子对象均有一个包含子接口的复杂接口。这是通过由 AML 标准接口类“Port”派生每一个接口的方式进行建模的，并且包含嵌套接口的连接。这个标准接口可通过 CAEX 内部链接进行链接。这个关系意味着两个端口彼此互联。所有子接口的内部链接没有在细节方面进行描述，只有接口被互相链接。此外，AML 允许在子接口之间建模和存储每一个单独的链接。

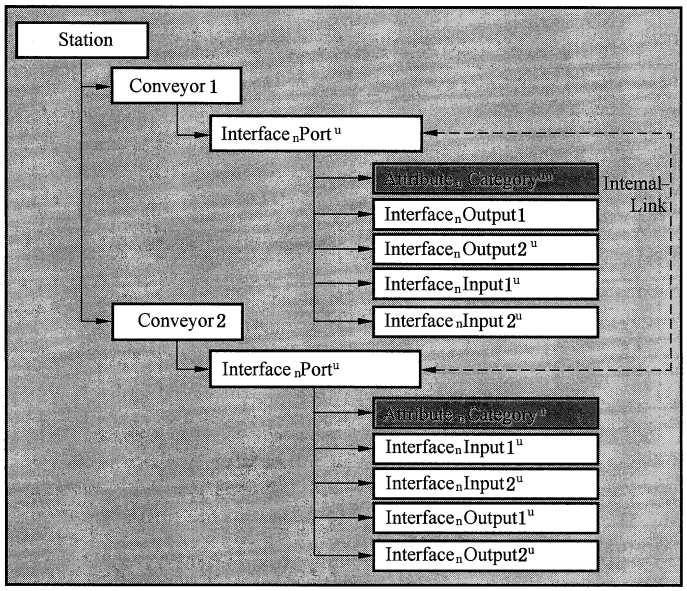


图 A.14 描述 AML 端口概念的示例

图 A.15 和图 A.16 描述了图 A.14 中的示例系统的 AML 实现。

InstanceHierarchy

Name	PortExample AML 2.10

InternalElement

Name	Station
ID	GUID1

InternalElement

Name	Conveyor1
ID	GUID2

ExternalInterface

Name	Port
ID	GUID3
RefBaseClassPath	AutomationMLInterfaceClassLib/.../Port

Attribute

Name	Direction
AttributeDataType	xs:string
RefAttributeType	AutomationMLBaseAttributeTypeLib/Direction
{} Value	Out

Attribute

Name	Category
AttributeDataType	xs:string
RefAttributeType	AutomationMLBaseAttributeTypeLib/Category
{} Value	MaterialFlow

ExternalInterface (4)

	Name	RefBaseClassPath	ID
1	Output1	MyInterfaceClassLib/SignalInterface	GUID5
2	Output2	MyInterfaceClassLib/SignalInterface	GUID6
3	Input1	MyInterfaceClassLib/SignalInterface	GUID7
4	Input2	MyInterfaceClassLib/SignalInterface	GUID8

RoleRequirements

RefBaseRoleClassPath	AutomationMLRoleClassLib/AutomationMLBaseRole/Resource

InternalElement

Name	Conveyor2
ID	GUID4

ExternalInterface

Name	Port
ID	GUID10
RefBaseClassPath	AutomationMLInterfaceClassLib/.../Port

Attribute

Name	Direction
AttributeDataType	xs:string
RefAttributeType	AutomationMLBaseAttributeTypeLib/Direction
{} Value	In

Attribute

Name	Category
AttributeDataType	xs:string
RefAttributeType	AutomationMLBaseAttributeTypeLib/Category
{} Value	MaterialFlow

ExternalInterface (4)

	Name	RefBaseClassPath	ID
1	Input1	MyInterfaceClassLib/SignalInterface	GUID12
2	Input2	MyInterfaceClassLib/SignalInterface	GUID13
3	Output1	MyInterfaceClassLib/SignalInterface	GUID14
4	Output2	MyInterfaceClassLib/SignalInterface	GUID15

RoleRequirements

RefBaseRoleClassPath	AutomationMLRoleClassLib/AutomationMLBaseRole/Resource

InternalLink (1)

	Name	RefPartnerSideA	RefPartnerSideB
1	L1	GUID3	GUID10

图 A.15　AML 端口概念的 XML 描述

```
<InstanceHierarchy Name="PortExample AML 2.10">
  <InternalElement Name="Station" ID="GUID1">
    <InternalElement Name="Conveyor1" ID="GUID2">
      <ExternalInterface Name="Port" ID="GUID3" RefBaseClassPath="AutomationMLInterfaceClassLib/.../Port">
        <Attribute Name="Direction" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction">
          <Value>Out</Value>
        </Attribute>
        <Attribute Name="Category" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Category">
          <Value>MaterialFlow</Value>
        </Attribute>
        <ExternalInterface Name="Output1" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID5"/>
        <ExternalInterface Name="Output2" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID6"/>
        <ExternalInterface Name="Input1" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID7"/>
        <ExternalInterface Name="Input2" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID8"/>
      </ExternalInterface>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource"/>
    </InternalElement>
    <InternalElement Name="Conveyor2" ID="GUID4">
      <ExternalInterface Name="Port" ID="GUID10" RefBaseClassPath="AutomationMLInterfaceClassLib/.../Port">
        <Attribute Name="Direction" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction">
          <Value>In</Value>
        </Attribute>
        <Attribute Name="Category" AttributeDataType="xs:string" RefAttributeType="AutomationMLBaseAttributeTypeLib/Category">
          <Value>MaterialFlow</Value>
        </Attribute>
        <ExternalInterface Name="Input1" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID12"/>
        <ExternalInterface Name="Input2" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID13"/>
        <ExternalInterface Name="Output1" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID14"/>
        <ExternalInterface Name="Output2" RefBaseClassPath="MyInterfaceClassLib/SignaleInterface" ID="GUID15"/>
      </ExternalInterface>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Resource"/>
    </InternalElement>
    <InternalLink Name="L1" RefPartnerSideA="GUID3" RefPartnerSideB="GUID10"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.16　AML 端口概念的 XML 文本描述

A.2.2.3　将端口作为用户自定义端口建模

端口能通过派生接口类来进行扩展。图 A.17 中的示例描述了一个派生自接口类“myPortInterface”的 XML 描述。这个类继承了标准类“Port”的所有属性并且添加了一个新的属性“Enable”。

InterfaceClass		
= Name	UserDefinedPort	
= RefBaseClassPath	AutomationMLInterfaceClassLib/.../Port	
Attribute (1)		
	= Name	= AttributeDataType
1	Enabled	xs:boolean

```
<InterfaceClass Name="UserDefinedPort" RefBaseClassPath="AutomationMLInterfaceClassLib/.../Port">
  <Attribute Name="Enabled" AttributeDataType="xs:boolean"/>
</InterfaceClass>
```

图 A.17　用户自定义 AML 端口类“UserDefinedPort”定义——表格视图、XML 文本

A.2.3　AML 面 Facet 概念

A.2.3.1　概念描述

面是提供父 AML 对象属性或接口子视图的一个 AML 对象。这个概念用于存储不同的配置设定，如 HMI 或 PLC 相关数据，还允许数个工程控制步骤的自动化。因此，AML 定义了 AML 角色类面“Facet”（见 6.4.4）。8.3 给出了标准化规定。

所描述的属性和接口分组是与特定工程方面相关的，它可存储对应的工程解决方案或模板信息。这些属性名称或值的语法或语义不在本部分的范围内，而是由一个了解对应信息语法和语义的外部工程工具进行解释。因此，这些算法只需要必要的面信息就可执行自动化工程任务。例如，一个对象的属性包括了 PLC 代码模板的名称，其接口描述了该模板输入或输出。那么，一个了解这些属性和接口语义的 PLC 代码生成算法可根据这些信息生成一个 PLC 代码。对于 HMI 模板也可能是同样的情况。上述的外部算法或对应属性或接口的语义不在 IEC 62714 范围内。结合 AML 组概念，工程步骤的自

动化是可以实现的。

A.2.3.2 示例

图 A.18 通过示例解释了 AML 面的概念：对象传送带“Conveyor1”由属性“A”“B”和接口“X”“Y”组成。预设的面对象“PLCFacet”包括属性“A”和接口“X”，而预设的面对象“HMIFacet”包括属性“A”“B”和接口“Y”。因此，两个面提供了与不同工程任务相关的特定工程信息筛选后的视图。

使用案例：属性“A”可以是供应商特定的 PLC 代码模板的名称，它描述了对象“Convoyer1”的功能。接口“X”可以是该代码模板所需输入信号的名称。属性“B”可以是传送带特定 HMI 模板的名称，接口“Y”可以是该 HMI 使用的信号。有了这些信息，一个 PLC 或 HMI 生成器就有可能自动生成解决方案。A.2.4.4 中对此进行了说明。

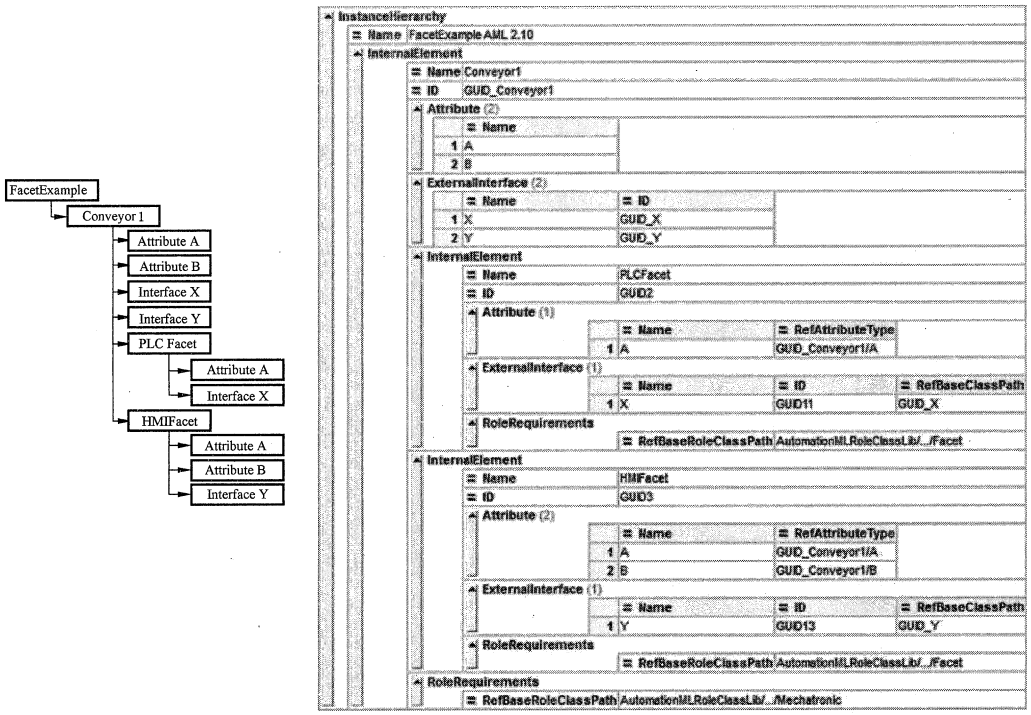


图 A.18 AML 面示例

```
<InstanceHierarchy Name="FacetExample AML 2.10">
  <InternalElement Name="Conveyor1" ID="GUID_Conveyor1">
    <Attribute Name="A"/>
    <Attribute Name="B"/>
    <ExternalInterface Name="X" ID="GUID_X"/>
    <ExternalInterface Name="Y" ID="GUID_Y"/>
    <InternalElement Name="PLCFacet" ID="GUID2">
      <Attribute Name="A" RefAttributeType="GUID_Conveyor1/A"/>
      <ExternalInterface Name="X" ID="GUID11" RefBaseClassPath="GUID_X"/>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Facet"/>
    </InternalElement>
    <InternalElement Name="HMIFacet" ID="GUID3">
      <Attribute Name="A" RefAttributeType="GUID_Conveyor1/A"/>
      <Attribute Name="B" RefAttributeType="GUID_Conveyor1/B"/>
      <ExternalInterface Name="Y" ID="GUID13" RefBaseClassPath="GUID_Y"/>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Facet"/>
    </InternalElement>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Mechatronic"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.19 AML 面示例的 XML 文本

A.2.4 AML 组 Group 概念

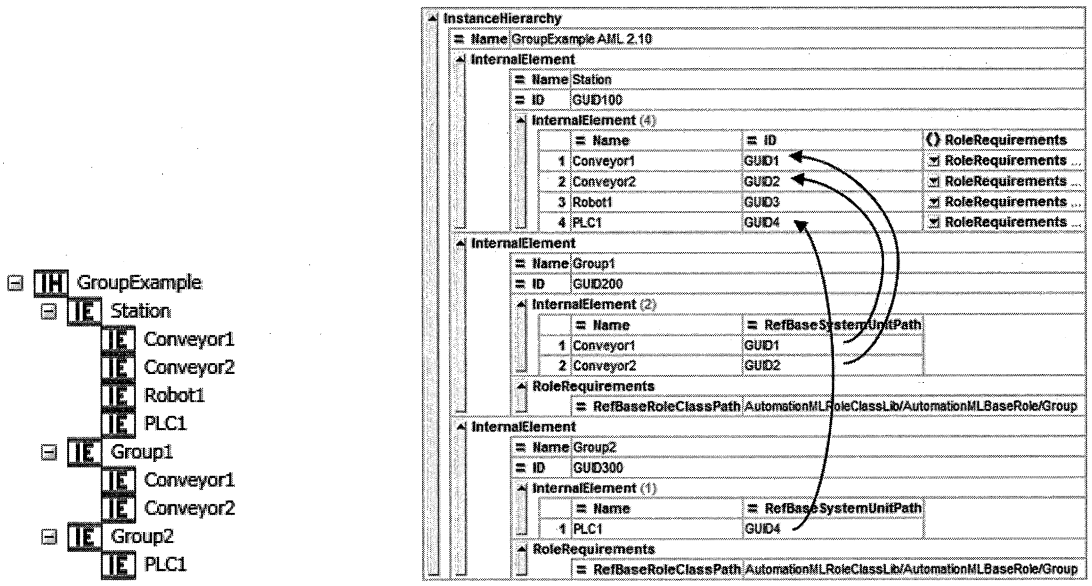
A.2.4.1 概念描述

AML 组概念允许将结构信息从实例信息中分离出来。由于异构工具中不同的工程工具对同一个数据可需要使用不同的视图，所以要将这些视图分开存储。使用 AML 组概念可以实现这一点，并且允许在不同的层次中对相同的对象进行结构化。

通过定义组属性“AssociatedFacet”，一个组可以与拥有唯一名称的面相联系。这允许外部工程算法自动识别相关对象和它们对应的面，用于从中提取工程信息。为此，AML 定义了 AML 角色类“Group”(见 6.4.3)。8.4 给出了标准化规定。

A.2.4.2 示例

图 A.20a)通过结构工位“Station”描述了组概念，该结构包含了对象传送带 1“Conveyor1”、传送带 2“Conveyor2”、机器人“Robot1”和 PLC“PLC1”。此外，对象组 1“Group1”、组 2“Group2”在不同层次描述了同一个对象：“Group1”只给出了传送带的结构视图，“Group2”只给出了 PLC 相关对象。依据 IEC 62424:2016 的 A.2.8.7，CAEX 提供了上述交叉结构的存储方式。图 A.20b)给出 AML 对该示例的运用，同时图 A.21 给出了对应的 XML 文本。A.2.4.3 描述了面概念和组概念的组合使用。



a) 对象树　　b) 表格视图

图 A.20 AML 组示例——对象树、表格视图

```
<InstanceHierarchy Name="GroupExample AML 2.10">
  <InternalElement Name="Station" ID="GUID100">
    <InternalElement Name="Conveyor1" ID="GUID1">
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Transport/Rollerbed"/>
    </InternalElement>
    <InternalElement Name="Conveyor2" ID="GUID2">
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Rollerbed"/>
    </InternalElement>
    <InternalElement Name="Robot1" ID="GUID3">
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Robot"/>
    </InternalElement>
    <InternalElement Name="PLC1" ID="GUID4">
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../PLC"/>
    </InternalElement>
  </InternalElement>
  <InternalElement Name="Group1" ID="GUID200">
    <InternalElement Name="Conveyor1" RefBaseSystemUnitPath="GUID1"/>
    <InternalElement Name="Conveyor2" RefBaseSystemUnitPath="GUID2"/>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Group"/>
  </InternalElement>
  <InternalElement Name="Group2" ID="GUID300">
    <InternalElement Name="PLC1" RefBaseSystemUnitPath="GUID4"/>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/AutomationMLBaseRole/Group"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.21　AML 组示例的 XML 文本

A.2.4.3　组 Group 与面 Facet 概念的组合

图 A.22 给出了组概念与面概念组合的示例。给出的实例分层结构描述了一个 AML 对象工位“Station”，它由 AML 对象传送带 1“Conveyor1”和传送带 2“Conveyor2”组成。每条传送带都有两个属性和两个接口。

AML 对象组“Group”代表了嵌套的组 1“Group1”和组 2“Group2”。两者都指向传送带对象，却有不同的面与之联系。

使用案例：一个控制代码生成算法可遍历实例分层结构从而识别出所有与“PLCFacet”相联系的组，然后执行代码生成来评估所引用的对象。

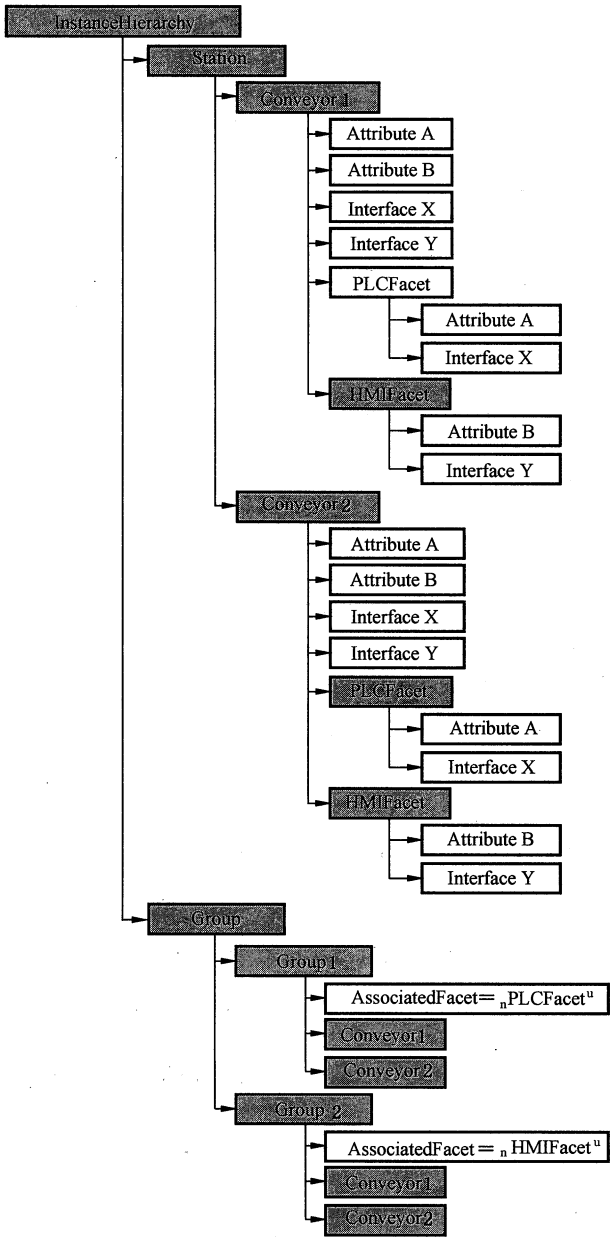


图 A.22　面概念与组概念的组合

图 A.23 给出了图 A.22 中示例对应的 XML 文本。

```
<InstanceHierarchy Name="FacetGroupCombination AML 2.10">
  <InternalElement Name="Station" ID="GUID0">
    <InternalElement Name="Conveyor1" ID="GUID1">
      <Attribute Name="A"/>
      <Attribute Name="B"/>
      <ExternalInterface Name="X" ID="GUID_X"/>
      <ExternalInterface Name="Y" ID="GUID_Y"/>
      <InternalElement Name="PLCFacet" ID="GUID2">
        <Attribute Name="A" RefAttributeType="GUID1/A"/>
        <ExternalInterface Name="X" ID="GUID11" RefBaseClassPath="GUID_X"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Facet"/>
      </InternalElement>
      <InternalElement Name="HMIFacet" ID="GUID3">
        <Attribute Name="B" RefAttributeType="GUID1/B"/>
        <ExternalInterface Name="Y" ID="GUID12" RefBaseClassPath="GUID_Y"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Facet"/>
      </InternalElement>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Transport/Rollerbed"/>
    </InternalElement>
    <InternalElement Name="Conveyor2" ID="GUID4">
      <Attribute Name="A"/>
      <Attribute Name="B"/>
      <ExternalInterface Name="X" ID="GUID_X2"/>
      <ExternalInterface Name="Y" ID="GUID_Y2"/>
      <InternalElement Name="PLCFacet" ID="GUID5">
        <Attribute Name="A" RefAttributeType="GUID4/A"/>
        <ExternalInterface Name="X" ID="GUID13" RefBaseClassPath="GUID_X2"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Facet"/>
      </InternalElement>
      <InternalElement Name="HMIFacet" ID="GUID6">
        <Attribute Name="B" RefAttributeType="GUID4/B"/>
        <ExternalInterface Name="Y" ID="GUID14" RefBaseClassPath="GUID_Y2"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Facet"/>
      </InternalElement>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Transport/Rollerbed"/>
    </InternalElement>
    <InternalElement Name="Group" ID="GUID7">
      <InternalElement Name="Group1" ID="GUID8">
        <Attribute Name="AssociatedFacet" RefAttributeType="AutomationMLBaseAttributeTypeLib/AssociatedFacet">
          <Value>HMIFacet</Value>
        </Attribute>
        <InternalElement Name="Conveyor1" RefBaseSystemUnitPath="GUID1"/>
        <InternalElement Name="Conveyor2" RefBaseSystemUnitPath="GUID4"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Group"/>
      </InternalElement>
      <InternalElement Name="Group2" ID="GUID9">
        <Attribute Name="AssociatedFacet" RefAttributeType="AutomationMLBaseAttributeTypeLib/AssociatedFacet">
          <Value>PLCFacet</Value>
        </Attribute>
        <InternalElement Name="Conveyor1" RefBaseSystemUnitPath="GUID1"/>
        <InternalElement Name="Conveyor2" RefBaseSystemUnitPath="GUID4"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLRoleClassLib/.../Group"/>
      </InternalElement>
    </InternalElement>
  </InternalElement>
</InstanceHierarchy>
```

图 A.23　面-组组合示例的 XML 文本视图

A.2.4.4　运用组 Group 和面 Facet 概念的 HMI 自动生成

基于上述示例，假设传送带的属性“B”代表一个变量“Y”可视化的 HMI 模板。图 A.24 展示了传送带的通用 HMI 模板。

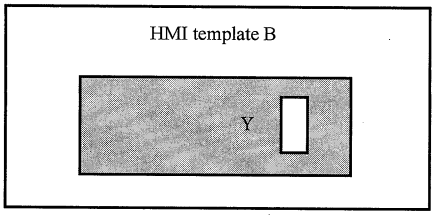


图 A.24　传送带过程变量“Y”可视化的通用 HMI 模板“B”

基于具体的传送带实例，一个工程算法能够识别 AML 对象组 2“Group2”与 HMI 面相联系。这里识别出实例传送带 1“Conveyor1”和传送带 2“Conveyor2”是 HMI 的一部分。该算法获取两条传送带各自与 HMI 相关的信息，识别对应的 HMI 模板，并且连接需要可视化的正确信号。图 A.25 给出了所生成的 HMI。

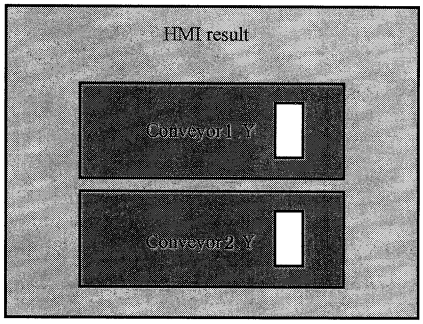


图 A.25　将两条传送带单独过程变量可视化的 HMI 生成结果“B”

A.2.5　过程 Process-产品 Product-资源 Resource 概念

A.2.5.1　概念描述

在复杂工厂工程数据结构化的过程中，将数据分为资源、过程和产品三种类型的方法已经在实践中得到了验证。这个概念可以被应用在许多领域，比如数字工厂工具，或者根据 IEC 62264 应用在 MES 层面：

——在以资源为中心的视图中，资源构成了模型的中心组件：它们执行过程并且处理产品。在 AML 中，一个资源是生产过程中的一个实体，它包括工厂、机器人、机器、它们的状态、设备、可能的讯息等等。因此，资源可能是一个生产系统的硬件组件，也可能是像 SCADA 系统一样的软件系统。在 AML 的范畴里，资源通常在一个构成工厂拓扑的工厂分层结构中建模。

——在以产品为中心的视图中，成品是关注的焦点。它决定了材料或中间产品宜采用哪种过程，以及宜使用哪些设备。这适用于连续控制、离散控制或批量控制领域。AML 中的产品描述了一个生产出的物件，它可能被分层创建。重要的是，产品不应是最终的产品。产品还包括测试结果、产品数据和对应文档。

——在以过程为中心的视图中，过程构成了模型的中心元素。AML 中的一个过程代表了一个包含子过程的生产过程。过程参数、过程链和过程规划构成了过程的一部分。在技术方面，过程改变了产品。这符合其在 AML 中的使用，比如最终产品是由许多不同的子产品生产出的，化学处理会改变物质。然而，过程与资源也存在一定关系，反之亦然。

在每一个案例中，资源、产品和过程之间都互相链接（见图 A.26）。将资源“传送带”赋予过程“运输”是合理的，过程“下压”能创造出“立方体废料”，而过程“焊接”能将两块“金属”“焊”在一起。

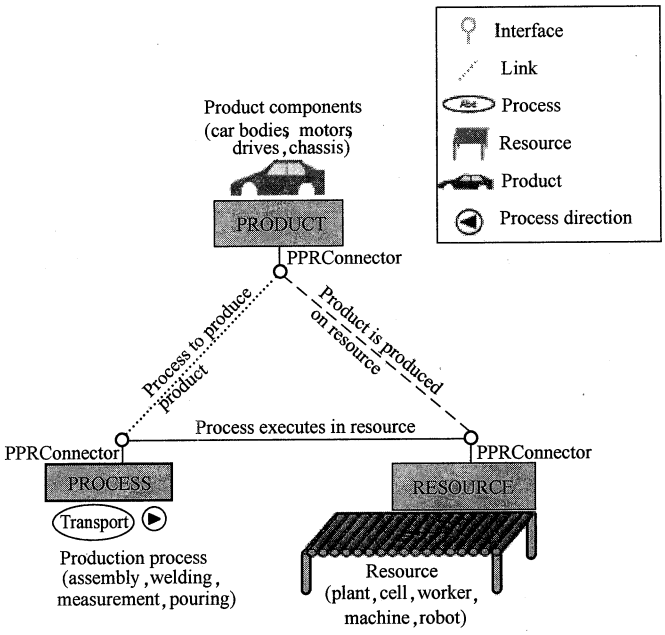


图 A.26 产品-过程-资源概念的基本元素

在这些元素之间建立链接需要接口。为此，AML 定义了标准接口类 PPR 连接器(见图 A.27)。6.3.5 给出了关于 PPR 连接器的标准化规范。通过该接口，元素之间能通过标准 CAEX 内部链接建立链接(见 5.5.3)。这样，资源能被链接到它们可以处理的产品上。

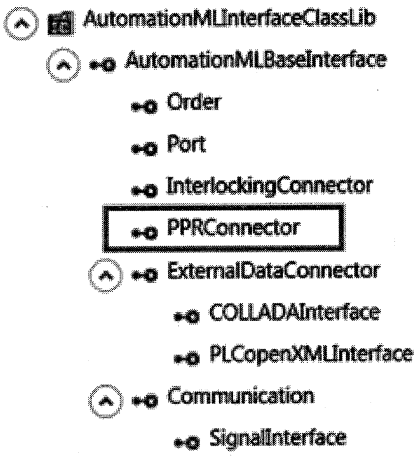


图 A.27 PPR 连接器接口

A.2.5.2 示例

下面的示例(见图 A.28)给出了在 AML 中该概念的应用。示例包含 2 个传送带(C1 和 C2)，一个转台(TT1)和一个机器人(RB1)。这些是工厂的资源。机器人将车轮装配到汽车上，轮胎和汽车都是产品。示例中的生产过程是运输、转向和装配。

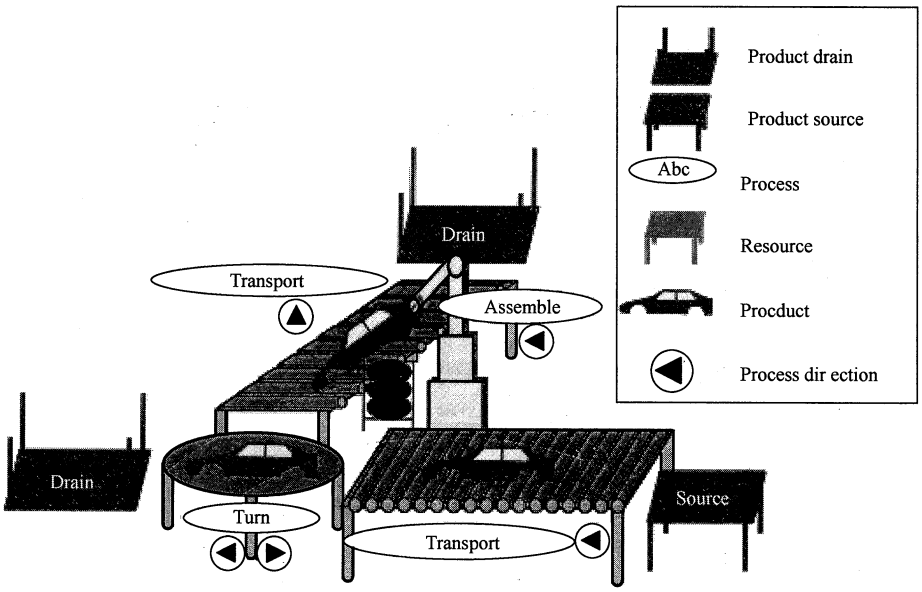


图 A.28　产品-过程-资源概念的示例

在 AML 中，过程-产品-资源概念是通过 CAEX 角色概念（见 IEC 62424:2016 中 A.2.9）以及元素间的关系建模的（见 5.5.3）。被赋予资源“Resource”、产品“Product”和过程“Process”角色的元素集合是两两互斥的，这意味着一个资源不能同时是一个产品或过程。对应的角色类是 AML 基础角色类库的一部分（见图 A.29 和 6.4）。

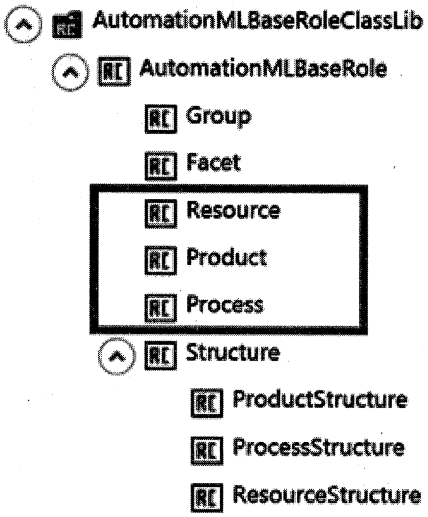


图 A.29　过程-产品-资源概念需要的 AML 角色

在示例中（见图 A.28），角色资源“Resource”被赋予传送带、机器人和转台。汽车和轮胎被赋予角色产品，角色过程被赋予运输、转向和装配。所有元素都存储在图 A.30 中的对应子树中。过程、产品或资源的顺序能通过对应的顺序类型接口 Order 之间的链接明确表达（由于可读性的原因，本例未做描述）。

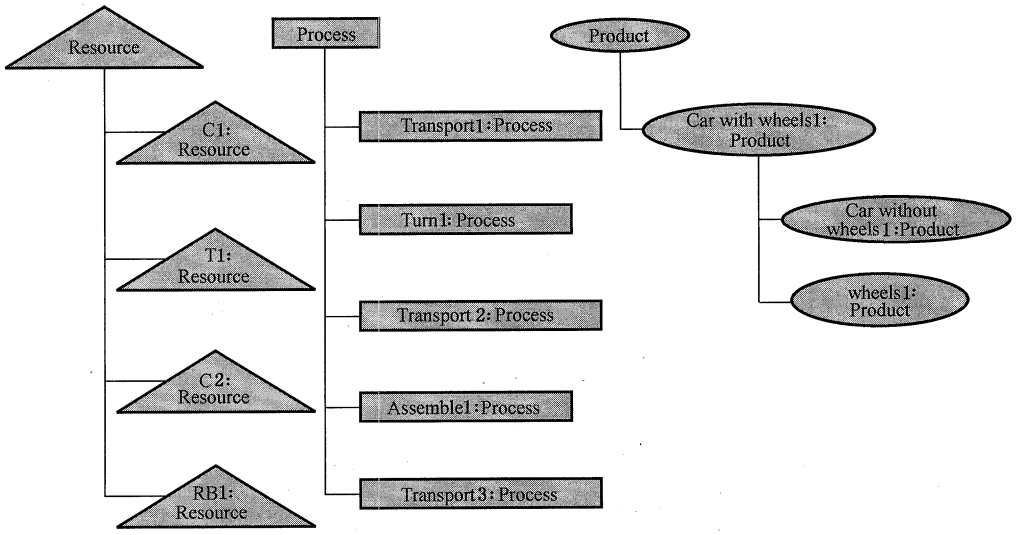


图 A.30　示例中的元素

示例中的每个元素都有一个 PPR 连接器接口。图 A.31 中给出本示例的完整链接。实线代表了由资源到过程的链接，点线代表了由过程到产品的链接，短划线代表了由资源到产品的链接。但这样的复杂性较高。因此，多余的连接能被忽略。

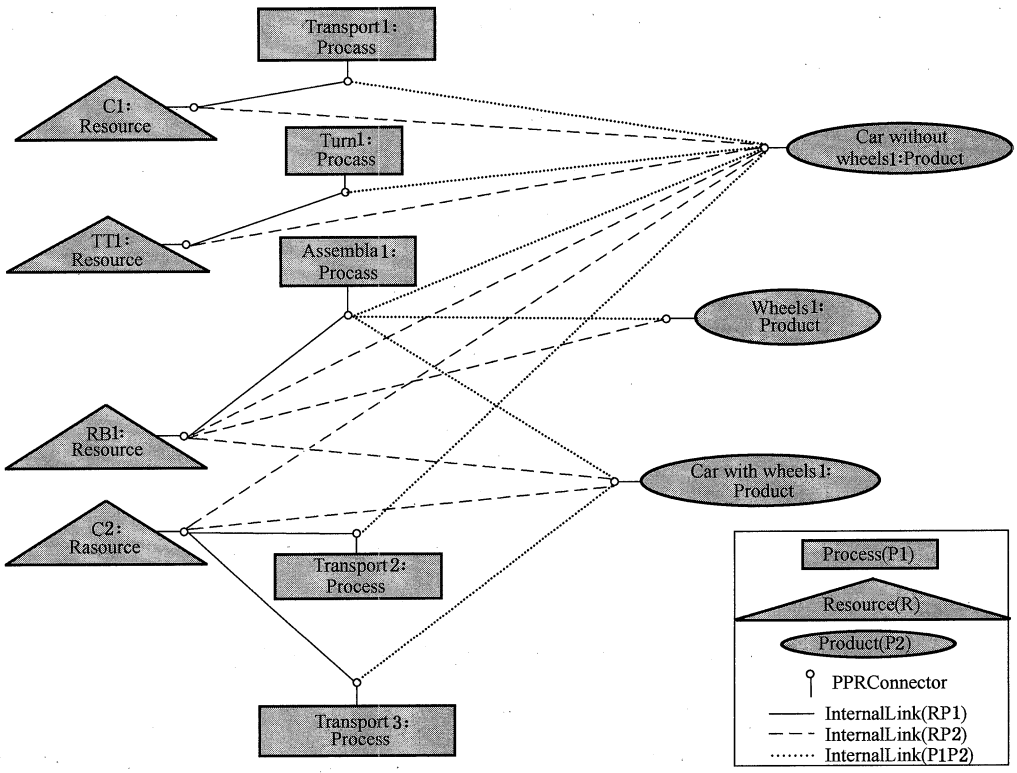


图 A.31　示例中的链接

图 A.32 给出了本示例中以资源为中心的视图。因此，只需要 12 条而不是 19 条链接。传送带“C1”、转台“TT1”和传送带“C2”都与产品无轮汽车“Car without wheels1”相连接。由于机器人在传送带“C2”上将轮胎装配到汽车上，所以机器人“RB1”链接到无轮汽车“Car without wheels1”、有轮汽车“Car with wheels1”和轮胎“Wheels1”。此外，传送带“C2”与有轮汽车“Car with wheels1”之间也有链接。过程“Transport1”连到“C1”，“Transport2”和“Transport3”与传送带“C2”连接。“Assemble1”连到

机器人“RB1”,“Turn1”连到转台“TT1”。由产品到过程的链接(图 A.31 中由点线表示)可派生自已经存在的链接。该模型能被任意旋转或排列,从而使得产品或过程类型的元素被置于中心。

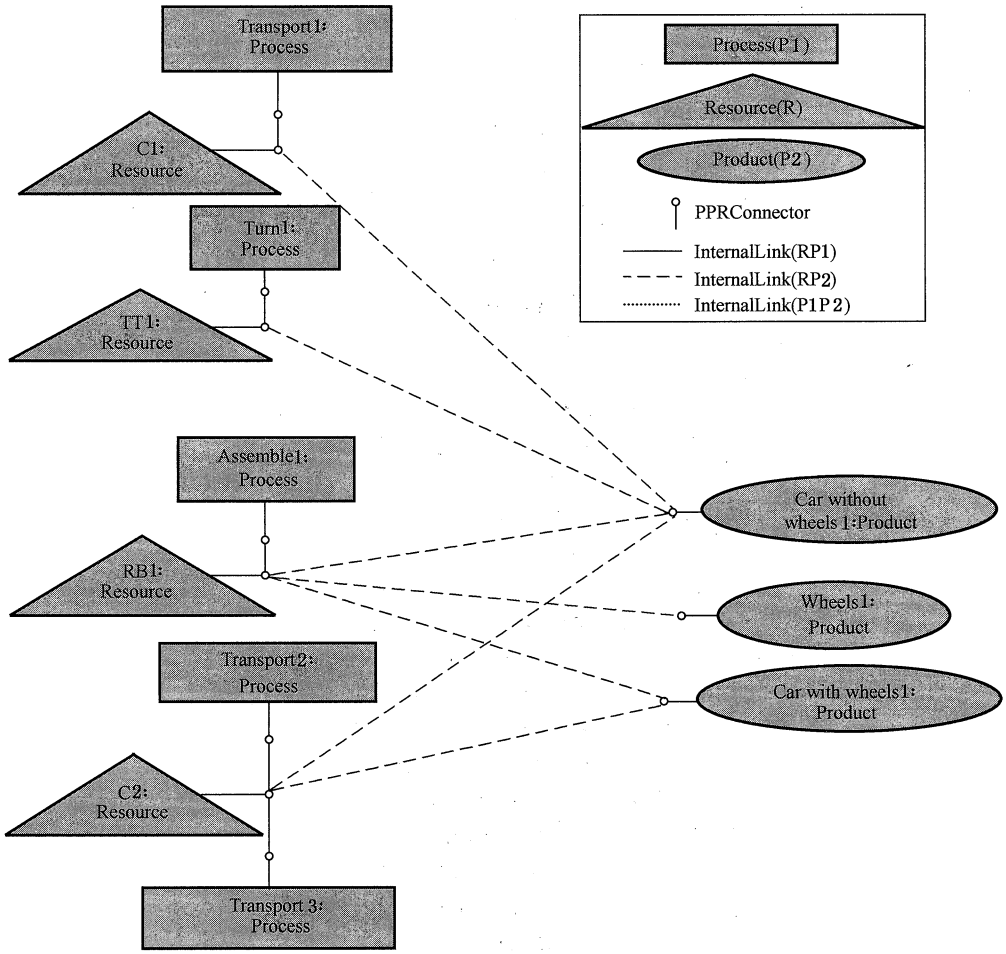


图 A.32　示例中以资源为中心视图的链接

图 A.33 给出了一个 AML 对象树,它突出了传送带“C1”和过程“Transport1”之间的链接。

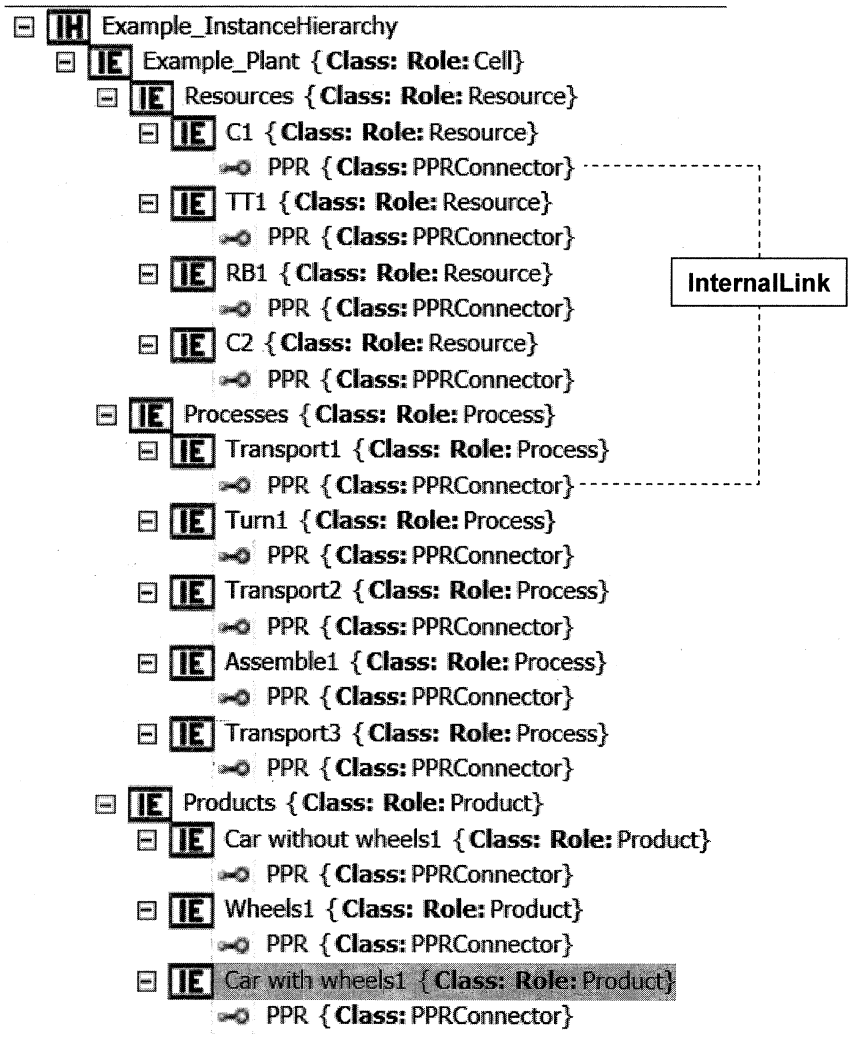


图 A.33　示例中的实例分层结构在 AML 中的表达

图 A.34 给出了对应的 XML 模型。在本例的第一层中有以 CAEX 内部元素建模的三个基本元素：资源“Resource”、过程“Process”和产品“Product”。

InstanceHierarchy

= Name	Example_InstanceHierarchy

InternalElement

= Name	Example_Plant
= ID	GUID1

InternalElement (3)

	= Name	= ID	{} InternalElement
1	Resources	GUID2	InternalElement (4)
2	Processes	GUID7	InternalElement (5)
3	Products	GUID13	InternalElement (3)

InternalElement (4)

	= Name	= ID
1	C1	GUID3
2	TT1	GUID4
3	RB1	GUID5
4	C2	GUID6

InternalElement (5)

	= Name	= ID
1	Transport1	GUID8
2	Turn1	GUID9
3	Transport2	GUID10
4	Assemble1	GUID11
5	Transport3	GUID12

InternalElement (3)

	= Name	= ID
1	CarWithoutWheels1	GUID14
2	Wheels1	GUID15
3	CarWithWheels1	GUID16

图 A.34　示例中的内部元素

对象资源“Resource”的下层存在本例中的四个元素：两条传送带、转台和机器人，它们也是内部元素类型。它们都有一个外部接口 PPR 连接器，并且被赋予角色类资源“Resource”。过程和产品也同样有一个接口和角色。本例中为了将各元素相链接，通常将内部链接布置在顶层基本元素所在的一层。图 A.35 给出了链接的 XML 表达。

InternalLink (12)

	= Name	= RefPartnerSideA	= RefPartnerSideB
1	C1_T1	GUID3_PPR	GUID8_PPR
2	TT1_Tu1	GUID4_PPR	GUID9_PPR
3	RB1_A1	GUID5_PPR	GUID11_PPR
4	C2_T2	GUID6_PPR	GUID10_PPR
5	C2_T3	GUID6_PPR	GUID12_PPR
6	C1_CwW1	GUID3_PPR	GUID14_PPR
7	TT1_CwW1	GUID4_PPR	GUID14_PPR
8	RB1_CwW1	GUID5_PPR	GUID14_PPR
9	RB1_W1	GUID5_PPR	GUID15_PPR
10	RB1_CW1	GUID5_PPR	GUID16_PPR
11	C2_CwW1	GUID6_PPR	GUID14_PPR
12	C2_CW1	GUID6_PPR	GUID16_PPR

图 A.35　示例中的内部链接

图 A.36 给出了本示例完整的 XML 表述。

```
<InstanceHierarchy Name="Example_InstanceHierarchy">
  <InternalElement Name="Example_Plant" ID="GUID1">
    <InternalElement Name="Resources" ID="GUID2">
      <InternalElement Name="C1" ID="GUID3">
        <ExternalInterface Name="PPR" ID="GUID3_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
      </InternalElement>
      <InternalElement Name="TT1" ID="GUID4">
        <ExternalInterface Name="PPR" ID="GUID4_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
      </InternalElement>
      <InternalElement Name="RB1" ID="GUID5">
        <ExternalInterface Name="PPR" ID="GUID5_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
      </InternalElement>
      <InternalElement Name="C2" ID="GUID6">
        <ExternalInterface Name="PPR" ID="GUID6_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
      </InternalElement>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
    </InternalElement>
    <InternalElement Name="Processes" ID="GUID7">
      <InternalElement Name="Transport1" ID="GUID8">
        <ExternalInterface Name="PPR" ID="GUID8_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Process"/>
      </InternalElement>
      <InternalElement Name="Turn1" ID="GUID9">
        <ExternalInterface Name="PPR" ID="GUID9_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Process"/>
      </InternalElement>
      <InternalElement Name="Transport2" ID="GUID10">
        <ExternalInterface Name="PPR" ID="GUID10_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Process"/>
      </InternalElement>
      <InternalElement Name="Assemble1" ID="GUID11">
        <ExternalInterface Name="PPR" ID="GUID11_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Process"/>
      </InternalElement>
      <InternalElement Name="Transport3" ID="GUID12">
        <ExternalInterface Name="PPR" ID="GUID12_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Process"/>
      </InternalElement>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Process"/>
    </InternalElement>
    <InternalElement Name="Products" ID="GUID13">
      <InternalElement Name="CarWithoutWheels1" ID="GUID14">
        <ExternalInterface Name="PPR" ID="GUID14_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Product"/>
      </InternalElement>
      <InternalElement Name="Wheels1" ID="GUID15">
        <ExternalInterface Name="PPR" ID="GUID15_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Product"/>
      </InternalElement>
      <InternalElement Name="CarWithWheels1" ID="GUID16">
        <ExternalInterface Name="PPR" ID="GUID16_PPR" RefBaseClassPath="AutomationMLInterfaceClassLib/.../PPRConnector"/>
        <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Product"/>
      </InternalElement>
      <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Product"/>
    </InternalElement>
    <InternalLink Name="C1_T1" RefPartnerSideA="GUID3_PPR" RefPartnerSideB="GUID8_PPR"/>
    <InternalLink Name="TT1_Tu1" RefPartnerSideA="GUID4_PPR" RefPartnerSideB="GUID9_PPR"/>
    <InternalLink Name="RB1_A1" RefPartnerSideA="GUID5_PPR" RefPartnerSideB="GUID11_PPR"/>
    <InternalLink Name="C2_T2" RefPartnerSideA="GUID6_PPR" RefPartnerSideB="GUID10_PPR"/>
    <InternalLink Name="C2_T3" RefPartnerSideA="GUID6_PPR" RefPartnerSideB="GUID12_PPR"/>
    <InternalLink Name="C1_CwW1" RefPartnerSideA="GUID3_PPR" RefPartnerSideB="GUID14_PPR"/>
    <InternalLink Name="TT1_CwW1" RefPartnerSideA="GUID4_PPR" RefPartnerSideB="GUID14_PPR"/>
    <InternalLink Name="RB1_CwW1" RefPartnerSideA="GUID5_PPR" RefPartnerSideB="GUID14_PPR"/>
    <InternalLink Name="RB1_W1" RefPartnerSideA="GUID5_PPR" RefPartnerSideB="GUID15_PPR"/>
    <InternalLink Name="RB1_CW1" RefPartnerSideA="GUID5_PPR" RefPartnerSideB="GUID16_PPR"/>
    <InternalLink Name="C2_CwW1" RefPartnerSideA="GUID6_PPR" RefPartnerSideB="GUID14_PPR"/>
    <InternalLink Name="C2_CW1" RefPartnerSideA="GUID6_PPR" RefPartnerSideB="GUID16_PPR"/>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Cell"/>
  </InternalElement>
</InstanceHierarchy>
```

图 A.36　示例的实例分层结构在 XML 中的表达

A.2.6 AML 多语言表达概念

A.2.6.1 概念描述

多语言表达概念的目的旨在相同 AML 文件中存储多种语言信息。

A.2.6.2 一个具有三个本地化标签属性的示例

图 A.37 给出了一个 AML 多语言表达示例。对象“PC123”包含属性“Label”。属性“Label”本身包含默认语言表达。实现本地化语言模型文本需要嵌套属性。为此，属性“Label”包含子属性“en-US”“de-DE”和“fr-FR”。子属性的名称与 RFC 5646 中规定的语言代码相对应，其值包含相应的本地化文本。

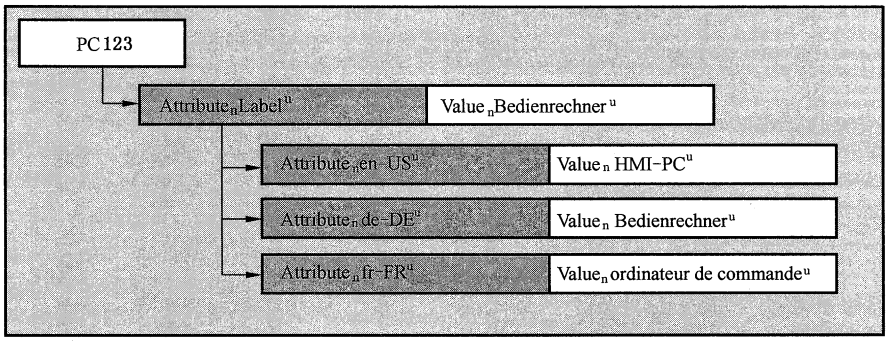


图 A.37 AML 多语言表达概念描述示例

图 A.38 给出了依照 8.6 中规定，使用 AML 进行建模的示例。

InternalElement				
= Name	PC123			
= ID	GUID1			
Attribute				
= Name	Label			
{} Description	Display name for this element. The sub attributes provide the language code according to RFC 5646.			
{} Value	Bedienrechner			
Attribute (3)				
		= Name	= RefAttributeType	{} Value
	1	en-US	AutomationMLBaseAttributeTypeLib/LocalizedAttribute	HMI-PC
	2	de-DE	AutomationMLBaseAttributeTypeLib/LocalizedAttribute	Bedienrechner
	3	fr-FR	AutomationMLBaseAttributeTypeLib/LocalizedAttribute	ordinateur de commande
RoleRequirements				
= RefBaseRoleClassPath	AutomationMLBaseRoleClassLib/.../Resource			

图 A.38 AML 多语言表达概念 XML 描述

图 A.39 给出了该示例对应的 XML 代码。

```
<InternalElement Name="PC123" ID="GUID1">
    <Attribute Name="Label">
        <Description>Display name for this element. The sub attributes provide the language code according to RFC 5646.</Description>
        <Value>Bedienrechner</Value>
        <Attribute Name="en-US" RefAttributeType="AutomationMLBaseAttributeTypeLib/LocalizedAttribute">
            <Value>HMI-PC</Value>
        </Attribute>
        <Attribute Name="de-DE" RefAttributeType="AutomationMLBaseAttributeTypeLib/LocalizedAttribute">
            <Value>Bedienrechner</Value>
        </Attribute>
        <Attribute Name="fr-FR" RefAttributeType="AutomationMLBaseAttributeTypeLib/LocalizedAttribute">
            <Value>ordinateur de commande</Value>
        </Attribute>
    </Attribute>
    <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
</InternalElement>
```

图 A.39 AML 多语言表达概念 XML 文本描述

A.2.6.3 一个具有三个本地化标签属性类型的示例

另一个使用案例是进行属性预定义，该属性包括库中所有需要的本地化语言表达。因此，以频繁使用的可视化台式机(HMI-PC)为例，如果对于这类特定对象，在库中进行建模是有用的。图 A.40 通过 CAEX 属性类型 HMI-PC-LabelType 给出了该描绘，它预定义了所有 3 个本地化语言表达。

AttributeTypeLib				
= Name	MyAttributeTypeLibrary			
AttributeType				
= Name	HMI-PC-LabelType			
= AttributeDataType	xs:string			
{} Description	Display name for this element. The sub attributes provide the language code according to RFC 5646.			
{} Value	Bedienrechner			
Attribute (3)				
		= Name	= RefAttributeType	{} Value
	1	en-US	AutomationMLBaseAttributeTypeLib/LocalizedAttribute	HMI-PC
	2	de-DE	AutomationMLBaseAttributeTypeLib/LocalizedAttribute	Bedienrechner
	3	fr-FR	AutomationMLBaseAttributeTypeLib/LocalizedAttribute	ordinateur de commande

图 A.40 多语言属性类型 AML 模型

图 A.41 给出了上述示例的 XML 代码。

```
<AttributeTypeLib Name="MyAttributeTypeLibrary">
    <AttributeType Name="HMI-PC-LabelType" AttributeDataType="xs:string">
        <Description>Display name for this element. The sub attributes provide the language code according to RFC 5646.</Description>
        <Value>Bedienrechner</Value>
        <Attribute Name="en-US" RefAttributeType="AutomationMLBaseAttributeTypeLib/LocalizedAttribute">
            <Value>HMI-PC</Value>
        </Attribute>
        <Attribute Name="de-DE" RefAttributeType="AutomationMLBaseAttributeTypeLib/LocalizedAttribute">
            <Value>Bedienrechner</Value>
        </Attribute>
        <Attribute Name="fr-FR" RefAttributeType="AutomationMLBaseAttributeTypeLib/LocalizedAttribute">
            <Value>ordinateur de commande</Value>
        </Attribute>
    </AttributeType>
</AttributeTypeLib>
```

图 A.41 多语言属性类型 XML 代码

这样，无论在任何地方，对于一个需要包含本地化语言表达的可视化台式机(HMI-PC)标签而言，它一定要引用该属性类型并可获得所有预定义本地化语言表达。该项技术可以用来对库中特定多语言表达的任意项目进行建模。

A.2.7 属性列表和数组

A.2.7.1 概念描述

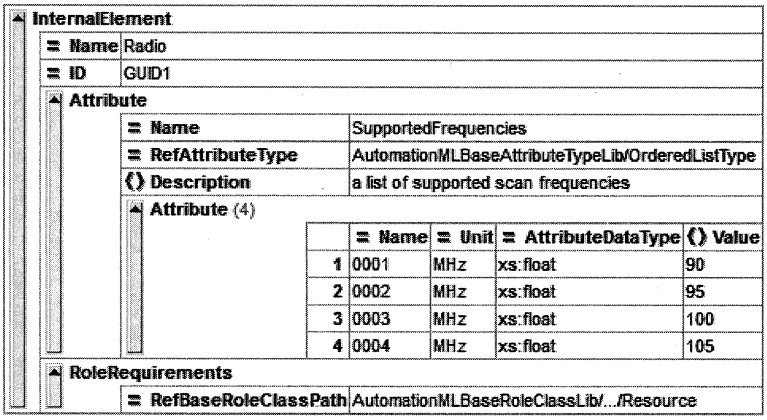
在实际应用中，属性通常是结构化的。数据类型的列表或数组需要进行建模并在工程工具间进行交换。列表包含相同数据类型项，也可包含更多列表，支持以任意维度进行数组的建模。

最后，采用 AML 进行列表建模，形成包含嵌套子属性的属性列表。属性列表作为列表的容器。为了将一个 CAEX 属性声明为一个列表，需要引用属性类型“ListType”或者“OrderedListType”，分别对无序列表或者有序列表建模。

列表或数组建模的标准化规定在 8.8 中定义。

A.2.7.2 示例

图 A.42 中，给出了对象“Radio”存储支持频率的列表示例。属性“SupportedFrequencies”引用了 AML 属性类型“OrderedListType”。子属性形成了列表项，每个名称都包含列表中的索引项。前导 0 是被允许存在及规范化添加的，以支持更好的分类。



InternalElement

Name	Radio
ID	GUID1

Attribute

Name	SupportedFrequencies
RefAttributeType	AutomationMLBaseAttributeTypeLib/OrderedListType
Description	a list of supported scan frequencies

Attribute (4)

	Name	Unit	AttributeDataType	Value
1	0001	MHz	xs:float	90
2	0002	MHz	xs:float	95
3	0003	MHz	xs:float	100
4	0004	MHz	xs:float	105

RoleRequirements

RefBaseRoleClassPath	AutomationMLBaseRoleClassLib/.../Resource

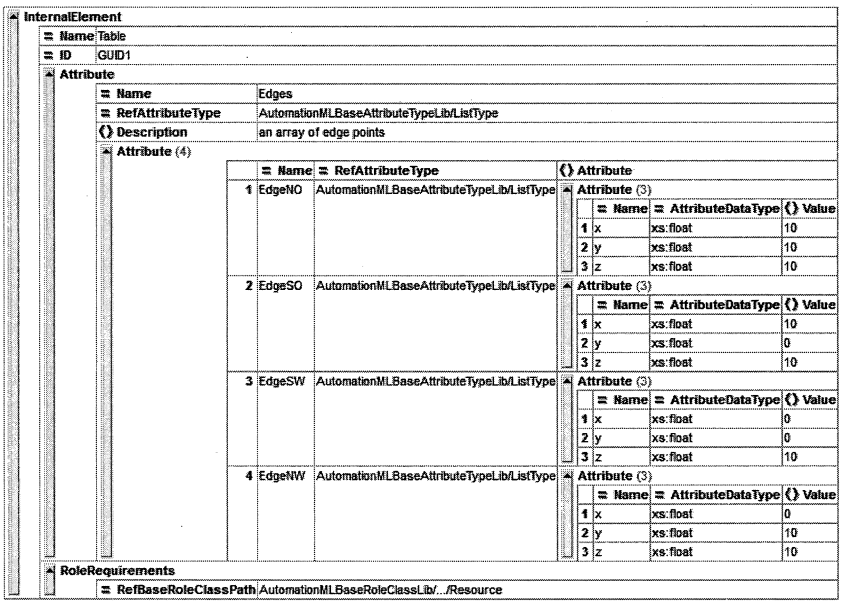
图 A.42　属性列表“SupportedFrequencies”

图 A.43 给出了示例的相关 XML 代码。

```
<InternalElement Name="Radio" ID="GUID1">
  <Attribute Name="SupportedFrequencies" RefAttributeType="AutomationMLBaseAttributeTypeLib/OrderedListType">
    <Description>a list of supported scan frequencies</Description>
    <Attribute Name="0001" Unit="MHz" AttributeDataType="xs:float">
      <Value>90</Value>
    </Attribute>
    <Attribute Name="0002" Unit="MHz" AttributeDataType="xs:float">
      <Value>95</Value>
    </Attribute>
    <Attribute Name="0003" Unit="MHz" AttributeDataType="xs:float">
      <Value>100</Value>
    </Attribute>
    <Attribute Name="0004" Unit="MHz" AttributeDataType="xs:float">
      <Value>105</Value>
    </Attribute>
  </Attribute>
  <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
</InternalElement>
```

图 A.43　属性列表“SupportedFrequencies”的 XML 代码

图 A.44 中，给出了对存储边缘点的对象“Table”进行数组建模的示例。属性“Edges”引用了 AML 属性类型“ListType”。子属性形成列表项，其名称是任意的，但是在同层级中要求唯一。它们再次形成列表，包含表中的位置 x，y，z。



InternalElement

Name	Table
ID	GUID1

Attribute

Name	Edges
RefAttributeType	AutomationMLBaseAttributeTypeLib/ListType
Description	an array of edge points

Attribute (4)

	Name	RefAttributeType	Attribute
1	EdgeNO	AutomationMLBaseAttributeTypeLib/ListType	Attribute (3)
2	EdgeSO	AutomationMLBaseAttributeTypeLib/ListType	Attribute (3)
3	EdgeSW	AutomationMLBaseAttributeTypeLib/ListType	Attribute (3)
4	EdgeNW	AutomationMLBaseAttributeTypeLib/ListType	Attribute (3)

Attribute (3) — EdgeNO

	Name	AttributeDataType	Value
1	x	xs:float	10
2	y	xs:float	10
3	z	xs:float	10

Attribute (3) — EdgeSO

	Name	AttributeDataType	Value
1	x	xs:float	10
2	y	xs:float	0
3	z	xs:float	10

Attribute (3) — EdgeSW

	Name	AttributeDataType	Value
1	x	xs:float	0
2	y	xs:float	0
3	z	xs:float	10

Attribute (3) — EdgeNW

	Name	AttributeDataType	Value
1	x	xs:float	0
2	y	xs:float	10
3	z	xs:float	10

RoleRequirements

RefBaseRoleClassPath	AutomationMLBaseRoleClassLib/.../Resource

图 A.44　数组“Edges”的 CAEX 模型示例

图 A.45 给出了该示例的对应 XML 代码。

```
<InternalElement Name="Table" ID="GUID1">
  <Attribute Name="Edges" RefAttributeType="AutomationMLBaseAttributeTypeLib/ListType">
    <Description>an array of edge points</Description>
    <Attribute Name="EdgeNO" RefAttributeType="AutomationMLBaseAttributeTypeLib/ListType">
      <Attribute Name="x" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
      <Attribute Name="y" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
      <Attribute Name="z" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
    </Attribute>
    <Attribute Name="EdgeSO" RefAttributeType="AutomationMLBaseAttributeTypeLib/ListType">
      <Attribute Name="x" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
      <Attribute Name="y" AttributeDataType="xs:float">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="z" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
    </Attribute>
    <Attribute Name="EdgeSW" RefAttributeType="AutomationMLBaseAttributeTypeLib/ListType">
      <Attribute Name="x" AttributeDataType="xs:float">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="y" AttributeDataType="xs:float">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="z" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
    </Attribute>
    <Attribute Name="EdgeNW" RefAttributeType="AutomationMLBaseAttributeTypeLib/ListType">
      <Attribute Name="x" AttributeDataType="xs:float">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="y" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
      <Attribute Name="z" AttributeDataType="xs:float">
        <Value>10</Value>
      </Attribute>
    </Attribute>
  </Attribute>
  <RoleRequirements RefBaseRoleClassPath="AutomationMLBaseRoleClassLib/.../Resource"/>
</InternalElement>
```

图 A.45　属性数组“Edges”的 XML 代码

附 录 B
（资料性附录）
标准 AML 基础库的 XML 表达

图 B.1 给出了标准 AML 接口类库、角色类库和属性类库的 XML 文本。

```
<CAEXFile xmlns="http://www.dke.de/CAEX" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
          instance" SchemaVersion="3.0" FileName="AutomationML2.10BaseLibraries"
          xsi:schemaLocation="http://www.dke.de/CAEX CAEX_ClassModel_V.3.0.xsd">
 <SuperiorStandardVersion>AutomationML 2.10</SuperiorStandardVersion>
 <SourceDocumentInformation OriginID="IEC SC65E WG 9" OriginName="IEC SC65E WG 9"
          OriginVersion="2.10.0" LastWritingDateTime="2016-08-25T09:58:00.0Z"
          OriginProjectID="Automation Markup Language Standard Library"
          OriginRelease="2.10.0" OriginVendor="IEC" OriginVendorURL="www.iec.ch"
          OriginProjectTitle="Automation Markup Language Standard Libraries"/>
 <InterfaceClassLib Name="AutomationMLInterfaceClassLib">
  <Description>Standard Automation Markup Language Interface Class Library</Description>
  <Version>2.10.0</Version>
  <InterfaceClass Name="AutomationMLBaseInterface">
   <InterfaceClass Name="Order" RefBaseClassPath="AutomationMLBaseInterface">
    <Attribute Name="Direction" AttributeDataType="xs:string"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction"/>
   </InterfaceClass>
   <InterfaceClass Name="Port" RefBaseClassPath="AutomationMLBaseInterface">
    <Attribute Name="Direction" AttributeDataType="xs:string"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction">
     <Constraint Name="AllowedValues">
      <NominalScaledType>
       <RequiredValue>In</RequiredValue>
       <RequiredValue>Out</RequiredValue>
       <RequiredValue>InOut</RequiredValue>
      </NominalScaledType>
     </Constraint>
    </Attribute>
    <Attribute Name="Cardinality"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/Cardinality">
     <Attribute Name="MinOccur" AttributeDataType="xs:unsignedInt"/>
     <Attribute Name="MaxOccur" AttributeDataType="xs:unsignedInt"/>
    </Attribute>
    <Attribute Name="Category" AttributeDataType="xs:string"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/Category"/>
   </InterfaceClass>
   <InterfaceClass Name="PPRConnector" RefBaseClassPath="AutomationMLBaseInterface"/>
   <InterfaceClass Name="ExternalDataConnector"
          RefBaseClassPath="AutomationMLBaseInterface">
    <Attribute Name="refURI" AttributeDataType="xs:anyURI"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI"/>
    <InterfaceClass Name="COLLADAInterface" RefBaseClassPath="ExternalDataConnector"/>
    <InterfaceClass Name="PLCopenXMLInterface" RefBaseClassPath="ExternalDataConnector"/>
    <InterfaceClass Name="ExternalDataReference" RefBaseClassPath="ExternalDataConnector">
     <Attribute Name="MIMEType" AttributeDataType="xs:string"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/MIMEType"/>
    </InterfaceClass>
   </InterfaceClass>
   <InterfaceClass Name="Communication" RefBaseClassPath="AutomationMLBaseInterface">
    <InterfaceClass Name="SignalInterface" RefBaseClassPath="Communication"/>
   </InterfaceClass>
  </InterfaceClass>
 </InterfaceClassLib>
 <RoleClassLib Name="AutomationMLBaseRoleClassLib">
  <Description>Automation Markup Language base role class library</Description>
  <Version>2.10.0</Version>
  <RoleClass Name="AutomationMLBaseRole">
   <RoleClass Name="Group" RefBaseClassPath="AutomationMLBaseRole">
    <Attribute Name="AssociatedFacet" AttributeDataType="xs:string"
          RefAttributeType="AutomationMLBaseAttributeTypeLib/AssociatedFacet"/>
   </RoleClass>
   <RoleClass Name="Facet" RefBaseClassPath="AutomationMLBaseRole"/>
   <RoleClass Name="Resource" RefBaseClassPath="AutomationMLBaseRole"/>
   <RoleClass Name="Product" RefBaseClassPath="AutomationMLBaseRole"/>
   <RoleClass Name="Process" RefBaseClassPath="AutomationMLBaseRole"/>
   <RoleClass Name="Structure" RefBaseClassPath="AutomationMLBaseRole">
    <RoleClass Name="ProductStructure" RefBaseClassPath="Structure"/>
    <RoleClass Name="ProcessStructure" RefBaseClassPath="Structure"/>
    <RoleClass Name="ResourceStructure" RefBaseClassPath="Structure"/>
   </RoleClass>
   <RoleClass Name="ExternalData" RefBaseClassPath="AutomationMLBaseRole"/>
```

图 B.1 标准 AML 接口类库、角色类库和属性类库的 XML 文本

```
    </RoleClass>
  </RoleClassLib>
  <AttributeTypeLib Name="AutomationMLBaseAttributeTypeLib">
    <Description>Standard Automation Markup Language Attribute Type Library</Description>
    <Version>2.10.0</Version>
    <AttributeType Name="Direction" AttributeDataType="xs:string">
      <Constraint Name="AllowedValues">
        <NominalScaledType>
          <RequiredValue>In</RequiredValue>
          <RequiredValue>Out</RequiredValue>
          <RequiredValue>InOut</RequiredValue>
        </NominalScaledType>
      </Constraint>
    </AttributeType>
    <AttributeType Name="Cardinality">
      <Attribute Name="MinOccur" AttributeDataType="xs:unsignedInt"/>
      <Attribute Name="MaxOccur" AttributeDataType="xs:unsignedInt"/>
    </AttributeType>
    <AttributeType Name="Category" AttributeDataType="xs:string"/>
    <AttributeType Name="refURI" AttributeDataType="xs:anyURI"/>
    <AttributeType Name="AssociatedFacet" AttributeDataType="xs:string"/>
    <AttributeType Name="ListType"/>
    <AttributeType Name="OrderedListType"/>
    <AttributeType Name="LocalizedAttribute" AttributeDataType="xs:string"/>
    <AttributeType Name="AssociatedExternalValue">
      <Attribute Name="refCAEXAttribute"/>
      <Attribute Name="refURI" RefAttributeType="AutomationMLBaseAttributeTypeLib/refURI"/>
      <Attribute Name="Direction"
            RefAttributeType="AutomationMLBaseAttributeTypeLib/Direction"/>
    </AttributeType>
    <AttributeType Name="MIMEType" AttributeDataType="xs:string"/>
    <AttributeType Name="DocLang" AttributeDataType="xs:string"/>
  </AttributeTypeLib>
</CAEXFile>
```

图 B.1（续）

附 录 NA
（资料性附录）
本部分使用的惯用词语中英文对照

AutomationMLBaseRole	AML 基础角色
AutomationMLBaseRoleClassLib	AML 基础角色类库
AutomationMLInterfaceClassLib	AML 接口类库
ExternalDataConnector	外部数据连接器
ExternalInterface	外部接口
InstanceHierarchy	实例分层结构
InterfaceClass	接口类
InternalElement	内部元素
InternalLink	内部链接
MappingObject	映射对象
PPRConnector	PPR 连接器
RoleClass	角色类
RoleRequirement	角色要求
SystemUnitClass	系统单元类

参 考 文 献

[1] IEC 60027(所有部分) 电气技术用字母符号(Letter symbols to be used in electrical technology)

[2] IEC 62264-1 企业控制系统集成 第1部分:模型和术语(Enterprise-control system integration—Part 1:Models and terminology)

[3] ISO 80000-1 数量和单位 第1部分:总则(Quantities and units—Part 1:General)

[4] Extensible Markup Language (XML) 1.0 1.0:2004, W3C Recommendation [viewed 2017-11-13]. Available at 〈http://www.w3.org/TR/2004/REC-xml-20040204/〉