ICS 25.040.40
CCS J07

中华人民共和国国家标准

GB/T XXXXX-20XX/ISO 20242-2020

工业自动化系统与集成 测试应用的服务接口 第5部分：应用程序服务接口

Industrial automation systems and integration - Service interface for testing applications -

Part 5: Application program service interface

(ISO 20242-5：2020)

（征求意见稿）

在提交反馈意见时，请将您知道的相关专利连同支持性文件一并附上。

20××-××-××发布　　20××-××-××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

本文件按照GB/T 1.1-2020《标准化工作导则 第一部分：标准化文件的结构和起草规则》的规定起草。本文件是GB/T 22270《自动化系统与集成 测试应用的服务接口》的第5部分。GB/T 22270已经发布了以下部分：

——第1部分：概述

——第2部分：资源管理服务接口

——第3部分：虚拟设备服务接口

——第4部分：设备能力专规模板

——第5部分：应用程序服务接口

本文件采用翻译法等同采用ISO 20242-5: 2020《A Industrial automation systems and integration — Service interface for testing applications - Part 5: Application program service interface》。请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中国机械工业联合会提出。

本文件由全国自动化系统与集成标准化技术委员会（SAC/TC159）归口。

本文件起草单位：浙江大学、北京机械工业自动化研究所有限公司、浙江中智达科技有限公司、浙江中创天成科技有限公司、浙江大学宁波理工学院、深圳职业技术学院等。

本文件主要起草人：

引 言

制定ISO 20242系列标准的想法来自国际汽车工业及其供应商，动机是促进自动化采集设备及其外围组件与计算机应用软件的集成。本标准定义在自动化应用环境下、测量应用环境下或自动化和测量应用环境下设备驱动的创建规则即其运行状态。

ISO 20242系列的主要目标是为用户与应用程序提供：

—对计算机操作系统的独立性；

—对设备连接技术（设备接口/网络）的独立性；

—对设备供应商的独立性；

—在给定的计算机平台的情况下，验证所连接设备的设备驱动程序及其行为的能力；

—对未来技术设备发展影响的独立性

ISO 20242系列不需要开发新的设备或利用专门的接口技术（网络）。 本标准将设备及其通信接口封装在一起，以和同类其他设备在指定应用情况下兼容。

GB/T 22270《自动化系统与集成 测试应用的服务接口》由以下部分构成：

——第1部分：概述；

——第2部分：资源管理服务接口；

——第3部分：虚拟设备服务接口；

——第4部分：设备能力专规模板；

——第5部分：应用程序服务接口；

——第6部分：一致性测试方法、标准和报告。

本文件是GB/T 22270《自动化系统与集成 测试应用的服务接口》的第5部分，需要与标准的其他部分结合使用。

本文件定义了一种使用协调器服务的面向对象的接口形式化、语义化描述方法，同时也定义了虚拟设备、虚拟环境的形式化、语义化描述方法。定义了应用程序服务接口的框架并给出了接口设计、接口编程的参考指南。

系列标准的第1部分定义了系列标准的组成结构和测试应用服务接口的架构。本部分标准与标准第3部分协同，指导集成设备驱动和虚拟设备服务接口；与标准第4部分协同，指导集成参数化实例描述、设备能力描述和设备接口专规。标准的第2部分定义集成系统资源，标准的第6部分与系列标准ISO 13209结合，给出了测试应用服务接口的一致性测试方法、标准和报告。

工业自动化系统与集成测试应用的服务接口
第5部分：应用程序服务接口

1 范围

本文件定义了一种使用协调器服务的面向对象的接口和虚拟设备配置和环境搭建的形式化、语义化的描述准则。

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 20242-1,工业自动化与系统与集成　测试应用的服务接口第1部分：概述

ISO 20242-2, 工业自动化与系统与集成　测试应用的服务接口 第2部分：资源管理服务接口

ISO 20242-3, 工业自动化与系统与集成　测试应用的服务接口 第3部分：虚拟设备服务接口

ISO 20242-4, 工业自动化与系统与集成　测试应用的服务接口 第4部分：设备能力专规模板

ISO 13209-1, 道路车辆　开路测试序列交换格式(OTX) 第1部分：总论和用例

ISO 13209-2, 道路车辆　开路测试序列交换格式(OTX) 第2部分：核心数据模型规格和要求

3 术语和定义

ISO 20242-1、ISO 20242-2、ISO 20242-3、ISO 20242-4、ISO 13209-1和ISO 13209-2中界定的术语和定义适用于本文件。

ISO和IEC在维护标准化中使用的术语数据库为以下网址：

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

IEC Electropedia: available at http://www.electropedia.org/

3.1

通信对象 communication object

可通过一个通信功能访问到，并能读写值的现有对象。

[来源: ISO 20242-1:2005, 2.3]

3.2

协调器 coordinator

具备特定接口的程序，负责应用程序对一个或多个设备驱动（3.5）的访问，并管理实时应用情况、同步和事件。

[来源: ISO 20242-1:2005, 2.4]

3.3

协调器能力描述 coordinator capability description

规定格式（即结构和语法）中包含有关协调器（3.2）能力信息的文本文件。

[来源: ISO 20242-1:2005, 2.5, 已修订]

3.4

协调器服务 coordinator services

协调器（3.2）为与应用程序交换数据的服务。

3.5

设备能力描述 device capability description

含具有规定格式（即结构和语法）的包含虚拟设备（3.10）能力信息的文本文件。

[来源: ISO 20242-1:2005, 2.5, 已修订]

3.6

设备驱动 device driver
提供具有服务功能的ISO 20242指定接口的软件模块，用于调用平台适配器访问物理设备。
[来源: ISO 20242-2:2010, 3.1]

3.7

功能对象 function object
描述虚拟设备（3.10）一项能力的实例。
[来源: ISO 20242-3:2011, 3.4]

3.8

操作 operation
描述一个完整过程的实例。
[来源: ISO 20242-3:2011, 3.5]

3.9

参数化实例描述 parameterization instance description
有关协调器（3.2）配置和虚拟设备（3.10）配置的信息。
[来源: ISO 20242-4:2011, 3.8]

3.10

虚拟设备 virtual device
表示一个或多个物理设备和/或独立软件模块，该设备或模块提供了通信接口资源的清晰视图
[来源: ISO 20242-3:2011, 3.7]

3.11

工作区 workspace
为应用程序提供访问点的协调器（3.2）资源分组
[来源: ISO 20242-3:2011, 3.5]

4 缩略语

下列缩略语适用于本文件。
PSI：应用程序服务接口（Application Program Service Interface）
ASCII：美国信息交换标准代码（American Standard Code for Information Interchange）
CCD：协调器能力描述（Coordinator Capability Description）
CO：通信对象（Communication Object）
DCD：设备能力描述（Device Capability Description）
DCPT：设备能力专规（Device Capability Profile Template）
Template DS：领域特定语言模板（Domain Specific Language）
EAI：扩展访问接口（Extended Access Interface）
FAI：完全访问接口（Full Access Interface）
FO：功能对象（Function Object）
NIL：空指针或空引用，不引用有效对象（Null pointer or null reference, does not refer to a valid object）
OP：操作（Operation）
OTX：开放式测试序列交换格式（Open Test sequence eXchange）
PID：参数化实例描述（Parameterization Instance Description）
SAI：智能访问接口（Smart Access Interface）
VD：虚拟设备（Virtual Device）
VDSI：虚拟设备服务接口（Virtual Device Service Interface）
XML：可扩展标记语言 (见REC-xml-2004020（eXtensible Markup Language）

5 应用程序服务接口

5.1 引言

本章节描述了一组由协调器所提供的服务，以供多个应用程序使用（图1）。

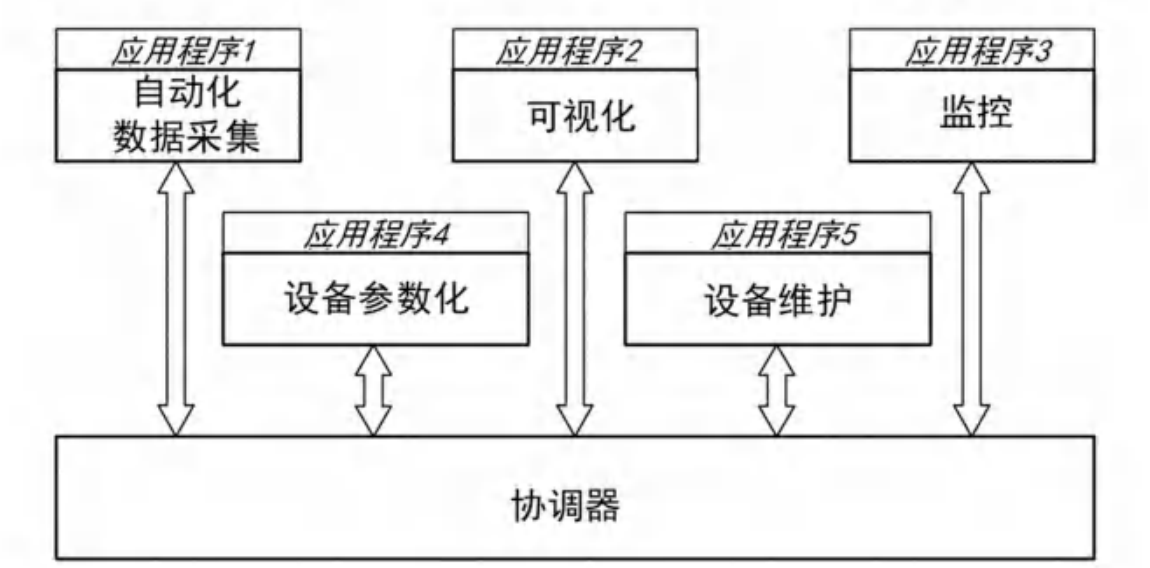


图1 使用协调器的示例性应用程序

应用程序服务接口（APSI）的概念是保持设备的中性，即在没有特定于设备的服务和参数的支持下，与任何设备通信以获取应用相关的数据（图1中应用程序1、2和3）。然而，对于设备的设置和其他与设备绑定的任务来说，也应能传输特定于设备的信息（图1中应用程序4和5）。因此，应用程序服务接口（APSI）定义了不同的接口（见5.4.2），即强制性的智能访问接口（SAI），不应处理任何特定于设备的访问；可选的完全访问接口（FAI），应允许访问设备驱动（见ISO 20242-3）中的虚拟设备。在ISO 20242系列的典型全栈用法中，协调器通过虚拟设备服务接口访问设备驱动（见附录I和ISO 20142-1：2005）。

应用程序应使用实例名称所标识的抽象数据源和汇。根据应用环境的不同，源和汇的数据类型在接口前已经定义或通过接口定义。使用APSI的应用程序通常不包含那些提供所需的信息处理的物理设备。此外，协调器应在应用程序数据与设备数据之间实现内存分离（参见5.5.7）。因此，特定于设备的数据元素应被封装并对应用不可见。

本标准描述了典型应用程序中协调器服务的基础及其用法，将一个面向对象模型与拥有类、属性和方法的基本描述一起使用。默认情况下，服务是同步操作，而异步操作则为可选。使用协调器时，应同时提供记录有协调器相关功能的说明（见I.2）。异步操作通过回调功能予以控制。对于工作空间（见5.4.1）中处理的全局事件来说，应在打开工作空间时对回调功能进行定义（如A.2.2.7中的refCBInformationReport）。此外，应在每个属性处注册本地回调（见5.5.2）。协调器服务通常会映射到所应用的技术、平台和编程语言（如CORBA、JAVA、C++）中去，所以本标准定义服务而而不对特定编程语言的接口进行定义。应在数据表（见I.2）中说明有关特定编程语言各自映射的详细信息（见I.2）。这也同样适用于错误处理，因为在进行相应应用时，应将数据处理的技术限制考虑在内。

附录H描述了本标准基于ISO 13209（OTX）的用法。该标准的符合性测试应按照附录I中的规定来执行。

5.2 参数化

协调器应能够动态链接和释放应用程序和设备驱动，并在设备驱动（ISO 20242-3中定义的虚拟设备、功能对象和通信对象）中能够创建和删除数据对象。这一功能流程应在所有设备驱动中都具备。创建和删除数据对象的顺序不应固定，应可以使用参数实例描述（如ISO 20242-4中定义的PID）来控制实例化程序。

注1：协调器并不了解特定应用程序需要哪些设备功能，但是它可以根据设备的能力描述（DCD，如ISO 20242-4中所定义）与每个设备进行通信。

注2：为了在应用程序所需功能与设备驱动所提供的设备功能之间取得平衡，通常要进行参数化流程的配置，配置将得到一个参数化实例描述（PID），其包含了协调器创建所有必要数据对象的信息。此过程是设备独立性的关键，因为这个过程使得应用程序不需要知道所用设备的特定功能。参数化工具可以创建参数化实例描述（PID），该工具可以是应用程序本身的一部分，也可以是单独的一个软件。

参数化应基于设备驱动可用的设备能力描述（DCD），并应得到ISO 20242-4：2011第8条中定义的多语言文本元素的支持。

注3：PID可以不依赖于协调器创建，也不必要于任何设备连接。

协调器不得更改PID。

参数实例描述中应包含下列信息：

—工作区名称；

—带有版本号的驱动（位置）；

—驱动的设备能力描述；

—虚拟设备（VD），包含用于创建的标识和参数;

—功能对象(FO)，包含用于创建的标识和参数;

—通信对象（CO）和其默认值;

—操作（OP）和其标识;

—具有用于输入参数值的in/out操作结构；

—初始化顺序；

—用于抽象源汇的应用程序引用（实例名称）；

参数实例描述（PID）应基于结构化数据类型的描述，DCD类型定义（typedef）部分的问题也会随之解决。有关详细信息，请参见ISO 20242-4。

注4：通过此参数化为与应用程序相关的配置建立虚拟设备、功能对象和通信对象的初始化值。

生成的参数化实例描述（PID）应分开存储，可读的配置应能被协调器随时调用。

5.3 配置

5.3.1 基于参数化实例描述（PID）的配置

拥有基于参数化实例描述(PID)配置时，在工作区创建的同时，应用将给协调器提供一个基于 ISO 20242-4规范所创建的XML文件。该文件应由协调器自动处理。当应用程序登录协调器时，所要使用的实例描述应通过名称来标识。

如果应用程序知道抽象数据源汇的实例名称和类型的话，则可以在协调器创建的对象引用上进行迭代。这是智能访问接口的一种典型用例。

如果应用程序不知道抽象数据源汇的实例名称和类型，则应在协调器创建的对象引用上进行迭代，并请求抽象数据源汇的名称和类型。这是扩展访问接口的一种典型用例。

协调器应对实例描述所给与的对象进行实例化，从而创建内部抽象数据源汇以供应用程序使用。分配给这些抽象数据源汇的设备驱动数据对象应在相应的虚拟设备中实例化。

注：在设备驱动的生命周期内，这些抽象数据源汇可以被应用读写以被更新。因为协调器了解内部抽象数据源汇与虚拟设备的功能对象之间的关系，协调器可以访问在设备驱动内实例化的数据对象以更新抽象数据源汇。

在使用智能访问接口（SAI）或扩展访问接口（EAI）运行参数化后，虚拟设备的运行状态将设置为“工作”（ISO 20242-3：2011，第7条）。在这种状态下是无法修改参数的。可通过GdiCoWorkspace::gdiSetAllowChangeParameter（boolean bEnable）方法充值参数不可变的状态，从而让应用可以写入参数（见A.2.29.12）。为此，协调器应将相应的虚拟设备转换到适当的状态。

5.3.2 基于服务配置和混合配置

应用程序本身可以在相应参数实例化描述（PID）不具备时完成基于服务配置和混合配置，前提是协调器中配备了完全访问接口

在基于服务的配置中可以创建接口的结构（类和数据类型）和内容（实例和参数）。有关配置过程的详细信息，请参见附录C。

在混合配置中可以在接口上提供没有实例和参数的PID（本标准中称为空PID）。协调器应基于空PID中包含的设备能力描述来构建框架，并创建实例和参数。

对于这两种情况，虚拟设备（VD）的运行状态都可以由应用程序控制。

5.4 协调器结构

5.4.1 工作区

本标准中将工作区定义为一组协调器资源，并为应用程序提供访问点。协调器应提供几个分别实行管理的工作区。一个应用程序可以使用一个或多个工作区。工作区应按名称标识，并连接且仅连接至一个应用程序（图2），可选的，也可以同时连接一个监控应用（有关详细信息见5.5.5）

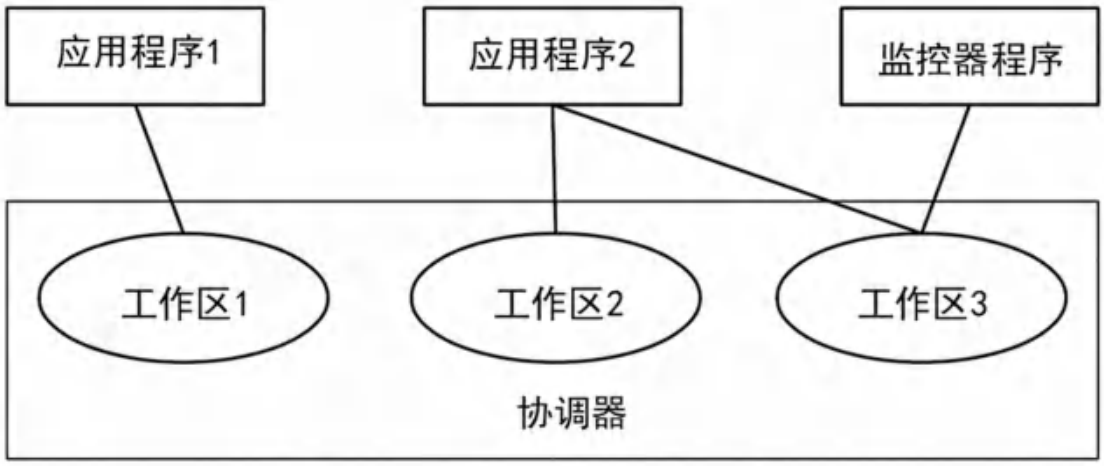


图2 应用程序和工作区

应用程序服务接口（APSI）应是多客户接口，因为这样多个工作区才可以被同时访问。无论设备数量、设备驱动和设备能力描述如何变化，参数话实力描述文献应对工作区进行准确描述。

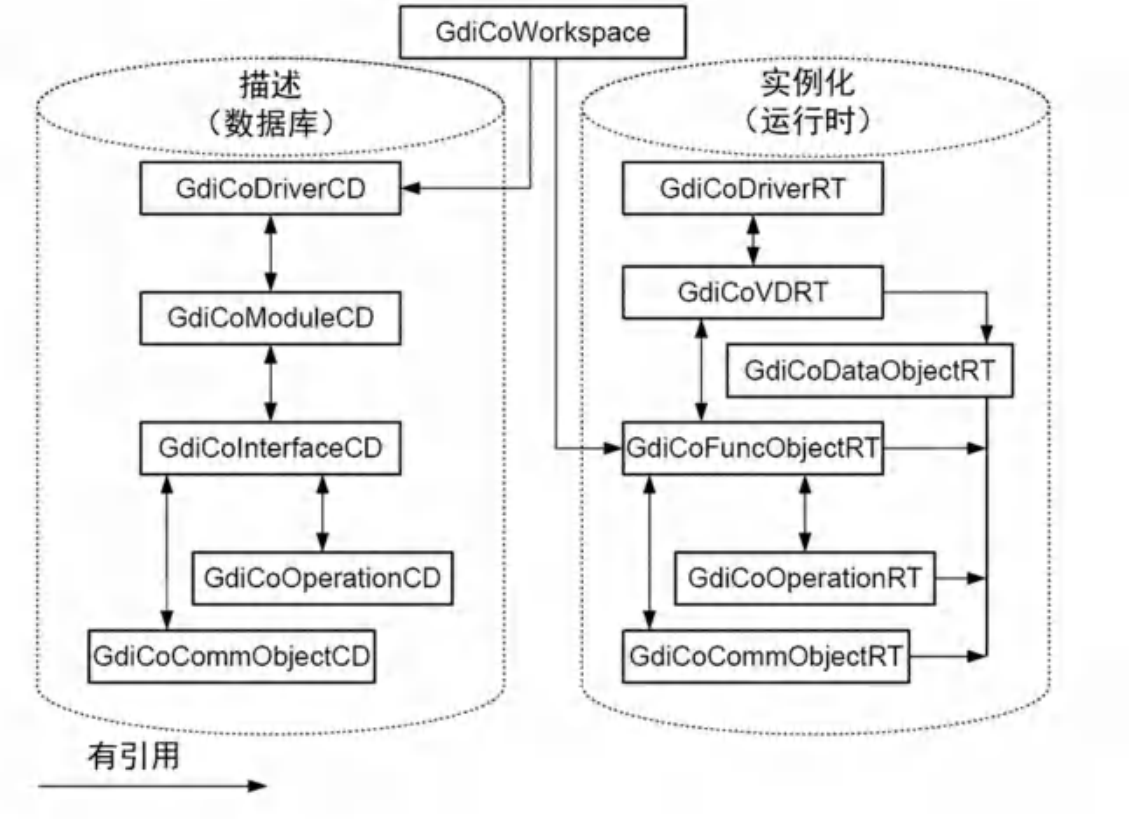


图3 服务组织的工作区视图

连接到工作区后，应用程序应设置句柄以用于辨识。这样的话，连接后的任何接口访问均以该应用程序手柄作为参数。

协调器应能够同时处理多个应用程序。协调器为每个应用程序设置的对象仅对此应用程序可见而对其他应用不可见。如果多个应用程序需要访问相同的协调器对象，则应由连接到工作区的应用程序来进行组织。

本标准中定义的访问服务应按照附录A中智能访问接口（SAI）、附录B中扩展访问接口（EAI）和附录C中完全访问接口（FAI）的规定来执行。提供有关枚举、状态和标识等信息的其他服务应按照附录D中的规定来执行。对工作区来说，服务分列其两侧，一侧称为“描述”，另一侧称为“实例化”（见图3）。

“描述”一端也被称之为数据库（Data Base）端，因为所包含的信息是静态的，并且代表了来自相应设备驱动的设备能力描述的类描述。实例化一端称为“运行时”端，因为它表示已实现的实例，包括在运行时使用的默认初始值。如果应用程序仅使用“运行时”一端的服务，那么各自的设备驱动和虚拟设备（VD）的分配应是看不到的，相反应提供抽象数据源汇的集中资源。

设备驱动和DCD组成一个单元。每一个设备驱动恰好会有一个“描述”端（代表相应的DCD）和一个“运行时”端。“描述”（Description）端的类和“运行时”（RunTime）的对象拥有一个树结构。将每一个元素分配给树中的一个明确位置。GdiCoDriverCD（见C.2.7）是“描述”端的根元素，而GdiCoDriverRT（见C.2.9）是“运行时”端的根元素。

工作区（“运行时”端）内功能对象的实例名称在所有使用的驱动上都应是独特而不重复的。

如果“运行时”端建立了一个对象，则应在设备驱动中隐式建立一个相应实例。当使用创建VD或FO的服务时，所有设置参数均应直接通过流式传输（见附录F）进行移交。

附录E描述的典型用例是为更好掌控服务并促进协调器软件的开发。

5.4.2 接口变量

本标准定义了三种接口。针对基于PID的配置，应包含：

—智能访问接口为必选项；

并包含：

—扩展访问接口为可选项；

针对基于服务或混合参数化的配置，应包含：

—完全访问接口为可选项。

从APSI的结构来看，这些不同的接口是相互依赖的。与智能访问接口相比，扩展访问接口内还拥有其他功能。对于每个接口变量（例如GdiCoFuncObjectRT）来说，单个对象的类和方法始终是相同的。一个应用程序可以使用所有应用的接口变量。创建工作区时，可以通过调整参数eRequiredCoordinatorInterface选择所需要的功能。

扩展访问接口除智能访问接口外只定义了GdiCoExtendedObjectNavigator对象（见B.2.6，其中包含用于浏览对象和数据的方法）。完全访问接口除了扩展访问接口外，还存在GdiCoObjectNavigator类（见C.2.18，它包含用于在对象和数据上导航的方法）和GdiCoFactory类（见C.2.11，其包含用于添加和删除对象的方法）。完全访问接口应能够更改（扩展、修改、删除）“运行时”端（实例化），并且还可以更改（扩展、修改）“数据库”端（描述）。在创建工区间后，不同接口能力应独立于配置类型（基于PID、基于服务或混合型）。

为了保持各个接口之间的兼容性，以下的方法包含在GdiCoWorkspace类中（见A.2.29）：

—gdiGetFactory

—gdiGetObjectNavigator和

—gdiGetExtendedObjectNavigator

它们将返回协调器的功能或NIL（若协调器不支持所需要的的共功能）给对象。

在本标准对接口的描述中，表1展示了类和方法对接口变量的从属关系与相应的标记。

表1 接口变量的类（无数据类型）

标记	代表含义
<<S,E,F>>	智能访问接口的类和方法。 该类和方法也可用于扩展访问接口和完全访问接口中。
<<E,F>>	扩展访问接口的类和方法。 该类和方法也可用于完全访问接口中。
<<F>>	完全访问接口的类和方法。 该类和方法也可用于只在完全访问接口中。

图4展示了文本标记和图形标记的对应关系。

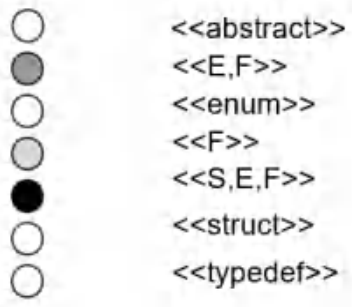


图4 文本标记和图形标记的对应关系

图5显示了所有接口变量的类。协调器应提供应用程序所请求的接口变量。有需要的话，智能访问接口应创建专门的工作区，以实现向完全访问接口的转化能力。使用PID对智能访问接口工作区进行配置后，应具备将此工作区转移到另一个使用完全访问接口的应用上，以增强另一个应用的能力有关详细信息，请参见5.5.4。

GdiCoManager::gdiGetCoordinatorType方法应返回一个位掩码，该位掩码将对接口变量的支持情况(<S>，<E>，<F>）做可视化。

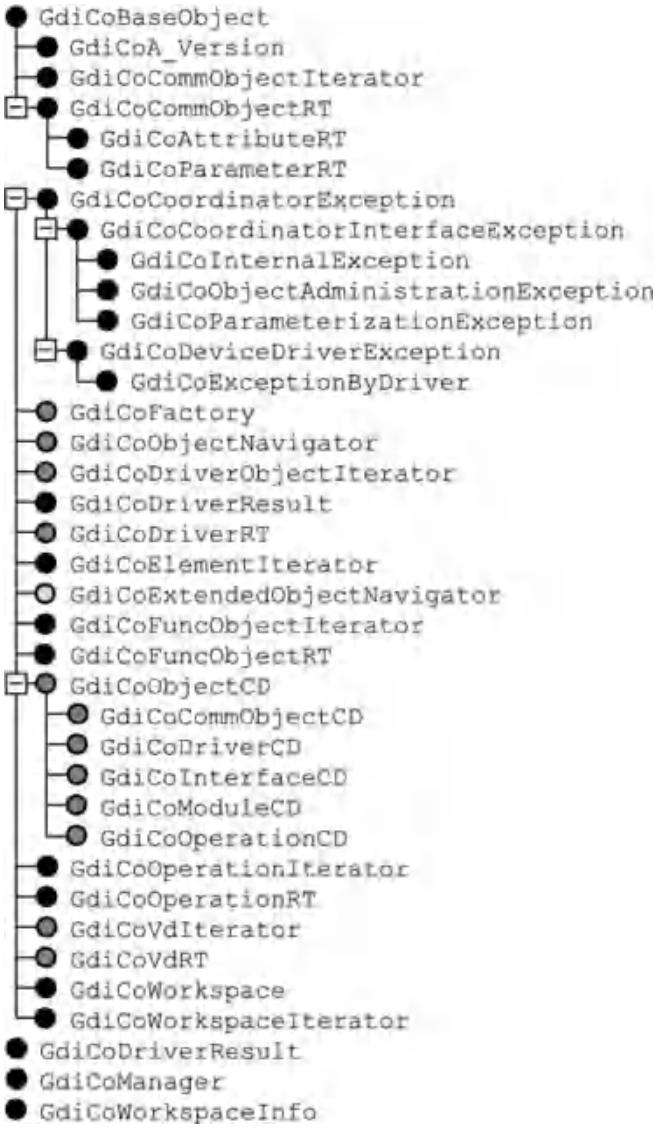


图5 接口变量的类（无数据类型）

具有已实现的完全访问接口的应用程序服务接口应支持附录C中所述的所有服务，并提供以下功能：

—在协调器内设置应用程序区域，并通过参数化数据或服务生成实例；

—获得对功能对象和通讯对象的引用；

—与协调器交换数据；

—在设备驱动内初始化数据传输；

—获取数据描述（类型和名称）；
—加载设备驱动；
—安装和控制虚拟设备
—生成实例（功能对象和通信对象）；
—设置数据元素（分配象征性名称和定义类型）；
—删除实例。

5.5 使用应用程序服务接口（APS）的详细说明

5.5.1 工作区结构

对于每个设备驱动和DCD来说，工作区中存在一个“描述”端和一个“实例化”端，他们具有树形结构。应用程序对单个对象的访问取决于所使用的接口变量。有关应用程序可以使用的类和对象的概述，请参见图5。“描述”一侧的类名称以CD结尾（Capability Description，能力描述），而“运行时”一侧的类名称以RT结尾。

可以使用对“描述”对象的引用、类名称（例如DCD中的接口名称）或类ID（DCD ID），将“运行时”对象实例化。协调器应提供以下服务来实例化“运行时”对象：

gdiCreateDriverRTByDescription(),
gdiCreateVdRTByDescription(),
gdiCreateFuncObjectRTByDescription(),
gdiCreateCommObjectByDescription()及
gdiCreateOperationRTByDescription() in GdiCoFactory (见C.2.11).

从“运行时”端可以通过“运行时”对象的方法访问相应的“描述”对象。即：

gdiGetDriverCD() in GdiCoDriverRT (见C.2.9), gdiGetModuleCD() in GdiCoVdRT (见C.2.29), gdiGetInterfaceCD() in GdiCoCommObjectCD (见C.2.4), gdiGetCommOjectCD() in GdiCoObjectNavigator (见C.2.18) 及
gdiGetOperationCD() in GdiCoObjectNavigator (见C.2.18)。

如果使用完全访问接口，则可以修改“描述”元素和“运行时”对象。

5.5.2 回调方法和引用

服务的异步执行为应用程序提供了一种处理同时触发流程的数据传输的简便方法，所以服务的异步执行应得到应用。

应通过回调方法实现异步执行。

通过GdiCoManager（见A.2.20）中gdiCreateWorkspace()的方法创建工作区后，refCBException、refCBAccept和refCBInformationReport这些参数应将回调应用带入应用程序内的方法中去，以处理基于事件的错误、设备主动数据的请求及来自设备的主动数据传输等事件。应使用空值（NIL）引用关闭回调。

在工作区的整个生命周期中，回调引用的全局部署是必要的，也应可在本地覆盖回调。

本地回调注册应通过以下方法完成：

—创建方法(VD、 FuncObjec、, ComObject), 及
—异步方法调用
(GdiCoCommObjectRT::gdiReadValue,GdiCoCommObjectRT::gdiWriteValue, GdiCoOperationRT::gdiExecuteOperation, GdiCoExtendedObjectNavigator::gdiReadCommObjectData, GdiCoExtendedObjectNavigator::gdiWriteCommObjectData 及 GdiCoExtendedObjectNavigator::gdiExecuteOperation)。

将本地回调引用随此注册一起移交，并临时覆盖各自应用程序的全局回调引用（由其句柄进行标识）。如果需注册更多的应用程序（见数据监视5.5.5），则回调应分配给所有已注册的应用程序。需要注销时应提供回调引用空值（NIL）。应有可能通过多个注册来更改“信息报告”（InformationReport）和“接受”（Accept）的本地引用。

使用gdiReleaseWorkspace或gdiReleaseMonitorAccess时，已注册的回调将失效。如果应用程序连接到一个现有工作区，则应重新注册回调引用。

这同样适用于“信息报告”（InformationReport）和“接受”（Accept）。全局回调引用应在创建工作区时设置，而本地回调引用应在GdiCoAttributeRT（gdiSetLocalCBAccept、gdiSetLocalCBInfReport）处设置。

5.5.3 工作区扩展

无论当前使用的接口变量如何设置，都应有通过应用（新的）PID文件的方法扩展现有配置的能力。

上述操作应遵循下列规则：

—现有配置应被保留；

—不应删除任何现有信息；

—配置元素仅能被增加而不能被删除；

—PID文件必须属于同一个设备。

首先，应根据ISO 20242-4的定义检查PID文件的语法和语义是否正确。

在下列配置检查中,PID应与工作区已经存在的配置一致。新的PID文件应与已经存在的对象做适配性测试，对比两者对象（OP,CO，FO）的实例名称和设备能力描述类。对于现有对象来说，对象和父对象（FO和/或VD）的应用程序引用名称应与DCD类名称一致。如果存在任何应用程序引用名称尚未实例化的CO或OP，并且父级（FO/VD）的应用程序引用名称和DCD类名相同，则应重新实例化。已经存在的VD和FO的任何创建参数都应被忽略。

如果“通信对象”（CO）和/或“操作”（OP）同类名称的应用程序引用名称不一致，则将会引发错误异常（见5.6）。如果“功能对象”（FO）相同类名的应用程序引用名称不一致，但父对象具有“虚拟设备”（VD）的相同应用程序引用名称和设备能力描述（DCD）类名，则应将该“功能对象”（FO）添加进来。

如果新的PID文档保存了现有CO的值，则应将这些CO写入以进行值更新。现有“功能对象”（FO）的创建参数通常不应进行更新。

所有其他“通信对象”（CO）均不应进行修改。另外，不应设置任何隐式默认值（例如，最小值）。

如果新的“参数化实例描述”（PID）文档标记了操作执行，则应予以执行。

当gdiExtendWorkspace方法完成时，相应VD的状态应处于工作状态（请参阅ISO 20242-4）。

如果为给定的PID文档注册了与一致性相关的任何错误，则过程停止。所有VD相关的工作区保持VD“准备”状态。 如果PID文档在运行时发生任何错误，则将执行相同的操作。

如果发生错误，则过程停止。所有相关的VD都应处于“准备”状态（见ISO 20242-4）。如果语法和语义检查出现了错误，则gdiExtendWorkspace方法将引发异常：

eINT_PID_FILE_MISMATCH (GdiCoInternalException, 见A.2.19)。

如果其运用新的配置，方法将引发异常：

ePAR_INCORRECT_PARAMETERIZATION (GdiCoParameterizationException, 见A.2.24)。

5.5.4 工作区转移

不同应用程序应可以连续接替使用同一工作区。当应用程序通过gdiReleaseWorkspace方法（见A.2.20.11）与工作区断连时，另一个应用程序可以使用工作区名称和gdiGetWorkspaceByName方法（请参阅A.2.20.7）连接到被释放的工作区中去。

注：工作区转移使APSI用户能够将大型应用程序可拆分为占用更少资源的较小而独立的模块。示例包括用于基本配置、扩展配置、维护设备和运行自动化的模块。

由于不同应用程序对工作区有顺序性的使用，始终会存在一个没有连接任何应用程序的暂短时间。当执行gdiReleaseWorkspace()时，协调器应将所有连接的“虚拟设备”（VD）设置为“准备“（Preparation）”状态，然后释放工作区。如果错误阻碍了向“准备”状态的过渡，协调器应拒绝释放工作区。

如果使用“智能访问接口”（SAI）或“扩展访问接口”（EAI）（eRequiredCoordinatorInterface参数中的eSMART或eEXTENDED），“虚拟设备”（VD）的gdiGetWorkspaceByName()应设置为“工作”状态。如果使用FAI，则应保持“准备”状态。

应通过GdiWorkspaceIterator（见A.2.31）和GdiCoWorkspaceInfo（见A.2.30）为任何应用程序提供工作区的状态。

释放工作区后,所有引用将失效。任何具有无效句柄的访问均应触发异常eINT_INVALID_ACCESS（见A.2.19）。

5.5.5 数据监控

如果应用程序通过gdiGetMonitorAccessByWorkspaceName连接到工作区（见A.2.20.6），则它应仅具有读取数据访问的权限（gdiGetDataValue,见A.2.9.2和通过refCBInformationReport中回调的InformationReport，见A.2.20.6），并且不会影响工作区的状态（eNOT_USED、eUSED）。此应用程序称为监控器应用程序。

一个监视器应用程序应有权访问“运行时”端的现有树以及通信参数和操作参数的数据类型，但无权对“数据库”端进行访问。任何试图通过监视器应用程序修改状态和配置的操作都应被拒绝，并应使用“eOAD_OBJECT_ACCESS”错误代码标识。

对于数据访问问题，例如由主应用程序访问所引起的问题，应使用eOAD_DATAINUSE_OR_INCONSISTENT 错误代码进行标识。

异常应在监视器应用程序调用方法时就以错误标识的形式抛出，并应可用gdiReleaseMonitorAccess方法释放连接。

通信对象的InformationReports只应发送到那些处理主要应用程序的对象的监视器应用程序中去。发送事件的对象管理应特定于主要应用。因此，监视器应用程序只应接收针对以下对象的信息报告：

—具有来自对象树中监视器应用程序的有效回调，并且

—主应用程序已经为那些对象启用了“信息报告”（InformationReport）。

如果主应用程序已成功处理一个属性的读写，则会从协调器向监视器发送一个附加的信息报告。

当使用gdicoattlert:：gdiSetLocalCBInfReport方法来定义回调时，应覆盖与属性有关的信息报告的全局回调引用（gdiGetMonitorAccessByWorkspaceName的refCBInformationReport中的参数）。

监视器应用的连接数量不应被限制。且若使用主要应用程序完成工作区转移，监控器应用程序也应保持有效。

如果主应用程序希望在连接监视器应用程序时通过gdideletespace删除工作区， 该操作将被拒绝，并抛出eINT_INVALID_ACCESS的错误代号。

5.5.6 数据类型

包括FO引用在内的所有已定义的数据类型都应被支持。在“描述”一侧，数据类型描述存储在DCD中。在“运行时”一侧，应存在与定义类型相对应的数据对象。复杂类型应在树结构中逐步创建。

应用程序与设备驱动之间的数据交换应由协调器通过提供与数据类型描述相对应的格式化和分配式的通信存储器进行协调。数据交换应分两步完成：

1.数据从源（应用程序或驱动）复制到抽象数据汇的通信内存中。

2.数据目标对象从该通信内存中读取数据。

APSI应在协调器内提供用于写入和读取通信存储器的服务。

gdiGet ... Value和gdiSet ... Value（例如B.2.4.1）这些方法应通过流技术来处理对协调器通信存储器的格式化访问，而无需在设备驱动程序上启动任何活动。 此外，每个复杂数据类型的子元素都应分别（逐步）设置子元素值。

通信存储器更新后，应该使用GdiCoExtendedObjectNavigator::gdiReadCommObjectData 及 GdiCoExtendedObjectNavigator::gdiWriteCommObjectData开始与设备驱动（通过GDI_Read和GDI_Write）的数据交换。

或者，GdiCoCommObject接口应该仅使用一个gdiReadValue或gdiWriteValue方法来执行这两个通信步骤（即使用应用程序数据更新通信存储器，并与设备驱动进行数据交换）。图6概述了APSI的数据类型处理。

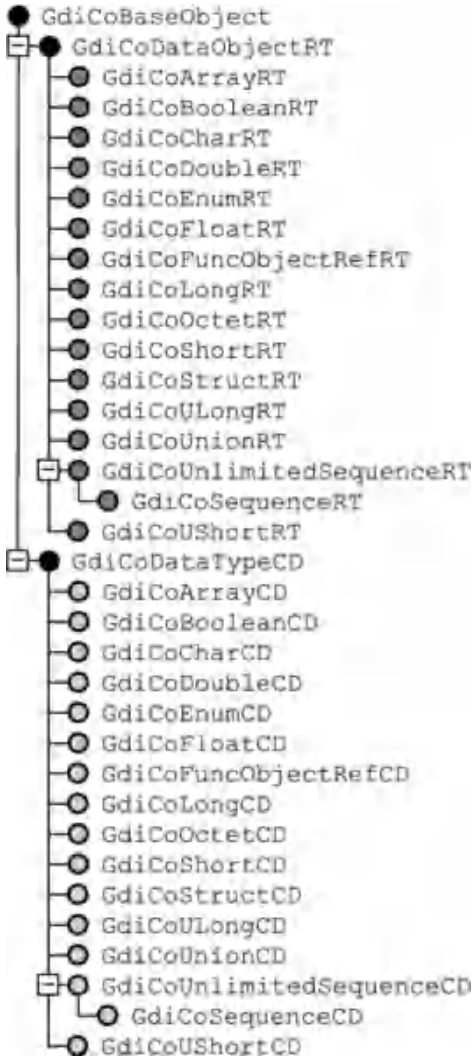


图6 数据类型类和实例化数据对象

5.5.7 流

为使形式化应用程序也能够访问复杂的数据类型，应在协调器与应用程序之间使用流技术进行数据交换。“设备能力描述 ”（DCD）中定义的设备驱动特定数据类型对于应用程序应是不可视的。

流对象应该用于应用程序与协调器之间的数据交换。该流应是一个字节序列的BLOB（二进制大对象）。 应根据DCD中的类型说明设置流（见ISO 20242-4）。BLOB流应满足各个数据交换所约定的对齐方式和字节顺序（见附录I中的协调器数据表）。

应使用非典型流，即不应提前也不应单独传输类描述或将其包含在内。

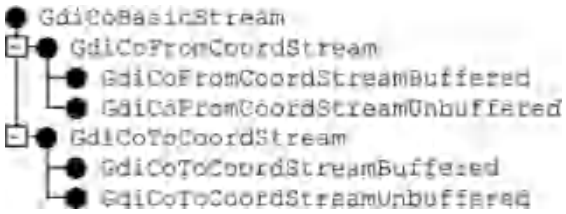


图7 针对流的对象

图7显示了为APSI中的流而定义的对象。一个流对象应允许应用程序与协调器之间基于流的数据交换。可能的实现方式是，其一，流对象首先建立两个点（数据源和汇）之间的连接，然后在这两个点之间提供无缓冲的数据交换；其二，提供一个容器，从而可以将数据填充到包数据中去并将包数据移传输至任何目的地。收件人可以从包数据中提取数据以做进一步处理。方式二一个有缓冲的数据交换。所以本标准定义的流式传输既可以是有缓冲的，也可以是无缓冲的。

流对象应使能应用程序与协调器之间的平台进行独立的数据交换。因此，应用在使用数据方面不受很多限制。

APSI定义了两种流对象（见A.2.14和A.2.26）：

—GdiCoFromCoordStream 及

—GdiCoToCoordStream,

这些均派生于GdiCoBasicStream。

还有其他对象可以区分缓冲数据交换和非缓冲数据交换（请参阅A.2.15，A.2.16和A.2.27，A.2.28）。

只有在应用程序和协调器之间的对齐方式和字节顺序相同时，无缓冲流才有效。如果协调器和应用程序具有不同的对齐方式和/或字节顺序，则应使用缓冲数据交换。

流对象可能会包含非常不同的对其方式和字节顺序（见附录G）。流对象中序列化数据的顺序由应用程序的类型决定。协调器必须通过传递所提交的流对象中有关数据顺序的信息以适应此顺序。这些上下文参数是构造流对象的参数。这些参数放在各自引用实现的两个常量cAppAlignment和cAppByteOrder(默认为文本)之前。使用流传输时，应解码使用的对齐方式和字节顺序。附录F中进一步定义了流对象的应用程序规则。

5.6 错误处理

应有异常报错,且应对设备驱动错误(包括设备异常)和协调器错误做好区分(前者由协调器传递给应用)。设备驱动错误不应反映在协调器错误中（见图8）。

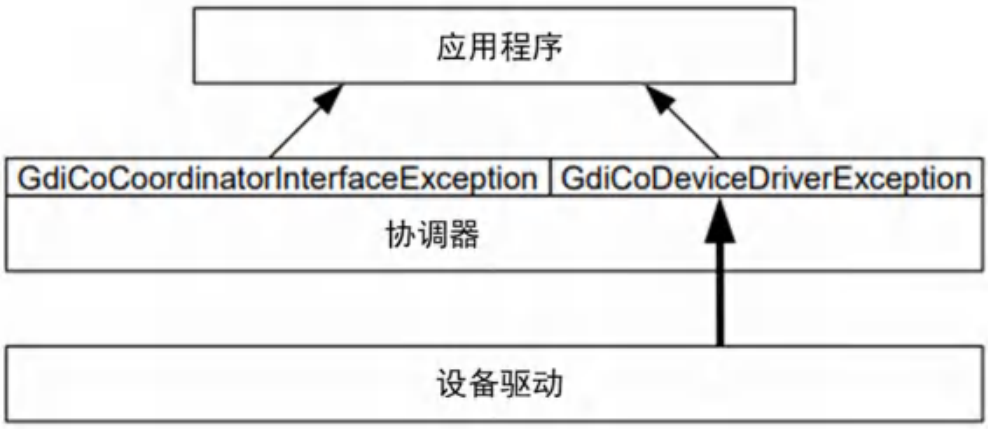


图8 协调器和设备驱动错误源

细节请详见A.2.7到A.2.10。

协调器错误有以下几种情况：

—参数化错误(PID文件错误),

—对象管理错误（APSI的错误使用），及

—内部错误（问题可能出在协调器、运行系统、版本、禁止访问监视器上）。

“应用程序服务接口”（APSI）的实现额外包含内部的数据检查功能具备对实际资源或内部限制的核验能力。本标准尚未在APSI中，对相应的能力支持做定义。但是，如果协调器检测到这种内部错误，应使用错误代码eINT_PRACTICAL_DATA_OUT_OF_RANGE。

附 录 A
（规范性）
编程参考指南—智能访问接口

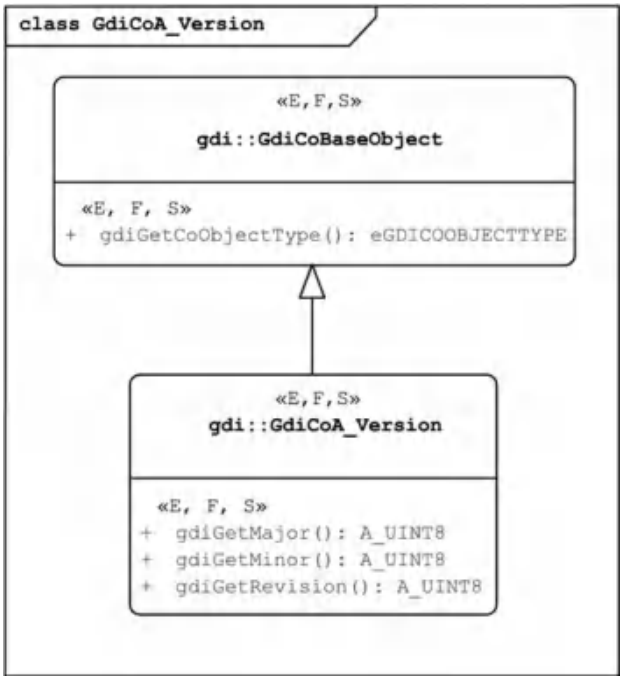
A.1 总则

智能访问接口（标记为«S»）的单一接口（类及其方法和属性）也应属于扩展访问接口（标记为«E»）和完整访问接口（标记为«F»）。因此，所有单一接口均用«S，E，F»标记。类及其方法的概述详见图A.1到A.31。

A.2 小型访问接口详细说明

A.2.1.1«S,E,F» GdiCoA_Version

此接口包含应用程序服务接口的版本信息。



图A.1 GdiCoA_Version层次图

A.2.1.2GdiCoA_Version :: «S,E,F» gdiGetMajor()

函数调用

```
gdiGetMajor
        (
        ) : A_UINT8。
```

函数说明

返回主版本号。

返回值

主版本号。

A.2.1.3GdiCoA_Version :: «S,E,F» gdiGetMinor()

函数调用

```
gdiGetMinor
        (
        ) : A_UINT8
```

函数说明

返回次要版本号。

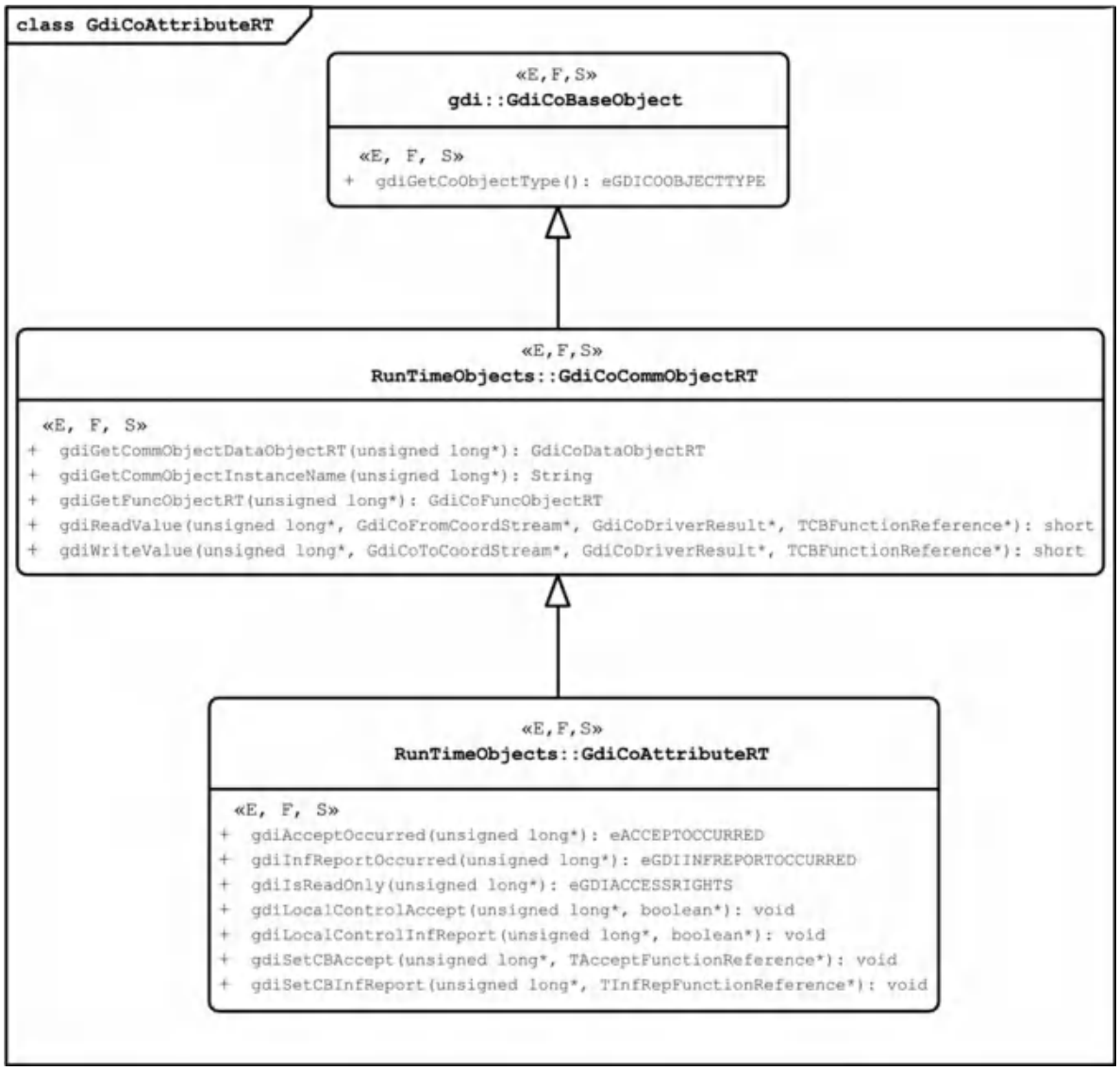
返回值
返回次要版本号。

A.2.1.4GdiCoA_Version :: «S,E,F» gdiGetRevision()

函数调用
gdiGetRevision
(
) : A_UINT8
函数说明
返回修订号。
返回值
返回修订号。

A.2.2«S,E,F» GdiCoAttributeRT

此接口包含用于处理属性的服务。这些通信对象是通过参数来区分的，这样属性也可以通过驱动程序发起的InformationReport和Accept来交换数据。



图A.2　GdiCoAttributeRT层次图

A.2.2.1 GdiCoAttributeRT :: «S,E,F» gdiAcceptOccurred()

函数调用

```
gdiAcceptOccurred
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):eACCEPTOCCURRED
```

函数说明

若执行上一次测试或gdiWriteValue的结果为Accept，则返回相应信息。若应用程序不支持回调，则使用该服务。

返回值

若未发生 Accept，返回 eNOACCEPT(0)。

若发生 Accept，返回 eACCEPT(1)。

A.2.2.2 GdiCoAttributeRT :: «S,E,F» gdiInfReportOccurred()

函数调用

```
gdiInfReportOccurred
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):eGDIINFREPORTOCCURRED
```

函数说明

若执行上一次测试或 gdiReadValue 的结果为 InformationReport，则返回相应信息。若应用程序不支持回调，则使用该服务。

返回值

若未发生 InormationReport，返回 eNOINFREPORT (0)。

若发生 InormationReport，返回 eINFREPORT (1)。

A.2.2.3 GdiCoAttributeRT :: «S,E,F» gdiIsReadOnly()

函数调用

```
gdiIsReadOnly
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):eGDIACCESSRIGHTS
```

函数说明

如果通信对象是只读的，则返回相应信息。

返回值

通信对象支持读写，返回 eREADWRITE (0)。

通信对象是只读的，返回 eREADONLY(1)。

A.2.2.4 GdiCoAttributeRT :: «S,E,F» gdiLocalControlAccept()

函数调用

```
gdiLocalControlAccept
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
```

```
            inout                   booleanbEnableControl
            /*inparameter
            true=enableAcceptCallbacks
            false=disableAcceptCallbacks*/
            ):void
```

函数说明

启用/禁用此通信对象的“接受回调”。

返回值

无

A.2.2.5 GdiCoAttributeRT :: «S,E,F» gdiLocalControlInfReport()

函数调用

```
gdiLocalControlInfReport
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   booleanbEnableControl
            /*inparameter
            true=enableInformationReport
            false=disableInformationReport*/
            ):void
```

函数说明

启用/禁用此属性的 InformationReport 回调。

如果在 GdiCoAttributRT 上启用了 InformationReport 标志，并且设备驱动程序不支持该属性的信息报告，则只要该属性被另一个客户端更改，协调器就应将 InformationReport 发送给监视客户端。

如果设备驱动程序向该属性发送一个 InformationReport，则该信息应传递给所有启用 InformationReport 标志的监视客户端。

返回值

无

A.2.2.6 GdiCoAttributeRT :: «S,E,F» gdiSetCBAccept()

函数调用

```
gdiSetCBAccept
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   TAcceptFunctionReferencerefCBAccept
            /*inparameter
            Referencetoacallbackmethodwhichhandlestheacceptmechanism.A
            NILreferencesignalsthatnocallbackissupportedortheglobal callbackreferenceistouse.
            Atthecoordinatorcallbacktotheapplication,anreferencetothe
            specificGdiCoCommObjectRTispassed.*/
            ):void
```

函数说明

为此通信对象注册一个本地 Accept 回调。全局回调（由 gdiCreateWorkspace 提供）应停用。

返回值

无

A.2.2.7 GdiCoAttributeRT :: «S,E,F» gdiSetCBInfReport()

函数调用

```
gdiSetCBInfReport
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   TInfRepFunctionReferencerefCBInformationReport
        /*inparameter
        Referencetoacallbackmethodwhichhandlestheinformationreport
        mechanism.ANILreferencesignalsthatnocallbackissupportedorthe
        globalcallbackreferenceistouse.
        Atthecoordinatorcallbacktotheapplication,anreferencetothe
        specificGdiCoCommObjectRTispassed.*/
        ):void
```

函数说明

为此通信对象注册本地 InformationReport 回调。全局回调（由 gdiCreateWorkspace 提供）应停用。

如果在 GdiCoAttributRT 上启用了 InformationReport 标志，并且设备驱动程序不支持该属性的信息报告，则只要该属性被另一个客户端更改，协调器就应将 InformationReport 发送给监视客户端。

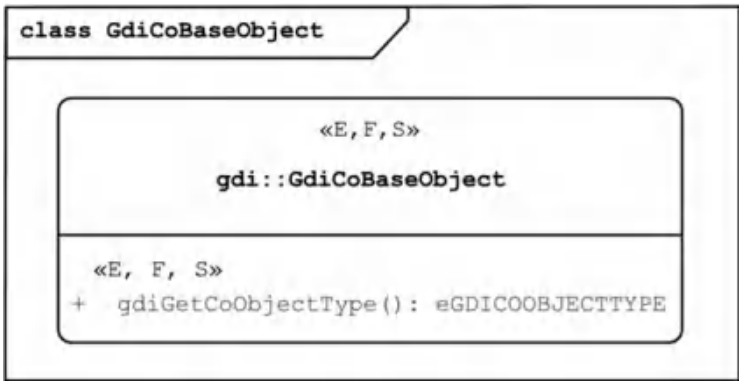
如果设备驱动程序将 InformationReport 发送到该属性,则该信息将传递给启用了 InformationReport 标志的所有监视客户端。

返回值

无

A.2.3 «S,E,F» GdiCoBaseObject

此接口是所有协调器对象的基本接口。此服务仅在继承中发生。



图A.3 GdiCoBaseObject层次图

A.2.3.1 GdiCoBaseObject :: «S,E,F» gdiGetCoObjectType()

函数调用

```
gdiGetCoObjectType
        (
        ):eGDICOOBJECTTYPE
```

函数说明

返回检测协调器对象引用类型的枚举。

返回值

返回 GDICOOBJECTTYPE 值.

A.2.4 «S,E,F» GdiCoBasicStream

此接口是流对象的基本接口。这些服务仅在继承中发生。数据被复制到所有客户端以供监视器访问。

图A.4 GdiCoBasicStream层次图

A.2.4.1 GdiCoBasicStream :: «S,E,F» gdiGetStreamAllignment()

函数调用

```
gdiGetStreamAllignment
          (
          ):short
```

函数说明

返回流对象中数据的对齐方式。

返回值

返回流对象中数据的对齐方式。

A.2.4.2 GdiCoBasicStream :: «S,E,F» gdiGetStreamByteOrder()

函数调用

```
gdiGetStreamByteOrder
          (
          ):eGDIBYTEORDER
```

函数说明

返回流中数据的 ByteOrder。

返回值

返回流中数据的 ByteOrder。

A.2.4.3 GdiCoBasicStream :: «S,E,F» gdiIsStreamEmpty()

函数调用

```
gdiIsStreamEmpty
          (
          ):bool
```

函数说明

此函数用于检测流中数据与应用程序中数据的类型兼容性。

如果在通信后返回值为 true，则该类型大概率已实现准确转译。

(类似于“文件结束”标志)

返回值

true = 流为空。

false = 流不为空。

A.2.4.4 GdiCoBasicStream :: «S,E,F» gdiSeekStream()

函数调用

```
gdiSeekStream
        (
        inout                   eORIGINeOriginator
        /*inparameter
        containstheinitialposition.*/
        inout                   longlPos
        /*inparameter
        containsthepositioninwhichbytetomovethestreampointerfrom origin.*/
        ):short
```

函数说明

将流中的流指针移动到特定位置。

返回值

0 ： 成功

!= 0 ： 错误

A.2.4.5 GdiCoBasicStream :: «S,E,F» gdiTellStreamPos()

函数调用

```
gdiTellStreamPos
        (
        ):unsignedlong
```

函数说明

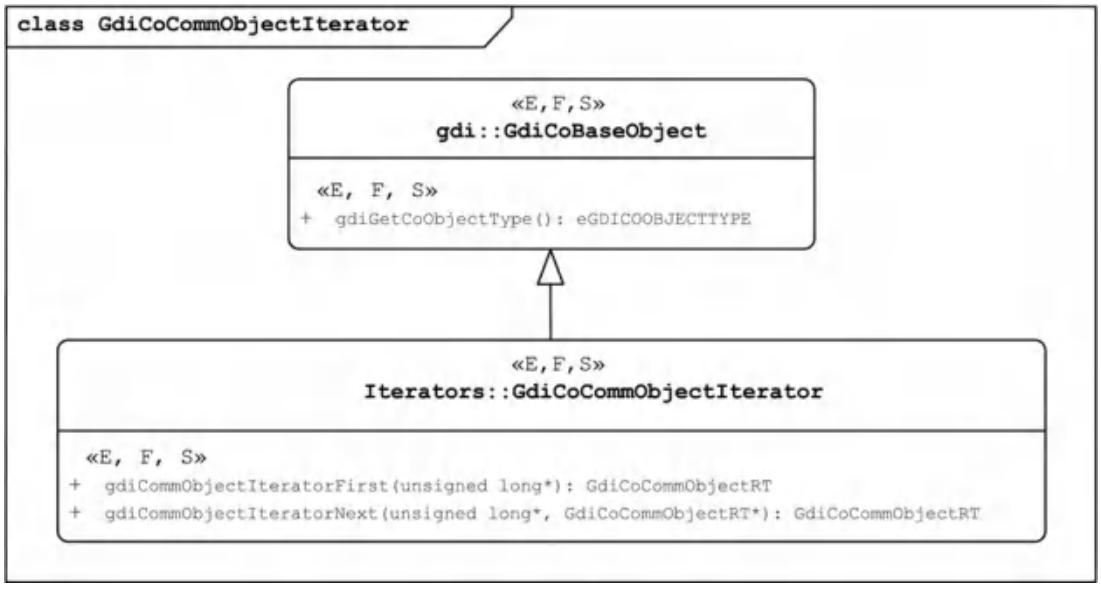
报告在输出流中的当前位置。

返回值

返回在流中的当前位置。

A.2.5 «S,E,F» GdiCoCommObjectIterator

此接口包含一个 CommObject 迭代器的应用程序服务，它是在函数对象中设置的所有通信对象的顺序输出。



图A.5 GdiCoCommObjectIterator层次图

A.2.5.1 GdiCoCommObjectIterator :: «S,E,F» gdiCommObjectIteratorFirst()

函数调用

```
gdiCommObjectIteratorFirst
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
```

```
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoCommObjectRT
```

函数说明

设置指向工作区内第一个 ComObject 的内部指针。

返回值

返回对第一个 GdiCoCommObject 的引用（如果存在），否则返回 NIL 引用。

A.2.5.2 GdiCoCommObjectIterator :: «S,E,F» gdiCommObjectIteratorNext()

函数调用

```
gdiCommObjectIteratorNext
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoCommObjectRTrefLastElement
            /*inparameter
            A reference to a element returned by gdiComObjectIteratorFirst or
            gdiComObjectIteratorNext.*/
            ):GdiCoCommObjectRT
```

函数说明

检测对功能对象中当前 ComObject（参数）的引用。将内部指针设置为 FO 中的下一个 ComObject（如果存在）并返回引用。

返回值

返回对协调器 CommObject（协调器内的管理单元）的引用。

A.2.6 «S,E,F» GdiCoCommObjectRT

该接口包含使用通信对象的服务。 这些服务仅发生在继承中。

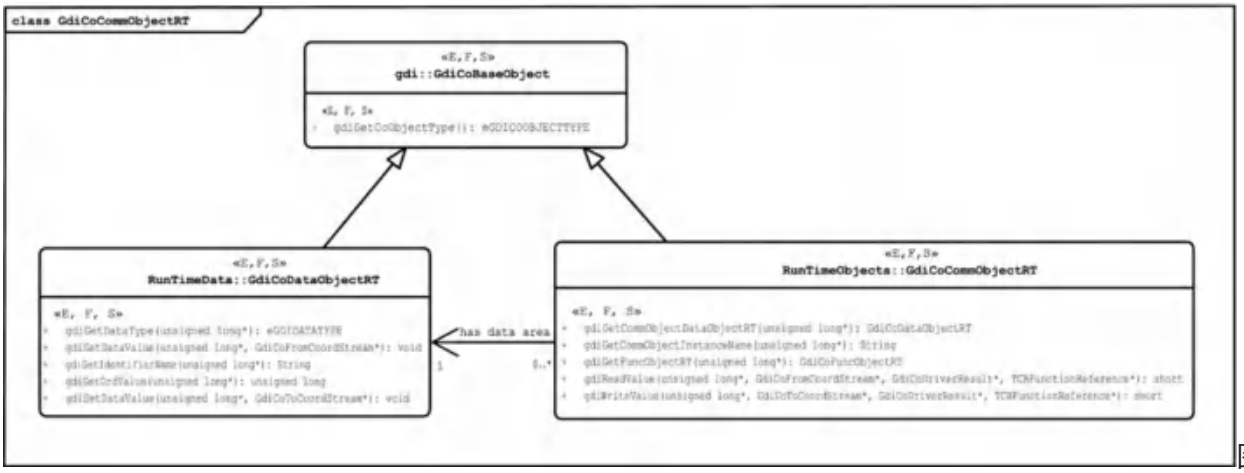


图 A.6 GdiCoCommObjectRT 层次图

A.2.6.1GdiCoCommObjectRT :: «S,E,F» gdiGetCommObjectDataObjectRT()

函数调用

```
gdiGetCommObjectDataObjectRT (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoDataObjectRT
```

函数说明

返回对给定通信对象的相应数据对象的引用。

返回值

返回对相应数据对象的引用

A.2.6.2 GdiCoCommObjectRT :: «S,E,F» gdiGetCommObjectInstanceName()

函数调用

```
gdiGetCommObjectInstanceName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

函数说明

返回参数化期间定义的对象的实例名称。

ComObject 实例仅可存在一个，因此接口内的属性/参数名称和对象名称可以相同。

返回值

返回对象的名称。

A.2.6.3 GdiCoCommObjectRT :: «S,E,F» gdiGetFuncObjectRT()

函数调用

```
gdiGetFuncObjectRT (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoFuncObjectRT
```

函数说明

创建对现有函数对象的引用。

被引用的函数对象包含此通信对象（用于评估与信息报告传递的通信对象相对应的函数对象）。

返回值

如果成功，则返回对 GdiCoFuncObjectRT 的引用。

A.2.6.4 GdiCoCommObjectRT :: «S,E,F» gdiReadValue()

函数调用

```
gdiReadValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoFromCoordStreamrefDestValue
          /*outparameter
          Referencetoastreamobject,thetype isidenticaltothetype ofthe
          Communicationobject.*/
          inout                  GdiCoDriverResultrefDestInfo
          /*outparameter
          contains the reference to the Information Object, warnings and
          informationsoftheDriverwillbestoredinit.*/
          inout                  TCBFunctionReferencerefCBCompleteRead
          /*inparameter
          Referencetoacallbackmethodwhichhandlesthereadmechanism.ANIL
```

```
            referencesignalsasynchronouscall.*/
            ):short
```

函数说明

从驱动程序中读取通信对象，并通过引用返回一个数据流。流中的数据是一个字节序列。

返回值

描述由 GDI 驱动程序执行的通信：

0 = 同步返回；

1 = 异步返回。

A.2.6.5 GdiCoCommObjectRT :: «S,E,F» gdiWriteValue()

函数调用

```
gdiWriteValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoToCoordStreamrefSourceValue
            /*inparameter
            Referencetoastreamobject,thetype isidenticaltothetype ofthe
            Communicationobject.*/
            inout                   GdiCoDriverResultrefDestInfo
            /*outparameter
            contains the reference to the Information Object, warnings and
            informationsoftheDriverwillbestoredinit.*/
            inout                   TCBFunctionReferencerefCBCompleteWrite
            /*inparameter
            Referencetoacallbackmethodwhichhandlesthewritemechanism.ANIL
            referencesignalssynchronouscall.*/
            ):short
```

函数说明

将通信对象的数据值（通过引用）写入协调器，并启动向驱动程序的传输。

返回值

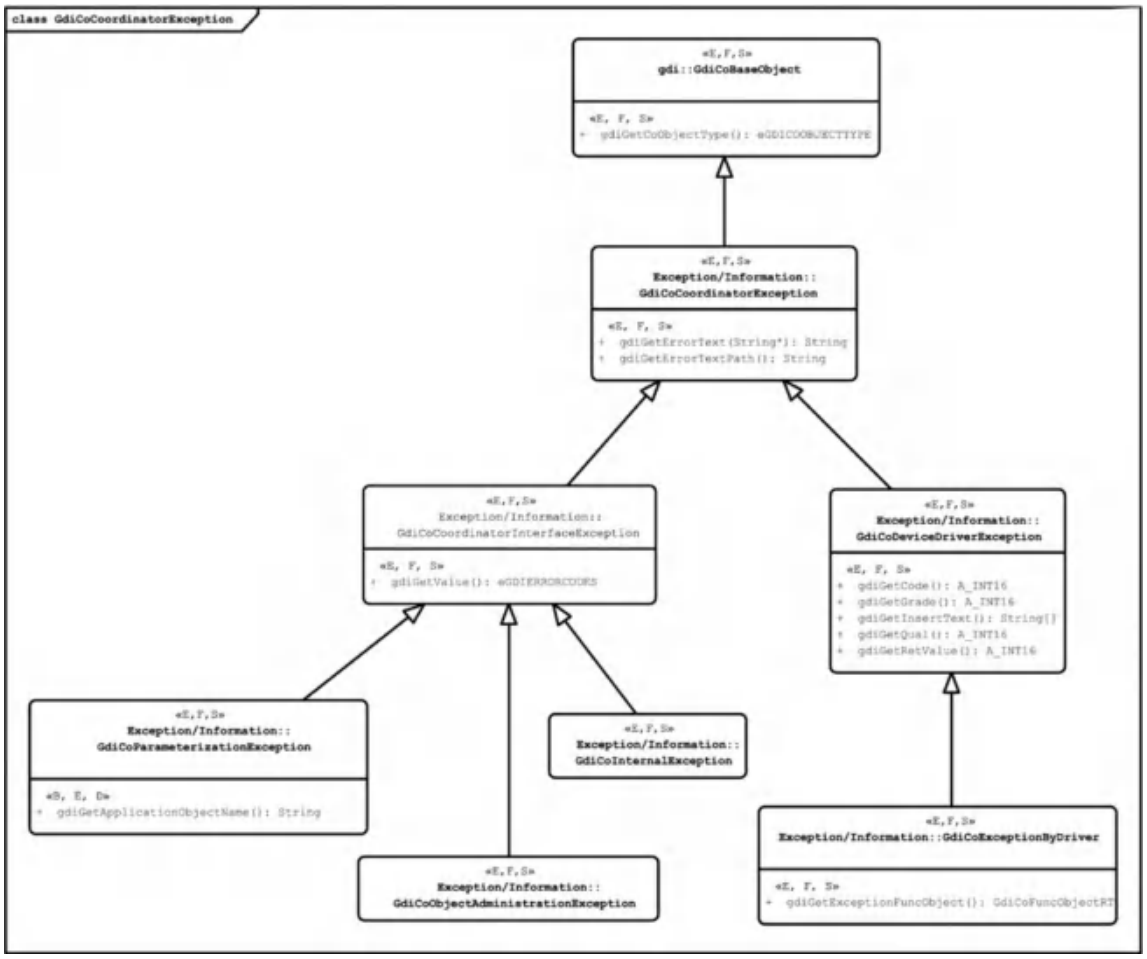
描述由 GDI 驱动程序执行的通信：

0 = 同步返回；

1 = 异步返回。

A.2.7 «S,E,F» GdiCoCoordinatorException

这是基本错误类，应用于错误处理。关于协调器内部错误和设备驱动程序错误的单独处理，另请参见 5.6。



图A.7　GdiCoCoordinatorException层次图

A.2.7.1 GdiCoCoordinatorException :: «S,E,F» gdiGetErrorText()

函数调用

```
gdiGetErrorText
        (
        inout                   StringstrLanguage
        /*inparameter
        requestedlanguagetokenaccordingtoISO 639.*/
        ):String
```

函数说明

以相应的语言返回相应的字符串，包括来自标准 DIT 的替换文本（如果存在）。
否则，返回一个空字符串。

返回值

以相应的语言返回相应的字符串，包括来自标准 DIT 的替换文本（如果存在）。
否则，返回一个空字符串。

A.2.7.2 GdiCoCoordinatorException :: «S,E,F» gdiGetErrorTextPath()

函数调用

```
gdiGetErrorTextPath
        (
        ):String
```

函数说明

将路径引用以 DIT 或 DII 的字符串返回到相应的错误消息。
此路径可用于向外部 DIT/DII 错误服务器请求特定语言的错误消息。

返回值

将路径引用以 DIT 或 DII 的字符串返回到相应的错误消息。
此路径可用于向外部 DIT/DII 错误服务器请求特定语言的错误消息。

A.2.8 «S,E,F» GdiCoCoordinatorInterfaceException

如果发生协调器特定的错误，则应抛出此异常类。错误的详细信息应通过子类进行描述。

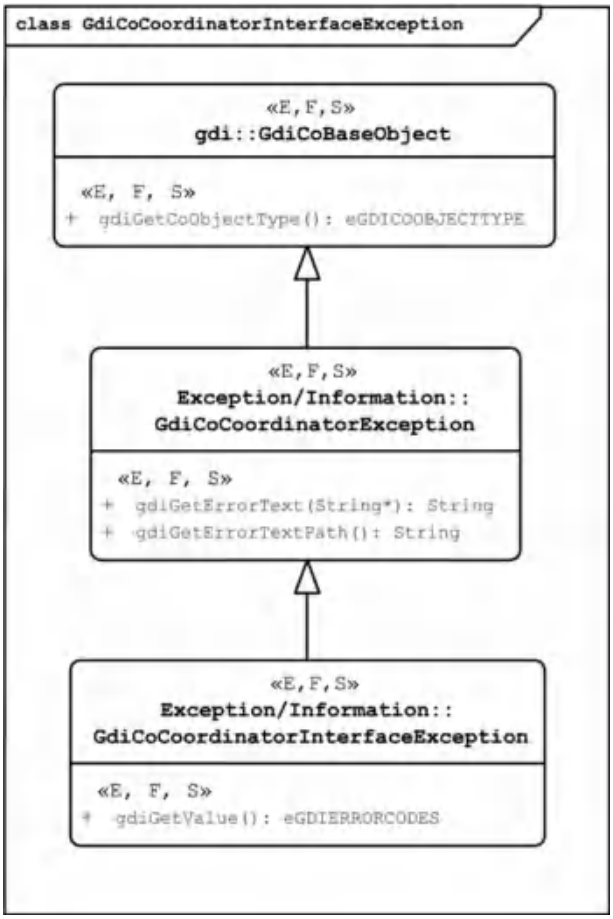


图 A.8 GdiCoCoordinatorInterfaceException层次图

A.2.8.1 GdiCoCoordinatorInterfaceException :: «S,E,F» gdiGetValue()

函数调用

```
gdiGetValue
        (
        ):eGDIERRORCODES
```

函数说明

以数字形式返回错误值。
详细信息见 eGDIERRORCODES 枚举。

返回值

返回 eGDIERRORCODES 枚举中定义的错误代码。

A.2.8.2 GdiCoDataObjectRT :: «S,E,F» gdiGetOrdValue()

函数调用

```
gdiGetOrdValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
```

```
):unsignedlong
```

函数说明

返回结构体或联合体中类型（子类型）的内部位置值。在联合体中，该服务用于检测子类型的开关值。在结构体内，该值仅作为信息使用。如果这个类型是序列或数组中的子类型，则此服务返回值为0。

返回值

返回此类型的内部位置作为结构体或联合体中的子类型。

A.2.8.3 GdiCoDataObjectRT :: «S,E,F» gdiSetDataValue()

函数调用

```
gdiSetDataValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoToCoordStreamrefSourceDataStream
        /*inparameter
        ContainsthereferencetothesourceDataStream.*/
        ):void
```

函数说明

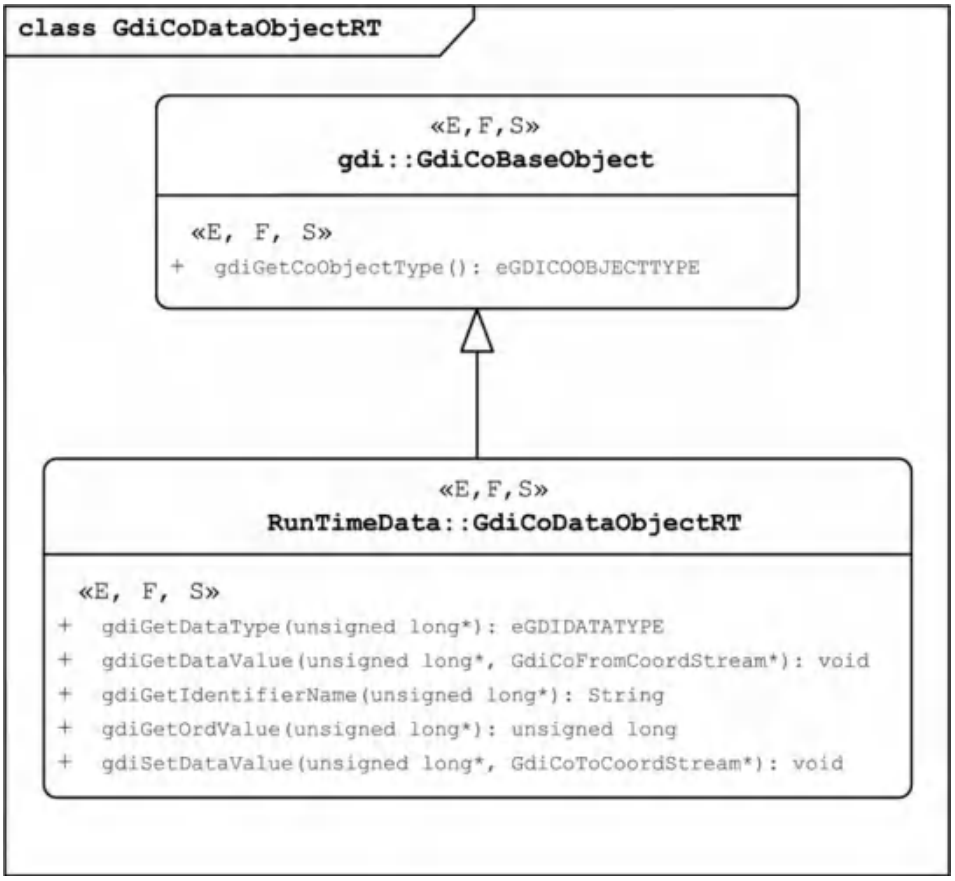
使用流引用传递的值设置协调器中数据区域的值。数据作为一个字节块进行处理。字节数由基准类型决定。

返回值

无

A.2.9 «S,E,F» GdiCoDataObjectRT

此接口包含用于访问协调器内数据的服务。



图A.9 GdiCoDataObjectRT层次图

A.2.9.1 GdiCoDataObjectRT :: «S,E,F» gdiGetDataType()

函数调用

```
gdiGetDataType
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):eGDIDATATYPE
```

函数说明

以数值（枚举）形式返回类型。

返回值

以数值形式返回类型（在语义中描述）。

A.2.9.2 GdiCoDataObjectRT :: «S,E,F» gdiGetDataValue()

函数调用

```
gdiGetDataValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoFromCoordStreamrefDestDataStream
          /*inparameter
          ContainsthereferencetothedestinationDataStreamintheapplication.
          */
          ):void
```

函数说明

返回协调器中数据区域的值，该值使用流引用传递的值设置。数据作为一个字节块进行处理。字节数由基准类型决定。

返回值

无

A.2.9.3 GdiCoDataObjectRT :: «S,E,F» gdiGetIdentifierName()

函数调用

```
gdiGetIdentifierName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

函数说明

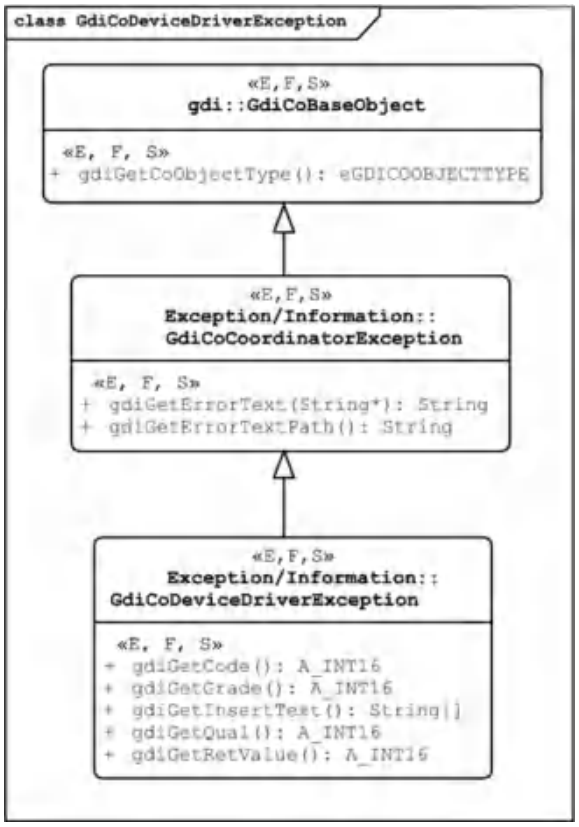
返回类型的名称。用于标识派生类型或子类型。

返回值

返回一个文本字符串（指针、类、sz 区域的地址）。

A.2.10 «S,E,F» GdiCoDeviceDriverException

此函数用于抛出设备驱动程序错误引起的异常。



图A.10 GdiCoDeviceDriverException层次图

A.2.10.1 GdiCoDeviceDriverException :: «S,E,F» gdiGetCode()

函数调用

gdiGetCode
(
):A_INT16

函数说明

返回 GDI API 错误的相应值。

返回值

返回 gdi 设备驱动程序 api 函数的错误代码。

A.2.10.2 GdiCoDeviceDriverException :: «S,E,F» gdiGetGrade()

函数调用

gdiGetGrade
(
):A_INT16

函数说明

返回 GDI API 错误的相应值。

返回值

返回 gdi 设备驱动程序 api 函数的错误代码。

A.2.10.3 GdiCoDeviceDriverException :: «S,E,F» gdiGetInsertText()

函数调用

gdiGetInsertText
(
):String[]

函数说明

以字符串列表形式从 GDI 驱动程序返回替换文本。

返回值

以字符串列表形式从 GDI 驱动程序返回替换文本。

A.2.10.4 GdiCoDeviceDriverException :: «S,E,F» gdiGetQual()

函数调用

```
gdiGetQual
          (
          ):A_INT16
```

函数说明

返回设备驱动程序 API 错误的相应值。

返回值

返回 GDI 设备驱动程序 API 函数的错误代码。

A.2.10.5 GdiCoDeviceDriverException :: «S,E,F» gdiGetRetValue()

函数调用

```
gdiGetRetValue
          (
          ):A_INT16
```

函数说明

返回 GDI API 错误的相应值。

返回值

返回 GDI 设备驱动程序 API 函数的错误代码。

A.2.11 «S,E,F» GdiCoDriverResult

此接口应用于传输来自设备驱动程序的警告和信息。

图A.11 GdiCoDriverResult层次图

A.2.11.1 GdiCoDriverResult :: «S,E,F» gdiGetCode()

函数调用

```
gdiGetCode
          (
```

):A_INT16

函数说明

返回 GDI API 信息的相应值。

返回值

返回 GDI 设备驱动程序 API 函数的信息。

A.2.11.2 GdiCoDriverResult :: «S,E,F» gdiGetGrade()

函数调用

```
gdiGetGrade
          (
          ):A_INT16
```

函数说明

返回 GDI API 信息的相应值。

返回值

返回 GDI 设备驱动程序 API 函数的信息代码。

A.2.11.3 GdiCoDriverResult :: «S,E,F» gdiGetQual()

函数调用

```
gdiGetQual
          (
          ):A_INT16
```

函数说明

返回 GDI API 信息的相应值。

返回值

返回 GDI 设备驱动程序 API 函数的信息代码。

A.2.11.4 GdiCoDriverResult :: «S,E,F» gdiIsWarningOrInformation()

函数调用

```
gdiIsWarningOrInformation
          (
          ):eGDIDRIVERRESULTTYPE
```

函数说明

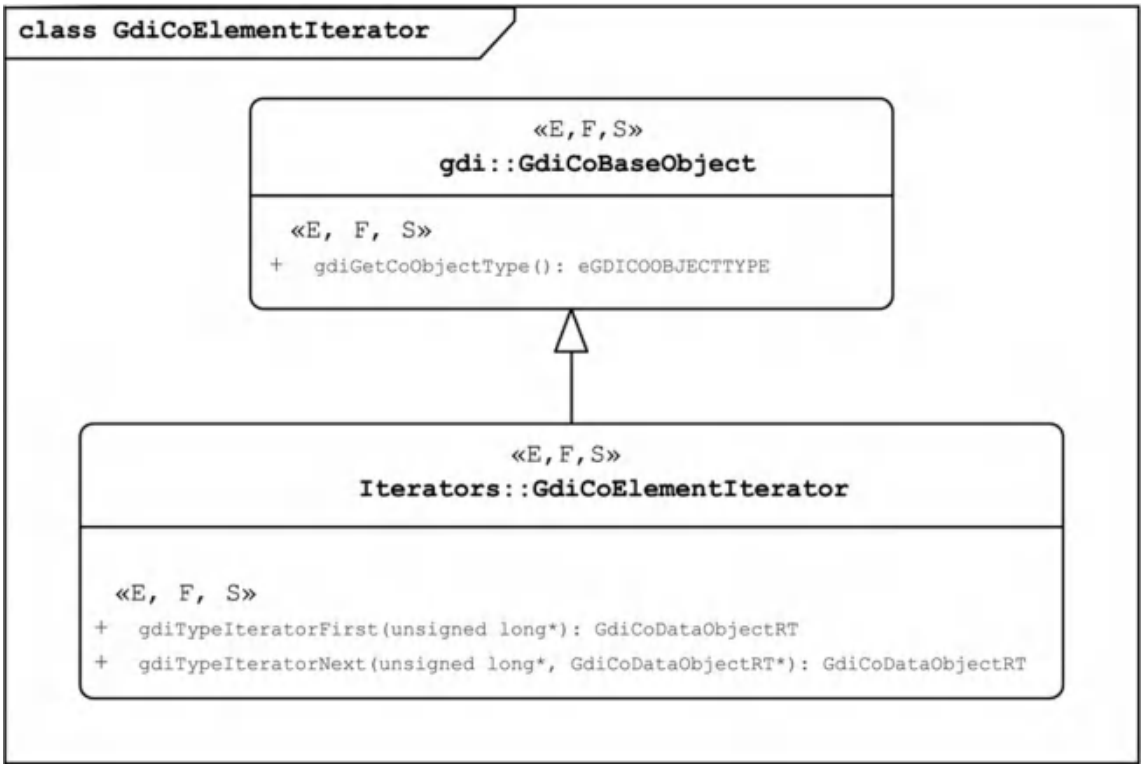
如果发生警告或信息，则返回有关信息的枚举。

返回值

如果发生警告或信息，则返回有关信息的枚举。

A.2.12 «S,E,F» GdiCoElementIterator

该接口应包含用于类型迭代器的服务，是结构体或联合体中已设所有子类型的顺序输出。



图A.12 GdiCoElementIterator层次图

A.2.12.1 GdiCoElementIterator :: «S,E,F» gdiTypeIteratorFirst()

函数调用

```
gdiTypeIteratorFirst
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoDataObjectRT
```

函数说明

设置指向控制接口内第一个类型的内部指针。

返回值

返回对第一个 GdiCoBaseType 的引用（如果存在），否则返回 NIL 引用。

A.2.12.2 GdiCoElementIterator :: «S,E,F» gdiTypeIteratorNext()

函数调用

```
gdiTypeIteratorNext
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoDataObjectRTrefLastElement
        /*inparameter
        A reference to a element returned by gdiTypeIteratorFirst or
        gdiTypeIteratorNext.*/
        ):GdiCoDataObjectRT
```

函数说明

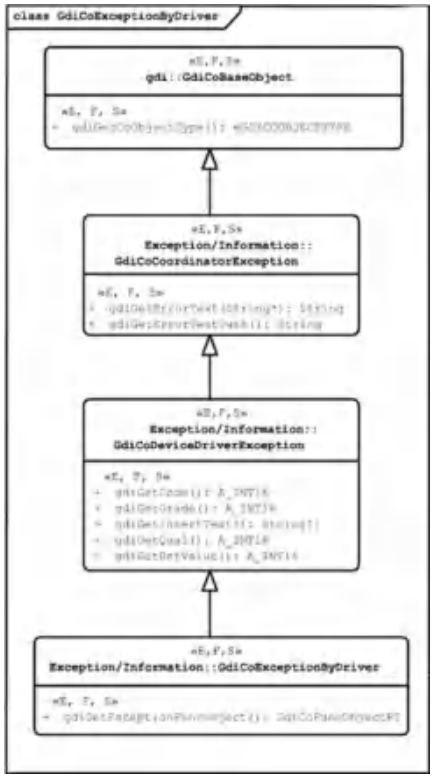
检测对控制接口中当前类型（参数）的引用。设置指向下一个类型（如果存在）的内部指针并返回引用。

返回值

返回对协调器基本类型（协调器内的管理单元）的引用。

A.2.13 «S,E,F» GdiCoExceptionByDriver

此接口应抛出用户定义函数对象的异常。设备驱动程序可能抛出 DCD 中描述的一个或多个 FuncObject 异常。
协调器应生成一个动态函数对象，并应参照 FO 抛出一个 GdiExceptionByDriver。
释放 GdiExceptionByDriver 对象后，协调器应删除动态函数对象。



图A.13　GdiCoExceptionByDriver层次图

A.2.13.1 GdiCoExceptionByDriver :: «S,E,F» gdiGetExceptionFuncObject()

函数调用

```
gdiGetExceptionFuncObject
          (
          ):GdiCoFuncObjectRT
```

函数说明

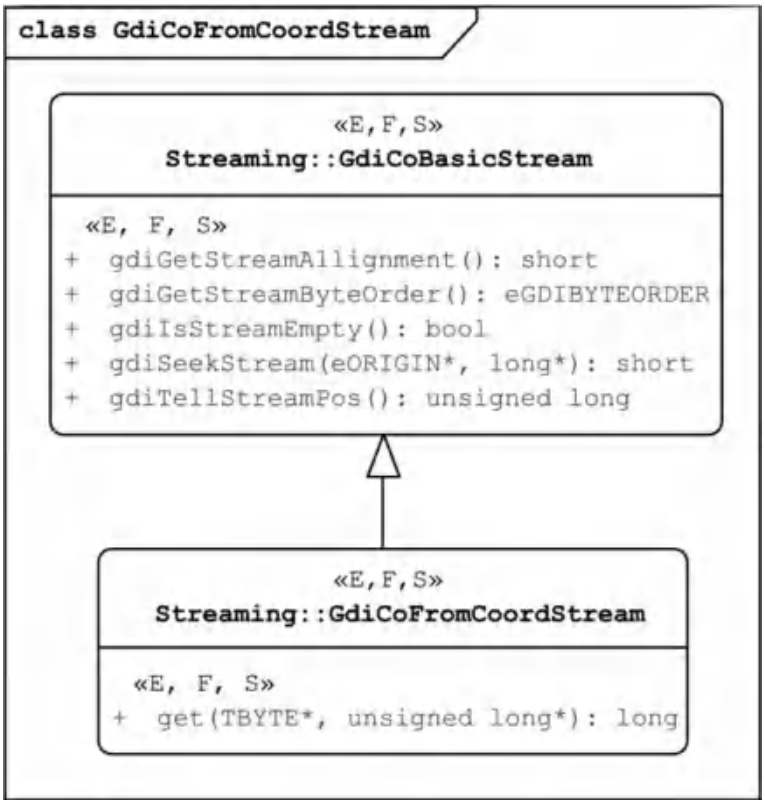
返回给定位置上的 FuncObject 的引用，该引用是 GDI 驱动程序已作为异常抛出的。

返回值

返回对 GdiCoFuncObject 对象的引用。如果发生错误，该函数将返回 NIL。

A.2.14 «S,E,F» GdiCoFromCoordStream

该接口用于从协调器读取未格式化的字节序列。



图A.14 GdiCoFromCoordStream层次图

A.2.14.1 GdiCoFromCoordStream :: «S,E,F» get()

函数调用

```
get
        (
        inout                TBYTE refDestination
        /* in parameter
        contains    the    reference    to    the    destination    buffer    in    the
        application. */
        inout                unsigned long ulnByteCount
        /* in parameter
        contains the requested number of bytes to be read. */
        ) : long
```

函数说明

从流中获取数据到应用程序。

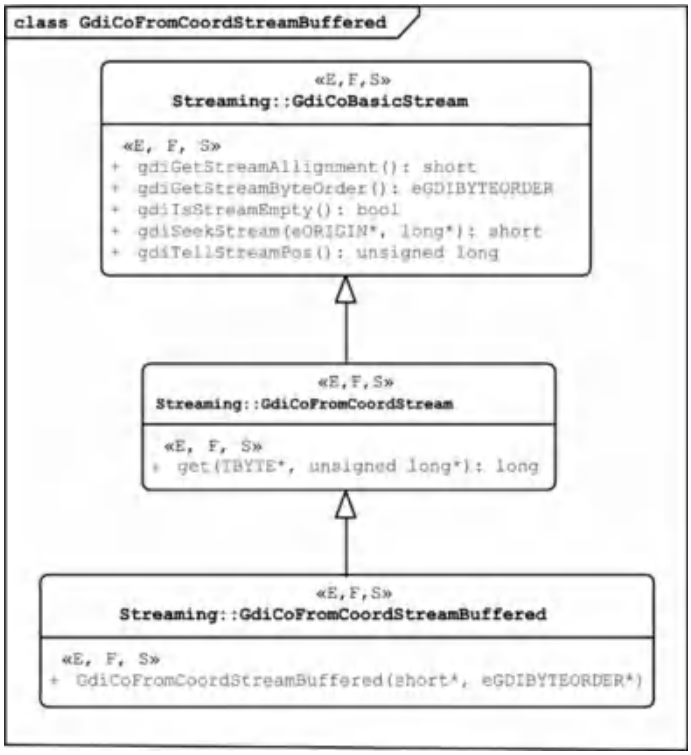
返回值

返回已读取的元素计数。

返回负数表示错误。

A.2.15 «S,E,F» GdiCoFromCoordStreamBuffered

该接口用于从协调器读取未格式化的字节序列。流对象中的数据应由协调器使用下一个 gdiRead,gdiExecute 和 gdiGetDataValue（在扩展访问中）进行更新。流中的数据应为协调器内部存储器的副本。



图A.15　GdiCoFromCoordStreamBuffered层次图

A.2.15.1 GdiCoFromCoordStreamBuffered :: «S,E,F» GdiCoFromCoordStreamBuffered()

函数调用

```
GdiCoFromCoordStreamBuffered
          (
          inout                  shortnAlignment
          /*inparameter
          definestherequestedAlignmentoftheApplication.
          ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator environment.*/
          inout                  eGDIBYTEORDEReByteOrder
          /*inparameter
          definestherequestedByteOrderoftheApplication.
          ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator environment.*/
          ):
```

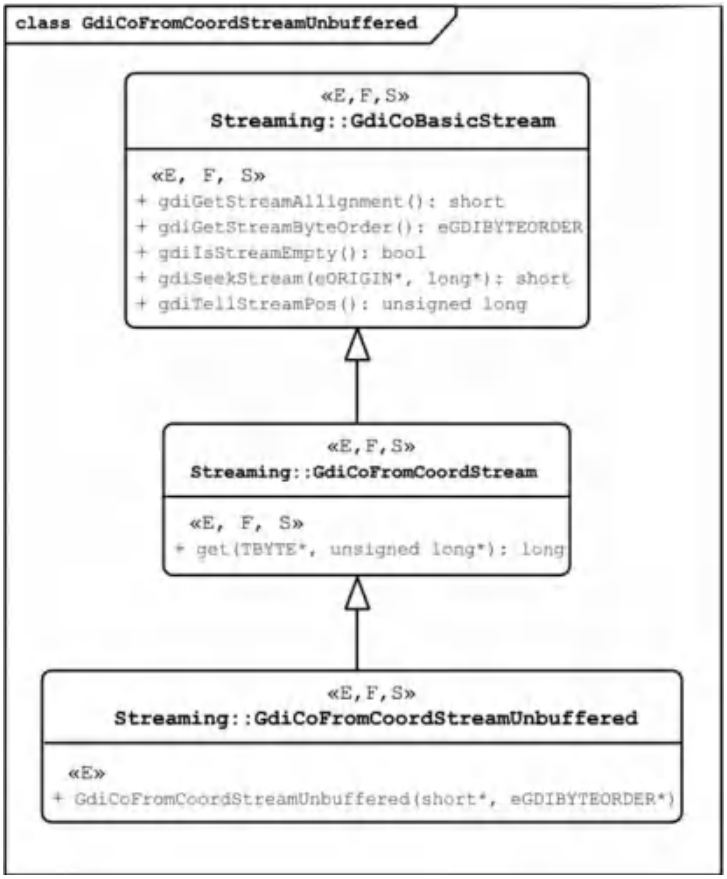
函数说明

流对象的构造函数，用于定义对齐方式和字节顺序。

如果与设备驱动程序通信（gdiRead，gdiExecute），则协调器将更新流对象中的数据。

A.2.16 «S,E,F» GdiCoFromCoordStreamUnbuffered

该接口用于从协调器读取未格式化的字节序列。流对象应引用协调器内部存储器。应通过修改流中的数据对协调器内部存储器进行修改。



图A.16　GdiCoFromCoordStreamUnbuffered层次图

A.2.16.1 GdiCoFromCoordStreamUnbuffered :: «E» GdiCoFromCoordStreamUnbuffered()

函数调用

```
GdiCoFromCoordStreamUnbuffered
        (
        inout                   shortnAlignment
        /*inparameter
        definestherequestedAlignmentoftheApplication.
        ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator
        environment.*/
        inout                   eGDIBYTEORDEReByteOrder
        /*inparameter
        definestherequestedByteOrderoftheApplication.
        ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator
        environment.*/
        ):
```

函数说明

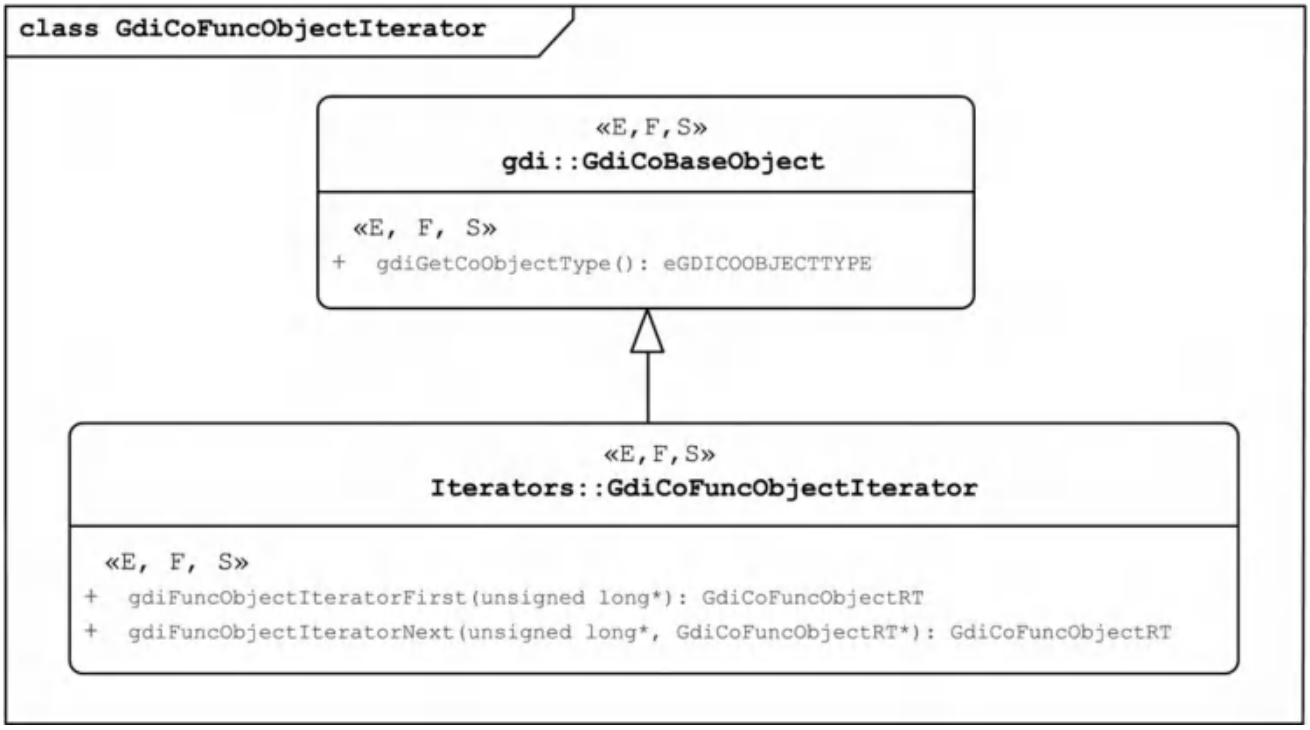
无缓冲流对象的构造函数，用于定义对齐方式和字节顺序。

流对象用作与 gdiReadValue 或 gdiExecute 的 out 参数相关联的无缓冲输出引用。

如果与设备驱动程序通信（gdiRead，gdiExecute），流对象中的数据将被更新。

A.2.17 «S,E,F» GdiCoFuncObjectIterator

此接口包含 FuncObject 迭代器应用程序的服务。它应显示已在 VD 中设置的所有应用程序功能对象的顺序输出。



图A.17 GdiCoFuncObjectIterator层次图

A.2.17.1 GdiCoFuncObjectIterator :: «S,E,F» gdiFuncObjectIteratorFirst()

函数调用

```
gdiFuncObjectIteratorFirst
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoFuncObjectRT
```

函数说明

设置指向工作区内第一个 VD 的内部指针。

返回值

返回对第一个 GdiCoFuncObject 的引用（如果存在），否则返回 NIL 引用。

A.2.17.2 GdiCoFuncObjectIterator :: «S,E,F» gdiFuncObjectIteratorNext()

函数调用

```
gdiFuncObjectIteratorNext
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoFuncObjectRTrefLastElement
        /*inparameter
        Areferencetoaelementreturnedbygdi FuncObjectIteratorFirstor
        gdiFuncObjectIteratorNext.*/
        ):GdiCoFuncObjectRT
```

函数说明

检测对工作空间中当前 FuncObject（参数）的引用。将内部指针设置为下一个 FuncObject（如果存在）并返回引用。

返回值

返回对下一个 GdiCoFuncObject 的引用（如果存在），否则返回 NIL 引用。

A.2.18 «S,E,F» GdiCoFuncObjectRT

此接口包含引用通信对象和函数对象操作的所有服务。

图A.18 GdiCoFuncObjectRT层次图

A.2.18.1 GdiCoFuncObjectRT :: «S,E,F» gdiGetCommObjectIterator()

函数调用

```
gdiGetCommObjectIterator
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoCommObjectIterator
```

函数说明

它用于输出函数对象的所有已设置通信对象。

返回值

返回对 GdiCoComObjectIterator 的引用（如果成功）。

A.2.18.2 GdiCoFuncObjectRT :: «S,E,F» gdiGetCommObjectRTByInstanceName()

函数调用

```
gdiGetCommObjectRTByInstanceName
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   StringsCommObjectInstanceName
        /*inparameter
```

```
            NameofexpectedInstance;definedduringparameterization*/
            ):GdiCoCommObjectRT
```

函数说明

用于检测对通信对象实例的引用。

返回值

返回对 GdiCoCommObjectRT 的引用（如果成功）。

A.2.18.3 GdiCoFuncObjectRT :: «S,E,F» gdiGetFuncObjectInstanceName()

函数调用

```
gdiGetFuncObjectInstanceName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):String
```

函数说明

返回参数化期间定义的对象的实例名称。

实例名称在 VD 内部必须唯一，因此名称与 DCD 中定义的相应接口的名称不同。

如果此对象是动态 FO，则实例名称应符合以下语法：

<InterfaceName>_<InstanceCounter>

InterfaceName: DCD 中的类名

InstanceCounter:计数器随所有动态 FOs 的实例数而增加。

不考虑删除的实例。

返回值

返回 FO 实例的实例名称。

A.2.18.4 GdiCoFuncObjectRT :: «S,E,F» gdiGetOperationIterator()

函数调用

```
gdiGetOperationIterator
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoOperationIterator
```

函数说明

用于输出函数对象的所有可用操作。

返回值

返回对 GdiCoOperationIterator 的引用（如果成功）。

A.2.18.5 GdiCoFuncObjectRT :: «S,E,F» gdiGetOperationRTByInstanceName()

函数调用

```
gdiGetOperationRTByInstanceName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsOperationObjectInstanceName
            /*inparameter
            NameofexpectedOperation;NamedefinedinDCDorredefinedduring parameterization.*/
            ):GdiCoOperationRT
```

函数说明

用于检测对可执行操作的引用。

返回值

返回对 GdiCoOperation 的引用（如果成功）。

A.2.19 «S,E,F» GdiCoInternalException

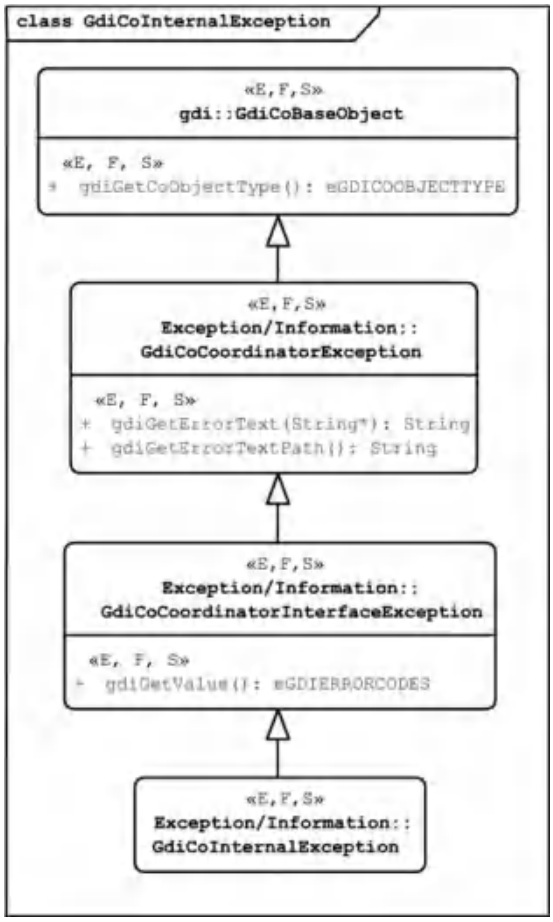
如果协调器实现中发生错误，则将抛出此异常类。

错误代码

— eINT_INTERNAL_ERROR（协调器故障，操作系统故障）和

— eINT_VERSION_NOT_SUPPORTED（不支持所请求的接口版本）

应与此例外相关联。



图A.19 GdiCoInternalException的层次结构图

A.2.20 [S, E, F] GdiCoManager

通过此接口的服务，应用程序可以连接和断开连接。该接口的服务应永久可用。在面向对象的实现中，该实例应在协调器的开头创建。

图A.20 GdiCoManager的层次结构图

A.2.20.1 GdiCoManager : : [S, E, F] gdiCreateWorkspace ()

函数调用

```
gdiCreateWorkspace
        (
        inout                   StringsInstanceDescription
        /*inparameter
        Containsparameterizationdataorthenameofaparameterizationfile
        */
        inout                   unsignedlongulTimeOut
        /*inparameter
        Maximumdurationtimefortheexecutionofacoordinatorservicein
        milliseconds(themaximumisabout50days)*/
        inout                   TErrorFunctionReferencerefCBException
        /*inparameter
        Referencetoamethodwhichhandleserrors.Ifaexceptionmechanism
        willbeused(e.g.C++exception)useNILreferenceforthisparameter.
        */
        inout                   TAcceptFunctionReferencerefCBAccept
        /*inparameter
        Referencetoacallbackmethodwhichhandlestheacceptmechanism.A
        NILreferencesignalsthatnocallbackissupported.
        Atthecoordinatorcallbacktotheapplication,anreferencetothe
        specificGdiCoCommObjectRTispassed.*/
        inout                   TInfRepFunctionReferencerefCBInformationReport
        /*inparameter
        Referencetoacallbackmethodwhichhandlestheinformationreport
        mechanism.ANILreferencesignalsthatnocallbackissupported.
        Atthecoordinatorcallbacktotheapplication,anreferencetothe
        specificGdiCoCommObjectRTispassed.*/
        inout                   StringsWorkspaceName
        /*inparameter
        InstanceNameofthenewWorkspace*/
        inout                   eGDICOIFKINDeRequiredCoordinatorInterface
        /*inparameter
        contains theexpected CoordinatorInterfaceTypeSmart,Extended,Full
        asbitmask.(seesemantics)*/
        inout                   unsignedlongulAppHnd
        /*outparameter
        DeliverstheIdentifieroftheapplication.*/
         ):GdiCoWorkspace
```

功能说明

该方法将创建一个新的工作空间，其中包含一组功能实例。这些功能实例可以是几个 VD 的成员。实例名称必须是强制性的，并且可以由用户定义，而与 PID 中给定的工作空间名称无关。空字符串作为参数将被拒绝。

回调指针应为异步完成的入口点或事件的信号入口（InformationReport，接受）。

参数 eRequiredCoordinatorInterface 定义了请求的协调器接口类型：

0 = eSMART1 = eEXTENDED

2 = eFULL.

如果协调器不支持预期的接口，则应拒绝工作空间的创建。在实例创建过程中发生任何错误的情况下，对GdiCoWorkspace 的引用应为 NIL 并且工作空间应立即删除。

返回值

如果成功，则返回 GdiCoWorkspace，否则返回 NIL。

A.2.20.2 GdiCoManager : : [S, E, F] gdideleteWorkspace ()

函数调用

```
gdiDeleteWorkspace
          (
          inout                  unsignedlongulAppHand
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoWorkspacerefWorkspace
          /*ReferencetoaexistingWorkspace*/
          ):void
```

功能说明

该服务应将应用程序与协调器断开连接。这样，将删除协调器和设备驱动程序中的所有实例。

协调器行为取决于创建时所需的协调器访问权限
(gdiCreateWorkspace(),gdiGetWorkspaceByName()).

在需要智能访问或扩展访问的情况下，协调器应删除所有通信对象，功能对象和虚拟设备。因为错误将被忽略，所以可以使用 Device Driver API 的 Abort 功能简单地实现。设备驱动程序本身也应解耦。

在需要完全访问接口的情况下，销毁通信对象，功能对象，VD 和设备驱动程序的所有活动均应由应用程序通过协调器完全接口直接控制。如果发生任何错误，则应用程序负责以下步骤。仅当工作空间为空时，在完全访问的情况下 gdiDeleteWorkspace（）将成功。这是“描述”和“运行时”端所必需的。

返回值

无

A.2.20.3 GdiCoManager : : [S, E, F] gdiGetbuildversion ()

函数调用

```
gdiGetBuildVersion
          (
          ):unsignedlong
```

功能说明

返回协调器的内部版本。

返回值

返回构建版本。

A.2.20.4 GdiCoManager :: «S,E,F» gdiGetCoordinatorType()

函数调用

```
gdiGetCoordinatorType
          (
          ):unsignedlong
```

功能说明

返回支持的接口访问类型作为枚举。接口类型：

0 = eSMART

1 = eEXTENDED
2 = eFULL.

返回值

以位组合形式返回支持的接口类型。

A.2.20.5 GdiCoManager :: «S,E,F» gdiGetCoordinatorVersion()

函数调用

```
gdiGetCoordinatorVersion
          (
          ):GdiCoA_Version
```

功能说明

返回 GDI 版本对象。该接口包含受支持的最高 GDI 版本。

返回值

返回对 GDIVersion 接口的引用。

A.2.20.6 GdiCoManager :: «S,E,F» gdiGetMonitorAccessByWorkspaceName()

函数调用

```
gdiGetMonitorAccessByWorkspaceName
          (
          inout                  StringsWorkspaceName
          /*inparameter
          containsthenameoftheexpectedworkspace.*/
          inout                  eGDICOIFKINDeRequiredCoordinatorInterface
          /*inparameter
          contains          theexpected          CoordinatorInterfaceTypeSmart,Extended,Full
          asbitmask.(seesemantics)*/
          inout                  TErrorFunctionReferencerefCBException
          /*inparameter
          Referencetoamethodwhichhandleserrors.Ifaexceptionmechanism
          willbeused(e.g.C++exception)useNILreferenceforthisparameter.
          */
          inout                  TInfRepFunctionReferencerefCBInformationReport
          /*inparameter
          Referencetoacallbackmethodwhichhandlestheinformationreport
          mechanism.ANILreferencesignalsthatnocallbackissupported.
          Atthecoordinatorcallbacktotheapplication,anreferencetothe
          specificGdiCoCommObjectRTispassed.*/
          inout                  unsignedlongulAppHnd
          /*outparameter
          DeliverstheIdentifieroftheapplication.*/
          ):GdiCoWorkspace
```

功能说明

返回工作空间的引用，对应于给定的工作空间名称。

参数 eRequiredCoordinatorInterface 描述了应用程序预期的协调器接口类型。

0 = eSMART
1 = eEXTENDED
2 = eFULL

如果协调器不支持预期的接口，则无法返回工作空间。

如果在 GdiCoAttributRT 上启用了信息报告标志，并且设备驱动程序不支持该属性的信息报告，则只要该属性被另一个客户端更改，协调器就会向监视客户端发送信息报告。

如果设备驱动程序向该属性发送信息报告，则该信息将传递给所有监视客户端，从而启用信息报告标志。

返回值

如果成功，则返回 GdiCoWorkspace 以进行监视器访问，否则返回 NIL。

A.2.20.7 GdiCoManager : : [S, E, F] gdiGetWorkspacebyName ()

函数调用

```
gdiGetWorkspaceByName
          (
          inout                  StringsWorkspaceName
          /*inparameter
          containsthenameoftheexpectedworkspace.*/
          inout                  eGDICOIFKINDeRequiredCoordinatorInterface
          /*inparameter
          containstheexpectedCoordinatorInterfaceType(Smart,Extended,Full
          asbitmask.(seesemantics)*/
          inout                  TErrorFunctionReferencerefCBException
          /*inparameter
          Referencetoamethodwhichhandleserrors.Ifaexceptionmechanism
          willbeused(e.g.C++exception)useNILreferenceforthisparameter.
          */
          inout                  TAcceptFunctionReferencerefCBAccept
          /*inparameter
          Referencetoacallbackmethodwhichhandlestheacceptmechanism.A
          NILreferencesignalsthatnocallbackissupported.
          Atthecoordinatorcallbacktotheapplication,anreferencetothe
          specificGdiCoCommObjectRTispassed.*/
          inout                  TInfRepFunctionReferencerefCBInformationReport
          /*inparameter
          Referencetoacallbackmethodwhichhandlestheinformationreport
          mechanism.ANILreferencesignalsthatnocallbackissupported.
          Atthecoordinatorcallbacktotheapplication,anreferencetothe
          specificGdiCoCommObjectRTispassed.*/
          inout                  unsignedlongulAppHnd
          /*outparameter
          DeliverstheIdentifieroftheapplication.*/
          ):GdiCoWorkspace
```

功能说明

返回工作空间的引用，对应于给定的工作空间名称。

参数 eRequiredCoordinatorInterface 描述了应用程序预期的协调器接口类型。

0 = eSMART

1 = eEXTENDED

2 = eFULL.

如果协调器不支持预期的接口，则无法返回工作空间。

如果参数 eRequiredCoordinatorInterface 为 eSMART 或 eEXTENDED，则所有 VD 都将转换为工作状态。

如果参数 eRequiredCoordinatorInterface 为 eFULL，则所有 VD 都保留在状态准备中。

返回值

返回 GdiCoWorkspace（如果存在），否则返回 NIL。

A.2.20.8 GdiCoManager : : [S, E, F] gdiGetWorkspaceTerter ()

函数调用

```
gdiGetWorkspaceIterator
          (
          ):GdiCoWorkspaceIterator
```

功能说明

服务所有现有工作区的输出。通过诊断工具使用。

参数 eRequiredCoordinatorInterface 描述了应用程序预期的协调器接口类型。

返回值

如果成功，则返回 GdiCoWorkspaceIterator（句柄，指针，类），否则返回 0。

A.2.20.9 GdiCoManager : : [S, E, F] gdiKillWorkspace ()

函数调用

```
gdiKillWorkspace
          (
          inout                  StringsWorkspaceName
          /*inparameter
          NameoftheWorkspacetokill.*/
          ):void
```

功能说明

使用给定的 workspacename 终止工作空间。

返回值

无

A.2.20.10 GdiCoManager :: [S, E, F] gdiReleaseMonitorAccess ()

函数调用

```
gdiReleaseMonitorAccess
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoWorkspacerefWorkspace
          /*inparameter
          referencetotheworkspace.*/
          ):void
```

功能说明

此方法应释放工作空间而不关闭它。注册的申请无效。

返回值

无

A.2.20.11 GdiCoManager :: [S, E, F] gdiReleaseWorkspace ()

函数调用

```
gdiReleaseWorkspace
gdiReleaseWorkspace
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoWorkspacerefWorkspace
          /*inparameter
          referencetotheworkspace.*/
          ):void
```

功能说明

此方法应释放工作空间而不关闭它。现在可以使用其他应用程序访问工作空间。

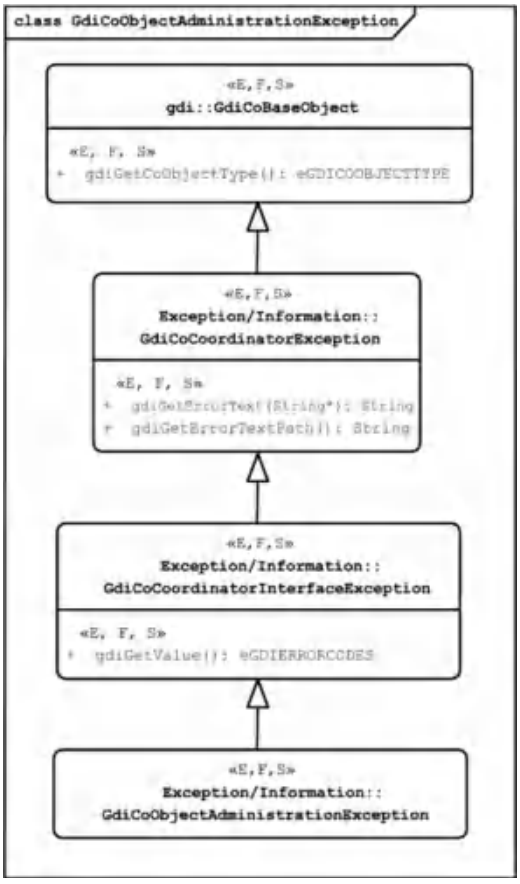
协调器将向工作空间的所有 VD 发起独立于其初始状态的状态更改为 Preperation，然后释放工作空间句柄。如果 VD 由于错误而不允许状态转换，则应拒绝工作空间的释放。

返回值

没有

A.2.21 «S,E,F» GdiCoObjectAdministrationException

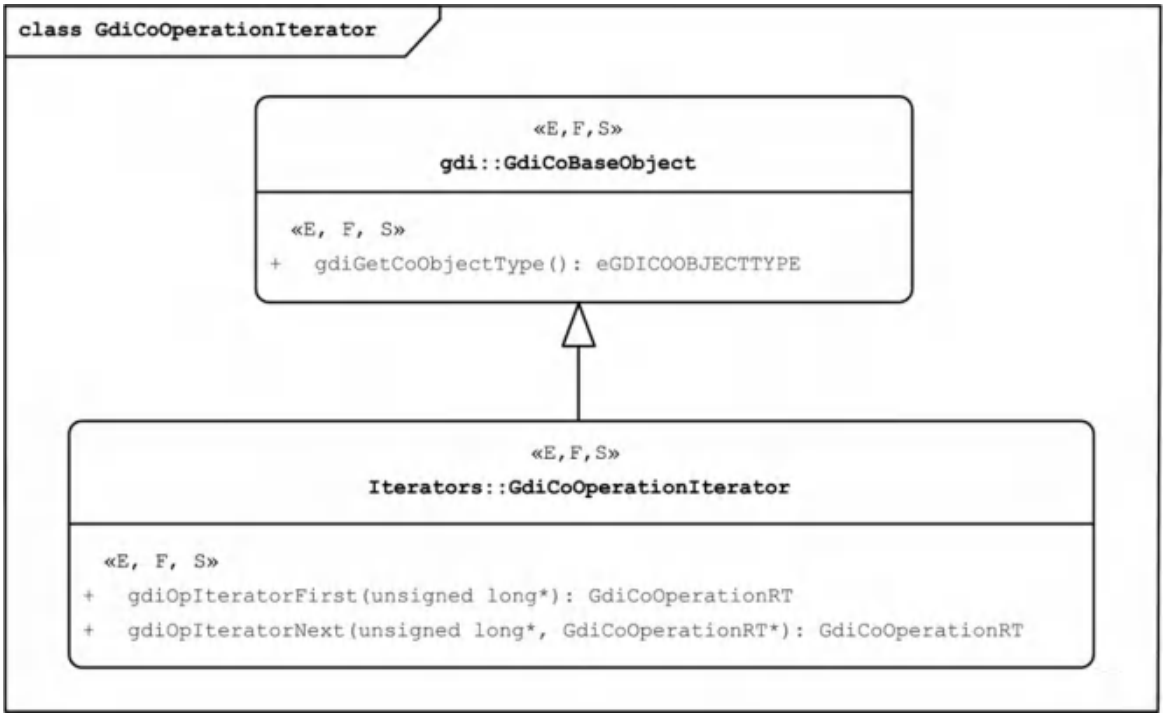
如果由于 Coordinator API 的处理错误而导致在协调器内部对象管理中发生错误，则抛出此异常类：
错误代码
— eOAD_OUT_OF_RANGE（序列，数组访问）
— eOAD_OPEN_SERVICE（正在使用 GDI 对象）
— eOAD_TYPE_NOT_ALLOWED（如果新元素是序列，并且是 Union 的一部分）
— eOAD_OBJECT_ACCESS（数据源接收器不可用）
与这个例外有关。



图A.21　GdiCoObjectAdministrationException的层次结构图

A.2.22 «S,E,F» GdiCoOperationIterator

此接口包含用于操作迭代器应用程序的服务，该迭代器是在 Function 对象中设置的所有操作的顺序输出。



图A.22 — GdiCoOperationIterator的层次结构图

A.2.22.1 GdiCoOperationIterator :: «S,E,F» gdiOpIteratorFirst()

函数调用

```
gdiOpIteratorFirst
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoOperationRT
```

功能说明

将内部指针设置为FO接口内的第一个操作。

返回值

返回对第一个GdiCoOperation的引用（如果存在），否则返回NIL引用。

A.2.22.2 GdiCoOperationIterator :: «S,E,F» gdiOpIteratorNext()

函数调用

```
gdiOpIteratorNext
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                    GdiCoOperationRTrefLastElement
          /*inparameter
          A reference to a element returned by gdiOPIteratorFirst or
          gdiOPIteratorNext.*/
          ):GdiCoOperationRT
```

功能说明

检测对功能对象中当前OP（参数）的引用。将内部指针设置为下一个OP（如果存在）并返回引用。

返回值

返回对协调器操作（协调器内的管理单元）的引用。

A.2.23 [S, E, F] GdiCoopeRT

此接口包含用于操作的服务。

图A.23 GdiCoOperationRT的层次结构图

A.2.23.1 GdiCoOperationRT ::«S, E, F»gdiExecuteOperationValue ()

函数调用

```
gdiExecuteOperationValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoToCoordStreamrefInValue
        /*inparameter
        Referencetoastreamobject,thetype          isidenticaltothetype          ofthe
        operationinparameterobject.*/
        inout                   GdiCoFromCoordStreamrefOutValue
        /*outparameter
        Referencetoastreamobject,thetype isidenticaltothetype ofthe
        outparamterobject.*/
        inout                   GdiCoDriverResultrefDestInfo
        /*outparameter
        contains   the   reference   to   the   Information   Object,   warnings   and
        informationsoftheDriverwillbestoredinit.*/
        inout                   TEXEFunctionReferencerefCBCompleteExec
        /*inparameter
        Referencetoacallbackmethodwhichhandles theexecutionmechanism.
        ANILreferencesignalsasynchronouscall.*/
        ):short
```

功能说明

该服务将数据接管到操作的“输入”区域中，开始操作并返回返回值。

返回值

描述由 GDI 驱动程序执行的通信：

0 =同步返回；

1 =异步返回；

小于 0 =错误。

A.2.23.2 GdiCoopert : : [S, E, F] gdiGetFuncObjectrt ()

函数调用

```
gdiGetFuncObjectRT (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoFuncObjectRT
```

功能说明

创建对现有功能对象的引用。

引用的功能对象包含此操作对象。

返回值

如果成功，则返回对 GdiCoFuncObjectRT 的引用。

A.2.23.3 GdiCoOperationRT :: «S,E,F» gdiGetOperationInstanceName()

函数调用

```
gdiGetOperationInstanceName
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):String
```

功能说明

返回参数化期间定义的对象的实例名称。

操作的实例是隐式的（不需要 API 活动）。接口的操作名称和 FuncObject 的操作名称可以相同。

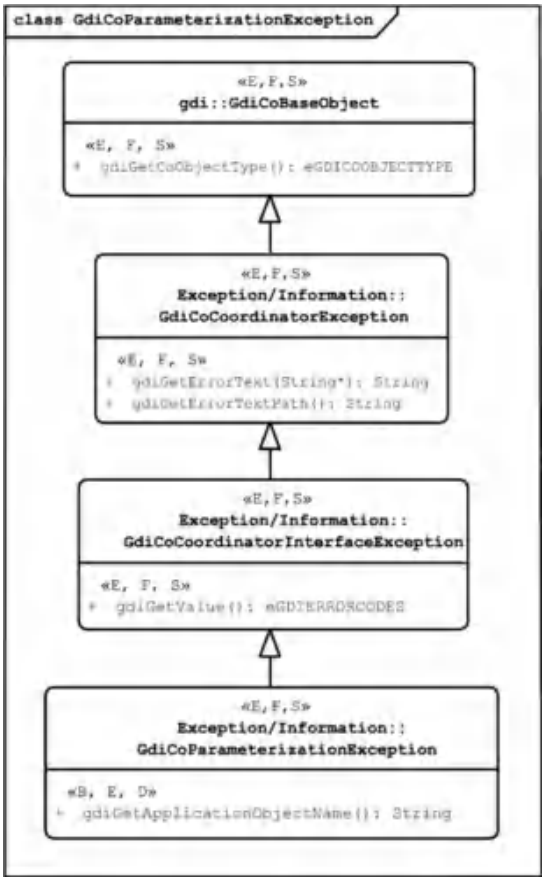
返回值

以字符串形式返回实例名称。

A.2.24 «S,E,F» GdiCoParameterizationException

如果 PID 文件不正确或无效，则抛出此异常类。

错误代码 ePAR_INCORRECT_PARAMETERIZATION 与该异常有关。



图A.24 — GdiCoParameterizationException的层次结构图

A.2.24.1 GdiCoParameterizationException :: «B,E,D» gdiGetApplicationObjectName()

函数调用

```
gdiGetApplicationObjectName
          (
          ):String
```

功能说明

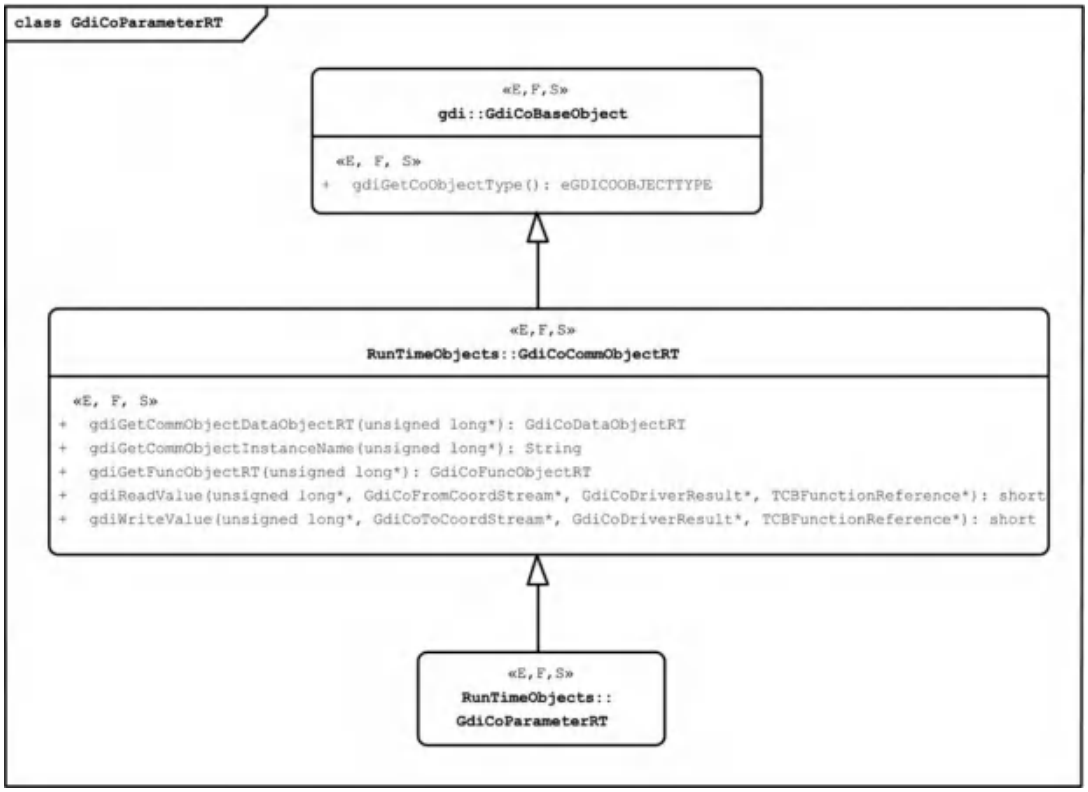
返回无法建立的抽象数据源接收器的名称。 PID 处理在此位置上失败。

返回值

返回无法建立的抽象数据源接收器的名称。(FO，CO，OP)。

A.2.25 [S， E， F] GdiCoParameterRT

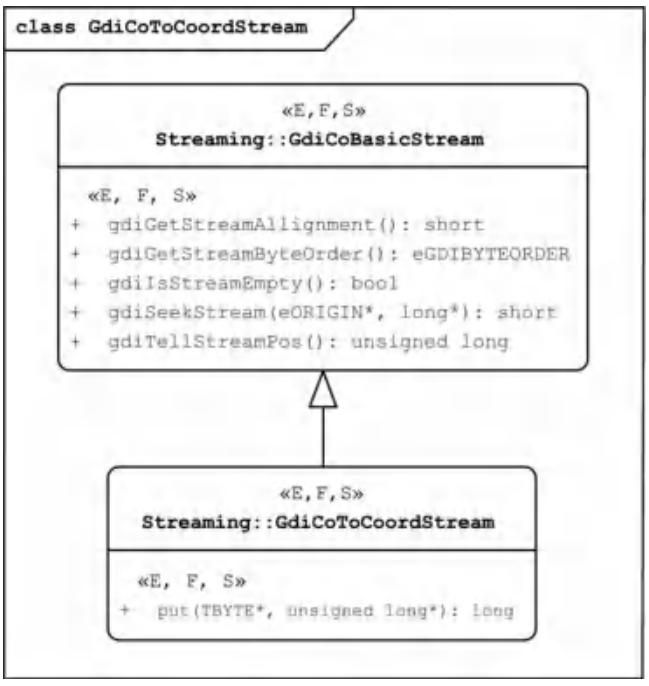
与通讯对象相比，没有其他服务。但是，在各个 VD 的状态下它是只读的。



图A.25　GdiCoParameterRT的层次结构图

A.2.26 «S,E,F» GdiCoToCoordStream

该接口用于从协调器写入未格式化的字节序列。



图A.26　GdiCoToCoordStream的层次结构图

A.2.26.1 GdiCoToCoordStream :: «S,E,F» put()

函数调用

```
put
    (
    inout          TBYTE refSource
```

```
        /* in parameter
        containsareferencetotheelementsintheapplicationwhichto
        write to the stream. */
        inout          unsigned long ulnByteCount
        /* in parameter
        contains the requested number of bytes to be read. */
        ) : long
```

功能说明

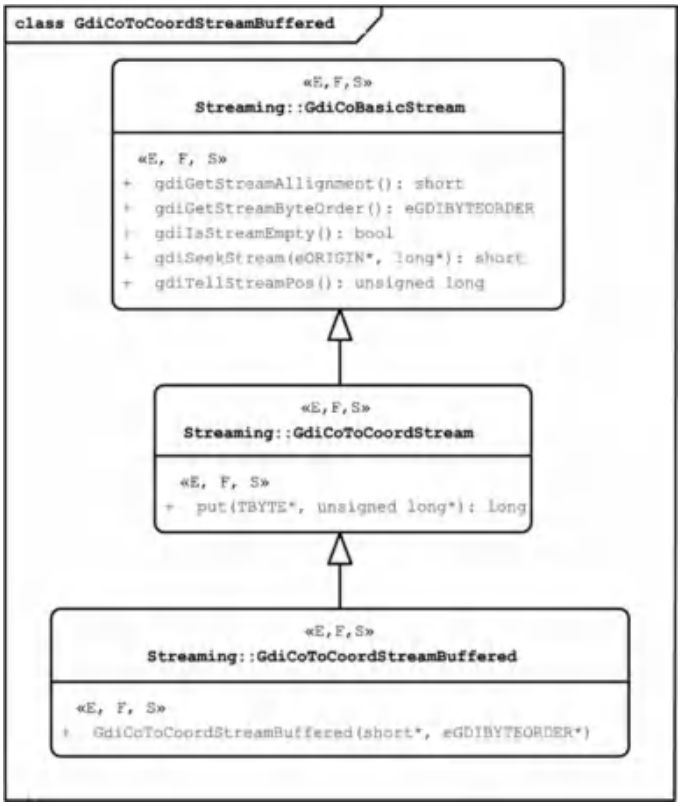
将数据从应用程序放到流中。

返回值

返回元素的计数，该计数已放入流中。负数表示错误。

A.2.27 «S,E,F» GdiCoToCoordStreamBuffered

该接口用于将未格式化的字节序列写入协调器。客户端负责创建和删除该象。
更新流对象中的数据不会影响对协调器内部存储器的直接修改。
下一个 gdiWrite，gdiExecute，gdiSetDataValue（在扩展访问中）操作将更新协调器内存。



图A.27 GdiCoToCoordStreamBuffered的层次结构图

A.2.27.1 GdiCoToCoordStreamBuffered :: «S,E,F» GdiCoToCoordStreamBuffered()

函数调用

```
GdiCoToCoordStreamBuffered
        (
        inout                   shortnAlignment
        /*inparameter
        definestherequestedAlignmentoftheApplication.
        ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator
        environment.*/
        inout                   eGDIBYTEORDEReByteOrder
        /*inparameter
```

```
definestherequestedByteOrderoftheApplication.
ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator environment.*/
):
```

功能说明

无缓冲 StreamObject 的构造方法。定义分配和字节顺序。

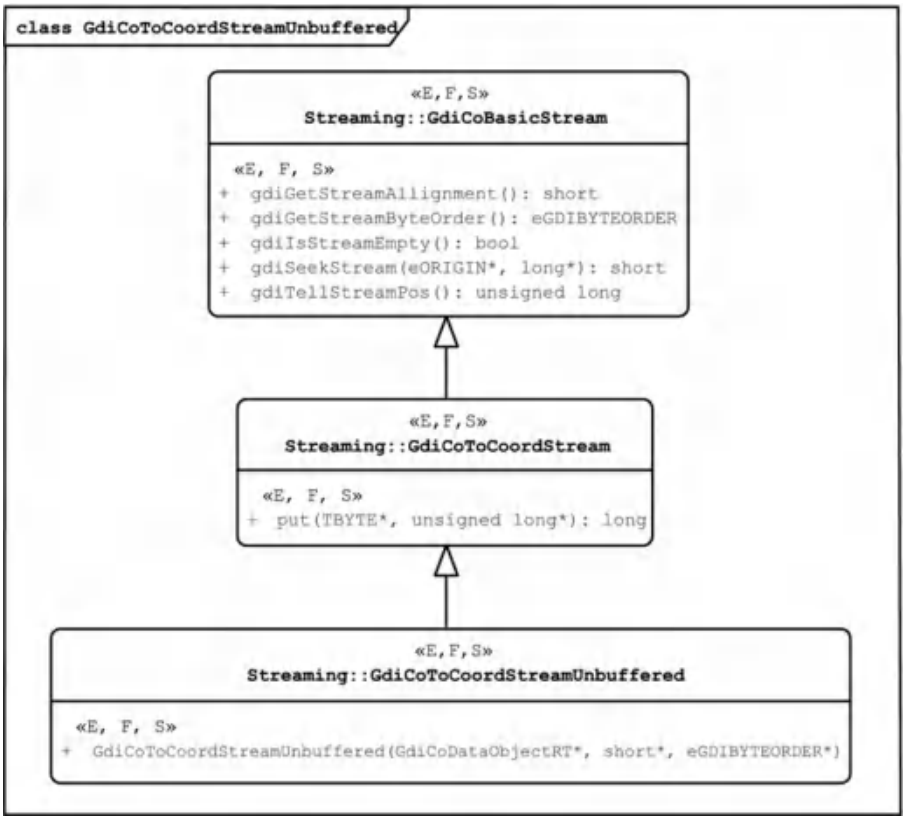
更新流对象中的数据不会影响对协调器内部存储器的直接修改。

下一个 gdiWrite，gdiExecute，gdiSetDataValue（在扩展访问中）操作将更新协调器内存。

A.2.28 «S,E,F» GdiCoToCoordStreamUnbuffered

该接口用于将未格式化的字节序列写入协调器。客户端负责创建和删除该象。

更新 Stream 对象中的数据将影响对 Coordinator 内部的直接修改记忆。



图A.28 GdiCoToCoordStreamUnbuffered的层次结构图

A.2.28.1 GdiCoToCoordStreamUnbuffered :: «S,E,F» GdiCoToCoordStreamUnbuffered()

函数调用

```
GdiCoToCoordStreamUnbuffered
        (
        inout                   GdiCoDataObjectRTrefGdiDestination
        /*inparameter
        definestheDestinationDataAreaintheCoordinator.
        ThisParametermustnotbeNIL.TheStreamObjectisusedasInReference
        inconnectionwithgdiWriteValueortheinParameterofagdiExecute.
        */
        inout                   shortnAlignment
        /*inparameter
        definestherequestedAlignmentoftheApplication.
        ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator
        environment.*/
        inout                   eGDIBYTEORDEReByteOrder
```

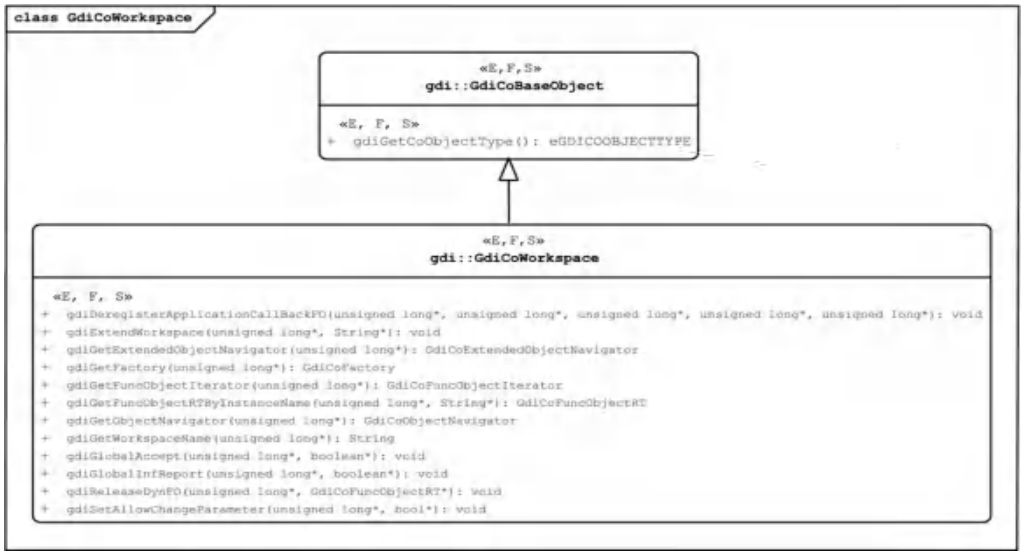
```
/*inparameter
definestherequestedByteOrderoftheApplication.
ThedefaultConstantisdefinedinthemanufacturerspecificCoordinator
environment.*/
):
```

功能说明

无缓冲 StreamObject 的构造方法。定义分配和字节序以及目的地协调器中的数据区。
此参数不得为 NIL。
流对象与 gdiWriteValue 或 gdiExecute 的 in 参数一起用作未缓冲的 In Reference。
更新流对象中的数据将影响对协调器内部存储器的直接修改。

A.2.29 [S, E, F] GdiCoWorkspace

该接口包含用于引用应用程序区域中现有 Function 对象的服务。可以通过功能对象的名称或工作空间中所有现有功能对象的迭代来检测功能对象。



图A.29 GdiCoWorkspace的层次结构图

A.2.29.1 GdiCoWorkspace :: «S,E,F» gdiDeregisterApplicationCallBackFO()

函数调用

```
GdiDeregisterApplicationCallBackFO
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   unsignedlongulFuncMem
        /*inparmeter
        TheFOvaluehastobespecifiedwithintheDCDofthedriver.*/
        inout                   unsignedlongrefApplicationFO
        /*inparameter
        Referenceorhandletotheapplication FO.Thisreferenceisgivenby theApplication.*/
        inout                   unsignedlongulFuMemId
        /*inparameter
        IdoftheApplicationCallBackCommunicationobjectaccordingtoDCD.
        */
        inout                   unsignedlongrefApplicationCO
        /*inparameter
        ReferenceorHandleoftheCOwhichisusedforaInformationReport.
```

```
            */
            ):void
```

功能说明

用其所有属性注销所指示的应用程序功能对象。

返回值

无

A.2.29.2 GdiCoWorkspace : : [S, E, F] gdiExtendWorkspace ()

函数调用

```
gdiExtendWorkspace
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsInstanceDescription
            /*inparameter
            Containsparameterizationdataorthenameofaparameterizationfile.
            */
            ):void
```

功能说明

功能描述中给出的实例用于扩展和更新现有工作空间。已经创建的对象可以更改，但是永远不会删除。

如果为给定的 PID 文件注册了与一致性相关的任何错误，或者在运行 PID 文件时发生错误，则不会扩展工作空间。所有受影响的 VD 仍处于状态准备中。

在第一种情况下，方法 gdiExtendWorkspace 引发异常 eINT_PID_FILE_MISMATCH（GdiCoInternalException），在第二种情况下，引发异常 ePAR_INCORRECT_PARAMETERIZATION（GdiCoParameterizationException）。

如果密钥不一致，则处理 PID 文件并扩展现有的工作区。在 PID 错误的情况下，行为如上所述，这意味着该过程将中止并且现有的工作区仍然有效，但是所有受影响的 VD 仍处于状态准备中。

返回值

无

A.2.29.3 GdiCoWorkspace :: «S,E,F» gdiGetExtendedObjectNavigator()

函数调用

```
gdiGetExtendedObjectNavigator
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoExtendedObjectNavigator
```

功能说明

如果给定的协调器支持扩展访问功能，则返回对 GdiCoExtendedObjectNavigator 的引用。

返回值

返回对 GdiCoExtendedObjectNavigator 的引用（如果存在）。

A.2.29.4 GdiCoWorkspace : : [S, E, F] gdiGetFactory ()

函数调用

```
gdiGetFactory
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
```

```
gdiGetWorkspaceByName()*/
):GdiCoFactory
```

功能说明

如果给定的协调器支持完全访问功能，则返回对 GdiCoFactory 的引用。

返回值

返回对 GdiCoDescriptionFactory 的引用（如果存在）。

A.2.29.5 GdiCoWorkspace :: «S,E,F» gdiGetFuncObjectIterator()

函数调用

```
gdiGetFuncObjectIterator
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):GdiCoFuncObjectIterator
```

功能说明

创建对迭代器的引用，该引用用于工作区所有已设置的 Function 对象的输出。

返回值

如果成功，则返回对 GdiFuncObjectIterator 的引用。

A.2.29.6 GdiCoWorkspace : : [S, E, F] gdiGetFuncObjecttbyInstanceName ()

函数调用

```
gdiGetFuncObjectRTByInstanceName
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                   StringsFOInstanceName
/*inparameter
IdentifierofexpectedInstance*/
):GdiCoFuncObjectRT
```

功能说明

创建对现有 Function 对象的引用。

返回值

返回对 GdiCoFuncObjectRT 的引用（如果存在）。

A.2.29.7 GdiCoWorkspace :: «S,E,F» gdiGetObjectNavigator()

函数调用

```
gdiGetObjectNavigator
(
inout                   unsigned long ulAppHnd
/* in parameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)
or by gdiGetWorkspaceByName() */
) : GdiCoObjectNavigator
```

功能说明

如果给定的协调器支持完全访问功能，则返回对 GdiCoObjectNavigator 的引用。

返回值

返回对 GdiCoObjectNavigator 的引用（如果存在）。

A.2.29.8 GdiCoWorkspace : : [S, E, F] gdiGetWorkspaceName ()

函数调用

```
gdiGetWorkspaceName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

功能说明

返回工作空间的标识符。

返回值

以字符串形式返回 Workspacename。

A.2.29.9 GdiCoWorkspace : : [S, E, F] gdiGlobalAccept ()

函数调用

```
gdiGlobalAccept
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  booleanbEnableControl
          /*true=enableAcceptCallbacksofthisCommunicationObjectswishwas localenabled.
          false=disableallAcceptCallbacks*/
          ):void
```

功能说明

对工作区中所有 VD 中的所有通信对象启用或禁用接受回调。

返回值

无

A.2.29.10 GdiCoWorkspace :: «S,E,F» gdiGlobalInfReport()

函数调用

```
gdiGlobalInfReport
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  booleanbEnableControl
          /*true=enableInformationReportsofthisCommunicationObjectswish
          waslocalenabled.
          false=disableallInformationReports*/
          ):void
```

功能说明

为工作区中所有 VD 中的所有通信对象启用或禁用信息报告回调。

返回值

无

A.2.29.11 GdiCoWorkspace :: «S,E,F» gdiReleaseDynFO()

函数调用

```
gdiReleaseDynFO (
          inout                  unsignedlongulAppHnd
          /*inparameter
```

```
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                   GdiCoFuncObjectRTrefFuncObject
/*inparameter
ReferencetoaFunctionobjectwithintheVD*/
):void
```

功能说明

如果应用程序不再需要动态功能对象，则此操作将释放协调器中的功能对象。

协调器将删除设备驱动程序中的相应功能对象及其所有通信对象，而无需进行任何转换。如果功能对象的操作，属性或参数正在进行任何活动，将生成一个 Exeption。

如果从应用程序重复访问，将生成错误。

返回值

没有

A.2.29.12 GdiCoWorkspace : : [S, E, F] gdiSetAllowChangeParameter ()

函数调用

```
gdiSetAllowChangeParameter
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   boolbEnable
        /*inparameter
        controlstheabilityoftheapplicationtomodifyparameterafterthe instantiationphase.*/
        ):void
```

功能说明

宣布在实例化阶段之后，应用程序将更改参数（通信对象）。

如果 bEnable = TRUE，则允许应用程序写参数。至少在对参数的写入过程中，受影响的 VD 将被转移到状态“修订”。

如果 bEnable = FALSE，则所有处于“修订”状态的受影响的 VD 都将转移到“工作”状态。如果无法进行转换，则将引发 GdiCoDriverException。

参数无法再更改。

返回值

无

A.2.30 [S, E, F] GdiCoWorkspaceInfo

该界面包含有关工作空间的信息。每个 wokspace 连接到一个 GdiCoWorkspaceInfo 对象。可以通过 GdiCoWorkspaceInfoIterator 确定工作区信息。

图A.30 GdiCoWorkspaceInfo的层次结构图

A.2.30.1 GdiCoWorkspaceInfo :: «S,E,F» getWorkspaceAccessState()

函数调用

```
getWorkspaceAccessState
        (
        inout                   unsignedlongulSec
        /*outparameter
        containstheelapsedtimeofthereturnedworkspacestate.*/
        ):eGDIACCESSSTATE
```

功能说明

此方法返回工作空间的当前访问状态。

注意　访问状态仅是快照。由于另一个应用程序可以访问该工作区，因此不能保证在接下来的几秒钟内可以访问该工作区。

返回值

返回工作空间的当前访问状态：

eNOT_USED，如果工作空间可用；

eUSED，如果另一个客户端正在使用工作空间。

A.2.30.2 GdiCoWorkspaceInfo : : [S, E, F] getWorkspaceName ()

函数调用

```
getWorkspaceName
        (
        ):String
```

功能说明

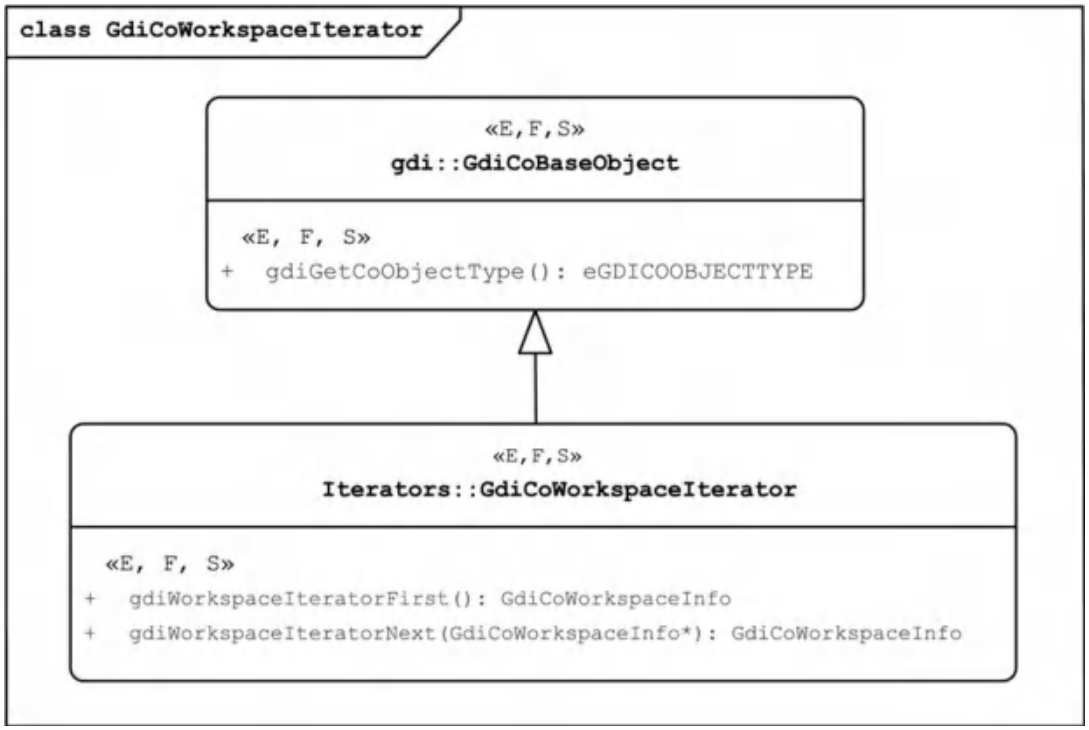
此方法返回工作区的实例名称。

返回值

返回工作空间实例的名称。

A.2.31 «S,E,F» GdiCoWorkspaceIterator

该接口包含用于工作空间迭代器应用程序的服务，该服务是在协调器中设置的所有工作空间的顺序输出。请注意，通过此服务，第二个应用程序可能在协调器内获得对 Function 对象的主权，而不会让工作区的所有者（应用程序）看到这一点。此服务应仅由诊断应用程序使用。



图A.31　GdiCoWorkspaceIterator的层次结构图

A.2.31.1 GdiCoWorkspaceIterator :: «S,E,F» gdiWorkspaceIteratorFirst()

函数调用

```
gdiWorkspaceIteratorFirst
          (
          ):GdiCoWorkspaceInfo
```

功能说明

将内部指针设置为协调器内的第一个工作区。

返回值

返回对 Iterator 的引用，该引用指向第一个 GdiCoWorkspaceInfo（如果存在），否则返回 NIL 引用。

A.2.31.2 GdiCoWorkspaceIterator :: «S,E,F» gdiWorkspaceIteratorNext()

函数调用

```
gdiWorkspaceIteratorNext
          (
          inout                  GdiCoWorkspaceInforefLastElement
          /*inparameter
          A reference to a element returned by gdiWorkspaceIteratorFirst or
          gdiWorkspaceIteratorNext.*/
          ):GdiCoWorkspaceInfo
```

功能说明

检测对协调器中当前工作区（参数）的引用。将内部指针设置为下一个工作区（如果存在）并返回引用。

返回值

返回对 GdiCoWorkspaceInfo 的引用，该引用指向下一个 GdiCoWorkspace（如果存在），否则返回 NIL 引用。

附 录 B
（规范性）
编程参考指南—扩展访问接口

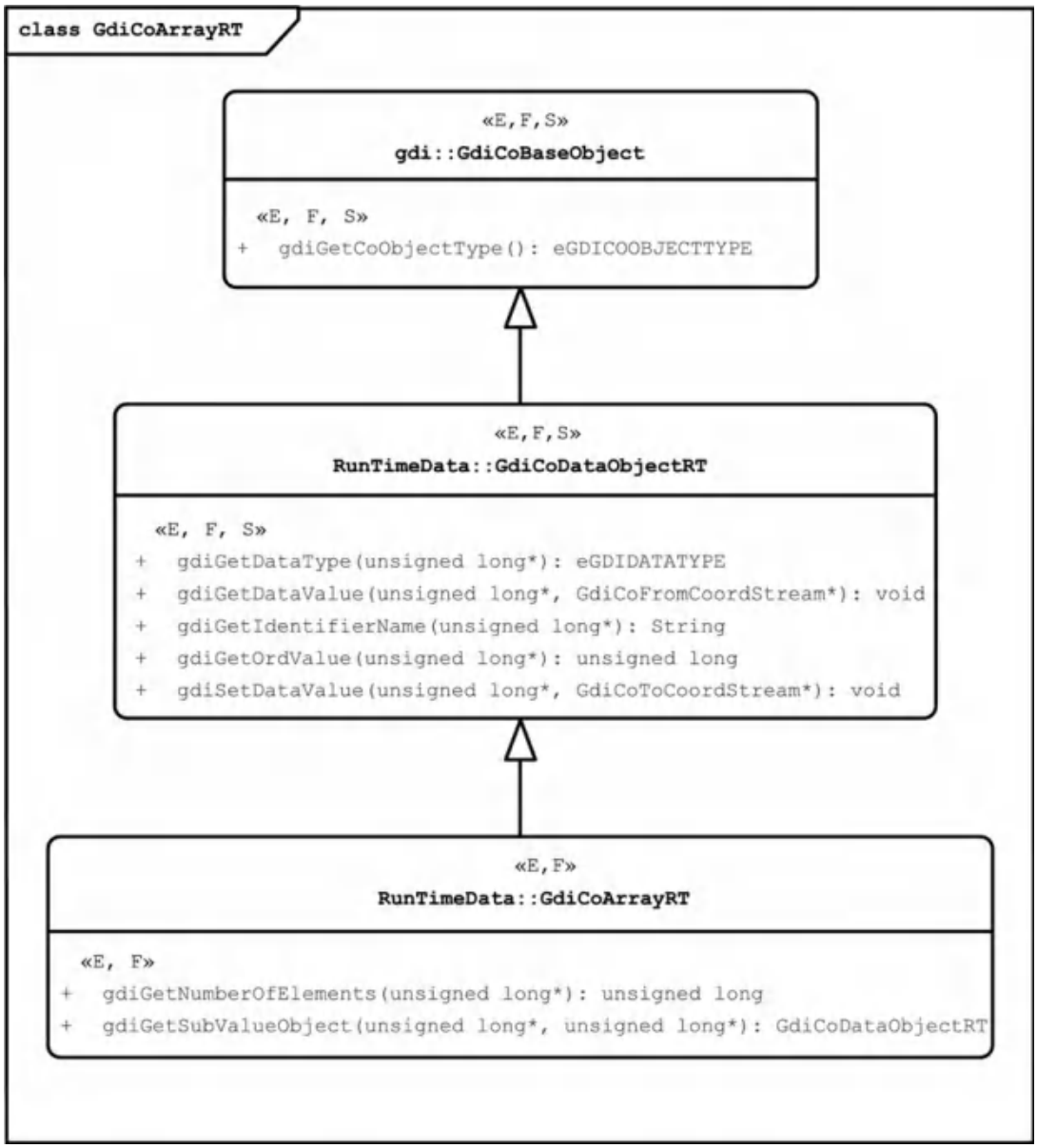
B.1 一般信息

标有«E»的扩展访问接口的单个接口（分别是类及其方法和属性）也应属于标有«F»的完全访问接口。因此，所有单个接口都标记有«E，F»。附录 A 中描述的智能访问接口的所有单个接口均应包含在扩展访问接口中，本附件中不再赘述。以下详细描述由图 B.1 至 B.17 完成，以概述类及其方法。

B.2 扩展访问接口的详细描述

B.2.1 «E,F» GdiCoArrayRT

此接口包含用于处理 Array 类型的服务。



图B.1 GdiCoArrayRT的层次结构图

B.2.1.1 GdiCoArrayRT :: «E,F» gdiGetNumberOfElements()

函数调用

gdiGetNumberOfElements

```
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):unsignedlong
```

功能说明

它用于检测数组元素的数量。

返回值

返回数组的元素数

B.2.1.2 GdiCoArrayRT :: «E,F» gdiGetSubValueObject()

函数调用

```
gdiGetSubValueObject
          (
          inout                   unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                   unsignedlongulIndex
          /*inparameter
          Indexvalueofthedataelement*/
          ):GdiCoDataObjectRT
```

功能说明

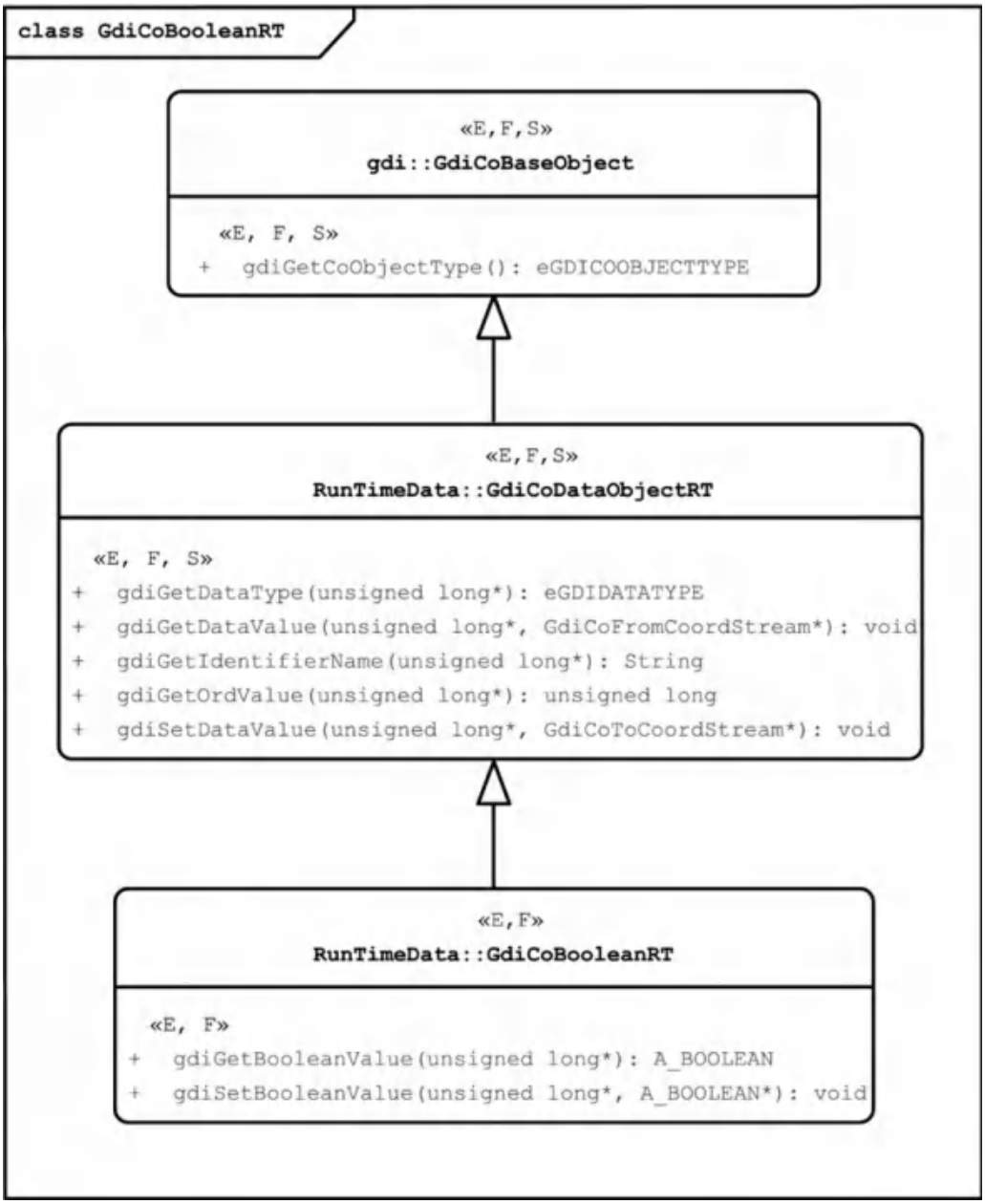
此活动将创建一个子值对象。

返回值

返回对数组的子值对象的引用。

B.2.2 «E,F» GdiCoBooleanRT

该接口包含用于处理布尔类型数据的服务。



图B.2 GdiCoBooleanRT的层次结构图

B.2.2.1 GdiCoBooleanRT :: ::«E, F»gdiGetBooleanValue ()

函数调用

```
gdiGetBooleanValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_BOOLEAN
```

功能说明

返回布尔数据的值。这也可能是数据的布尔类型的子元素（数组，序列，结构，联合）。

返回值

将（Sub）基准的布尔值返回为 A_BOOLEAN：

0 = FALSE;

>0= TRUE。

B.2.2.2 Gdicobooleanrt : : [E, F] gdiSetbooleanvalue ()

函数调用

```
gdiSetBooleanValue
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                    A_BOOLEANucValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

功能说明

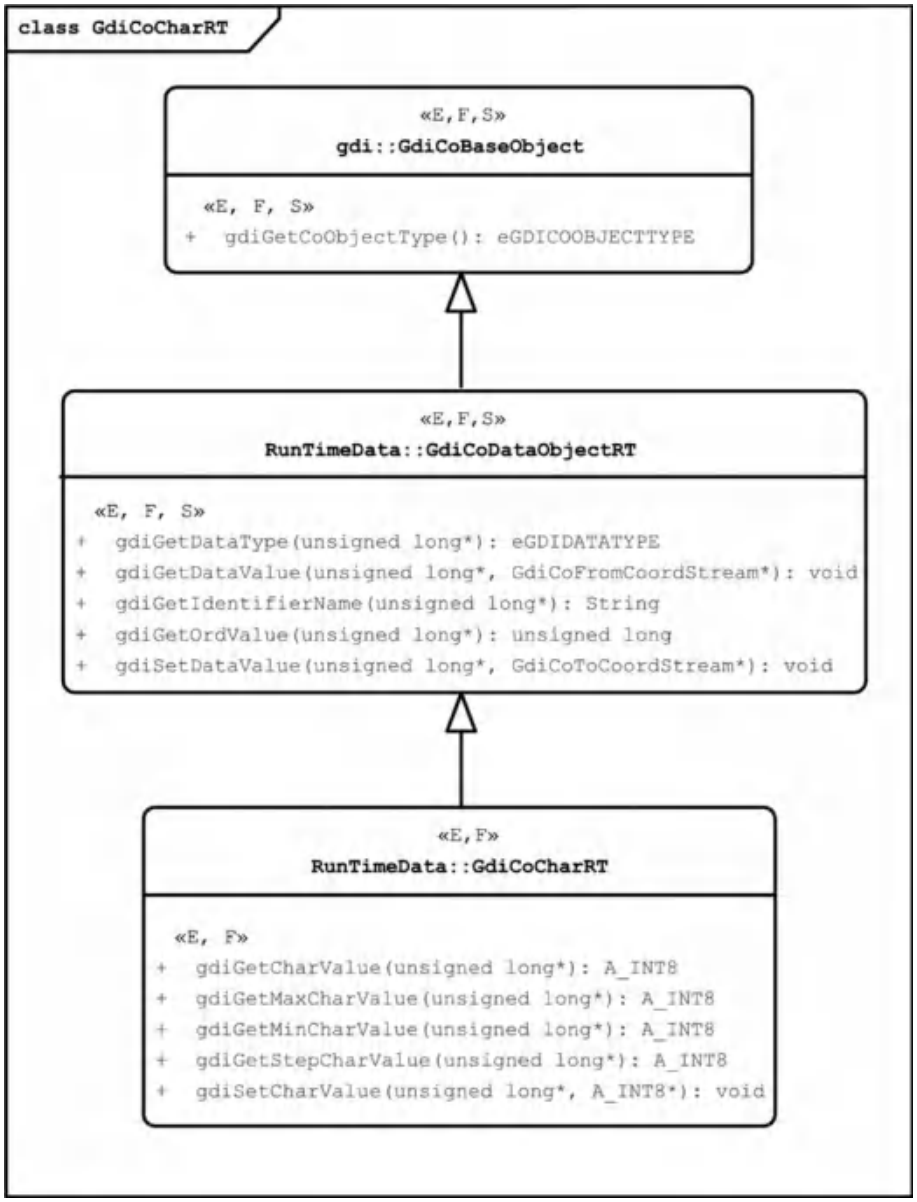
将布尔值复制到协调器的数据区域。该服务还用于用值（数组，序列，结构，联合的元素）填充基准的子区域。

返回值

无

B.2.3《 E, F》 GdiCoCharRT

此接口包含用于检测 Char 类型的限制的服务。所有其他标量类型都包含等效服务，并根据基本类型返回值。



图B.3 GdiCoCharRT的层次结构图

B.2.3.1 GdiCoCharrt : : [E, F] gdiGetCharvalue ()

函数调用

```
gdiGetCharValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_INT8
```

函数描述

返回 Char 数据的值。也可能是数据（数组、序列、结构体、联合体）的 Char 类型的子元素。

返回值

返回（子）数据的 Char 值。

B.2.3.2 GdiCoCharRT :: «E,F» gdiGetMaxCharValue()

函数调用

```
gdiGetMaxCharValue
        (
```

```
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):A_INT8
```

函数描述

根据数据类型，返回数据的最大值。如果未明确设置最大值，则取值为 127。

返回值

根据数据类型，返回数据的最大值.

B.2.3.3 GdiCoCharRT :: «E,F» gdiGetMinCharValue()

函数调用

```
gdiGetMinCharValue
          (
          inout                   unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT8
```

函数描述

根据数据类型，返回数据的最小值。如果未明确设置最大值，则取值为-128。

返回值

根据数据类型，返回数据的最小值。

B.2.3.4 GdiCoCharRT :: «E,F» gdiGetStepCharValue()

函数调用

```
gdiGetStepCharValue
          (
          inout                   unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT8
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据类型，返回其步长。

B.2.3.5 GdiCoCharRT :: «E,F» gdiSetCharValue()

函数调用

```
gdiSetCharValue
          (
          inout                   unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                   A_INT8cValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

函数描述

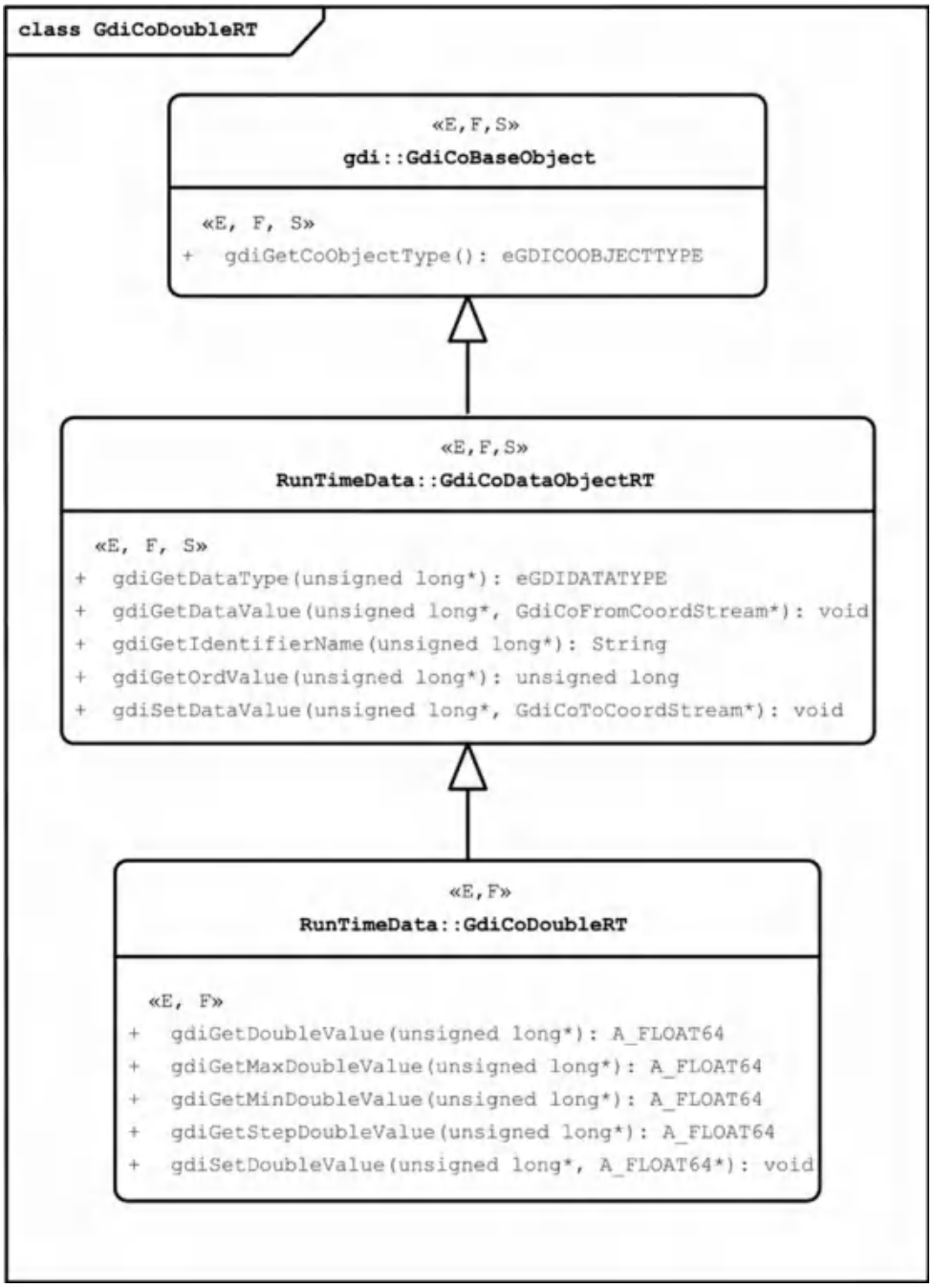
将 Char 值复制到协调器的数据域。该服务也用于用值（数组、序列、结构体、联合体的元素）填充数据的子域。

返回值

无。

B.2.4 «E,F» GdiCoDoubleRT

此接口包含用于处理 Double 类型的服务。



图B.4 GdiCoDoubleRT的层级图

B.2.4.1 GdiCoDoubleRT :: «E,F» gdiGetDoubleValue()

函数调用

```
gdiGetDoubleValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
```

```
        ):A_FLOAT64
```

函数描述

返回 Double 型数据的值，也可能是一个数据（数组、序列、结构体、联合体）的子元素。

返回值

返回（子）数据的 Double 型的值。

B.2.4.2 GdiCoDoubleRT :: «E,F» gdiGetMaxDoubleValue()

函数调用

```
gdiGetMaxDoubleValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_FLOAT64
```

函数描述

根据数据类型，返回数据的最大值。如果没有明确设置最大值，则取 17976931348623158 E+308。

返回值

根据数据类型，返回数据的最大值。

B.2.4.3 GdiCoDoubleRT :: «E,F» gdiGetMinDoubleValue()

函数调用

```
gdiGetMinDoubleValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_FLOAT64
```

函数描述

根据数据类型，返回数据的最小值。如果没有明确设置最小值，则取 2,2250738585072014 E-308。

返回值

根据数据类型，返回数据的最小值。

B.2.4.4 GdiCoDoubleRT :: «E,F» gdiGetStepDoubleValue()

函数调用

```
gdiGetStepDoubleValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_FLOAT64
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据数据类型，返回其步长。

B.2.4.5 GdiCoDoubleRT :: «E,F» gdiSetDoubleValue()

函数调用

```
gdiSetDoubleValue
        (
```

```
inout                    unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                    A_FLOAT64dValue
/*inparameter
referencetothevaluetobeset.*/
):void
```

函数描述

将 Double 值复制到协调器的数据域。该服务也用于用值（数组、序列、结构体、联合体的元素）填充数据的子域。

返回值

空。

B.2.5 «E,F» GdiCoEnumRT

该接口包含用于处理设置类型枚举的服务。

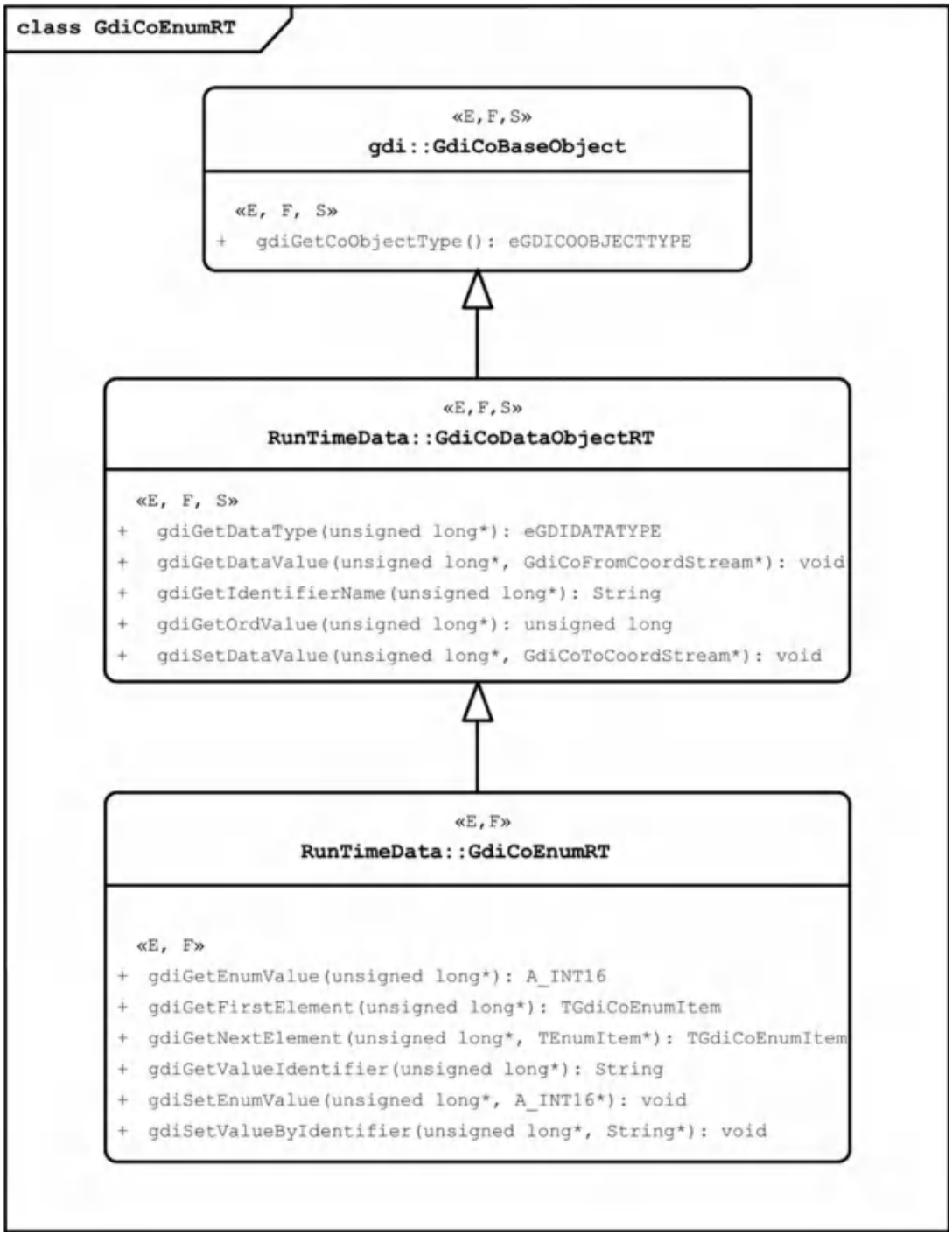


图 B.5 —GdiCoEnumRT层次图

B.2.5.1 GdiCoEnumRT :: «E,F» gdiGetEnumValue()

函数调用

```
gdiGetEnumValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_INT16
```

函数描述

返回枚举类型数据的值。也可能是一个枚举类型数据（数组、序列、结构体、联合体）的子元素。

返回值

.返回（子）数据的 short 值。

B.2.5.2 GdiCoEnumRT :: «E,F» gdiGetFirstElement()

函数调用

```
gdiGetFirstElement
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):TGdiCoEnumItem
```

函数描述

返回单个元素枚举的第一个元素，通常是最低值的元素。返回的元素是由子元素 String 和 short 组成的结构体。

返回值

返回是由子元素 String 和 short 组成的结构体。如果没有有效元素，则“ noGDIElement”作为返回值名称，并返回数字值 32768。

B.2.5.3 GdiCoEnumRT :: «E,F» gdiGetNextElement()

函数调用

```
gdiGetNextElement
          (
          inout                  unsigned long ulAppHnd
          /* in parameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)
          or by gdiGetWorkspaceByName() */
          inout                  TEnumItem refLastElement
          /* in parameter
          contains the reference to the last Enumeration Description. */
          ) : TGdiCoEnumItem
```

函数描述

返回单个元素枚举的下一个元素。返回的元素是由子元素 String 和 short 组成的结构体。

返回值

如果没有有效元素，则“noGDIElement”作为返回值名称，并返回数字值 32768。

B.2.5.4 GdiCoEnumRT :: «E,F» gdiGetValueIdentifier()

函数调用

```
gdiGetValueIdentifier
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

函数描述

以字符串返回 Enumeration 数据值的标识符。也可能是数据（数组、序列、结构体、联合体）的 short 型子元素。

返回值

将当前值作为（Sub）数据的字符串（枚举成员）返回值。

如果无有效元素，则返回字符串“ noGDIElement”。

B.2.5.5 GdiCoEnumRT :: «E,F» gdiSetEnumValue()

函数调用

```
gdiSetEnumValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_INT16nValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

函数描述

将 Short 型值复制到协调器的数据域。该服务也用于用值（数组、序列、结构体、联合体的元素）填充数据的子域。

返回值

空。

B.2.5.6 GdiCoEnumRT :: «E,F» gdiSetValueByIdentifier()

函数调用

```
gdiSetValueByIdentifier
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  StringszValueIdentifier
          /*inparameter
          Identifieroftheenumerationelement.*/
          ):void
```

函数描述

复制相应的 Short 值到协调器的数据域.该服务也用于用值（数组、序列、结构体、联合体的元素）填充数据的子域。

返回值

空。

B.2.6 «E,F» GdiCoExtendedObjectNavigator

该静态接口包含用于迭代 GDI 对象和 GDI 类型的服务。

图 B.6 —GdiCoExtendedObjectNavigator 层次图

B.2.6.1 GdiCoExtendedObjectNavigator :: «E,F» gdiExecuteOperationData()

函数调用

```
gdiExecuteOperationData
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoOperationRTrefOperationRT
        /*inparameter
        containsthereferencetotheOperation,towhichthecommunication shallbeperformed.*/
        inout                   GdiCoDriverResultrefDestInfo
        /*outparameter
        contains the reference to the Information Object, warnings and
        informationsoftheDriverwillbestoredinit.*/
        inout                   TEXEFunctionReferencerefCBCompleteExec
        /*inparameter
        Referencetoacallbackmethodwhichhandles                  theexecutionmechanism.
        ANILreferencesignalsasynchronouscall.*/
        ):short
```

函数描述

该服务开始一个操作，输入和返回值保留在协调器的区域内。

返回值

描述由 GDI 驱动程序执行的通信：

0 =同步返回；

1 =异步返回；

小于 0 =错误。

B.2.6.2 GdiCoExtendedObjectNavigator :: «E,F» gdiGetOperationInDataObjectRT()

函数调用

```
gdiGetOperationInDataObjectRT (
        inout                   unsignedlongulAppHnd
```

```
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoOperationRTrefOperationRT
        /*inparameter
        containsthereferencetotheGdiCoOperationRTobject.*/
        ):GdiCoDataObjectRT
```

函数描述

该服务返回对给定操作的输入数据类型的引用。

返回值

如果输入数据有效，返回 GdiCoDataObjectRT，如果无输入数据，返回 0。

B.2.6.3 GdiCoExtendedObjectNavigator :: «E,F» gdiGetOperationOutDataObjectRT()

函数调用

```
gdiGetOperationOutDataObjectRT (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoOperationRTrefOperationRT
        /*inparameter
        containsthereferencetotheGdiCoOperationRTobject.*/
        ):GdiCoDataObjectRT
```

函数描述

该服务返回对给定操作的输出数据类型的引用.

返回值

如果输出数据存在，返回 GdiCoDataObjectRT 的引用。如果没有输出数据，则返回 0。

B.2.6.4 GdiCoExtendedObjectNavigator :: «E,F» gdiReadCommObjectData()

函数调用

```
gdiReadCommObjectData
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoCommObjectRTrefCommObjectRT
        /*inparameter
        contains the reference to the Communication Object to which the
        communicationwillbeexecuted.*/
        inout                   GdiCoDriverResultrefDestInfo
        /*outparameter
        contains the reference to the Information Object, warnings and
        informationsoftheDriverwillbestoredinit.*/
        inout                   TCBFunctionReferencerefCBCompleteRead
        /*inparameter
        Referencetoacallbackmethodwhichhandlesthereadmechanism.ANIL
        referencesignalsasynchronouscall.*/
        ):short
```

函数描述

开始与驱动程序（GdiRead）进行数据交换。

使用 GdiDataObject 内部的存储数据进行写入或在读取过程中将日期存储到 GdiDataObject。

引导协调器和驱动程序之间的数据交换。

返回值

描述由 GDI 驱动程序执行的通信：

0 =同步返回；

1 =异步返回。

B.2.6.5 GdiCoExtendedObjectNavigator :: «E,F» gdiWriteCommObjectData()

FunctionCall 函数调用

```
gdiWriteCommObjectData
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoCommObjectRTrefCommObjectRT
          /*inparameter
          contains the reference to the Communication Object to which the
          communicationwillbeexecuted.*/
          inout                  GdiCoDriverResultrefDestInfo
          /*outparameter
          contains the reference to the Information Object, warnings and
          informationsoftheDriverwillbestoredinit.*/
          inout                  TCBFunctionReferencerefCBCompleteWrite
          /*inparameter
          Referencetoacallbackmethodwhichhandlesthewritemechanism.ANIL
          referencesignalsasynchronouscall.*/
          ):short
```

函数描述

开始与驱动程序（GdiWrite）进行数据交换。

使用 GdiDataObject 内部的存储数据进行写入或在读取过程中将日期存储到 GdiDataObject。

引导协调器和驱动程序之间的数据交换。

返回值

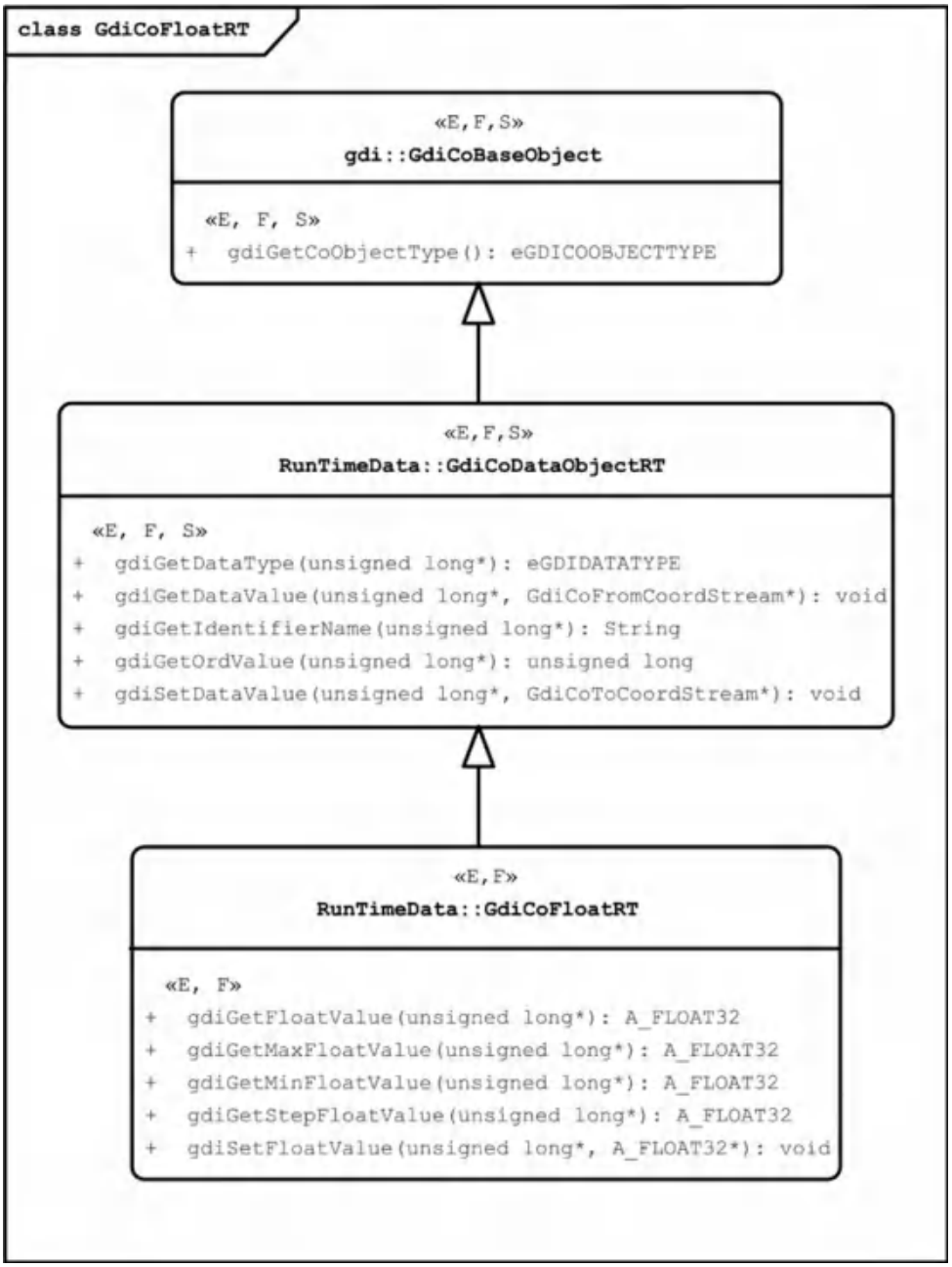
描述由 GDI 驱动程序执行的通信：

0 =同步返回；

1 =异步返回。

B.2.7 «E,F» GdiCoFloatRT

该接口包含用于处理 Float 类型的服务。



图B.7 GdiCoFloatRT的层次图

B.2.7.1 GdiCoFloatRT :: «E,F» gdiGetFloatValue()

函数调用

```
gdiGetFloatValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_FLOAT32
```

函数描述

ReturnsthevalueofaFloatdatum.Thismightalsobeasub-elementofthetypeCharofadatum(Array, Sequence,Structure,Union).

返回一个 Float 型数据的值。也可能是一个数据的子元素。

返回值

返回（子）数据的 Float 值。

B.2.7.2 GdiCoFloatRT :: «E,F» gdiGetMaxFloatValue()

函数调用

```
gdiGetMaxFloatValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_FLOAT32
```

函数描述

根据数据类型，返回数据的最大值。如果没有明确设置最大值，则取 3,402823466 E + 38。

返回值

根据数据类型，返回数据的最大值。

B.2.7.3 GdiCoFloatRT :: «E,F» gdiGetMinFloatValue()

函数调用

```
gdiGetMinFloatValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_FLOAT32
```

函数描述

根据数据类型，返回数据的最小值。如果没有明确设置最小值，则取 1,175494351 E -38.

返回值

根据数据类型，返回数据的最小值。

B.2.7.4 GdiCoFloatRT :: «E,F» gdiGetStepFloatValue()

函数调用

```
gdiGetStepFloatValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_FLOAT32
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据类型，返回数据的步长。

B.2.7.5 GdiCoFloatRT :: «E,F» gdiSetFloatValue()

函数调用

```
gdiSetFloatValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_FLOAT32fValue
          /*inparameter
```

referencetothevaluetobeset.*/
):void

函数描述

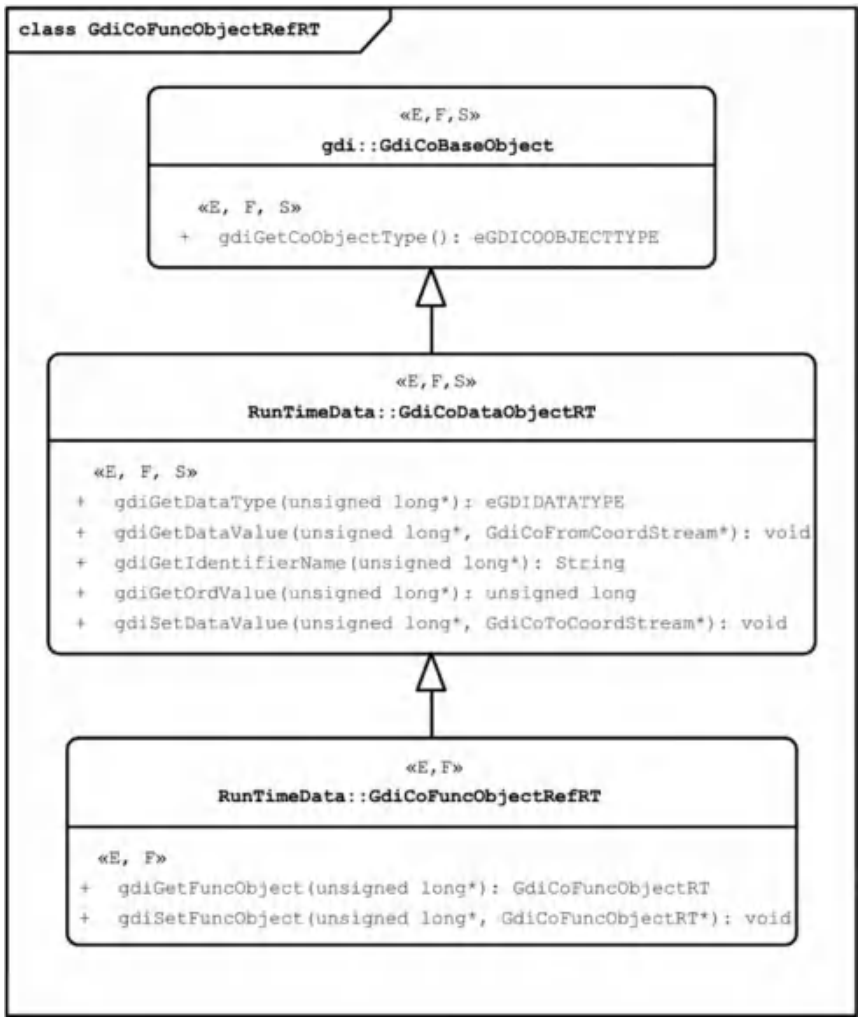
复制相应的 Float 值到协调器的数据域.该服务也用于用（数组、序列、结构体、联合体的元素）值填充数据的子域。

返回值

空

B.2.8 «E,F» GdiCoFuncObjectRefRT

该接口应包含用于处理 GdiCoFuncObjRefRT 类型的服务。引用类型应与 gdiWorkspace 的 gdiGetFuncObjectRTByName 方法的结果类型相同。引用的功能对象应由协调器实例化。如果通过写操作更改了这种类型的属性，则不应删除引用的功能对象该属性应引用另一个功能对象。



图B.8 GdiCoFuncObjectRefRT的层级图

B.2.8.1 GdiCoFuncObjectRefRT :: «E,F» gdiGetFuncObject()

函数调用

gdiGetFuncObject
(
inout unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):GdiCoFuncObjectRT

函数描述

返回数据的值（FO 实例引用）。这也可能是数据（数组、序列、结构体、联合体）的 Char 类型的子元素。

返回值

返回函数对象的引用，这也可能是（数组、序列、结构体、联合体）数据的 FoReference 类型的子元素

B.2.8.2 GdiCoFuncObjectRefRT :: «E,F» gdiSetFuncObject()

函数调用

```
gdiSetFuncObject
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoFuncObjectRTrefFuncObject
        /*inparameter
        referenceofafuncobject*/
        ):void
```

函数描述

将函数对象的句柄和 id 复制到协调器的数据区域。该服务还用于用值（数组，序列，结构，联合的元素）填充基准的子区域。

返回值

空。

B.2.9 «E,F» GdiCoLongRT

此接口包含用于处理 Long 类型的服务。

图 B.9 GdiCoLongRT 的层级图

B.2.9.1 GdiCoLongRT :: «E,F» gdiGetLongValue()

函数调用

```
gdiGetLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

返回 Long 型数据的值。这也可能是数据（数组、序列、结构体、联合体）的 Char 类型的子元素。

返回值

返回（子）数据的 Long 值.

B.2.9.2 GdiCoLongRT :: «E,F» gdiGetMaxLongValue()

函数调用

```
gdiGetMaxLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

根据数据类型，返回数据的最大值。如果没有明确设置最大值，则取 2147483647。

返回值

根据数据类型，返回数据的最大值。

B.2.9.3 GdiCoLongRT :: «E,F» gdiGetMinLongValue()

函数调用

```
gdiGetMinLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

根据数据类型，返回数据的最小值。如果没有明确设置最大值，则取 - 2147483648.。

返回值

根据数据类型，返回数据的最小值。

B.2.9.4 GdiCoLongRT :: «E,F» gdiGetStepLongValue()

函数调用

```
gdiGetStepLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为1。

返回值

根据类型，返回数据的步长。

B.2.9.5 GdiCoLongRT :: «E,F» gdiSetLongValue()

函数调用

```
gdiSetLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_INT32lValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

函数描述

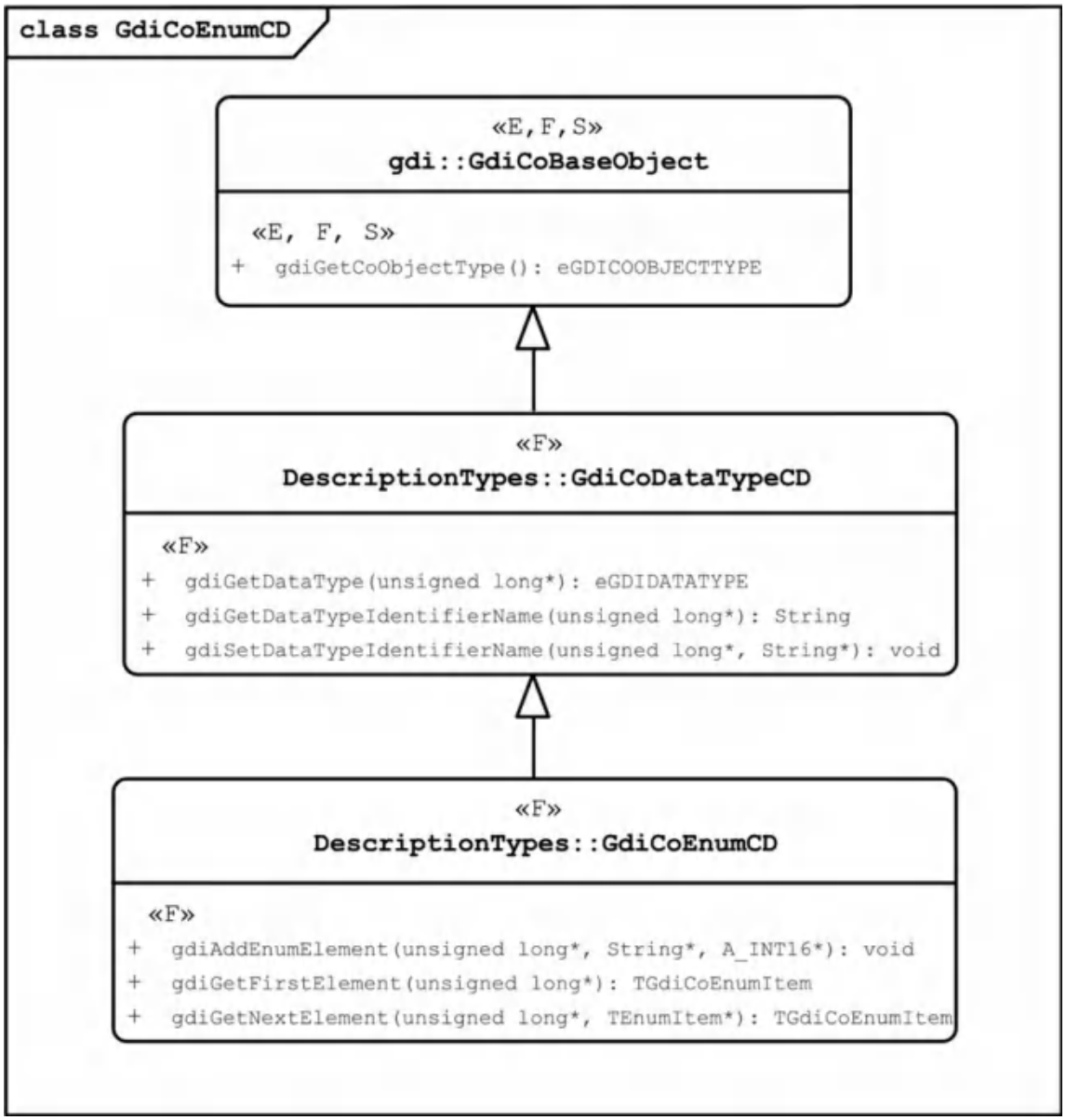
将一个Long值复制到协调器的数据区域。该服务还用于用值（数组、序列、结构体、联合体的元素）填充数据的子区域。

返回值

空。

B.2.10 «E,F» GdiCoOctetRT

此接口包含用于处理八位位组类型（无符号字符）的服务。



图B.10 GdiCoOctetRT的层级图

B.2.10.1 GdiCoOctetRT :: «E,F» gdiGetMaxOctetValue()

函数调用

```
gdiGetMaxOctetValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_UINT8
```

函数描述

根据数据类型，返回数据的最大值。如果没有明确设置最大值，则取 255。

返回值

根据数据类型，返回数据的最大值。

B.2.10.2 GdiCoOctetRT :: «E,F» gdiGetMinOctetValue()

函数调用

```
gdiGetMinOctetValue
        (
```

```
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_UINT8
```

函数描述

根据数据类型，返回数据的最小值。如果没有明确设置最小值，则取 0。

返回值

根据数据类型，返回数据的最小值。

B.2.10.3 GdiCoOctetRT :: «E,F» gdiGetOctetValue()

函数调用

```
gdiGetOctetValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_UINT8
```

函数描述

返回一个无符号 Char 型数据的值，这也可能是数据（数组、序列、结构体、联合体）的 Char 类型的子元素。

返回值

返回（子）数据的无符号 Char 值。

B.2.10.4 GdiCoOctetRT :: «E,F» gdiGetStepOctetValue()

函数调用

```
gdiGetStepOctetValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_UINT8
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据数据类型，返回数据的步长。.

B.2.10.5 GdiCoOctetRT :: «E,F» gdiSetOctetValue()

函数调用

```
gdiSetOctetValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_UINT8ucValue
            /*inparameter
            referencetothevaluetobeset.*/
            ):void
```

函数描述

将一个无符号 Char 值复制到协调器的数据区域。该服务还用于用值（数组、序列、结构体、联合体的元

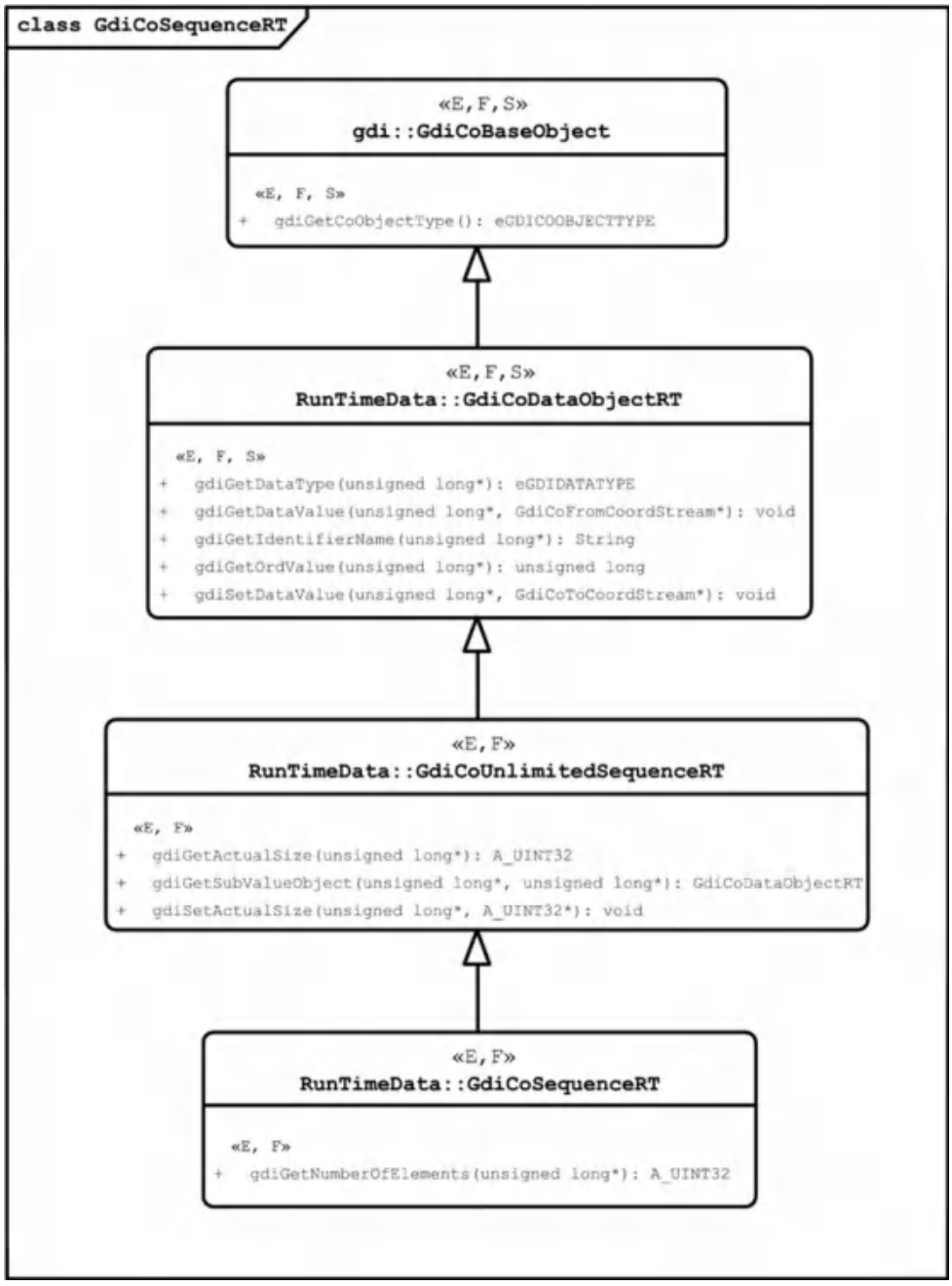
素）填充数据的子区域。

返回值

空。

B.2.11 «E,F» GdiCoSequenceRT

该接口包含用于处理序列类型的服务。



图B.11 GdiCoSequenceRT的层级图

B.2.11.1 GdiCoSequenceRT :: «E,F» gdiGetNumberOfElements()

函数调用

```
gdiGetNumberOfElements
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
```

```
            gdiGetWorkspaceByName()*/
            ):A_UINT32
```

函数描述

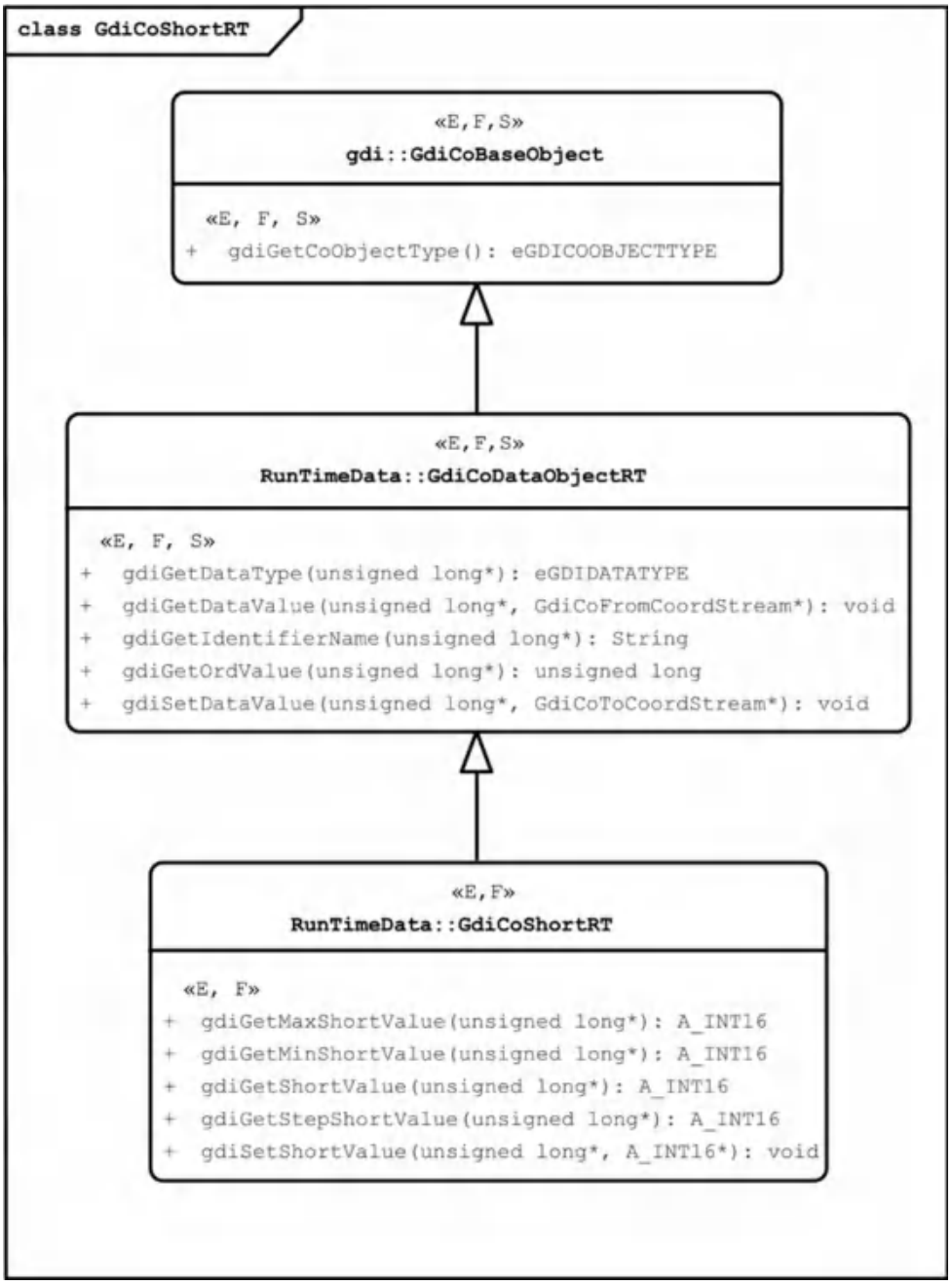
用于检测序列的最大元素数。

返回值

返回序列的最大元素数.

B.2.12 «E,F» GdiCoShortRT

此接口包含用于处理 Short 类型的服务。



图B.12 GdiCoShortRT的层级图

B.2.12.1 GdiCoShortRT :: «E,F» gdiGetMaxShortValue()

函数调用

```
gdiGetMaxShortValue
            (
```

```
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_INT16
```

函数描述

根据数据类型，返回数据的最大值。如果没有明确设置最大值，则取 32767.。

返回值

根据数据类型，返回数据的最大值。

B.2.12.2 GdiCoShortRT :: «E,F» gdiGetMinShortValue()

函数调用

```
gdiGetMinShortValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_INT16
```

函数描述

根据数据类型，返回数据的最小值。如果没有明确设置最小值，则取-32768。

返回值

根据数据类型，返回数据的最小值。

B.2.12.3 GdiCoShortRT :: «E,F» gdiGetShortValue()

函数调用

```
gdiGetShortValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_INT16
```

函数描述

返回一个 Short 型数据的值，这也可能是数据（数组、序列、结构体、联合体）的 Char 类型的子元素。

返回值

返回（子）数据的 Short。

B.2.12.4 GdiCoShortRT :: «E,F» gdiGetStepShortValue()

函数调用

```
gdiGetStepShortValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):A_INT16
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据数据类型，返回数据的步长。

B.2.12.5 GdiCoShortRT :: «E,F» gdiSetShortValue()

函数调用

```
gdiSetShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_INT16nValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

函数描述

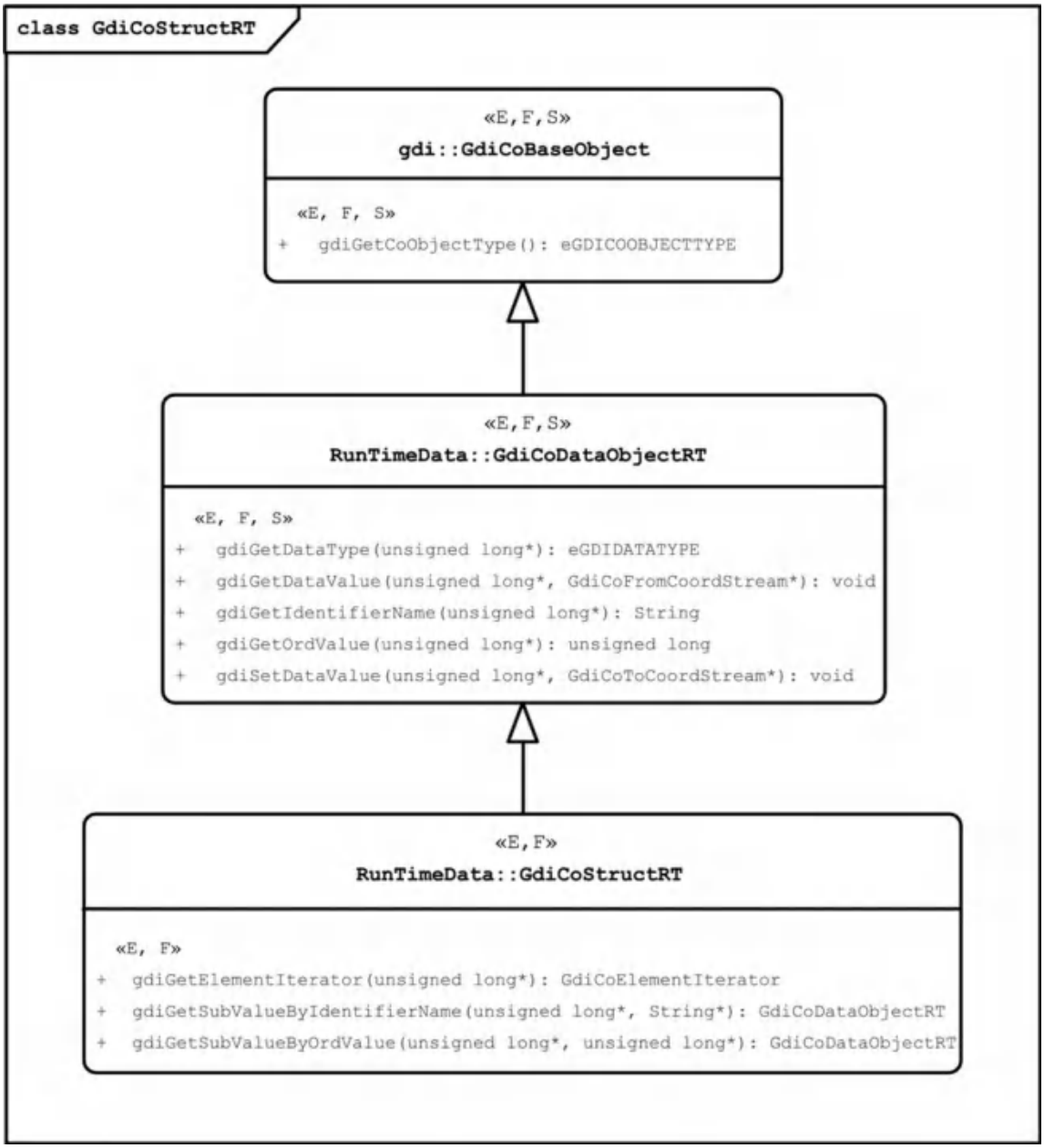
将一个 Short 值复制到协调器的数据区域。该服务还用于用值（数组、序列、结构体、联合体的元素）填充数据的子区域。

返回值

空。

B.2.13 «E,F» GdiCoStructRT

该接口包含用于处理结构体类型的服务。



图B.13 GdiCoStructRT的层级图

B.2.13.1 GdiCoStructRT :: «E,F» gdiGetElementIterator()

函数调用

```
gdiGetElementIterator
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoElementIterator
```

函数描述

生成元素迭代器，通过该元素迭代器可以检测结构体的所有子元素。迭代器被重置。

返回值

返回对元素迭代器的引用（指针、句柄、类）。

B.2.13.2 GdiCoStructRT :: «E,F» gdiGetSubValueByIdentifierName()

函数调用

```
gdiGetSubValueByIdentifierName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  StringsElementName
          /*inparameter
          ElementIdentifierofthereferencedstructureelement.*/
          ):GdiCoDataObjectRT
```

函数描述

返回对结构体元素的引用。返回数据对象实例作为基本类型的引用。该名称用作选择器。可以通过gdiGetType（）检测结构体元素的实际类型（如果未知）。然后，该引用可以用作对检测到的数据类型的引用。

返回值

返回数据的引用，如果不存在的话，返回NIL。

B.2.13.3 GdiCoStructRT :: «E,F» gdiGetSubValueByOrdValue()

函数调用

```
gdiGetSubValueByOrdValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  unsignedlongulSection
          /*inparameter
          Ordinalvalueofthestructureelement.*/
          ):GdiCoDataObjectRT
```

函数描述

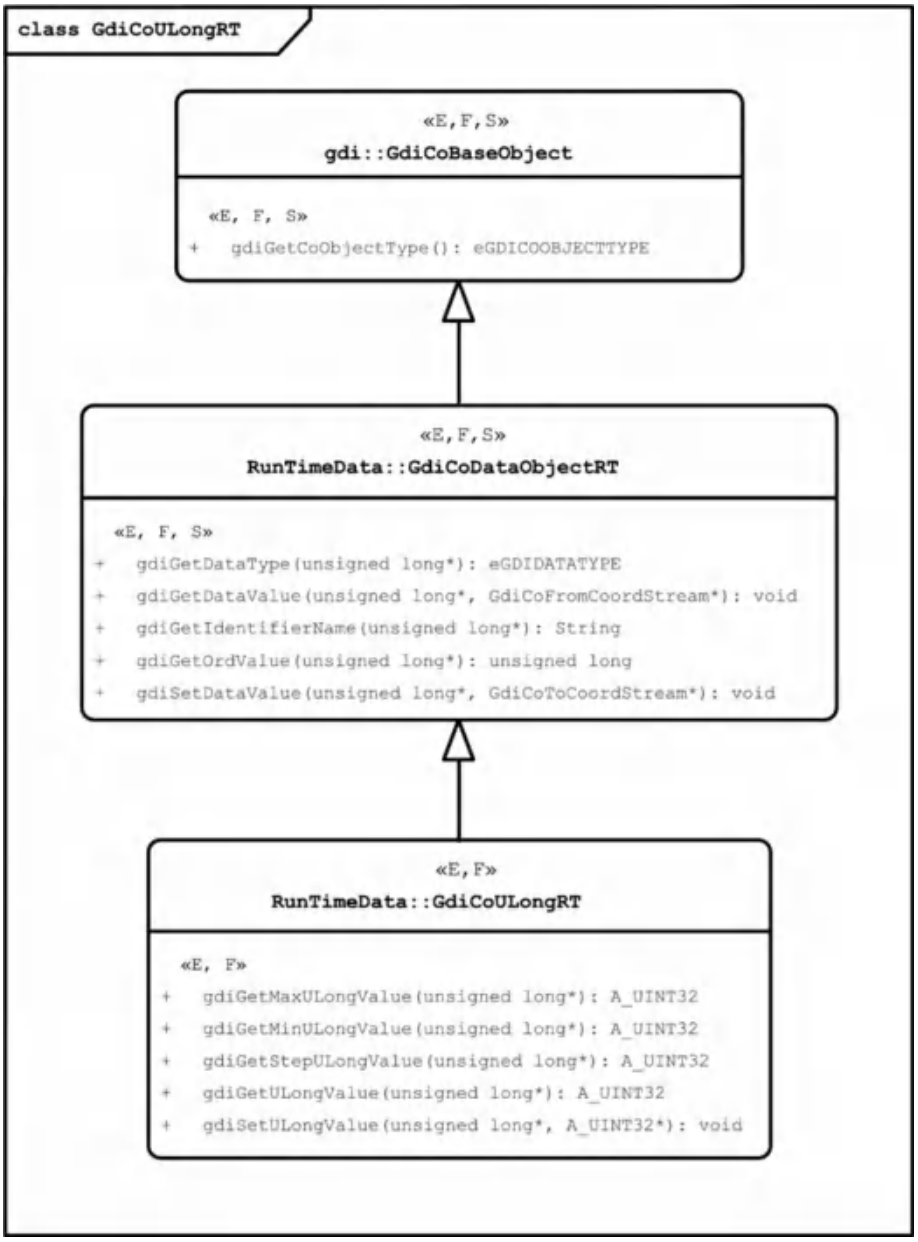
返回对结构体元素的引用。返回基本类型作为参考。描述结构体元素在结构体中的位置的数值用作选择器。可以通过gdiGetType（）检测结构体元素的实际类型（如果未知）。然后，该引用可以用作对检测到的数据类型的引用。

返回值

返回数据的引用。

B.2.14 «E,F» GdiCoULongRT

Thisinterface containsservicesforthe handlingofthe type unsigned Long.



图B.14 GdiCoULongRT的层级图

B.2.14.1 GdiCoULongRT :: «E,F» gdiGetMaxULongValue()

函数调用

```
gdiGetMaxULongValue
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

函数描述

根据数据类型，返回数据的最大值。如果没有明确设置最大值，则取 4294967295。

返回值

根据数据类型，返回数据的最大值

B.2.14.2 GdiCoULongRT :: «E,F» gdiGetMinULongValue()

函数调用

```
gdiGetMinULongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

函数描述

根据数据类型，返回数据的最小值。如果没有明确设置最小值，则取 0。

返回值

根据数据类型，返回数据的最小值

B.2.14.3 GdiCoULongRT :: «E,F» gdiGetStepULongValue()

函数调用

```
gdiGetStepULongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

函数描述

返回类型的步长，步长定义该数据从最小值开始的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据数据类型，返回数据的步长。

B.2.14.4 GdiCoULongRT :: «E,F» gdiGetULongValue()

函数调用

```
gdiGetULongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

函数描述

ReturnsthevalueofanUnsignedLongdatum.Thismightalsobeasub-elementofthetypeCharofa datum(Array, Sequence, Structure,Union).

ReturnValue

Returnsthe Unsigned Longvalue of the (Sub)datum.

B.2.14.5 GdiCoULongRT :: «E,F» gdiSetULongValue()

函数调用

```
gdiSetULongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT32ulValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

函数描述

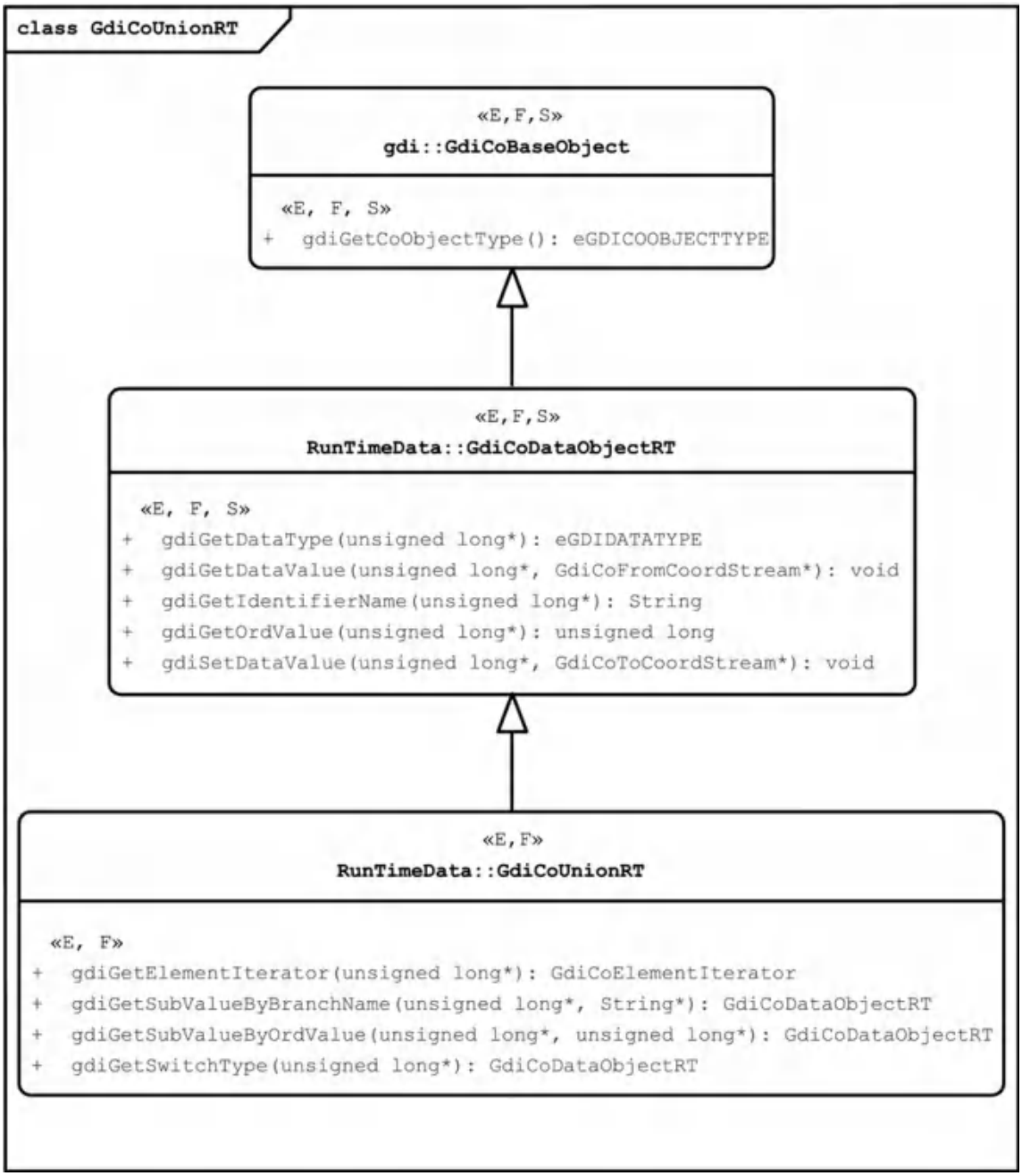
将一个无符号 Long 值复制到协调器的数据区域。该服务还用于用值（数组、序列、结构体、联合体的元素）填充数据的子区域。

返回值

空

B.2.15 «E,F» GdiCoUnionRT

该接口包含用于处理 Union 类型的服务。



图B.15 GdiCoUnionRT的层级图

B.2.15.1 GdiCoUnionRT :: «E,F» gdiGetElementIterator()

函数调用

```
gdiGetElementIterator
        (
        inout                   unsignedlongulAppHnd
```

```
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoElementIterator
```

功能说明

生成元素迭代器，通过该元素迭代器可以检测联合的所有子元素。迭代器设置为联合区域的第一个元素。

返回值

返回对类型迭代器的引用（指针，句柄，类）。

B.2.15.2 GdiCoUnionrt : : [E, F] gdiGetSubvaluebnchname ()

函数调用

```
gdiGetSubValueByBranchName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsBranch
            /*inparameter
            Sectionnameofthereferencedstructureelement.*/
            ):GdiCoDataObjectRT
```

功能说明

返回对Union的overlay元素的引用。返回基本类型作为参考。元素名称用作选择器。可以通过gdiGetType（）检测结构元素的实际类型（如果未知）。然后，该引用可以用作对检测到的数据类型的引用。

如果给出了默认分支的名称，则将返回默认分支的类型。

返回值

返回对基准的引用。

B.2.15.3 GdiCoUnionRT : : [E, F] gdiGetSubValueByValue 值 ()

函数调用

```
gdiGetSubValueByOrdValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   unsignedlongulBranch
            /*inparameter
            Switchvalueofthestructureelement.*/
            ):GdiCoDataObjectRT
```

功能说明

返回对 Union 的 overlay 元素的引用。返回基本类型作为参考。开关值用作选择器（分支）。可以通过gdiGetType（）检测结构元素的实际类型（如果未知）。然后，该引用可以用作对检测到的数据类型的引用。

如果没有分支与设置值匹配，则使用默认分支（如果存在）。

返回值

返回对基准的引用。

B.2.15.4 GdiCoUnionrt : : [E, F] gdiGetSwitchtype ()

函数调用

```
gdiGetSwitchType
            (
```

```
inout                    unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):GdiCoDataObjectRT
```

功能说明

生成对 RefGdiCoBaseType 的引用。开关值的实际类型由服务 gdiGetType()相对于返回的引用进行检测。

可以在返回的子对象上设置分支选择器。如果没有分支与设置值匹配，则使用默认分支（如果存在）。

返回值

返回 Switch 值的类型。

B.2.16 «E,F» GdiCoUnlimitedSequenceRT

该接口包含用于处理无限序列类型的服务。

图B.16 GdiCoUnlimitedSequenceRT的层次结构图

B.2.16.1 GdiCoUnlimitedSequenceRT :: «E,F» gdiGetActualSize()

函数调用

```
gdiGetActualSize
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

功能说明

返回 Sequence 数组中有效数据元素的数量。从索引 0 开始，数据连续位于数组中。

返回值

返回可用序列数组元素的数量。

B.2.16.2 GdiCoUnlimitedSequenceRT :: «E,F» gdiGetSubValueObject()

函数调用

```
gdiGetSubValueObject
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  unsignedlongulIndex
          /*inparameter
          IndexValueofthedataelement*/
          ):GdiCoDataObjectRT
```

功能说明

返回对序列或数组中数据元素的引用。此后，可以通过 gdiGetType（）检测此元素的类型（如果未知），并且可以根据该类型访问数组元素的数据区域。

返回值

返回对该序列的子值对象的引用。

B.2.16.3 GdiCoUnlimitedSequenceRT :: «E,F» gdiSetActualSize()

函数调用

```
gdiSetActualSize
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT32ulActualSize
          /*inparameter
          NewnumberofvalliddataofaSequence*/
          ):void
```

功能说明

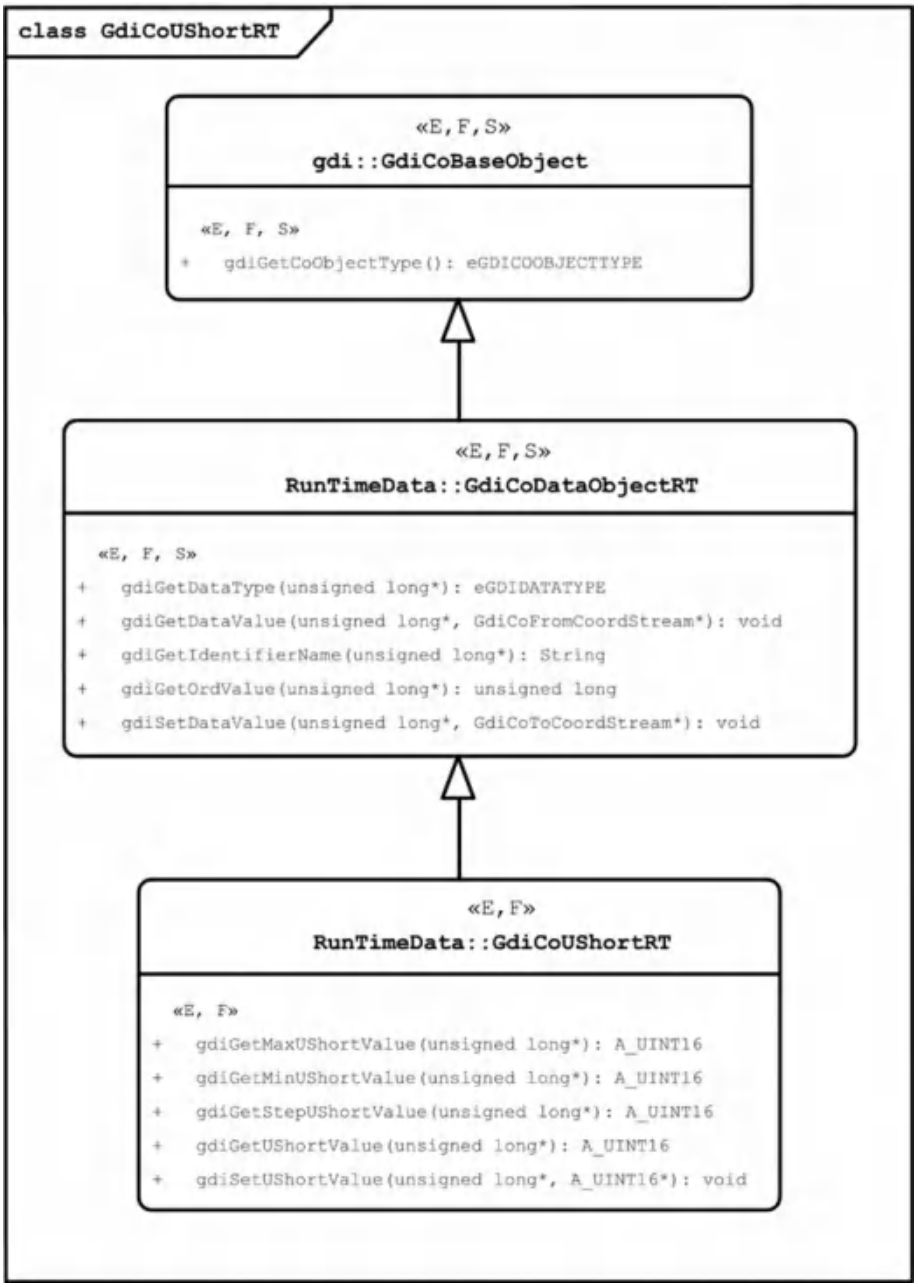
设置序列数组中有效数据元素的数量，从索引 0 开始，数据连续位于数组中。如果增加数据数量，则未定义添加数据的内容。

返回值

无

B.2.17 «E,F» GdiCoUShortRT

此接口包含用于处理 unsigned Short 类型的服务。



图B.17　GdiCoUShortRT的层次结构图

B.2.17.1 GdiCoUShortRT ::«E, F»gdiGetMaxUShortValue（）

函数调用

```
gdiGetMaxUShortValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_UINT16
```

功能说明

根据此类型返回基准的最大值。如果尚未显式设置最大值，则其值为65535。

返回值

根据此类型返回基准的最大值。

B.2.17.2 GdicoushortRT ： ： [E, F] gdiGetMinushortvalue （）

函数调用

```
gdiGetMinUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT16
```

功能说明

根据此类型返回基准的最小值。如果未明确设置最小值，则取值为 0。

返回值

根据此类型返回基准的最小值。

B.2.17.3 GdiCoushortRT : : [E, F] gdiGetStepUShortvalue ()

函数调用

```
gdiGetStepUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT16
```

功能说明

返回类型的步长,该步长定义从最小值开始的基准数据的有效元素。如果未明确定义最小值,则取值为 1。

返回值

根据此类型返回基准的步长。

B.2.17.4 GdiCoushortRT : : [E, F] gdiGetushortvalue ()

函数调用

```
gdiGetUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT16
```

功能说明

返回无符号短基准的值。这也可能是缺少数据（数组，序列，结构，联合）的无符号类型的子元素。

返回值

返回（Sub）数据的无符号 Short 值。

B.2.17.5 GdiCoushortRT : : [E, F] gdiSetUshortvalue ()

函数调用

```
gdiSetUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT16unValue
          /*inparameter
          referencetothevaluetobeset.*/
          ):void
```

功能说明

将无符号短值复制到协调器的数据区域。该服务还用于用值（数组，序列，结构，联合的元素）填充基准的子区域。

返回值

无

附 录 C
（规范性）
编程参考指南—完全访问接口

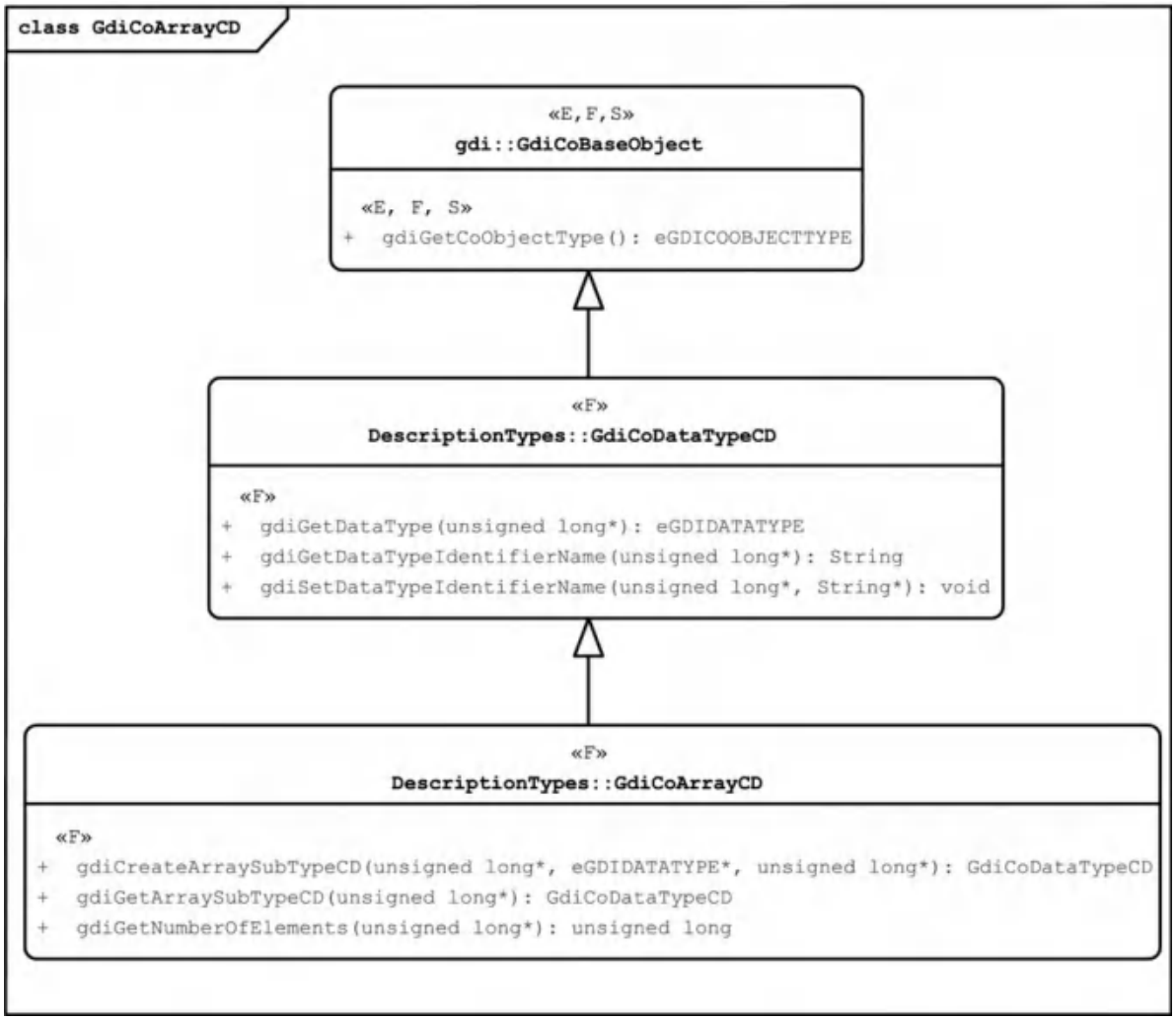
C.1 一般信息

具有《F》标记的完全访问接口的单个接口（类及其方法和属性），不属于智能访问接口或扩展访问接口。因此，所有单个接口都标记有《F》。附录 A 中描述的智能访问接口和附录 B 中描述的扩展访问接口的所有单个接口应包含在完全访问接口中，本附件中不再赘述。下面的详细描述由图 C.1 至 C.29 完成，以概述类及其方法

C.2 完全访问接口的详细描述

C.2.1《 F》 GdiCoArrayCD

该接口包含用于处理数组类型描述的服务。该对象引用是有效的，直到创建或引用下一个子描述为止无



图C.1 GdiCoArrayCD的层次结构图

C.2.1.1 GdiCoArrayCD :: «F» gdiCreateArraySubTypeCD()

函数调用

```
gdiCreateArraySubTypeCD (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
```

```
            inout                   eGDIDATATYPEeNewType
            /*inparameter
            GDItypeoftheArrayelements
            ThevaluerangeincludesbothASAMDatatypeenumerationandGDIType enumeration.*/
            inout                   unsignedlongulNumberOfElements
            /*SizeoftheArrayinelements*/
            ):GdiCoDataTypeCD
```

功能说明

创建数组的子类型描述。

描述数组元素的类型和数量。元素类型可能很复杂。通过返回值可以更详细地指定元素类型。如果不允许该元素的基本类型，则将传递一个 NIL 引用（例如，如果 Array 本身是 Union 的一部分，则 Array 的元素类型不能为 Sequence）。

返回值

如果成功，则返回对 GdiCoDataTypeCD 的引用。否则为 NIL 参考。

C.2.1.2 GdiCoArrayCD :: «F» gdiGetArraySubTypeCD()

函数调用

```
GDigetarraysublicd(
gdiGetArraySubTypeCD (
            inout                   unsignedlongulAppHnd
            /*                      inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoDataTypeCD
```

功能说明

返回对子数据类型的描述的引用。

返回值

返回对子类型的描述的引用（如果存在），否则返回 NIL 引用。

C.2.1.3 GdiCoArrayCD :: «F» gdiGetNumberOfElements()

函数调用

```
gdiGetNumberOfElements
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):unsignedlong
```

功能说明

它用于检测数组元素的数量。

返回值

返回 Array 的元素数。

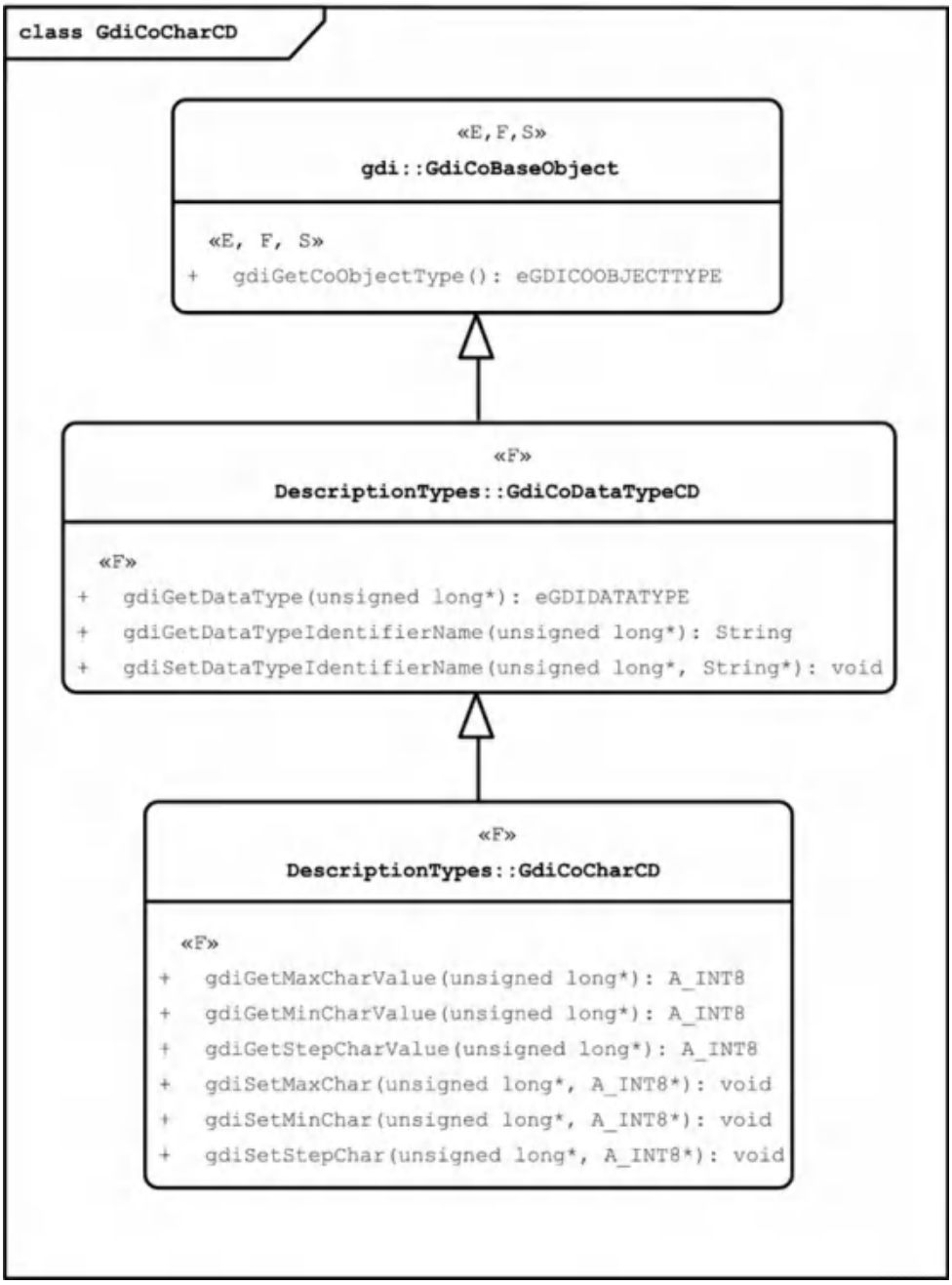
C.2.2《 F》 GdiCoBooleanCD

包含有关限制定义的布尔类型的服务。

图C.2 GdiCoBooleanCD的层次结构图

C.2.3《 F》 GdiCoCharCD

包含与用于限制定义的 Char 类型有关的服务。



图C.3 GdiCoCharCD的层次结构图

C.2.3.1 GdiCoCharCD :: «F» gdiGetMaxCharValue()

函数调用

```
gdiGetMaxCharValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT8
```

功能说明

根据此类型返回基准的最大值。如果未明确设置最大值，则取值为127。

返回值

根据此类型返回基准的最大值。

C.2.3.2 GdiCoCharCD :: «F» gdiGetMinCharValue()

函数调用

```
gdiGetMinCharValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT8
```

功能说明

根据此类型返回基准的最小值。如果未明确设置最小值，则其值为-128。

返回值

根据此类型返回基准的最小值。

C.2.3.3 GdiCoCharCD :: «F» gdiGetStepCharValue()

函数调用

```
gdiGetStepCharValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT8
```

功能说明

返回类型的步长,该步长定义从最小值开始的基准数据的有效元素。如果未明确定义最小值,则取值为1。

返回值

根据此类型返回基准的步长。

C.2.3.4 GdiCoCharCD :: «F» gdiSetMaxChar()

函数调用

```
gdiSetMaxChar
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_INT8cLimitValue
          /*inparameter
          newmaximumvalue(<128)*/
          ):void
```

功能说明

根据此类型定义基准的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.3.5 GdiCoCharCD :: «F» gdiSetMinChar()

函数调用

```
gdiSetMinChar
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
```

```
inout                    A_INT8cLimitValue
/*newminimumvalue(>-129)*/
):void
```

功能说明

根据该类型定义基准的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.3.6 GdiCoCharCD :: «F» gdiSetStepChar()

函数调用

```
gdiSetStepChar
          (
inout                    unsignedlongulAppHnd
/*inparameter
Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                    A_INT8cLimitValue
/*inparameter
newstepwidth(0<LimitValue<127)*/
):void
```

功能说明

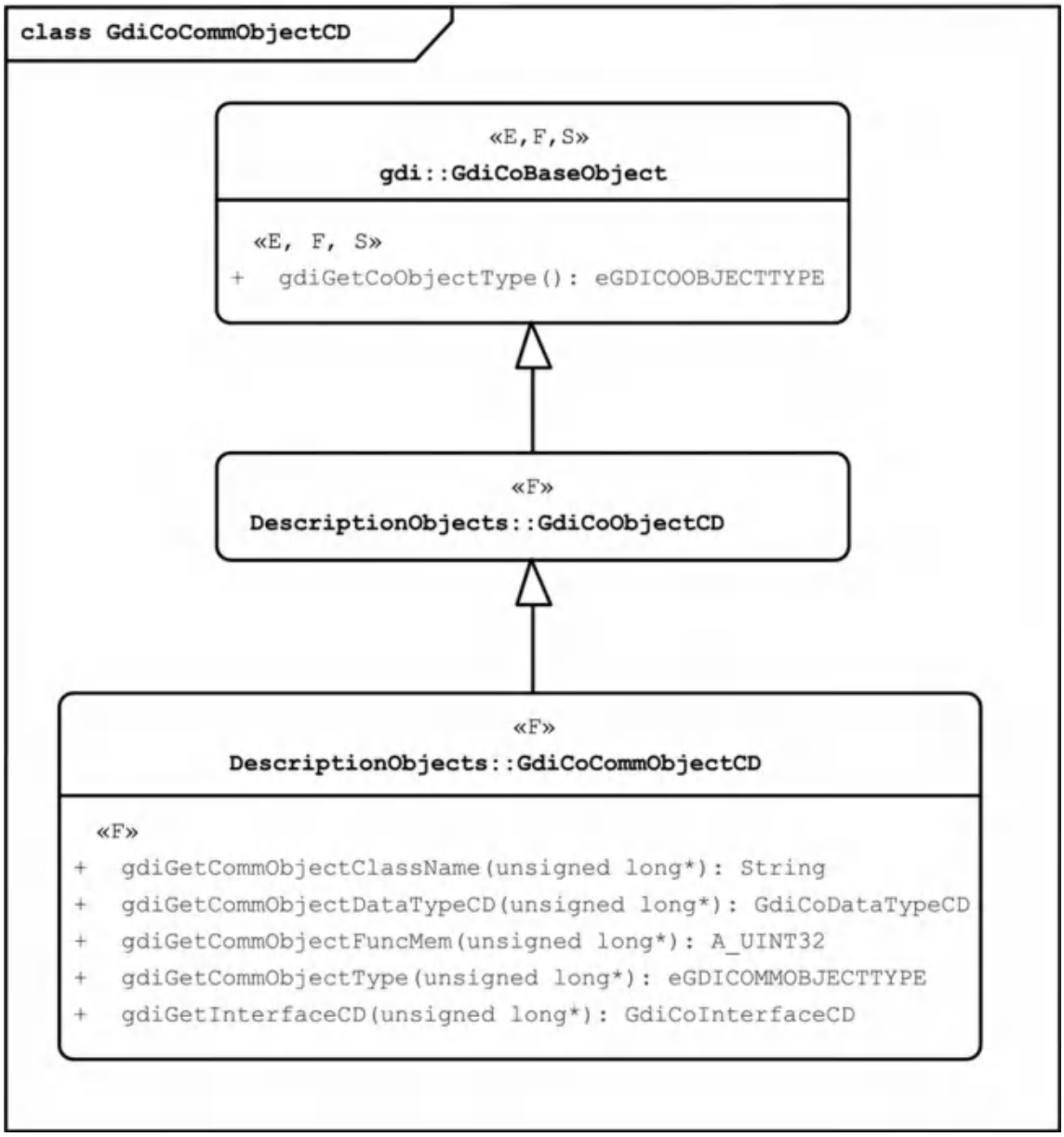
根据此类型在最小和最大之间定义有效数据的步长。最小值是起始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.4《 F》 GdiCoCommObjectCD

该接口描述了通信对象。该对象引用是有效的，直到创建或引用下一个子描述为止。



图C.4 GdiCoCommObjectCD的层次结构图

C.2.4.1 GdicocommObjectcd : [F] gdiGetCommObjectClassName ()

函数调用

```
gdiGetCommObjectClassName
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):String
```

功能说明

返回此 CO 描述中描述的通信对象的类名称。

如果因为协调器处理 PID 文件而已经设置了 CO 名称，则可以通过调用此方法来获取 CO 名称。

返回值

返回通信对象的类名，如模块说明中所述。

C.2.4.2 GdicocommObjectcd : [F] gdiGetCommObjectDataTypeCD ()

函数调用

```
gdiGetCommObjectDataTypeCD (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoDataTypeCD
```

功能说明

返回通信对象的类型描述。

如果由于协调器处理 PID 文件而已经设置了类型描述，则可以通过调用此方法来获取类型描述。

返回值

返回通信对象的类型描述。

C.2.4.3 GdicocommObjectcd : [F] gdiGetCommObjectFuncMem ()

函数调用

```
gdiGetCommObjectFuncMem
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
         ):A_UINT32
```

功能说明

返回通信对象描述中描述的通信对象的绝对索引。

如果由于协调器处理 PID 文件而已经设置了绝对索引，则可以通过调用此方法来获取绝对索引。

返回值

返回此通信对象描述中描述的绝对索引（FuncMem）。

C.2.4.4 GdiCoCommObjectCD :: «F» gdiGetCommObjectType()

函数调用

```
gdiGetCommObjectType
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):eGDICOMMOBJECTTYPE
```

功能说明

返回此描述对象的通信对象的类型。

返回值

返回通信对象的类型。

C.2.4.5 GdicocommObjectcd : [F] gdiGetInterfacecd ()

函数调用

```
gdiGetInterfaceCD (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoInterfaceCD
```

功能说明

创建对上方 GdiCoFuncObjectCD 的引用。

返回值

返回 GdiCoCommObjectCD。

C.2.5《 F》 GdiCoDataTypeCD

此接口是所有数据类型描述的基本类型。

图C.5 GdiCoDataTypeCD的层次结构图

C.2.5.1 GdiCoDataTypeCD :: [F] gdiGetDataType ()

函数调用

```
gdiGetDataType
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):eGDIDATATYPE
```

功能说明

以数字值（枚举）的形式返回类型。

返回值

将类型返回为数值（在语义中描述）。

C.2.5.2 GdiCoDataTypeCD :: «F» gdiGetDataTypeIdentifierName()

函数调用

```
gdiGetDataTypeIdentifierName
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
```

```
            gdiGetWorkspaceByName()*/
            ):String
```

功能说明

返回此数据类型描述中描述的数据类型的名称。

如果由于协调器处理 PID 文件而已经设置了名称，则可以通过调用此方法来获取数据类型名称。

返回值

返回数据类型名称，如数据类型描述中所述。

如果未设置名称，将返回一个空字符串。

C.2.5.3 GdiCoDataTypeCD :: «F» gdiSetDataTypeIdentifierName()

函数调用

```
gdiSetDataTypeIdentifierName
            (
            inout                   unsignedlongulAppHnd
            /*                      inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsName
            /*                      containsthenameofthedescribeddatatype.*/
            ):void
```

功能说明

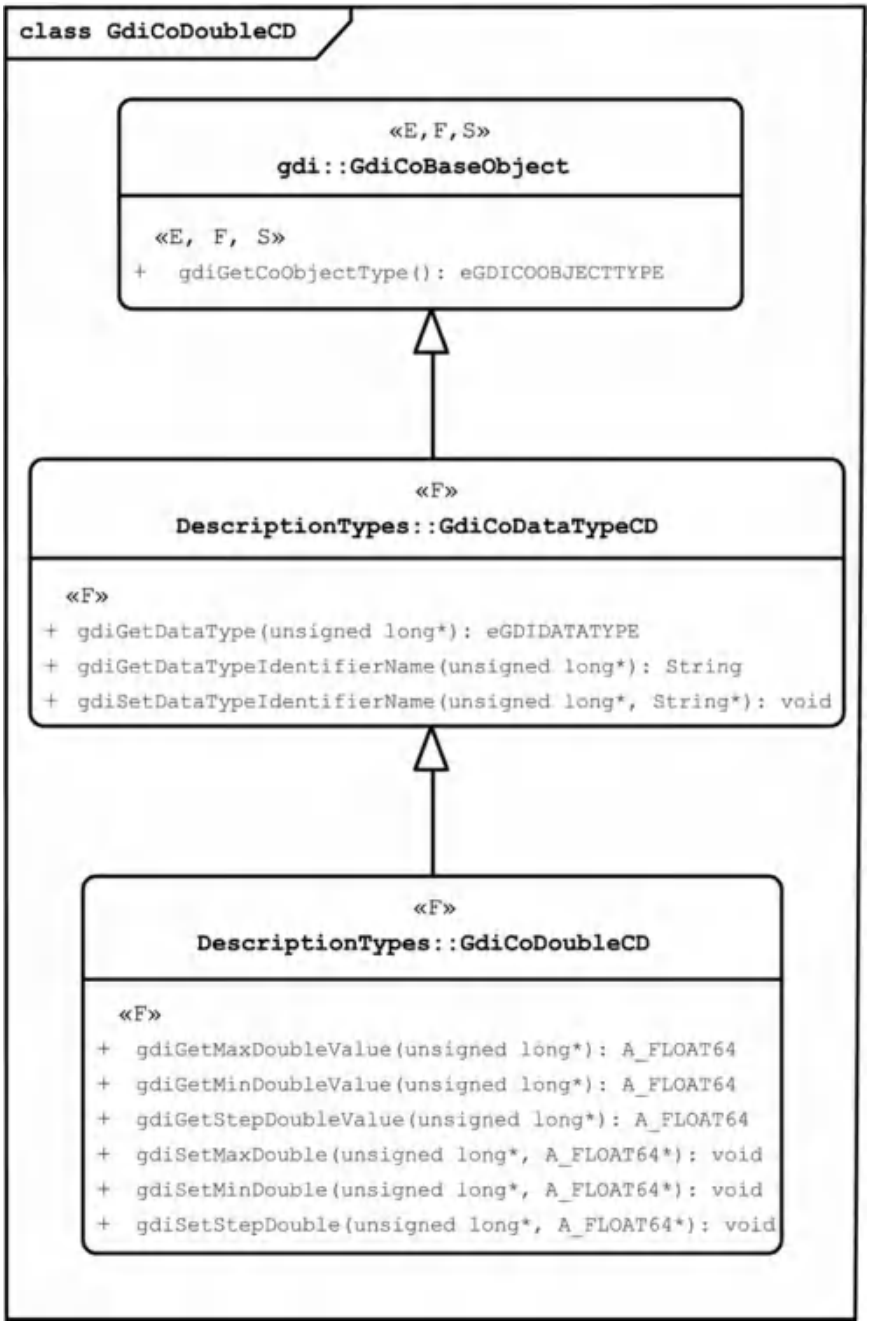
设置此描述对象的数据类型名称。

返回值

无

C.2.6《 F》 GdiCoDoubleCD

包含与用于限制定义的 Double 类型有关的服务。



图C.6 GdiCoDoubleCD的层次结构图

C.2.6.1 GdiCoDoubleCD :: «F» gdiGetMaxDoubleValue()

函数调用

```
gdiGetMaxDoubleValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_FLOAT64
```

功能说明

根据此类型返回基准的最大值。如果未明确设置最大值，则其值为 1,7976931348623158 E + 308。

返回值

根据此类型返回基准的最大值。

C.2.6.2 GdiCoDoubleCD :: «F» gdiGetMinDoubleValue()

函数调用

```
gdiGetMinDoubleValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_FLOAT64
```

功能说明

根据此类型返回基准的最小值。如果未明确设置最小值，则它将取值为 2,2250738585072014 E-308。

返回值

根据此类型返回基准的最小值。

C.2.6.3 GdiCoDoubleCD :: «F» gdiGetStepDoubleValue()

函数调用

```
gdiGetStepDoubleValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_FLOAT64
```

功能说明

返回类型的步长,该步长定义从最小值开始的基准数据的有效元素。如果未明确定义最小值,则取值为 1。

返回值

根据此类型返回基准的步长。

C.2.6.4 GdiCoDoubleCD :: «F» gdiSetMaxDouble()

函数调用

```
gdiSetMaxDouble
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_FLOAT64dLimitValue
          /*inparameter
          newmaximumvalue(<1,7976931348623158E+309)*/
          ):void
```

功能说明

根据此类型定义基准的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.6.5 GDICodubleCD:: [F] gdiSetminDouble ()

函数调用

```
gdiSetMinDouble
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
```

```
            inout                   A_FLOAT64dLimitValue
            /*inparameter
            newminimumvalue(>2,2250738585072014E-309)*/
            ):void
```

功能说明

根据该类型定义基准的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.6.6 GdiCoDoubleCD :: «F» gdiSetStepDouble()

函数调用

```
gdiSetStepDouble
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_FLOAT64dLimitValue
            /*inparameter
            newstepwidth(0<LimitValue<1,7976931348623158E+308)*/
            ):void
```

功能说明

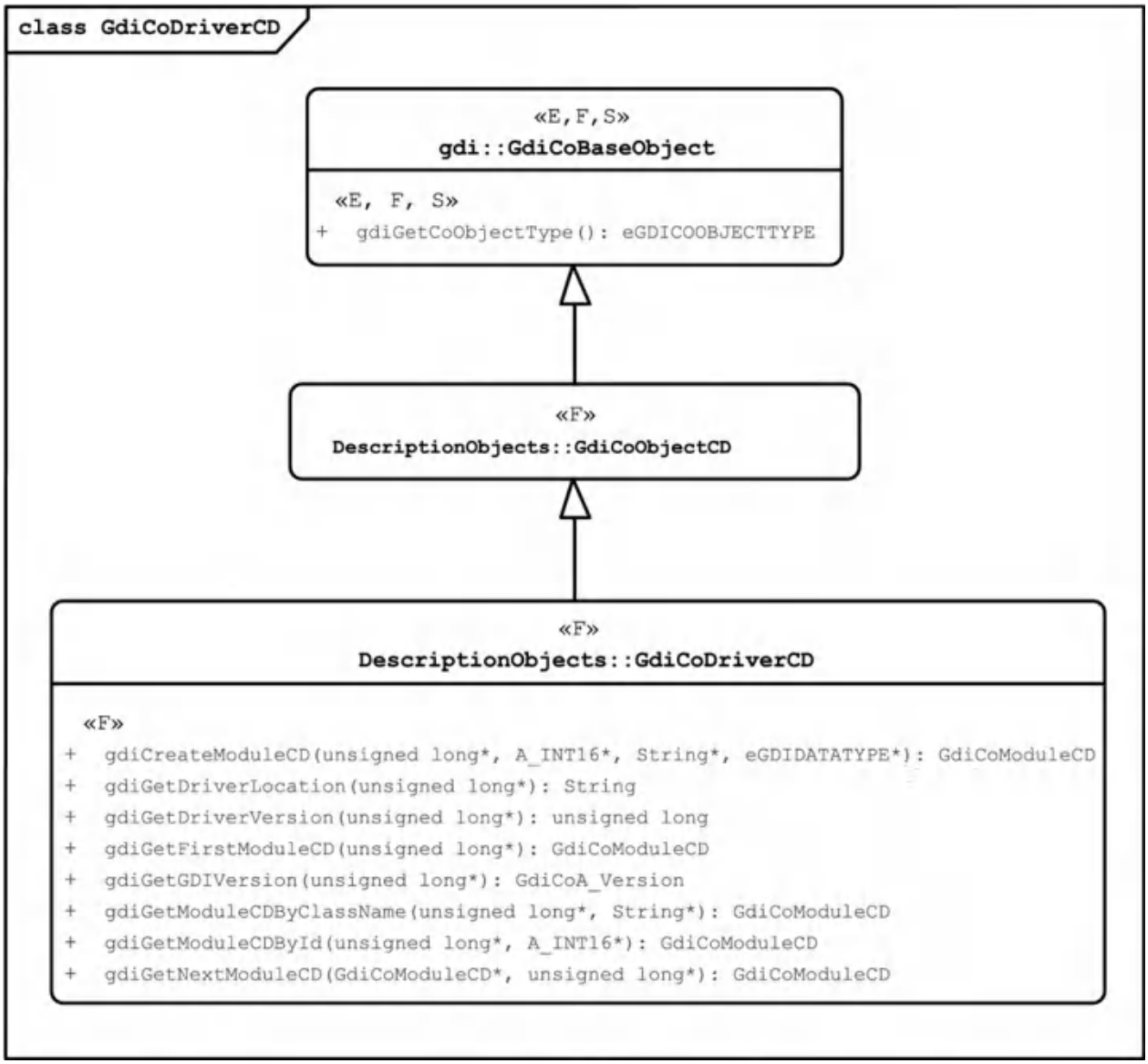
根据此类型在最小和最大之间定义有效数据的步长。最小值是起始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.7 [F] GdiCoDriverCD

描述 GDI 驱动程序对象。该对象引用是有效的，直到创建或引用下一个子描述为止。



图C.7 GdiCoDriverCD的层次结构图

C.2.7.1 GdiCoDriverCD :: [F] gdicreateModuleCD ()

函数调用

```
gdiCreateModuleCD (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   A_INT16nModuleId
        /*inparameter
        containsthemoduleID*/
        inout                   StringsModuleClassName
        /*inparameter
        containstheclassnameofthemodule.*/
        inout                   eGDIDATATYPEeCreateParamType
        /*inparmeter
        containsthetypeofthecreateparmeter.
        eDT_NO_DEFINITIONifnotnecessary.*/
        ):GdiCoModuleCD
```

功能说明

返回一个新的 GdiCoModuleCD 对象。

返回值

返回一个新的 GdiCoModuleCD 对象。

C.2.7.2 GdiCoDriverCD :: [F] gdiGetDriverLocation ()

函数调用

```
gdiGetDriverLocation
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

功能说明

返回驱动程序描述中描述的驱动程序的位置规范。

如果由于协调器处理 PID 文件而已经设置了驱动程序的位置说明，则可以通过调用此方法来获取位置说明。

返回值

返回驱动程序描述中描述的驱动程序的位置规范。

C.2.7.3 GdiCoDriverCD :: [F] gdiGetDriverversion ()

函数调用

```
gdiGetDriverVersion
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):unsignedlong
```

功能说明

返回驱动程序说明中描述的驱动程序版本。

如果由于协调程序处理 PID 文件而已经设置了驱动程序版本，则可以通过调用此方法来获取驱动程序版本。

返回值

返回驱动程序说明中描述的驱动程序版本。

C.2.7.4 GdiCoDriverCD :: [F] gdiGetFirstModuleCD ()

函数调用

```
gdiGetFirstModuleCD (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoModuleCD
```

功能说明

如果存在，则在驱动程序描述中创建对第一个模块描述的引用。

返回值

如果存在，则返回 GdiCoModuleCD。

C.2.7.5 GdiCoDriverCD :: [F] gdiGetGdiversion ()

函数调用

```
gdiGetGDIVersion
```

```
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):GdiCoA_Version
```

功能说明

返回驱动程序说明中描述的 GDI 版本。

如果由于协调器处理 PID 文件而已经设置了 GDI 版本，则可以通过调用此方法来获取 GDI 版本。

返回值

返回驱动程序说明中描述的 GDI 版本。

C.2.7.6 GdiCoDriverCD :: [F] gdiGetmoduleCDByClassname ()

函数调用

```
gdiGetModuleCDByClassName
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                   StringsModuleName
/*inparameter
containsthemodulename.*/
):GdiCoModuleCD
```

功能说明

返回与给定型号名称相对应的模块描述。

返回值

返回与给定模块名称相对应的模块描述。

C.2.7.7 GdiCoDriverCD :: [F] gdigetmoduleCDBYID ()

函数调用

```
gdiGetModuleCDById
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                   A_INT16nModuleId
/*inparameter
containsthemoduleID.*/
):GdiCoModuleCD
```

功能说明

返回与给定模块 ID 对应的模块描述。

返回值

返回与给定型号 ID 对应的模块描述。

C.2.7.8 GdiCoDriverCD :: [D, F] gdiGetNextmodulecd ()

函数调用

```
gdiGetNextModuleCD (
inout                   GdiCoModuleCDrefLastElement
/*inparameter
referencetothelastModuledescription.*/
inout                   unsignedlongulAppHnd
```

```
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):GdiCoModuleCD
```

功能说明

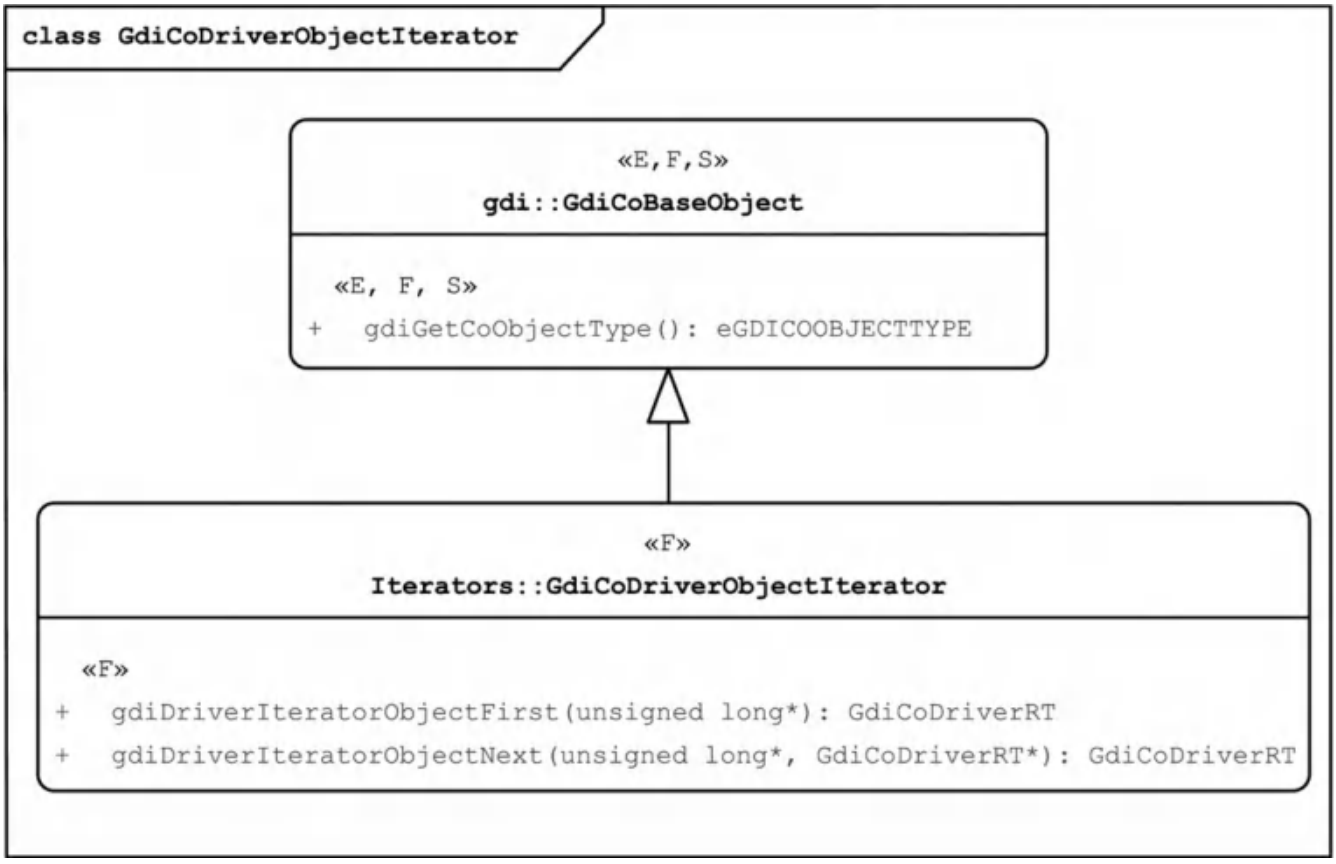
返回对驱动程序描述中下一个模块描述的引用。

返回值

如果存在，则返回 GdiCoModuleCD。

C.2.8《 F》 GdiCoDriverObjectIterator

此接口包含用于驱动程序迭代器应用程序的服务，该服务是在工作区中设置的所有驱动程序对象的顺序输出。



图C.8 GdiCoDriverObjectIterator的层次结构图

C.2.8.1 GdiCoDriverObjectIterator :: «F» gdiDriverIteratorObjectFirst()

函数调用

```
gdiDriverIteratorObjectFirst
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):GdiCoDriverRT
```

功能说明

将内部指针设置为工作区中的第一个驱动程序对象。

返回值

返回对协调程序驱动程序对象（协调程序内的管理单元）的引用（如果存在），否则返回 NIL。

C.2.8.2 GdiCoDriverObjectIterator :: «F» gdiDriverIteratorObjectNext()

函数调用

```
gdiDriverIteratorObjectNext
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoDriverRTrefLastElement
          /*inparameter
          A reference to a element returned by gdiDriverObjectFirst or
          gdiDriverObjectNext.*/
          ):GdiCoDriverRT
```

功能说明

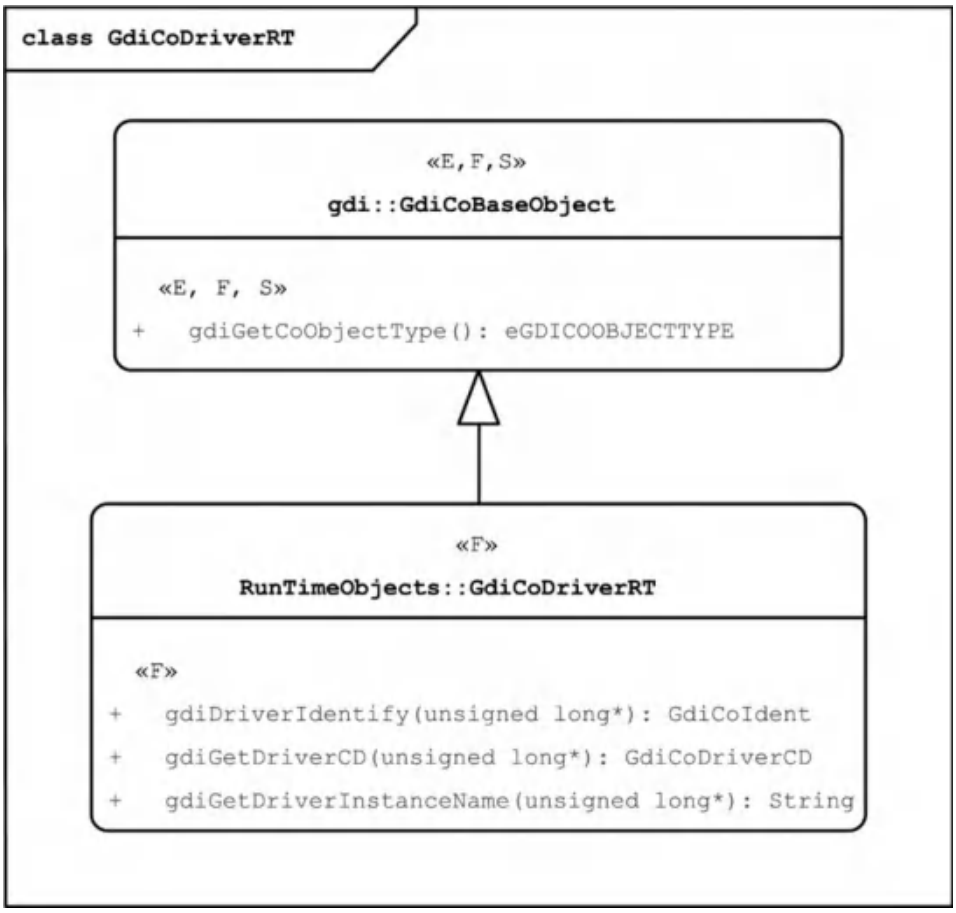
检测对工作空间中当前 DriverRT Object（参数）的引用，将内部指针设置为下一个 DriverRT Object（如果存在），然后返回引用。

返回值

如果存在，则返回对协调程序驱动程序对象（协调程序内的管理单元）的引用，否则返回 NIL。

C.2.9《 F》 GdiCoDriverRT

该界面包含用于设置和删除 Control VD 的服务。



图C.9 GdiCoDriverRT的层次结构图

C.2.9.1 GdiCoDriverRT :: «F» gdiDriverIdentify()

函数调用

```
gdiDriverIdentify
```

```
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoIdent
```

功能说明

影响对设备驱动程序的“识别”调用的执行。检测到的数据与设备驱动程序的标识数据相对应。

返回值

根据 GDI 规范返回 GDIIDENT 数据。

C.2.9.2 GdiCoDriverRT :: [F] gdiGetDrivercd ()

函数调用

```
gdiGetDriverCD (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoDriverCD
```

功能说明

生成对驱动程序描述的引用作为 GdiCoDriverCD 对象。

返回值

返回对 GdiCoDriverCD 的引用。

C.2.9.3 GdiCoDriverRT :: [F] gdiGetDriverInstanceName ()

函数调用

```
gdiGetDriverInstanceName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):String
```

功能说明

返回参数化过程中定义的 Driver 对象的实例名称。

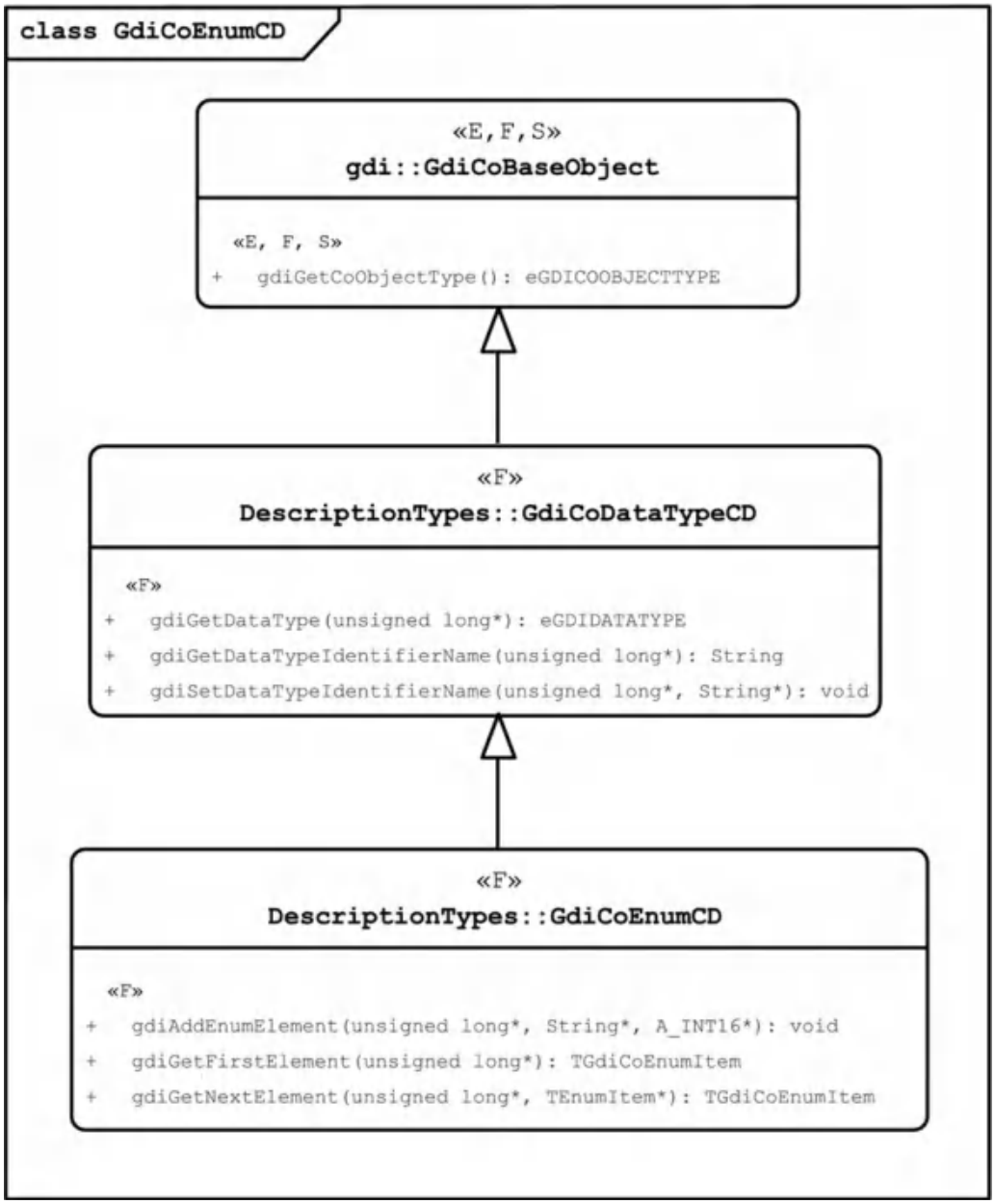
在工作空间内应是唯一的。

返回值

以字符串形式返回驱动程序实例名称。

C.2.10 «F» GdiCoEnumCD

包含用于枚举元素定义的服务。



图C.10 GdiCoEnumCD的层次结构图

C.2.10.1 GdiCoenumCD : : [F] gdiAdenumElement ()

函数调用

```
gdiAddEnumElement
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   StringsName
        /*inparameter
        Identifieroftheenumerationelement.*/
        inout                   A_INT16nValue
        /*inparameter
        numericvalueoftheenumerationelement*/
        ):void
```

功能说明

向枚举添加新元素。允许覆盖。关键值是名称。

返回值

无

C.2.10.2 GdiCoEnumCD :: «F» gdiGetFirstElement()

函数调用

```
gdiGetFirstElement
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):TGdiCoEnumItem
```

功能说明

返回单个元素枚举的第一个元素。通常这是具有最低值的元素。 return 元素是一个由子元素 String 和 short 组成的结构。

返回值

返回由子元素 String 和 short 组成的结构。如果没有可用的元素，则返回值“ noGDIElement”作为名称，并返回数字值 32768。

C.2.10.3 GdiCoenumCD : : [F] gdiGetNextlement ()

函数调用

```
gdiGetNextElement
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                    TEnumItemrefLastElement
          /*inparameter
          containsthereferencetothelastEnumerationDescription.*/
          ):TGdiCoEnumItem
```

功能说明

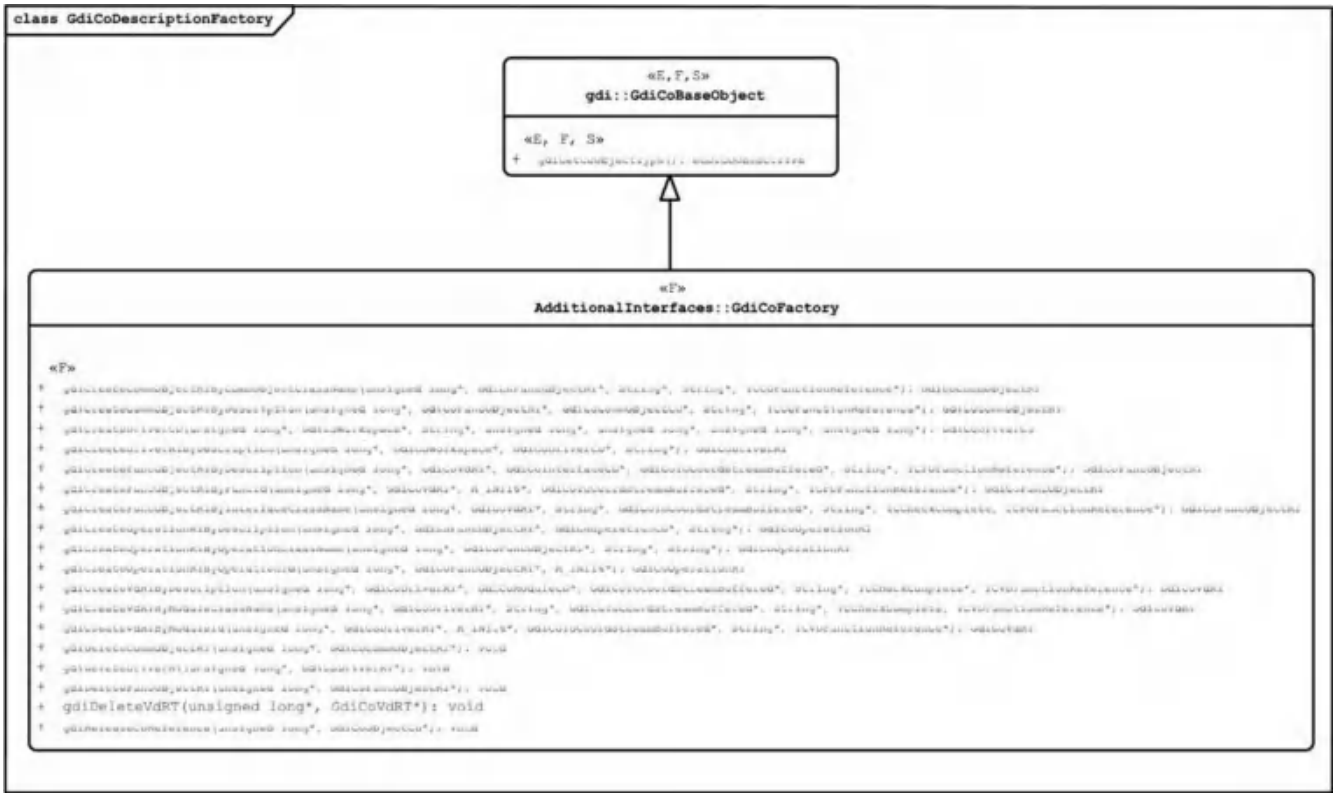
返回单个元素枚举的下一个元素。 return 元素是一个由子元素 String 和 short 组成的结构。

返回值

如果没有可用的元素，则返回值“ noGDIElement”作为名称，并返回数字值 32768。

C.2.11 [F] GdiCoFactory

该接口是用于创建和删除 GDI 对象的静态类。



图C.11　GdiCoFactory的层次结构图

C.2.11.1 GdiCoFactory ： ： [F] gdicreateCommobjectrtByCommobectClassname （）

函数调用

```
gdiCreateCommObjectRTByCommObjectClassName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoFuncObjectRTrefFuncObjectRT
          /*inparameter
          Containsthereferencetothefunctionobject.TheCommunicationObject
          willbeconnectedtothisFO.*/
          inout                  StringsCommObjectClassName
          /*inparameter
          CommunicationObjectClassname(DCD)oftheCommunicationObject.*/
          inout                  StringsCommObjectInstanceName
          /*inparameter
          instancenameofthenewcreatedCommunicationObject.*/
          inout                  TCCOFunctionReference
                                    refCBCompleteCreateCommObject
          /*inparameter
          Reference  to  a  call  back  method  which  handles  the  asynchronous
          initializationmechanism.ANILreferencesignalsasynchronouscall.
          */
          ):GdiCoCommObjectRT
```

功能描述

根据给定类（DCD）名称定义的通信对象描述，在协调器内创建通讯对象并作为设备驱动程序中的实例。在异步操作的情况下，将通过完整的回调传递对通信对象的引用。如果没有给出回调参考，则该操作将同步执行。

返回值

以 GdiCoAttributeRT 的形式返回对新创建的 GdiCoCommObjectRT 实例的引用。

如果是异步操作，则为 NIL。

C.2.11.2 GdiCoFactory :: «F» gdiCreateCommObjectRTByDescription()

函数调用

```
gdiCreateCommObjectRTByDescription
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoFuncObjectRTrefFuncObjectRT
          /*inparameter
          Containsthereferencetothefunctionobject.TheCommunicationObject
          willbeconnectedtothisFO.*/
          inout                  GdiCoCommObjectCDrefCommObjectCD
          /*inparameter
          referencetotheGdiCoCommObjectCDwhichdescribestheCommunication
          Object.*/
          inout                  StringsCommObjectInstanceName
          /*inparameter
          InstanceNameofthecommunicationobject.*/
          inout                  TCCOFunctionReference
                                    refCBCompleteCreateCommObject
          /*inparameter
          Reference  to  a  call  back  method  which  handles  the  asynchronous
          initializationmechanism.ANILreferencesignalsasynchronouscall.
          */
          ):GdiCoCommObjectRT
```

功能描述

根据给定的通信对象描述，在协调器中创建通信对象并在设备驱动程序中作为实例。

在异步操作的情况下，将通过完整的回调传递对通信对象的引用。如果没有给出回调参考，则该操作将同步执行。

返回值

以 GdiCoAttributeRT 的形式返回对新创建的 GdiCoCommObjectRT 实例的引用。

如果是异步操作，则为 NIL。

C.2.11.3 GdiCoFactory :: «F» gdiCreateDriverCD()

函数调用

```
gdiCreateDriverCD (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoWorkspacerefWorkspace
          /*inparameter
          containsthereferencetotheworkspace.Thedriverwillbeconnected tothisworkspace.*/
          inout                  StringsLocation
          /*inparmater
          containsthelocationspecificationofthedriver.*/
          inout                  unsignedlongulDriverVersion
          /*inparameter
```

```
            Containsthemanufacturerdriverversion.*/
            inout                   unsignedlongulGdiVersionMajor
            /*inparameter
            ContainstheGDImajorversion.*/
            inout                   unsignedlongulGdiVersionMinor
            /*inparameter
            ContainstheGDIminorversion.*/
            inout                   unsignedlongulGdiVersionRevision
            /*inparameter
            ContainstheGDIrevisionnumber.*/
            ):GdiCoDriverCD
```

功能描述

返回一个新的 GdiCoDriverCD 对象

返回值

返回一个新的 GdiDriverCD 对象

C.2.11.4 GdiCoFactory :: «F» gdiCreateDriverRTByDescription()

函数调用

```
gdiCreateDriverRTByDescription
            (
            inout                   unsigned long ulAppHnd
            /* in parameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)
            or by gdiGetWorkspaceByName() */
            inout                   GdiCoWorkspace refWorkspace
            /* in parameter
            contains    the    reference    to    the    workspace.    The    driver    will    be
            connected to this workspace. */
            inout                   GdiCoDriverCD refDriverCD
            /* in parameter
            Reference to the Description of the driver. */
            inout                   String sDriverInstanceName
            /* in parameter
            contains the Driver instance name. */
            ) : GdiCoDriverRT
```

功能描述

如果尚未加载到协调器，加载 GDI 驱动程序，并将驱动程序与给定的工作区连接。否则，驱动程序的管理计数器将递增。

返回值

如果成功，则返回对 GdiCoDriverRT 的引用。

C.2.11.5 GdiCoFactory :: «F» gdiCreateFuncObjectRTByDescription()

函数调用

```
gdiCreateFuncObjectRTByDescription
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoVdRTrefVdRT
            /*inparameter
            ContainsthereferencetotheapplicationVD.Thefunctionobjectwill
            beconnectedtothisVD.*/
```

```
inout                    GdiCoInterfaceCDrefInterfaceCD
/*inparameter
referencetotheGdiCoInterfaceCDwhichdescribesthefunction object.
*/
inout                    GdiCoToCoordStreamBufferedrefCreateParameter
/*inparameter
thegivenStreamobjectcontainsthecreateparameteroftheVD.
IfnocreateParameterexists,NILishandedover.*/
inout                    StringsFuncObjectInstanceName
/*inparameter
NameoftheFOinstance.*/
inout                    TCFOFunctionReference
                            refCBCompleteCreateFuncObject
/*inparameter
Reference   to   a   call   back   method   which   handles   the   initialization
mechanism.ANILreferencesignalsasynchronouscall.*/
):GdiCoFuncObjectRT
```

功能描述

未在驱动程序内创建实例，影响协调器内功能对象的控制结构的设置。

在异步操作的情况下，对功能对象的引用将通过完整的回调传递。如果没有给出回调参考，则该操作将同步执行。

返回值

返回对新创建的 GdiCoFuncObjectRT 的引用。

异步操作时为 NIL。

C.2.11.6 GdiCoFactory :: «F» gdiCreateFuncObjectRTByFuncId()

函数调用

```
gdiCreateFuncObjectRTByFuncId
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoVdRTrefVdRT
          /*inparameter
          containsthereferencetotheVD.TheFunctionObject                willbeconnected
          tothisVD.*/
          inout                  A_INT16nFuncId
          /*inparameter
          FunctionObjectvalue,towhichtheFOshallbecreated.TheFOvalue
          hastobespecifiedwithintheDCDofthedriver.*/
          inout                  GdiCoToCoordStreamBufferedrefCreateParameter
          /*inparameter
          thegivenStreamobjectcontainsthecreateparameteroftheVD.
          IfnocreateParameterexists,NILishandedover.*/
          inout                  StringsFuncObjectInstanceName
          /*inparameter
          NameoftheFOinstance.*/
          inout                  TCFOFunctionReference
                                    refCBCompleteCreateFuncObject
          /*inparameter
          Reference   to   a   call   back   method   which   handles   the   initialization
          mechanism.ANILreferencesignalsasynchronouscall.*/
```

```
):GdiCoFuncObjectRT
```

功能描述

未在驱动程序内创建实例。影响协调器内功能对象的控制结构的设置。

在异步操作的情况下，对功能对象的引用将通过完整的回调传递。如果没有给出回调参考，则该操作将同步执行。

返回值

返回对新创建的 GdiCoFuncObjectRT 的引用。

异步操作时为 NIL。

C.2.11.7 GdiCoFactory :: «F» gdiCreateFuncObjectRTByInterfaceClassName()

函数调用

```
gdiCreateFuncObjectRTByInterfaceClassName
          (
          inout                  unsignedlongulAppHnd
           /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoVdRTrefVdRT
          /*inparameter
          ContainsthereferencetotheapplicationVD.Thefunctionobjectwill
          beconnectedtothisVD.*/
          inout                  StringsInterfaceClassName
          /*inparameter
          ContainstheclassnameoftheFOtocreate.*/
          inout                  GdiCoToCoordStreamBufferedrefCreateParameter
          /*inparameter
          thegivenStreamobjectcontainsthecreateparameteroftheVD.
          IfnocreateParameterexists,NILishandedover.*/
          inout                  StringsFuncObjectInstanceName
          /*inparameter
          NameoftheFOinstance.*/
          in                     TCCheckCompleterefCBCheckComplete
          /*inparameter
          Referencetoacallbackmethodwhichhandlestheunsolicitedreportof
          theVDstatecheckcomplete.
          IfaNILreferenceisgiven,noeventwillberaisedbythecoordinator.
          */
          inout                  TCFOFunctionReference
                                    refCBCompleteCreateFuncObject
          /*inparameter
          Reference to a call back method which handles the initialization
          mechanism.ANILreferencesignalsasynchronouscall.*/
          ):GdiCoFuncObjectRT
```

功能描述

未在驱动程序内创建实例。影响协调器内功能对象的控制结构的设置。

在异步操作的情况下，对功能对象的引用将通过完整的回调传递。如果没有给出回调参考，则该操作将同步执行。

返回值

返回对新创建的 GdiCoFuncObjectRT 的引用。

异步操作时为 NIL。

C.2.11.8 GdiCoFactory :: «F» gdiCreateOperationRTByDescription()

函数调用

```
gdiCreateOperationRTByDescription
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                    GdiCoFuncObjectRTrefFuncObjectRT
          /*inparameter
          Containsthereferencetothefunctionobject.TheoperationObjectwill
          beconnectedtothisFO.*/
          inout                    GdiCoOperationCDrefOperationCD
          /*inparameter
          referencetotheGdiCoOperationCDwhichdescribestheOperation.*/
          inout                    StringsOperationObjectInstanceName
          /*inparameter
          instancename(alias)ofthenewoperationinstance.*/
          ):GdiCoOperationRT
```

功能描述

操作的创建在协调器内进行，并没有与驱动程序进行交互。该服务仅设置管理结构，管理结构不必显式删除，它将与 Function 对象实例一起自动删除。

返回值

返回对新创建的 GdiCoOperationRT 的引用。

C.2.11.9 GdiCoFactory :: «F» gdiCreateOperationRTByOperationClassName()

函数调用

```
gdiCreateOperationRTByOperationClassName
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                    GdiCoFuncObjectRTrefFuncObjectRT
          /*inparameter
          Containsthereferencetothefunctionobject.TheoperationObjectwill
          beconnectedtothisFO.*/
          inout                    StringsOperationClassName
          /*inparmeter
          Classname(DCD)oftheOperation,whichshouldbecreated.*/
          inout                    StringsOperationObjectInstanceName
          /*inparmeter
          Instancename,whichissettotheoperationinstance.*/
          ):GdiCoOperationRT
```

功能描述

操作的创建在协调器内进行，并没有与驱动程序进行交互。该服务仅设置管理结构，管理结构不必显式删除，它将与 Function 对象实例一起自动删除。

返回值

返回对新创建的 GdiCoOperationRT 的引用。

C.2.11.10 GdiCoFactory :: «F» gdiCreateOperationRTByOperationId()

函数调用

```
gdiCreateOperationRTByOperationId
          (
          inout                    unsignedlongulAppHnd
```

```
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoFuncObjectRTrefFuncObjectRT
            /*inparameter
            Containsthereferencetothefunctionobject.TheoperationObjectwill
            beconnectedtothisFO.*/
            inout                   A_INT16nOperationId
            /*inparameter
            containstheoperationIdaccordingtotheDescription(DCD)*/
            ):GdiCoOperationRT
```

功能描述

操作的创建在协调器内进行，并没有与驱动程序进行交互。该服务仅设置管理结构，管理结构不必显式删除，它将与 Function 对象实例一起自动删除。

返回值

返回对新创建的 GdiCoOperationRT 的引用。

C.2.11.11 GdiCoFactory :: «F» gdiCreateVdRTByDescription()

函数调用

```
gdiCreateVdRTByDescription
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoDriverRTrefDriverRT
            /*inparameter
            contains the reference to the driver. The new Application VD
            (administrationstructure)willbeconnectedtothisdriver.*/
            inout                   GdiCoModuleCDrefModuleCD
            /*inparameter
            Referencetoagivenmoduledescription.*/
            inout                   GdiCoToCoordStreamBufferedrefCreateParameter
            /*inparameter
            thegivenStreamobjectcontainsthecreateparameteroftheVD.
            IfnocreateParameterexists,NILishandedover.*/
            inout                   StringsVdInstanceName
            /*inparameter
            InstancenameofthenewVD.*/
            inout                   TCCheckCompleterefCBCheckComplete
            /*inparameter
            Referencetoacallbackmethodwhichhandlestheunsolicitedreportof
            theVDstatecheckcomplete.
            IfaNILreferenceisgiven,noeventwillberaisedbythecoordinator.
            */
            inout                   TCVDFunctionReferencerefCBCompleteCreateVd
            /*inparameter
            Reference to a call back method which handles the initialization
            mechanism.ANILreferencesignalsasynchronouscall.*/
            ):GdiCoVdRT
```

功能描述

在协调器和设备驱动程序中影响 VD 的设置

在异步操作的情况下，将通过完整的回调传递对 VD 的引用。如果没有给出回调参考，则该操作将同步执

行。

返回值

返回对新创建的 GdiCoVdRT 的引用。

异步操作时为 NIL。

C.2.11.12 GdiCoFactory :: «F» gdiCreateVdRTByModuleClassName()

函数调用

```
gdiCreateVdRTByModuleClassName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoDriverRTrefDriverRT
          /*inparameter
          contains the reference to the driver. The new Application VD
          (administrationstructure)willbeconnectedtothisdriver.*/
          inout                  StringsModuleClassName
          /*inparameter
          containstheClassNameofthemodule,accordingtotheDCD.*/
          inout                  GdiCoToCoordStreamBufferedrefCreateParameter
          /*inparameter
          thegivenStreamobjectcontainsthecreateparameteroftheVD.
          IfnocreateParameterexists,NILishandedover.*/
          inout                  StringsVdInstanceName
          /*inparameter
          InstancenameofthenewVD.*/
          in                     TCCheckCompleterefCBCheckComplete
          /*inparameter
          Referencetoacallbackmethodwhichhandlestheunsolicitedreportof
          theVDstatecheckcomplete.
          IfaNILreferenceisgiven,noeventwillberaisedbythecoordinator.
          */
          inout                  TCVDFunctionReferencerefCBCompleteCreateVd
          /*inparameter
          Reference to a call back method which handles the initialization
          mechanism.ANILreferencesignalsasynchronouscall.*/
          ):GdiCoVdRT
```

功能描述

在协调器和设备驱动程序中影响 VD 的设置。

在异步操作的情况下，将通过完整的回调传递对 VD 的引用。如果没有给出回调参考，则该操作将同步执行。

返回值

返回对新创建的 GdiCoVdRT 的引用。

异步操作时为 NIL。

C.2.11.13 GdiCoFactory :: «F» gdiCreateVdRTByModuleId()

函数调用

```
gdiCreateVdRTByModuleId
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
```

```
gdiGetWorkspaceByName()*/
inout                   GdiCoDriverRTrefDriver
/*inparameter
contains   the   reference   to   the   driver.   The   new   Application   VD
(administrationstructure)willbeconnectedtothisdriver.*/
inout                   A_INT16nModuleId
/*inparameter
ModuleID,towhichtheVDshallbecreated.ThemoduleIDhastobe
specifiedwithintheDCDofthedriverormoduledescription.*/
inout                   GdiCoToCoordStreamBufferedrefCreateParameter
/*inparameter
thegivenStreamobjectcontainsthecreateparameteroftheVD.
IfnocreateParameterexists,NILishandedover.*/
inout                   StringsVdInstanceName
/*inparameter
InstancenameofthenewVD.*/
inout                   TCVDFunctionReferencerefCBCompleteCreateVd
/*inparameter
Reference   to   a   call   back   method   which   handles   the   initialization
mechanism.ANILreferencesignalsasynchronouscall.*/
):GdiCoVdRT
```

功能描述

在协调器和设备驱动程序中影响 VD 的设置。

在异步操作的情况下，将通过完整的回调传递对 VD 的引用。如果没有给出回调参考，则该操作将同步执行。

返回值

如果成功，则返回对 GdiCoVDRT 对象的调用。

C.2.11.14 GdiCoFactory :: «F» gdiDeleteCommObjectRT()

函数调用

```
gdiDeleteCommObjectRT
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoCommObjectRTrefCommObjectRT
        /*inparameter
        ReferencetotheCommunicationObject.*/
        ):void
```

功能描述

在驱动程序和协调器内实现删除给定的通讯对象实例。

返回值

无

C.2.11.15 GdiCoFactory :: «F» gdiDeleteDriverRT()

函数调用

```
gdiDeleteDriverRT
         (
         inout                  unsignedlongulAppHnd
         /*inparameter
         IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
         gdiGetWorkspaceByName()*/
```

```
            inout                   GdiCoDriverRTrefDriverRT
            /*inparameter
            referencetotheDriverRTobject.*/
            ):void
```

功能描述

取消引用 GDI 驱动程序，如果驱动程序中仍存在该工作区的应用程序 VD，则这些应用程序将中止。如果驱动程序中包含其他应用程序 VD（其他工作区），则管理计数器递减。如果管理计数器为 0，则删除驱动程序的 Control VD 并卸载驱动程序。为了避免在延迟的时间内继续加载和卸载定期使用的驱动程序，需要定义卸载时间。

返回值

无

C.2.11.16 GdiCoFactory :: «F» gdiDeleteFuncObjectRT()

```
FunctionCall
gdiDeleteFuncObjectRT
            (
            inout                   unsigned long ulAppHnd
            /* in parameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)
            or by gdiGetWorkspaceByName() */
            inout                   GdiCoFuncObjectRT refFuncObjectRT
            /* Reference to a Function object within the VD. */
            ) : void
```

功能描述

如果函数对象是在设备驱动程序中设置的，则函数对象及其所有通信对象都将在驱动程序中删除。FO 的管理结构及其在协调器中的所有 COs 都将被删除。

返回值

无

C.2.11.17 GdiCoFactory :: «F» gdiDeleteVdRT()

函数调用

```
gdiDeleteVdRT
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoVdRTrefVdRT
            /*inparameter
            ReferencetoaVD.*/
            ):void
```

功能描述

如果删除设备驱动程序中的 VD 成功，则管理结构也将被删除。 VD 不得包含任何 Function 对象。

返回值

无

C.2.11.18 GdiCoFactory :: «F» gdiReleaseCDReference()

函数调用

```
gdiReleaseCDReference
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
```

```
gdiGetWorkspaceByName()*/
inout                    GdiCoObjectCDrefCD
/*inparameter
ReferencetotheGdiObjectDescriptionobject*/
):void
```

功能描述

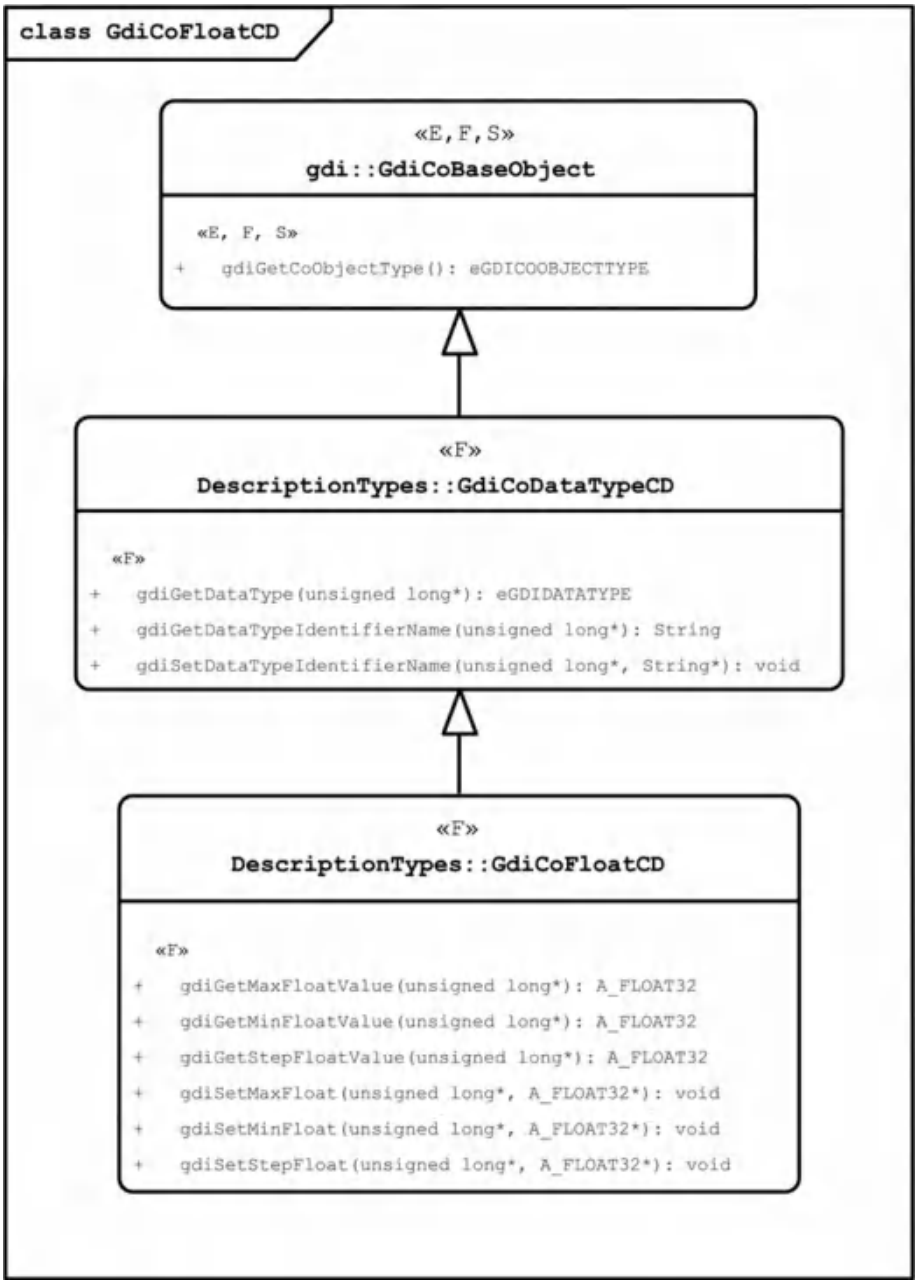
删除对 GdiCoDriverCD 对象的引用及其对描述对象的所有子引用。现有的 GDI 对象实例将不受影响。

返回值

无

C.2.12 «F» GdiCoFloatCD

包含与用于定义限制的 Float 类型有关的服务。



图C.12 Hierarchical diagram of GdiCoFloatCD

C.2.12.1 GdiCoFloatCD :: «F» gdiGetMaxFloatValue()

FunctionCall/函数调用

gdiGetMaxFloatValue

```
(
inout                   unsignedlongulAppHnd
/*                      inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):A_FLOAT32
```

功能描述

根据此数据返回基准的最大值。如果未明确地设置最大值，则其值为 3,402823466 E + 38。

返回值

根据此数据返回基准的最大值。

C.2.12.2 GdiCoFloatCD :: «F» gdiGetMinFloatValue()

函数调用

```
gdiGetMinFloatValue
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):A_FLOAT32
```

功能描述

根据此数据返回基准的最小值。如果未明确设置最小值，则其值为 1,175494351 E-38。

返回值

根据此数据返回基准的最大值。

C.2.12.3 GdiCoFloatCD :: «F» gdiGetStepFloatValue()

函数调用

```
gdiGetStepFloatValue
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
):A_FLOAT32
```

功能描述

返回模型的步长，该步长定义为从最小值开始的基准数据的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据此数据返回基准的步长。

C.2.12.4 GdiCoFloatCD :: «F» gdiSetMaxFloat()

函数调用

```
gdiSetMaxFloat
(
inout                   unsignedlongulAppHnd
/*inparameter
IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
gdiGetWorkspaceByName()*/
inout                   A_FLOAT32fLimitValue
/*inparameter
newmaximumvalue(<3,402823466E+39)*/
):void
```

功能描述

根据此模型定义基准的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.12.5 GdiCoFloatCD :: «F» gdiSetMinFloat()

函数调用

```
gdiSetMinFloat
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_FLOAT32fLimitValue
          /*inparameter
          newminimumvalue(>1,175494351E-39)*/
          ):void
```

功能描述

根据该模型定义基准的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.12.6 GdiCoFloatCD :: «F» gdiSetStepFloat()

函数调用

```
gdiSetStepFloat
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_FLOAT32fLimitValue
          /*inparameter
          newstepwidth(0<LimitValue<3,402823466E+38)*/
          ):void
```

功能描述

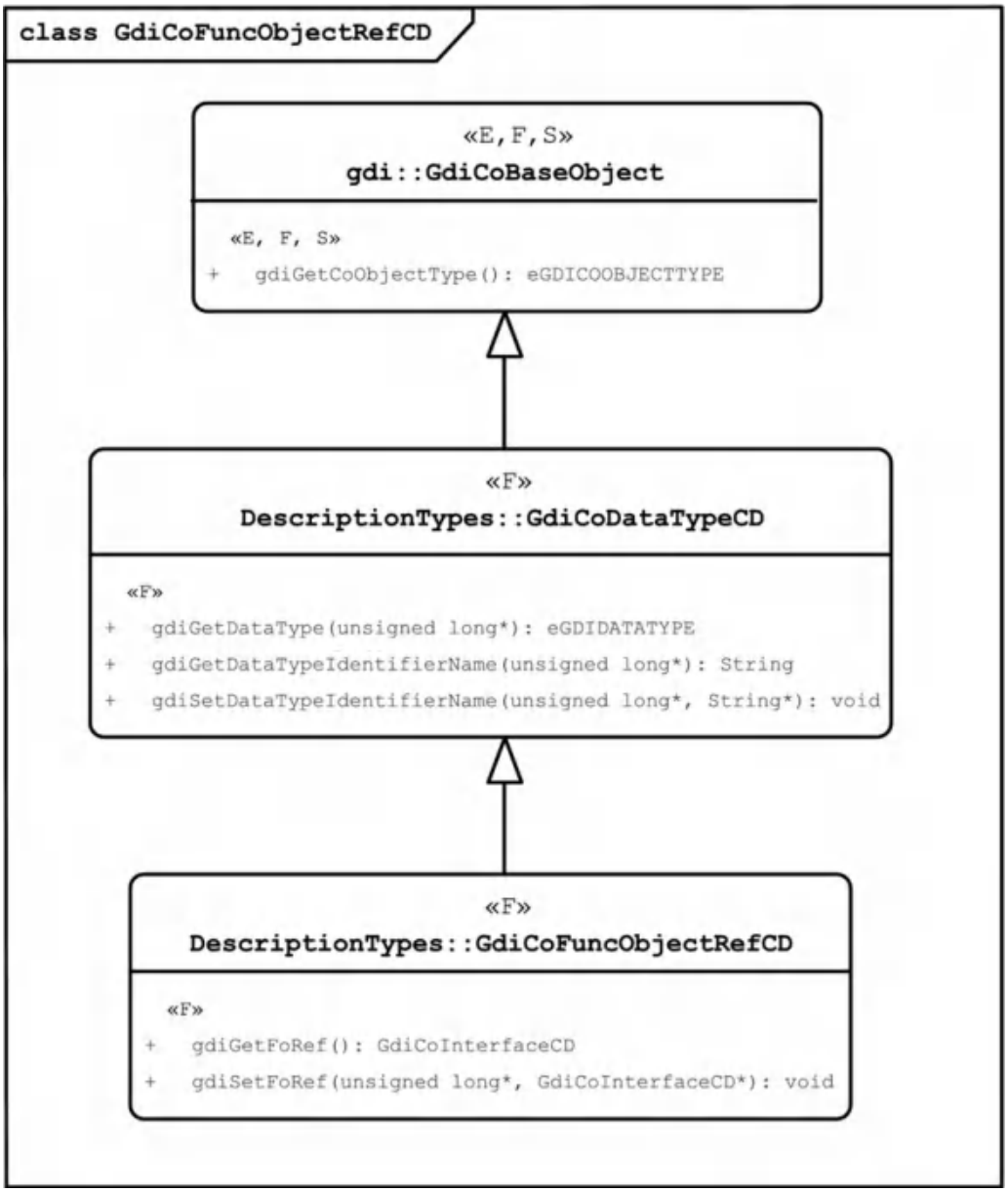
根据此模型在最小和最大之间定义有效数据的步长。最小值是起始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.13 «F» GdiCoFuncObjectRefCD

此接口包含用于处理类型函数对象引用的服务。



图C.13 Hierarchical diagram of GdiCoFuncObjectRefCD

C.2.13.1 GdiCoFuncObjectRefCD :: «F» gdiGetFoRef()

函数调用

```
gdiGetFoRef
          (
          ):GdiCoInterfaceCD
```

功能描述

返回所引用函数对象的描述。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

返回对动态接口的描述的引用。

C.2.13.2 GdiCoFuncObjectRefCD :: «F» gdiSetFoRef()

函数调用

```
gdiSetFoRef
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
```

```
gdiGetWorkspaceByName()*/
inout                   GdiCoInterfaceCDrefInterfaceCD
/*inparameter
containthereferencetothedescriptionofthedynamicInterface.*/
):void
```

功能说明

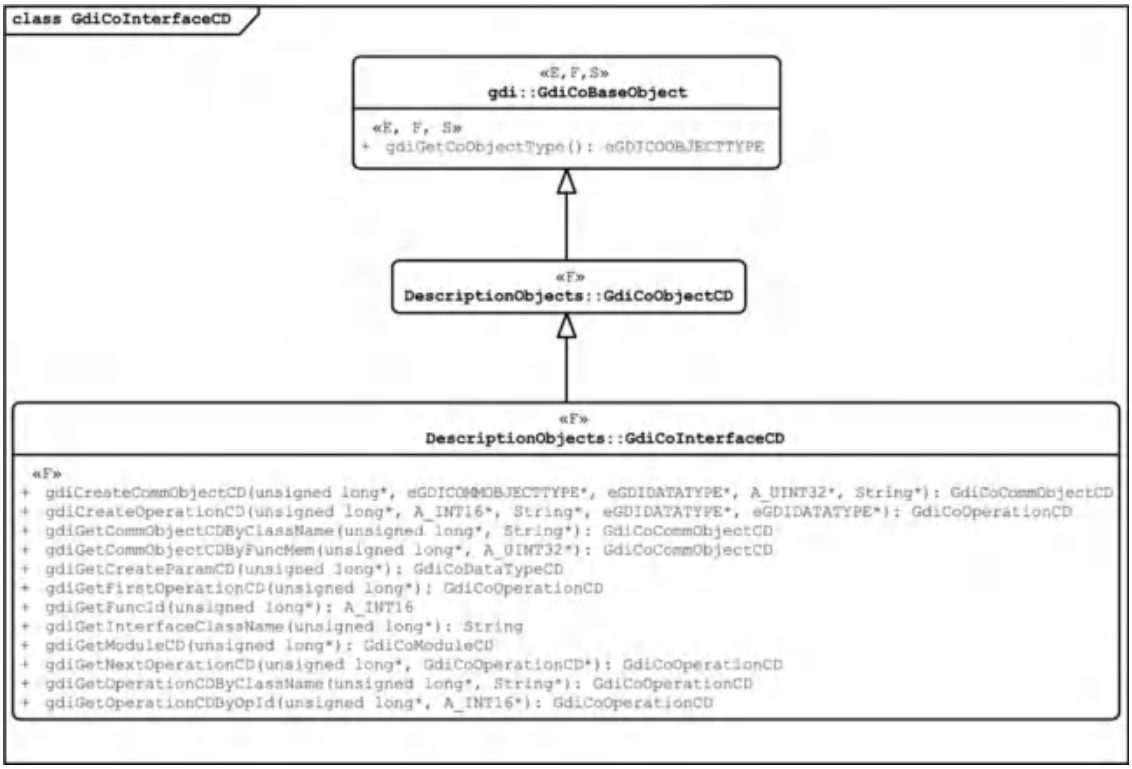
定义功能对象引用的功能 ID。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无。

C.2.14 «F» GdiCoInterfaceCD

该接口描述了一个功能对象。该对象引用是有效的，直到创建或引用下一个子描述为止。



图C.14 — Hierarchical diagram of GdiCoInterfaceCD

C.2.14.1 GdiCoInterfaceCD :: «F» gdiCreateCommObjectCD()

函数调用

```
gdiCreateCommObjectCD
        (
        inout                   unsigned long ulAppHnd
        /* in parameter
        Identifier of the application returned by gdiCreateWorkspace(...) or by
        gdiGetWorkspaceByName() */
        inout                   eGDICOMMOBJECTTYPE eCommObjectType
        /* in parameter
        Controls the instancing of the administration structure as attributes
        or as parameters. */
        inout                   eGDIDATATYPE eDataType
        /* in parameter
        contains the GDI type of the element. */
```

```
            inout                   A_UINT32 ulCommObjectFuncMem
            /* in parameter
            contains the absolute index of the communication object.
            The index must be unique inside the scope of this FO.
            Inherited COs are resolved. */
            inout                   String sCommObjectClassName
            /* in parameter
            contains the class name of the communication object. */
            ) : GdiCoCommObjectCD
```

功能描述

返回一个新的 GdiCoCommObjectCD 的调用

返回值

返回一个新的 GdiCoCommObjectCD 的调用

C.2.14.2 GdiCoInterfaceCD :: «F» gdiCreateOperationCD()

函数调用

```
gdiCreateOperationCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_INT16nOperationId
            /*inparameter
            containstheoperationID*/
            inout                   StringsOperationClassName
            /*inparameter
            containstheclassnameoftheOperation.*/
            inout                   eGDIDATATYPEeInType
            /*inparameter
            containsthetypeoftheinparameteroftheoperation.
            eDT_NO_DEFINITIONifvoid.*/
            inout                   eGDIDATATYPEeOutType
            /*inparameter
            containsthetypeoftheoutparameteroftheoperation.
            eDT_NO_DEFINITIONifvoid.*/
            ):GdiCoOperationCD
```

功能描述

返回一个新的 GdiCoCommObjectCD 的调用

返回值

返回一个新的 GdiOperationCD 的调用

C.2.14.3 GdiCoInterfaceCD :: «F» gdiGetCommObjectCDByClassName()

函数调用

```
gdiGetCommObjectCDByClassName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsCommObjectName
            /*inparameter
            containsthecommunicationobjectname.*/
```

```
        ):GdiCoCommObjectCD
```

功能描述

返回对与给定操作名称相对应的通信对象描述的调用。

返回值

返回与给定通信对象名称相对应的通信对象描述。

C.2.14.4 GdiCoInterfaceCD :: «F» gdiGetCommObjectCDByFuncMem()

函数调用

```
gdiGetCommObjectCDByFuncMem
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   A_UINT32ulFuncMem
        /*inparmeter
        containstheabsoluteindexoftheCommunicationObject*/
        ):GdiCoCommObjectCD
```

功能描述

返回对与有功能记忆的索引对应的通信对象描述的调用。绝对索引在 FO 范围内是唯一的。

返回值

返回与给定通信对象绝对索引相对应的通信对象描述

C.2.14.5 GdiCoInterfaceCD :: «F» gdiGetCreateParamCD()

函数调用

```
gdiGetCreateParamCD
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoDataTypeCD
```

功能描述

返回功能对象描述中创建的参数模型。

如果协调器处理 PID 文件时，已经设置了创建参数模型，那么可以通过调用此案例来获取创建参数模型。

返回值

返回创建参数类型，如函数对象描述中所述。

如果未使用创建参数功能，则将返回 NIL 引用。

C.2.14.6 GdiCoInterfaceCD :: «F» gdiGetFirstOperationCD()

函数调用

```
gdiGetFirstOperationCD
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoOperationCD
```

功能描述

在 FO 说明中创建对第一个操作说明的调用（如果存在）。

返回值

返回对第一个 gdiCoOperationCD 的引用（如果存在），否则返回 NIL。

C.2.14.7 GdiCoInterfaceCD :: «F» gdiGetFuncId()

函数调用

```
gdiGetFuncId
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT16
```

功能描述

返回 FO 描述中描述的 FO 的 ID。

如果协调器处理 PID 文件时，已经设置了 FO ID，则可以通过调用此案例来获取模块 ID。

返回值

返回 FO ID，在此功能对象中进行描述。

C.2.14.8 GdiCoInterfaceCD :: «F» gdiGetInterfaceClassName()

函数调用

```
gdiGetInterfaceClassName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

功能描述

返回此 FO 描述中描述的 FO 的类名。

如果协调器处理 PID 文件而已经设置了 FO 类名称，则可以通过调用此案例来获取 FO 类名称。

返回值

返回如功能对象说明中所描述的接口。

C.2.14.9 GdiCoInterfaceCD :: «F» gdiGetModuleCD()

函数调用

```
gdiGetModuleCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoModuleCD
```

功能描述

创建对上述 GdiCoModuleCD 的引用

返回值

返回对 GdiCoModuleCD 的调用。

C.2.14.10 GdiCoInterfaceCD :: «F» gdiGetNextOperationCD()

函数调用

```
gdiGetNextOperationCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
```

```
            inout                   GdiCoOperationCDrefLastElement
            /*inparameter
            containsthereferencetothelastoperationdescription*/
            ):GdiCoOperationCD
```

功能描述

在模块中创建对下一个操作描述的调用（如果存在）。

返回值

如果上述调用存在，则返回 GdiCoOperationCD，否则返回 NIL。

C.2.14.11 GdiCoInterfaceCD :: «F» gdiGetOperationCDByClassName()

函数调用

```
gdiGetOperationCDByClassName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsOperationName
            /*inparameter
            containstheoperationname.*/
            ):GdiCoOperationCD
```

功能描述

返回与给定操作名称相对应的操作描述。

返回值

返回与给定操作名称相对应的操作描述。

C.2.14.12 GdiCoInterfaceCD :: «F» gdiGetOperationCDByOpId()

函数调用

```
gdiGetOperationCDByOpId
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_INT16nOpId
            /*inparameter
            containstheFOID.*/
            ):GdiCoOperationCD
```

功能描述

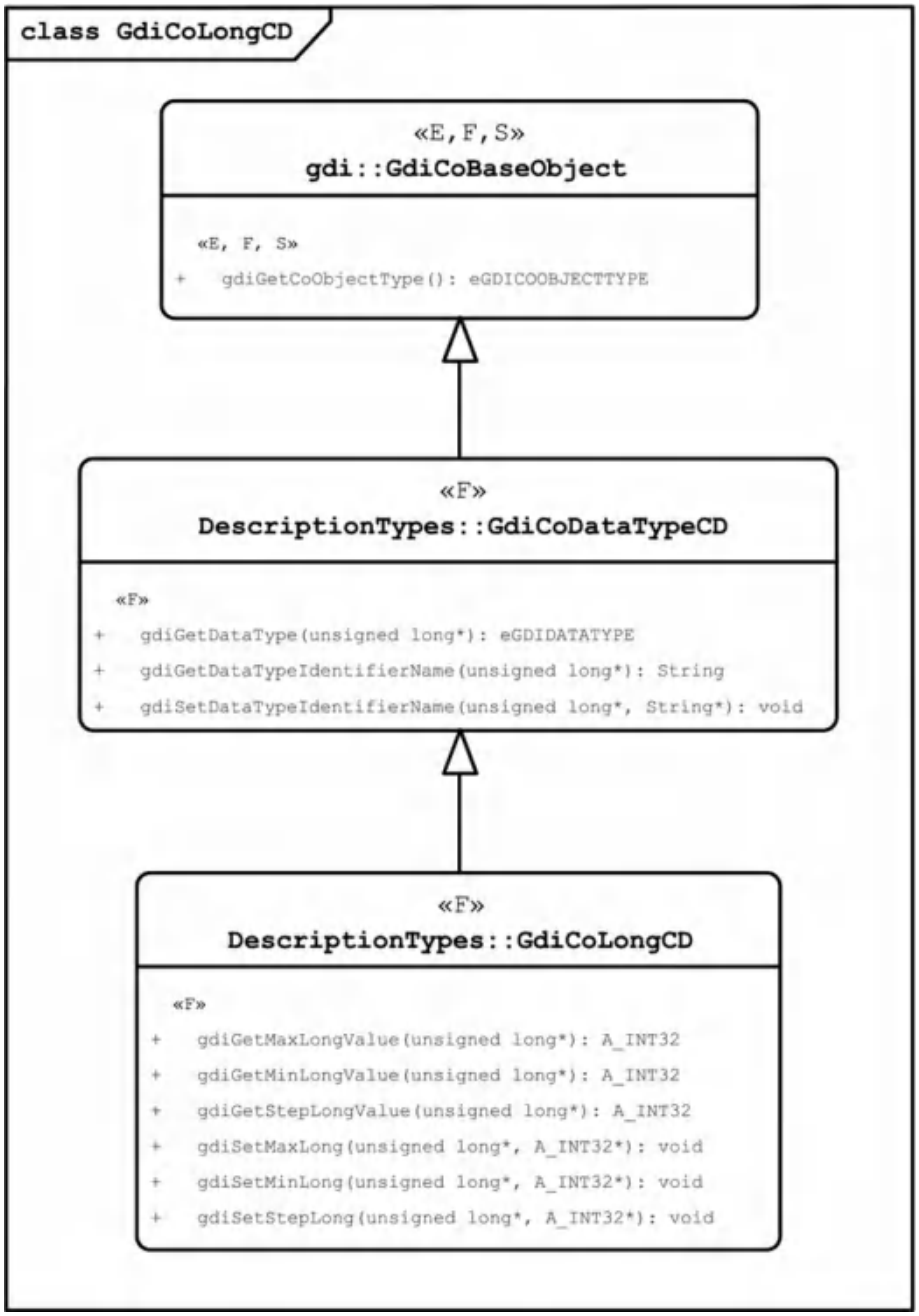
返回与给定操作 ID 对应的操作描述。

返回值

返回与给定操作 ID 对应的操作描述。

C.2.15 «F» GdiCoLongCD

包含与用于限制定义的类型有关的服务。



图C.15 Hierarchical diagram of GdiCoLongCD

C.2.15.1 GdiCoLongCD :: «F» gdiGetMaxLongValue()

函数调用

```
gdiGetMaxLongValue
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

根据数据类型返回其最大值。如果未明确设置最大值，则其值为2147483647。

返回值

根据数据类型返回其最大值。

C.2.15.2 GdiCoLongCD :: «F» gdiGetMinLongValue()

函数调用

```
gdiGetMinLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

根据数据类型返回其最小值。如果未明确设置最小值，则其值为 －2147483648。

返回值

根据数据类型返回其最小值。

C.2.15.3 GdiCoLongCD :: «F» gdiGetStepLongValue()

函数调用

```
gdiGetStepLongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT32
```

函数描述

返回类型的步长，该步长定义了从最小值开始的基准数据的有效元素。如果未明确定义最小值，则取值为1。

返回值

根据数据类型返回其步长。

C.2.15.4 GdiCoLongCD :: «F» gdiSetMaxLong()

函数调用

```
gdiSetMaxLong
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_INT32lLimitValue
          /*inparameter
          newmaximumvalue(<2147483648)*/
          ):void
```

函数描述

根据数据类型定义其最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.15.5 GdiCoLongCD :: «F» gdiSetMinLong()

函数调用

```
gdiSetMinLong
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
```

```
            gdiGetWorkspaceByName()*/
            inout                   A_INT32lLimitValue
            /*inparameter
            newminimumvalue(>-2147483649)*/
            ):void
```

函数描述

根据数据类型定义其最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.15.6 GdiCoLongCD :: «F» gdiSetStepLong()

函数调用

```
gdiSetStepLong
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_INT32lLimitValue
            /*inparameter
            newstepwidth(0<LimitValue<2147483647)*/
            ):void
```

函数描述

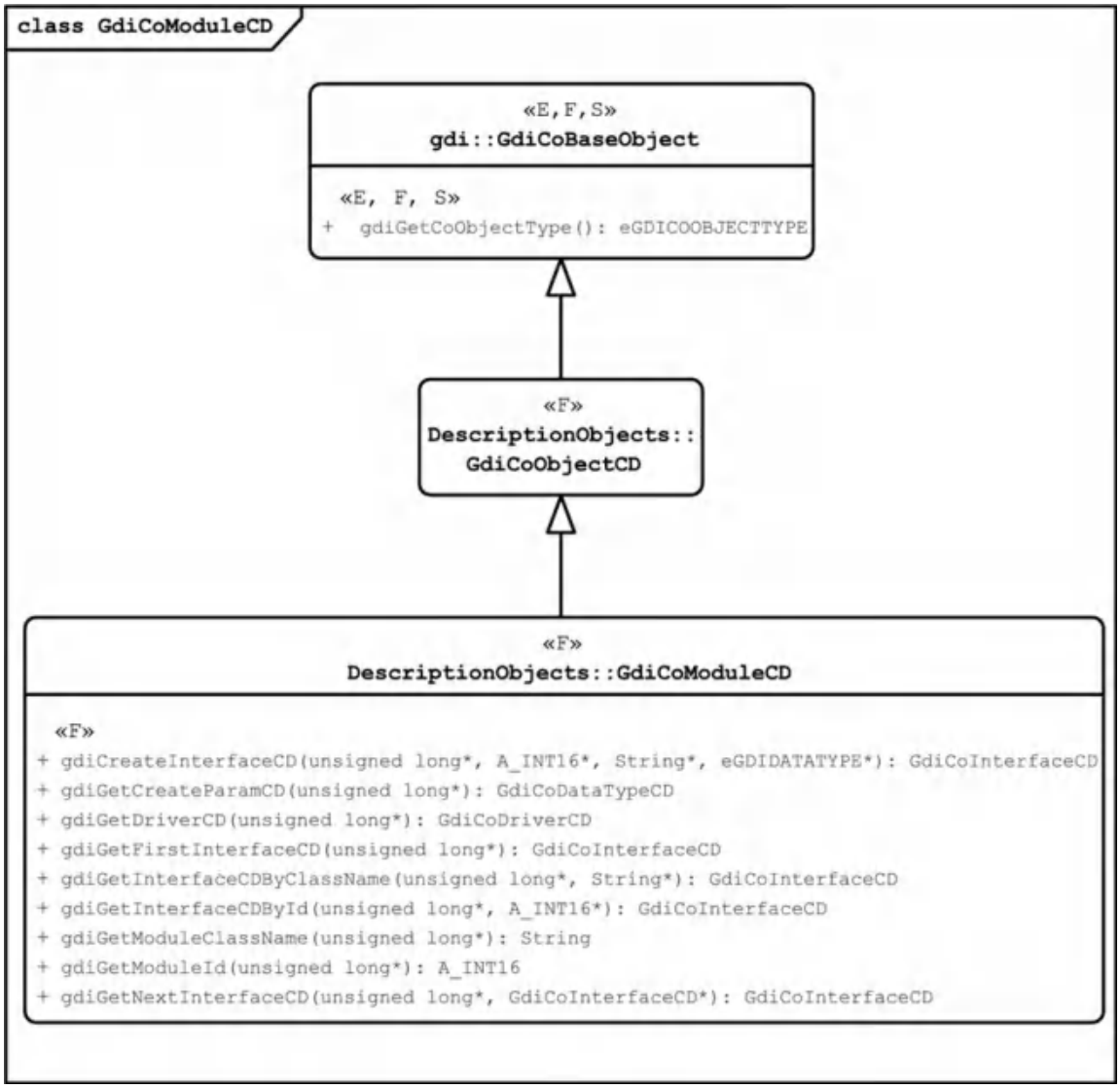
根据此类型在最小和最大之间定义有效数据的步长。最小值是初始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.16 «F» GdiCoModuleCD

该接口描述了一个模块。直到创建或引用下一个子描述为止，该对象引用都是有效的。



图C.16 GdiCoModuleCD层次图

C.2.16.1 GdiCoModuleCD :: «F» gdiCreateInterfaceCD()

函数调用

```
gdiCreateInterfaceCD
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   A_INT16nFuncId
        /*inparameter
        containstheFunctionobjectID*/
        inout                   StringsInterfaceClassName
        /*inparameter
        containstheclassnameoftheFuncObject.*/
        inout                   eGDIDATATYPEeCreateParamType
        /*inparmeter
        containsthetypeofthecreateparmeter.
        eDT_NO_DEFINITIONifnotnecessary.*/
        ):GdiCoInterfaceCD
```

函数描述

返回一个新的 GdiCoFuncObjectCD。

返回值

返回一个新的 GdiFuncObjectCD 对象。

C.2.16.2GdiCoModuleCD :: «F»gdiGetCreateParamCD()

函数调用

```
gdiGetCreateParamCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoDataTypeCD
```

函数描述

返回模块描述中描述的创建参数类型。

如果由于协调器处理 PID 文件而已经设置了创建参数类型，则可以通过调用此方法来获取创建参数类型。

返回值

返回创建参数类型，如模块说明中所述。

如果未使用 create 参数，则将返回 NIL 引用。

C.2.16.3 GdiCoModuleCD :: «F» gdiGetDriverCD()

函数调用

```
gdiGetDriverCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoDriverCD
```

函数描述

建立一个上层 GdiCoDriverCD 的引用。

返回值

返回 GdiCoDriverCD 的引用。

C.2.16.4 GdiCoModuleCD :: «F» gdiGetFirstInterfaceCD()

函数调用

```
gdiGetFirstInterfaceCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoInterfaceCD
```

函数描述

在模块描述中创建对第一个 OF 描述的引用（如果存在）。

返回值

在模块描述中创建对第一个 OF 描述的引用（如果存在），否则返回 NIL。

C.2.16.5 GdiCoModuleCD :: «F» gdiGetInterfaceCDByClassName()

函数调用

```
gdiGetInterfaceCDByClassName
          (
```

```
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   StringsInterfaceName
            /*inparameter
            containstheFOlassname.*/
            ):GdiCoInterfaceCD
```

函数描述

返回与给定的 FO 类名称相对应的 FO 描述。

返回值

返回与给定的 FO 类名称相对应的 FO 描述。

C.2.16.6 GdiCoModuleCD :: «F» gdiGetInterfaceCDById()

函数调用

```
gdiGetInterfaceCDById
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_INT16nFuncId
            /*inparameter
            containstheFOID.*/
            ):GdiCoInterfaceCD
```

函数描述

返回与给定 FO ID 对应的 FO 描述。

返回值

返回与给定 FO ID 对应的 FO 描述。如果此模块描述中不存在具有给定 ID 的 FO 描述，则返回 NIL。

C.2.16.7 GdiCoModuleCD :: «F» gdiGetModuleClassName()

函数调用

```
gdiGetModuleClassName
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):String
```

函数描述

返回模块描述中描述的模块类名。

如果由于协调器处理 PID 文件而已经设置了模块名称，则可以通过调用此方法来获取模块名称。

返回值

返回模块描述中描述的模块类名。

C.2.16.8 GdiCoModuleCD :: «F» gdiGetModuleId()

函数调用

```
gdiGetModuleId
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
```

```
            ):A_INT16
```

函数描述

返回模块描述中描述的模块 ID。

如果由于协调器处理 PID 文件而已经设置了模块 ID，则可以通过调用此方法来获取模块 ID。

返回值

返回模块描述中描述的模块 ID。

C.2.16.9 GdiCoModuleCD :: «F» gdiGetNextInterfaceCD()

函数调用

```
gdiGetNextInterfaceCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoInterfaceCDrefLastElement
            /*inparameter
            containsthereferencetothelastFunctionObjectdescription*/
            ):GdiCoInterfaceCD
```

函数描述

在模块描述中创建对下一个 FO 描述的引用（如果存在）。

返回值

返回 GdiFuncObjectCD（如果存在），否则返回 NIL。

C.2.17 «F» GdiCoObjectCD

这是所有 GDI 对象描述的抽象基本接口。

图C.17 GdiCoObjectCD层次图

C.2.18 «F» GdiCoObjectNavigator

该静态接口包含用于迭代 GDI 描述对象，实例和 GDI 类型的服务。

图C.18 GdiCoObjectNavigator层次图

C.2.18.1 GdiCoObjectNavigator :: «F» gdiGetCommObjectCD()

函数调用

```
gdiGetCommObjectCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoCommObjectRTrefCommObjectRT
          /*inparameter
          ContainstheReferencetotheCommunicationObject.*/
          ):GdiCoCommObjectCD
```

函数描述

生成对给定通信对象的描述引用作为 GdiCoCommObjectCD 对象。

返回值

返回 GdiCoCommObjectCD 的引用。

C.2.18.2 GdiCoObjectNavigator :: «F» gdiGetDriverRT()

函数调用

```
gdiGetDriverRT
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoVdRTrefVdRT
          /*inparameter
          containsthereferencetotheApplicationVdRT.*/
```

```
            ):GdiCoDriverRT
```

函数描述

创建对现有驱动程序对象（RT）的引用。

引用的 DriverRT 包含给定的 Application VdRT（作为参数传递）。

返回值

返回 GdiCoDriverRT.

C.2.18.3 GdiCoObjectNavigator :: «F» gdiGetFirstDriverCD()

函数调用

```
gdiGetFirstDriverCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoWorkspacerefWorkspace
            /*inparameter
            containsthereferencetotheworkspace.Thedriverwillbeconnected tothisworkspace.*/
            ):GdiCoDriverCD
```

函数描述

返回第一个 GdiCoDriverCD（如果存在）。

返回值

返回第一个 GdiCoDriverCD（如果存在），否则返回 NIL。

C.2.18.4 GdiCoObjectNavigator :: «F» gdiGetInterfaceCD()

函数调用

```
gdiGetInterfaceCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoFuncObjectRTrefFuncObjectRT
            /*inparameter
            ContainstheReferencetotheFunctionObject.*/
            ):GdiCoInterfaceCD
```

函数描述

生成对给定功能对象的描述的引用作为 GdiCoInterfaceCD 对象。

返回值

返回 GdiCoInterfaceCD 的引用。

C.2.18.5 GdiCoObjectNavigator :: «F» gdiGetNextDriverCD()

函数调用

```
gdiGetNextDriverCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoWorkspacerefWorkspace
            /*inparameter
            containsthereferencetotheworkspace.Thedriverwillbeconnected
            tothisworkspace.*/
```

```
            inout                   GdiCoDriverCDrefLastDriverCD
            /*inparameter
            containsthereferencetothelastDriverDescription.*/
            ):GdiCoDriverCD
```

函数描述

返回连接到工作区的下一个驱动程序描述。

返回值

返回 GdiCoDriverCD（如果存在），否则返回 NIL。

C.2.18.6 GdiCoObjectNavigator :: «F» gdiGetOperationCD()

函数调用

```
gdiGetOperationCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoOperationRTrefOperationRT
            /*inparameter
            ContainstheReferencetotheOperationRTobject.*/
            ):GdiCoOperationCD
```

函数描述

生成对给定 OperationRT 的描述的引用作为 GdiCoOperationCD 对象。

返回值

返回 GdiCoOperationCD 的引用。

C.2.18.7 GdiCoObjectNavigator :: «F» gdiGetVdRT()

函数调用

```
gdiGetVdRT
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoFuncObjectRTrefFuncObjectRT
            /*inparameter
            Containsthereferencetothefunctionobject.*/
            ):GdiCoVdRT
```

函数描述

创建对现有 Application VD 的引用。

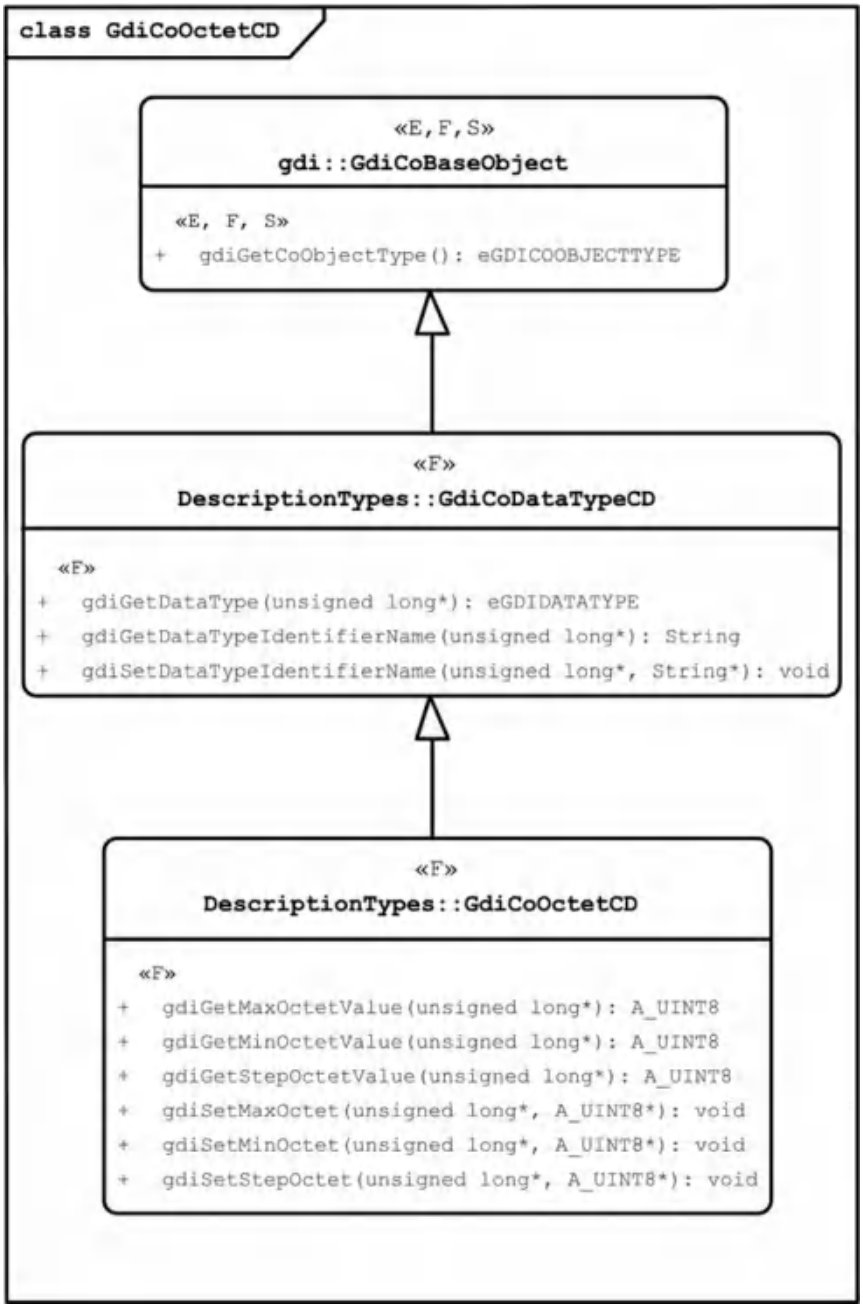
引用的应用程序 VD 包含给定的功能对象（作为参数传递）。

返回值

返回对与给定 FuncObjectRT 对应的 GdiCoVdRT 的引用。

C.2.19 «F» GdiCoOctetCD

包含用于限制定义的 unsigned Char 类型有关的服务。



图C.19 GdiCoOctetCD层次图

C.2.19.1 GdiCoOctetCD :: «F» gdiGetMaxOctetValue()

函数调用

```
gdiGetMaxOctetValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_UINT8
```

函数描述

根据数据类型返回其最大值。如果未明确设置最大值，则其值为 255。

返回值

根据数据类型返回其最大值。

C.2.19.2 GdiCoOctetCD :: «F» gdiGetMinOctetValue()

函数调用

```
gdiGetMinOctetValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT8
```

函数描述

根据数据类型返回其最小值。如果未明确设置最小值，则其值为 0.

返回值

根据数据类型返回其最小值。

C.2.19.3 GdiCoOctetCD :: «F» gdiGetStepOctetValue()

函数调用

```
gdiGetStepOctetValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT8
```

函数描述

根据此类型在最小和最大之间定义有效数据的步长。最小值是初始值。如果输入错误,则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

根据数据类型返回其步长。

C.2.19.4 GdiCoOctetCD :: «F» gdiSetMaxOctet()

函数调用

```
gdiSetMaxOctet
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT8ucLimitValue
          /*inparameter
          newmaximumvalue(<128)*/
          ):void
```

函数描述

根据数据类型定义数据的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.19.5 GdiCoOctetCD :: «F» gdiSetMinOctet()

函数调用

```
gdiSetMinOctet
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
```

```
            gdiGetWorkspaceByName()*/
            inout                   A_UINT8ucLimitValue
            /*                      newminimumvalue(0)*/
            ):void
```

函数描述

根据数据类型定义数据的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.19.6 GdiCoOctetCD :: «F» gdiSetStepOctet()

函数调用

```
gdiSetStepOctet
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_UINT8ucLimitValue
            /*inparameter
            newstepwidth(0<LimitValue<127)*/
            ):void
```

函数描述

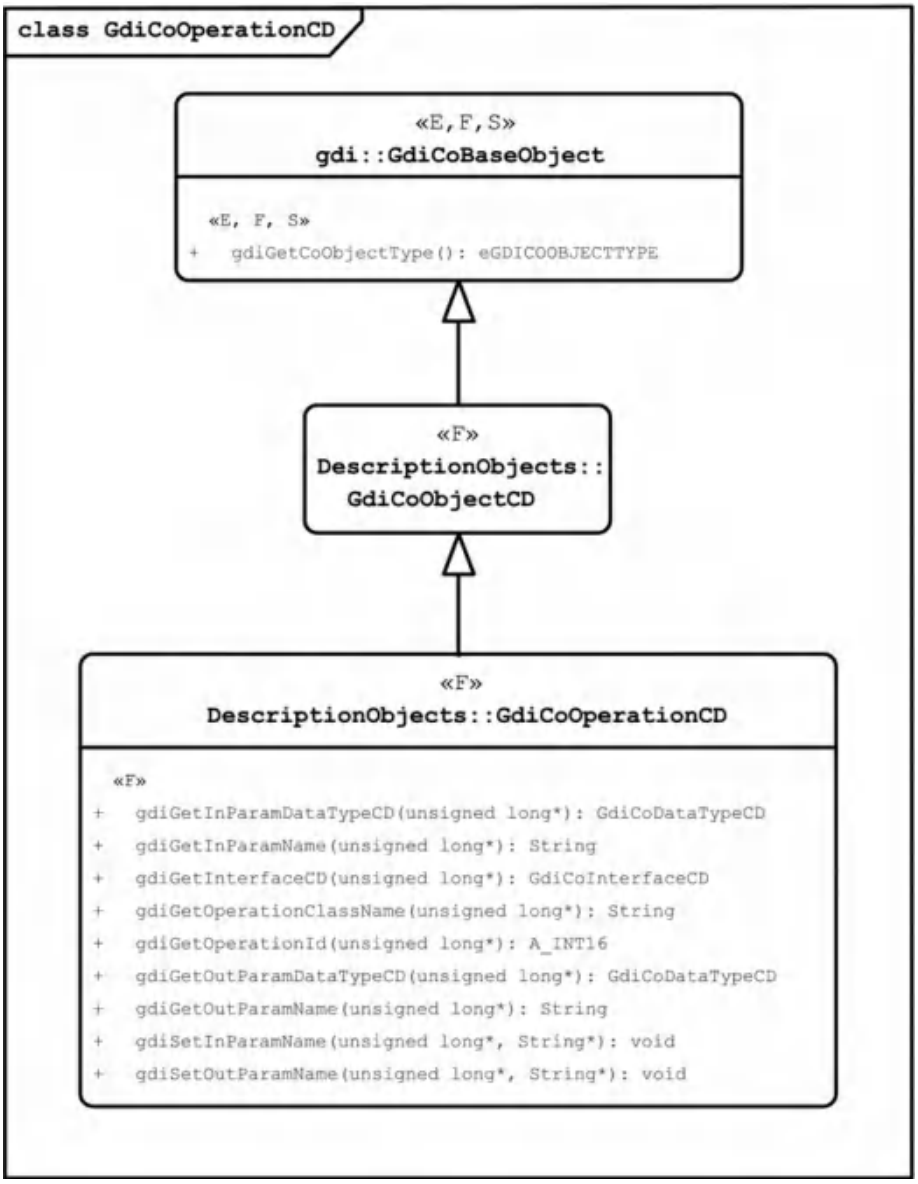
根据此类型在最小和最大之间定义有效数据的步长。最小值是初始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.20 «F» GdiCoOperationCD

ThisinterfacedescribesanOperationObject.Thisobjectreferenceisvalid,tillthecreationorreferencing the nextsubdescription.



图C.20 GdiCoOperationCD层次图

C.2.20.1 GdiCoOperationCD :: «F» gdiGetInParamDataTypeCD()

函数调用

```
gdiGetInParamDataTypeCD
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoDataTypeCD
```

函数描述

返回输入参数的类型描述。

如果由于协调器处理 PID 文件而已经设置了类型描述，则可以通过调用此方法来获取类型描述。

返回值

返回输入参数的类型描述。

如果未设置类型说明，则将返回 NULL 引用。

C.2.20.2 GdiCoOperationCD :: «F» gdiGetInParamName()

函数调用

```
gdiGetInParamName
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

函数描述

返回 In 参数的标识符名称。

返回值

返回 in 参数的标识符。

如果未设置参数名称，则返回一个空字符串。

C.2.20.3 GdiCoOperationCD :: «F» gdiGetInterfaceCD()

函数调用

```
gdiGetInterfaceCD
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoInterfaceCD
```

函数描述

建立一个上层 GdiCoFuncObjectCD 的引用。

返回值

返回 GdiCoFuncObjectCD。

C.2.20.4 GdiCoOperationCD :: «F» gdiGetOperationClassName()

函数调用

```
gdiGetOperationClassName
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

函数描述

返回此操作描述中描述的操作的类名。

如果由于协调程序处理 PID 文件而已经设置了操作类名称，则可以通过调用此方法来获取操作类名称。

返回值

返回操作的类名，如操作说明中所述。

如果没有设置类名，将返回一个空字符串。

C.2.20.5 GdiCoOperationCD :: «F» gdiGetOperationId()

函数调用

```
gdiGetOperationId
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT16
```

函数描述

返回操作描述中描述的操作的 ID。

如果由于协调器处理 PID 文件而已经设置了模块 ID，则可以通过调用此方法来获取模块 ID。

返回值

返回操作 ID，如操作说明中所述。

C.2.20.6 GdiCoOperationCD :: «F» gdiGetOutParamDataTypeCD()

函数调用

```
gdiGetOutParamDataTypeCD (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoDataTypeCD
```

函数描述

返回输出参数的类型描述。

如果由于协调器处理 PID 文件而已经设置了类型描述，则可以通过调用此方法来获取类型描述。

返回值

返回输出参数的类型描述。

如果未设置类型说明，则将返回 NIL 引用。

C.2.20.7 GdiCoOperationCD :: «F» gdiGetOutParamName()

函数调用

```
gdiGetOutParamName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):String
```

函数描述

返回 Out 参数的标识符名称。

返回值

返回 out 参数的标识符。

如果未设置参数名称，则返回空字符串。

C.2.20.8 GdiCoOperationCD :: «F» gdiSetInParamName()

函数调用

```
gdiSetInParamName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  StringsInParamName
          /*inparameter
          containsthenameoftheInParameter.*/
          ):void
```

函数描述

设置 in 参数的名称。

返回值

无。

C.2.20.9 GdiCoOperationCD :: «F» gdiSetOutParamName()

函数调用

```
gdiSetOutParamName
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  StringsOutParamName
          /*inparameter
          containsthenameoftheOutParameter.*/
          ):void
```

函数描述

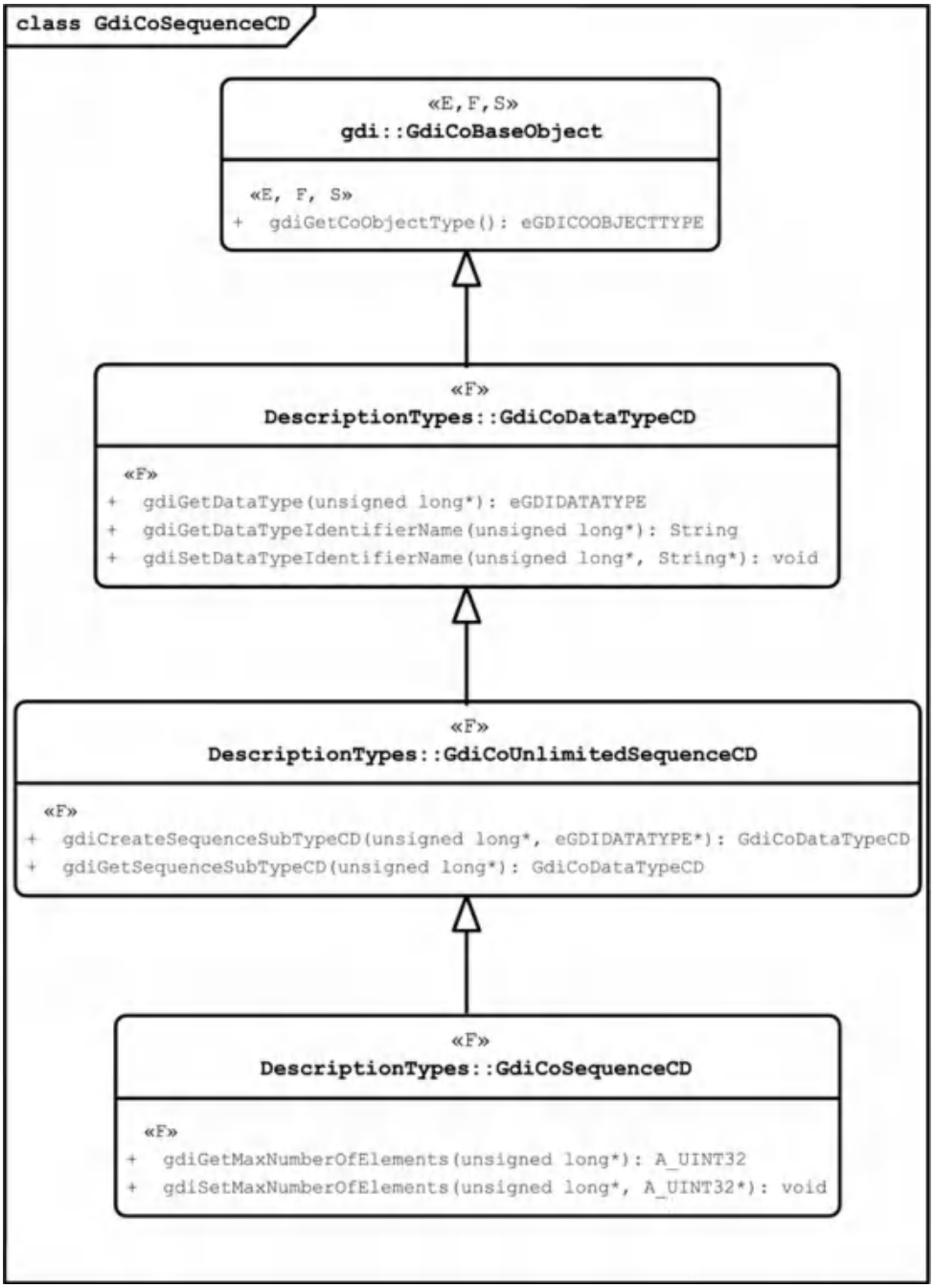
设置 in 参数的名称。

返回值

无。

C.2.21 «F» GdiCoSequenceCD

序列实例的描述。直到创建或引用下一个子描述为止，该对象引用都是有效的。



图C.21 GdiCoSequenceCD层次图

C.2.21.1 GdiCoSequenceCD :: «F» gdiGetMaxNumberOfElements()

函数调用

```
gdiGetMaxNumberOfElements
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):A_UINT32
```

函数描述

它用于检测序列的最大元素数。

返回值

返回序列中元素的最大数量。

C.2.21.2 GdiCoSequenceCD :: «F» gdiSetMaxNumberOfElements()

函数调用

```
gdiSetMaxNumberOfElements
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT32ulLimit
          /*inparameter
          containsthemaximalnumberofelementsofthelimitedsequence.*/
          ):void
```

函数描述

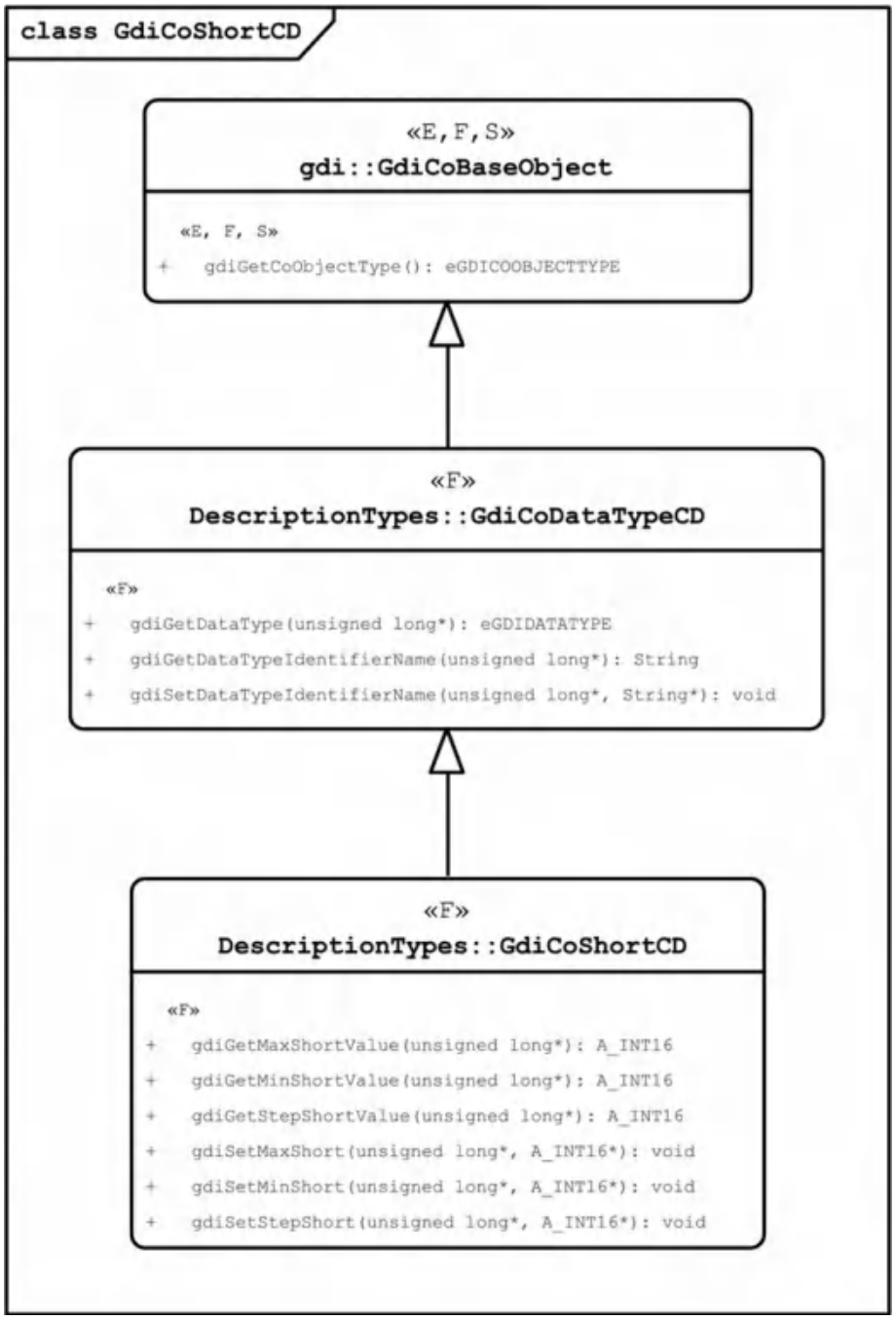
定义序列中元素的最大数量。

返回值

无。

C.2.22 «F» GdiCoShortCD

Containsservicesconcerningthe type Shortforthe definitionof restrictions.



图C.22 GdiCoShortCD层次图

C.2.22.1 GdiCoShortCD :: «F» gdiGetMaxShortValue()

函数调用

```
gdiGetMaxShortValue
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT16
```

函数描述

根据数据类型返回其最大值。如果未明确设置最大值，则其值为 32767.

返回值

根据数据类型返回其最大值。

C.2.22.2 GdiCoShortCD :: «F» gdiGetMinShortValue()

函数调用

```
gdiGetMinShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT16
```

函数描述

根据数据类型返回其最小值。如果未明确设置最小值，则其值为-32768.

返回值

根据数据类型返回其最小值。

C.2.22.3 GdiCoShortCD :: «F» gdiGetStepShortValue()

函数调用

```
gdiGetStepShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_INT16
```

函数描述

返回类型的步长，该步长定义从最小值开始的基准数据的有效元素。如果未明确定义最小值，则取值为1。

返回值

根据数据类型返回其步长。

C.2.22.4 GdiCoShortCD :: «F» gdiSetMaxShort()

函数调用

```
gdiSetMaxShort
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_INT16nLimitValue
          /*inparameter
          newmaximumvalue(<32768)*/
          ):void
```

函数描述

根据此类型定义基准的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.22.5 GdiCoShortCD :: «F» gdiSetMinShort()

函数调用

```
gdiSetMinShort
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
```

```
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_INT16nLimitValue
            /*inparameter
            newminimumvalue(>-32769)*/
            ):void
```

函数描述

根据该类型定义基准的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.22.6 GdiCoShortCD :: «F» gdiSetStepShort()

函数调用

```
gdiSetStepShort
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   A_INT16nLimitValue
            /*inparameter
            newstepwidth(0<LimitValue<32767)*/
            ):void
```

函数描述

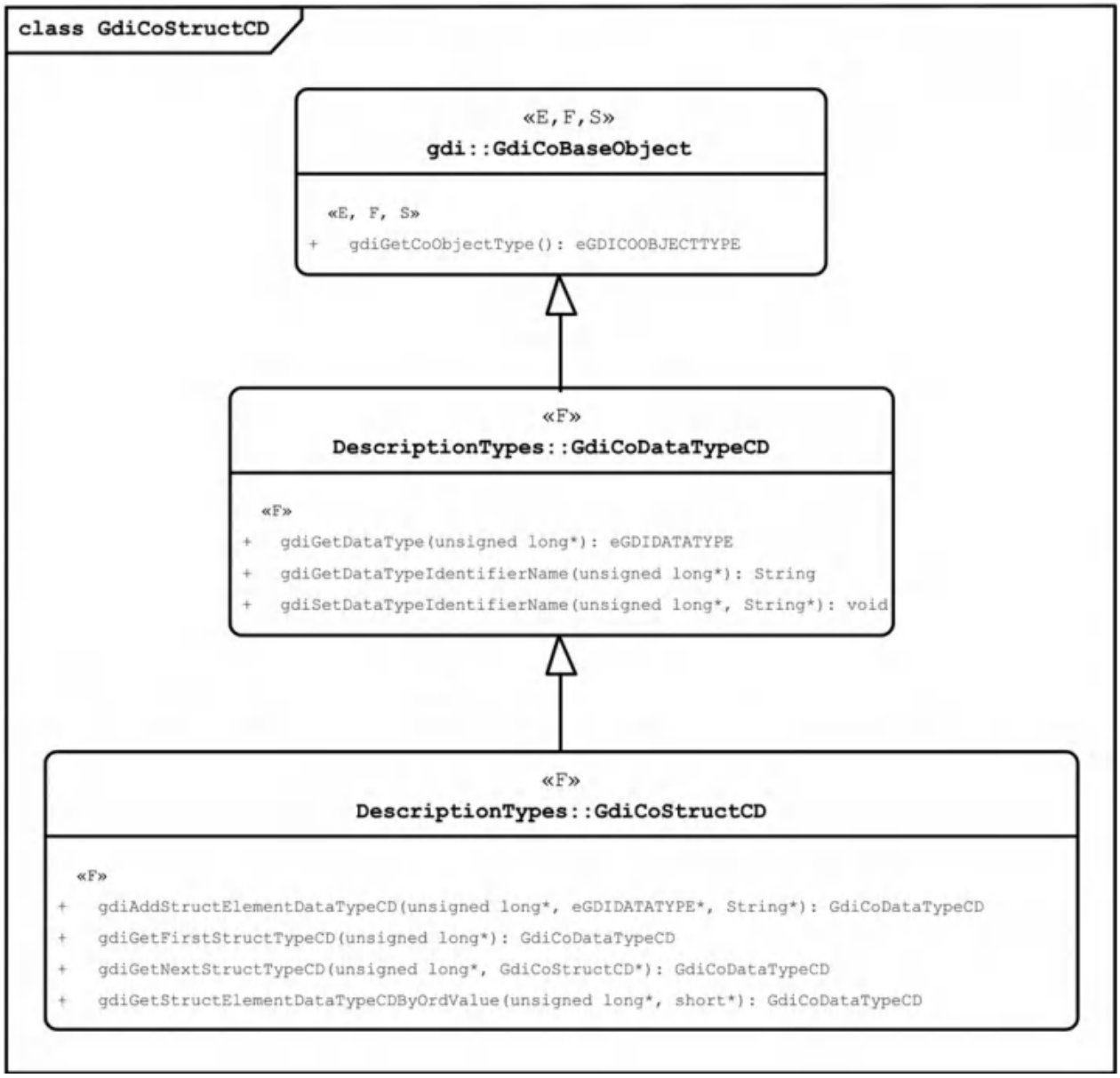
根据此类型在最小和最大之间定义有效数据的步长。最小值是起始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.23 «F» GdiCoStructCD

Containsservicesforthecreationof a structure type.



图C.23GdiCoStructCD层次图

C.2.23.1 GdiCoStructCD :: «F» gdiAddStructElementDataTypeCD()

函数调用

```
gdiAddStructElementDataTypeCD
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   eGDIDATATYPEeNewType
        /*inparameter
        GDItypeofthestructureelement
        ThevaluerangeincludestheGDITypeenumeration.*/
        inout                   StringsElementIdentifier
        /*inparameter
        containstheElementIdentifierofthenewstructelementasstring.The
        elementnamecanbeempty,ifnoIdentifierNameisused.*/
        ):GdiCoDataTypeCD
```

函数描述

向结构中添加其他元素。元素类型可能很复杂。通过返回值可以更详细地指定元素类型。如果不允许该元素的基本类型，则将引发异常。

第一个结构元素的序数值为 1。最后一个元素获得最高的序数值。

返回值

如果成功，则返回对元素类型描述的引用。

C.2.23.2 GdiCoStructCD :: «F» gdiGetFirstStructTypeCD()

函数调用

```
gdiGetFirstStructTypeCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoDataTypeCD
```

函数描述

生成对结构中第一个类型描述的引用。使用这种方法，可以检测到结构的所有子类型。

返回值

返回对结构的第一个子类型的引用（指针，句柄，类）。

C.2.23.3 GdiCoStructCD :: «F» gdiGetNextStructTypeCD()

函数调用

```
gdiGetNextStructTypeCD
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  GdiCoStructCDrefLastElement
          /*inparameter
          containsthereferencetothelaststructuredescriptionobject.*/
          ):GdiCoDataTypeCD
```

函数描述

生成对结构中第一个类型描述的引用。使用这种方法，可以检测到结构的所有子类型。

使用这种方法，可以检测结构的所有子类型。

返回值

返回对结构的下一个子类型的引用（指针，句柄，类）。

C.2.23.4 GdiCoStructCD :: «F» gdiGetStructElementDataTypeCDByOrdValue()

函数调用

```
gdiGetStructElementDataTypeCDByOrdValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  shortnOrdValue
          /*inparameter
          containstheselectoroftherequestedstructmember.*/
          ):GdiCoDataTypeCD
```

函数描述

返回对结构元素的引用。返回基本类型作为参考。一个描述结构元素在结构中的位置的数值用作选择器。

可以通过 gdiGetType()检测结构元素的实际类型（如果未知）。然后，该参考可以用作对检测到的数据类

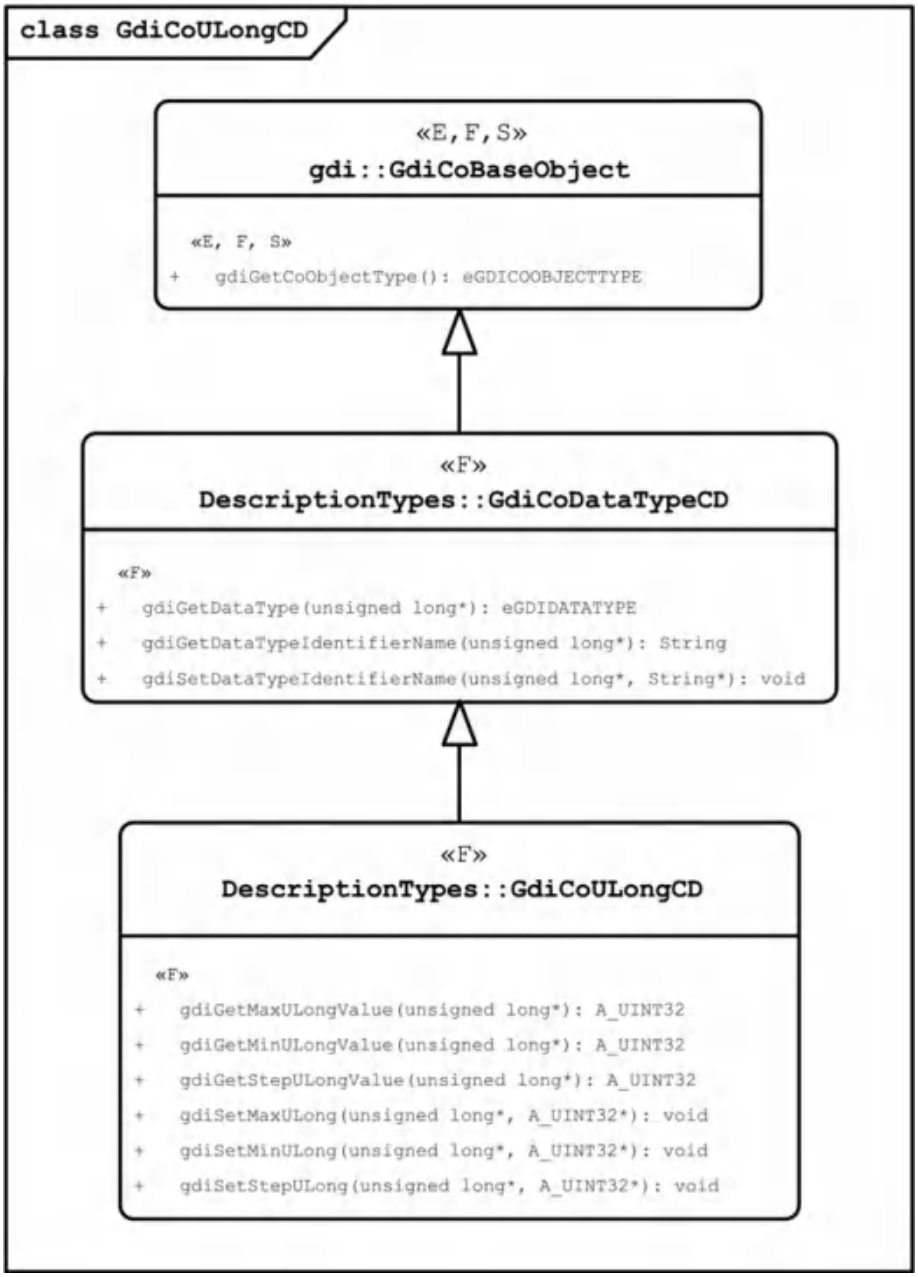
型描述的参考。

返回值

返回一个 GdiCoDataTypeCD 的引用。

C.2.24 «F» GdiCoULongCD

包含与 unsigned Long 类型有关的服务，用于定义限制。



图C.24 GdiCoULongCD层次图

C.2.24.1 GdiCoULongCD :: «F» gdiGetMaxULongValue()

函数调用

```
gdiGetMaxULongValue
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
```

```
):A_UINT32
```

函数描述

根据数据类型返回其最大值。如果未明确设置最大值，则其值为 4294967295。

返回值

根据数据类型返回其最大值。

C.2.24.2 GdiCoULongCD :: «F» gdiGetMinULongValue()

函数调用

```
gdiGetMinULongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

函数描述

根据数据类型返回其最小值。如果未明确设置最小值，则其值为 0。

返回值

根据数据类型返回其最小值。

C.2.24.3 GdiCoULongCD :: «F» gdiGetStepULongValue()

函数调用

```
gdiGetStepULongValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT32
```

函数描述

返回类型的步长，该步长定义从最小值开始的基准数据的有效元素。如果未明确定义最小值，则取值为 1。

返回值

根据数据类型返回其步长。

C.2.24.4 GdiCoULongCD :: «F» gdiSetMaxULong()

函数调用

```
gdiSetMaxULong
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT32ulLimitValue
          /*newmaximumvalue(<4294967296)*/
          ):void
```

函数描述

根据此类型定义基准的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.24.5 GdiCoULongCD :: «F» gdiSetMinULong()

函数调用

```
gdiSetMinULong
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT32ulLimitValue
          /*inparameter
          newminimumvalue(0)*/
          ):void
```

函数描述

根据该类型定义基准的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.24.6 GdiCoULongCD :: «F» gdiSetStepULong()

函数调用

```
gdiSetStepULong
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT32ulLimitValue
          /*inparameter
          newstepwidth(0<LimitValue<4294967295)*/
          ):void
```

函数描述

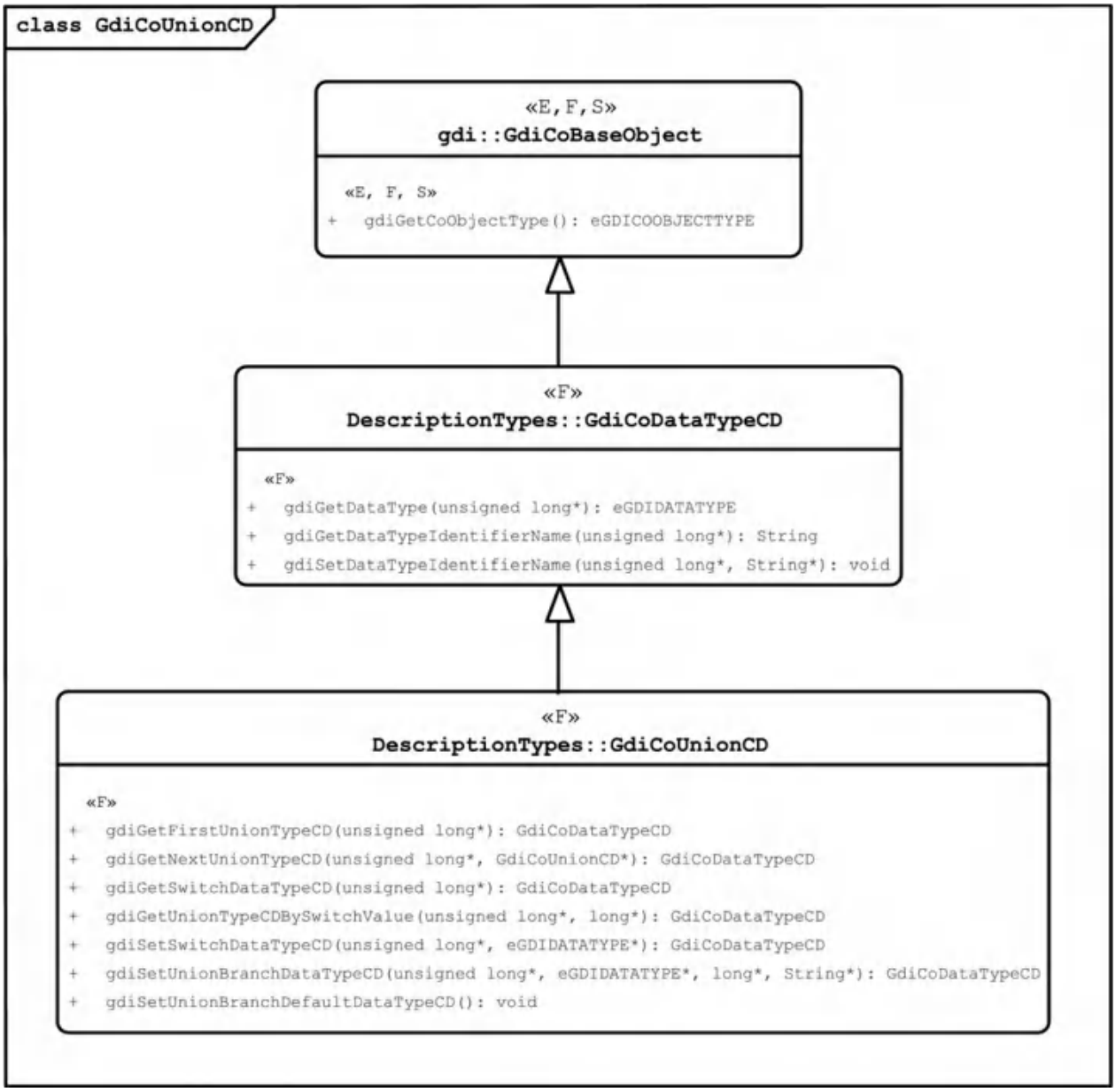
根据此类型，在最小值和最大值之间定义有效数据的步长。最小值是起始值。如果输入错误，则可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序会读取它以获取详细的类型信息。

返回值

无

C.2.25 «F» GdiCoUnionCD

Containsservicesforthecreationandexpansionofa Uniontype.Thisobjectreferenceisvalid,tillthe creationorreferencingthe next subdescription.



图C.25 GdiCoUnionCD层次图

C.2.25.1 GdiCoUnionCD :: «F» gdiGetFirstUnionTypeCD()

函数调用

```
gdiGetFirstUnionTypeCD
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoDataTypeCD
```

函数描述

生成对联合体中第一个类型描述的引用。使用这种方法，可以检测联合的所有子类型。

返回值

返回对联合体的第一个子类型的引用（指针，句柄，类）。

C.2.25.2 GdiCoUnionCD :: «F» gdiGetNextUnionTypeCD()

函数调用

```
gdiGetNextUnionTypeCD
```

```
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   GdiCoUnionCDrefLastElement
            /*inparameter
            containsthereferencetothelastUniondescriptionobject.*/
            ):GdiCoDataTypeCD
```

函数描述

生成对 Union 中下一个 Type 描述的引用。使用这种方法，可以检测联合的所有子类型。

返回值

返回对联合的下一个子类型的引用（指针，句柄，类）。

C.2.25.3 GdiCoUnionCD :: «F» gdiGetSwitchDataTypeCD()

函数调用

```
gdiGetSwitchDataTypeCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):GdiCoDataTypeCD
```

函数描述

生成对 GdiDescriptionDataType 的引用。开关值的实际类型由服务 gdiGetType（）相对于返回的引用进行检测。

返回值

返回开关值的类型（如果已设置）。

C.2.25.4 GdiCoUnionCD :: «F» gdiGetUnionTypeCDBySwitchValue()

函数调用

```
gdiGetUnionTypeCDBySwitchValue
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   longlSwitchValue
            /*inparameter
            Switchvalueoftheunionelement,whichisconnectedwiththisdatatype.*/
            ):GdiCoDataTypeCD
```

函数描述

返回对联合体的覆盖元素的引用。基本类型作为引用返回。开关值用作选择器。联合体元素的实际类型（如果未知）可以通过 gdiGetType（）来检测。然后，可以将该引用用作对检测到的数据类型的引用。

返回值

返回对基准的引用。

C.2.25.5 GdiCoUnionCD :: «F» gdiSetSwitchDataTypeCD()

函数调用

```
gdiSetSwitchDataTypeCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
```

```
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orbygdiGetWorkspaceByNa
            me()*/
            inout                   eGDIDATATYPEeNewType
            /*inparameter
            GDItypeoftheelement
            ThevaluerangeincludesbothASAMDatatypeenumerationandGDIType
            enumeration*/
            ):GdiCoDataTypeCD
```

函数描述

定义开关类型。这必须是标量类型。

返回值

如果成功，则返回对元素的类型描述的引用。

C.2.25.6 GdiCoUnionCD :: «F» gdiSetUnionBranchDataTypeCD()

函数调用

```
gdiSetUnionBranchDataTypeCD
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            inout                   eGDIDATATYPEeNewType
            /*inparameter
            ThevaluerangeincludesbothASAMDatatypeenumerationandGDIType
            enumeration*/
            inout                   longlSwitchValue
            /*inparameter
            Switchvalue,whichisconnectedwiththisdatatype.*/
            inout                   StringsBranch
            /*inparmeter
            containsthesectionnameofthenewUnionElementasString.ThisString
            canbeempty,isnoBranchnameisused.*/
            ):GdiCoDataTypeCD
```

函数描述

向联合体添加附加元素。元素类型可能很复杂。通过返回值，可以更详细地指定元素类型。如果不允许元素的基本类型，则将引发异常。

返回值

如果成功，则返回对元素的类型说明的引用。

C.2.25.7 GdiCoUnionCD :: «F» gdiSetUnionBranchDefaultDataTypeCD()

函数调用

```
gdiSetUnionBranchDefaultDataTypeCD
          (
           ):void
```

函数描述

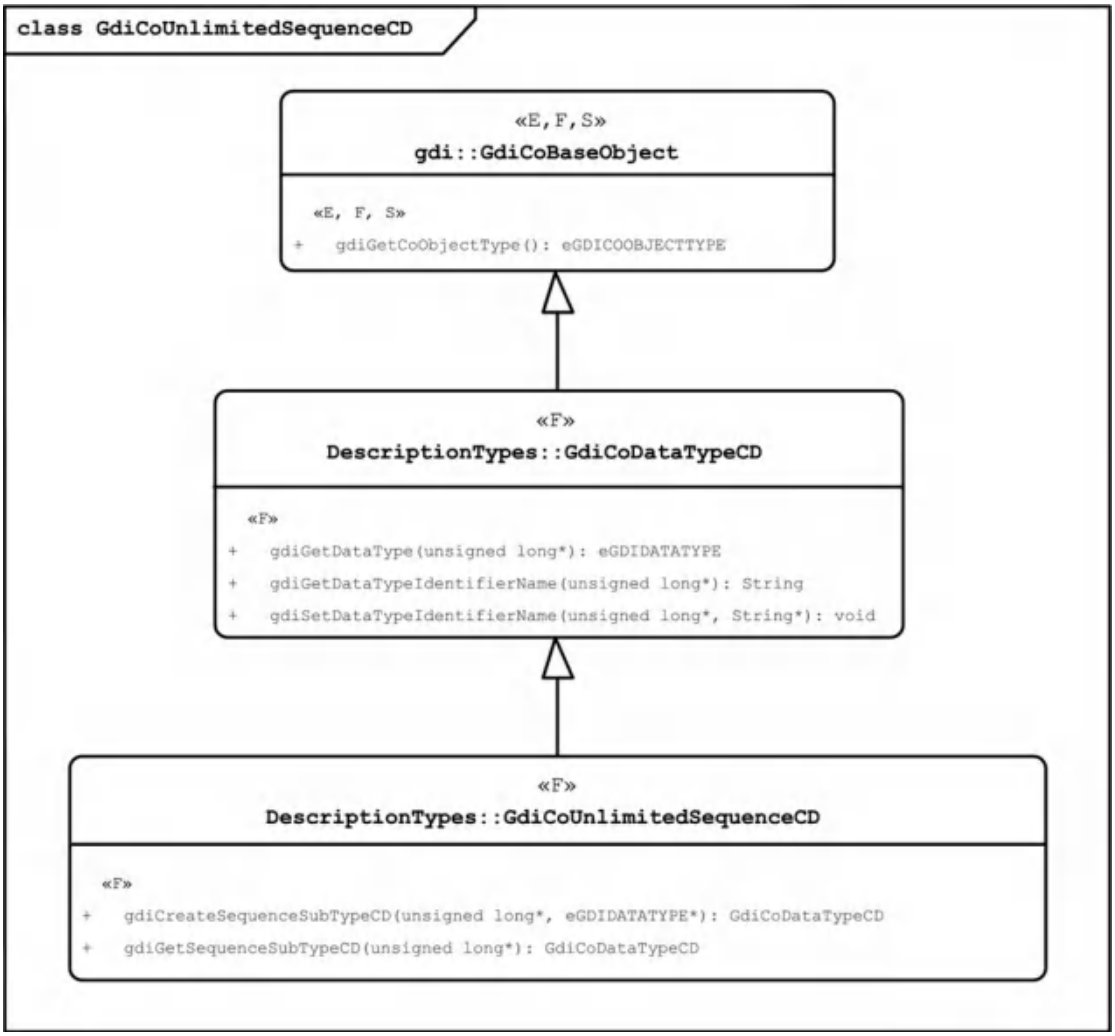
设置联合体的默认元素。元素类型可能很复杂。通过返回值，可以更详细地指定元素类型。如果不允许元素的基本类型，则将引发异常。

返回值

无。

C.2.26 «F» GdiCoUnlimitedSequenceCD

无限序列类型的实例描述。此对象引用有效，直到创建或引用下一个子描述为止。



图C.26-GDIConLimitedSequenceCD的层次图

C.2.26.1 GdiCoUnlimitedSequenceCD :: «F» gdiCreateSequenceSubTypeCD()

函数调用

```
gdiCreateSequenceSubTypeCD
    (
    inout  unsigned long ulAppHnd
    /* in parameter
    Identifier of the application returned by gdiCreateWorkspace(...) or by
    gdiGetWorkspaceByName() */
    inout  eGDIDATATYPE eNewType
    /* in parameter
    GDI type of the Sequence elements
    The value range includes both ASAM Datatype enumeration and GDI Type
    enumeration. */
    ) : GdiCoDataTypeCD
```

函数描述

描述无限序列的类型和元素数。元素类型可能很复杂。通过返回值，可以更详细地指定元素类型。

返回值

如果成功，则返回对序列元素的类型描述的引用。

C.2.26.2 GdiCoUnlimitedSequenceCD :: «F» gdiGetSequenceSubTypeCD()

函数调用

```
gdiGetSequenceSubTypeCD
```

```
(
 inout                    unsignedlongulAppHnd
 /*inparameter
 IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
 gdiGetWorkspaceByName()*/
 ):GdiCoDataTypeCD
```

函数描述

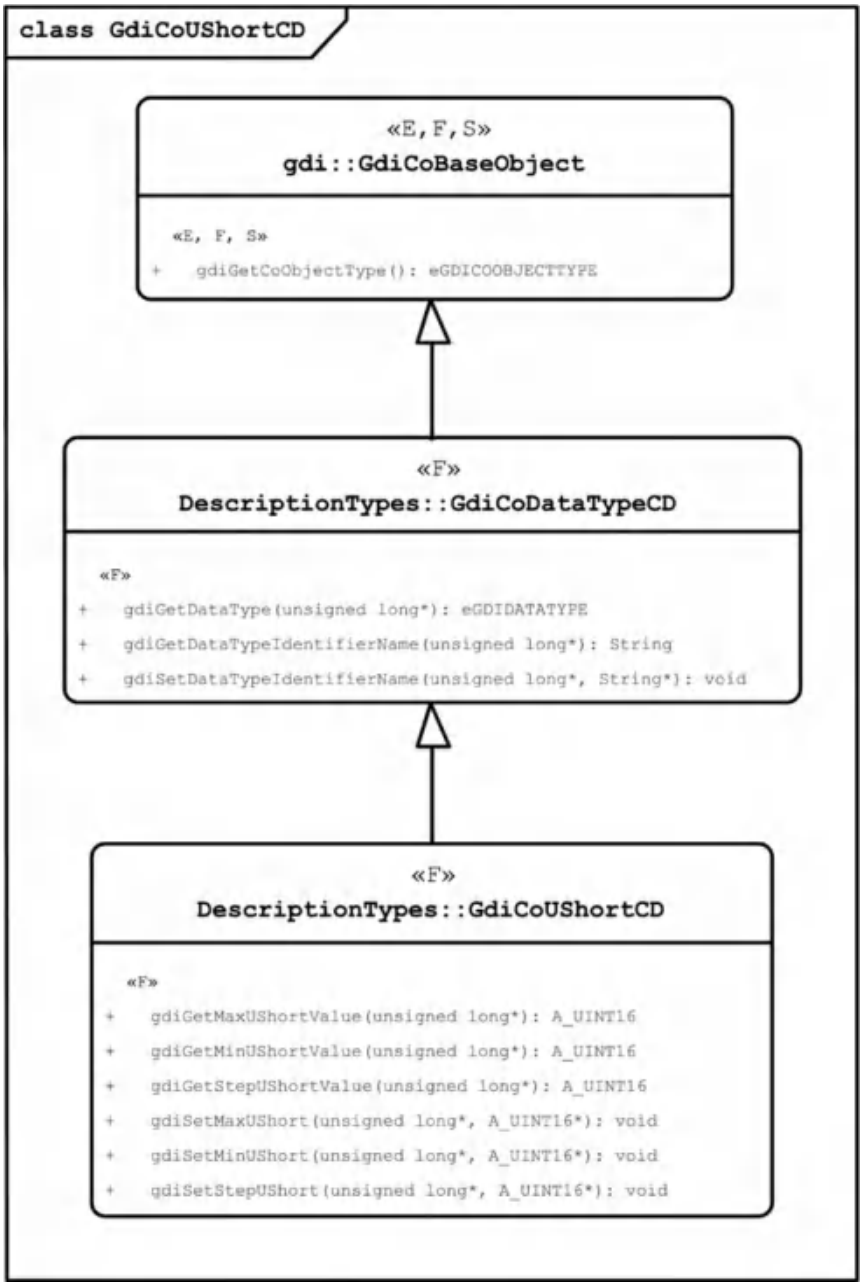
返回对子数据类型说明的引用。

返回值

返回对子类型描述的引用（如果存在），否则返回零引用。

C.2.27 «F» GdiCoUShortCD

包含与定义限制的无符号短整型有关的服务。



图C.27-GdiCoUShortCD的层次结构图

C.2.27.1 GdiCoUShortCD :: «F» gdiGetMaxUShortValue()

函数调用

```
gdiGetMaxUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT16
```

函数描述

返回基于此类型的基准的最大值。如果没有显式设置最大值，则取 65535。

返回值

返回基于此类型的基准的最大值。

C.2.27.2 GdiCoUShortCD :: «F» gdiGetMinUShortValue()

函数调用

```
gdiGetMinUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT16
```

函数描述

返回基于此类型的基准的最小值。如果未显式设置最小值，则取值 0。

返回值

返回基于此类型的基准的最小值。

C.2.27.3 GdiCoUShortCD :: «F» gdiGetStepUShortValue()

函数调用

```
gdiGetStepUShortValue
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):A_UINT16
```

函数描述

返回类型的步长，步长定义了从最小值开始的有效基准元素。如果未显式定义最小值，则取值 1。

返回值

返回基于此类型的基准的步长。

C.2.27.4 GdiCoUShortCD :: «F» gdiSetMaxUShort()

函数调用

```
gdiSetMaxUShort
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  A_UINT16unLimitValue
          /*inparameter
          newmaximumvalue(<32768)*/
          ):void
```

函数描述

定义基于此类型的基准的最大值。该值对与设备驱动程序的通信没有影响。某些应用程序读取它以获取详细的类型信息。

返回值

无。

C.2.27.5 GdiCoUShortCD :: «F» gdiSetMinUShort()

函数调用

```
gdiSetMinUShort
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   A_UINT16unLimitValue
        /*inparameter
        newminimumvalue(0)*/
        ):void
```

函数描述

定义基于此类型基准的最小值。该值对与设备驱动程序的通信没有影响。某些应用程序读取它以获取详细的类型信息。

返回值

无。

C.2.27.6 GdiCoUShortCD :: «F» gdiSetStepUShort()

函数调用

```
gdiSetStepUShort
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   A_UINT16unLimitValue
        /*inparameter
        newstepwidth(0<LimitValue<65535)*/
        ):void
```

函数描述

基于此类型在最小值和最大值之间定义有效数据的步长。最小值是起始值。如果输入错误，可能会出现空集。该值对与设备驱动程序的通信没有影响。某些应用程序读取它以获取详细的类型信息。

返回值

无。

C.2.28 «F» GdiCoVdIterator

此接口包含虚拟磁盘迭代器的应用程序服务，所有已在工作区内设置的应用程序虚拟磁盘的输出。

图C.28-GDICov迭代器的层次结构图

C.2.28.1 GdiCoVdIterator :: «F» gdiVdIteratorFirst()

函数调用

```
gdiVdIteratorFirst
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):GdiCoVdRT
```

函数描述

设置指向工作区内第一个虚拟磁盘的内部指针。

返回值

返回对第一个 GdiCoVdRT 的引用（如果存在），否则返回 NIL 引用。

C.2.28.2 GdiCoVdIterator :: «F» gdiVdIteratorNext()

函数调用

```
gdiVdIteratorNext
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        inout                   GdiCoVdRTrefLastElement
        /*inparameter
        A reference to a element returned by gdiVDIteratorFirst or
          gdiVDIteratorNext.*/
        ):GdiCoVdRT
```

函数描述

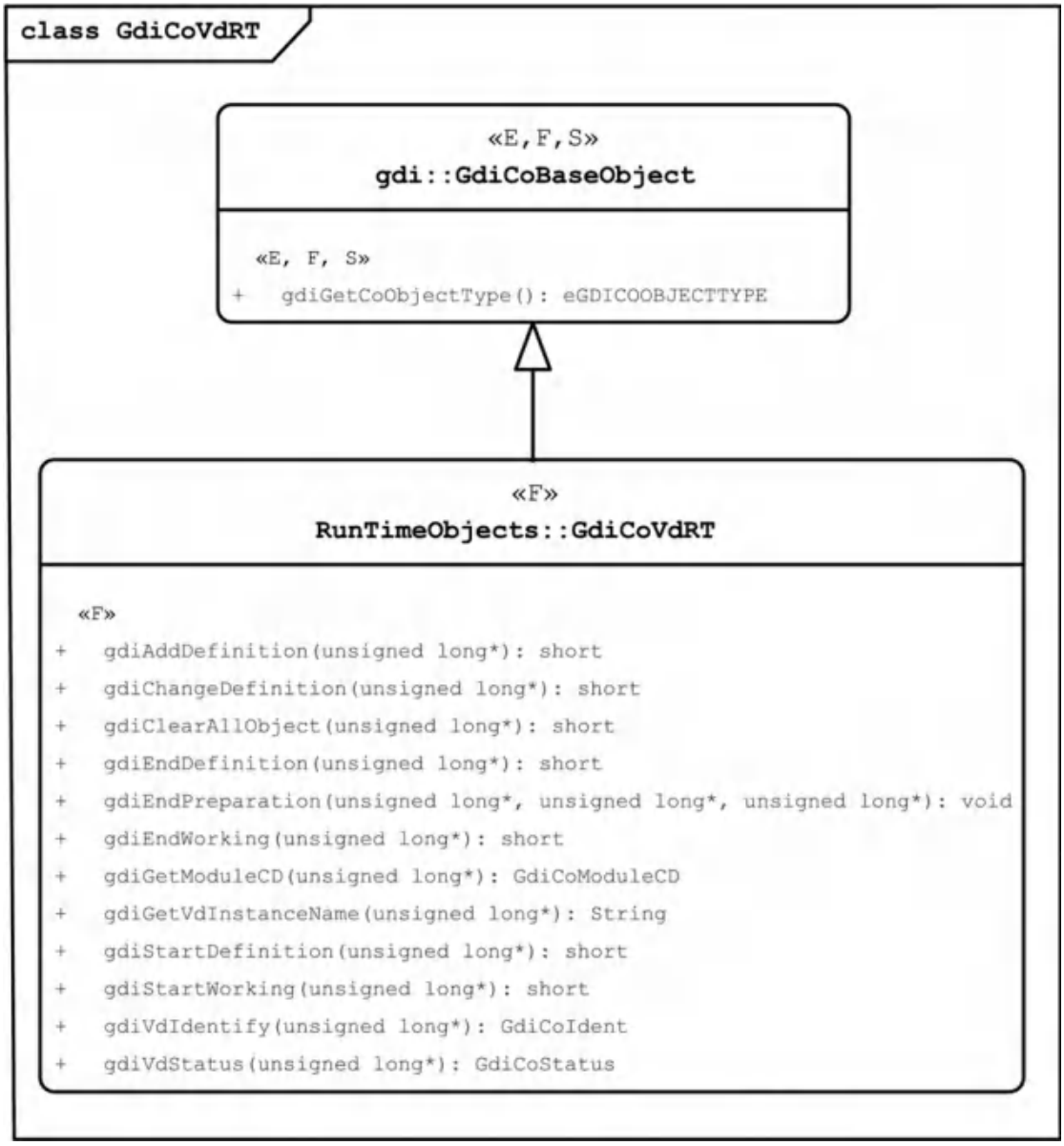
检测对工作区内当前虚拟磁盘（参数）的引用。设置指向下一个虚拟磁盘（如果存在）的内部指针并返回引用。

返回值

返回对第一个 GdiCoVdRT 的引用（如果存在），否则返回 NIL 引用。

C.2.29 «F» GdiCoVdRT

包含用于创建和控制应用程序虚拟磁盘状态的所有服务。



图C.29-GdiCoVdRT的层次结构图

C.2.29.1 GdiCoVdRT :: «F» gdiAddDefinition()

函数调用

```
gdiAddDefinition
          (
          inout                    unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):short
```

函数描述

将应用虚拟磁盘转换为修改状态。

返回值

如果成功，==0

否则，!=0，

C.2.29.2 GdiCoVdRT :: «F» gdiChangeDefinition()

函数调用

```
gdiChangeDefinition
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):short
```

函数描述

将应用虚拟磁盘转换为准备状态。

返回值

如果成功，==0

否则，!=0，

C.2.29.3 GdiCoVdRT :: «F» gdiClearAllObject()

函数调用

```
gdiClearAllObject
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):short
```

函数描述

将应用虚拟磁盘转换为初始状态。

返回值

如果成功，==0

否则，!=0，

C.2.29.4 GdiCoVdRT :: «F» gdiEndDefinition()

函数调用

```
gdiEndDefinition
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):short
```

函数描述

将应用虚拟磁盘转换为检查状态。

返回值：

如果成功，==0

否则，!=0，

C.2.29.5 GdiCoVdRT :: «F» gdiEndPreparation()

函数调用

```
gdiEndPreparation
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          inout                  unsignedlongulMsPollInterval
          /*inparameter
          containstheintervalofpollingtheStateoftheCheckProcess.*/
          inout                  unsignedlongulMsTimeOut
          /*inparameter
          containsthemaximumwaitingTimetilltheCheckprocessbefinished.
          TheglobaltimeoutsetbygdiCreateWorkspacewillnotbeaffected.*/
           ):void
```

函数描述

此方法在驱动程序中启动检查过程。虚拟磁盘中的给定配置由驱动程序验证。

如果检查过程正常并且执行了到工作的转换，则该方法成功。

如果虚拟磁盘中的给定配置不正确或存在硬件故障，则该方法将失败。

返回值

无。

C.2.29.6 GdiCoVdRT :: «F» gdiEndWorking()

函数调用

```
gdiEndWorking
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):short
```

函数描述

将应用虚拟磁盘转换为评估状态。

返回值

如果成功，==0

否则，!=0，

C.2.29.7 GdiCoVdRT :: «F» gdiGetModuleCD()

函数调用

```
gdiGetModuleCD (
          inout                  unsignedlongulAppHnd
          /*inparameter
          IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoModuleCD
```

函数描述

生成对虚拟磁盘的描述的引用，即 gdicomulecd 对象。

返回值：

返回对 gdicomulecd 的引用。

C.2.29.8 GdiCoVdRT :: «F» gdiGetVdInstanceName()

函数调用：

```
gdiGetVdInstanceName
```

```
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):String
```

函数描述：

以字符串形式返回虚拟磁盘实例名称。

返回值：

以字符串形式返回虚拟磁盘实例名称。

C.2.29.9 GdiCoVdRT :: «F» gdiStartDefinition()

函数调用

```
gdiStartDefinition
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):short
```

函数描述

将应用虚拟磁盘转换为评估状态。

返回值

如果成功，==0

否则，!=0，

C.2.29.10 GdiCoVdRT :: «F» gdiStartWorking()

函数调用

```
gdiStartWorking
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
            ):short
```

函数描述

将应用虚拟磁盘转换为工作状态。

返回值

如果成功，==0

否则，!=0，

C.2.29.11 GdiCoVdRT :: «F» gdiVdIdentify()

函数调用

```
gdiVdIdentify
            (
            inout                   unsignedlongulAppHnd
            /*inparameter
            IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
            gdiGetWorkspaceByName()*/
             ):GdiCoIdent
```

函数描述

影响与应用虚拟磁盘连接的设备的标识数据的检测。

返回值：

返回与 GDI 规范对应的 GDI 事件数据。

C.2.29.12 GdiCoVdRT :: «F» gdiVdStatus()

函数调用

```
gdiVdStatus
          (
          inout                  unsignedlongulAppHnd
          /*inparameter
          Identifieroftheapplicationreturnedbygdi CreateWorkspace(...)orby
          gdiGetWorkspaceByName()*/
          ):GdiCoStatus
```

函数描述

影响虚拟设备的状态数据和连接到应用虚拟磁盘的物理设备物理数据的检测。

返回值

返回与 GDI 规范对应的 GDISTATUS 数据。

附 录 D
（规范性）
编程参考指南—枚举和状态/标识信息

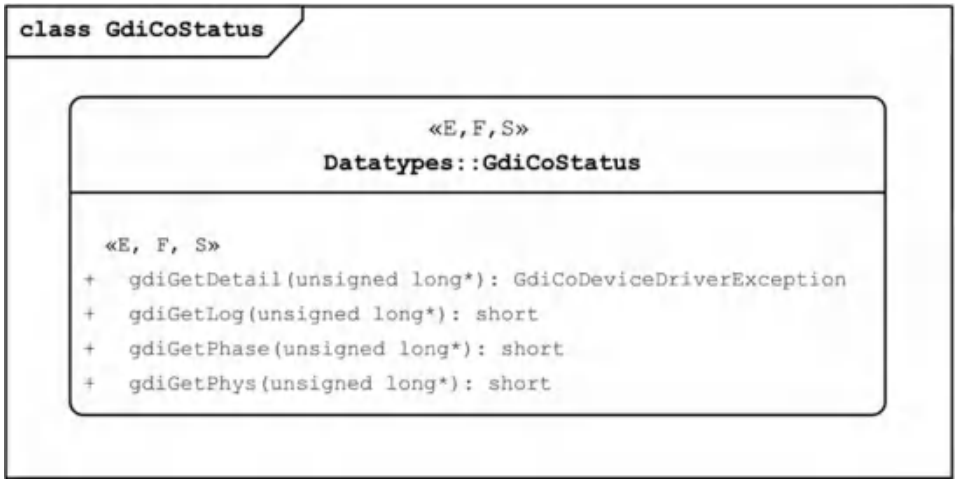
D.1 通用信息

协调人应使用本附件中定义的枚举和常量，向用户通报几种服务的详细信息。下面的详细描述由图 D.1 和 D.2 完成，以概述类及其方法。

D.2 枚举和状态/标识信息的详细说明

D.2.1 «S,E,F» GdiCoStatus

返回虚拟磁盘的状态。



图D.1-GdiCoStatus的层次图

D.2.1.1 GdiCoStatus :: «S,E,F» gdiGetDetail()

函数调用：

```
gdiGetDetail
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
         ):GdiCoDeviceDriverException
```

函数描述

获取错误类，该类描述虚拟磁盘的当前错误。

返回值

返回错误类。

D.2.1.2 GdiCoStatus :: «S,E,F» gdiGetLog()

函数调用

```
gdiGetLog
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):short
```

函数描述

获取日志值（它描述 API 函数的减少或不受限制的使用。）

返回值
返回日志值。

D.2.1.3 GdiCoStatus :: «S,E,F» gdiGetPhase()

函数调用

```
gdiGetPhase
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
         ):short
```

函数描述
获取相位值（它描述 VD 的当前转换状态。）
返回值
返回相位值。

D.2.1.4 GdiCoStatus :: «S,E,F» gdiGetPhys()

函数调用

```
gdiGetPhys
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):short
```

函数描述
获取物理值（它描述虚拟磁盘的减少或不受限制的使用。）
返回值
返回物理值。

D.2.2 «S,E,F» GdiCoIdent

向虚拟磁盘返回特定的详细信息。

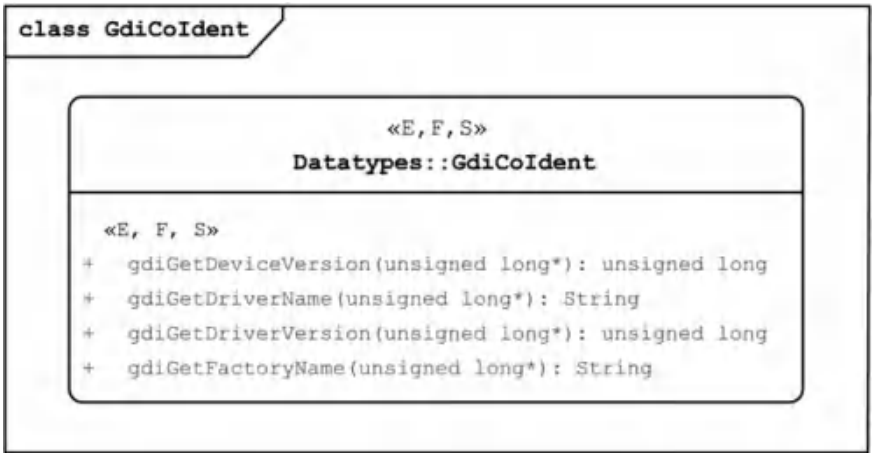


图2.GDICOD层次图

D.2.2.1 GdiCoIdent :: «S,E,F» gdiGetDeviceVersion()

函数调用

```
gdiGetDeviceVersion
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
```

```
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):unsignedlong
```

函数描述

获取设备版本号。

返回值

返回设备版本号。

D.2.2.2 GdiCoIdent :: «S,E,F» gdiGetDriverName()

函数调用

```
gdiGetDriverName
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):String
```

函数描述

获取驱动程序的名称。

返回值

返回驱动程序的名称。

D.2.2.3 GdiCoIdent :: «S,E,F» gdiGetDriverVersion()

函数调用

```
gdiGetDriverVersion
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):unsignedlong
```

函数描述：

获取驱动程序版本号。

返回值：

返回 GDI 驱动程序的版本号。

D.2.2.4 GdiCoIdent :: «S,E,F» gdiGetFactoryName()

函数调用：

```
gdiGetFactoryName
        (
        inout                   unsignedlongulAppHnd
        /*inparameter
        IdentifieroftheapplicationreturnedbygdiCreateWorkspace(...)orby
        gdiGetWorkspaceByName()*/
        ):String
```

函数描述：

获取设备制造商的名称。

返回值：

返回工厂的名称。

D.2.3 «enum» eACCEPTOCCURRED

如果自上次测试或 gdiWriteValue（见表 D.1）以来执行了“接受”操作，则 eacceptocurred 将返回信息。

表D.1各委员会成员

名称	注释	约束和标记
eACCEPT	接受发生	默认值：1
eACCEPT	不接受发生	默认值：0

D.2.4 «enum» eGDIACCESSRIGHTS

eGDIACCESSRIGHTS 定义了对通信对象的访问权限（见表 D.2）。

表D.2eGDIACCESSRIGHTS成员

名称	注释	约束和标记
eREADONLY		默认值：1
eREADWRITE		默认值：0

D.2.5 «enum» eGDIACCESSSTATE

eGDIACCESSSTATE 指定工作区实例的访问状态（见表 D.3）。

表D.3eGDIACCESSSTATE成员

名称	注释	约束和标记
eNOT_USED	工作区未被任何应用程序使用。工作区已准备好登录。	默认值：0
eUSED	工作区由任何其他应用程序使用。工作区尚未准备好登录。	默认值：1

D.2.6 «enum» eGDIBYTEORDER

eGDIBYTEORDER 定义数据流的对齐和字节顺序（见表 D.4）。

表D.4eGDIBYTEORDER成员

名称	注释	约束和标记
eLITTLE_ENDIAN	最低字节优先	默认值：0
eBIG_ENDIAN	最高字节优先	默认值：1

D.2.7 «enum» eGDICOIFKIND

eGDICOIFKIND 描述了应用程序预期的协调器接口类型（见表 D.5）。

0 = eSMART

1 = eEXTENDED

2 = eFULL

表D.5 eGDICOIFKIND成员

名称	注释	约束和标记
eSMART	请求智能功能	默认值：0
eEXTENDED	请求扩展功能	默认值：1
eFULL	需要全部能力	默认值：2

D.2.8 «enum» eGDICOMMOBJECTTYPE

类型 GDICOMOBJECTTYPE 指定相应设备功能的项的类型（见表 D. 6）。

表D.6- eGDICOMMOBJECTTYPE成员

名称	注释	约束和标记
ePARAM	device函数的成员是一个参数	默认值：2
eROATTRIBUTE	device函数的成员是只读属性	默认值：1
eRWATTRIBUTE	device函数的成员是一个属性	默认值：0

D.2.9 «enum» eGDICOOBJECTTYPE

eGDICOOBJECTTYPE 是一个检测 CD 引用类型的枚举（见表 D.7）。

表D.7 eGDICOOBJECTTYPE成员

名称	注释	约束和标记
eGDICOA_VERSION		默认值:
eGDICOARRAYCD		默认值:
eGDICOARRAYRT		默认值:
eGDICOATTRIBUTERT		默认值:
eGDICOBASEOBJECT		默认值:
eGDICOBASICSTREAM		默认值:
eGDICOBOOLEANCD		默认值:
eGDICOBOOLEANRT		默认值:
eGDICOCHARCD		默认值:
eGDICOCHARRT		默认值:
eGDICOCOMMOBJECTCD		默认值:
eGDICOCOMMOBJECTITERATOR		默认值:
eGDICOCOMMOBJECTRT		默认值:
eGDICOCOORDINATOREXCEPTION		默认值:
eGDICOCOORDINATORINTERFACEEXCEPTION		默认值:
eGDICODATAOBJECTRT		默认值:
eGDICODATATYPECD		默认值:
eGDICODESCRIPTIONFACTORY		默认值:
eGDICODESCRIPTIONOBJECTNAVIGATOR		默认值:
eGDICODEVICEDRIVEREXCEPTION		默认值:
eGDICODOUBLECD		默认值:

表D.7 （续）

名称	注释	约束和标记
eGDICODOUBLERT		默认值:
eGDICODRIVERCD		默认值:
eGDICODRIVEROBJECTITERATOR		默认值:
eGDICODRIVERRT		默认值:
eGDICOELEMENTITERATOR		默认值:
eGDICOENTRY		默认值:
eGDICOENUMCD		默认值:
eGDICOENUMRT		默认值:
eGDICOEXCEPTIONBYDRIVER		默认值:
eGDICOEXTENDEDOBJECTNAVIGATOR		默认值:
eGDICOFLOATCD		默认值:
eGDICOFLOATRT		默认值:
eGDICOFROMCOORDSTREAM		默认值:
eGDICOFROMCOORDSTREAMBUFFERED		默认值:
eGDICOFROMCOORDSTREAMUNBUFFERED		默认值:
eGDICOFUNCOBJECTITERATOR		默认值:
eGDICOFUNCOBJECTREFCD		默认值:
eGDICOFUNCOBJECTREFRT		默认值:
eGDICOFUNCOBJECTRT		默认值:
eGDICOINTERFACECD		默认值:
eGDICOINTERNALEXCEPTION		默认值:
eGDICOLONGCD		默认值:
eGDICOLONGRT		默认值:
eGDICOMANAGER		默认值:
eGDICOMANAGER_4_4_0		默认值:
eGDICOMODULECD		默认值:
eGDICOOBJECTADMINISTRATIONEXCEPTION		默认值:
eGDICOOBJECTCD		默认值:
eGDICOOCTETCD		默认值:
eGDICOOCTETRT		默认值:
eGDICOOPERATIONCD	GdiCoCD 对象的引用是 gdicOperationCD 引用	默认值:
eGDICOOPERATIONITERATOR		默认值:
eGDICOOPERATIONRT		默认值:
eGDICOPARAMETERIZATIONEXCEPTION		默认值:
eGDICOPARAMETERRT		默认值:

表D.7 （续）

名称	注释	约束和标记
eGDICOSEQUENCECD		默认值:
eGDICOSEQUENCERT		默认值:
eGDICOSHORTCD		默认值:
eGDICOSHORTRT		默认值:
eGDICOSTRUCTCD		默认值:
eGDICOSTRUCTRT		默认值:
eGDICOTOCOORDSTREAM		默认值:
eGDICOTOCOORDSTREAMBUFFERED		默认值:
eGDICOTOCOORDSTREAMUNBUFFERED		默认值:
eGDICOULONGCD		默认值:
eGDICOULONGRT		默认值:
eGDICOUNIONCD		默认值:
eGDICOUNIONRT		默认值:
eGDICOUNLIMITEDSEQUENCECD		默认值:
eGDICOUNLIMITEDSEQUENCERT		默认值:
eGDICOUSHORTCD		默认值:
eGDICOUSHORTRT		默认值:
eGDICOVDITERATOR		默认值:
eGDICOVDRT		默认值:
eGDICOWORKSPACE		默认值:
eGDICOWORKSPACEITERATOR		默认值:
eNOTDEFINED	未定义GdiCoCD引用的类型	默认值: 0
eGDIDRIVERRESULT		默认值:

D.2.10 «enum» eGDIDATATYPE

eGDIDATATYPE 指定所有发生的 DCD 数据类型定义（见表 D.8）。

表D.8 eGDIDATATYPE成员

名称	注释	约束和标记
eDT_ARRAY		默认值:20
eDT_BOOLEAN		默认值:2
eDT_CHAR		默认值:1
eDT_DOUBLE		默认值:9
eDT_ENUM		默认值:5
eDT_FLOAT		默认值:8
eDT_FUNCOBJECTREF		默认值:30
eDT_LONG		默认值:6
eDT_NO_DEFINITION		默认值:0
eDT_OCTET		默认值:25
eDT_SEQUENCE		默认值:21
eDT_SHORT		默认值:3
eDT_STRUCT		默认值:23
eDT_ULONG		默认值:7
eDT_UNION		默认值:24
eDT_UNLIMITEDSEQUENCE		默认值:22
eDT_USHORT		默认值:4

D.2.11 «enum» eGDIDIRECTION

eGDIDIRECTION 定义数据交换的方向==0 写入驱动程序！=0 从驱动器读取（见表 D.9）。

表D.9 eGDIDIRECTION成员

名称	注释	约束和标记
eFROMDRIVER		默认值:0
eTODRIVER		默认值:1

D.2.12 «enum» eGDIDRIVERRESULTTYPE

如果出现警告或信息（见表 D.10），eGDIDRIVERRESULTTYPE 是关于信息的枚举。

表D.10 eGDIDRIVERRESULTTYPE成员

名称	注释	约束和标记
eNO_ERROR		默认值:0
eIS_INFORMATION		默认值:1
eIS_WARNING		默认值:2

D.2.13 «enum» eGDIERRORCODES

此枚举定义所有非驱动程序特定的协调器内部错误代码（见表 D.11）。

表D.11 eGDIERRORCODES成员

名称	注释	约束和标记
eINT_CALLBACK_FUNCTIONALITY_ NOT_SUPPORTED	协调器不支持回调功能	默认值:11
eINT_INTERNAL_ERROR	发生内部错误。(操作系统错误，协调器内部错误)	默认值:1
eINT_INVALID_ACCESS	应用程序对此操作没有访问权限，或者工作区被其他应用程序使用，或者给定的应用程序句柄无效	默认值:10
eINT_REQUESTED_COORDINATOR_ CAPABILITY_NOT_SUPPORTED	协调器制造商不支持请求的协调器功能。	默认值:8
eINT_VERSION_NOT_SUPPORTED	不支持请求的协调器接口版本。(GDI 版本)	默认值:2
eOAD_INSTANCE_NAME_ NOT_ ALLOWED	给定的实例名称已存在或包含无效字符。	默认值:9
eOAD_OBJECT_ACCESS	请求的资源（数据源/接收器/描述）不存在。 如果应用程序是监视器应用程序，则不允许该操作。	默认值:3
eOAD_OPEN_SERVICE	无法删除 GDI 对象，因为服务已打开。	默认值:4
eOAD_OUT_OF_RANGE	索引超出范围 （序列/阵列访问）	默认值: 5
eOAD_TYPE_NOT_ALLOWED	类型不能在此上下文中使用（给定的 FO 不是动态 FO)。	默认值: 6
ePAR_INCORRECT_ PARAMETERIZATION	实例描述中出现错误或资源不可用。	默认值: 7
eOAD_DATAINUSE_OR_ INCONSISTENT	无法访问请求的数据，因为正在进行数据交换或数据不一致。	默认值: 12
eINT_PRACTICAL_DATA_OUT_OF_RANGE	应用程序或设备驱动程序错误地使用了实际数据： val 错误， 序列处理出错或未定义所用的联合分支。	默认值: 13
eINT_PID_FILE_MISMATCH	给定的 PID 文件与工作区中现有的 GDI 实例不匹配。	默认值: 14
eOAD_ELEMENT_ALREADY_EXIST	具有给定名称或 ID 的 CD 对象已存在。	默认值: 15

D.2.14 «enum» eGDIINFREPORTOCCURRED

如果自上次测试或 GDIRADVALUE（见表 D.12）以来已执行信息报告，则返回信息。

表D.12 eGDIINFREPORTOCCURRED成员

名称	注释	约束和标记
eINFREPORT		默认值:1
eNOINFREPORT	没有信息报告	默认值:0

D.2.15 «enum» eGDIRUNCONTROLSYNC

egdiruncontrollsync 指定 GDI 驱动程序和协调器之间的通信类型（请参见表 D.13）,包含同步通信和异步通信。

表D.13 eGDIRUNCONTROLSYNC成员

名称	注释	约束和标记
eASYNC	如果可能，设备驱动程序函数会立即返回。在设备驱动程序功能完全执行后，设备驱动程序将启动一个完整的回调。	默认值:1
eSYNC	设备驱动程序函数在设备驱动程序函数完成执行后返回。	默认值:0

D.2.16 «enum» eGDISTREAMALIGNMENT

eGDISTREAMALIGNMENT 返回结构化数据类型的对齐方式（见表 D.14）。

表D.14 eGDIRUNCONTROLSYNC成员

名称	注释	约束和标记
eEIGHT	校准是 64 位。（四字）	默认值:8
eEVEN	校准是 16 位。（文字）	默认值:2
eFOUR	校准是 32 位。（双字）	默认值:4
ePACKED	校准为 8 位（字节）	默认值:1
eSIXTEEN	校准是 128 位。（八字）	默认值:16

D.2.17 «enum» eORIGIN

eORIGIN 描述了流中的初始位置（见表 D.15）。

表D.15 eORIGIN成员

名称	注释	约束和标记
eBEGIN	初始位置是流中的第一个元素。	默认值:0
eCURRENT	初始位置是流中的当前元素。	默认值:1
eEND	初始位置是流中的最后一个元素。	默认值:2

D.2.18 «struct» TGdiCoEnumItem

TGdiCoEnumItem 是枚举元素（值和值标识符）的结构（见表 D.16）。

表D.16 TGdiCoEnumItem成员

名称	注释	约束和标记
nValue	数值	默认值:
eCURRENT	值标识符为字符串。	默认值:

D.2.19 «typedef» TBYTE

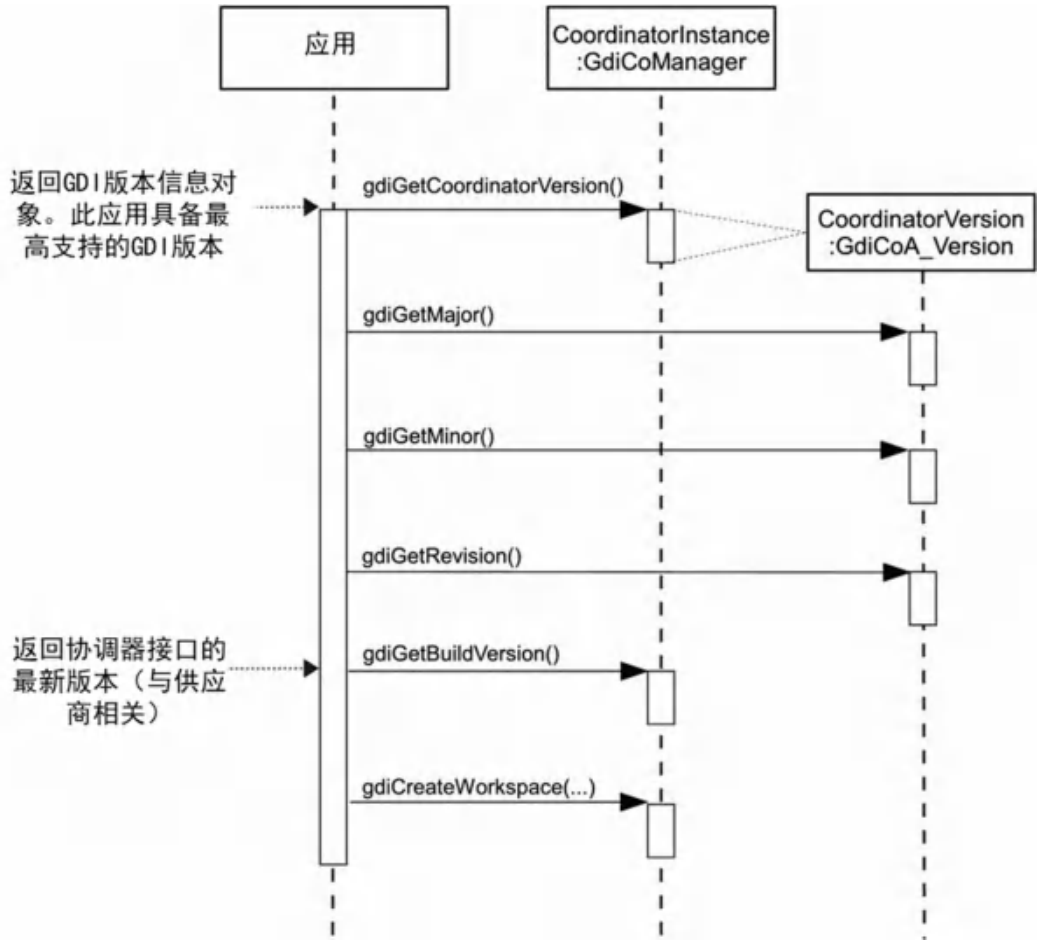
TBYTE 是一种标定流式传输方式的的非形式化通信数据类型。

D.2.20 «typedef» TDoubleSequence

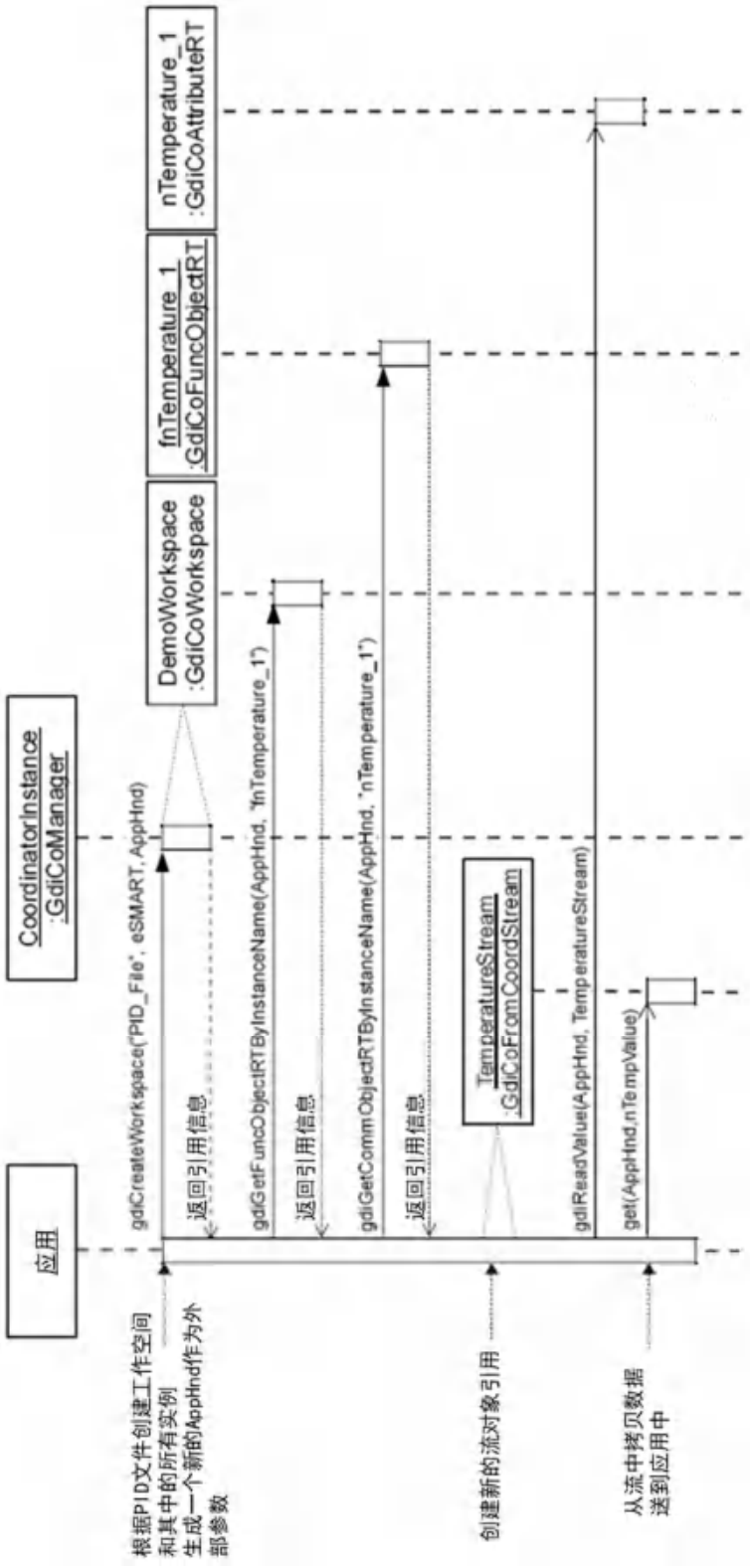
TDoubleSequence是一种标定流式处理方式的双值序列类型。

附 录 E
（资料性）
时序图

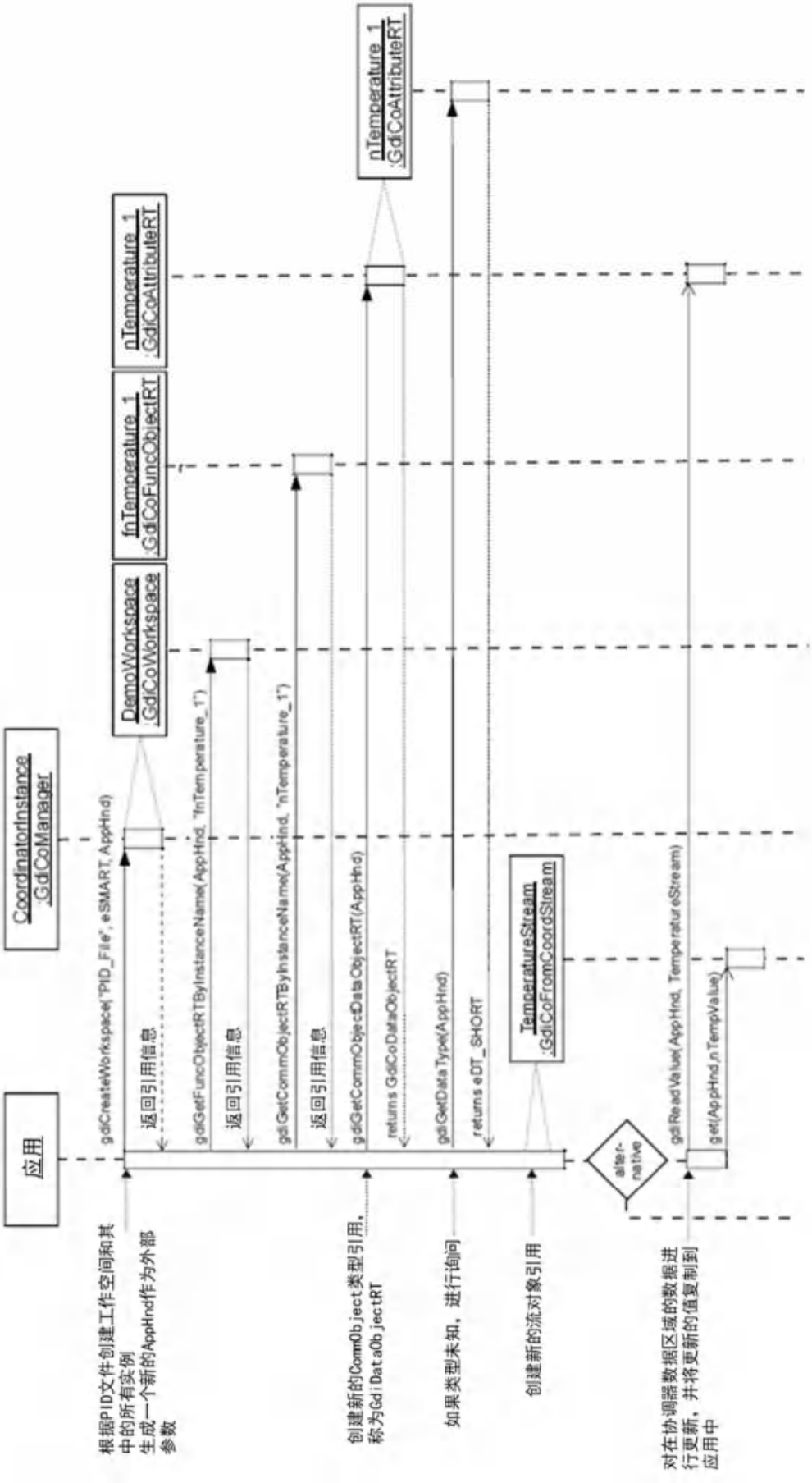
本附录包含时序图（图E.1至E.8），以便程序员创建协调器软件。时序图有助于理解具备附录A、B、C中所述接口的协调器的功能。



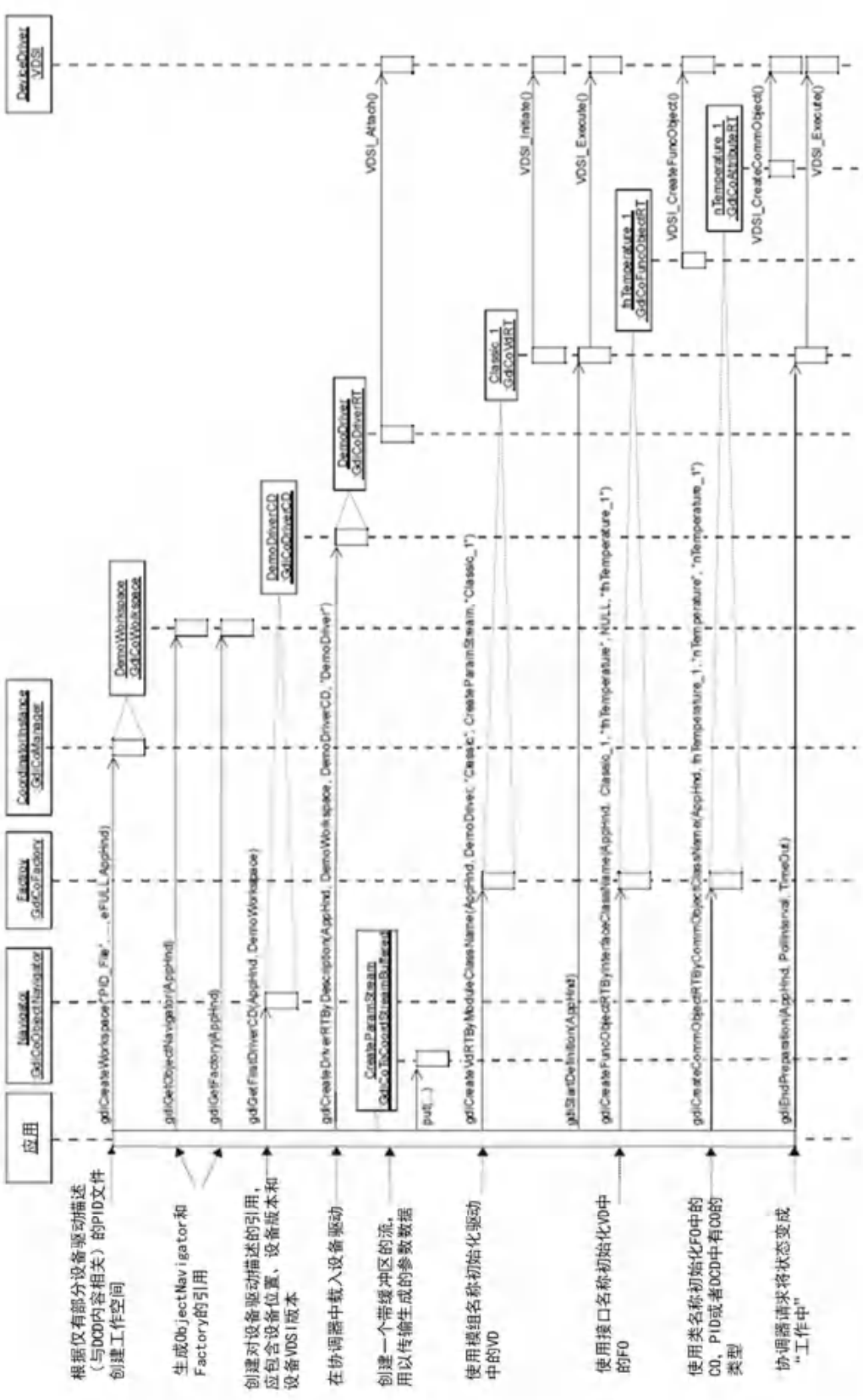
图E.1-主通道



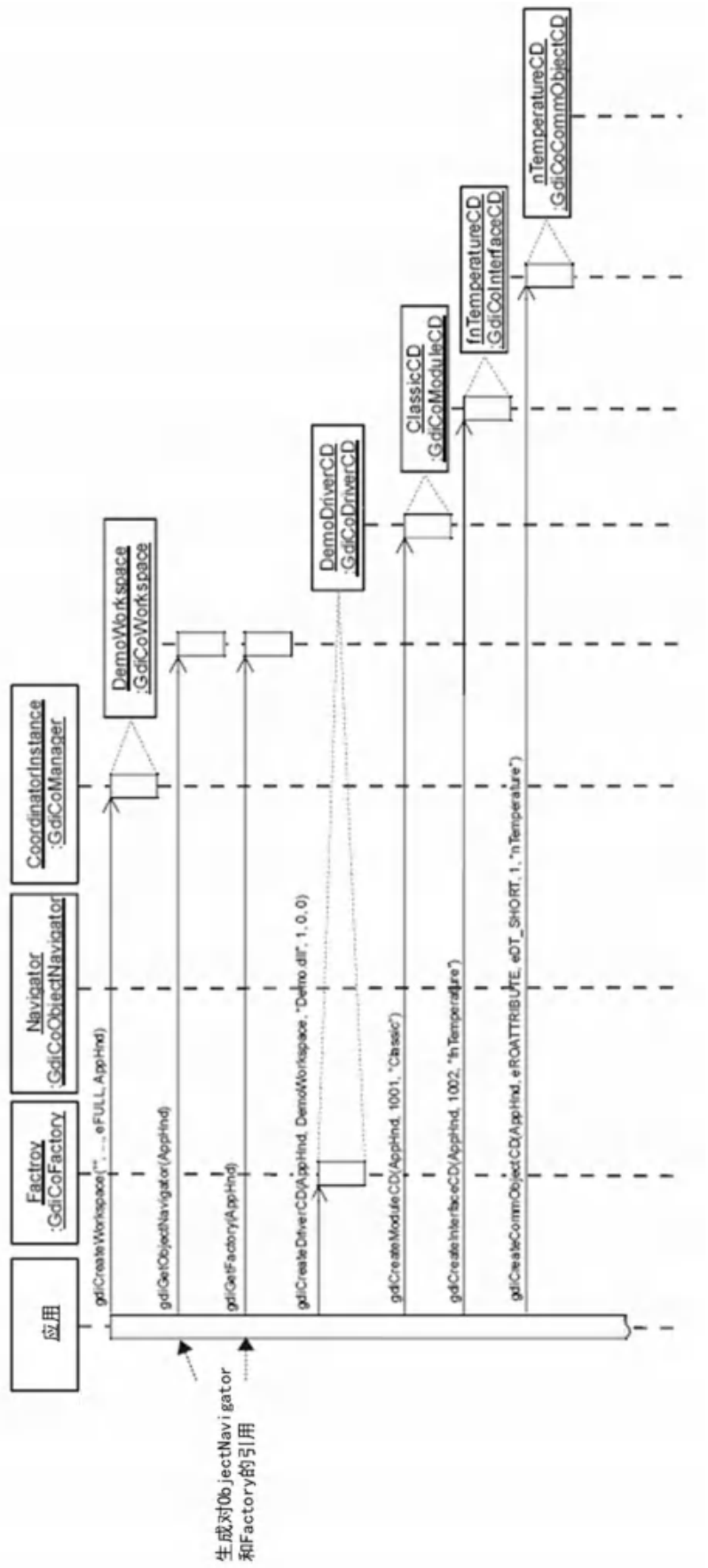
图E.2　通过使用智能访问接口使用参数化文件



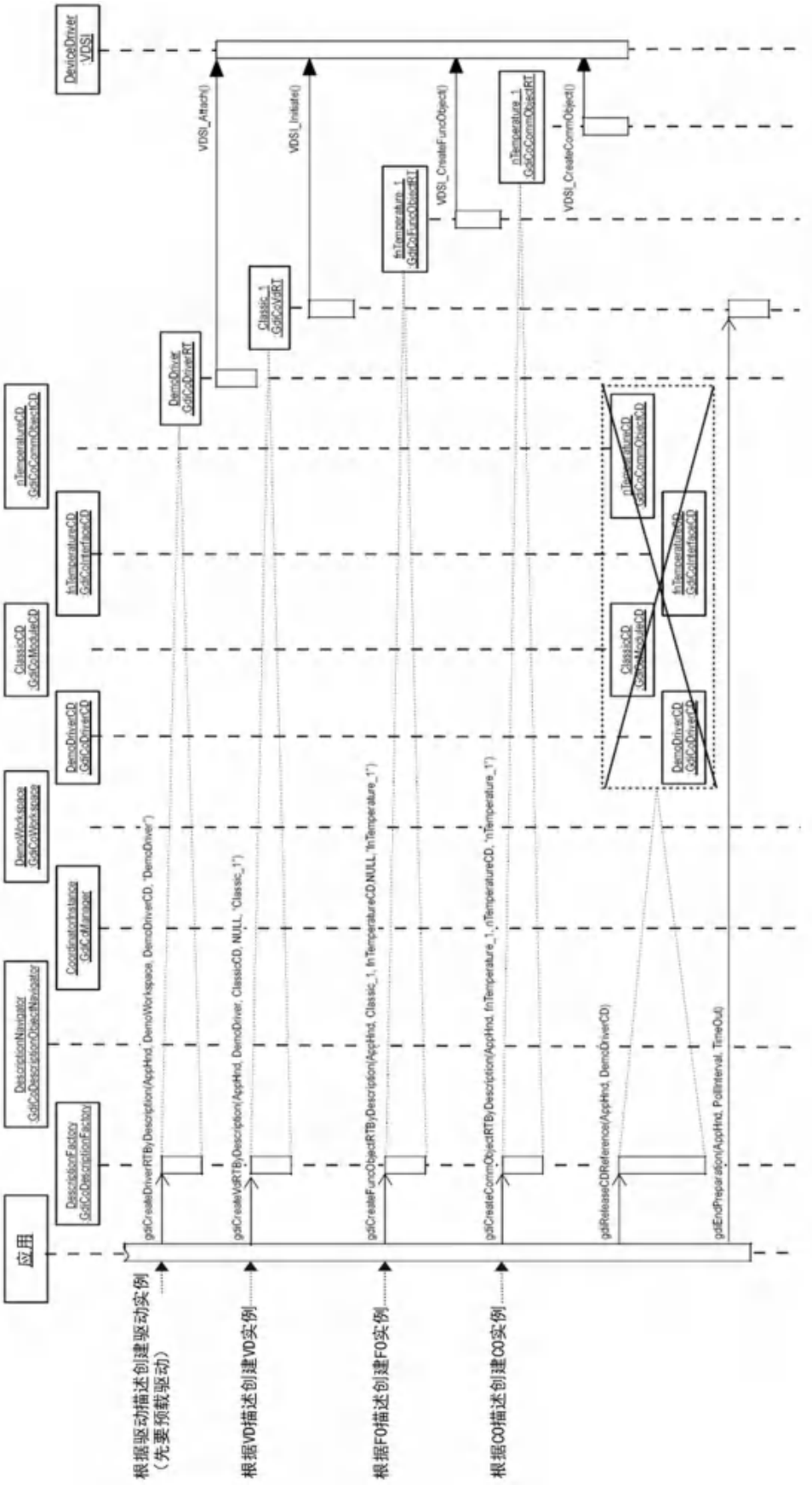
图E. 3　通过扩展访问接口使用参数化文件



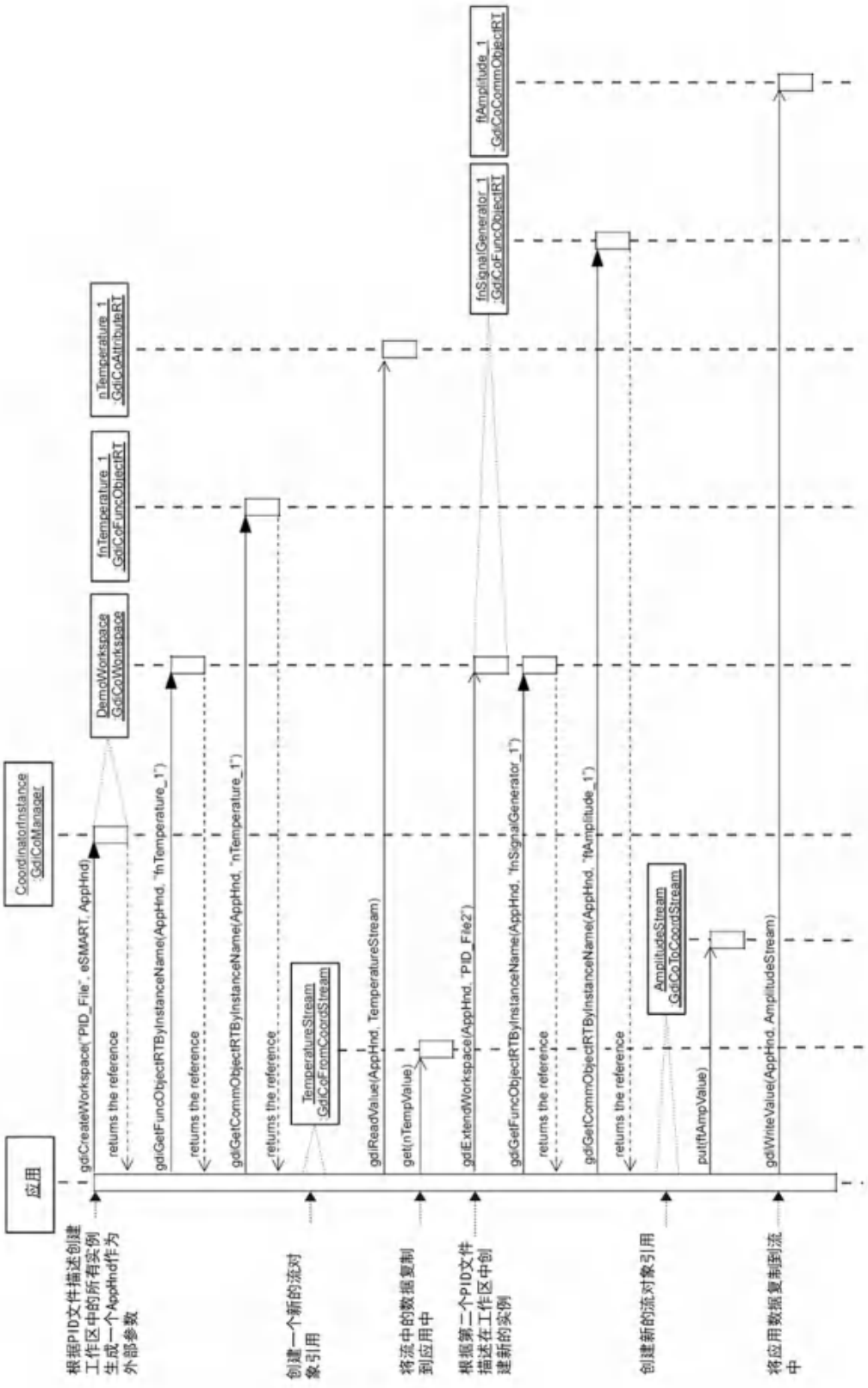
图E.4　使用完全访问接口使用参数化文件（没有实例的PID）



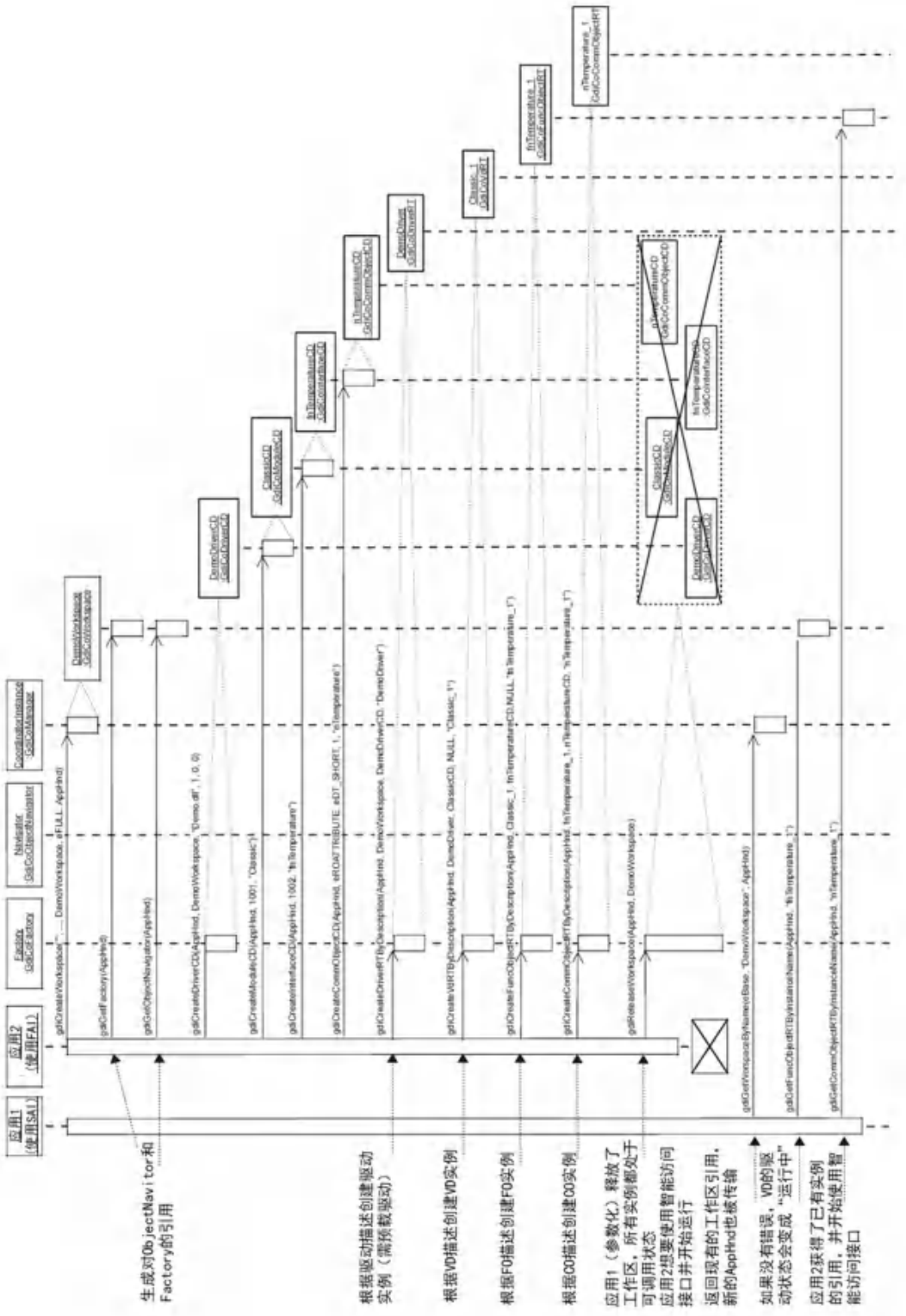
图E. 5　使用完全访问接口实现描述



图E.6 使用完全访问接口实现运行时对象


应用
CoordinatorInstance
:GdiCoManager
DemoWorkspace
:GdiCoWorkspace
fnTemperature_1
:GdiCoFuncObjectRT
nTemperature_1
:GdiCoAttributeRT
fnSignalGenerator_1
:GdiCoFuncObjectRT
ftAmplitude_1
:GdiCoCommObjectRT
TemperatureStream
:GdiCoFromCoordStream
AmplitudeStream
:GdiCoToCoordStream
gdiCreateWorkspace("PID_File", eSMART, AppHnd)
returns the reference
gdiGetFuncObjectRTByInstanceName(AppHnd, "fnTemperature_1")
returns the reference
gdiGetCommObjectRTByInstanceName(AppHnd, "nTemperature_1")
returns the reference
gdiReadValue(AppHnd, TemperatureStream)
get(nTempValue)
gdiExtendWorkspace(AppHnd, "PID_File2")
gdiGetFuncObjectRTByInstanceName(AppHnd, "fnSignalGenerator_1")
returns the reference
gdiGetCommObjectRTByInstanceName(AppHnd, "ftAmplitude_1")
returns the reference
put(ftAmpValue)
gdiWriteValue(AppHnd, AmplitudeStream)
根据PID文件描述创建工作区中的所有实例生成一个AppHnd作为外部参数
创建一个新的流对象引用
将流中的数据复制到应用中
根据第二个PID文件描述在工作区中创建新的实例
创建新的流对象引用
将应用数据复制到流中

图E. 7 扩展工作区


应用1
(使用SAL)
应用2
(使用FAL)
生成对ObjectNavitor和
Factory的引用
根据驱动描述创建驱动
实例（需预载驱动）
根据VD描述创建VD实例
根据FD描述创建FD实例
根据CD描述创建CD实例
应用1（参数化）释放了
工作区，所有实例都处于
可调用状态
应用2想要使用智能访问
接口并开始运行
返回现有的工作区引用，
新的AppHnd也被传输
如果没有错误，VD的驱
动状态会变成“运行中”
应用2获得了已有实例
的引用，并开始使用智
能访问接口

图E.8 转移工作区

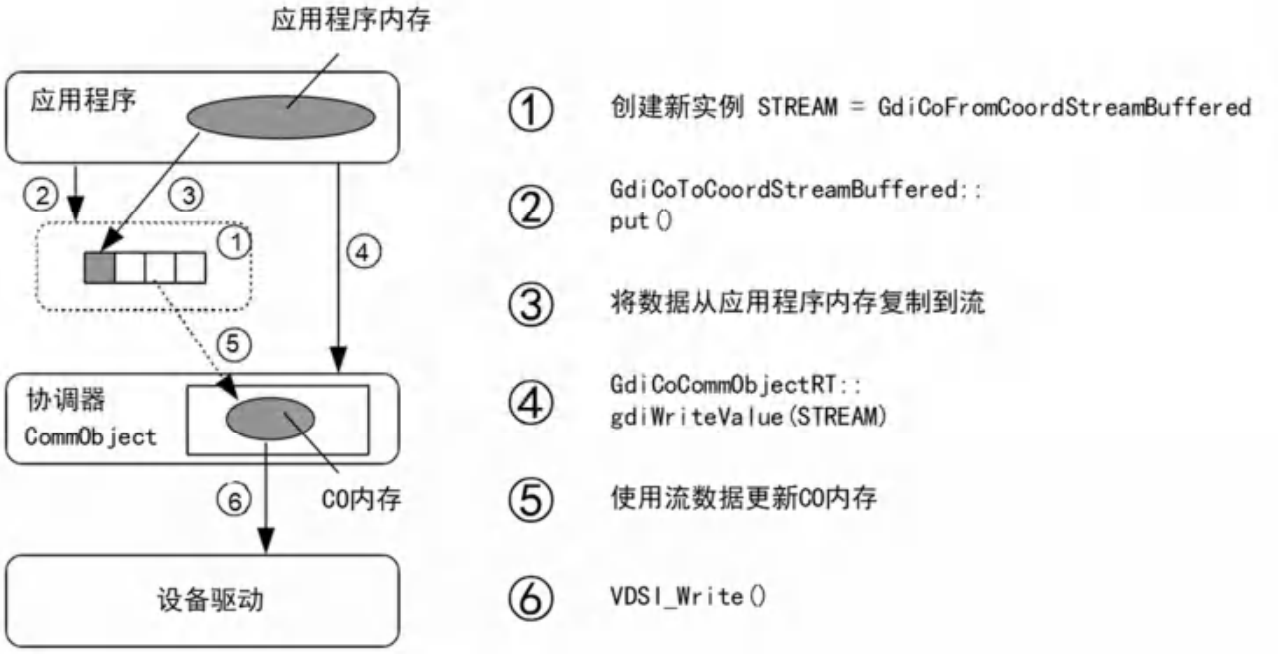
附 录 F
（资料性）
流

F.1 流过程

本小节包含一些简单的有少量文本扩展的图（见图 F.1-F.4）来可视化流的过程。这是为了方便程序员创建协调器软件。

图F.1 缓冲读取

图F.2 非缓冲读取

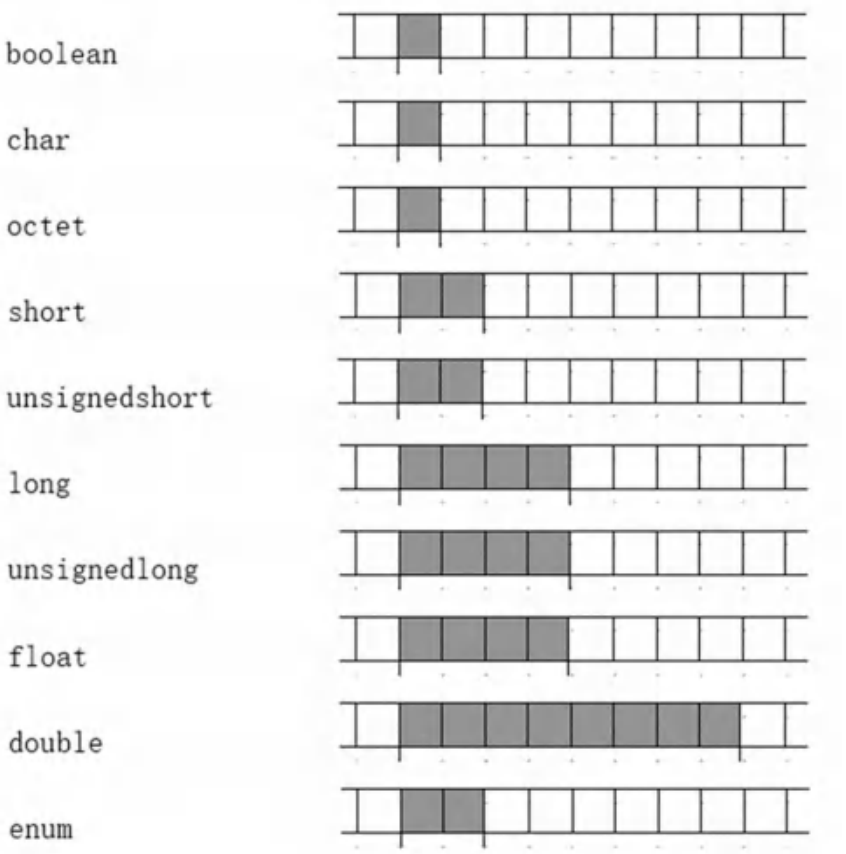


图F.3　缓冲写入

图F.4　非缓冲写入

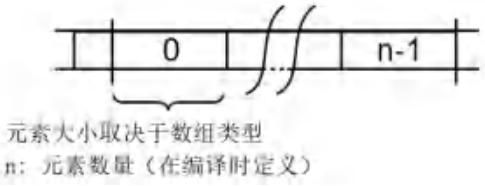
F.2　流映射

本小节包含一些简单的有少量文本扩展的图（见图 F.5-F.8）来可视化流的映射。这是为了方便程序员创建协调器软件。

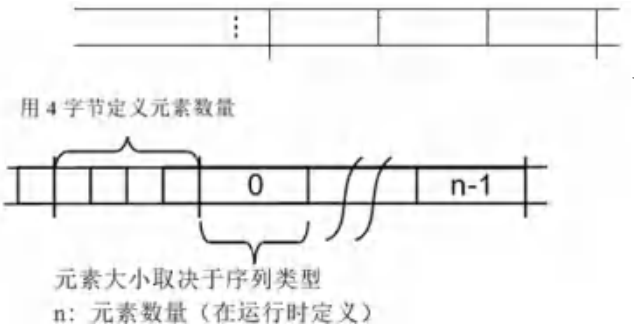


图F.5　流中简单类型的字节组织

注：流具有字节顺序和对齐方式，这在图中没有反映出来。

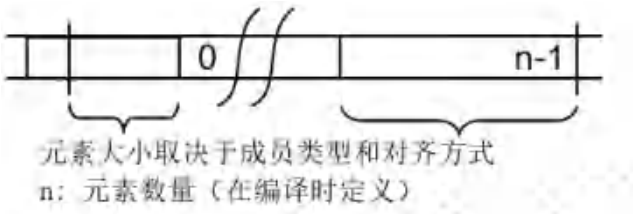


图F.6　数组



图F.7　序列

注：序列是一个数组，其中的元素数量在“运行时”定义。元素的数量被添加到数组数据流的前面。



图F.8　结构

附 录 G
（资料性）
数据对齐方式和字节顺序

G.1 对齐方式

G.1.1 简单类型

尽管可以基于任何对齐方式进行数据访问，但数据应该按照其自然限制(或整数的倍数)对齐，以避免性能损失。

枚举是常量整数，被处理为 32 位整数值(仅将枚举与图 F.5 中的流进行比较)。表 G.1 显示了类型定义和以下对齐值的推荐存储:

— Byte — 8 Bits
—Word— 16 Bits
—Doubleword— 32 Bits
—Quadword—64 Bits
—Octaword—128 Bits

表 G.1 —标量数据类型的对齐

标量类型	ASAM（自动化及测量系统标准协会）类型	型的内存大小(字节)	建议对齐方式	协调器 API 枚举定义的对齐方式 (eGDISTREAMALIGNMENT)
INT8	A_INT8	1	Byte	ePACKED
UINT8	A_UINT8	1	Byte	ePACKED
INT16	A_INT16	2	Word	eEVEN
UINT16	A_UINT16	2	Word	eEVEN
INT32	A_INT32	4	Doubleword	eFOUR
UINT32	A_UINT32	4	Doubleword	eFOUR
FP32 (单精度)	A_FLOAT32	4	Doubleword	eFOUR
FP64 (双精度)	A_FLOAT64	8	Quadword	eEIGHT

如数组、结构和联合等类型数据需要更加严格的对其方式，以确保内存的一致性和数据的整合使用。本附录的后续部分将对数组、结构和联合进行定义。

G.1.2 数组

数组包含一组有序的相邻数据对象。单个对象被称为元素。一个数组中的所有元素具有相同的大小和数据类型。

G.1.3 结构

结构包含一组数据对象。与数组元素相反，一个结构的数据对象可能具有不同的大小和类型。结构中的单个数据对象被称为成员。

G.1.4 联合

联合时一种由一组给定成员组成的对象，其中的成员可以根据需要随意替代。集合的成员可以是任何类型的。为该联合分配的内存与该联合的最大成员所需要的内存相对应，如果需要（如为了内存对齐），还会加上空内存。

G.1.5 对齐方式细节

聚合对齐规则：

— 数组的对齐方式与数组元素的对齐方式一致。

— 结构或联合的初始对齐方式是单个成员的最大对齐方式。结构或联合中的每个成员都必须按照上表中定义的对齐方式正确对齐。在此情况下，可能需要隐式填充，这取决于前面的成员。

— 结构的大小必须是其对齐方式的整数倍，因此在最后一个成员之后可能需要填充。因为结构和联合可以分成数组，所以结构或联合的每个数组元素都必须以先前定义的正确对齐方式开始和结束。

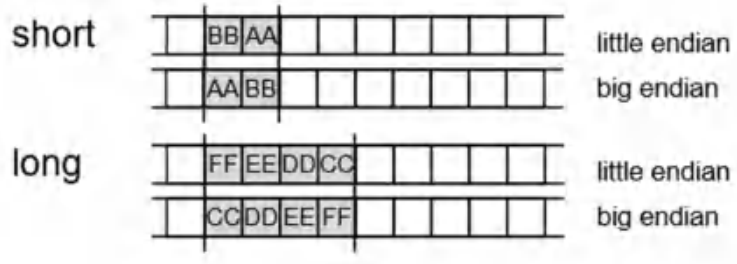
— 只要满足上述规则，数据的对齐方式可能比对齐所需的大小更大。

— 某个编译器可能会因为大小的原因调整压缩结构。例如，微软编译器允许通过选项/Zp(结构成员的对齐方式)调整压缩结构。在这里，空间是根据更短的限制对齐的。

G.2 字节顺序

APSI（应用程序服务接口）不依赖于特定的机器、操作系统或编程语言。为此，应用程序和协调器之间的数据传输须在数据表中定义和描述(见 I.2.2.4)。对于通过网络或软件接口以流的方式传输数据，需要预先设定传输的字节顺序。

传输或存储二进制数据的一种格式是小字节序。在这里，最低位字节(LSB)在第一个位置传输。与小字节序相反，在大字节序的情况下，最高位字节(MSB)在第一个位置传输。示例图 G.1 显示了短值 0xAABB 和长值 0xCCDDEEFF 字节在存储器中的不同位置。



图G.1　小尾数和大字节序的字节顺序

附 录 H
（资料性）
用 ISO 20242(APSI)和 ISO 13209(OTX) 系列标准测试应用程序

H.1 简介

协调器服务区别于以下接口：

— 智能访问接口，
— 扩展访问接口以及
— 完全访问接口.

APSI 到 OTX 的连接基于智能访问接口的使用。因此，使用基于 PID 的配置和配置文件来创建工作区。此文件中描述的实例是在创建工作空间期间设置和实例化的，并传递用于处理抽象数据源和接收的应用程序的引用。应用程序知道数据源和接收器的各个数据类型。

OTX [ISO/IEC 1309 -2]是一种标准的、平台独立的、测试者独立的描述格式，用于可执行测试序列、诊断过程或自动化描述。

基于xml的流语言提供了跨部门、工具和流程边界共享测试序列的机会。储存在这些流动中的专业知识不会丢失，即使多年后仍可使用。测试规格说明和测试执行之间是分离的。通过这种分离，OTX对于底层的开发过程步骤是非常灵活的。由语言元素支持，OTX序列的创建可以在不同的细节级别上完成。例如，测试序列可以首先指定，然后由相关的技术专家在稍后的过程步骤中细化，这样他们就可以处理一个特定的功能，从而产生一个可执行的描述。

图H.1 用于APSI的OTX扩展

OTX有一个与GDI一起使用的简单的标准化扩展机制(参见图H.1)。为此，APSI的OTX扩展定义了以下几个所需属性：

— 数据类型，
— 异常，

— 变量，
— 方法和
— 术语

这意味着所有的语言结构都可以生成一个流，其中涉及 APSI 通信对象和 APSI 操作。创建的序列作为 OTX XML 描述提供给多供应商交换。为了执行 OTX 程序，它们必须首先转换为可执行代码，用于在所使用的特定平台使用(例如，C++)。APSI 协调器提供了一个运行时库，其中对于每个 OTX 操作和术语都有一个到各自平台(例如，c++)的 APSI 服务的连接。

H.2 扩展描述

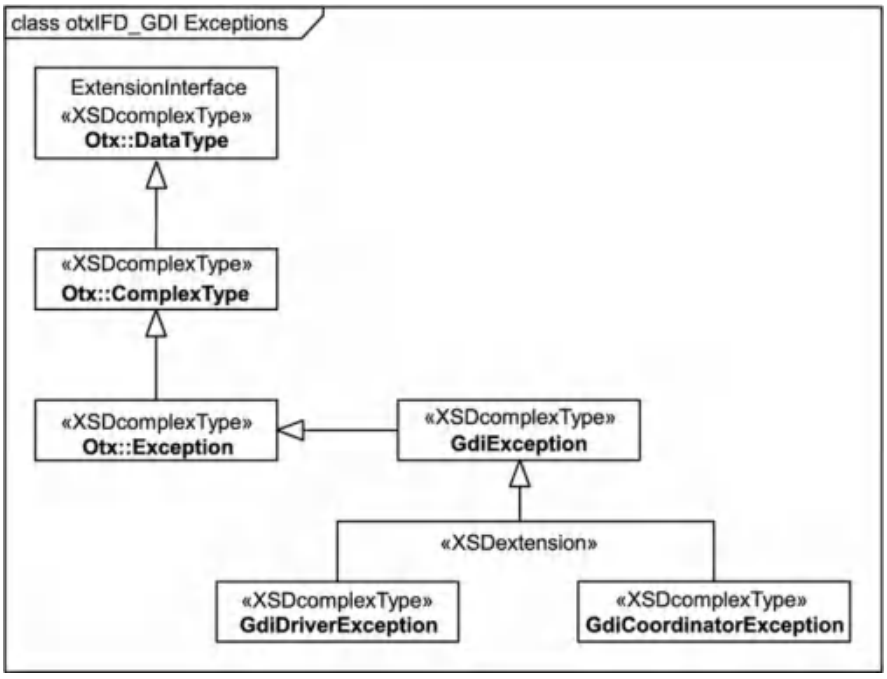
H.2.1 数据类型

H.2.1.1 简介

OTX otxIFD_GDI 扩展引入的所有数据类型都派生自 OTX:ComplexType，这意味着他们定义了 ISO 13209-2 中定义的复杂数据类型。

H.2.1.2 语法

otxIFD_GDI 扩展的数据类型声明的语法如图 H.2 所示。



图H.3 otxIFD_GDI 异常

H.2.2.3 语义

H.2.2.3.1 概要

因为所有 otxIFD_GDI 异常类型都是隐式异常，没有初始化部分，所以不能将它们声明为常量。

H.2.2.3.2 GdiCoordinatorException

如果在 GDI 子系统(协调器)管理期间发生错误(例如编程和访问错误)，则会抛出GdiCoordinatorException 异常。这个异常由 GDI 子系统(协调器)抛出。

H.2.2.3.3 GdiDriverException

如果在服务期间发生错误(ExecuteRead/ExecuteWrite/ExecuteOperation)，则会抛出GdiDriverException。此异常由 GDI 设备驱动生成，下列操作可能引发异常:

—createWorkspaceByPID,

—executeRead,

—executeWrite,

—executeOperation.

H.2.2.3.4 GdiException

GdiException是GDI扩展中所有异常的超类。GdiException应使用于本节其余部分中描述的更具体的异常类型不适用于当前的问题的情况。

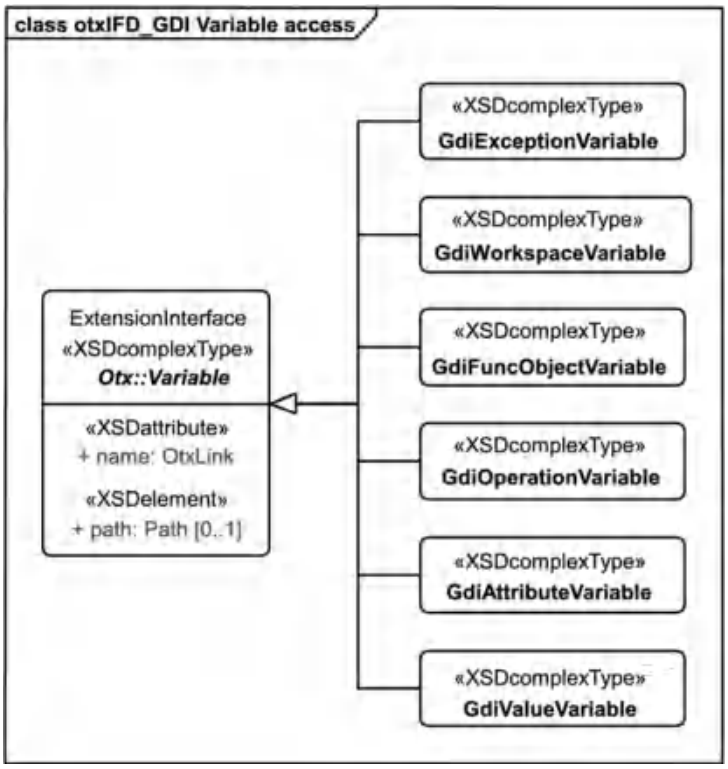
H.2.3 变量访问

H.2.3.1 简介

按照 ISO 13209-2 的规定，OTX 扩展应该为定义的每个数据类型定义一个变量访问类型(包括异常类型)。所有的变量访问类型都派生自 OTX Core Variable 扩展接口。下面指定了为otxIFD_GDI 扩展定义的所有变量访问类型。

H.2.3.2 语法

otxIFD_GDI 扩展的变量访问类型的语法如图 H.4 所示。



图H.4　otxIFD_GDI变量访问

H.2.3.3 语义

通用语义适用于所有变量访问类型。详情请参阅 ISO 13209-2。

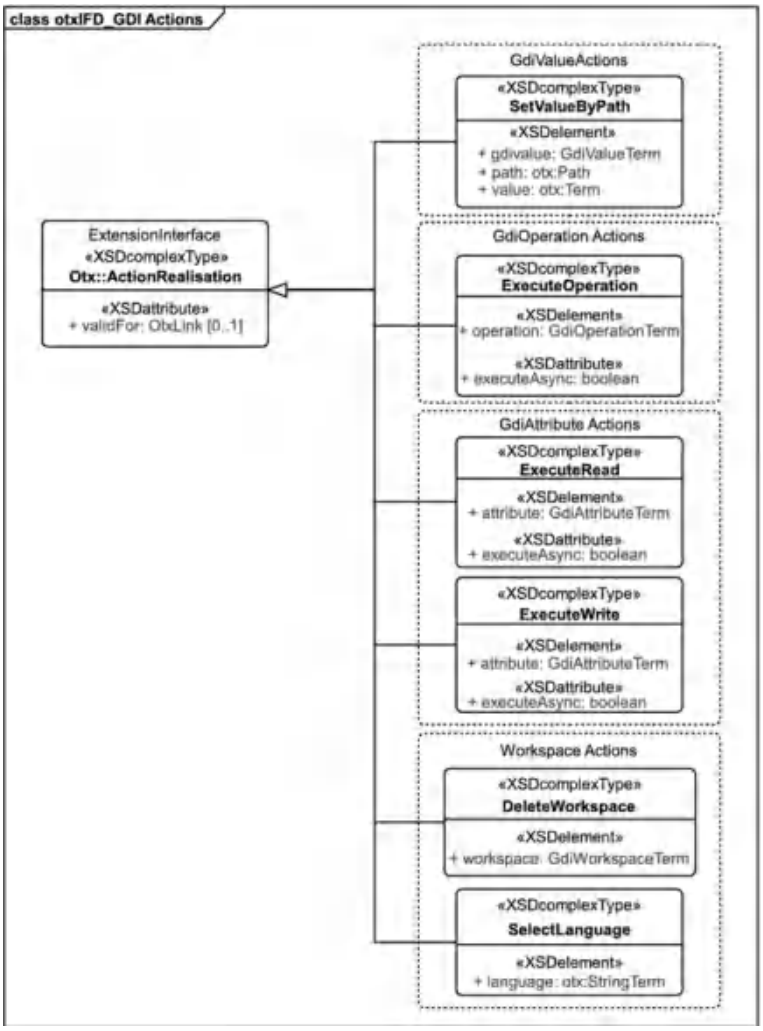
H.2.4 方法

H.2.4.1 简介

以下描述的所有元素扩展了 otx:ActionRealisation 扩展接口，如 ISO 13209-2 所定义的。

H.2.4.2 语法

图 H.5 显示了 otxIFD_GDI 扩展中所有方法的语法。



图H.5　otxIFD_GDI 方法

H.2.4.3 GdiAttribute 方法的语义

H.2.4.3.1 ExecuteRead

语义：

ExecuteRead 方法用于对属性执行读操作。OTX 序列中的 ExecuteRead 节点显示“运行时”系统，即 GDI 驱动程序中的一个读操作将被执行。如果驱动程序生成了 GDI 错误或 GDI 子系统中发生了错误，则会抛出一个异常。

参数：

— <attribute>:GdiAttributeTerm[1..1]
该参数指定 executeRead 方法将在其上执行的属性。

— executeAsync:boolean[1..1]
这个选项告诉通信后端使这个读操作是非阻塞的。这意味着如果 executeAsync 被设置为 true，OTX 执行流将立即转移到下一个操作，而不等待 ExecuteRead 操作的结果。因此，这个 ExecuteRead 操作定义的任何 out 值映射都会被忽略，因为 read 操作没有完成 ExecuteRead 节点的执行，静态映射来包含服务响应的 OTX 变量此时不能包含值。使用 executeAsync 能力总是需要 OTX 序列执行动态输出值解释。OTX 序列可以使用 AttributeCallBackEventSource 术语在读取操作完成时得到通知。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

GdiDriverException.

—设备驱动程序的所有 APSI 错误都会导致此异常。

InvalidReferenceException

—如果给定引用未知或无效，将引发此异常。

H.2.4.3.2 ExecuteWrite

语义：

ExecuteWrite 方法用于对属性执行读操作。OTX 序列中的 ExecuteWrite 节点显示运行时系统，即设备驱动程序中要执行的写操作。如果驱动程序生成 APSI 错误或 APSI 子系统中发生错误，将引发一个异常。

参数：

—<attribute> : GdiAttributeTerm [1..1]

该参数指定了 executeWrite 方法将在 Attribute 上执行。

—executeAsync : boolean [1..1]

这个选项告诉通信后端使这个写操作是非阻塞的。这意味着如果 executeAsync 被设置为 true，OTX 执行流将立即转移到下一个操作，而不等待 ExecuteWrite 方法的结果。当写操作完成时，OTX 序列可以使用 AttributeCallBackEventSource 术语来得到通知。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

GdiDriverException

—设备驱动程序的所有 APSI 错误都会导致此异常。

InvalidReferenceException

—如果给定引用未知或无效，将引发此异常。

H.2.4.4 GdiOperation Actions 的语义

H.2.4.4.1 ExecuteOperation

语义：

ExecuteOperation 方法用于执行操作。OTX 序列中的 ExecuteOperation 节点显示运行时系统，即必须执行 GDI 驱动程序中的操作。如果驱动程序生成 APSI 错误或 APSI 子系统中发生错误，将引发一个异常。

参数：

—<operation> : GdiOperationTerm [1..1]

该参数指定了 executeOperation 方法将在 Operation 上执行。

—executeAsync : boolean [1..1]

此选项告诉通信后端将此操作方法变为非阻塞操作。这意味着如果 executeAsync 被设置为 true，OTX 执行流将立即转移到下一个操作，而不等待 ExecuteOperation 操作的结果。因此,任何值的映射定义为这 ExecuteOperation 行动被忽略的操作行动不一定完成的执行 ExecuteOpertion 节点,静态的第一项变量映射到包含服务的响

应不能包含一个值。使用 executeAsync 能力总是需要 OTX 序列执行动态输出值解释。OTX 序列可以使用 OperationCallBackEventSource 术语在操作操作完成时得到通知。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

GdiDriverException

—设备驱动程序的所有 APSI 错误都会导致此异常。

InvalidReferenceException

—如果给定引用未知或无效，将引发此异常。

H.2.4.5 GdiValueActions 的语义

H.2.4.5.1 SetValueByPath

语义：

该方法为 GdiValue 元素设置一个特定的值。要设置的值以 OTX 简单类型或 OTX 列表的形式提供。SetValueByPath 方法接受 GdiValueTerm、要设置的值的路径和 otx:Term 的值。此操作的前提是参数名称在参数结构的一个层次结构级别中是唯一的。

参数：

—<gdivalue> : GdiValueTerm [1..1]

该参数指定要设置的 GdiValue 句柄。

该值句柄是以下术语一个返回值:

— GetValue,

— GetInValue,

— GetOutValue.

—<path> : otx:Path [1..1]

该路径元素指定所需值的路径。

—<value> : otx:Term [1..1]

这个参数指定了 GdiValue 的设定值。允许的输入类型有 OTX 简单类型，OTX 字节字段，列表和映射。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

TypeMismatchException

—如果给定路径与数据类型不匹配，将引发此异常

H.2.4.6 Workspace Actions 的语义

H.2.4.6.1 DeleteWorkspace

语义：

关闭并删除由给定 WorkspaceTerm 标识的 Workspace。

参数：

—<workspace> : GdiWorkspaceTerm [1..1]

该参数指定 DeleteWorkspace 方法应该在 Workspace 上执行。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会抛出此异常。

H.2.4.6.2 SelectLanguage

语义：

SelectLanguage 方法用于选择错误、警告和信息消息的语言。OTX 序列中的 SelectLanguage 节点显示运行时系统，该语言必须被更改。根据 APSI 规范的国家代码(语言编码[ISO 639-1] _CountryCode [ISO3166-1])选择语言。如果这个国家缩写 DIT 的 PID 文件不可用，将引发一个异常。

参数：

—<language> : otx:StringTerm [1..1]

该参数指定语言。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

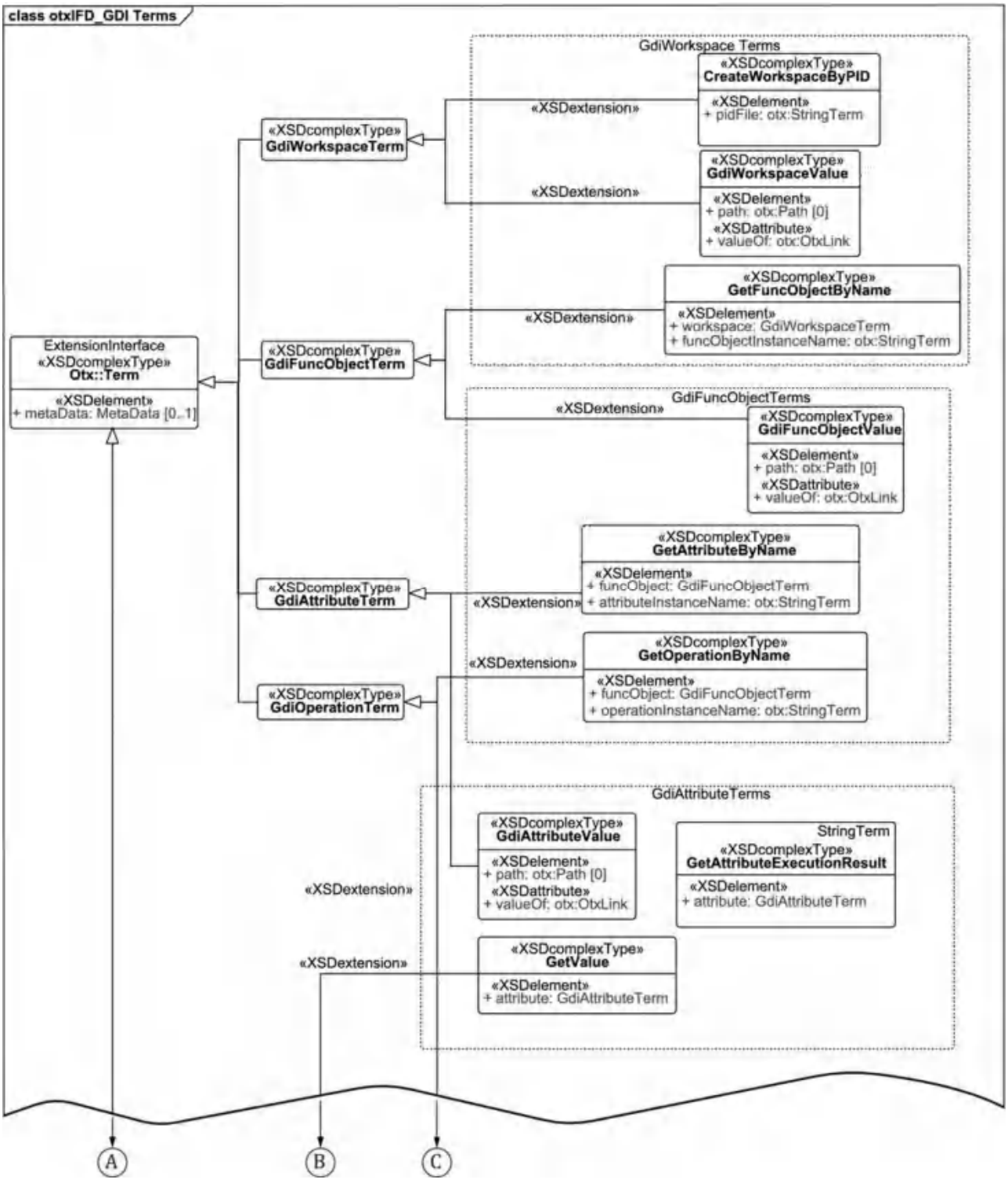
H.2.5 术语

H.2.5.1 简介

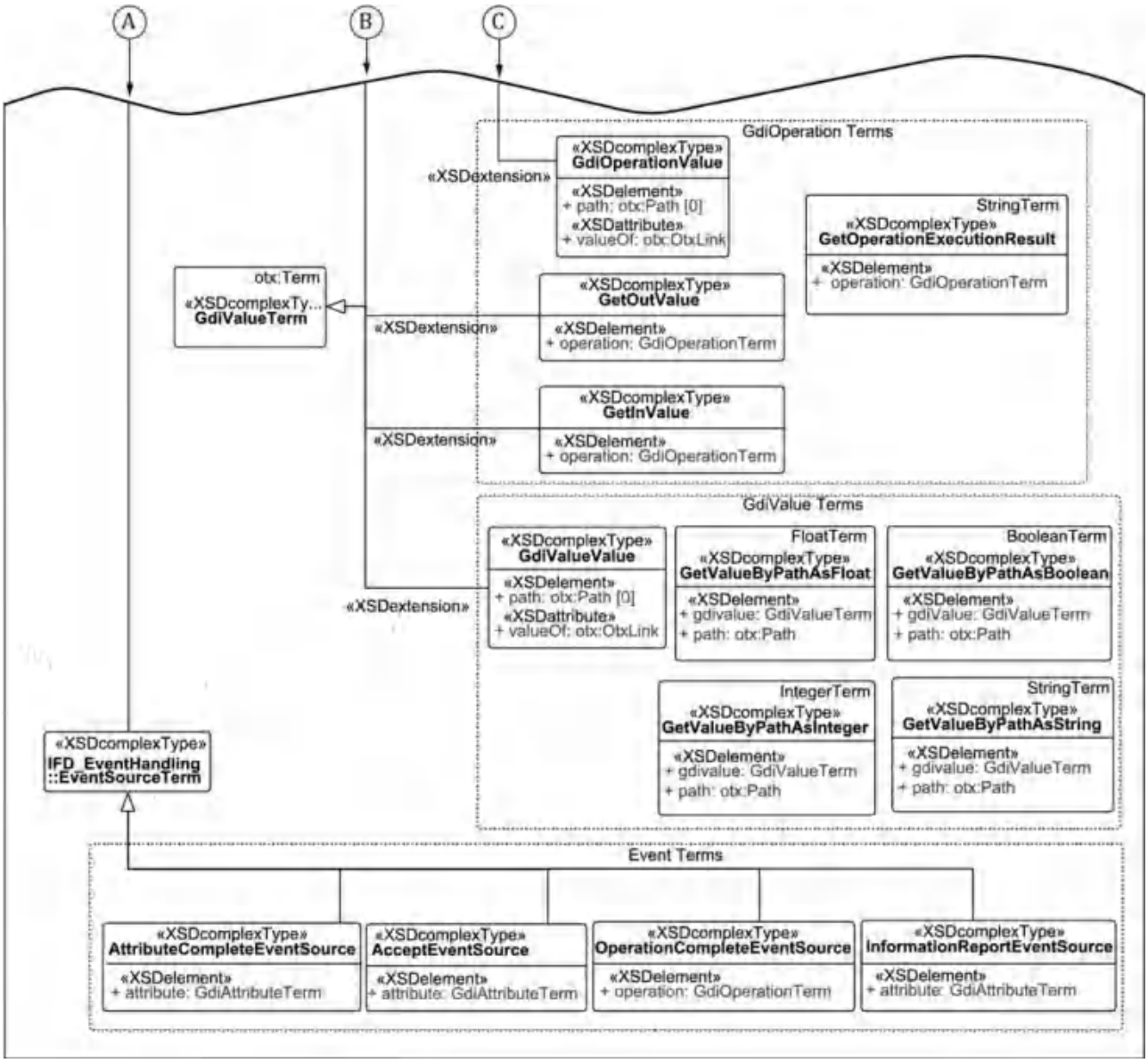
图 H.6 中所示的所有 otxIFD_GDI 术语都扩展了 ISO 13209-2 定义的 otx:Term 扩展接口。有关术语的特定超类的信息在下面的个别术语描述子句中提供。

H.2.5.2 语法

图 H.6 显示了 otxIFD_GDI 相关术语的语法。


class otxIFD_GDI Terms
GdiWorkspace Terms
«XSDcomplexType»
CreateWorkspaceByPID
«XSDelement»
+ pidFile: otx:StringTerm
«XSDextension»
«XSDcomplexType»
GdiWorkspaceTerm
«XSDcomplexType»
GdiWorkspaceValue
«XSDelement»
+ path: otx:Path [0]
«XSDattribute»
+ valueOf: otx:OtxLink
«XSDextension»
«XSDcomplexType»
GetFuncObjectByName
«XSDelement»
+ workspace: GdiWorkspaceTerm
+ funcObjectInstanceName: otx:StringTerm
«XSDextension»
ExtensionInterface
«XSDcomplexType»
Otx::Term
«XSDelement»
+ metaData: MetaData [0..1]
«XSDcomplexType»
GdiFuncObjectTerm
GdiFuncObjectTerms
«XSDextension»
«XSDcomplexType»
GdiFuncObjectValue
«XSDelement»
+ path: otx:Path [0]
«XSDattribute»
+ valueOf: otx:OtxLink
«XSDcomplexType»
GetAttributeByName
«XSDelement»
+ funcObject: GdiFuncObjectTerm
+ attributeInstanceName: otx:StringTerm
«XSDcomplexType»
GdiAttributeTerm
«XSDextension»
«XSDextension»
«XSDcomplexType»
GetOperationByName
«XSDelement»
+ funcObject: GdiFuncObjectTerm
+ operationInstanceName: otx:StringTerm
«XSDcomplexType»
GdiOperationTerm
GdiAttributeTerms
«XSDcomplexType»
GdiAttributeValue
«XSDelement»
+ path: otx:Path [0]
«XSDattribute»
+ valueOf: otx:OtxLink
StringTerm
«XSDcomplexType»
GetAttributeExecutionResult
«XSDelement»
+ attribute: GdiAttributeTerm
«XSDextension»
«XSDcomplexType»
GetValue
«XSDelement»
+ attribute: GdiAttributeTerm
«XSDextension»
A
B
C

a)



b)图H.6　otxIFD_GDI 术语

H.2.5.3 事件术语的语义

H.2.5.3.1 AcceptEventSource

语义：

GdiAcceptEventSource 术语接受作为事件源的 GdiAttribute 对象。这个术语使 OTX 序列能够使用 GdiAttribute 作为 OTX EventHandling 扩展上下文中的事件源。每当设备驱动程序请求这个 GdiAttribute 上的 accept 时，GdiAttribute 将触发一个事件(请与 GDI 规范中的 accept 处理进行比较)。设备驱动程序触发此事件后，OTX 序列必须将实际数据写入此属性。可以使用 GetAttributeFromEvent 术语计算该属性。AcceptEventSource 是一个事件:EventSource。

返回值类型：

EventSourceTerm

参数：

—<attribute> : GdiAttributeTerm [1..1]
表示应该连接到事件源的GdiAttribute。
—<metaData> : MetaData [0..1] (derived from Term)
这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

—NONE

H.2.5.3.2 AttributeCompleteEventSource

语义：

AttributeCompleteSource术语接受作为事件源的GdiAttribute对象。这个术语使OTX序列能够使用GdiAttribute作为OTX EventHandling扩展上下文中的事件源。每当这个GdiAttribute上的一个异步操作(executeAsync = true)完成时，一个GdiAttribute将触发这个事件。在这个事件被设备驱动程序触发后，OTX序列必须评估结果，通过GetResultFromAttribute术语来获得在调用属性动作期间可能会传递的错误或信息。此外，OTX序列必须根据这个GdiAttribute计算GdiValue，以防止ExecuteRead被调用来检索out值。可以通过使用GetAttributeFromEvent术语从事件中计算该属性。AttributeCompleteEventSource是一个事件:

返回值类型：

EventSourceTerm

参数：

—<attribute> : GdiAttributeTerm [1..1]
表示应该连接到事件源的GdiAttribute。
—<metaData> : MetaData [0..1] (derived from Term)
这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

—NONE

H.2.5.3.3 InformationReportEventSource

语义：

InformationReportEventSource术语接受要作为事件源的GdiAttribute对象。这个术语使OTX序列能够使用GdiAttribute作为OTX EventHandling扩展上下文中的事件源。每当设备驱动程序请求向OTX序列发送数据时，GdiAttribute都会触发此事件(请与ISO 20243 -3中的信息报告处理进行比较)。当设备驱动程序触发该事件后，OTX序列必须使用GdiValue从该属性读取实际数据。可以使用GetAttributeFromEvent术语计算该属性。AcceptEventSource是一个事件:EventSource。

返回值类型：

EventSourceTerm

参数：

—<attribute> : GdiAttributeTerm [1..1]
表示应该连接到事件源的GdiAttribute。
—<metaData> : MetaData [0..1] (derived from Term)
这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

—NONE

H.2.5.3.4 OperationCompleteEventSource

语义：

OperationCompleteEventSource 术语接受作为事件源创建的 GdiOperation 对象。这个术语允许 OTX 序列在 OTX EventHandling 扩展的上下文中使用 GdiOperation 作为事件源。每当这个 GdiOperation 上的异步操作(ExecuteOperation)(executeAsync = true)完成时，GdiOperation 将触发此事件。在这个事件被设备驱动程序触发后，OTX 序列必须评估结果，以获得可能在调用操作操作期间通过 GetResultFromOperation 项传递的错误或信息。此外，OTX 序列必须根据这个 GdiOperation 计算 GdiValue，以防出现 out 值。可以使用 GetOperationFromEvent 术语从事件计算操作。OperationCompleteEventSource 是一个事件:EventSource。

返回值类型：

EventSourceTerm

参数：

—<operation> : GdiOperationTerm [1..1]

表示应该连接到事件源的 GdiAttribute。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

—NONE

H.2.5.4 GdiAttributeTerms 的语义

H.2.5.4.1 GdiAttributeValue

语义：

GdiAttributeTerm 的值

返回值类型：

GdiAttributeTerm

参数：

—<path> : otx:Path [0..0]

该参数处理复杂结构的各个部分。更多信息请参考 ISO 13209 - 2。

—valueOf : otx:OtxLink [1..1]

包含存储值的变量、常量或参数的名称。更多信息请参考 ISO 13209-2。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。更多信息请参考 ISO 13209-2。

异常：

OutOfBoundsException

—当仅设置了<path>:<path>指向一个不存在的位置。

H.2.5.4.2 GetAttributeExecutionResult

语义：

这个术语根据给定的属性返回最后执行的操作(ExecuteRead/ExecuteWrite)的结果文本(警告/信息)。GetResultFromAttribute 是一个 StringTerm。

返回值类型：

StringTerm

参数：

—<attribute> : GdiAttributeTerm [1..1] 最后警告/信息将返回到GdiAttributeTerm。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如GDI扩展API处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

H.2.5.4.3 GetValue

语义：

这个术语根据给定的GdiAttributeTerm返回GdiValue。GetValue是一个GdiValueTerm。

返回值类型：

GdiValueTerm

参数：

—<attribute> : GdiAttributeTerm [1..1]

最后警告/信息将返回到GdiAttributeTerm。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如GDI扩展API处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

H.2.5.5 GdiFuncObjectTerms的语义

H.2.5.5.1 GdiFuncObjectValue

语义：

GdiFuncObjectTerm的值

返回值类型：

GdiFuncObjectTerm

参数：

—<path> : otx:Path [0..0]

该参数处理复杂结构的各个部分。更多信息请参考ISO 13209 - 2。

—valueOf : otx:OtxLink [1..1]

包含存储值的变量、常量或参数的名称。更多信息请参考ISO 13209-2。

—<metaData> :MetaData [0..1] (derivedfromTerm)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

OutOfBoundsException

— 当仅设置了<path>:<path>指向一个不存在的位置。

H.2.5.5.2 GetAttributeByName

语义：

这个术语返回一个 GdiAttribute 值，它根据给定的实例名称标识给定设备函数(函数对象)实例中的通信对象实例。在 PID 文件中设置实例名称。GetAttributeByName 是 GdiAttributeTerm。

返回值类型：

GdiAttributeTerm

参数：

—<funcObject> : GdiFuncObjectTerm [1..1]
专用属性实例所属的 GdiFuncObjectTerm。

—<attributeInstanceName> : otx:StringTerm [1..1]
此元素表示一个字符串标识属性实例。如果根据给定 GdiFuncObjectTerm 的属性实例不存在，则会引发异常。

—<metaData> : MetaData [0..1] (derived from Term)
这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

GdiCoordinatorException
—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException
— 如果给定引用未知或无效，将引发此异常。

H.2.5.5.3 GetOperationByName

语义：

这个术语返回一个 GdiOperation 值，它根据给定的实例名称标识给定设备函数(函数对象)实例中的通信对象实例。在 PID 文件中设置实例名称。GetOperationByName 是 GdiOperationTerm。

返回值类型：

GdiOperationTerm

参数：

—<funcObject> : GdiFuncObjectTerm [1..1]
专用属性实例所属的 GdiFuncObjectTerm。

—<operationInstanceName> : otx:StringTerm [1..1]
此元素表示一个字符串标识属性实例。如果根据给定 GdiFuncObjectTerm 的属性实例不存在，则会引发异常。

—<metaData> : MetaData [0..1] (derived from Term)
这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

GdiCoordinatorException
—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException
— 如果给定引用未知或无效，将引发此异常。

H.2.5.6 GdiOperation Terms 的语义

H.2.5.6.1 GdiOperationValue

语义：

GdiOperationTerm 的值

返回值类型：

GdiOperationTerm

参数：

—<path> : otx:Path [0..0]

该参数处理复杂结构的各个部分。更多信息请参考 ISO 13209 - 2。

—valueOf : otx:OtxLink [1..1]

包含存储值的变量、常量或参数的名称。更多信息请参考 ISO 13209-2。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

OutOfBoundsException

— 当仅设置了<path>:<path>指向一个不存在的位置。

H.2.5.6.2 GetInValue

语义：

这个术语根据给定的 GdiOperationTerm 返回 GdiValue 值。GetInValue 是一个 GdiValueTerm。

返回值类型：

GdiValueTerm

参数：

—<operation> : GdiOperationTerm [1..1]

In Value 所属的 GdiOperationTerm.

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 GDI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

H.2.5.6.3 GetOperationExecutionResult

语义：

这个术语根据给定的操作返回最后执行的方法（ExecuteOperation）的结果文本（警告/信息）。GetResultFromOperation 是一个 StringTerm。

返回值类型：

StringTerm

参数：

—<operation> : GdiOperationTerm [1..1]

警告/信息所属的 Operation Term。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

H.2.5.6.4 GetOutValue

语义：

这个术语根据给定的 GdiOperationTerm 返回 GdiValue 值。GetOutValue 是一个 GdiValueTerm。

返回值类型：

GdiValueTerm

参数：

—<operation> : GdiOperationTerm [1..1]

Out Value 所属的 GdiOperationTerm.

异常：

GdiCoordinatorException

— 如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

H.2.5.7 GdiValue Terms 的语义

H.2.5.7.1 GdiValueValue

语义：

GdiValueTerm 的值

返回值类型：

GdiValueTerm

参数：

—<path> : otx:Path [0..0]

该参数处理复杂结构的各个部分。更多信息请参考 ISO 13209 - 2。

—valueOf : otx:OtxLink [1..1]

包含存储值的变量、常量或参数的名称。更多信息请参考 ISO 13209-2。

异常：

OutOfBoundsException

— 当仅设置了<path>:<path>指向一个不存在的位置。

H.2.5.7.2 GetValueByPathAsBoolean

语义：

该方法为 GdiValue 元素设置一个特定的值。该值以 OTX 简单类型或 OTX 列表的形式提供。SetValueByPath 方法接受 GdiValueTerm、要设置的值的路径和 otx:Term 的值。此方法的前提是参数名称在参数结构的一个层次结构级别中是惟一的。

返回值类型：

BooleanTerm

参数：

—<gdiValue> : GdiValueTerm [1..1]
这个元素指定要设置的 GdiValue 句柄。
GdiValue 句柄是一个以下术语的返回值:
— GetValue,
— GetInValue,
— GetOutValue.

—<path> : otx:Path [1..1]
路径元素指定所需值的路径。

—<metaData> : MetaData [0..1] (derived from Term)
这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

GdiCoordinatorException
—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException
— 如果给定引用未知或无效，将引发此异常。

TypeMismatchException
— 如果给定路径与数据类型不匹配，将引发此异常。

H.2.5.7.3 GetValueByPathAsFloat

语义：

该方法为 GdiValue 元素设置一个特定的值。该值以 OTX 简单类型或 OTX 列表的形式提供。SetValueByPath 方法接受 GdiValueTerm、要设置的值的路径和 otx:Term 的值。此方法的前提是参数名称在参数结构的一个层次结构级别中是惟一的。

返回值类型：

FloatTerm

参数：

—<gdivalue> : GdiValueTerm [1..1]
这个元素指定要设置的 GdiValue 句柄。
GdiValue 句柄是一个以下术语的返回值:
— GetValue,
—GetInValue,
— GetOutValue.

—<path> : otx:Path [1..1]
路径元素指定所需值的路径。

—<metaData> : MetaData [0..1] (derived from Term)
这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

GdiCoordinatorException
—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException
— 如果给定引用未知或无效，将引发此异常。

TypeMismatchException
— 如果给定路径与数据类型不匹配，将引发此异常。

H.2.5.7.4 GetValueByPathAsInteger

语义：

该方法为GdiValue元素设置一个特定的值。该值以OTX简单类型或OTX列表的形式提供。SetValueByPath方法接受GdiValueTerm、要设置的值的路径和otx:Term的值。此方法的前提是参数名称在参数结构的一个层次结构级别中是惟一的。

返回值类型：

IntegerTerm

参数：

—<gdivalue> : GdiValueTerm [1..1]

这个元素指定要设置的GdiValue句柄。

GdiValue句柄是一个以下术语的返回值:

— GetValue,

— GetInValue,

— GetOutValue.

—<path> : otx:Path [1..1]

路径元素指定所需值的路径。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如APSI扩展API处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

TypeMismatchException

— 如果给定路径与数据类型不匹配，将引发此异常。

H.2.5.7.5 GetValueByPathAsString

语义：

该方法为GdiValue元素设置一个特定的值。该值以OTX简单类型或OTX列表的形式提供。SetValueByPath方法接受GdiValueTerm、要设置的值的路径和otx:Term的值。此方法的前提是参数名称在参数结构的一个层次结构级别中是惟一的。

返回值类型：

StringTerm

参数：

—<gdivalue> : GdiValueTerm [1..1]

这个元素指定要设置的GdiValue句柄。

GdiValue句柄是一个以下术语的返回值:

—GetValue,

— GetInValue,

— GetOutValue.

—<path> : otx:Path [1..1]

路径元素指定所需值的路径。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

TypeMismatchException

— 如果给定路径与数据类型不匹配，将引发此异常。

H.2.5.8 GdiWorkspace Terms 的语义

H.2.5.8.1 CreateWorkspaceByPID

语义：

这个术语创建了一个工作区实例，对应于给定的 PID 文件。工作区配置了所有设备功能实例，并准备好工作。所有必要的 GDI 驱动程序都已加载，设备函数的参数化和实例化已经完成。在调用这个术语之后，就可以通过使用术语 GetFuncObjectByName 来使用设备函数了。

返回值类型：

GdiWorkspaceTerm

参数：

—<pidFile> : otx:StringTerm [1..1]

这个元素表示一个标识 PID 文件的字符串(带有绝对路径或相对路径)，应使用用于创建工作空间。如果 PID 文件中描述的工作区已经存在，那么这个术语的执行不会执行其他操作。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如 APSI 扩展 API 处理错误，则会引发此异常。

GdiDriverException

— 设备驱动程序的所有 APSI 错误都会导致此异常。

H.2.5.8.2 GdiWorkspaceValue

语义：

GdiWorkspaceTerm 的值

返回值类型：

GdiWorkspaceTerm

参数：

—<path> : otx:Path [0..0]

该参数处理复杂结构的各个部分。更多信息请参考 ISO 13209 - 2。

—valueOf : otx:OtxLink [1..1]

包含存储值的变量、常量或参数的名称。更多信息请参考 ISO 13209-2。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

异常：

OutOfBoundsException

— 当仅设置了<path>:<path>指向一个不存在的位置。

H.2.5.8.3 GetFuncObjectByName

语义：

GetFuncObjectByName是一个FuncObjecTerm。这个术语返回一个GdiFuncObject值，它根据给定的实例名称标识一个设备函数实例。在PID文件中设置实例名称。

返回值类型：

GdiFuncObjectTerm

参数：

—<workspace> : GdiWorkspaceTerm [1..1]

专用设备函数实例所属的GdiWorkspaceTerm。

—<funcObjectInstanceName> : otx:StringTerm [1..1]

此元素表示标识设备函数实例的字符串。如果设备函数实例不存在，将引发异常。

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

异常：

GdiCoordinatorException

—如果发生了协调器特定的错误，例如APSI扩展API处理错误，则会引发此异常。

InvalidReferenceException

— 如果给定引用未知或无效，将引发此异常。

H.2.5.9 语义

H.2.5.9.1 GdiAttributeTerm

语义：

AttributeTerm 抽象类型属于otx:Term的一个类型。它是所有返回Attribute的具体术语的基础。它没有具体的成员。

参数：

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

H.2.5.9.2 GdiFuncObjectTerm

语义：

GdiFuncObjectTerm 抽象类型属于otx:Term的一个类型。它是所有返回GdiFuncObject的具体术语的基础。它没有具体的成员。

参数：

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示otx:Term元素的元数据。如需更多资料，请参阅ISO 13209-2。

H.2.5.9.3 GdiOperationTerm

语义：

GdiOperationTerm 抽象类型属于 otx:Term 的一个类型。它是所有返回 GdiOperation 的具体术语的基础。它没有具体的成员。

参数：

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

H.2.5.9.4 GdiValueTerm

语义：

GdiValueTerm 抽象类型属于 otx:Term 的一个类型。它是所有返回 GdiValue 的具体术语的基础。它没有具体的成员。

参数：

H.2.5.9.5 GdiWorkspaceTerm

语义：

GdiWorkspaceTerm 抽象类型属于 otx:Term 的一个类型。它是所有返回 GdiWorkspace 类型的具体术语的基础。它没有具体的成员。

参数：

—<metaData> : MetaData [0..1] (derived from Term)

这个值表示 otx:Term 元素的元数据。如需更多资料，请参阅 ISO 13209-2。

H.3 OTX 的用法

与 APSI 的定义类似，APSI OTX 扩展允许使用不同设备及其功能进行通用工作。然而，这要求程序创建者了解 ISO 20242 系列堆栈的具体信息。

OTX 提供了在过程中集成 OTX 程序的可能性。OTX 过程表示子程序/宏，它们可以与序列中的 OTX 操作和术语类似地使用。与 PID 文件和代码生成器一起，过程概念允许生成特定于设备的存根，它隐藏了 OTX 程序创建者对 GDI 堆栈的通用访问。因此，OTX 程序创建者可以直接使用设备功能，而不需要知道通用的 GDI 访问。

H.4 示例

图 E.2 所示的示例使用 APSI OTX 扩展映射到一个 OTX 序列。

表H.1 — APSI OTX扩展与协调器服务一起工作的C++程序列

```
voidwithSmartAccess()
{
    CGdiCoManager*pCppManager=NULL; CGdiCoWorkspace*pWorkspace=NULL; unsigned
    longulAppHnd; CGdiCoFuncObjectRT*pTemp1; CGdiCoCommObjectRT*pnTempRT;
    shortnTemperature=0;
    //get coordinatorentryobject pCppManager=getCPPCoordinator();
    //createworkspace
    pWorkspace=pCppManager->gdiCreateWorkspace("GDI_Demo_OTX_1.xml");
    //get instances
    pTemp1
    =pWorkspace->gdiGetFuncObjectRTByInstanceName(ulAppHnd,"fnTemperature_1");
    pnTempRT=pTemp1->gdiGetCommObjectRTByInstanceName(ulAppHnd,"nTemperature");
    //readdata
    CGdiCoDriverResult Result;
    CGdiCoFromCoordStreamBuffered*pFromCoordStream=new
    CGdiCoFromCoordStreamBuffered;
    pnTempRT->gdiReadValue(ulAppHnd,pFromCoordStream,&Result);
    //check ifa warningorinformation hasbeendelivered if
    (Result.gdiIsWarningOrInformation())
    {
        //evaluate information Result.gdiGetQual(); Result.gdiGetGrade();
        Result.gdiGetCode();
    }
    pFromCoordStream->get(&nTemperature,sizeof(short));
    //deleteworkspace
    pCppManager->gdiDeleteWorkspace(ulAppHnd,pWorkspace);
    deletepFromCoordStream;
}
```

各自的诊断、测试或自动化序列的创建取决于用户的能力。例如，可以通过图形化生成(参见图H.7)。

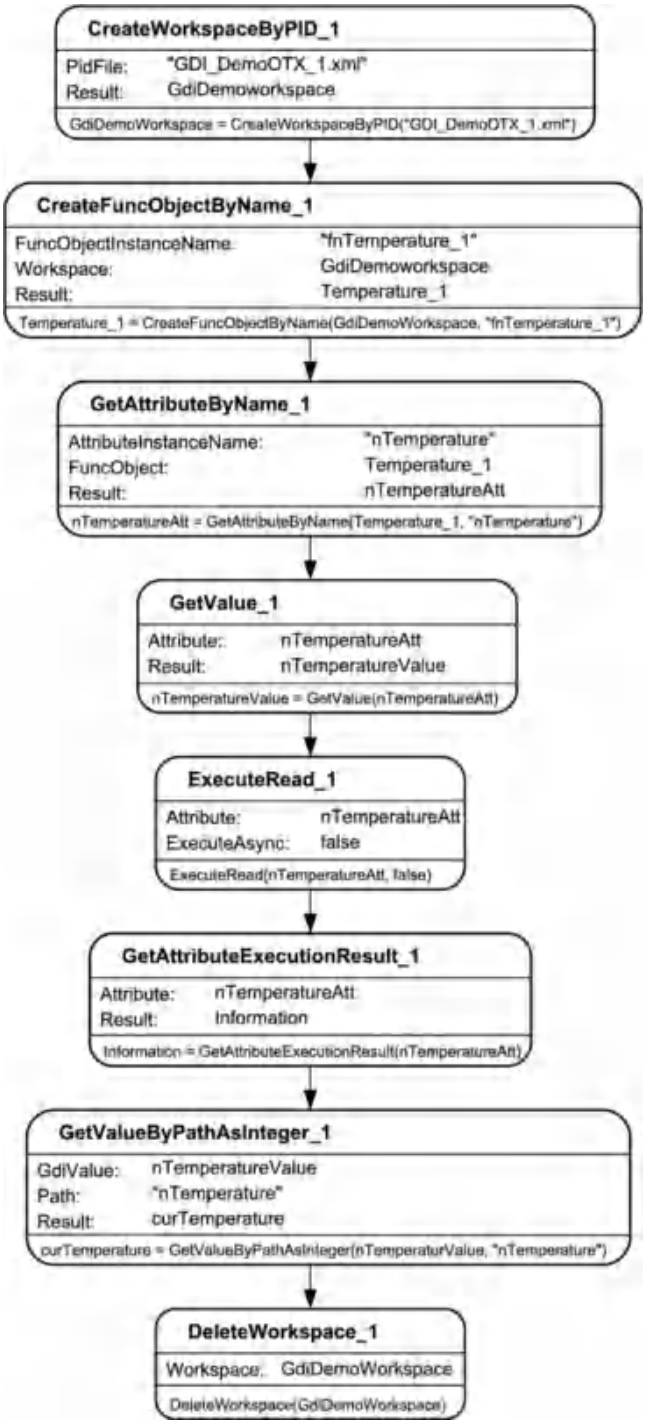

CreateWorkspaceByPID_1
PidFile: "GDI_DemoOTX_1.xml"
Result: GdiDemoworkspace
GdiDemoWorkspace = CreateWorkspaceByPID("GDI_DemoOTX_1.xml")
CreateFuncObjectByName_1
FuncObjectInstanceName: "fnTemperature_1"
Workspace: GdiDemoworkspace
Result: Temperature_1
Temperature_1 = CreateFuncObjectByName(GdiDemoWorkspace, "fnTemperature_1")
GetAttributeByName_1
AttributeInstanceName: "nTemperature"
FuncObject: Temperature_1
Result: nTemperatureAtt
nTemperatureAtt = GetAttributeByName(Temperature_1, "nTemperature")
GetValue_1
Attribute: nTemperatureAtt
Result: nTemperatureValue
nTemperatureValue = GetValue(nTemperatureAtt)
ExecuteRead_1
Attribute: nTemperatureAtt
ExecuteAsync: false
ExecuteRead(nTemperatureAtt, false)
GetAttributeExecutionResult_1
Attribute: nTemperatureAtt
Result: Information
Information = GetAttributeExecutionResult(nTemperatureAtt)
GetValueByPathAsInteger_1
GdiValue: nTemperatureValue
Path: "nTemperature"
Result: curTemperature
curTemperature = GetValueByPathAsInteger(nTemperaturValue, "nTemperature")
DeleteWorkspace_1
Workspace: GdiDemoWorkspace
DeleteWorkspace(GdiDemoWorkspace)

图 H.7 — OTX 序列

附 录 I
（规范性）
一致性测试方法、标准和报告

I.1 ISO 20242 系列标准的用法

I.1.1 实现具有一致性评估的接口

I.1.1.1 完全符合 ISO 20242 系列标准

实现 ISO 20242 系列标准接口的软件模块应提供已定义可使用的接口，并能与提供低层 ISO 20242 系列接口的其他软件模块集成(见图 I.1)。如果做到的话，则称软件模块“完全符合 ISO 20242 系列”。

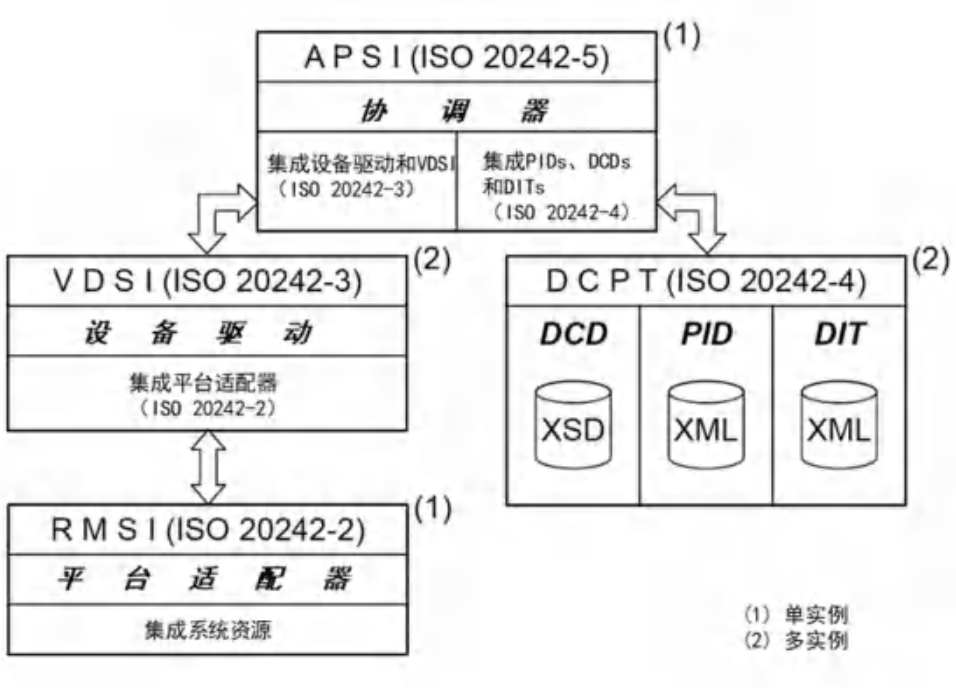


图 I.1 - ISO 20242 系列的全栈实现

I.1.1.2 限制符合 ISO 20242 系列标准

出于特定的目的或经济原因,可能会存在一些变体,它们提供 ISO 20242 系列中定义的接口，但不能将其他软件模块与 ISO 20242 系列接口集成。这些变体是“限制符合 ISO 20242 系列”。

I.1.1.3 部分符合 ISO 20242 系列标准

如果 ISO 20242 系列接口的所有定义和强制服务不都在一个软件模块中实现，那么就是“部分符合 ISO 20242 系列”。不受支持的服务应在数据表中列出(见 I.2)。

I.1.1.4 不符合 ISO 20242 系列标准

如果 ISO 20242 系列接口的任何服务都没有按照 ISO 20242 系列相应部分中描述的方式实现，则该实现不符合 ISO 20242 系列，且不属于 ISO 20242 系列。

如果在软件模块中实现超出 ISO 20242 系列定义的附加服务，则该附加服务不属于 ISO 20242 系列接口，也不与使用 ISO 20242 系列接口有任何依赖关系。否则，该实现不符合 ISO 20242 系列，且不属于 ISO 20242 系列。

如果存在如由设备和资源的物理条件引起的内部条件和对象之间的依赖关系，它们应该在实现内部处理，不能在接口上体现。否则，该实现不符合 ISO 20242 系列，且不属于 ISO 20242 系列。

I.2 ISO 20242 系列标准实施的数据表

I.2.1 简介

如果提供了 ISO 20242 系列接口的实现(软件模块)，应提供数据表，说明实现和符合性评估的技术细节。

I.2.2 通用使用说明

I.2.2.1 基本的一致性评估

应指出，下列符合标准(如 I.1.1 定义)中哪一项得到满足:

—完全符合 ISO 20242 系列标准

— 限制符合 ISO 20242 系列标准

— 部分符合 ISO 20242 系列标准

如果实现仅部分符合 ISO 20242 系列，应提供不支持服务的清单。

I.2.2.2 平台

开发和测试软件模块的操作系统及其版本。

I.2.2.3 版本

实现的版本应该描述，因为它在对应的 ISO 20242 系列接口上是可读的。

I.2.2.4 实现

对编程语言和特定的接口参数（如数据对齐，字节顺序和函数调用约定）进行描述。对软件模块的用户运行应用程序必需的任何参数进行描述。

I.2.2.5 通信事件

如果使用 Accept 和 InformationReport 进行未经请求的数据传输，应当进行描述。

I.2.2.6 异步通信

如果可能异步通信，应该进行描述。

I.2.2.7 多语言支持

如果使用 DIT 实现了多种语言支持，以及实现了哪些语言，应该加以说明。

I.2.2.8 局限和限制

说明在使用软件模块时存在哪些局限和限制。例如，使用该软件模块的应用程序的最大的数量，或可以集成到该软件模块中的其他 ISO 20242 系列软件模块的最大的数量。

I.2.3 协调器使用说明

I.2.3.1 其它访问接口

除强制智能访问接口(SAI)之外，注意是否有可选扩展访问接口(EAI)和/或可选完全访问接口(FAI)的实现。

I.2.3.2 扩大工作区

注意是否可以通过应用程序扩展工作区。详见 5.5.3。

I.2.3.3 流

应注意是否可以使用缓冲流。详见 5.5.7。

I.2.4 设备驱动使用说明

除 I.2.2 的一般使用说明外，不需要任何其它使用说明。

I.2.5 平台驱动器声明

如 ISO 20242-2 中定义的，资源管理服务接口(RMSI)提供了通用的输入/输出服务来与外围设备通信。将软件模块与扩展的服务接口集成以处理外围通信是可能的。
应该注意平台适配器支持哪些类型的外设接口，每种类型都有一个参数列表。

I.3 测试 ISO 20242 系列接口实现

I.3.1 测试文档

ISO 20242 系列接口实现的测试应根据交付的数据表(见 I.2)形成文件。测试环境应形成文件(见 I.3.2)。

I.3.2 测试实现

为了测试软件模块，必须有一个可以要求软件模块的所有功能的环境(见图 I.2)。

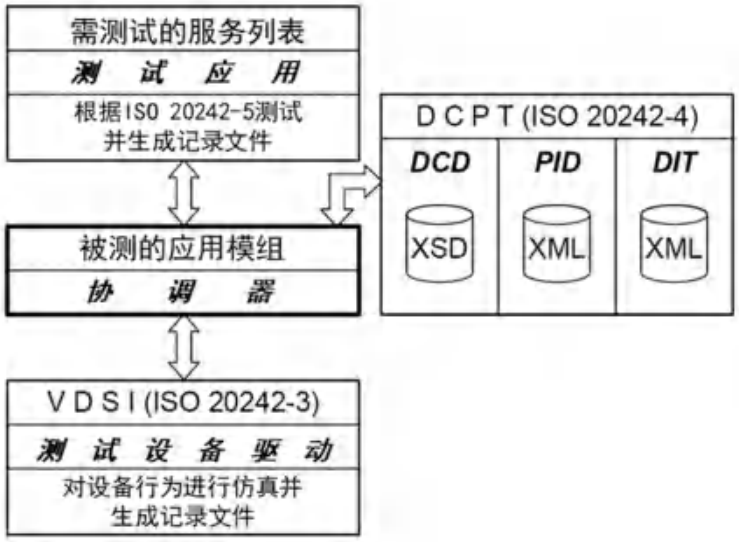


图 I.2—协调器的典型测试环境

根据数据表，测试和测试环境应具备运行测试所需的工具。例如，如果一个应用实现被检定为“限制符合 ISO 20242 系列标准”，必须要有一个测试应用程序来体现强调应用中 ISO 20242 系列接口的实现。

参 考 文 献

[1] ISO 15745-1, Industrial automation and systems integration — Open systems application integration frameworks — Part 1: Generic reference description

[2] ISO/IEC 19501, Information technology — Open Distributed Processing — Unified Modeling Language(UML) Version 1.4.2

[3] W3C XML 1.0 : http://www.w3.org/TR/REC-xml

[4] W3C XSD 1.0 : http://www.w3.org/TR/xmlschema-0/, xmlschema-1, xmlschema-2