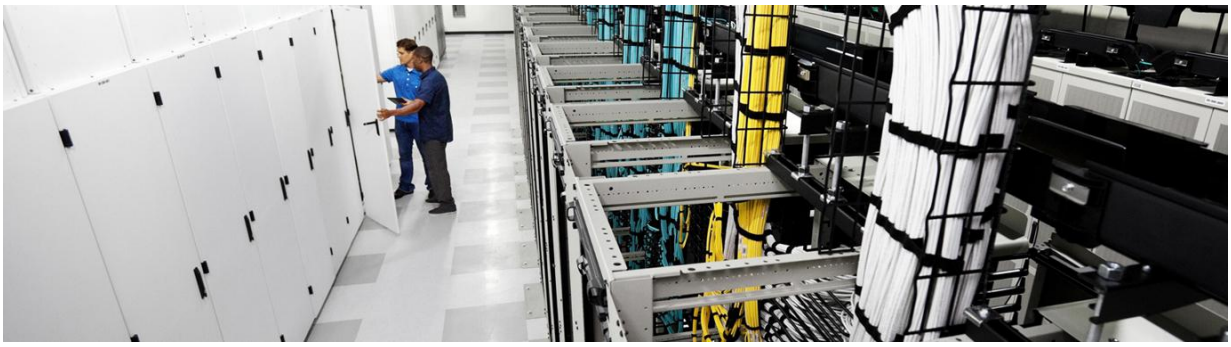
白皮书

凭借思科以应用为中心的基础设施实现网络可编程性



概述

本文档探讨思科® 以应用为中心的基础设施 (ACI) 的可编程性支持。Cisco ACI 可编程性模型允许对以应用为中心的基础设施实施进行完整编程访问。Cisco ACI 通过标准具象状态传输 (REST) API，提供对底层对象模型的读取访问，这是整个系统的每个物理和逻辑属性的代表。通过这种访问，客户可以将网络部署集成到管理和监控工具，并以编程方式部署新的工作负载。

目前网络可编程性方法面临的挑战

目前使用的大多数网络建立在硬件与软件紧密结合的基础之上，目的是通过命令行界面 (CLI) 来实施托管和管理。在静态网络配置、静态工作负载以及可预测的应用扩展变化率较低的情况下，这些系统运行良好。随着数据中心网络开始虚拟化并迁移到云和敏捷的 IT 模型，这种模型将不再起作用。

因此，供应商正在努力将可编程性添加到现有产品和设备的操作系统中。虽然这种方法可以增加功能，但并不是包含可编程性的理想方法。这种模型通过引入一个全新的管理点（通常为网络控制器），试图人为地将应用和用户策略映射到不灵活的网络结构，从而会使管理越来越复杂。此外，这些网络控制器及其模型受到网络功能的限制，无法进行扩展，不能支持基础设施的其余部分。真正的可编程性需要从基础上进行结合，而不是在事后被纳入。基础设施组件和结构需要采用开发人员可快速理解和使用的模型，在基础上具有可编程性设计。

凭借对象导向数据模型和 REST API 实现 Cisco ACI 可编程性

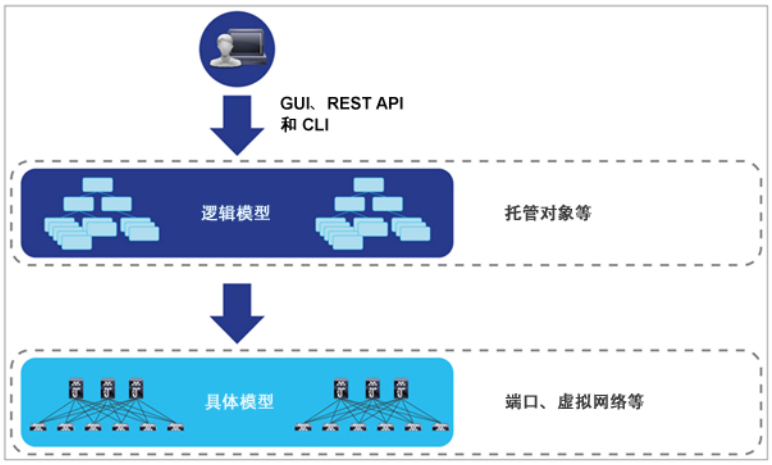
利用 Cisco ACI 解决方案，思科已经采取了一种基础方法来构建一个可编程的网络基础设施。此基础设施可作为单一系统在交换矩阵级别运行，由集中的思科应用策略基础设施控制器 (APIC) 进行控制。通过这种方法，数据中心网络可作为一个整体结合在一起，并用作一个可支持业务应用的智能传输系统。在该交换矩阵的网络设备上，我们对操作系统核心进行了改写，用以支持该系统视图，并可提供一个在基础上具有可编程性的架构。

与上一代的软件定义网络 [SDN] 解决方案不同（它们仅通过可编程接口开放了一部分网络功能），现在整个基础设施均可进行可编程访问。这是通过提供对 Cisco ACI 对象模型的访问来实现的。该模型可显示整个基础设施中每个单独的软件和硬件组件的完整配置和运行状态。此外，此对象模型可通过标准的 REST 接口来实施。从而能够更容易地访问和操控对象模型、系统配置和运行状态。

在顶层，Cisco ACI 对象模型基于承诺理论，可提供可扩展的控制架构，具有自治对象，可负责实施由控制器集群提供的状态变化。这种方法比传统的自上而下管理系统更具扩展性，因为它无需详细了解低级别配置和当前状态。凭借承诺理论，所需的状态变化向下渗透，对象实施变化，在需要时返回故障。

在这个高层次概念的下方，是 Cisco ACI 可编程性的核心：对象模型。该模型可以分为两个主要部分：逻辑和物理。基于模型的框架提供了一种表示数据的优雅方式。Cisco ACI 模型提供对底层信息模型的综合访问，可提供策略抽象、物理模型，以及调试和实施数据。图 1 描述了 Cisco ACI 模型框架。通过 REST API 可访问该模型，从而开启了系统的可编程性。

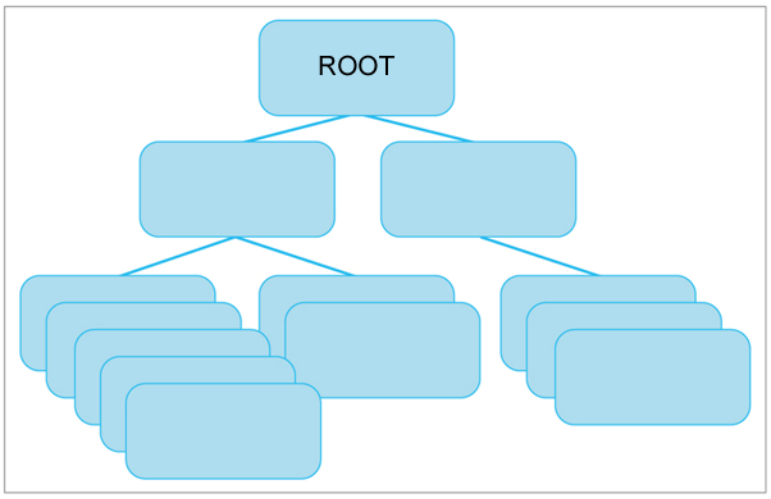
图 1. Cisco ACI 对象导向数据模型和 REST API



如图 1 所示，该逻辑模型是与系统的接口。管理员或上层云管理系统，通过 API、CLI 或 GUI 与该逻辑模型进行交互。然后，将逻辑模型的变化向下渗透到物理模型，通常会成为硬件配置。

逻辑模型本身包含可以操控的对象（配置、策略和运行状态）以及这些对象的属性。在 Cisco ACI 框架中，这种模型被称为管理信息树 (MIT)。MIT 中的每个节点代表一个托管对象或对象组。这些对象采用分层方式组织在一起，以创建逻辑对象容器。图 2 描述了 MIT 对象模型的逻辑层。

图 2. 管理信息树 (MIT)



MIT 中的对象

Cisco ACI 使用基于信息模型的架构，其中的模型描述了可由管理流程控制的所有信息。对象实例被称为托管对象 (MO)。系统中的每个托管对象通过唯一的可区别名称 (DN) 加以标识。采用此种方法，可以全局引用对象。

除了可区别名称外，每个对象还可以通过其相对名称 (RN) 进行标识。相对名称可标识与其父对象相关的对象。任何给定对象的可区别名称都由自身的相对名称与父对象的可区别名称组成。可区别名称直接映射到 URL。无论是相对名称还是可区别名称，都可用于访问对象，具体情况取决于 MIT 中的当前位置。托管对象、相对名称和可区别名称之间的关系，如图 3 所示。

图 3. 托管对象、相对名称和可区别名称

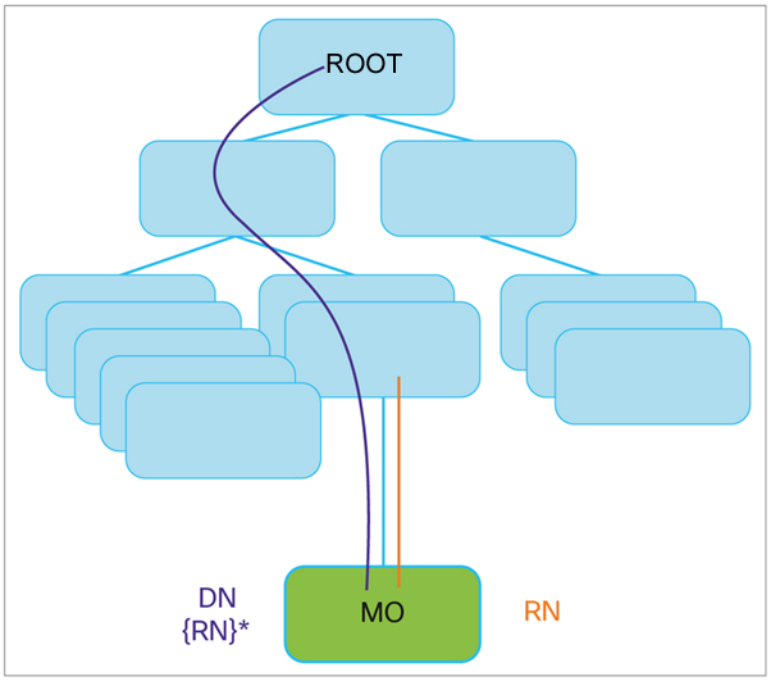


图 3 描述了唯一代表任何给定托管对象实例的可区别名称以及本地位于其父托管对象下方的相对名称。树中的所有对象都位于根对象下。

由于树的分层特性和用于确定对象类的属性系统，可以采用几种方法查询树来获取托管对象信息。可在对象上通过对象的可区别名称执行查询，在对象类（如交换机机箱）或在树级，发现对象的所有成员。图 4 显示了两个树级别查询。

图 4. 树级别查询

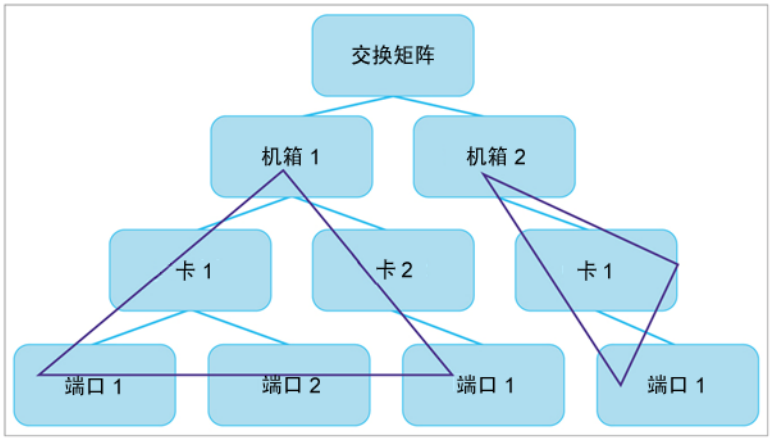
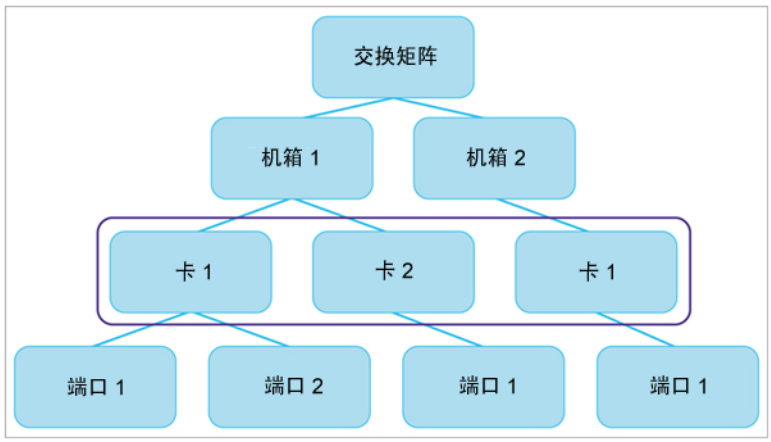


图 4 显示了在树级别上对两个机箱进行查询。这两个查询返回引用的对象及其子对象。这种方法是发现一个更大系统的所有组件的有用工具。

图 4 中的例子发现了给定交换机机箱的卡和端口。图 5 显示了另一种类型的查询：类级别查询。

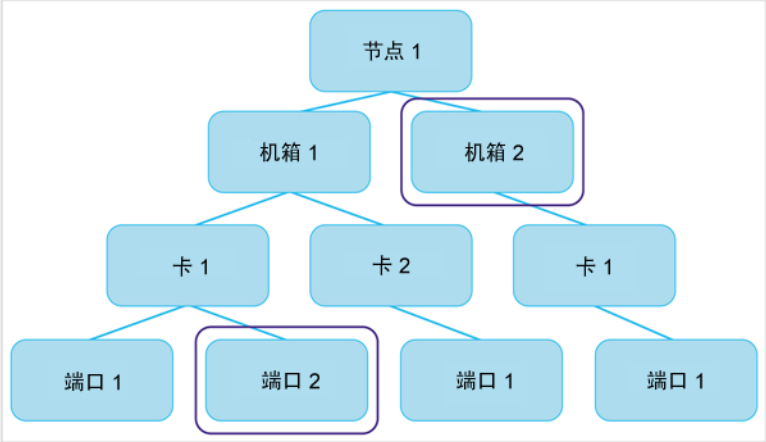
图 5. 类级别查询



如图 5 所示，类级别查询返回给定类的所有对象。这种方法对于发现 MIT 中提供的某种类型的所有对象非常有用。在这个例子中，使用的类是卡，可返回类型为卡的所有对象。

第三个查询类型是对象级别查询。在对象级别查询中，采用可区别名称来返回特定对象。图 6 描述了两个对象级别查询：一个适用于机箱 2 中的节点 1，另一个适用于端口 2 中的 卡 1 中的机箱 1 中的节点 1。

图 6. 类级别查询

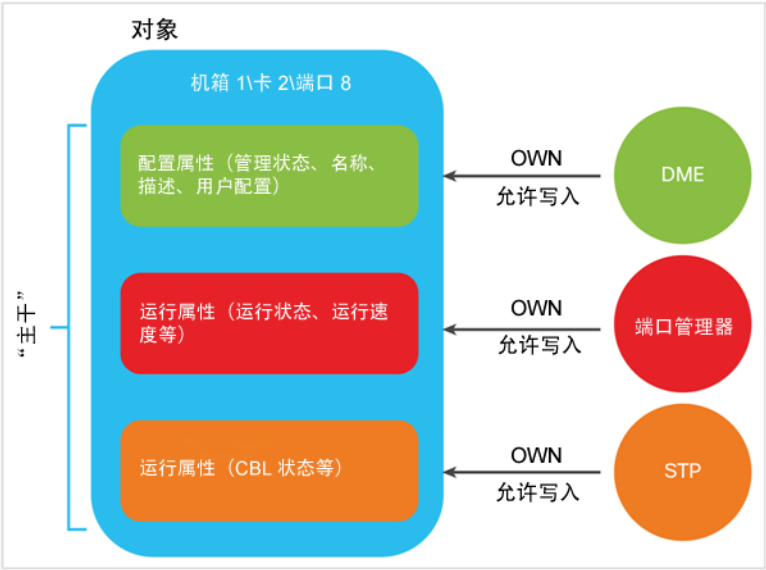


对于所有 MIT 查询，您可以选择返回整个子树或部分子树。此外，系统中基于角色的访问控制 (RBAC) 机制规定了返回的对象，只有用户有权查看的对象将永远返回。

托管对象属性

Cisco ACI 中的托管对象包含一些对托管对象给出定义的属性。托管对象中的属性分为多个数据块，由操作系统内部的特定流程进行管理。任何特定对象都可能有多个可访问该对象的流程。所有这些属性均在运行时一起编译，并作为一个单一对象呈现给用户。图 7 展示了这种关系的一个示例。

图 7. 托管对象属性



在图 7 中，示例对象具有三个可写入对象属性数据块的流程。作为 Cisco APIC（因此用户）和对象之间的接口的数据管理引擎 (DME)，处理端口配置的端口管理器，以及与该对中的数据块交互的生成树协议 (STP)。对象本身通过 API（在运行时编译的单一实体）来呈现给用户。

通过 REST 接口访问对象数据

REST 是适用于万维网等分布式系统的软件架构形式。过去几年，REST 已经逐渐成为主要的 Web 服务设计模型。REST 的形式更为简单，它正越来越多地取代其他设计模型，如简单对象访问协议 (SOAP) 和 Web 服务描述语言 (WSDL)。Cisco APIC 支持可编程访问整个 Cisco ACI 解决方案的 REST 接口。

Cisco ACI 的基于对象的信息模型使其完美适合 REST 接口：URL 和 URI 直接映射到不同的名称，以标识树上的对象，MIT 上的任何数据可被描述为采用 XML 进行编码的独立的结构化文本数文件或 JavaScript 对象符号 (JSON)。对象具有可使用可区别名称和属性进行识别的父子关系，可通过一组创建、读取、更新和删除 (CRUD) 操作来读取并修改。

对象可在其定义良好的地址和 REST URL 上进行访问，可以使用标准 HTTP 命令对 Cisco APIC 对象数据执行检索和操作。使用的 URL 格式可表示为：

```
<system>/api/[mo|class]/[dn|class][:method].[xml|json]?{options}
```

前述 URL 的各种构建块如下：

- **System**：系统标识符；IP 地址或 DNS 可解析主机名
- **mo | class**：指示这是否是托管对象或树 (MIT) 或类级别查询
- **class**：查询对象的托管对象类（在信息模型中指定）；类名可表示为 <pkgName><ManagedObjectClassName>
- **dn**：查询对象的可区别名称（MIT 树中对象的独特分层名称）
- **method**：在对象上可调用的方法的可选指示；仅适用于 HTTP POST 请求
- **xml | json**：编码格式
- **options**：查询选项、过滤器和参数

凭借使用 REST URL 可对单个对象或对象类进行寻址和访问的功能，可以实现对整个对象树的完整编程访问，并由此而及整个系统。

适用于编程环境的软件开发包

适用于 Cisco ACI 的 REST API 可实现与任何编程环境的轻松集成，而无需考虑所使用的语言和开发方法。为进一步加快在通用编程环境中的开发速度，可使用适用于 Cisco ACI 的软件开发包 (SDK)。Cisco ACI-pysdk 是基于 Python 的 SDK，就是这样一种适用于 Python 编程环境的 SDK。Python 库和 API 是底层 REST API 调用的 SDK 抽象的组成部分，可轻松快速集成到基于 Python 的软件包。

总结

Cisco ACI 对象导向数据模型的设计基础来源于网络可编程性。在设备级别，操作系统已经改写为一个适用于 Cisco ACI 的完全基于对象的交换机操作系统。Cisco ACI 的组件由可提供全功能 REST API 的 Cisco APIC 管理。在 API 的顶部是用于日常管理的 CLI 和 GUI。

使用 REST API，对象模型可实现流畅的可编程性，以及对底层基础设施组件的完全访问。对象可逻辑集成到分层模型并存储在 MIT 中。这种方法为网络控制和可编程性提供了框架，具有其他系统不具备的开放性。

更多详情

请访问 http://www.cisco.com/go/aci。

美洲总部
Cisco Systems, Inc.
加州圣何西

亚太地区总部
Cisco Systems (USA) Pte.Ltd.
新加坡

欧洲总部
Cisco Systems International BV
荷兰阿姆斯特丹

思科在全球设有 200 多个办事处。 地址、电话号码和传真号码均列在思科网站 www.cisco.com/go/offices 中。

美国印刷

C11-729999-00 11/13