

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 12/24 (2006.01)



[12] 发明专利说明书

专利号 ZL 200480031438.0

[45] 授权公告日 2009年7月22日

[11] 授权公告号 CN 100518080C

[22] 申请日 2004.9.7

[21] 申请号 200480031438.0

[30] 优先权

[32] 2003.9.26 [33] FR [31] 0311345

[86] 国际申请 PCT/FR2004/002271 2004.9.7

[87] 国际公布 WO2005/034429 法 2005.4.14

[85] 进入国家阶段日期 2006.4.24

[73] 专利权人 法国电信公司

地址 法国巴黎

[72] 发明人 埃米尔·斯特凡

杰西·勒莫尔拉克 - 朱威特

埃里克·莫罗

[56] 参考文献

US20030056193A1 2003.3.20

US5291583A 1994.3.1

US5737518A 1998.4.7

基于 SNMP 网络管理协议的 MIB 库分析及存取实现. 赵小蓉, 吕斌, 宋凯. 电脑开发与应用, 第 14 卷第 6 期. 2001

审查员 邹 婷

[74] 专利代理机构 北京市金杜律师事务所

代理人 朱海波

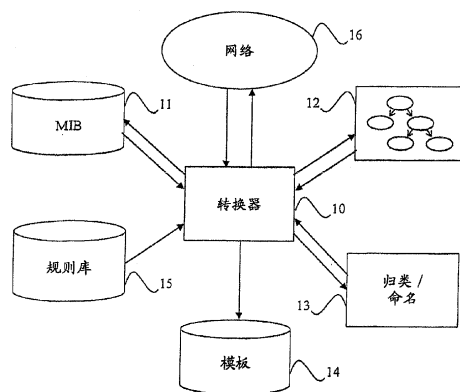
权利要求书 4 页 说明书 34 页 附图 19 页

[54] 发明名称

用于传送通信网络管理信息的方法和系统

[57] 摘要

本发明涉及一种用于传送通信网络管理信息的方法和系统。本发明的系统包括转换器模块(10), 目的是使传送通信网络管理信息所必需的带宽最小化, 通信管理信息即涉及与硬件单元或软件单元或网络操作有关的对象的信息, 这些对象列于管理信息数据库(11)中并且每个对象具有与之相关联的正式的语言规范。模块(10)设计为诸如根据每个对象的规范生成字对, 该字对的第一个字的值与该对象的类型有关, 并且该字对的第二个字的值与该对象的信息长度有关。此外, 模块(10)设计为诸如生成一个或多个模板, 所述模板包括一组定制的字对和一个标识符, 因此支持对应于每个模板的定制信息流的随后的传送。



1. 一种用于使传送通信网络管理信息所需的带宽最小化的方法，所述信息涉及与硬件、软件或网络操作单元有关的对象，所述对象列于管理信息库（11）中并且每个对象与一个正式的语言规范相关联，所述方法的特征在于其包括步骤：

- 基于用于每个对象的所述规范生成（48-50）字对（121），所述字对中的第一个字的值与 所述对象的类型有关，并且所述字对中的第二个字的值与所述对象的信息长度有关；

- 构造（41-47）模板，所述模板包括一组定制的所生成的字对（121-133）和所述模板的标识符（119-120），使得能够在随后发送对应于所述模板的定制信息串（99-113）。

2. 根据权利要求1所述的方法，其特征在于包括步骤：

- 遍历（43-46）所述管理信息库（11）的树，该树的每个节点与一个对象相关联；

- 在每个节点上测试（44）所述对象是标量类型的还是表类型的；

- 如果所述对象是标量类型的，则通过将所述生成的字对附加到所述模板来构造（41-47）所述模板；

- 如果所述对象是表类型的，则为所述表的对象构造（51-61）表模板。

3. 根据权利要求1或2所述的方法，其特征在于包括步骤：

为可以以更改的方式访问的对象另外创建（33-37,19-22）包括所述生成的字对的配置模板。

4. 一种发送通信网络管理信息的方法，所述信息涉及与硬件、软件或网络操作单元有关的对象，所述对象列于管理信息库（11）中并且每个对象与一个正式的语言规范相关联，所述方法的特征在于包括以下步骤：

- 获得模板，所述模板一方面包括所述模板的标识符（119-120），并且另一方面包括一组定制的字对（121-133），每个字对都是基于

与所述对象相关联的规范而为所述对象中的一个对象生成的,并且每个字对所包括的第一个字的值与 所述对象的类型有关且每个字对所包括的第二个字的值与所述对象的信息长度有关;

- 发送对应于所述模板的定制信息串(99-113)。

5. 一种用于使传送通信网络管理信息所需的带宽最小化的系统,所述信息涉及与硬件、软件或网络操作单元有关的对象,所述对象列于管理信息库(11)中并且每个对象与一个正式的语言规范相关联,所述系统的特征在于包括:

转换器模块(10), 包括:

用于基于用于每个对象的所述规范生成字对的装置,所述字对中的第一个字的值与 所述对象的类型有关,并且第二个字的值与所述对象的信息长度有关; 并且

用于生成模板的装置,所述模板包括一组定制的字对和标识符,使得能够在随后发送对应于所述模板的定制信息串。

6. 根据权利要求5所述的系统,其特征不在于所述转换器模块(10)包括:

用于遍历所述管理信息库(11)的树的装置,该树的每个节点与一个对象相关联;

用于在每个节点上测试所述对象是标量类型还是表类型的装置; 并且

用于如果所述对象是标量类型的则通过将所述生成的字对附加到所述模板来构造所述模板,或者如果所述对象是表类型的则为所述表的对象构造表模板的装置。

7. 根据权利要求5或6所述的系统,其特征不在于所述转换器模块(10)包括:

用于为可以可更改地访问的对象另外构造包括所述生成的字对的配置模板的装置。

8. 根据权利要求5-6中的任何一项所述的系统,其特征不在于包括:

管理模块(87),其设计为收集测量数据;以及
输出模块(88),其设计为向服务器(92)发送与这些测量有关的至少一个数据票证。

9. 根据权利要求8所述的系统,其特征在于所述输出模块(88)包括用于进行如下操作的装置:

- 发送包括对一个模板引用的数据票证;
- 在发送时,首先发送在所述数据票证中所引用的模板。

10. 一种转换器模块(10),用于一种用于使传送通信网络管理信息所需的带宽最小化的系统,所述信息涉及与硬件、软件或网络操作单元有关的对象,所述对象列于管理信息库(11)中并且每个对象与一个正式的语言规范相关联,所述转换器模块(10)的特征在于包括:

用于基于用于每个对象的所述规范生成字对的装置,所述字对中的第一个字的值与 所述对象的类型有关,并且所述字对中的第二个字的值与所述对象的信息长度有关;以及

用于生成模板的装置,所述模板包括一组定制的字对和一个标识符,使得能够在随后发送对应于所述模板的定制信息串。

11. 根据权利要求10所述的转换器模块(10),其特征在于包括:

用于遍历所述管理信息库(11)的树的装置,该树的每个节点与一个对象相关联;以便

用于在每个节点上测试所述对象是标量类型还是表类型的装置;并且

用于如果所述对象是标量类型的则通过将所述生成的字对附加到所述模板来构造所述模板,或者如果所述对象是表类型的则为所述表的对象构造表模板的装置。

12. 根据权利要求10或11所述的转换器模块(10),其特征在于包括:

用于为可以可更改地访问的对象另外构造包括所述生成的字对

的配置模板的装置。

13. 一种管理模块(87)，用于根据权利要求5所述的系统，其特征在于所述管理模块(87)包括：

用于收集测量数据的装置，基于这些测量数据发送所述管理信息。

14. 一种输出模块(88)，用于根据权利要求5所述的系统，所述系统包括根据权利要求13所述的管理模块(87)，其特征在于所述输出模块(88)包括：

用于向服务器(92)发送与由所述管理模块(87)执行的测量有关的至少一个数据票证的装置。

15. 根据权利要求14所述的输出模块(88)，其特征在于所述输出模块(88)包括用于进行如下操作的装置：

- 发送包括对一个模板的引用的数据票证；
- 在发送时，首先发送在所述数据票证中所引用的模板。

用于传送通信网络管理信息的方法和系统

技术领域

本发明所涉及的领域是通信网络的管理。术语“管理”在此取其最广泛的含义，也就是说其同时涉及配置和管理，或者甚至涉及对通信网络的设备的控制。

背景技术

为了配置一项设备，例如可以使用诸如已知的 TELNET 协议（远程登录协议）之类的交互式接口协议。这种协议已经标准化，但是可访问的数据并没有标准化。特别是就网络设备的通常相当可观的数量而言，这将会带来某种问题。

管理包括故障的预测和检测等。在现有技术中，作为例子，需要提到的是文献 WO 02/46928，其公开了一种对从传感器获得的信息进行处理的系统，其中的传感器与列于称作 MIB（管理信息库）的管理信息库中的变量相关联。为了实现与多项设备有关的解译和大规模处理，通过称作 SMI（管理信息结构）的标准化语言规定了变量的定义。同样进行了标准化的称作 SNMP（简单网络管理协议）的协议使得能够通过在网络设备之间交换“查询/响应”而对变量进行访问。

如同在例如文献 WO 02/17094 中的情况下，变量可以与设备有关，这些设备更可能是关于温度、报警状态或 IP（网际协议）类型的网络地址的传感器而不是更复杂的设备。针对这些设备，可以引用列于一个或多个不同的 MIB 中的对象。该文献公开了用于基于 SNMP 协议对设备进行接口连接的装置。

基于 SNMP/SMI/MIB 这三者的方法已经达到一定的完备程度。其从语义和大小这两个方面规定了 MIB 的规范本身以及列于其中的

对象的规范。可以通过例如在文献 US 600 9431 中提出的那些自动化处理器 (handler) 来对 MIB 的规则进行微调。利用标准 (X208 和 X209) 绝对地标识对象并利用实例索引绝对地标识对象的实例, 意味着 MIB 提供了操作者重视的标准化基准。如同文献 WO 01/44924、EP 115 8720 或者还有 WO 02/47322 所证明的那样, SNMP 协议被广泛地用于众多类型的设备和服务。

然而, 对于传输大量数据来说, SNMP 协议是不令人满意的, 原因是这种协议由于附加信息而增加了相当大的开销。“查询/响应”模式(轮询)使得很难在网络设备中实现对数据的内部管理的最优化。设备通信吞吐量的增长增大了计算机中尾数过量 (mantissa overshoot) 的风险, 连锁效应是因此增大了“查询/响应”的频繁程度(需要这种查询/响应来避免这种尾数的过量)。在机器之间对实例标识符的交换将所需的带宽显著地增大到 3 倍的第一近似值, 达到了损及用于用户数据的有用带宽的程度。尽管大约 85% 的 MIB 对象是表字段, 尽管大约 99.9% 的实例是这些对象的实例, 即表字段的实例, 但是 SNMP 协议不能使表查询最优化。尽管表中一行的字段具有相同的索引, 但是 SNMP 协议针对每个字段重复索引, 因此针对表中所查询的每一行增加了大约一百字节的额外吞吐量。

发明内容

本发明的一个目的是一种用于使传送通信网络管理信息所需的带宽最小化的方法, 所述信息涉及与硬件、软件或网络操作单元有关的对象, 所述对象列于管理信息库中并且每个对象与一个正式的语言规范相关联。

该方法的特征在于其包括以下步骤:

- 基于用于每个对象的所述规范生成字对, 所述字对中的第一个字的值与对所述对象的类型有关, 并且所述字对中的第二个字的值与所述对象的信息长度有关;

- 构造模板, 所述模板包括一组定制的所生成的字对和所述模板

的标识符，使得能够在随后发送对应于所述模板的定制信息串。

更特别地，该方法包括步骤：

- 遍历所述管理信息库的树，该树的每个节点与一个对象相关联；
- 在每个节点上测试所述对象是标量类型的还是表类型的；
- 如果所述对象是标量类型的，则通过将所述生成的字对附加到所述模板来构造所述模板；
- 如果所述对象是表类型的，则为所述表的对象构造表模板。

有利地，该方法包括步骤：

- 为可以以更改的方式访问的对象另外构造包括所述生成的字对的配置模板。

本发明的另一个目的是一种用于使传送通信网络管理信息所需的带宽最小化的系统，所述信息涉及与硬件、软件或网络操作单元有关的对象，所述对象列于管理信息库中并且每个对象与一个正式的语言规范相关联。

该系统的特征在于其包括转换器模块，该转换器模块包括：

- 用于基于用于每个对象的所述规范生成字对的装置，所述字对中的第一个字的值与对所述对象的类型有关，并且所述字对中的第二个字的值与所述对象的信息长度有关；以及

- 用于生成模板的装置，所述模板包括一组定制的字对一个模板的标识符，使得能够在随后发送对应于所述模板的定制信息串。

更特别地，该转换器模块包括：

- 用于遍历所述管理信息库的树的装置，该树的每个节点与一个对象相关联；以便

- 用于在每个节点上测试所述对象是标量类型还是表类型的装置；并且

- 用于如果所述对象是标量类型的则通过将所述生成的字对附加到所述模板来构造所述模板，或者 - 如果所述对象是表类型的则为所述表的对象构造表模板的装置。

有利地，该转换器模块包括：

- 用于为可以可更改地访问的对象另外构造包括所述生成的字对的配置模板的装置。

例如，该系统包括：

- 管理模块，其设计为收集测量数据；以及
- 输出模块，其设计为向服务器发送与这些测量有关的至少一个数据票证，在此之前先发送该数据票证的模板。

附图说明

根据下文中参考附图而描述的示例性实施例，可以更好地理解本发明，其中：

图 1 是根据本发明的系统示图；

图 2-图 7 示出了根据本发明的方法步骤；

图 8 表示用于一个特定的表类型对象的树结构；

图 9 示出了图 8 中表示的对象的模板；

图 10 示出了根据图 9 中的模板的示例性数据票证 (data ticket)；

图 11 表示用于另外一个特定的表类型对象的树结构；

图 12 示出了图 11 中表示的对象的模板；

图 13 示出了根据图 12 中的模板的示例性数据票证；

图 14 是可能的工业应用的示图；

图 15 示出了由图 14 中的系统生成的一个表模板；

图 16 示出了根据图 15 中的模板的示例性数据票证；

图 17 示出了由图 14 中的系统生成的另一个表模板；

图 18a-图 18c 示出了根据图 17 中的模板的示例性数据票证。

具体实施方式

在下文中参考图 1 所描述的系统，转换器 10 包括与网络 16 进行通信以便接收命令并发送数据的装置。转换器 10 是诸如计算机之

类的机器,其包括处理器以及用于实现稍后将参考图 2-图 6 进行描述的方法步骤所需的程序。转换器 10 包括用于对数据库 11 和数据库 15 进行读取的装置,用于对大容量存储器 14 进行写入的装置,用于对存储器 12 进行读取和写入的装置,以及用于访问模块 13 的装置。

数据库 11 包含管理信息库(代表管理信息库的 MIB 或代表处理信息库的 PIB)的规范。

数据库 15 包含转换规则,每个转换规则用于为在数据库 11 中规定的至少一个对象提供一个模板字段类型。

存储器 14 设计为包含由转换器 10 生成的模板。

存储器 12 设计为包含与 MIB 对象或 PIB 对象有关的树。

模块 13 设计为执行对由转换器 10 传送的对象的归类(typing)或命名。模块 13 配备了处理器和实现稍后将参考图 7 进行描述的方法步骤所需的程序。

参考图 2, 第一步骤 1 包括通过在转换器 10 中将树与一个名为“ObjectRoot”(对象根)的参数相关联而激活用于树的模板的生成。该树可以指向一组 MIB 或一组 PIB, 指向一个 MIB、MIB 的一个子树或 MIB 的一个表。可以根据需要进行这种激活,或在基于网络 16 而对系统进行初始化的框架下进行这种激活。这种激活提供了用于对模板进行编号的基准,该基准是名为“base_template_ID”(基准_模板_标识符)的参数。当然可以理解,在本说明书中所引用的标记中给出的名称只是以便于记忆的方式选择的,不具有任何特定的意义并且其具有的值是非限制性的,也就是说,在不偏离本发明的框架的情况下可以选择任意其他的名称,当然,用于说明书的所选名称应当始终表示与参考附图时所用的数据相同的数据。这种激活还提供了与配置模板的生成有关的参数,并且该参数在此名为“Generation_templates_configuration”(生成_模板_配置)。当这一参数的值例如设定为 0 时,这表示该方法应当只生成标准模板,也就是主要与监控和信令相关的那些模板。当这一参数的值例如设定为 1 时,这表示该方法还应当生成配置模板,也就是主要与设备配置相关

的那些模板。标准模板适于所考虑的树的所有对象和索引，而配置模板适于索引、可以写访问（write-accessible）的对象和对象的创建。

第二步骤 2 包括从存储器 11 中读取所请求的 MIB 或 PIB 的规范。该存储器可以是任意类型的存储器（随机存取存储器、磁盘或网络）。该步骤是可选步骤。如果在转换器 10 中该规范已经可用，则不需要这一步骤。在处理过程中，如有必要，可以重复这一步骤。

第三步骤 3 包括创建树，其用于根据 MIB 或 PIB 中包含的对象的标识符的值来命名这些对象。如果在转换器 10 中已经存在该树，则这一步骤是可选步骤。为了构造该对象命名树（object naming tree），转换器 10 读取 MIB 或 PIB 中以 SMI 语言编写的数据模型的规范，解译每个对象的定义的每个子句以便使用对象的对象标识符（OID）将每个对象放入该命名树中。重新提及，对象标识符（OID）是 ASN.1 [X208] 语言的一种特殊类型，其使用所定义的唯一的一棵树，以便将可移植标识符（portable identifier）与数据项相关联。该标识符是绝对的且可传送的。通常使用的传送功能在 [X209] 中定义。例如，SNMP 使用 X209 的编码规则。X209 通常称作 BER（基本编码规则）。

在第四步骤 4 中，转换器 10 通过执行下文中参考图 3a 和图 3b 所描述的步骤创建模板。

步骤 39 包括读取步骤 1 中激活的关于“ObjectRoot”对象的树中存在的对象的模板生成请求，以及读取模板编号基准参数“base_template_ID”。

步骤 40 包括创建一个或多个常量，其使得能够按照模板适于 MIB、适于 PIB 或者适于另一种规范语言类别来区分模板的类别。在图 3a 的例子中，将适于 MIB 的常量称作“DefinitionOf TemplateOf MIB”（对 MIB 模板的定义），为其分配对所有 MIB 模板都通用的值 2。可以为其他常量分配其他值，表明在后续步骤中构造的结构是 PIB 或其他模板。

步骤 41 包括创建用于该树的模板“DefTicket_ObjectRoot”（定义票证_对象根）。该模板是初始为空的例如成对的 16 位字（sixteen-bit

word) 的列表。

步骤 42 包括创建变量

“NumberOftheCurrentField_DefTicket_ObjectRoot” (当前字段编号_定义票证_对象根) 并为其分配值 0, 创建变量 “current_template_ID” (当前_模板_标识符) 并为其分配参数 “base_template_ID” (基准_模板_标识符) 的值, 创建变量 “ObjectRoot_template_ID” (对象根_模板_标识符) 并为其分配 “current_template_ID” 的值, 并且最后递增变量 “current_template_ID” 的值。第一个变量用于对将在后续步骤中为构造模板而逐步添加的数据字段进行编号。第二个变量用于对将在后续步骤中创建的不同模板进行编号, 因此树的模板本身利用第三个变量进行编号, 第三个变量的值是步骤 1 中提供的基准参数的值。最后递增第二个变量, 以便对于在后续步骤中根据树的对象生成另一个模板时可以利用该第二个变量的新的值。

在创建标准模板的步骤 41 和步骤 42 之后, 步骤 35 包括测试参数 “Generation_templates_configuration” (生成_模板_配置) 是否等于 1。如果测试结果为肯定, 则开始用于创建与标准模板并列的配置模板的步骤 36 和步骤 37。如在下文中将看到的, 配置模板只与标准模板的对象中的可更改的对象有关。由此得到的较小大小减小了在配置期间必需的信息流, 从而加快了配置的进行。标准模板的目的是在变量的当前状态下分配变量的值, 通常是在从进行测量到进行处理之间进行这种分配。配置模板一部分用于固定的变量值, 甚至用于在测量系统中创建新的测量, 其流程的方向通常是从进行管理到进行测量。

步骤 36 与步骤 41 性质相同, 其包括创建名为 “DefTicket_configuration_ObjectRoot” (定义票证_配置_对象根) 的配置模板。

步骤 37 与步骤 42 性质相同, 其包括创建名为 “DefTicket_configuration_ObjectRoot_Template_ID” (定义票证_配置_对象根_模板_标识符) 的变量并为其分配 “current_template_ID”

的值,以及创建名为“NumberOfField_configuration_ObjectRoot”(字段编号_配置_对象根)的变量并为其分配值0。第一个配置变量用于对紧随标准模板之后的配置模板进行编号。第二个配置变量用于对配置模板的“NumberOfField”(字段编号)进行计数,配置模板的“NumberOfField”小于或等于标准模板的“NumberOfField”。

在步骤36和步骤37之后,步骤38包括递增变量“current_template_ID”以便标识随后有可能出现的模板创建。在步骤35的测试的结果为否定的情况下,还要激活步骤38,以便独立于创建或不创建配置模板的事实而递增变量“current_template_ID”。因此,每个标准模板都可以具有相同性质(例如偶数)的标识符,并且相应的配置模板可以具有后一数字(因此是奇数)的标识符,这使得必定可以区分模板的性质并将相同性质的模板匹配起来。

分别地,由步骤41和步骤42创建初始为空的的标准模板,并且(如果在步骤1中给出了指令)由步骤36和步骤37创建初始为空的配置模板,在步骤38之后,将通过用于对树进行遍历的程序来继续该方法,下文将参考图3b对此进行描述。

步骤43-47的一系列步骤包括从根“ObjectRoot”开始扫描包含在存储器12中的树。

步骤43包括将当前的对象初始化为根对象。

步骤44包括测试该对象是否为标量类型的对象。一个MIB包含两类对象,单个对象也称作标量或表,表将对象一起分组为行,每一行用索引来标识。对步骤44的测试的肯定响应将触发一个标量步骤。对步骤44的测试的否定响应将触发步骤45。

步骤45包括测试该对象是否为表类型的对象。对步骤45的测试的肯定响应将触发一个表步骤。对步骤45的测试的否定响应将直接触发稍后描述的步骤47。稍后将分别参考图4和图5对表步骤和标量步骤进行描述。

作为执行表步骤和标量步骤中的一种步骤的前导,步骤9包括从规则库15中提取表对象或标量对象的转换规则。

在根据具体情况执行表步骤或标量步骤之后，步骤 46 测试当前对象是否为该树的最后一个对象。对步骤 46 的测试的肯定响应将触发稍后将参考图 6 进行描述的结束步骤。对步骤 46 的测试的否定响应将触发步骤 47。步骤 47 取该树上的下一个对象作为当前对象并且使该方法的执行循环回步骤 44。

在执行结束步骤之后，步骤 6 对该方法生成的模板进行传送，以响应从网络 16 发起的、直接从另一项设备发起的或在内部发起的查询。

参考图 4a，表步骤开始于步骤 30，在步骤 30 中创建名为“DefTicket_Table_template_ID”（定义票证_表_模板_标识符）的变量并为其分配“current_template_ID”的当前值。该变量用作在步骤 44 中检测的表的标准模板的标识符。

在步骤 31 中，递增“current_template_ID”的值，以便可用于可能的随后的模板创建。

然后，步骤 51 包括创建名为“DefTicket_Table_Ti”（定义票证_表_Ti）的表模板。与在步骤 41 中创建的树模板相同，在步骤 51 中创建的表模板也是初始为空的列表。

在进行标准模板创建的步骤 30、31 和 51 之后，步骤 19 包括测试参数“Generation_templates_configuration”（生成_模板_配置）是否等于 1。对该测试的肯定响应将触发用于创建与标准模板并列的配置模板的步骤 20-22。如同下文中将看到的，配置模板只涉及标准模板的对象中的可更改的对象。由此得到的较小大小减小了在配置期间必需的信息流，从而加快了配置的进行。

与步骤 51 性质相同的步骤 20 包括创建名为“DefTicket_configuration_Table_Ti”（定义票证_配置_表_Ti）的配置模板。

与步骤 30 性质相同的步骤 21 包括创建名为“DefTicket_configuration_Table_Ti_Template_ID”（定义票证_配置_表_Ti_模板_标识符）的变量并为其分配“current_template_ID”的值。

步骤 22 包括创建名为 “NumberOfField_Configuration_Table_Ti” (字段编号_配置_表_Ti) 并为其分配值 0。

在步骤 21 中创建并设置的变量用于对紧随标准模板之后的配置模板进行编号。在步骤 22 中创建并设置的变量用于对配置模板的 “NumberOfField” 进行计数，配置模板的 “NumberOfField” 小于或等于标准模板的 “NumberOfField”。

在步骤 22-22 之后，步骤 23 包括递增变量 “current_template_ID” 的值，以便标识可能的随后的模板创建。在对步骤 19 的测试的响应为否定的情况下，还要激活步骤 23，以便独立于创建或不创建配置模板的事实而递增变量 “current_template_ID”。因此，每个标准模板都可以具有相同性质（例如偶数）的标识符，并且相应的配置模板可以具有后一数字的标识符，这使得必定可以区分模板的性质并将相同性质的模板匹配起来。

步骤 52 包括创建名为 “NumberOftheCurrentField_DefTicket_Table_Ti” (当前字段编号_定义票证_表_Ti) 的本地变量并为其分配值 0。在步骤 51 之后或在步骤 23 之后执行步骤 52，当前字段的编号用于对标准表模板的数据字段进行索引并用于计算其量。

分别地，由步骤 30、51 和 52 创建初始为空的的标准模板，并且（如果在步骤 1 中给出了指令）由步骤 20-22 创建初始为空的配置模板，该方法将继续进行表遍历步骤，下文将参考图 4b 对此进行描述。

在步骤 54 中，转换器 10 读取表的录入行 (entry row) 的 SMI 定义以及该定义的 INDEX (索引) 子句。

在此之后的步骤 55-57 的循环包括依次读取该索引的每个对象的定义 SMI。在步骤 55 中，转换器从执行了步骤 54 之后的第一个对象开始读取索引的对象。在触发了作为前导的步骤 9 以便从规则库 15 中提取用于该对象的转换规则（如果存在该转换规则的话）之后，转换器 10 随后触发用于步骤 55 中读取的对象的标量步骤。步骤 9 实际上是可选的。

步骤 56 包括测试当前对象是否为该索引的最后一个对象。对该测试的否定响应使进程循环回用于该索引的下一个对象的步骤 57，以便迭代地重复步骤 55。对步骤 56 的测试的肯定响应表明已经处理了该索引的所有对象。

在最近执行步骤 56 之后，步骤 58-61 的循环包括对不是该索引的一部分的表的对象进行处理。

在步骤 58 中，转换器 10 从最近执行步骤 56 之后的表的行定义的第一个对象开始读取行对象。

在步骤 59 中，转换器 10 测试当前对象是否为该索引的当前对象。对该测试的否定响应在执行用于当前对象的步骤 9 之后触发标量步骤，在该标量步骤之后激活步骤 60。对步骤 59 的测试的肯定响应直接接触步骤 60。

在步骤 60 中，转换器 10 测试当前对象是否为该表中的最后一个对象。对步骤 60 的测试的否定响应触发步骤 61，在步骤 61 中转换器 10 重新激活用于该表的下一个对象的步骤 58。对步骤 60 的测试的肯定响应触发结束步骤，稍后将参考图 6 对该结束步骤进行描述。

现在参考图 5 描述标量步骤。在步骤 48 中，名为“NumberOftheCurrenField”的值包括标量步骤的调用参数。因此，当从步骤 44 中触发标量步骤时，变量“NumberOftheCurrenField”对应于变量“NumberOftheCurrenField_DefTicket_ObjectRoot”。当从步骤 55 或步骤 59 中触发标量步骤时，变量“NumberOftheCurrenField”对应于“NumberOftheCurrenField_DefTicket_Table_Ti”。步骤 48 递增变量“NumberOftheCurrenField”的值。

在步骤 49 中，转换器 10 读取对象的 SMI 定义，以便从中提取对象的类型、对象的子类型、对象的最小大小、对象的最大大小以及对象的访问模式。访问模式使得可能区分该当前对象是否为可更改的。

在步骤 7 中，转换器 10 将该对象的类型、该对象的子类型、该对象的最小大小、该对象的最大大小、“NumberOftheCurrenField”

和用于该对象的转换规则发送到命名归类模块 13。

在步骤 8 中，归类模块 13 将有关该当前对象的字对（类型、长度）的值发送到转换器 10。

在步骤 50 中，转换器 10 将该字对（类型、长度）添加到在本地命名为“DefTicket_X”（定义票证_X）的模板，如果从步骤 44 中触发标量步骤，则该模板为“DefTicket_ObjectRoot”（定义票证_对象根），如果从步骤 55 或步骤 59 中触发标量步骤，则该模板为“DefTicket_Table_Ti”（定义票证_表_Ti）。

在执行步骤 50 之后，转换器 10 激活测试步骤 32，步骤 32 包括验证是否创建了配置模板，也就是说“Generation_template_configuration”（缩写为 Gtc）是否等于 1，以及验证当前对象是否可更改。

对步骤 32 的测试的肯定响应触发步骤 33，在步骤 33 中递增在本地命名的变量“NumberOfFields_Configuration”（字段编号_配置）。如果从步骤 44 中触发标量步骤，则变量“NumberOfFields_Configuration”对应于“NumberOfFields_Configuration_ObjectRoot”（字段编号_配置_对象根），如果从步骤 55 或步骤 59 中触发标量步骤，则变量“NumberOfFields_Configuration”对应于“NumberOfFields_Configuration_Table_Ti”（字段编号_配置_表_Ti）。

在步骤 34 中，转换器 10 将字对（类型、长度）添加到在本地命名为“DefTicket_Configuration_X”（定义票证_配置_X）的模板，如果从步骤 44 中触发标量步骤，则该模板为“DefTicket_Configuration_ObjectRoot”（定义票证_配置_对象根），如果从步骤 55 或步骤 59 中触发标量步骤，则该模板为“DefTicket_Configuration_Table_Ti”（定义票证_配置_表_Ti）。可以看到，采用与在标准模板中相同的标记即当前字段编号将字对（类型、长度）添加到配置模板。因此，以相同的方式在标准模板和配置模板中标记

与同一对象的类型和长度有关的每个字段，因此利用该方法实现了设备对数据的处理。

在执行步骤 34 之后，或在对步骤 32 的测试的否定响应之后，标量步骤结束并返回，以便继续执行该方法，如果从步骤 44 中触发标量步骤，则从步骤 46 开始继续执行，如果从步骤 55 中触发标量步骤，则从步骤 56 开始继续执行，如果从步骤 59 中触发标量步骤，则从步骤 60 开始继续执行。

下文将参考图 6 描述结束步骤。

在步骤 62 中，转换器 10 在本地名为“DefTicket_X”的模板的开头插入包括字对（类型、长度）的元素的定义，如果从步骤 46 中触发该结束步骤，则该模板对应于名称“DefTicket_ObjectRoot”，如果从步骤 60 中触发该结束步骤，则该模板对应于名称“DefTicket_Table_Ti”。

在步骤 63 中，转换器 10 为在步骤 62 中插入的元素的定义的第一项分配在本地名为“Template_X_ID”（模板_X_标识符）的变量的值，为其第二项分配“NumberOftheCurrentField”。如同之前那样，根据步骤的调用情况，用“ObjectRoot”或“Table_Ti”来代替“X”。分配给第一项的类型值包括来自所创建的标准模板的模板的标识符。为第二项分配的字段编号给出了模板的长度，用字段量来表示，每个字段对应于与根据具体情况在标量步骤或在表步骤中进行处理的对象有关的字对（类型、长度）。

在步骤 64 中，转换器 10 在从步骤 62 中得到的且名称“DefTicket_X”对应于与步骤 62 中相同的模板名称的模板的开头添加包括字对（类型、长度）的元素定义。

在步骤 65 中，转换器 10 将步骤 40 中的模板定义常量的值分配给在步骤 64 中插入的元素定义的第一项。例如，值 2 表示所生成的字对串是 MIB 模板。转换器 10 将值 2 添加到“NumberOftheCurrentfield”以便包括两个字段，每个字段分别对应于在步骤 62 中插入的字对和在步骤 64 中插入的字对。当字对的每项包

括例如 2 字节的字时，转换器 10 将之前获得的结果乘以 4,以便获得以字节表示的模板长度，随后将该模板长度分配给在步骤 64 中插入的字对的第二项。

在步骤 5 中，转换器 10 将因此获得的标准模板记录在大容量存储器 14 中。

步骤 24 包括测试是否请求创建配置模板，也就是说参数“Gtc”（缩写）是否等于 1。否定响应使步骤直接返回，不需要创建任何配置模板。

对步骤 24 的测试的肯定响应触发第二个测试步骤 25，步骤 25 包括测试配置字段的编号是否为正值。对该测试的否定响应使该步骤返回，原因是当针对配置模板而扫描的对象都不可更改时，“NumberOfField”保持为 0,并且没有理由再记录配置模板。

对步骤 25 的测试的肯定响应触发一连串的步骤 26-29,这一连串的步骤终止于步骤 5,步骤 5 用于在步骤返回之前记录配置模板，应当注意，存在至少一个对应于可更改对象的字对。

在步骤 26 中，转换器 10 在本地名为“DefTicket_Configuration_X”的模板的开头插入包括字对（类型、长度）的元素的定义，如果从步骤 46 中触发该结束步骤，则该模板对应于名称“DefTicket_Configuration_ObjectRoot”，如果从步骤 60 中触发该结束步骤，则该模板对应于名称“DefTicket_Configuration_Table_Ti”。

在步骤 27 中，转换器 10 为在步骤 26 中插入的元素的定义的第一项分配在本地名为“Template_Configuration_X_ID”(模板_配置_X_标识符)的变量的值，为其第二项分配当前的“NumberOftheFields”。如同之前那样，根据步骤的调用情况，用“ObjectRoot”或“Table_Ti”来代替“X”。分配给第一项的类型值包括来自所创建的配置模板的模板的标识符。为第二项分配的字段编号给出了模板的长度，用字段量来表示，每个字段对应于与根据具体情况在标量步骤或在表步骤中处理的对象有关的字对（类型、长度）。

在步骤 28 中，转换器 10 在从步骤 27 中得到的且名称“DefTicket_

Configuration_X”对应于与步骤 26 中相同的模板名称的模板的开头插入包括字对（类型、长度）的元素定义。

在步骤 29 中，转换器 10 将步骤 40 中的模板定义常量的值分配给在步骤 64 中插入的元素定义的第一项。例如，值 2 表示所生成的字对串是 MIB 模板。转换器 10 将值 2 添加到“NumberOftheCurrentfield”以便包括两个字段，每个字段分别对应于在步骤 26 中插入的字对和在步骤 28 中插入的字对。当字对的每项包括例如 2 字节的字时，转换器 10 将之前获得的结果乘以 4，以便获得以字节表示的模板长度，随后将该模板长度分配给在步骤 64 中插入的字对的第二项。

在步骤 5 中，转换器 10 将因此获得的标准模板记录在大容量存储器 14 中。

命名归类模块 13 执行在下文中将参考图 7 进行描述的方法。

在步骤 70 中，模块 13 接收转换器 10 在标量步骤的步骤 7 中发送给模块 13 的对象的 OID、对象的类型、对象的子类型、对象的最小大小、对象的最大大小、“NumberOftheCurrenField”以及转换规则。

在步骤 71 中，模块 13 创建名为“Type_field_template”（类型_字段_模板）的变量并为其分配在步骤 70 中接收到的“NumberOftheCurrentField”。该步骤使得模块 13 发送默认类型，其为标准模板中的当前字段编号。

在步骤 72 中，模块 13 创建名为“length_field_template”（长度_字段_模板）的变量并为其分配在步骤 70 中接收到的对象的最大大小。该步骤使得模块 13 发送默认长度，其为针对当前对象一个字段所能具有的最大长度。

在步骤 73 中，模块 13 测试是否存在当前对象的 SMI 子类型。对该测试的肯定响应触发步骤 74，并且对该测试的否定响应触发步骤 75。

在步骤 74 中，模块 13 确定 SMI 子类型的最大长度并将该长度分配给变量“length_field_template”。

步骤 75 包括测试转换规则是否将模板字段类型与对应于所接收到的 OID 或其 SMI 类型或其 SMI 子类型的对象相关联。对该测试的肯定响应触发步骤 76,对该测试的否定响应触发步骤 78。

在步骤 76 中,模块 13 随后将该规则所提供的字段类型分配给变量 “type_field_template”。

在步骤 77 中,模块 13 将该字段类型的长度分配给名为 “length_field_template” 的变量。此外,如果存在将长度限制与对应于所接收到的 OID 的对象相关联的转换规则,则将该长度分配给变量 “length_field_template”。

在步骤 78 中,模块 13 向转换器 10 发送字对(类型、长度),该字对的第一个值为变量 “type_field_template” 的值,该字对的第二个值为变量 “length_field_template” 的值。

现在将基于参考图 8-图 10 的作为一个已知表 “bufferControlTable”(缓冲控制表)的示例性对象和参考图 11-图 13 的作为另一个已知表 “Aa5VccTable”的示例性对象来对正在执行的方法进行描述。

图 8 是 MIB 表的树表示,该表的附录 1 给出了 SMI 定义提取,在因特网站点 <http://www.ietf.org> 上可以获得的 RFC 2819 中的第 75-77 页中可以找到对 SMI 定义提取的更详细描述。

该排列的左边第一栏给出了所引用对象的节点编号(对象标识符的 OID),第二栏中给出其名称,第三栏中给出其类型并且第四栏中给出其长度。如果存在子类型,则在右边的最后一栏中给出子类型的长度。节点编号 “1.3.6.1.2.16” 中的第一个数字串给出了 MIB 11 在 MIB 的树配置中的位置。下一个数字 “8.1.” 表示对象 “BufferControlTable”在该 MIB 中的位置。数字 1.1-1.13 给出了该表的对象的树配置。

图 8 中示出的树或子树是转换器 10 基于附录 1 的 SMI 定义得到的存储器 12 中的内容的一个例子。

图 9 示出了通过在下文中将描述的方法生成的示例性标准模板。

当转换器 10 对存储器 12 中的参考编号为 1.3.6.1.2.1.16.8 的 MIB 11 的树进行遍历时，会在步骤 44 中遇到对象 “bufferControlTable”，其为存储器 12 中的参考编号为 1.3.6.1.2.1.16.8.1 的表。然后，转换器 10 执行表步骤，为此在步骤 30 中假定例如变量 “current_template_ID” 此时等于 302。

在步骤 31 中，将变量 “current_template_ID” 设定为 303。在步骤 51 中，转换器 10 创建图 9 中的初始为空的模板。为简单起见，在步骤 19 中假定 “Genertion_template_configuration” 等于 0。于是，在步骤 23 中，变量 “current_template_ID” 等于 304。在步骤 52 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti” 等于 0。

在步骤 55 中，转换器 10 遇到对象 “bufferControlIndex”（缓冲控制索引）并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti”（当前字段编号_定义票证_表_Ti）等于 0。

在步骤 49 中，该对象的 SMI 定义给出具有最大大小为 2 的子类型的类型 Integer32（整数 32）。在步骤 8 中，归类模块返回值为 (1,2) 的字对。在步骤 50 中，转换器 10 以两个二进制数字 121（每个数字两个字节）的形式将字对 (1,2) 添加到图 9 中的模板。

在步骤 56 和步骤 57 之后，在步骤 55 中，转换器 10 遇到对象 “bufferControlChannelIndex”（缓冲控制信道索引）并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti” 等于 2。

在步骤 49 中，该对象的 SMI 定义给出具有最大大小为 2 的子类型的类型 Integer32。在步骤 8 中，归类模块返回值为 (2,2) 的字对。在步骤 50 中，转换器 10 以两个二进制数字 122（每个数字两个字节）的形式将字对 (2,2) 添加到图 9 中的模板。

在步骤 56 之后，转换器 10 再一次遇到之前的两个对象并直接在

步骤 61 中转到下一个对象。

在步骤 58 中，转换器 10 遇到对象 “bufferControlfullStatus”，并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti”等于 3。

在步骤 49 中，该对象的 SMI 定义给出具有子类型大小 2 的类型 INTEGER。在步骤 8 中，归类模块返回值为 (3,2) 的字对。在步骤 50 中，转换器 10 以两个二进制字 123（每个数字两个字节）的形式将字对 (3,2) 添加到图 9 中的模板。

可以对图 8 的排列中的每个随后的对象重复上述过程，随后在步骤 50 中转换器 10 分别针对在步骤 58 中遇到的每个对象，分别以两个二进制字 124、125、126、127、128、129、130、131、132、133(每个字两个字节)的形式将字对 (4,2)、(5,2)、(6,2)、(7,4)、(8,4)、(9,4)、(10,4)、(11,4)、(12,32)、(13,4) 添加到图 9 中的模板。

于是，在步骤 60 中到达对象 “bufferControlStatus”（缓冲控制状态）之后，图 9 中的模板包括字对串 121-133。转换器 10 执行结束步骤。

在步骤 62 中，转换器 10 将一对二进制字 120 添加到模板的开头。在步骤 63 中，转换器 10 为该字对的第一个字分配值 302 并为其第二个字分配值 13。

在步骤 64 中，转换器 10 将一对二进制字 119 添加到模板的开头。在步骤 65 中，转换器 10 为该字对的第一个字分配值 2，并为其第二个字分配值 60。

在步骤 5 中，转换器 10 如同模板最终显示于图 9 中那样地记录该模板。

图 9 中的模板使得可以对其值包含于表 “bufferControlTable” 的行中的数据票证进行发送。

图 10 示出了一个示例性的数据票证。假定该表的行 34 给出了以

下的值:

- bufferControlIndex = 34
- bufferControlChannelIndex = 654
- bufferControlFullStatus = 2
- bufferControlFullAction = 542
- bufferControlCaptureSliceSize = 512
- bufferControlDownloadSliceSize = 200
- bufferControlDownloadOffset = 32
- bufferControlMaxOctetsRequested = 100000000
- bufferControlMaxOctetsGranted = 1000000
- bufferControlCapturedpackets =10000
- bufferControlTurnOnTime = 3454764364
- bufferControlOwner = acme
- bufferControlStatus = 3

在图 10 的相应数据票证中，2 字节的字 99 包含引用了图 9 的模板的值 302。2 字节的字 100 包含值 72,其代表具有两个字 99 和 100 的用图 9 的模板表示的 13 个数据字段的总长度，即数据票证的总长度。

此后的字段 101-106 中每个字段的长度分别由模板的字对 121-126 的第二个字给出，分别包含值 34、654、2、542、512 和 200。字段 107-111 中每个字段的长度分别由模板的字对 127-131 中的第二个字给出，包含值 32、100000000、1000000 和 3454764364。字段 112 的长度由字对 132 的第二个字给出，包含例如以 ASCII（美国信息交换标准码）编码的字符串“acme”。字段 113 的长度由字对 133 的第二个字给出，包含值 3。

对于数据票证的每个字段值，由模板的字对 121-133 中的每个字对的第一个字给出的类型实现该票证的接收以识别该值属于哪个对象。

应当注意，数据票证的使用使得节省带宽和简化计算成为可能。

其中忽略了 SNMP 消息的报头数据。图 10 中的数据票证只包含 4 个报头字节，前两个字节用于引用相应的模板，并且后两个字节用于表示其长度以便对由 SNMP Get 在此之前获得的表的行的内容进行发送。

为了通过 SNMP 获得相同的行，必须首先进行询问查询（interrogation query），然后进行响应查询。询问查询对于每一对象需要三组内容（对象标识符 + 索引值、对象标识符 + 长度、对象标识符 + 值），其中“对象标识符”是表字段的 OID，索引值是用于更精确地标识对象实例的索引“BufferControlIndex”的值，也就是说 $12 \times (11+2)$ 个字节，即 156 个字节。该响应还需要一个 30 个字节的值，即一共 186 个字节。

与 72 个字节的数据票证相比，SNMP 在消息方面并且尤其是在字节速率方面的容量需求更大，原因是其中产生了 4 倍的流量。的确，通过票证来发送数据需要先发送先前以图 9 为例描述过的模板（代表 60 个字节），与 SNMP 相比，将流量的生成削减到了只有一半用于第一个票证的发送。然而，同一个模板可以用于后续的票证而不需要再次发送，因此以 2:4 的比例减小了流量。

图 11 是 MIB ATM 的表的树表示，图 11 的附录 2 给出了 SMI 定义提取，在因特网站点 <http://www.ietf.org> 上可以获得的 RFC 1695 的第 61-63 页中可以找到对 SMI 定义提取的详细描述。

该排列的左边第一栏给出了所引用对象的节点编号（对象标识符的 OID），第二栏中给出其名称，第三栏中给出其类型并且第四栏中给出其长度。如果存在子类型，则在右边的最后一栏中给出该对象的子类型的长度。MIB 的一般树配置中的节点编号也称作 OID（对象标识符）。该节点编号 1.3.6.1.2.1.37 的第一个数字串给出了 MIB ATM 在 MIB 的树配置中的位置。下一个数字 8.1 表示对象“aa15VccTable”在该 MIB ATM 中的位置。数字 1.1-1.5 给出了该表的对象的树配置。

在附录 2 的 SMI 定义中，表的索引包括三个对象“ifIndex”、“aa15VccVpi”和“aa15VccVci”。后面两个对象属于所考虑的表。

第一个对象属于另一个适于该物理接口的 MIB 的 “ifTable”表，该 MIB 名为 MIBII，并位于数字 1.3.6.1.2.1 表示的 MIB 的树配置中。数字 2.2 表示该表位于 MIBII 中，数字 1.1 表示作为该表的一部分的对象 “ifIndex”位于 “ifTable”表的树配置中。

基于附录 2 中的 SMI 定义以及标识 3 中的 MIB II，转换器 10 在存储器 5 中构造命名树。该树是基于图 11 中的 OID 的数字而定义的。

图 12 示出了通过在下文中描述的方法生成的示例性标准模板。

当转换器 10 对存储器 12 中的包括 MIB II 和 MIB ATM 的树的子树 1.3.6.1.2.1.37 进行遍历时，会在步骤 44 中遇到对象 “aa15VccTable”，其为存储器 12 中的参考编号为 1.3.6.1.2.1.37.1.12 的表。然后，转换器 10 执行表步骤，为此在步骤 30 中假定例如变量 “current_template_ID”此时等于 303。

在步骤 31 中，将变量 “current_template_ID”设定为 304。在步骤 51 中，转换器 10 创建图 12 中的初始为空的模板。为简单起见，在步骤 19 中假定 “Genertion_template_configuration”等于 0。于是，在步骤 23 中，变量 “current_template_ID”等于 305。在步骤 52 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti”等于 0。

在步骤 55 中，转换器 10 遇到索引 “ifIndex”，即索引的第一项，并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti”等于 1。

在步骤 49 中，该对象的 SMI 定义给出最大大小为 4 的类型 INTEGER。在步骤 8 中，归类模块返回值为 (1,4) 的字对。在步骤 50 中，转换器 10 以两个二进制字 81（每个数字两个字节）的形式将字对 (1,4) 添加到图 12 中的模板。

在步骤 56 和步骤 57 之后，在步骤 55 中，转换器 10 遇到对象 “aa15VccVpi”，即索引的第二项，并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti”等于 2。

在步骤 49 中，该对象的 SMI 定义给出具有最大大小为 2 的子类型的类型 `AtmVpIdentifier`。在步骤 8 中，归类模块返回值为 (2,2) 的字对。在步骤 50 中，转换器 10 以两个二进制字 82（每个数字两个字节）的形式将字对 (2,2) 添加到图 12 中的模板。

在步骤 56 和步骤 57 之后，在步骤 55 中，转换器 10 遇到索引的第三项，即对象 “aa15VccVci”，并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti” 等于 3。

在步骤 49 中，该对象的 SMI 定义给出具有最大大小为 2 的子类型的类型 `AtmVcIdentifier`。在步骤 8 中，归类模块返回值为 (3,2) 的字对。在步骤 50 中，转换器 10 以两个二进制字 83（每个数字两个字节）的形式将字对 (3,2) 添加到图 12 中的模板。

在步骤 56 之后，转换器 10 再一次遇到之前的三个对象并且每次都经过步骤 58、59、60 和 61 直接在步骤 61 转到下一个对象。

在步骤 58 中，转换器 10 遇到对象 “aa15VccCrcErrors” 并针对其执行标量步骤。

在步骤 48 中，变量 “NumberOftheCurrentField_DefTicket_Table_Ti” 等于 4。

在步骤 49 中，该对象的 SMI 定义给出最大大小为 4 的类型 `Counter32`（计数 32）。在步骤 8 中，归类模块返回值为 (4,4) 的字对。在步骤 50 中，转换器 10 以两个二进制字 84（每个数字两个字节）的形式将字对 (4,4) 添加到图 12 中的模板。

可以对图 11 的排列中的每个随后的对象重复上述过程，随后在步骤 50 中转换器 10 分别针对在步骤 58 中遇到的每个对象，分别以两个二进制字 85、86（每个字两个字节）的形式将字对 (5,4)、(6,4) 添加到图 12 中的模板。

于是，在步骤 60 中到达对象 “aa15VccOverSizedSDUs” 之后，图 12 中的模板包括一系列字对 81-86。转换器 10 执行结束步骤。

在步骤 62 中，转换器 10 将一对二进制字 80 添加到模板的开头。

在步骤 63 中,转换器 10 为该字对的第一个字分配值 303 并为其第二个字分配值 6。

在步骤 64 中,转换器 10 将一对二进制字 79 添加到模板的开头。在步骤 65 中,转换器 10 为该字对的第一个字分配值 2,并为其第二个字分配值 32。

在步骤 5 中,转换器 10 如同模板最终显示于图 12 中那样地记录该模板。

图 12 中的模板使得可以对其值包含于表“aal5VccTable”的行中的数据票证进行发送。该票证通常一起分组为准备发送到收集器的票证块。ifIndex 12

图 13 示出了一个示例性的数据票证。假定该表的行描述通过接口 12 行进的 ATM 电路的业务的质量并且标识为对应于 Vpi 100 的 Vci 1000 的虚拟电路,则给出以下值:

- ifIndex = 12
- aal5VccVpi = 100
- aal5VccVci = 1000
- aal5VccCrcErrors = 15432
- aal5VccSarTimeOuts = 456
- aal5VccOverSizedSDUs = 567

在图 13 的相应数据票证中,2 字节的字 199 包含引用了图 12 的模板的值 302。2 字节的字 200 包含值 24,其代表具有两个字 199 和 200 的用图 9 中的模板表示的 6 个数据字段的总长度,即数据票证的总长度。

此后的字段 201-206 中每个字段的长度分别由模板的字对 81-86 的第二个字给出,分别包含值 12、100、1000、15432、456 和 567。

对于数据票证的每个字段值,由模板的字对 81-86 中的每个字对的第一个字给出的类型实现该票证的接收以确定该值属于哪个对象。

在此应当再次注意,数据票证的使用使得节省带宽和简化计算成为可能。

参考图 14,测量系统 91 包括用于测量 IP 网络的性能的探测器 89 和探测器 89 所连接的系统控制器 90。

SNMP 代理 93 以代理模式实现 MIB II IPPM REPORTING MIB。在 www.ietf.org/html.characters/ippm-character.html 中对其进行了定义。代理 93 位于代理服务器 92 中,以便为网络管理系统 94 提供用于测量系统 91 的标准管理接口,网络管理系统 94(NMS 代表网络管理站)随后使用 SNMP 常规协议来查询正在进行的测量及测量结果。

应当理解,已知的管理接口 IPPM REPORTING MIB 定义了多种表,其中在此作为例子采用了表 `ippmNetworkMeasureTable` (ippm 网络测量表)和 `ippmHistoryTable` (ippm 历史表),以说明本方法的工业应用,即本发明的主题。名为 `ippmNetworkTable` (ippm 网络表)的网络管理表 95 携带有由位于控制器 90 中的管理程序 87 在探测器之间适当地设定的测量的定义。管理程序 87 收集探测器 89 发起的测量的结果。名为 `ippmHistoryTable` 的历史表 96 存储从管理程序接收到的测量结果。代理 93 包括这两个表的副本等。

为了能够使用之前描述的方法,系统 91 与图 1 中的表示一致。在图 14 中可以再次看到转换器 10 和 MIB 规范 11。在此,其中转换器 10 的功能是生成用于表 `ippmNetworkTable` 和 `ippmHistoryTable` 的模板。

系统 91 还包括用于输出模板和数据票证的模块 88。代理服务器 92 包括用于接收模板和数据票证的模块 18。为了进行通信,模块 18 和模块 88 使用特别适于在模块之间交换模板和数据票证的名为 IPFIX 的协议。

代理服务器 92 还包括转换器 10 和 MIB 11 的实例或有权访问转换器 10 和 MIB 11 的实例。因此,当系统 91 通过模块 88 将模板分发给代理服务器 92 时,代理服务器 92 通过模块 18 接收该模板,并可以为接收数据票证做准备,该数据票证包含系统继模板之后发出的描述和测量结果。

接着,测量系统 91 使用 IPFIX 协议与信息流一起将正在进行的

测量以及这些测量的结果传送给代理服务器 92。然后，站 94 通过 SNMP 以常规的方式接收代理 93 的测量数据。

图 15 给出了通过之前描述的方法在系统 91 中获得的 `ippmHistory` 表的示例性模板。假设实际上不同字段中的值是以二进制形式表示的，则在此以十进制表示这些值以便实现其快速读取。

字对 139 的第一个字和第二个字分别包含表示数据结构为模板的值 2 以及以字节表示模板长度的值 36。

字对 140 的第一个字和第二个字分别包含表示模板适于历史表的值 304 和表示模板的数据字段量的值 7。

适于对象 “`ippmHistoryMeasureOwner`”（`ippm` 历史测量所有者）的字对 141 的第一个字和第二个字分别包含表示字段类型的值 1 和以字节表示字段长度的值 32。

适于对象 “`ippmHistoryMeasureIndex`”（`ippm` 历史测量所有者）的字对 142 的第一个字和第二个字分别包含表示字段类型的值 2 和以字节表示字段长度的值 4。

适于对象 “`ippmHistoryMetricIndex`”（`ippm` 历史度量索引）的字对 143 的第一个字和第二个字分别包含表示字段类型的值 3 和以字节表示字段长度的值 4。

适于对象 “`ippmHistoryIndex`”（`ippm` 历史索引）的字对 144 的第一个字和第二个字分别包含表示字段类型的值 4 和以字节表示字段长度的值 4。

适于对象 “`ippmHistorySequence`”（`ippm` 历史序列）的字对 145 的第一个字和第二个字分别包含表示字段类型的值 5 和以字节表示字段长度的值 4。

适于对象 “`ippmHistoryTimestamp`”（`ippm` 历史时戳）的字对 146 的第一个字和第二个字分别包含表示字段类型的值 6 和以字节表示字段长度的值 8。

适于对象 “`ippmHistoryValue`”（`ippm` 历史值）的字对 147 的第一个字和第二个字分别包含表示字段类型的值 7 和以字节表示字段

长度的值 4。

图 16 给出了在系统 91 中获得以便通过图 15 的模板读取的 `ippmHistoryTable` 表的示例性数据票证。如同其他的模板或数据票证图形的情况一样，行的宽度代表 4 个字节，然而行的高度不与超过 8 个字节的字段的字节数目成正比，以便使该图形的表示可以保持合理的页数。

第一个字段 149 包含两个字节，其值 304 是相应模板中的字对 140 的第一个字的值。

第二个字段 150 包含两个字节，其值 64 表示数据票证的字节量。

适于对象 “`ippmHistoryMeasureOwner`” 的字段 151 包含以二进制编码的字符串 FTRD。

适于对象 “`ippmHistoryMeasureIndex`” 的字段 152 包含对测量进行索引的值 5 和对简单的外线延迟 (outward delay) 进行索引的值 6。

适于对象 “`ippmHistoryMetricIndex`” 的字段 153 包含对所测量的度量 (metric) 的类型进行索引的值 6，在这种情况下值 6 对应于无方向性的延迟。

适于对象 “`ippmHistoryIndex`” 的字段 154 包含值 123，以表示在该票证中发送的测量结果为第 123 个测量结果。

适于对象 “`ippmHistorySequence`” 的字段 155 包含值 1057582058，以表示在 2002 年 10 月 14 日 9 点 54 分 18 秒执行测量。

适于对象 “`ippmHistoryTimestamp`” 的字段 156 包含值 4578845678，其代表标记碎片共享的时间 (time stamping fractional share)。

适于对象 “`ippmHistoryValue`” 的字段 156 包含值 567，其为延迟测量的值，单位由字段 152 给出。

图 17 给出了通过之前描述的方法在系统 91 中获得的表 `ippmNetworkTable` (`ippm` 网络表) 的示例性模板。假设实际上不同字段中的值是以二进制表示的，则在此再一次以十进制表示这些值以便实现其快速读取。

字对 159 的第一个字和第二个字分别包含表示数据结构是 MIB 模板的值 2 和以字节表示该模板长度的值 116。

字对 160 的第一个字和第二个字分别包含表示该模板适于网络测量的表的值 302 和表示由该模板格式化的数据字段的数量的值 27。

适于对象 “ippmNetworkMeasureOwner” (ippm 网络测量所有者) 的字对 161 的第一个字和第二个字分别包含表示字段类型的值 1 和以字节表示该字段长度的值 32。

适于对象 “ippmNetworkMeasureIndex” (ippm 网络测量索引) 的字对 162 的第一个字和第二个字分别包含表示字段类型的值 2 和以字节表示该字段长度的值 2。

适于对象 “ippmNetworkMetricName” (ippm 网络度量名称) 的字对 163 的第一个字和第二个字分别包含表示字段类型的值 3 和以字节表示该字段长度的值 256。

适于对象 “ippmNetworkMeasureMetrics” (ippm 网络测量度量) 的字对 164 的第一个字和第二个字分别包含表示字段类型的值 4 和以字节表示该字段长度的值 8。

适于对象 “ippmNetworkMeasureBeginTime” (ippm 网络测量开始时间) 的字对 165 的第一个字和第二个字分别包含表示字段类型的值 5 和以字节表示该字段长度的值 4。

适于对象 “ippmNetworkMeasureCollectionRateUnit” (ippm 网络测量收集速率单位) 的字对 166 的第一个字和第二个字分别包含表示字段类型的值 6 和以字节表示该字段长度的值 4。

适于对象 “ippmNetworkMeasureCollectionRate” (ippm 网络测量收集速率) 的字对 167 的第一个字和第二个字分别包含表示字段类型的值 7 和以字节表示该字段长度的值 4。

适于对象 “ippmNetworkMeasureDurationUnit” (ippm 网络测量持续单位) 的字对 168 的第一个字和第二个字分别包含表示字段类型的值 8 和以字节表示该字段长度的值 4。

适于对象 “ippmNetworkMeasureDuration” (ippm) 的字对 169

的第一个字和第二个字分别包含表示字段类型的值 9 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureHistorySize”（ippm 网络测量历史大小）的字对 170 的第一个字和第二个字分别包含表示字段类型的值 10 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureFailureMgmtMode”（ippm 网络测量故障管理模式）的字对 171 的第一个字和第二个字分别包含表示字段类型的值 11 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureResultMgmt”（ippm 网络测量结果管理）的字对 172 的第一个字和第二个字分别包含表示字段类型的值 12 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureSrcTypeP”（ippm 网络测量 SrcTypeP）的字对 173 的第一个字和第二个字分别包含表示字段类型的值 13 和以字节表示该字段长度的值 512。

适于对象“ippmNetworkMeasureSrc”（ippm 网络测量 Src）的字对 174 的第一个字和第二个字分别包含表示字段类型的值 14 和以字节表示该字段长度的值 512。

适于对象“ippmNetworkMeasureDstTypeP”（ippm 网络测量 DstTypeP）的字对 175 的第一个字和第二个字分别包含表示字段类型的值 15 和以字节表示该字段长度的值 512。

适于对象“ippmNetworkMeasureDst”（ippm 网络测量 Dst）的字对 176 的第一个字和第二个字分别包含表示字段类型的值 16 和以字节表示该字段长度的值 512。

适于对象“ippmNetworkMeasureTransmitMode”（ippm 网络测量发送模式）的字对 177 的第一个字和第二个字分别包含表示字段类型的值 17 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureTransmitPacketRateUnit”（ippm 网络测量发送分组速率单位）的字对 178 的第一个字和第二个字分别包含表示字段类型的值 18 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureTransmitPacketRate”（ippm 网络测量发送分组速率）的字对 179 的第一个字和第二个字分别包含表示字段类型的值 19 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureDeviationOrBurstsize”（ippm 网络测量偏差突发大小）的字对 180 的第一个字和第二个字分别包含表示字段类型的值 20 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureMedianOrInterBurstsize”（ippm 网络测量突发中或突发间大小）的字对 181 的第一个字和第二个字分别包含表示字段类型的值 21 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureLossTimeout”（ippm 网络测量丢失超时）的字对 182 的第一个字和第二个字分别包含表示字段类型的值 22 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureL3PacketSize”（ippm 网络测量第 3 层分组大小）的字对 183 的第一个字和第二个字分别包含表示字段类型的值 23 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureDataPattern”（ippm 网络测量数据模式）的字对 184 的第一个字和第二个字分别包含表示字段类型的值 24 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureMap”（ippm 网络测量映射）的字对 185 的第一个字和第二个字分别包含表示字段类型的值 25 和以字节表示该字段长度的值 256。

适于对象“ippmNetworkMeasureSingletons”（ippm 网络测量个体）的字对 186 的第一个字和第二个字分别包含表示字段类型的值 26 和以字节表示该字段长度的值 4。

适于对象“ippmNetworkMeasureOperState”（ippm 网络测量 OperState）的字对 187 的第一个字和第二个字分别包含表示字段类型的值 27 和以字节表示该字段长度的值 4。

图 18 给出了在系统 91 中获得以便通过图 17 的模板进行读取的表 ippmNetworkTable 的示例性数据票证。

第一个字段 259 包含值 302,其为相应模板中的字对 159 的第一个字的值。

第二个字段 260 包含指示以字节表示的票证长度的值 2676。

适于对象“ippmNetworkMeasureOwner”的字段 261 包含以二进制编码的字符串 FTRD, 以表示测量的执行者。

适于对象“ippmNetworkMeasureIndex”的字段 262 包含表示测量编号的值 5。

适于对象“ippmNetworkMetricName”的字段 263 包含以二进制编码的字符串“One-way-delay between Paris and Lannion”, 以纯文本形式表示测量指向的内容。

适于对象“ippmNetworkMeasureMetrics”的字段 264 包含表示测量类型的值 6。

适于对象“ippmNetworkMeasureBeginTime”的字段 265 包含值 1034582058, 表示该测量开始于 2003 年 9 月 14 日 9 点 54 分 18 秒。

适于对象“ippmNetworkMeasureCollectionRateUnit”的字段 266 包含代表标记碎片共享时间的值 0。

适于对象“ippmNetworkMeasureCollectionRate”的字段 267 包含表示采样速率的值 10。

适于对象“ippmNetworkMeasureDurationUnit”的字段 268 包含值 6,表示测量持续时间的单位是秒。

适于对象“ippmNetworkMeasureDuration”的字段 269 包含值 120, 表示测量持续了 120 秒。

适于对象“ippmNetworkMeasureHistorySize”的字段 270 包含值 1000,表示测量历史的大小。

适于对象“ippmNetworkMeasureFailureMgmtMode”)的字段 271 包含值 1,表示自动模式。

适于对象“ippmNetworkMeasureResultsMgmt”的字段 272 包含值 1,表示与自动循环之间传送结果。

适于对象“ippmNetworkMeasureSrcTypeP”的字段 273 包含字符

串“IP UDP”,表示构成测量主题的开始从巴黎开始的通信所采用的传输协议。

适于对象“ippmNetworkMeasureSrc”的字段 274 包含值 80.168.0.1 3456,表示通信的源地址,在此是巴黎。

适于对象“ippmNetworkMeasureDstTypeP”的字段 275 包含字符串“IP UDP”,表示构成测量主题的目的地为拉尼翁(Lannion)的通信所采用的传输协议。

适于对象“ippmNetworkMeasureDst”的字段 276 包含值 180.168.0.1 6543,表示通信的目的地地址,在此是拉尼翁。

适于对象“ippmNetworkMeasureTransmitMode”的字段 277 包含值 1,表示传输模式是定期传输。

适于对象“ippmNetworkMeasureTransmitPacketRateUnit”(ippm)的字段 278 包含值 6,表示所发送的 60 字节的分组速率单位是每秒。

适于对象“ippmNetworkMeasureTransmitPacketRate”的字段 279 包含值 100,表示发送速率是每秒 100 个分组。

适于对象“ippmNetworkMeasureDeviationOrBurstsize”的字段 280 包含值 0,表示偏差为零。

适于对象“ippmNetworkMeasureMedianOrInterBurstsize”的字段 281 包含值 0,表示突发大小为零。

适于对象“ippmNetworkMeasureLossTimeout”的字段 282 包含值 15,表示延迟为 15 秒,当该延迟到期时认为分组已丢失。

适于对象“ippmNetworkMeasureL3PacketSize”的字段 283 包含值 64,表示形成测量主题的通信中发送分组的大小为 64 字节。

适于对象“ippmNetworkMeasureDatePattern”的字段 284 包含值 FFFF,表示测量数据的格式。

适于对象“ippmNetworkMeasureMap”的字段 285 包含字符串“internal network”,以明文形式表示网络类型。

适于对象“ippmNetworkMeasureSingletons”的字段 286 包含默认的值 0。

适于对象“ippmNetworkMeasureOperState”的字段 287 包含默认的值 0。

本领域普通技术人员将比较容易理解，基于刚才已经描述的数据票证的两个例子，与在这一级别上使用 SNMP 协议所必需的带宽相比，节省了在系统 91 与服务器 92 之间传送测量数据所必需的带宽。

根据本发明在上述系统中实现的方法是特别有用的，原因是这种方法使得有可能基于 MIB 或 PIB 自动地生成用于任意类型的数据结构模板，使人们从在没有本发明时将不得不人工地为每种特定的数据结构（在本领域中可能的数据结构的量是相当可观的）定义模板这种烦琐的任务中解脱出来。

附录 1

bufferControlTable OBJECT-TYPE

SYNTAX SEQUENCE OF {

bufferControlIndex Integer32,
bufferControlChannelIndex Integer32,
bufferControlFullStatus INTEGER,
bufferControlFullAction INTEGER,
bufferControlCaptureSliceSize Integer32,
bufferControlDownloadSliceSizeInteger32,
bufferControlDownloadOffset Integer32,
bufferControlMaxOctetsRequested Integer32,
bufferControlMaxOctetsGranted Integer32,
bufferControlCapturedPackets Integer32,
bufferControlTurnOnTime TimeTicks,
bufferControlOwner OwnerString,
bufferControlStatus EntryStatus

}

INDEX {bufferControlIndex}

::={capture 1}

附录 2

aal5VccTable OBJECT-TYPE

SYNTAX SEQUENCE OF {

 aal5VccVpi INTEGER,

 aal5VccVci INTEGER,

 aal5VccCrcErrors Counter32,

 aal5VccSarTimeOuts Counter32,

 aal5VccOverSizedSDUs Counter32

}

INDEX {ifIndex, aal5VccVpi, aal5VccVci}

::= {atmMIBObjects 1}

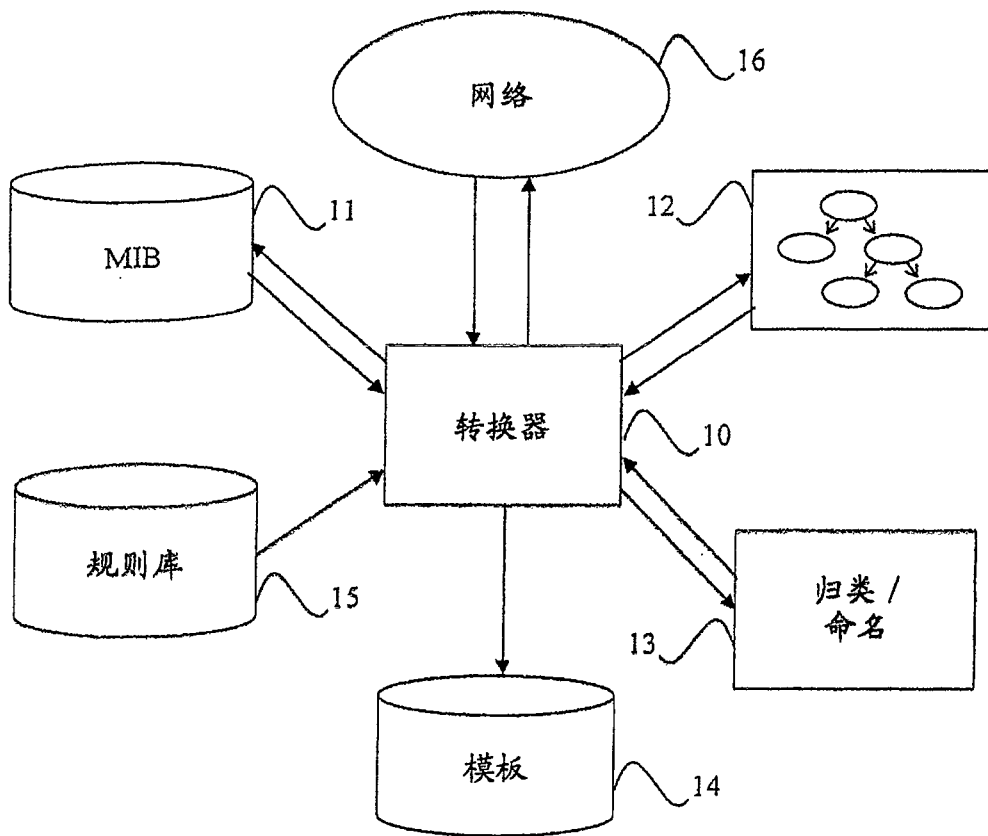


图 1

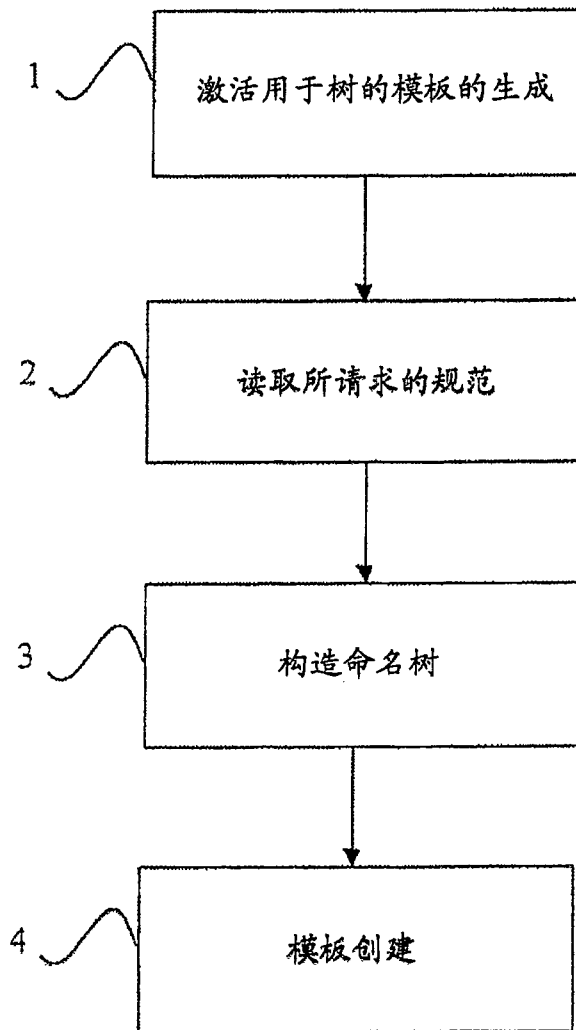


图 2

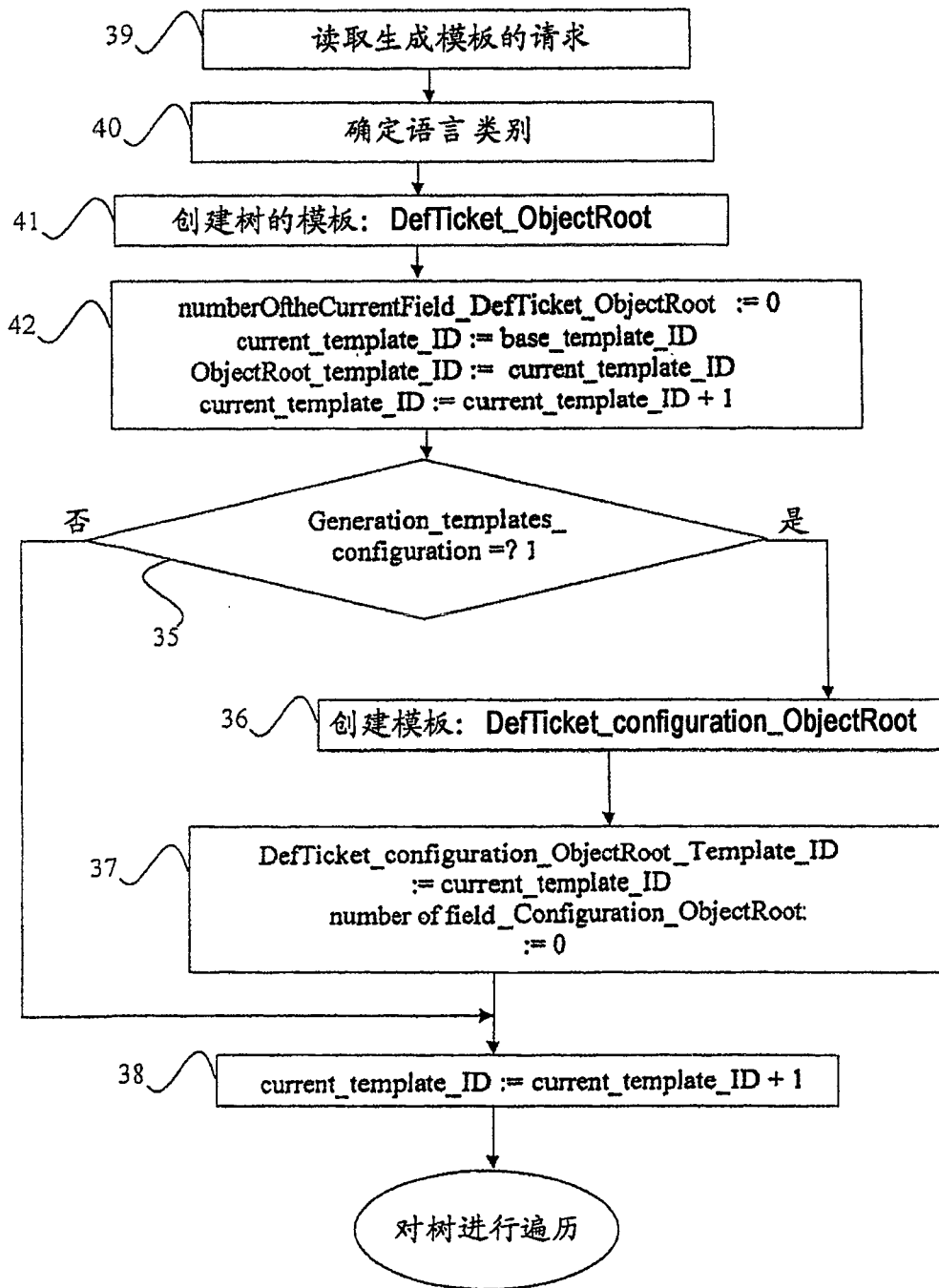


图 3a

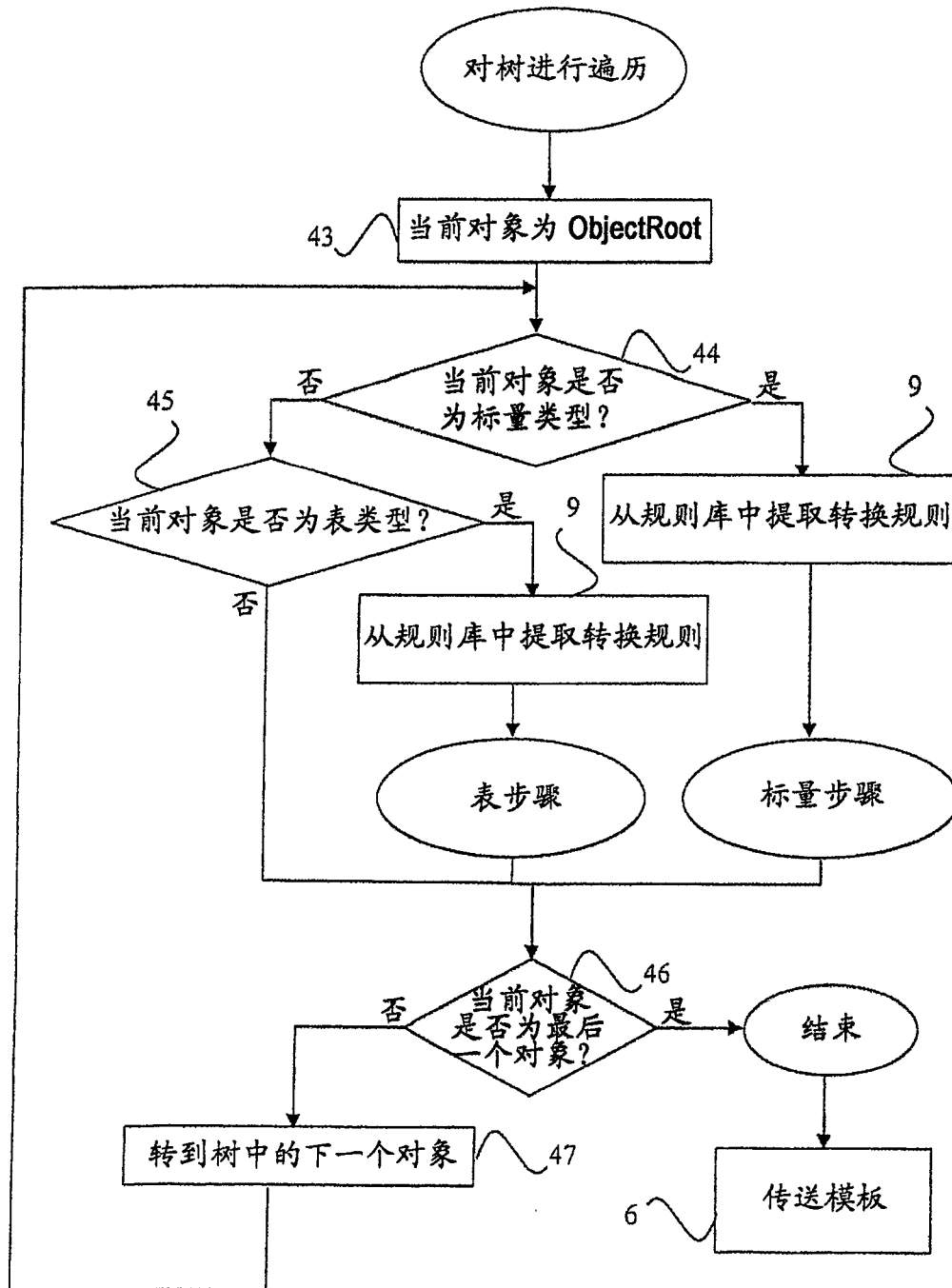


图 3b

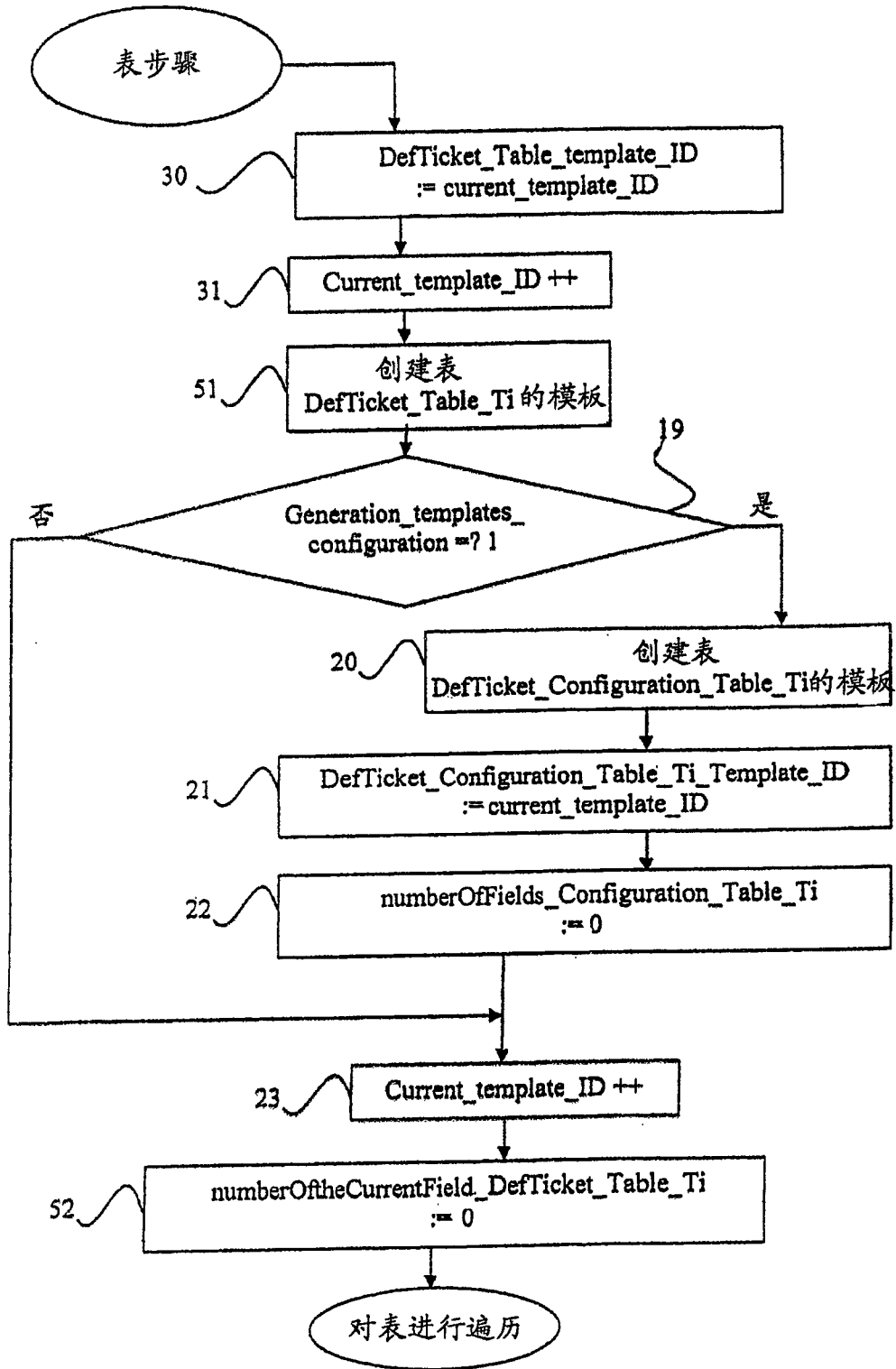


图 4a

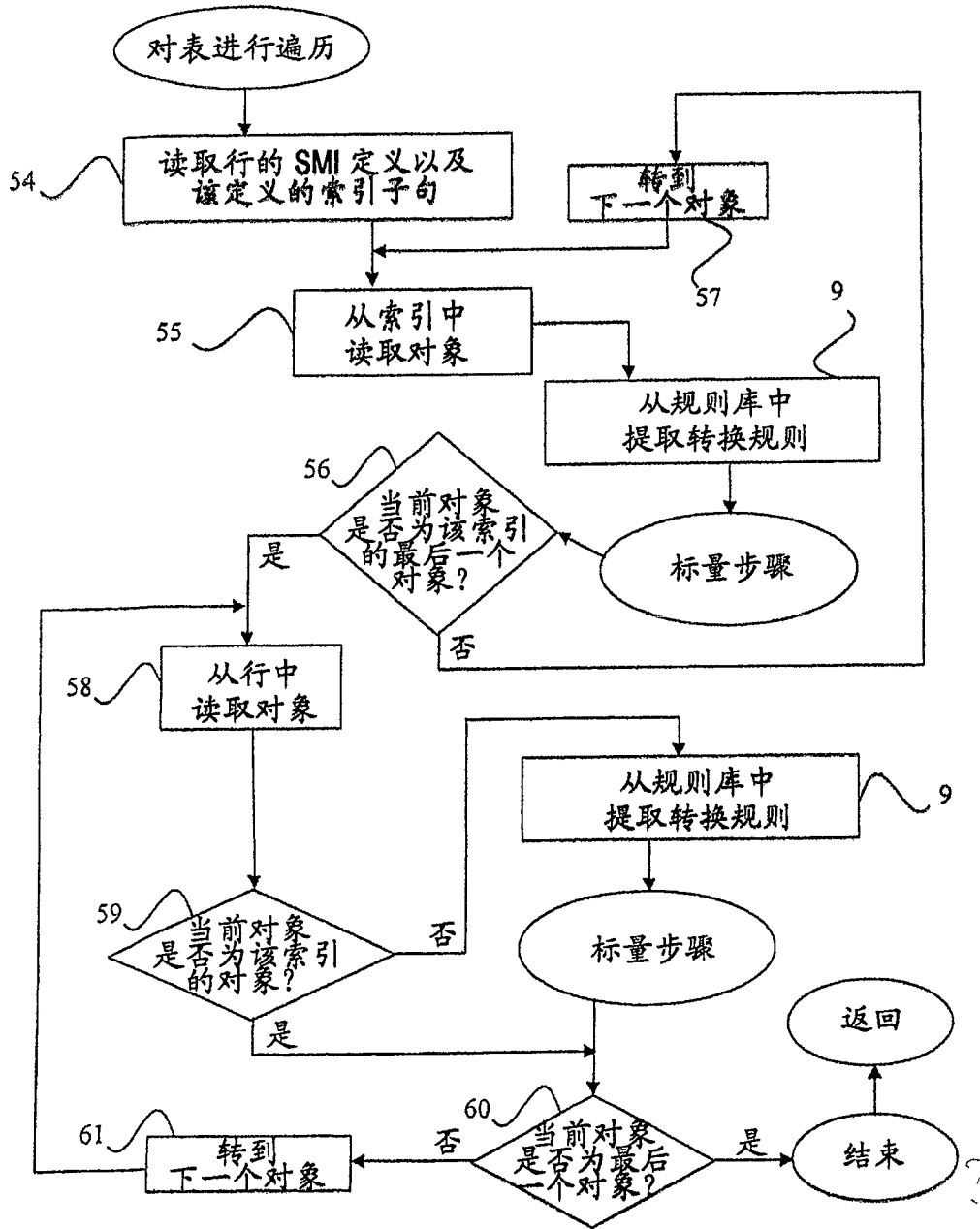


图 4b

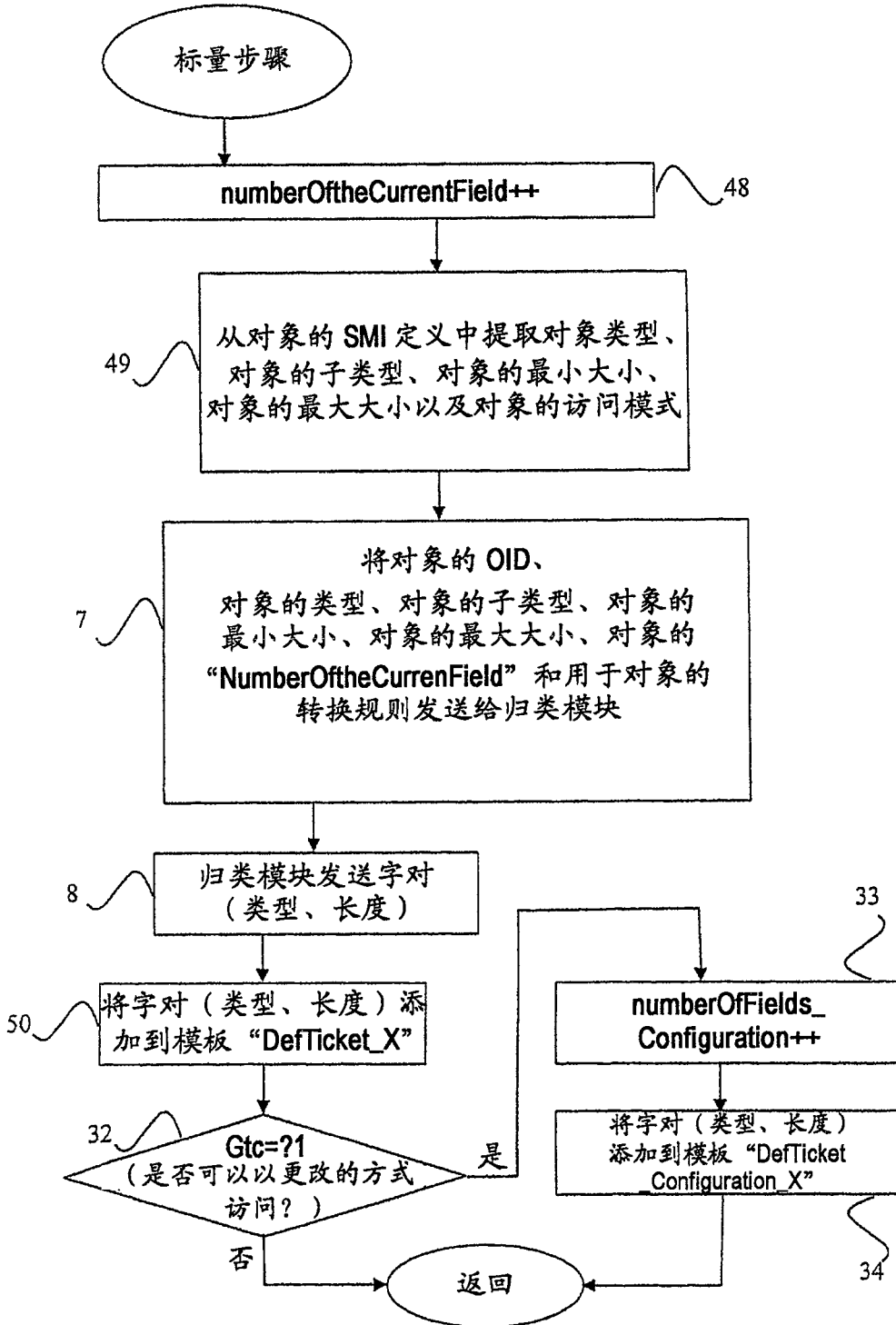


图 5

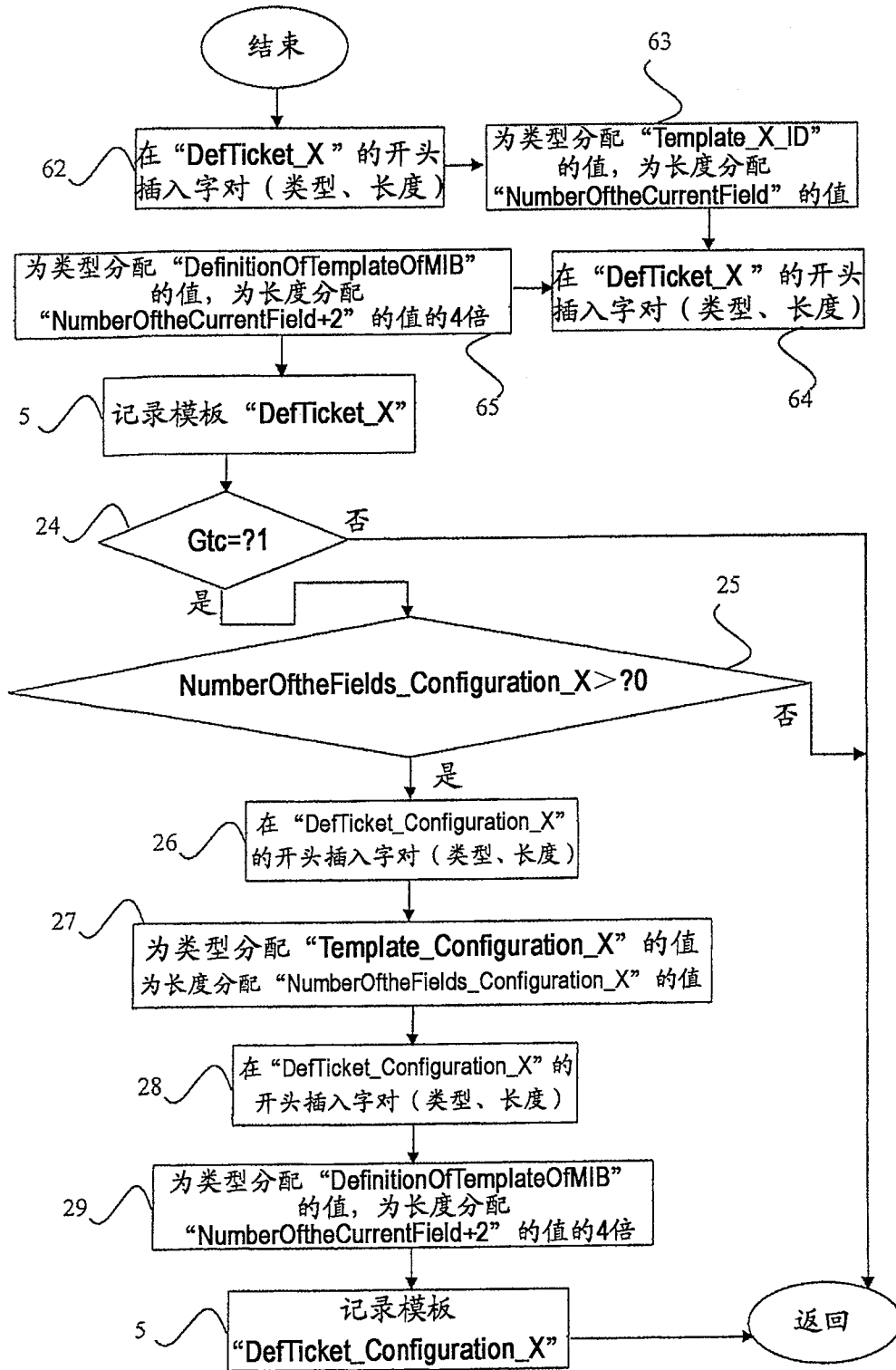


图 6

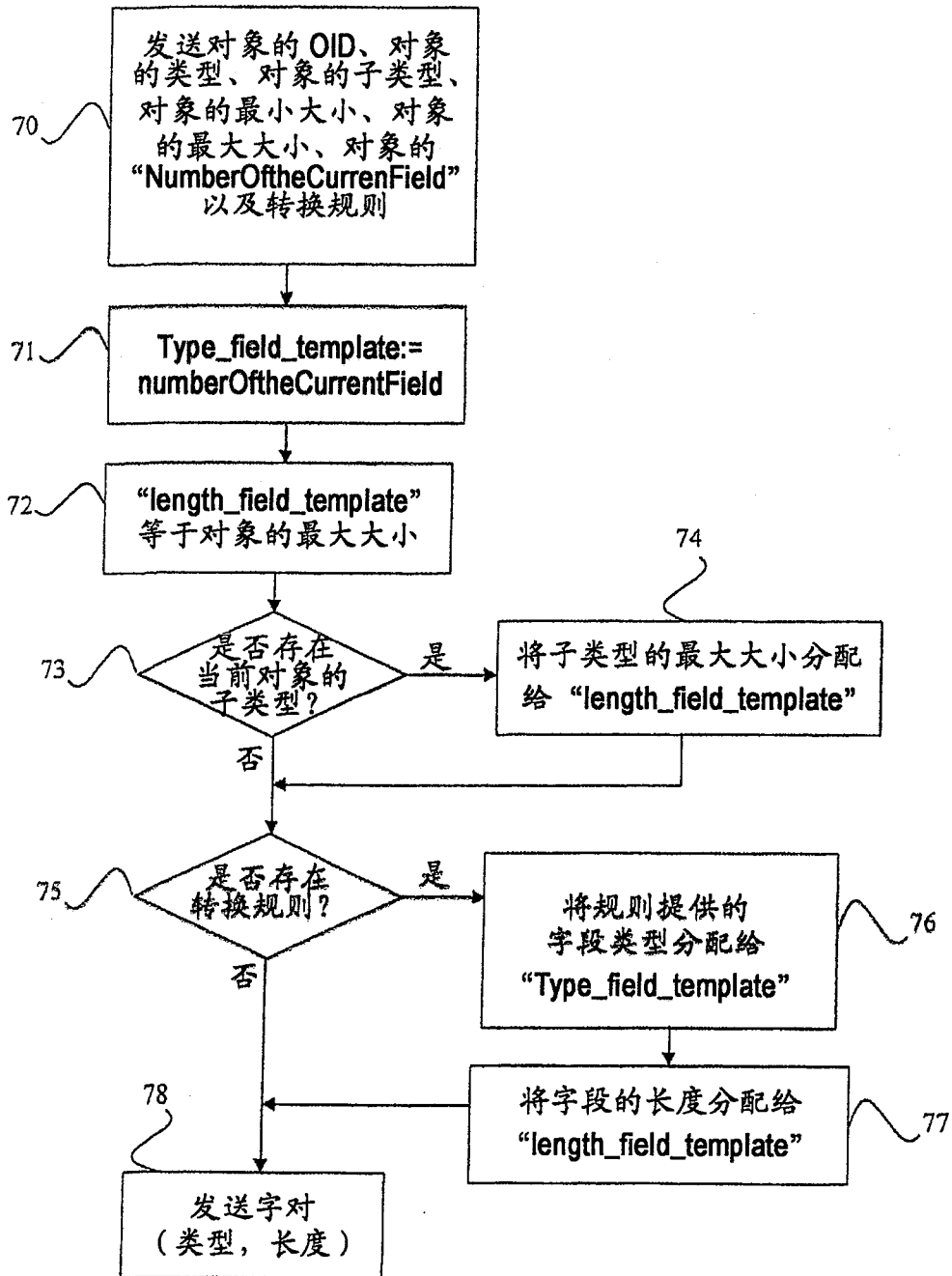


图 7

OID	名称	类型	L 类型	L 子 类型
1.3.6.1.2.1.16.8.1	bufferControlTable	SequenceOf	-	-
1.3.6.1.2.1.16.8.1.1.1	bufferControlIndex	Integer32	1-4	1-2
1.3.6.1.2.1.16.8.1.1.2	bufferControlChannelIndex	Integer32	1-4	1-2
1.3.6.1.2.1.16.8.1.1.3	bufferControlFullStatus	INTEGER	1-4	2
1.3.6.1.2.1.16.8.1.1.4	bufferControlFullAction	INTEGER	1-4	2
1.3.6.1.2.1.16.8.1.1.5	bufferControlCaptureSliceSize	Integer32	1-4	1-2
1.3.6.1.2.1.16.8.1.1.6	bufferControlDownloadSliceSize	Integer32	1-4	1-2
1.3.6.1.2.1.16.8.1.1.7	bufferControlDownloadOffset	Integer32	1-4	-
1.3.6.1.2.1.16.8.1.1.8	bufferControlMaxOctetsRequested	Integer32	1-4	-
1.3.6.1.2.1.16.8.1.1.9	bufferControlMaxOctetsGranted	Integer32	1-4	-
1.3.6.1.2.1.16.8.1.1.10	bufferControlCapturedPackets	Integer32	1-4	-
1.3.6.1.2.1.16.8.1.1.11	bufferControlTurnOnTime	TimeTicks	1-4	-
1.3.6.1.2.1.16.8.1.1.12	bufferControlOwner	OwnerString	1-32	-
1.3.6.1.2.1.16.8.1.1.13	bufferControlStatus	EntryStatus	4	-

图 8

119	00000000 00000010	00000000 00111100
120	00000001 01011100	00000000 00001101
121	00000000 00000001	00000000 00000010
122	00000000 00000010	00000000 00000010
123	00000000 00000011	00000000 00000010
124	00000000 00000100	00000000 00000010
125	00000000 00000101	00000000 00000010
126	00000000 00000110	00000000 00000010
127	00000000 00000111	00000000 00000100
128	00000000 00001000	00000000 00000100
129	00000000 00001001	00000000 00000100
130	00000000 00001010	00000000 00000100
131	00000000 00001011	00000000 00000100
132	00000000 00001100	00000000 00100000
133	00000000 00001101	00000000 00000100

图 9

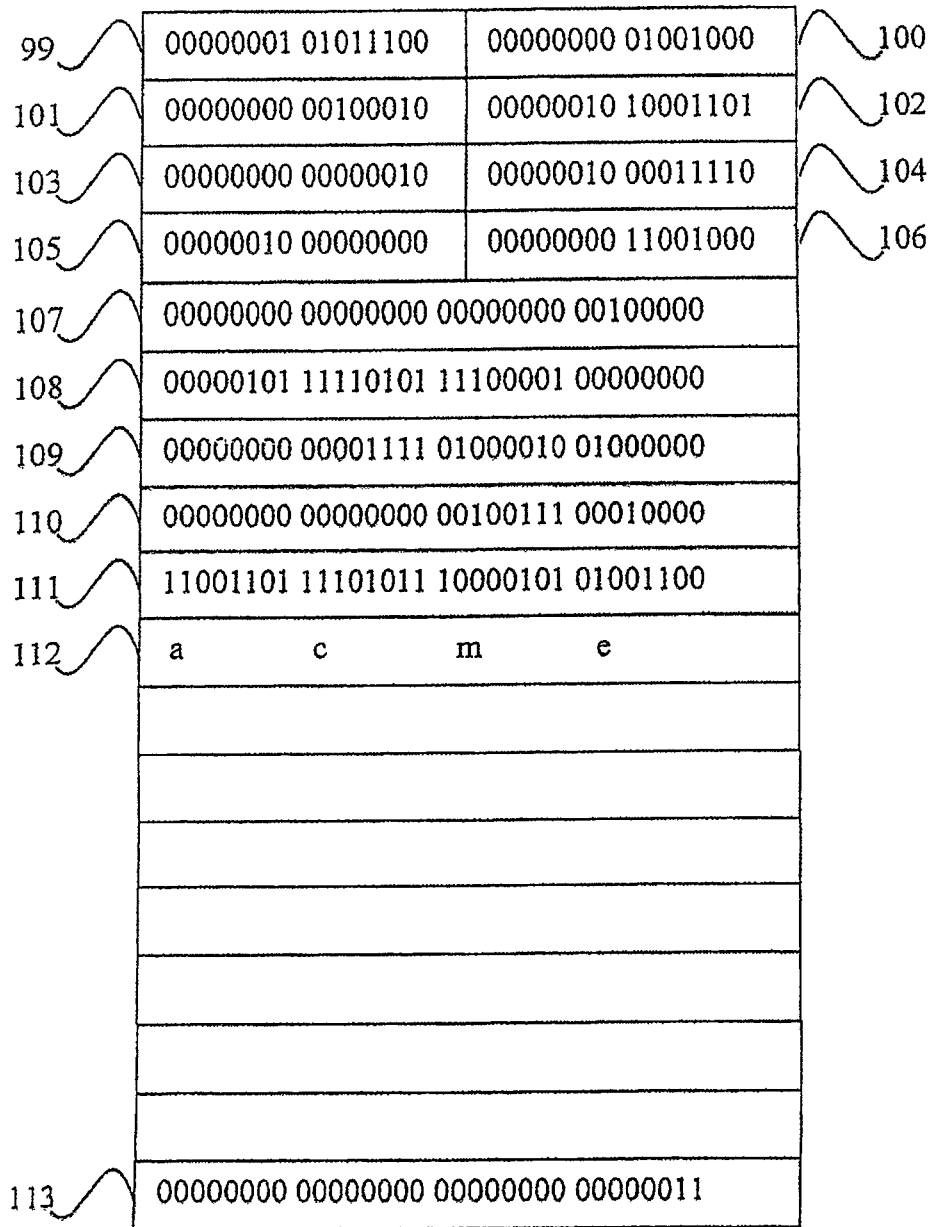


图 10

OID	名称	类型	L 类型	L 子 类型
1.3.6.1.2.1.37.1.12	Aal5VccTable	SequenceOf	-	-
1.3.6.1.2.1.2.1.1.1	ifindex	INTEGER	1-4	-
1.3.6.1.2.1.37.1.12.1.1	Aal5VccVpi	AtmVpIdentifier	1-4	1-2
1.3.6.1.2.1.37.1.12.1.2	Aal5VccVci	AtmVpIdentifier	1-4	1-2
1.3.6.1.2.1.37.1.12.1.3	Aal5VccCrcErrors	Counter32	1-4	-
1.3.6.1.2.1.37.1.12.1.4	Aal5VccTimeOuts	Counter32	1-4	-
1.3.6.1.2.1.37.1.12.1.5	Aal5VccOverSizedSDUs	Counter32	1-4	-

图 11

79	00000000 00000000	00000000 00100000
80	00000001 00101111	00000000 00000110
81	00000000 00000001	00000000 00000100
82	00000000 00000010	00000000 00000010
83	00000000 00000011	00000000 00000010
84	00000000 00000100	00000000 00000100
85	00000000 00000101	00000000 00000100
86	00000000 00000110	00000000 00000100

图 12

199	00000001 01011100	00000000 00011000	200
201	00000000 00000000	00000000 00001100	
202	00000000 01100100	00000011 11101000	203
204	00000000 00000000	00111100 01001000	
205	00000000 00000000	00000001 11001000	
206	00000000 00000000	00000010 00110111	

图 13

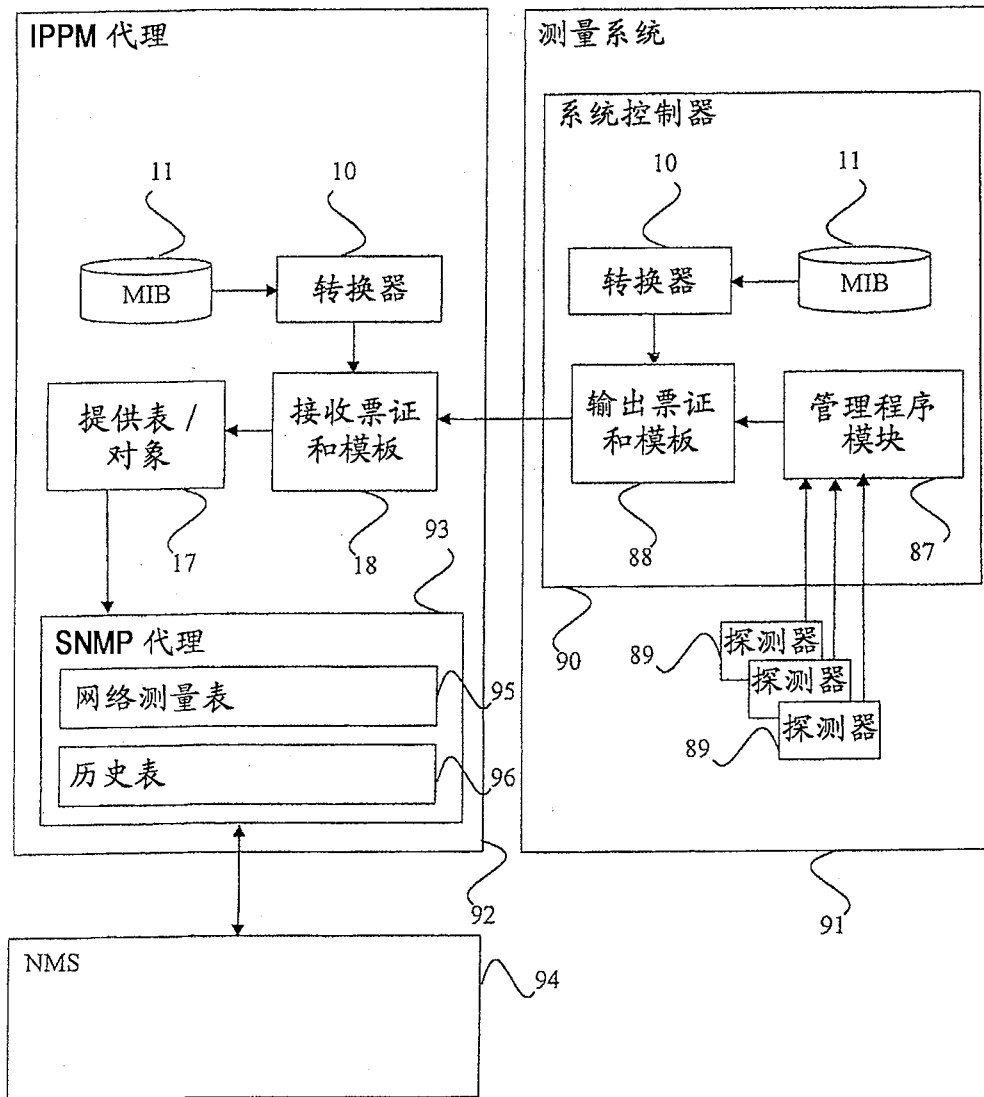


图 14

139	2	36
140	304	7
141	1	32
142	2	4
143	3	4
144	4	4
145	5	4
146	6	8
147	7	4

图 15

149	304	64	150
151	FTRD		
152	5		
153	6		
154	123		
155	1057582058		
156	4578845678		
157	567		

图 16

159	2	116	302	27	160
161	1	32	2	2	162
163	3	256	4	2	164
165	5	8	6	4	166
167	7	4	8	4	168
169	9	4	10	4	170
171	11	4	12	4	172
173	13	512	14	512	174
175	15	512	16	512	176
177	17	4	18	4	178
179	19	4	20	4	180
181	21	4	22	4	182
183	23	4	24	4	184
185	25	256	26	4	186
187	27	4			

图 17

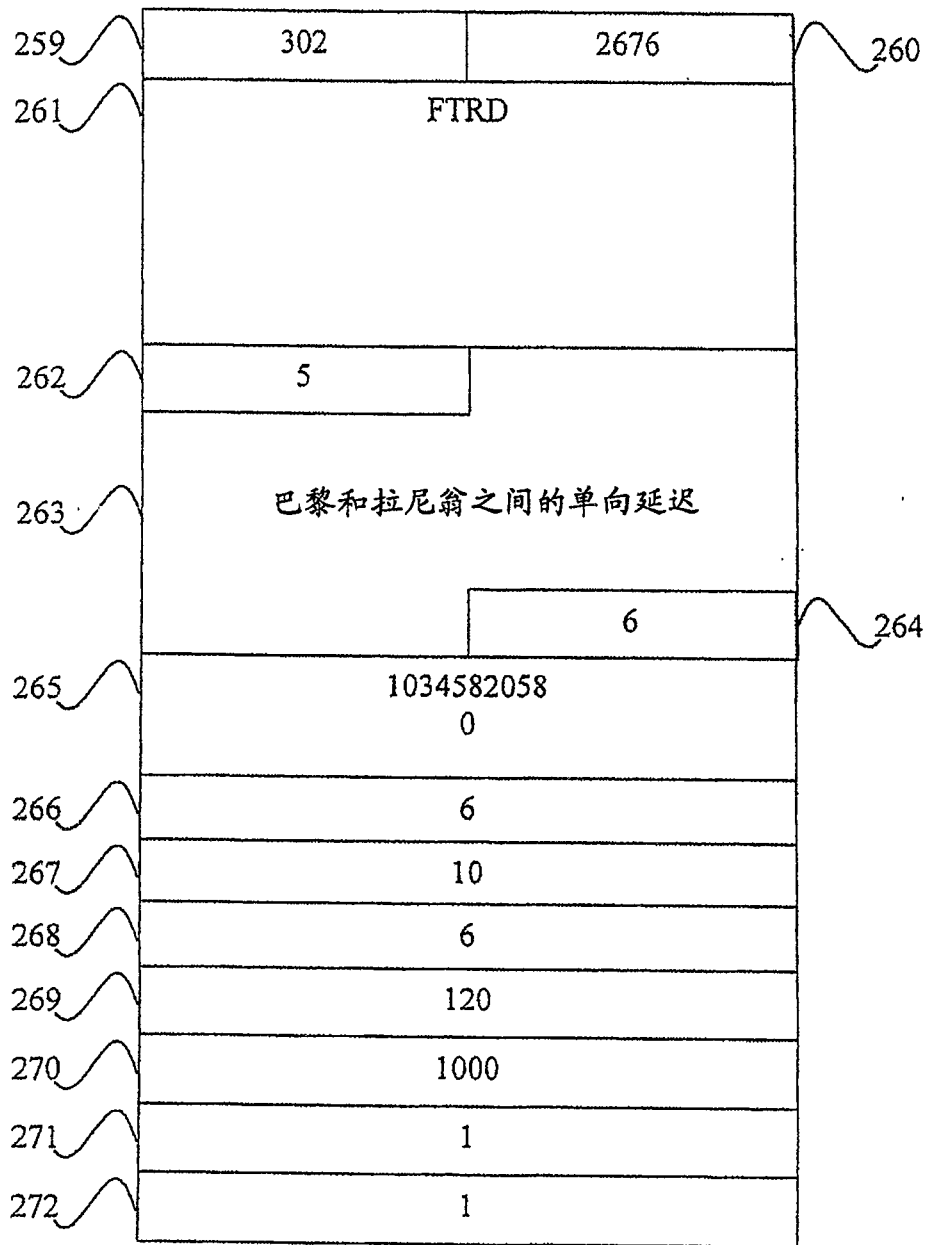


图 18a

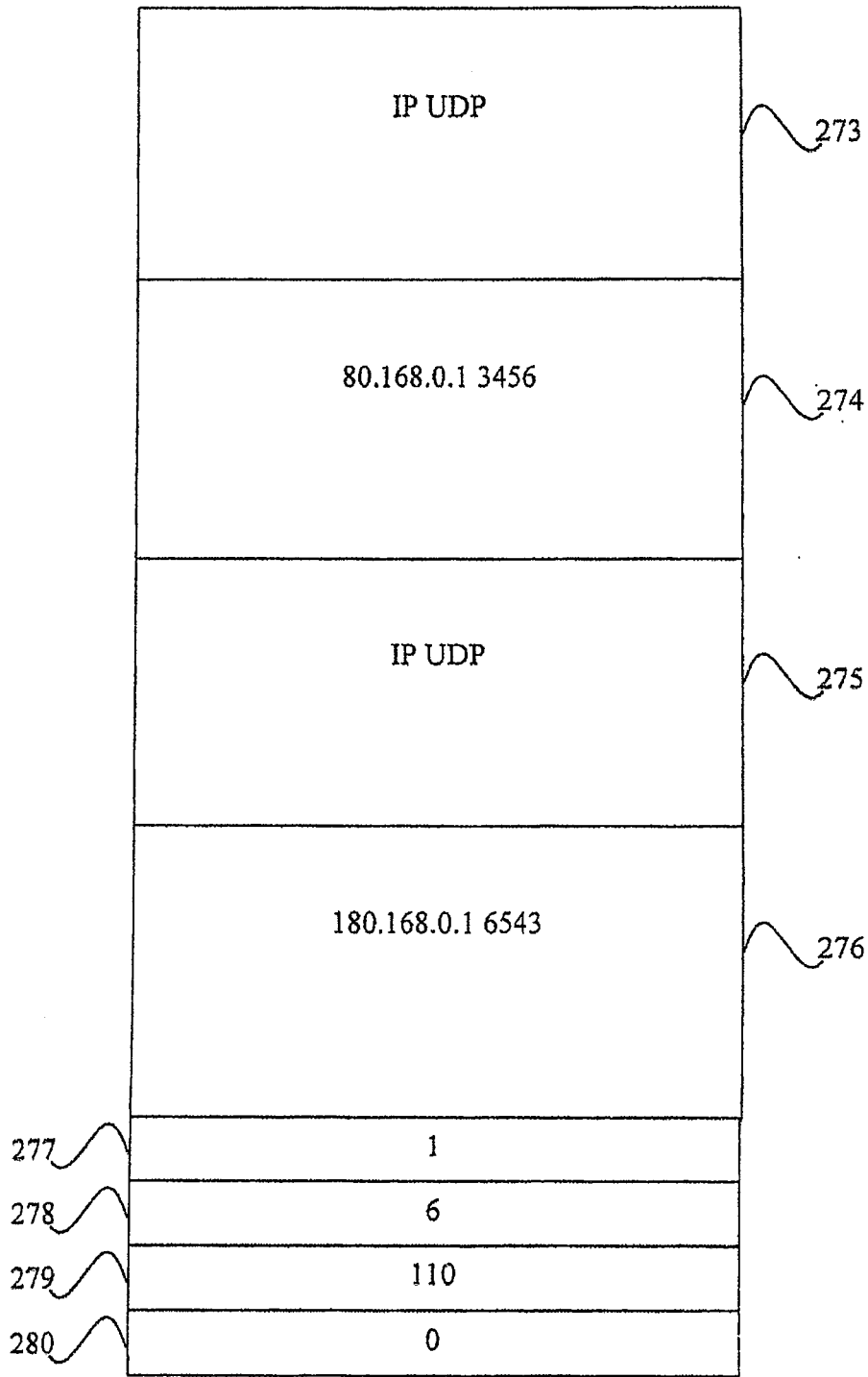


图 18b

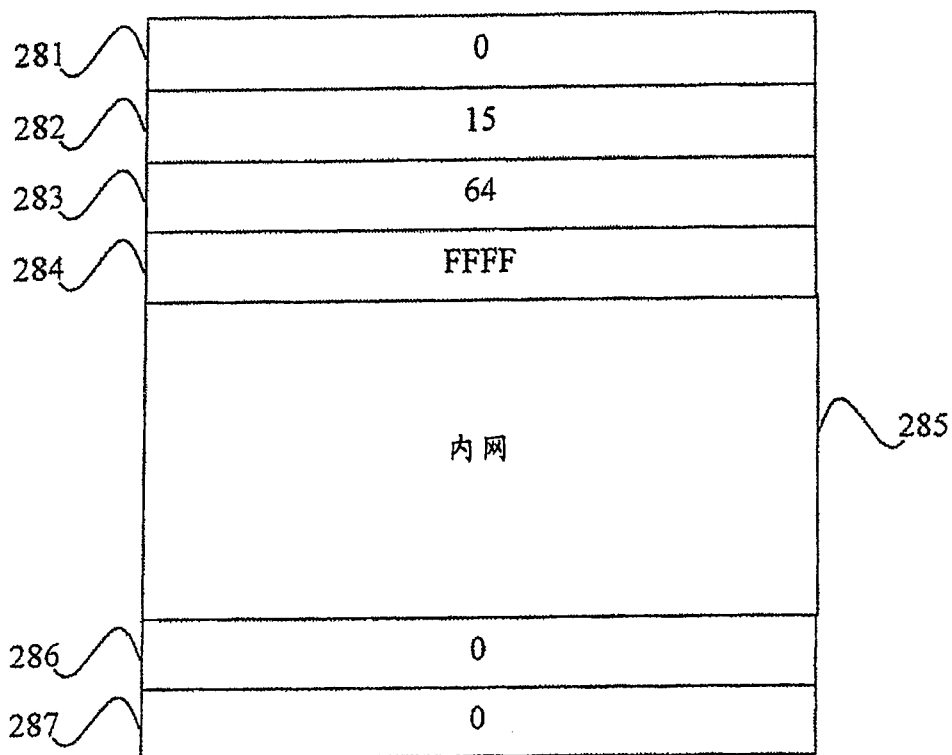


图 18c