(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 1685312 B
(45)授权公告日 2010.05.26

(21)申请号 03822498.4

(22)申请日 2003.07.10

(30)优先权数据
10/199,963 2002.07.19 US

(85)PCT申请进入国家阶段日
2005.03.21

(86)PCT申请的申请数据
PCT/US2003/021862 2003.07.10

(87)PCT申请的公布数据
WO2004/010294 EN 2004.01.29

(73)专利权人 开放创新网络有限责任公司
地址 美国纽约州

(72)发明人 克里斯托弗·T·英格索尔
杰亚拉姆·R·卡西
亚历山大·霍姆斯 迈克尔·克拉克
阿肖克·阿莱蒂
萨蒂什·B·K·塞纳西
海伦·S·尤安

(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 郭定辉 黄小临

(51)Int.Cl.
G06F 9/44(2006.01)

(56)对比文件
US 6393442 B1,2002.05.21, 全文.
US 6538673 B1,2003.03.25, 全文.
CN 1286447 A,2001.03.07, 全文.

审查员 谢志远

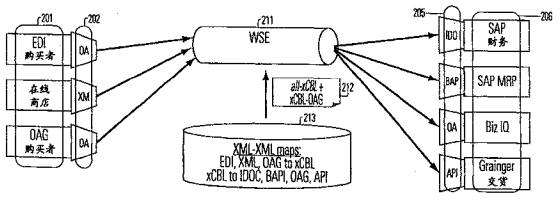
权利要求书 2 页 说明书 13 页 附图 11 页

(54)发明名称

陈列和选择可应用于文档族的语义变换的计算机辅助方法

(57)摘要

本发明涉及用于在商家或应用之间交换的文件的注册表驱动变换的系统和方法。本发明尤其涉及用于使用一个或多个公共可访问注册表来在不同接口之间变换电子商务文档的系统和协议，其中文档最好是XML文档。在权利要求书、说明书和附图中描述了本发明的各特定方面。



CN 1685312 B

1. 一种陈列可应用于文档族的语义变换的计算机辅助方法，包括：

保持文档族注册表数据结构；以及

通过提供将要远程执行的逻辑，来响应用于语义变换的请求，

其中，对于特定的文档族，该文档族注册表数据结构包括：

由库、文档标识符、发布版本和模式类型标识的文档族成员；

文档族成员之间的语义变换，包括远程执行用来实现语义变换的逻辑；以及

语义变换与源和目标文档族成员的关联。

2. 如权利要求 1 所述的方法，还包括遍历文档族成员之间的语义变换的关联，以及标识从源文档族成员到目标文档族成员的一个或多个变换的一个或多个顺序。

3. 如权利要求 2 所述的方法，其中：

保持语义变换还包括保持变换成功分值；及

响应请求还包括计算变换的顺序的综合成功分值。

4. 如权利要求 3 所述的方法，其中变换成功分值对应于从源文档的字段逐字地翻译成目标文档的字段的片断。

5. 如权利要求 3 所述的方法，其中变换成功分值对应于在目标文档的字段中逐字地找出的源文档的字段中的文本片断。

6. 如权利要求 1 所述的方法，其中：

保持文档族注册表数据结构还包括：

保持对应于在文件交换中的一个或多个参与者的、将要应用的变换的特定规则；及

通过提供将要远程执行的逻辑的响应通过引用与文档族注册表数据结构分离地保持的、专门保持的变换来实现。

7. 如权利要求 1 所述的方法，其中：

保持文档族注册表数据结构还包括：

保持对应于在文件交换中的一个或多个参与者的、将要应用的变换的特定规则；及

输出还包括引用安全地保持在文档族注册表数据结构中的、专门保持的变换。

8. 一种选择可应用于文档族的语义变换的计算机辅助方法，包括：

保持文档族注册表数据结构；以及

通过将文档族注册表数据结构作为定向图遍历并且输出通过遍历确定的一个或多个变换顺序，来响应用于将文件从源语义版本向目标语义版本转换的语义变换的请求，

其中，对于特定的文档族，该文档族注册表数据结构包括：

由库、文档标识符、发布版本和模式类型标识的文档族成员；

文档族成员之间的语义变换，包括远程执行来实现语义变换的逻辑；以及

语义变换与源和目标文档族成员的关联。

9. 如权利要求 8 所述的方法，其中

保持语义变换还包括保持变换成功分值；及

遍历文档族注册表数据结构还包括计算变换顺序的综合成功分值。

10. 如权利要求 9 所述的方法，其中变换成功分值对应于从源文档的字段逐字地翻译成目标文档的字段的片断。

11. 如权利要求 9 所述的方法，其中变换成功分值对应于在目标文档的字段中逐字地

找出的源文档的字段中的文本片断。

12. 如权利要求 8 所述的方法，其中：

保持文档族注册表数据结构还包括保持对应于在文件交换中的一个或多个参与者的、将要应用的变换的特定规则；及

输出还包括引用与文档族注册表数据结构分开保持的、专门保持的变换。

13. 如权利要求 8 所述的方法，其中：

输出还包括引用与文档族注册表数据结构分开保持的、专门保持的变换。

陈列和选择可应用于文档族的语义变换的计算机辅助方法

[0001] 版权声明

[0002] 本专利文件的公开部分包含接受版权保护的材料。版权所有者对出现在专利或商标局专利文件或记录中的任何专利文档或专利公开的任何一个的复制副本其没有异议，但在其他任何情况下保留所有版权权利。

技术领域

[0003] 本发明涉及用于在商家或应用之间交换的文档的注册表驱动语义变换的系统和方法。本发明尤其涉及使用一个或多个公共可访问注册表来在不同接口之间变换电子商务文档的系统和协议，所述文档最好是 XML 文档。

背景技术

[0004] 商家对商家 (B2B) 和应用对应用 (A2A) 电子商务正在取代用于电子数据交换的在先协议。由于使用 B2B 和 A2A 系统的商家致力于改善其效率，涌现出许多不兼容的平台和互相竞争的标准。共识的需要是将文档从一个系统转换到另一个系统。

[0005] 由于必须被应用于创建内嵌标记的 XML 的严格的语法规则使得计算机程序解释和处理起来相对简单，因此 XML 已经成为广泛使用的数据类型。例如，以 XML 写成的定购单可以由定购单项 (entry) 应用软件处理，所述定购单项应用了解怎样读取用于描述购买的物品和购买人等信息的标记注释。作为文档编码模式逐渐接受的 XML，已经导致企业适配器工具 (EAI) 厂商开发了许多合法应用的 XML 化应用程序接口。

[0006] EAI 厂商在一个应用接一个应用的基础上，将一个系统与下一个系统桥接起来。在设计期间，通过设计实现互用性。系统或应用之间的连接是静态的。应用的新版本的实施需要修改静态连接。应用之间的路由通常是在企业内部。基于点对点开发集成 (integration) 逻辑。语义逻辑编码成 EAI 例程。语义逻辑和句法逻辑混合在编码中。文档的发送方或源负责保证它们所发送的部分正是通告目标或接受方所接收的部分。与要求完整的兼容性相反，没有模拟接口兼容程度的概念。由于其要求所有的客户机更新到服务接口的最新版本并且同时更新这些接口，所以很难实现极好的兼容性。很难重新使用变换部分。没有提供可公共访问的资料库来抓取个人变换首选或支持基于用户简档的变换。EAI 厂商途径很难使变换例程从一对系统或应用改编到另一对系统或应用，并且需要大量的成本。

[0007] 图 1 图解了 EAI 厂商途径，其应用于将新的 (incoming) 定购单输入四个不同的系统中的供应厂商处理。在该图中，从三个源 101，即电子数据交换 (EDI) 购买者、在线商店的客户和遵从开放式应用组商家对象文档 (OAGBOD) 的购买者，产生新定购单。每个源具有用于产生输入到 EAI 基础结构 103 中定购单的本地接口 102。文件的格式可以包括 EDI、XML 和 OAG。四个目标系统 106 包括 SAP 财务系统、SAP MRP 系统、Biz IQ 系统和 Granger（美国农民协进会会员）发货系统。由这些目标系统接受的文档本地格式 105 包括 IDOC、BAPI、OAG 和自定义应用程序接口 (API)。为了连接源和目标，需要克服句法和语义两者的差别。点对点适配器 104 一对一对地将源文档变换为目标文档。甚至在利用相同句法的系统之间

的文档变换（诸如 OAG 到 OAG 变换）也包含不同的语义，因此需要适配器。当源或目标系统更新（例如 Oracle 财务系统替换为 SAP 财务系统或安装升级的发货系统）时，需要写入新的适配器。很可能 EAI 基础结构保留旧的和新的适配器。当系统更新时，越来越多的适配器需要接受修改或替换。单一变换引擎管理变换处理并且提供变换资源。

[0008] 因此，存在设计一种能够公共管理不同接口之间的文档变换的方法和结构的机会，从而提供实时互用性和分布式变换执行。

发明内容

[0009] 本发明涉及用于在商家或应用之间交换的文档的注册表驱动变换的系统和方法。本发明尤其涉及使用一个或多个公共可访问注册表来在不同接口之间变换电子商务文档的系统和协议，所述文档最好是 XML 文档。本发明的具体各方面描述在权利要求书、说明书和附图中。

[0010] 根据本发明的一个方面，一种陈列可应用于文档族的语义变换的计算机辅助方法包括：保持文档族注册表数据结构；以及通过提供将要远程执行的逻辑，响应用于语义变换的请求，其中，对于特定的文档族，该文档族注册表数据结构包括：由库、文档标识符、发布版本和模式类型标识的文档族成员；文档族成员之间的语义变换，包括远程执行来实现语义变换的逻辑；以及语义变换与源和目标文档族成员的关联。

[0011] 根据本发明的一个方面，一种选择可应用于文档族的语义变换的计算机辅助方法，包括：保持文档族注册表数据结构；以及通过将文档族注册表数据结构作为定向图遍历并且输出通过遍历确定的一个或多个变换顺序，响应用于将文件从源语义版本向目标语义版本转换的语义变换的请求，其中，对于特定的文档族，该文档族注册表数据结构包括：由库、文档标识符、发布版本和模式类型标识的文档族成员；文档族成员之间的语义变换，包括远程执行来实现语义变换的逻辑；以及语义变换与源和目标文档族成员的关联。

附图说明

[0012] 图 1 是现有技术的、使用点对点连接的变换处理的高级方框图；

[0013] 图 2 是使用网络服务引擎的变换处理的高级方框图；

[0014] 图 3 是文档族和版本的分级图；

[0015] 图 4 是文档库、名称空间、模式和文档族的方框图；

[0016] 图 5 是文档族成员和成员间变换的网络图；

[0017] 图 6 和 7 是变换顺序和用于实现变换顺序的逻辑组件的表；

[0018] 图 8 是包括文档库、名称空间、文档类型和模式、文档族和变换的类图；

[0019] 图 9 是实现变换的软件组件的高级方框图；

[0020] 图 10 是对应的行动图；

[0021] 图 11 图解变换顺序；

[0022] 图 12 和 13 是描述确定将源文档转换为目标文档的变换的优选顺序的流程图；以及

[0023] 图 14 和 15 图解支持文档族的管理和查找变换的搜索的用户接口。

具体实施方式

[0024] 下面参照附图进行详细描述。描述优选实施例来说明本发明而不是限制权利要求书所定义的范围。本领域普通人员可以认识到，有许多与下面的描述等同的变换。

[0025] 图 2 描述以四个不同系统为目的地的新定购单的供应者处理。新定购单源于三个源 201，即 EDI 购买者、在线商店客户和遵从 OAG 的购买者。由这三个源 210 利用的本地格式包括 EDI、XML 和 OAG。四个目标系统 206 包括 IDOC、BAPI、OAG 和自定义 API。在该系统中，网络服务引擎 211 使用公共句法库 (base) 执行语义变换。例如，EDI 和 OAG 文件转换为 XML，作为公共句法库。从 XML 到 XML 的变换处理源和目标文档之间的语义差异。XML 文档可以重新转换为本地格式 (如 EDI、OAG、IDOC 或 BAPI)。从 XML 或到 XML 的句法变换可以作为网络服务引擎 211 的一部分来处理，或由与源 201 和目标 206 相关联的接口或适配器 202、205 来处理。

[0026] 网络服务引擎 211 访问包括使用公共句法库的变换的各种变换 213。这些变换可以重新使用。可以调用一个以上的变换来将文档从源语义转换到目标语义。可以考虑利用公共语义库来进行变换，例如，将新文档变换为被广泛了解的、诸如用于电子商务文档 212 的 xCBL 模式的文件模式。通过将新文档变换为公共语义库，降低点对点变换的需要。可以链接并且可以重新使用变换。变换可以是同构或同态的。即，变换不需要是完全可逆的。通常将通过先验方法或通过在变换前或后比较源和目标语义来评价变换，以便估计变换所导致的损失程度。变换成功分值可以用来选择从源到目标语义的变换的替换顺序。通过在目标文档中包含一个或多个用于从源文档中获取没有很好地翻译的信息的字段，可以补偿变换所导致的损失。用户可以查看这些字段，以便与源相关的用户、目标或中间服务提供商可以响应计算机实现的变换服务的不完整性。或者，利用对已经不完全地变换或怀疑没有完全变换的的源文档的各部分的引用，可以将源文档和目标文档发送到目标。这些引用可以是部分目标文档或分离的文件，如错误文件。它们可以是字符串、指针或某些其他引用格式。引用可以提供给非完全变换的信息所属于的目标文档的一个或多个部分。对目标文档的引用可以是目标文档的单元 (element) 或小节 (subsection)，或者是在单元或子部分内的具体位置。在另一实施例中，利用对源文档的摘录和对任意目标文档的引用，可以将源文档的摘录和目标文档发送到目标。

[0027] 使用基于 XML 模式定义 (XSD) 的 XML 电子商务文档、或更一般地利用编码文本字符的字符数据的句法模式和根据文档的逻辑结构标识存储单元组的标记数据，图 2 中部分地图解的公共可访问注册表使团体管理变得容易。在需要变换的设计和执行中，在至少一个注册表中保持变换使重新使用变得更容易。变换的公共可访问注册表还可以允许分布式执行。网络服务引擎可以使用源、目标或中间服务的资源。一旦确定了由源和目标使用的接口，就可以从公共可访问注册表或从保存事先从公共可访问注册表获得的变换逻辑的高速缓冲器中，获得适合的变换逻辑。根据在一个或多个注册表中的条目和在一个或多个资料库中驻留的逻辑，在运行时间建立互用性。在运行时间，动态确定在源和目标之间的连接。当源或目标执行版本变化，连接的动态确定是版本变化的原因。

[0028] 公共可访问注册表可以提供所谓的语义中枢 (hub)。公共可访问注册表可以保持用于提供服务 (例如电子商务服务) 的应用的服务描述。最好以 XSD 定义的格式，将到达 (inbound) 和发出 (outbound) 文档接口注册为服务描述的一部分。服务可以自由注册多个

接口，例如自由支持多个电子商务文档标准（即，xCBL2.0、xCBL3.0 或 xCBL3.5）的多个版本，或自有支持多个文档标准（即，xCBL、IDOC、OAG 或 BAPI）。文档族概念的引入提供了一种跨多种文档标准和标准版本以及自定义系统来管理模式和文档类型的方式。文档族将代表相同商务事件的文档类型关联到文档族中。变换映射或变换将管理标准和自定义逻辑，以便在文档族成员间转换。使用特定变换的成本可能反映文档的不完整翻译。另外，根据现有经验，通过先验方法，或通过动态比较应用变换前和后的文件语义内容，可以将变换成功分值与变换相关联。

[0029] 首选使用 XML 作为公共句法库保持变换，但不是必要的。XML 是丰富的、自我描述的数据表示，其有利于声明的映射逻辑 (declarative mappinglogic)。几种语义库（诸如 xCBL 组件模型）提供一致的语义库来利用 XML 的强大语义。XML 文档对语义注册表的模拟有利于重新使用和新的变换的快速开发，从而提高现有变换的价值。使用公共句法库甚至公共语义库，集中于语义映射降低了开发新的变换的复杂程度。商业分析家而不是程序员可以有能力使用变换授权工具来定义 XML 到 XML 的语义转换。

[0030] 如图 3 所示，文档族允许分类或分组文档。相关的文档在相同的文档族 300 下分组。通过文档标识指定文档。文档标识符逻辑结构用于表示消息的根元素，例如 XML 电子商务文档的根元素。文档标识符可以指定文档 ID、其关系、版本和族关联。XML 和非 XML 文档都可以配有标识符并且储存在公共注册表中。文档标识符的属性可以包括文档标识符（如，名称“Order”）；名称空间（如，urn :x-commerceone :document :com :commerceone :XCBL30 : XCB-L30.sox）；文档库名（如，xCBL、DTD、EDIFACT）；模式语言（如，SOX、XSDL）；版本（如，3.0）；和文档族名（如，PurchaseOrderFamily、PriceInquir-yFamily、QuoteFamily）。文档族通过文档标识符以版本级的方式组织文件。在图 3 中，文档族树 300 或其他数据结构用于组织单独族 310、320。例如，定购单族 310 可以包括一个或更多主要版本 311、312、313。在类似的树型结构中，一个或更多主要版本可以与次要版本（未示出）相关联。版本属性可以记录主要和次要版本 (versioning)。主要和次要版本之间可能的区别是主要版本具有需要变换的显著变化，而次要版本不具有结构上的差异，仅仅是子元素的扩展。系统用户可能公共地扩展文件的子元素而不修改文档类型本身。该子元素扩展可以按照处理文档类型的修改那样的方式，作为次要版本进行处理。因此，文档类型节点代表文档类型模式和标记文件元素的所有模式。例如，如果扩展 LineItem 元素，并且该扩展类型用于定购单的实例中，则定购单文档类型被版本化 (version)。当版本化子元素时，用户注册新的文档类型。它们指定父文档类型节点并且给该父类型分配新的次要版本关系。生成版本 ID 并且分配给新的节点。

[0031] 如图 4 所示，注册表将模式细分为名称空间。使用模式名称空间管理组件，可以注册并管理 XML 名称空间（如，XSD、SOX、RosettaNet、CIDX）和非 XML 名称空间（例如，EDI，EDIFACT）。模式名称空间可以具有不同的属性，这些属性包括 :名称空间 URI ;名称 ;分类、名称空间状态 ;有效状态（对于 XSD 名称空间）;名称空间版本 ;描述 ;文档库名 ;模式语言（对于 XSD 名称空间）;模式文件 ;豆罐 (bean jar) 文档名 ;从属名称空间（如果有，对于 XSD 和 SOX 名称空间）;和外部或信息 URL。通常，名称空间的不同版本具有不同的 URI。例如，用于主要 xCBL 版本 3.0401 和用于主要 xCBL 版本 3.5402 的文档库可能分别具有一个或更多的名称空间 (411、412、413) 和 414，这可以用于支持次要版本。一种包括模式的方

式是使用用于 n 个文件模式的 n 个文件。名称空间管理员可以存储有关名称空间的元数据 (meta data)、与名称空间相关联的模式文件和包含对应于模式文件的 JavaBeans 和类的 Java 罐文件。使用基于浏览器工具的图形用户接口可以用于管理注册、激活、去激活和名称空间族的删除。可以使用诸如 XML 工具的有效 API 之类的工具先确认公布的名称空间涵盖相关的模式文件。在图 4 中，有两个文档库 401、402。每个文档库分别包括三个模式名称空间 (411、412、413) 和 414。名称空间与模式文件 421-425 和 426 相关联。用于为定购单 (例如分别为 xCBL3.0 和 xCBL3.5 型的定购单 432、433) 从名称空间族 431 备份树的行动分别与特定的文件模式 421、426 相关联。

[0032] 图 5 表示文档族的另一个图表，在这里描述为通过变换互连的文档族成员的网络。在该定购单族中，文档 501、502、503 由库 (library)、文档标识符、版本和模式类型标识。例如，文档 501A 来自于 xCBL 库，被标识为 Order(定购单)、版本 4.0、使用模式类型 XSD。文档 501C 也来自于 xCBL 库，被标识为 Order、版本 3.5、使用模式类型 SOX。文档 502A 来自于 X12 标记库，被标识为 850 文档、版本 4200、使用模式类型 XSD。文档 502B 是以 XML 标记的自定义平面文件 (flat file)。这是可以用诸如模板或文字处理软件准备的文档类型。在该图中，为每个文档族成员之间的转换方向标识单独的变换。标识的变换类型包括 Contivo 映射、XST 映射、XSLT 映射、在 XSD 和 SOX 之间的 Java 类翻译、Java 子串替换和 Java 映射 (XDK)。不同变换类型可以用于文档族成员之间的变换和反向变换。系统可以用于新的或不同变换类型，例如，作为现有类的扩展。例如，从 xCBL 版本 3.5501B 翻译成 xCBL3.0501F 牵涉应用 XSLT 变换。翻译相对方向牵涉应用 Java 组件。通过变换互连的文档族成员的网络，在节点 (文档族成员) 之间的互连是具有不同属性的定向链接 (单方向变换) 的意义下，可以考虑为定向图。可以应用公知的算法来遍历 (traverse) 该网络而避免循环或循环引用。在该图中未示出，先验变换成功分值或基于经验变换成功分值可以与链接文档族成员的每个变换相关联。

[0033] 图 6 和图 7 描述可以用于在如图 5 描述的文档族中标识变换的表。这些表可以在运行时间通过变换引擎访问，来标识优选的变换。可以将某些变换高速缓冲。在图 6 中，通过以列出的次序应用一个或多个逻辑组件，实现从一个文档族 601 向另一个文档族 602 的变换。这些逻辑组件可以是 Java 类文件、XSLT 映射、XST 映射或任何其他通用或自定义变换，其适应当前和未来的文档标准和变换标准。变换成功分值 610 测量从翻译源 601 到目标文档 602 引起的不完整性。在这个实例中，变换项 (entry) 由源和目标文档属性索引。这些属性集包括文档族名称空间、族名、协议、模式语言、文档类型、XML Qname 和版本 ID。当搜索变换项时，可在搜索中使用通配符。变换项可以选择性地包括用于特殊规则 603-608 的标记符。自定义变换可以应用到贸易伙伴级、服务级或行动级。源或目标贸易伙伴 ID 可以标记为 603、604，来指示特殊的逻辑组件应该用于特定的源或目标贸易伙伴。类似地，服务和行动可以标记为 605-608，来表示特殊的逻辑组件应该用于特定的服务或行动。变换引擎应该使用可用的最明确的变换定义。例如，贸易伙伴的特定定义、服务和行动三元组应该被认为对于由贸易伙伴指定的变换是更明确的。在为不同的变换定义不同三元组元素的情况下，可以将分级重要性分配给贸易伙伴、服务或行动。例如，如果两个变换匹配源和目标文档类型，则可以认为贸易伙伴比服务更重要，这里的一个变换是专用于贸易伙伴的，而另一变换是专用于服务的。变换的其他属性可以引出 (evoke) 特殊的规则。本发明不限于

由贸易伙伴、服务和行动分类的特殊规则。图 7 提供关于用作列 611 中的变换组件的逻辑组件 701 的附加信息。为逻辑组件 710 提供类型 702、实现 703、配置 704、包 (package)705 和版本 706。

[0034] 图 8 描述可以用于表示文档族的类。这些类的某些方面对应于在图 3 和 4 中描述的逻辑结构。文档库 801 是针对文件和模式的组织的最高级。文档库名由诸如“xCBL”之类的字符串表示。库可以任意版本化 802。库版本由字符串表示。对于版本化或非版本化的库,可以提供名称空间 811。在名称空间中,可以存在从属关系,如由从名称空间类回指到名称空间类的关系循环所指示的那样。名称空间的属性包括名称空间 URI、名称、分类、模式语言、名称空间状态、有效状态、名称空间版本和描述。这些属性可以由字符串表示。此外,可以提供标记符和标记值来表示名称空间是有激活的、不活动的、旧的或废弃的。也可以提供标记符和标记值来表示名称空间是有效的或无效的。与名称空间相关联的是外部链接 803、全局元素 821、模式文件 824 和外部文件 827。在该实施例中,名称空间可以由 URL 外部地链接到统一资源名。还可以提供外部链接 803 的描述。名称空间可以链接到一组全局元素 821。这些全局元素表示 XML 文档的有效根元素名,其对应于在名称空间中识别的文档类型。该全局元素类可能对于其他类中保持的数据来说是冗余的。名称空间也可以链接到一组模式文件 824。可以提供到根模式文件和到其他模式文件盒 (container) 的两个不同的链接。根模式文件是连接或包括其他模式文件的根文件。模拟名称空间之中的从属关系,允许检索名称空间的所有模式文件和所有从属名称空间,以及确保模式不被意外地移除而导致其他名称空间处于不一致状态。模式文件的属性可以包括文档名字符串和相关路径字符串。可以选择性地提供绝对路径。模式文件元素 824 由外部文件 827 表示。外部文件对象用于模拟文件的物理位置,并且可以由需要物理文件表示的任何实体引用。例如,外部文件可以为直接链接到名称空间的豆罐文件。

[0035] 在该实施例中,名称空间通过文档 ID 类 812 链接到文件和文档族。文档 ID 812 可以实际上具有到名称空间的两种类型的链接,二者之一是属于根名称空间,另一个用于扩展名称空间。这支持主要版本和次要版本。主要版本文档 ID 可以是没有扩展文件的先前版本的文件的新版本。次要版本文档 ID 可以扩展主要或次要版本文档 ID。主要版本文档 ID 将只具有单一的名称空间关系,其引用其中定义了根元素的名称空间。次要版本文档 ID 引用父(主要版本)文档 ID 的名称空间连同其中存在任何扩展的任何其他名称空间。文档 ID 812 可以与文档族 804、外部 ID 805、文件规则 813、变换映射 823 和 XML 文档 ID 822。文档 ID 的属性可以包括名称、URI 和主要替换 URI。URI 是利用以下三个组件自动产生的:名称空间 URI、DocID 名称 DocID 版本。这个 Doc Id URI 用于引用这个 DocID。如果用户想要自定义 DocID 命名方案,则它们可以输入它们自己 URI,并且这在主要 AltId 关系中设置。用户还可以具有多于一个命名方案,在这种情况下,其他 Id 关系模拟这些名称。所有这些名称应该是唯一的。文档 ID 的属性还可以包括显示名称、描述和文件版本。所有这些属性可以保持为字符串。对于 XML 文档来说,文档 ID 的专门化是 XML 文档 ID 822。专门化的属性可以包括 XML 元素名、版本类型、豆类名及主要和次要版本。作为 XML 的特性,关系循环指示可以表示嵌套 (nested) 元素的 XML 文档 ID。外部 ID 805 可以与文档 ID 相关联。外部 ID 805 可以是注册密钥或 URI 的别名。主要的默认链接和一个或多个用户提供的别名都可以链接文档 ID 和外部 ID。

[0036] 可以充分概括文档 ID 规则 813 来支持变换、有效性和显示映射。有时称为变换映射的变换 823 是文档 ID 规则的专门化。实现变换的逻辑通过一组变换组件 825 链接到文档 ID 规则 813。变换组件依次链接到外部文件 827。变换映射 823 的属性可以包括成本(cost)或变换成功分值、变换 URI 和位置 URI。变换 URI 在注册表中唯一标识变换映射。位置 URI 是可选的标识符,用于表示应该在哪里发生变换。例如,如果在网络中只有一个主机能够执行变换,那么,其 URI 分配给位置 URI 属性,并且变换 / 路由器将要执行的变换发送到该主机。变换组件 825 的属性可以包括变换组件 URI、名称、说明、组件类型、实现文件、包名和执行次序。变换组件 825 可以作为一个组链接到文档 ID 规则 813。如果需要多于一次的变换,则执行次序属性确认应用变换的顺序。在该实施例中,变换逻辑可以包括 XSLT 映射、XST 映射、Java 组件或 Contivo 映射中的一个或多个。变换组件链接到一组配置元素 826 上。配置元素的属性可以包括名称或值。文档 ID 规则 813 还链接到一组映射上下文(context)字符串 814 上。如图 6 和 7 的上下文中所述,这些字符串将文档 ID 规则 813 与特定的贸易团体(发送 / 源或接收 / 目标团体)相关联,或与特定的服务或行动相关联。

[0037] 如图 9 所示,可以方便地通过 XML 变换模块访问用于重新得到或执行变换的逻辑。XTM 模块由注册表服务 905 支持,其服务来自本地和远程注册表的变换逻辑。注册表客户应用程序接口 904 保持关于变化是否从本地高速缓冲器或注册表 906 或远程注册表中检索的透明度。检索到的变换或变换引用可以传送到文档变换应用程序接口 907,在该实施例中,其包括用于各种变换类型 908 的资源。如果在替换实施例中,可以从也称为文档变换服务的文档变换 API 907 中调用注册表客户 API 904。可以由在服务居民社区中或来自远程社区的 XTM 模块 902 调用文档变换服务 907,诸如正在向居民社区发送文件的社区。变换服务的更新可以包括添加新的变换 908 的类型和变换引擎 907 的新版本。在更新文档变换 API 907 和组件变换 908 后,可以协调地更新在 XTM 模块和文档变换 API 之间的连接器。可以从非居民社区的不同社区中调用文档变换服务。例如,从社区 A 向社区 B 发送定购单的服务可以调用社区 B 中的服务。为了执行所需的变换以便将使用社区 A 的语义的、准备的定购单 (PO) 对于社区 B 是可接受的,可能需要调用仅仅在社区 B 中的变换引擎上运行的变换。在这种情况下,社区 A 中的 XTM 模块将调用社区 B 中的文档变换 API,来远程执行一个或多个变换,将定购单从社区 A 的语义转换为社区 B 的语义。

[0038] 变换可以在到达 (inbound) 消息 901 中标识,但是其可能最好不包括应该应用哪个变换 (transform) 来完成变换的详细信息。在图 10 中,将所谓互用性契约文档 (ICD) 1011 以像要变换的消息那样的信封 901,发送到 XTM 1001。ICD 可以包括变换命令的路径和沿将文档从源转移到目标路径的连接器。在一个实施例中,XTM 模块与 B2B 应用的社区中的连接器组件相关联,该社区可以属于一个或多个通信网络。在执行任何变换时,XTM 模块可以访问 ICD 并且确定其包含的变换命令是否标识其连接器。如果没有当前连接器或变换的 XTM 模块执行变换,则 XTM 模块可以返回成功,并且可以选择性地登录通过事件。如果由当前 XTM 模块执行变换,则其解析变换命令,并且从注册表客户 API 1002 中获得执行变换的顺序 1002、1003。XTM 从信封 901 中提取源文档。它将源文档属性与将要执行的第一变换相匹配,如果出现不匹配则指示错误。在步骤 1014,其使用将要检索并执行的变换的列表来调用文档变换 API 1003。如果在变换处理中产生错误,则可以标明错误,或者中断变换并且返回错误消息。XTM 模块 1001 可以实现源和变换的目标文档的安全性、认可

(non-repudiation)、调试或其它目的(未示出)。XTM模块确定目标是否首选含有所发送的源文档以及变换后的目标文档,如果是这样的话,则当在步骤1016创建发出的信封903时将其附加上。XTM模块应该以线程安全的方式实现。变换后的信封903在步骤1017返回。

[0039] ICD包含在像要变换的消息那样的信封901中,可以使用下列模式来标识所需的变换:

[0040] <? xml version =“1.0”encoding =“UTF-8”2>

[0041] <xs:schema xmlns:xs =“http://www.w3.org/2001/XMLSchema”

[0042] elementForm Default =“qualified”attributeFormDefault =“unqualified”>

[0043] <xs:element name =“WransformationContract”>

[0044] <xs:annotation>

[0045] <xs:documentation>Transformation Instructions</xs:documentation>

[0046] </xs:annotation>

[0047] <xs:complexType>

[0048] <xs:sequence>

[0049] <xs:element name =“Attachment”type =“xs:boolean”minOccurs =“0”/>

[0050] <xs:element name =“Transformation”minOccurs =“0”maxOccurs =

[0051] “unbounded”>

[0052] <xs:complexType>

[0053] <xs:sequence>

[0054] <xs:element name =“Connector”type =“xs:anyURI”/>

[0055] <xs:element name =“StartDocTypeName”type =“xs:

[0056] QName”/>

[0057] <xs:element name =“StartDocVersion”type =“xs:string”/>

[0058] <xs:element name =“EndDocTypeName”type =

[0059] “xs:QName”/>

[0060] <xs:element name =“End DocVersion”type =“xs:string”/>

[0061] <xs:element name =“CommunityID”type =“xs:string”

[0062] minOccurs =“0”/>

[0063] <xs:element name =“ComponentID”type =“xs:string”

[0064] maxOccurs =“unbounded”/>

[0065] </xs:sequence>

[0066] </xs:complexType>

[0067] </xs:element>

[0068] </xs sequence>

[0069] </xs:complexType>

[0070] </xs:element>

[0071] </xs:schema>

[0072] 根据上述模式,变换命令的实例为 :

[0073] <xs:element name =“Transformatio”minOccurs =“0”maxOccu rs =“unbounded”>

[0074] <xs:complexType>

[0075] <xs:sequence>

[0076] <xs:element name =“Connector”type =“xs:anyURI”/>

[0077] <xs:element name =“StartDocTypeName”type =“xs:QName”/>

[0078] <xs:element name =“StartDocVersionID”type =“xs:string”/>

[0079] <xs:element name =“EndDocTypeName”type =“xs:QName”/>

[0080] <xs:element name =“EndDocVersionID”type =“xs:string”/>

[0081] <xs:element name =“CommunityID”type =“xs:string”minOccurs =“0”/>

[0082] <xs:element name =“ComponentID”type =“xs:string”maxOccurs =

[0083] “unbounded”/>

[0084] </xs:sequence>

[0085] </xs:complexType>

[0086] </xs:element>

[0087] 在该实例中,源文档类型由 StartDocTypeName 和 StartDocVersion 标识。StartDocTypeName 应该是完全合格的文档类型,在 XML 术语中为 QName,包括名称空间和文档类型的根元素的本地名称。或者,利用适合的管理规定,能够使用唯一的命名以加强在相关范围内的唯一性。应该提供版本标识符来区别相同文件的变化。用户可以在定购单中扩展地址元素,例如,扩展具有不同的次要版本 ID 而不是主要版本。EndDocTypeName 和 EndDocVersion 标识通过变换得出的目标文档。社区 ID 指定了注册变换的社区。例如经由变换组件 825,组件 ID 用于查找变换逻辑。

[0088] 在下面的模式摘录中表达了指定用于接收(或不接收)除变换后的目标文档外的原始源文档的 ICD 的目标的首选的实现 :

[0089] <xs:element name =“TransformationContract”>

[0090] <xs:annotation>

[0091] <xs:documentation>Transformation Instructions</xs:documentation>

[0092] </xs:annotation>

[0093] <xs:complexType>

[0094] <xs:sequence>

[0095] <xs:element name =“Attachment”type =“xs:boolean”minOccurs =“0”/>

[0096] 附加标注表示是否应该附加原始源文档。在该元素缺失的情况下,默认值可以附加文档或不附加文档。

[0097] 图 11 描述用于将文档从源语义转换到目标语义的变换的链。在该图中,由方框表示文档状态,而由实线或虚线表示状态-状态变换。实线和虚线表示替换变换。这些变换可以是公用和专用变换,也可以是通常可用和专门选择的变换。在第一实例中,源 1101 希望向目标 1104 发送定购单。源的文档标准或本地 (native) 接口为 IDOC。在 IODC 语义内,该定购单的文档名和版本是 ORDERS2。模式类型是 XSD。目标的本地接口是 OAG。该文档名是 Purchase Order(定购单)。这个定购单的版本是 7.2.1。模式类型是 XSD。在该实例中,来自源或发送方注册表 1131 和目标或接收方注册表 1132 两者的变换都要使用。这一系列变换被追踪 1141。源文档接受源注册表 1131 变换 1101-1102 和 1102-1112。这些变换将 ORDERS02 文件转换为 xCBL 版本 4.0 定购单文档。然后,应用来自目标注册表的两个附加变换 1103-1113 和 1113-1104。因此,通过应用四个变换,IDOC 接口文档转换为 OAG 接口文档。在该实施例中,公共中间语义库是 xCBL。或者,通过检查图 11,很明显的是可以使用发送方注册表 1131 的三个变换和接收方注册表 1132 的单个变换来转换 IDOC 接口文档。另一路径已经使用在发送方注册表 1131 中的变换 1112-1122 从 xCBL 版本 4.0 转换到了版本 3.5。然后,将不需要使用具有相似功能 1103-1113 的接收方注册表 1132 变换。此转换的路径选择 1141 可以通过接收方注册表中 1103 和 1113 之间的虚线解释。这意味着目标首选使用其自身的变换用于版本 4.0 和 3.5 之间的转换。在第二个实例中,源 1121 希望向目标 1124 发送其 XYZ 定购单。在 1142 中,使用三个变换 1121-1122、1113-1123 和 1123-1124。此外,变换的语义库是 xCBL。使用自定义变换来将标记的平面文件转换为 xCBL 版本 3.5。其后,使用非自定义变换来将文件转换为 X12 标记格式。虽然这些实例说明了在源和目标注册表中存储的变换,但可以很好地使用其他注册表的配置,诸如单个公共注册表,或公共注册表和用于具有自定义逻辑组件的源和目标的补充注册表。

[0098] 在图 12 和 13 中提供关于使用变换的源和目标注册表的变换顺序的计算的详细内容。图 12 是整体流程图。图 13 描述能够用于跟踪通过一个或多个文档族成员的注册表的路径的多种算法中的一种。图 12 从关于源文档和源与目标的标识符的信息开始 1201。源文档由文档类型属性集说明。源和目标由团体、服务和行动三者说明。第一逻辑分支 1202 确定是否已经设置了反对变换的政策。在目标要求源来承担错误变换的所有风险的情况下,可能应用这种类型的政策,所以公用变换元素的使用处于源自身的风险中。如果存在反对变换的政策,则在 1211 返回不能变换命令消息。通过逻辑分支 1202,在 1203 检索目标的文档类型。这可以从上述的注册表检索。给出关于源和目标文档的信息,在 1204 确定替换变换顺序或路径,其可以包括路径的变换成功分值,还可以包括源和目标的变换首选。在 1205 检查替换路径的列表,并且标识用于产生理想的目标文档类型的候选路径。如果在列表中没有出现产生理想的目标文档类型的路径,则在 1211 返回不能变换命令消息。通过逻

辑分支 1206，在 1207 选择并提取优选路径。优选路径可以具有优选变换成功分值，或其符合源、目标或两者的变换首选。在 1208 创建变换命令并在 1209 返回。

[0099] 图 13 图解从特定文档族成员开始跟踪通过源和目标注册表的变换顺序路径。概括地说，算法查询源和目标注册表，以获得源和目标文档族中相同的文档类型的交集。在该图中没有图解其执行完整性和错误检查。对于多部分消息的每一部分，其确定目标文档，并且运行用于递归遍历文档族图的成本算法，下面的变换链接在文档状态节点之间。如果节点的文档类型在预先确定的相同文档类型的交集中，算法分解为通过注册表二者的路径。如果应用变换政策需要无损变换（最好的变换成功分值），则忽略有损变换路径。这种遍历和成本是 Dijkstra’s 算法的变型，该算法用于解决关于其中所有权是非负数的边缘加权 (edge-weighted) 的单一源、最短路径问题。其逐一地从某些起始节点找出到所有其他节点的最短路径。在 Dijkstra’s 算法中，以其加权长度的次序，从最短的开始遍历路经，一直到最长为止。通常，可以使用从源文档到目标文档的可用文档族的遍历，并且文档族可以足够小，以使得所使用的特定遍历对计算成本影响很小。

[0100] 参照图 13 中的流程图，这个部分的算法开始于 1204 起始节点或文档族成员，以及标识源和目标的团体 / 服务 / 行动三者。在步骤 1301，计算源和目标注册表之间的节点的交集。例如，源和目标都处理 xCBL 版本 3.0 或 xCBL 版本 3.5 文件么？如果在源和目标处理的文件语义之间没有交集时，则变换顺序不可用。参照图 11，交集可以是 xCBL 版本 4.0 和 3.5(1112-1103 和 1122-1113)。由源节点、处理过的节点和变换顺序的该处理算法保持列表。这些列表的某些或所有可以保持在堆栈中或递归分配和处理的变量的堆中。参照图 5，方框（如，501、502 或 503）是源节点，其中定向图从源节点开始行进。根据行进进程，可以标注或不标注源节点。通过将起始节点添加到源节点列表 1302 开始行进。在 1303 和 1305 限制的循环和通过由 1311 和 1324 限制的内循环中处理列表。在 1303，开始在源节点列表中的所谓 i 节点的处理。标注当前的 i 节点。然后，在 1311 考虑还没有标注的文档族的连接的成员。例如，参照图 5，对于 i 节点 501B，连接的文档族成员节点是 501A、501C、501F 和 502C。没有标注的连接的节点称为 y 节点 1311。在 1312 测试 y 节点来确定其是否在处理节点列表中，如果不是，则在 1321 将其加入到列表并且在 1313 进行处理。如果 y 节点在处理的节点列表中，则算法确定到 y 节点的当前路经是否好于预先计算的路径。在步骤 1313，将到达当前 y 节点的成本与到达相同节点的先前成本进行比较。如果当前的成本好于先前成本，则处理前进到步骤 1314，在这里更新处理的节点的列表。在步骤 1315，y 节点被添加到源节点列表，用于后面的处理。此外，在步骤 1313，如果当前成本不是更好的，则处理前进到步骤 1322，在这里测试成本是否相同。如果成本相同，则在 323 可能使用各种标准来断开连接 (tie)。当相同的节点出现在接收方和发送方注册表中时，一个标准是提供在接收方注册表中的 y 节点的实例。另一个标准是提供在发送方注册表中的 y 节点的实例。再一个标准是提供包含最少节点或转发的路径。在步骤 1324，处理循环到 1311，在这里处理没有标注的下一个连接节点。如果处理了所有的未标注的连接节点，则下一个步骤是 1305，在这里处理循环到 1303，处理源节点中下一个 i 节点。在 1305，当处理了所有的源节点时，该处理的结果在 1306 返回。

[0101] 替换变换顺序和优选变换顺序的计算可以在不同的环境中运行。下面的使用情况说明了这些环境中的某些环境。在第一种使用情况下，不需要变换。调用用于确定变换顺

序的模块，但是源和目标文档是相同类型的。不需要变换。在第二种使用情况中，在源和目标之间没有可用变换。这种情况可能是：在不同的源和目标文档之间不能计算出变换顺序，或变换政策是“没有变换”并且源和目标文档不同，或只接受无损变换但所有所计算出的变换顺序是有损的，如它们的变换成功分值所示。产生操作异常。在第三种使用情况中，源和目标处在同一的社区中，所以仅仅查询一个变换注册表，并且存在有效的路径。确定一个或更多变换顺序。确定优选的顺序。在第四种使用情况中，源和目标在分离的社区中，并且存在有效路经。查询两个变换注册表。与在第三种情况中相同，确定一个或多个变换顺序，并且确定优选的顺序。

[0102] 可以通过经验或动态地、或更一般地通过有损语义变换的任何度量来确定上述先验变换成功分值。根据分序和测试的组合，将先验分值分配给变换。分值不随着经验改变。基于经验的分值可以以先验分值或默认分值开始，并且根据经验调整。例如，可以应用下面解释的用于动态计算成功的方法，用于使用的所选择的变换，并且更新对应的变换成功分值，例如作为加权或移动平均数，不是放弃最旧的历史成功分值就是分配相对的权来忽略并表示成功分值。动态确定成功分值的一种途径是将变换应用到候选文件并且分析变换的文件。变换应用到源或中间源文档，产生目标或中间目标文档。列出源和目标文档的元素的内容（在 XML 或相似的文件中），例如频率表。无论差异是正数还是负数，源和目标频率之间的差异降低变换成功分值。选择性地报告差异。成功分值可以依赖于元素内容之间的精确匹配，或通过程度加权。下面的实例帮助说明动态评分的途径。分值文件片断是：

[0103] <NameAddress>

[0104] <Name>Pikachu Pokemon</Name>

[0105] <Address1>125 Henderson Drive</Address1>

[0106] <City>Pleasanton</City>

[0107] <State>CA</State>

[0108] </NameAddress>

[0109] 变换的目标文档片断是：

[0110] <NameAddress>

[0111] <Name>Pikachu Pokemon</Name>

[0112] <Street>Henderson Drive</Street>

[0113] <HouseNumber>125</HouseNumber>

[0114] <City>Pleasanton</City>

[0115] <State>CA</State>

[0116] </NameAddress>

[0117] 根据源文档片段的元素和调整 (key) 到精确匹配的频率比较是：

[0118]

内容	源文档频率	目标文档频率
PikachuPokemon	1	1
125HendersonDrive	1	0

内容	源文档频率	目标文档频率
Pleasanton	1	1
CA	1	1

[0119] 对应于在目标文档中作为字段逐字出现的源文档中的字段的片段的动态成功分值，可以表示为 75％成功，或 25％成本被分配给该实例。由于目标文档的房屋号元素匹配源文档的地址 1 元素的一个标签，如果算出部分匹配，则分配给不同的分值。成功分值可以对应于源文档字段（其将逐字地出现在目标文档的字段中）中的文本的片断。为某些目的，分值顺序的应用需要计算总成功分值。当单个分值组合为总变换成功分值时，组合可以是相加、平均或相乘。构造总变换成功的方法可能顺序考虑这些变换（这与成功分值的相乘组合是相同的），或者可能积累（不合成）错误（这与成本的相加组合是相同的）。例如，在相乘组合中，如果变换是 T1、T2 和 T3，则可以为三个变换中的每一个计算有损百分比，并且组合为 (1-T1)*(1-T2)*(1-T3)。通常，总变换成功分值可以是从源到目标文档的有损变换的变换顺序的任意度量。

[0120] 在图 14 和 15 中图解了用于管理文档族信息并用于搜索变换的用户接口。图 14 描述用于支持文档族管理的用户接口。文件树 1401 显示文件 1402 的主要 1403 和次要 1404 版本的分级的相互关系。对于族，在 1411 显示族成员的公共文档族信息。图 15 描述支持搜索的用户接口来找出可用变换，例如，来准备新变换顺序。在 1511 显示的结果标识变换顺序的一部分 23，该顺序用于将源文档（定购单、CBL、SOX、Y、200）从 xCBL 版本 2.0 转换到版本 3.0。使用标准 1501 或高级 1502 查询接口指定搜索标准。结果中的一或多行 1512 可以被删除或者用于创建新变换 1513。在该实例中，通过所表达的发送方 1514 或接收方 1515 的首选、变换 1516 的成本或有损性和实现变换顺序 1517 的逻辑组件，来改变返回的变换顺序。

[0121] 尽管已参照本发明的详细优选实施例和实例描述了本发明，但是将理解的是，这些实例适用于说明而不是用于限定的。计算机辅助处理暗含在描述的实施例中。因此，可以以计算机辅助处理、包含逻辑来实现变换处理的系统、具有逻辑来实现变换处理的媒体、具有逻辑来实现变换处理的数据流或计算机可访问变换处理服务实现本发明。但本领域内的普通技术人员将理解的是，可在不背离由所附权利要求书限定的本发明宗旨和范围的前提下对本发明进行各种形式和细节上的修改。

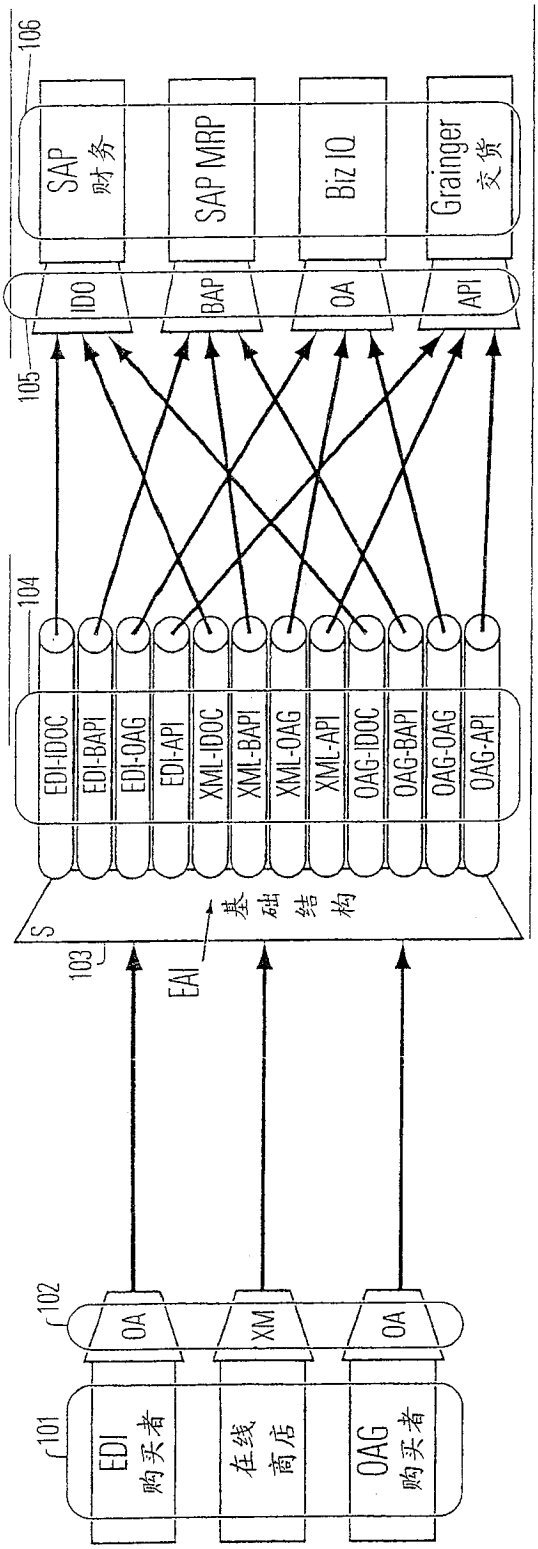

106
SAP 财务
SAP MRP
Biz IQ
Grainger 交货
IDO
BAP
OA
API
105
104
EDI-IDOC
EDI-BAPI
EDI-OAG
EDI-API
XML-IDOC
XML-BAPI
XML-OAG
XML-API
OAG-IDOC
OAG-BAPI
OAG-OAG
OAG-API
S
基础结构
EAI
103
102
OA
XM
OA
101
EDI 购买者
在线商店
OAG 购买者

图 1

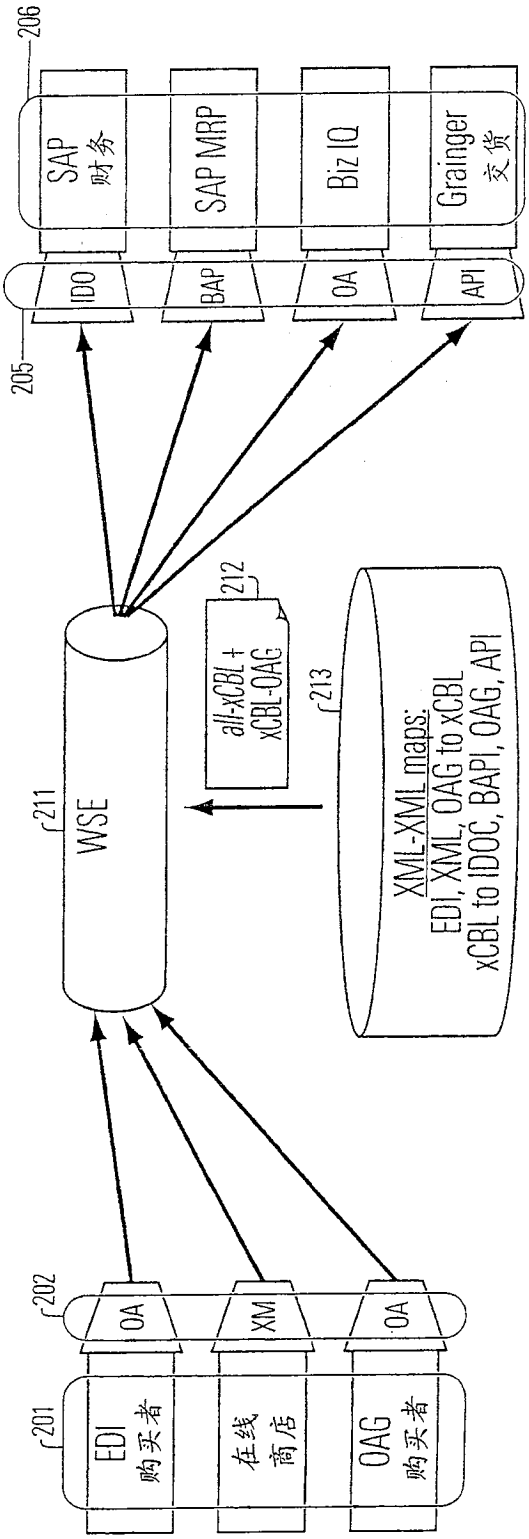

206
SAP 财务
SAP MRP
Biz IQ
Grainger 交货
IDO
BAP
OA
API
205
212
all-xCBL + xCBL-OAG
213
XML-XML maps:
EDI, XML, OAG to xCBL
xCBL to IDOC, BAPI, OAG, API
211
WSE
202
OA
XM
OA
201
EDI 购买者
在线商店
OAG 购买者

图 2

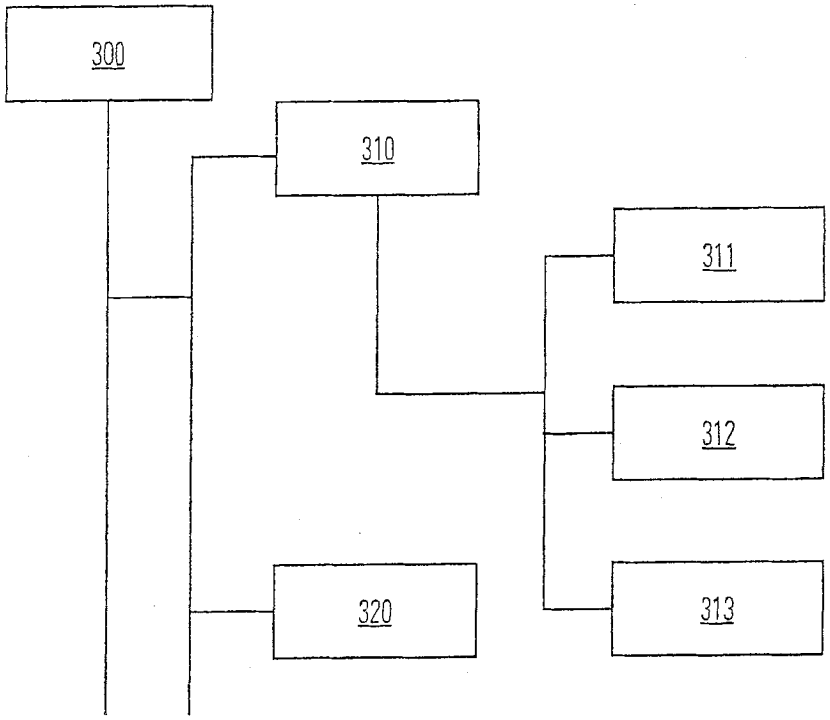

300
310
311
312
320
313

图 3

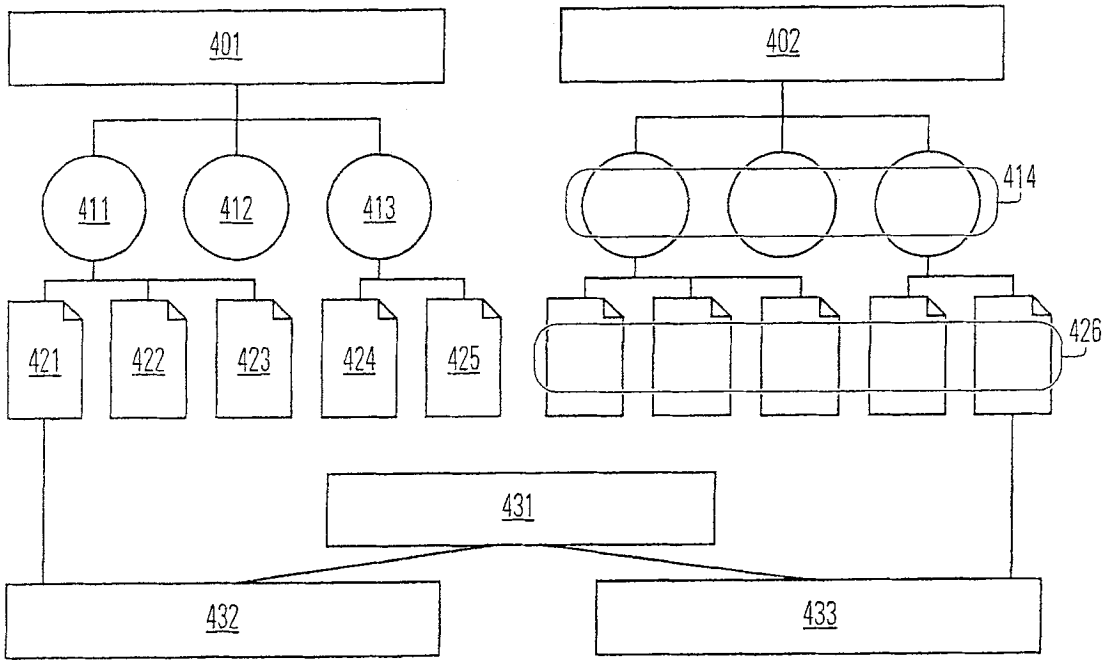

401
402
411
412
413
414
421
422
423
424
425
426
431
432
433

图 4

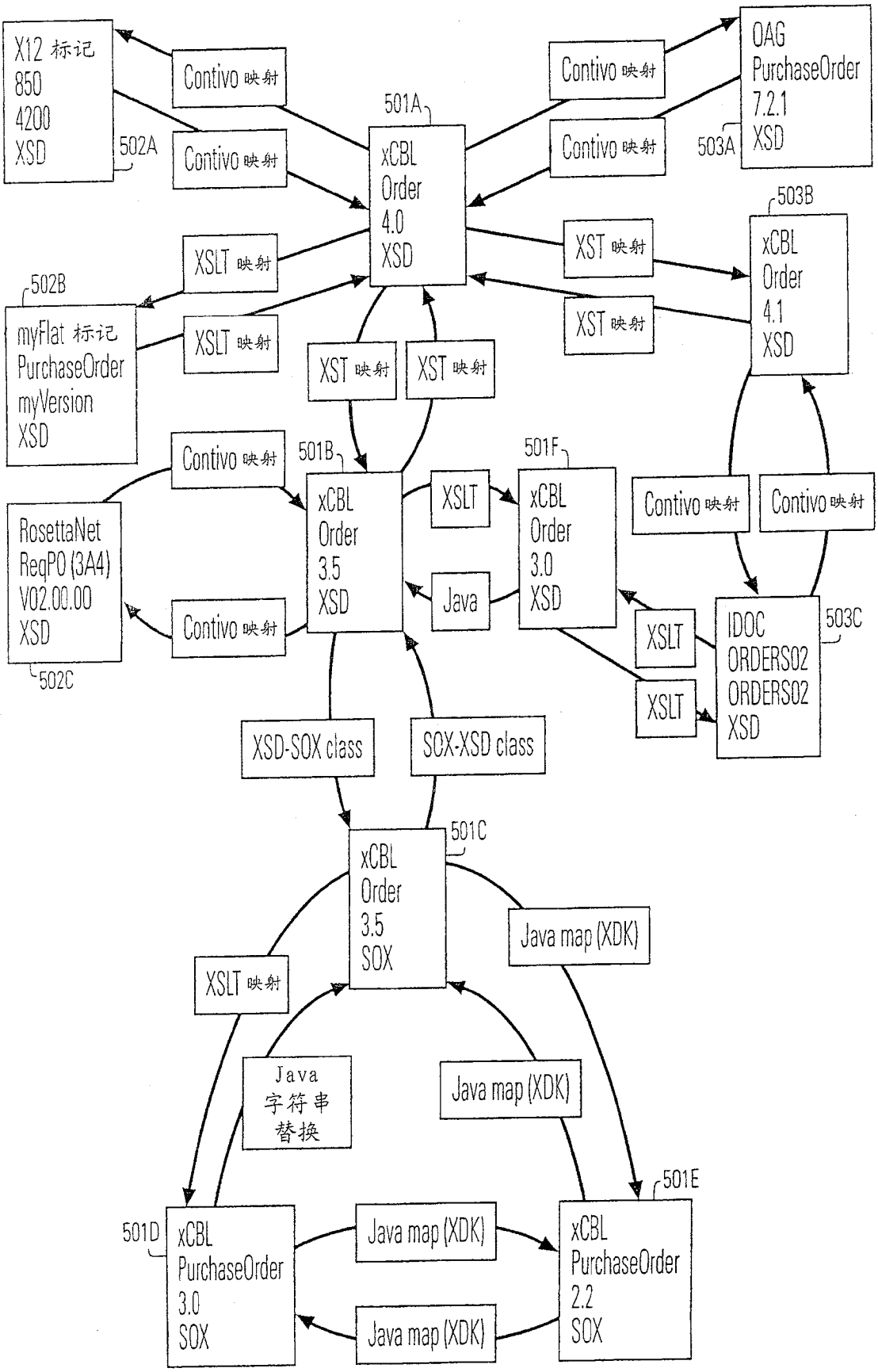
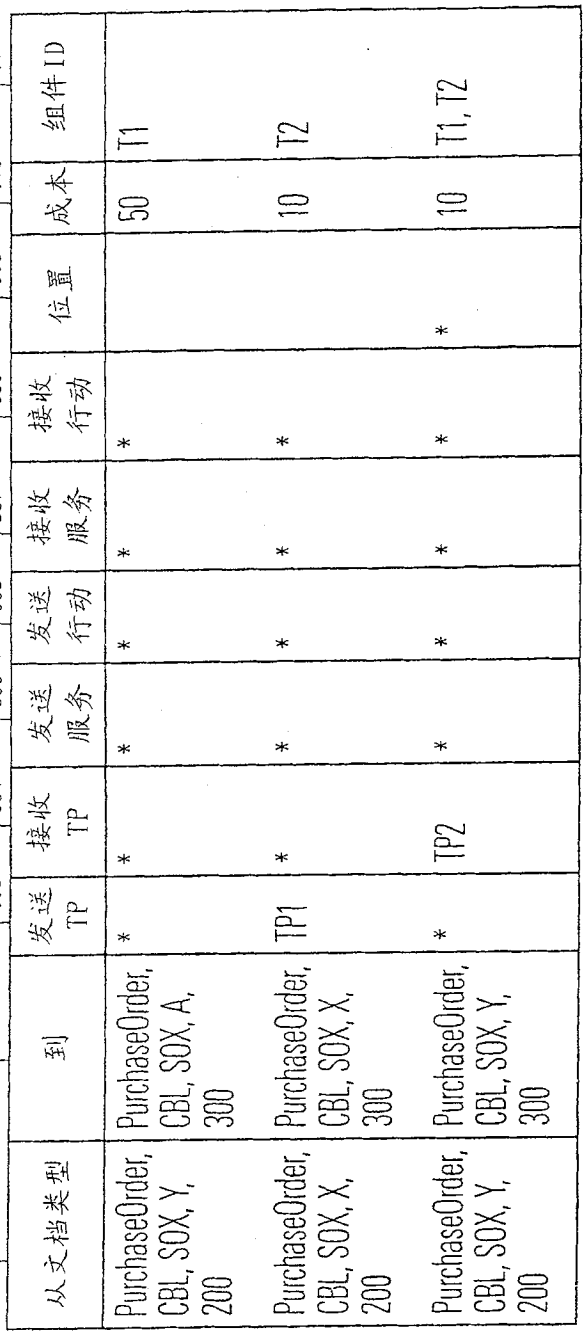
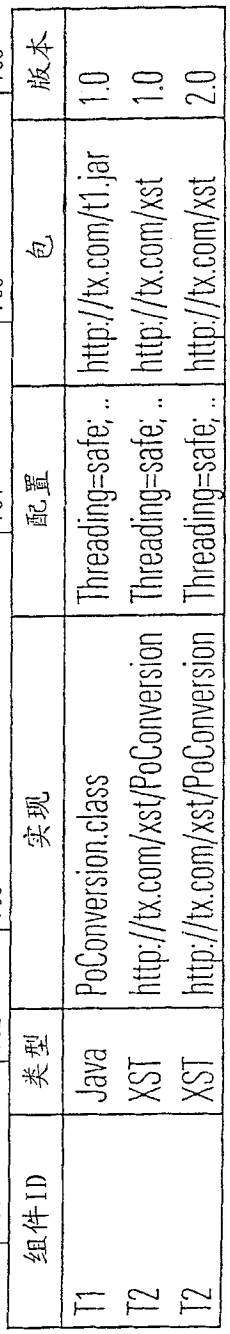

X12 标记
850
4200
XSD
502A
Contivo 映射
Contivo 映射
501A
xCBL
Order
4.0
XSD
Contivo 映射
Contivo 映射
OAG
PurchaseOrder
7.2.1
XSD
503A
503B
xCBL
Order
4.1
XSD
XST 映射
XST 映射
XSLT 映射
XSLT 映射
502B
myFlat 标记
PurchaseOrder
myVersion
XSD
XST 映射
XST 映射
Contivo 映射
501B
xCBL
Order
3.5
XSD
XSLT
Java
501F
xCBL
Order
3.0
XSD
Contivo 映射
Contivo 映射
RosettaNet
ReqPO (3A4)
V02.00.00
XSD
502C
Contivo 映射
XSLT
XSLT
IDOC
ORDERS02
ORDERS02
XSD
503C
XSD-SOX class
SOX-XSD class
501C
xCBL
Order
3.5
SOX
Java map (XDK)
XSLT 映射
Java
字符串
替换
Java map (XDK)
501E
501D
xCBL
PurchaseOrder
3.0
SOX
Java map (XDK)
Java map (XDK)
xCBL
PurchaseOrder
2.2
SOX

图 5



601 从文档类型	602 到	603 发送TP	604 接收TP	605 发送服务	606 发送行动	607 接收服务	608 接收行动	609 位置	610 成本	611 组件ID
PurchaseOrder, CBL, SOX, Y, 200	PurchaseOrder, CBL, SOX, A, 300	*	*	*	*	*	*		50	T1
PurchaseOrder, CBL, SOX, X, 200	PurchaseOrder, CBL, SOX, X, 300	TP1	*	*	*	*	*		10	T2
PurchaseOrder, CBL, SOX, Y, 200	PurchaseOrder, CBL, SOX, Y, 300	*	TP2	*	*	*	*	*	10	T1, T2

图 6



701 组件ID	702 类型	703 实现	704 配置	705 包	706 版本
T1	Java	PoConversion.class	Threading=safe; ..	http://tx.com/t1.jar	1.0
T2	XST	http://tx.com/xst/PoConversion	Threading=safe; ..	http://tx.com/xst	1.0
T2	XST	http://tx.com/xst/PoConversion	Threading=safe; ..	http://tx.com/xst	2.0

图 7

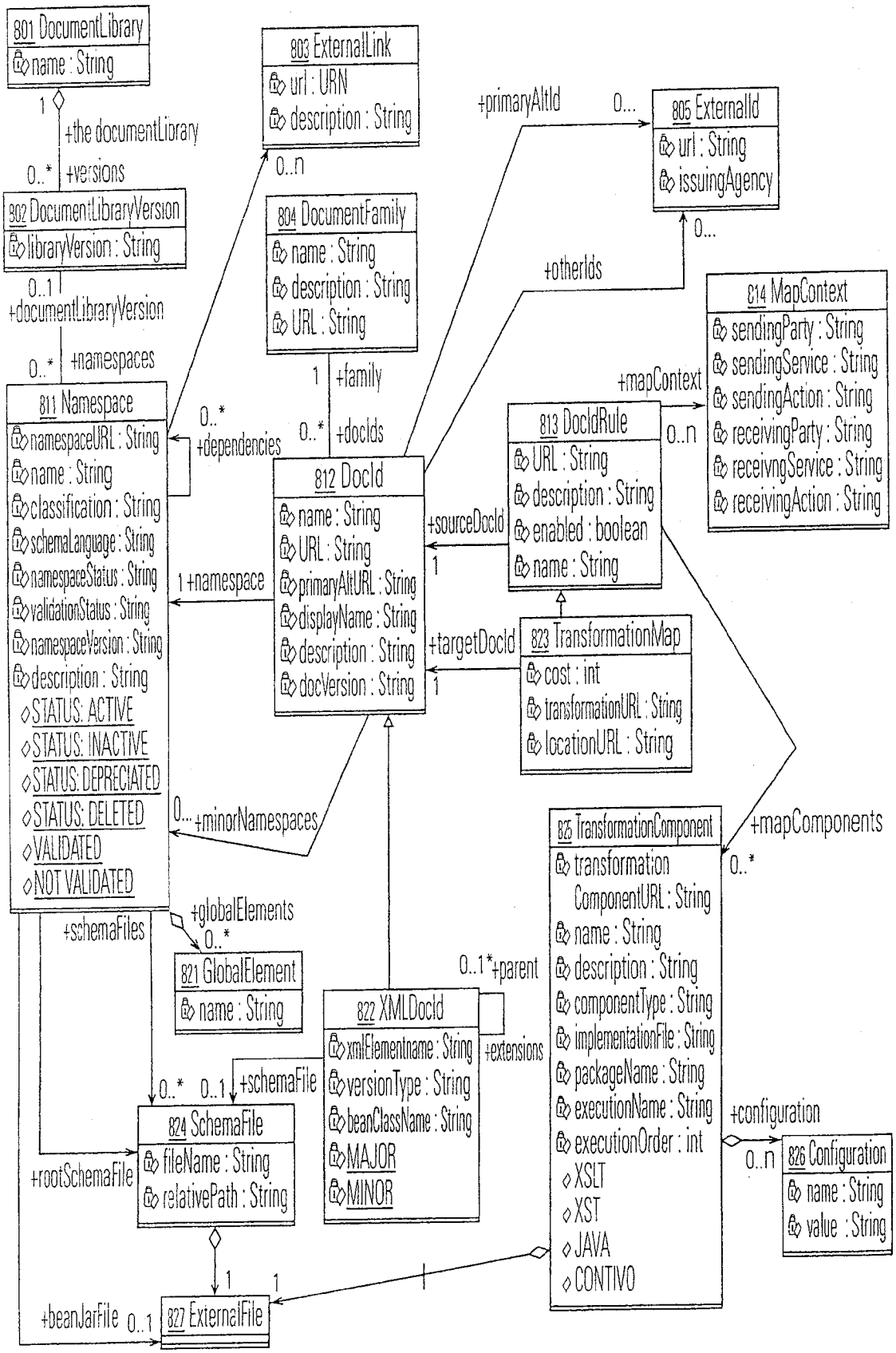

801 DocumentLibrary
name : String
1
+the documentLibrary
0..*
+versions
802 DocumentLibraryVersion
libraryVersion : String
0..1
+documentLibraryVersion
0..*
+namespaces
811 Namespace
namespaceURL : String
name : String
classification : String
schemaLanguage : String
namespaceStatus : String
validationStatus : String
namespaceVersion : String
description : String
STATUS: ACTIVE
STATUS: INACTIVE
STATUS: DEPRECIATED
STATUS: DELETED
VALIDATED
NOT VALIDATED
803 ExternalLink
url : URN
description : String
0..n
804 DocumentFamily
name : String
description : String
URL : String
1
+family
0..*
+docIds
0..*
+dependencies
812 DocId
name : String
URL : String
primaryAltURL : String
displayName : String
description : String
docVersion : String
1 +namespace
0... +minorNamespaces
+primaryAltId
0...
805 ExternalId
url : String
issuingAgency
0...
+otherIds
814 MapContext
sendingParty : String
sendingService : String
sendingAction : String
receivingParty : String
receivngService : String
receivingAction : String
+mapContext
0..n
813 DocIdRule
URL : String
description : String
enabled : boolean
name : String
+sourceDocId
1
823 TransformationMap
cost : int
transformationURL : String
locationURL : String
+targetDocId
1
825 TransformationComponent
transformation ComponentURL : String
name : String
description : String
componentType : String
implementationFile : String
packageName : String
executionName : String
executionOrder : int
XSLT
XST
JAVA
CONTIVO
+mapComponents
0..*
+configuration
0..n
826 Configuration
name : String
value : String
+globalElements
0..*
821 GlobalElement
name : String
+schemaFiles
822 XMLDocId
xmlElementname : String
versionType : String
beanClassName : String
MAJOR
MINOR
0..1*+parent
+extensions
0..* 0..1 +schemaFile
824 SchemaFile
fileName : String
relativePath : String
+rootSchemaFile
1 1
+beanJarFile 0..1
827 ExternalFile

图 8

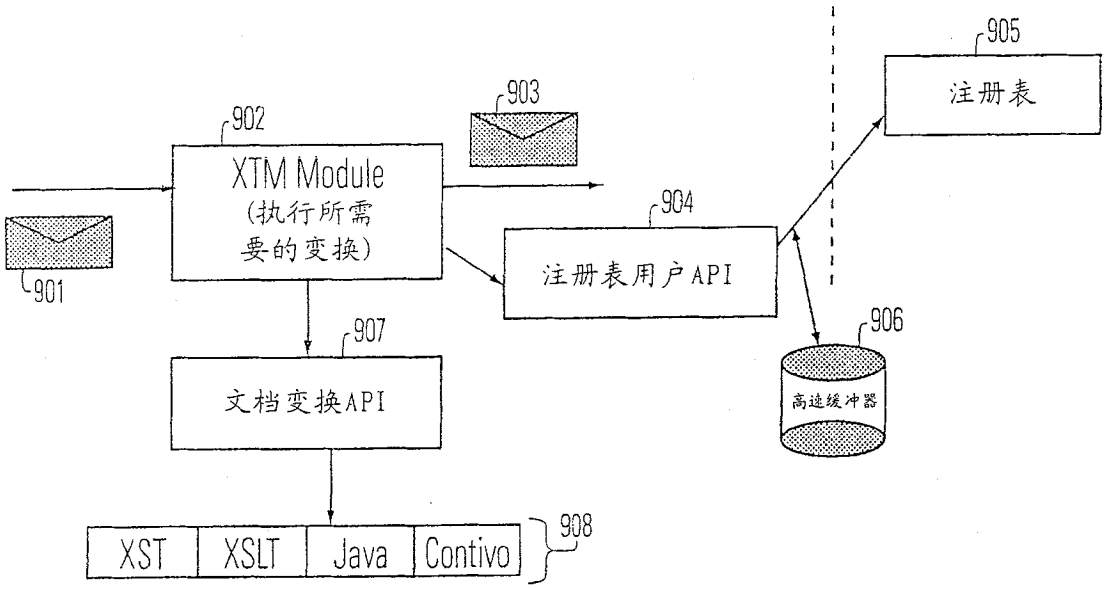

905
注册表
903
902
XTM Module
(执行所需
要的变换)
904
注册表用户API
901
907
906
文档变换API
高速缓冲器
XST
XSLT
Java
Contivo
908

图 9

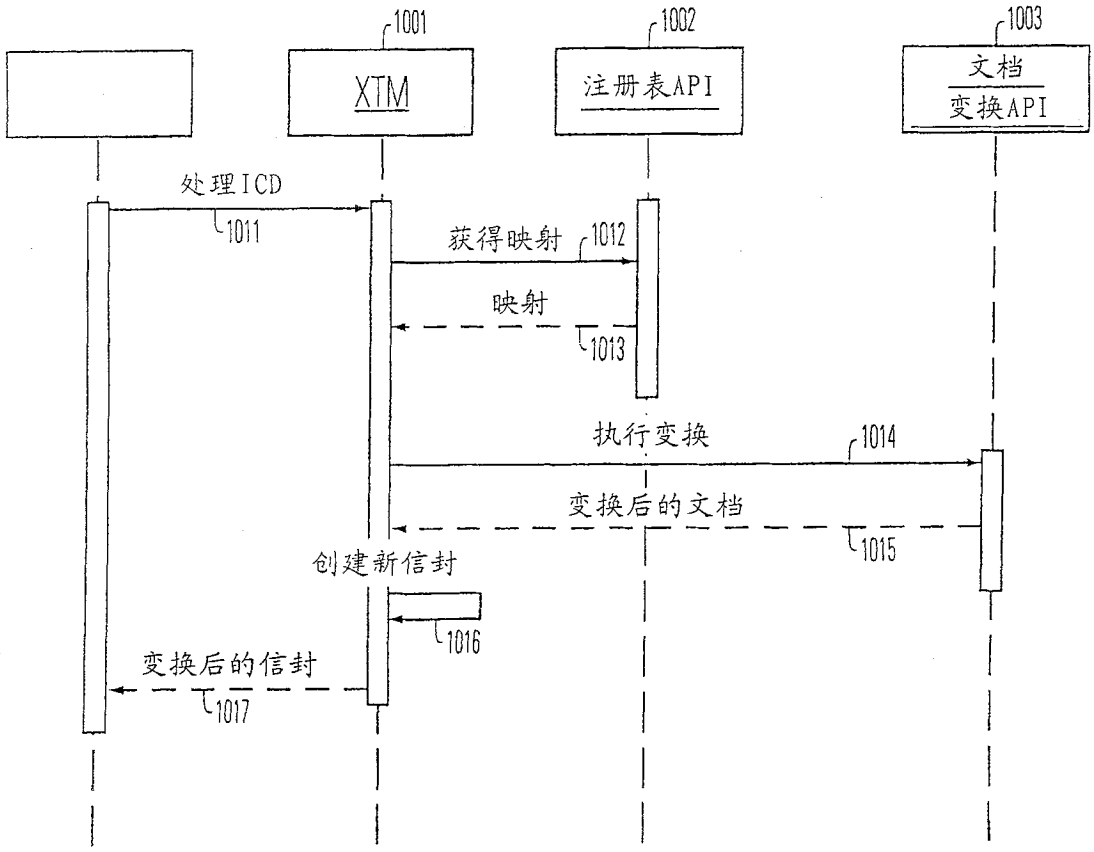

1001
XTM
1002
注册表API
1003
文档
变换API
处理ICD
1011
获得映射
1012
映射
1013
执行变换
1014
变换后的文档
1015
创建新信封
1016
变换后的信封
1017

图 10

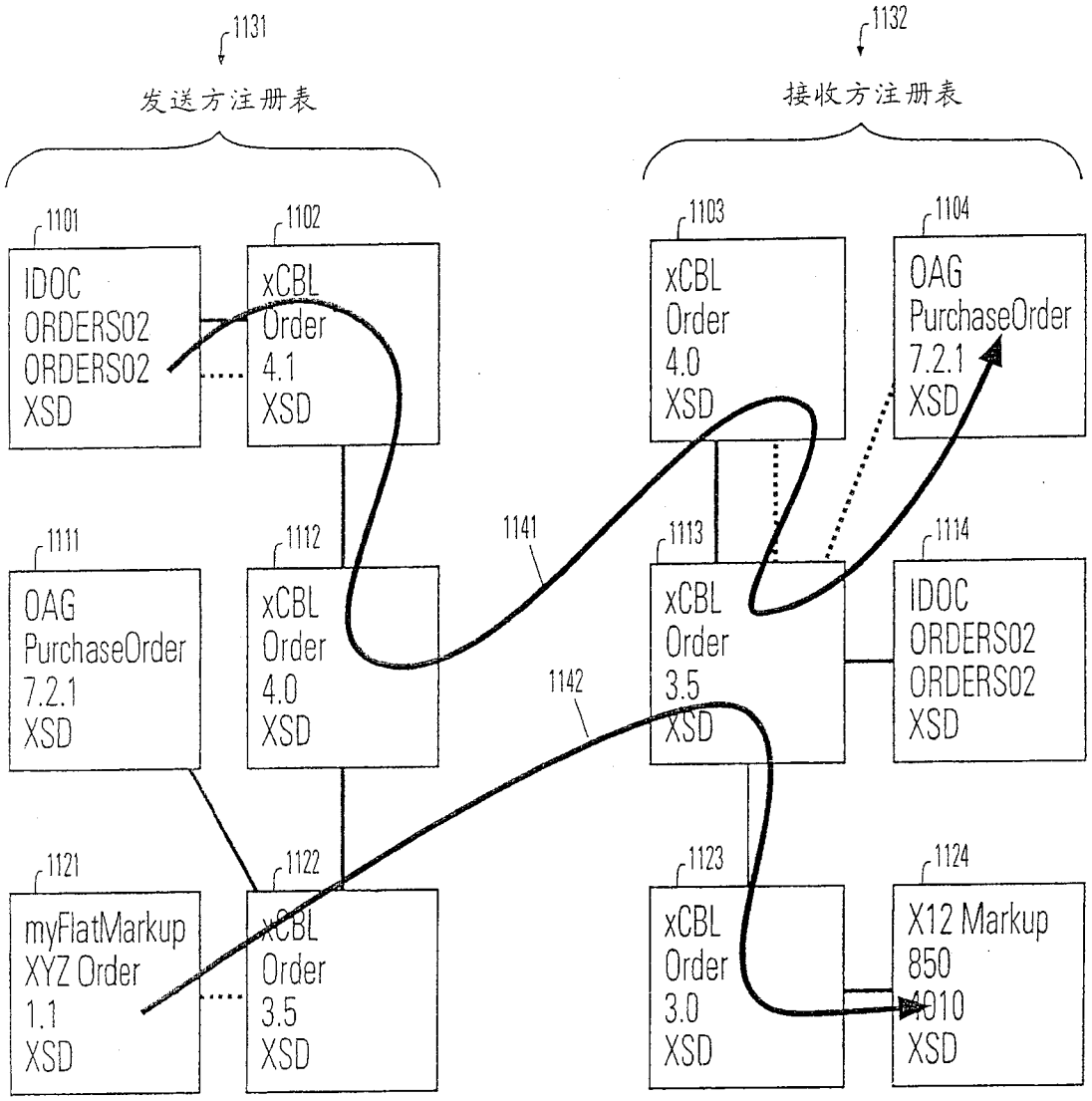

1131
发送方注册表
1132
接收方注册表
1101
IDOC
ORDERS02
ORDERS02
XSD
1102
xCBL
Order
4.1
XSD
1103
xCBL
Order
4.0
XSD
1104
OAG
PurchaseOrder
7.2.1
XSD
1111
OAG
PurchaseOrder
7.2.1
XSD
1112
xCBL
Order
4.0
XSD
1141
1113
xCBL
Order
3.5
XSD
1114
IDOC
ORDERS02
ORDERS02
XSD
1142
1121
myFlatMarkup
XYZ Order
1.1
XSD
1122
xCBL
Order
3.5
XSD
1123
xCBL
Order
3.0
XSD
1124
X12 Markup
850
4010
XSD

图 11

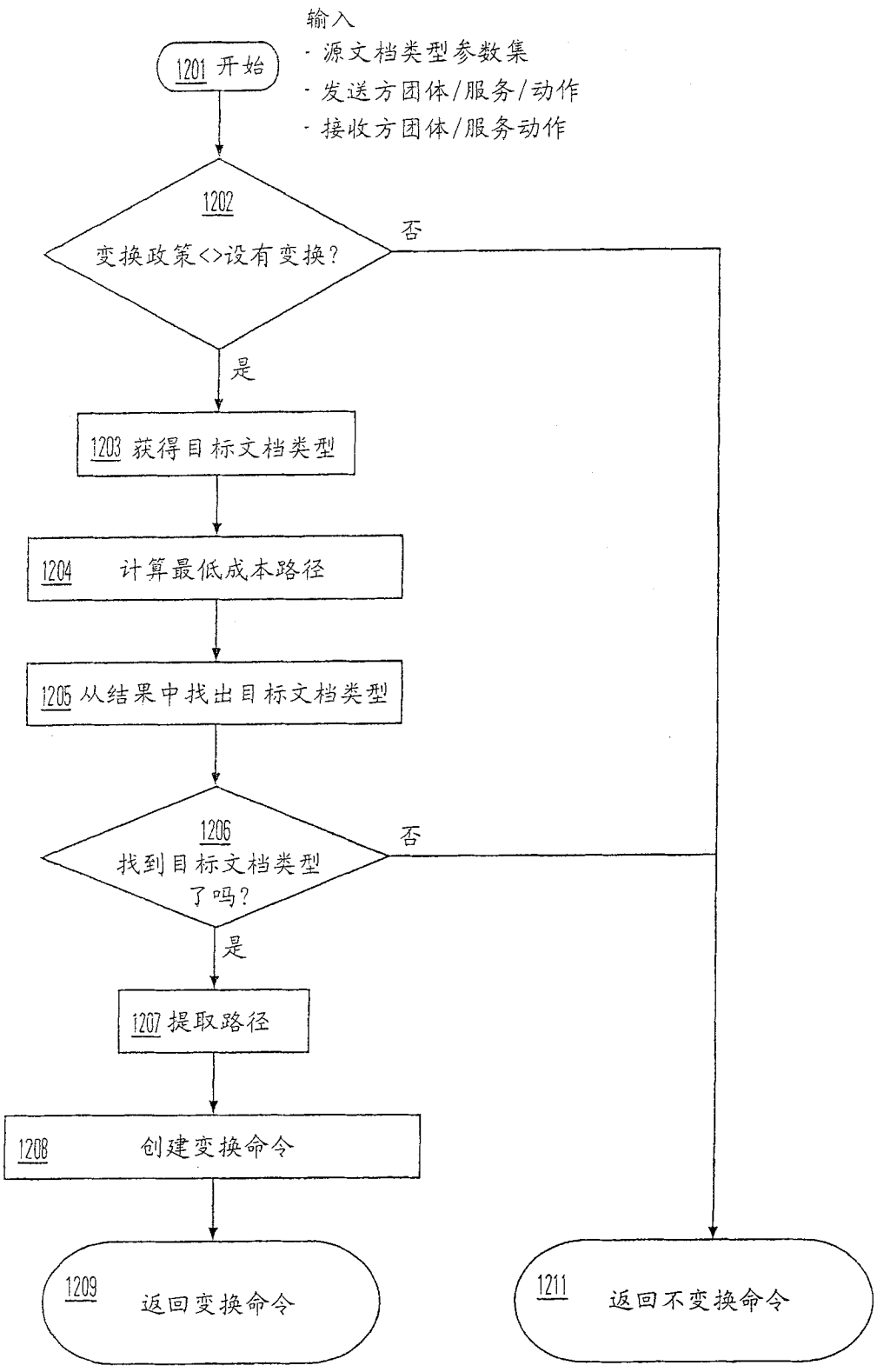

输入
· 源文档类型参数集
· 发送方团体/服务/动作
· 接收方团体/服务动作
1201 开始
1202
变换政策<>设有变换?
否
是
1203 获得目标文档类型
1204 计算最低成本路径
1205 从结果中找出目标文档类型
1206
找到目标文档类型了吗?
否
是
1207 提取路径
1208 创建变换命令
1209 返回变换命令
1211 返回不变换命令

图 12

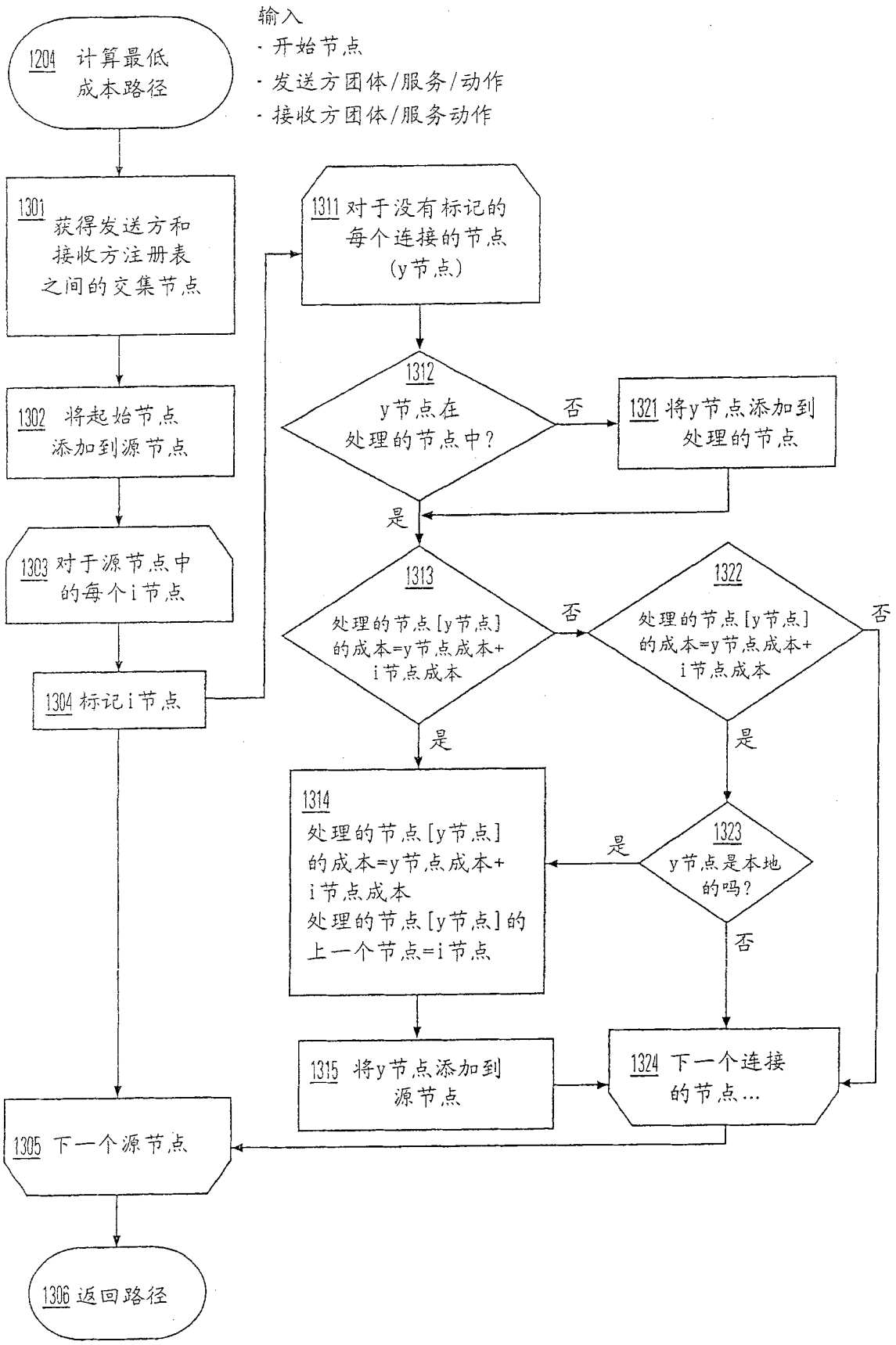

输入
· 开始节点
· 发送方团体/服务/动作
· 接收方团体/服务动作
1204 计算最低成本路径
1301 获得发送方和接收方注册表之间的交集节点
1302 将起始节点添加到源节点
1303 对于源节点中的每个i节点
1304 标记i节点
1305 下一个源节点
1306 返回路径
1311 对于没有标记的每个连接的节点(y节点)
1312 y节点在处理的节点中?
否
1321 将y节点添加到处理的节点
是
1313 处理的节点[y节点]的成本=y节点成本+i节点成本
否
是
1314 处理的节点[y节点]的成本=y节点成本+i节点成本
处理的节点[y节点]的上一个节点=i节点
1315 将y节点添加到源节点
1322 处理的节点[y节点]的成本=y节点成本+i节点成本
否
是
1323 y节点是本地的吗?
是
否
1324 下一个连接的节点...

图 13

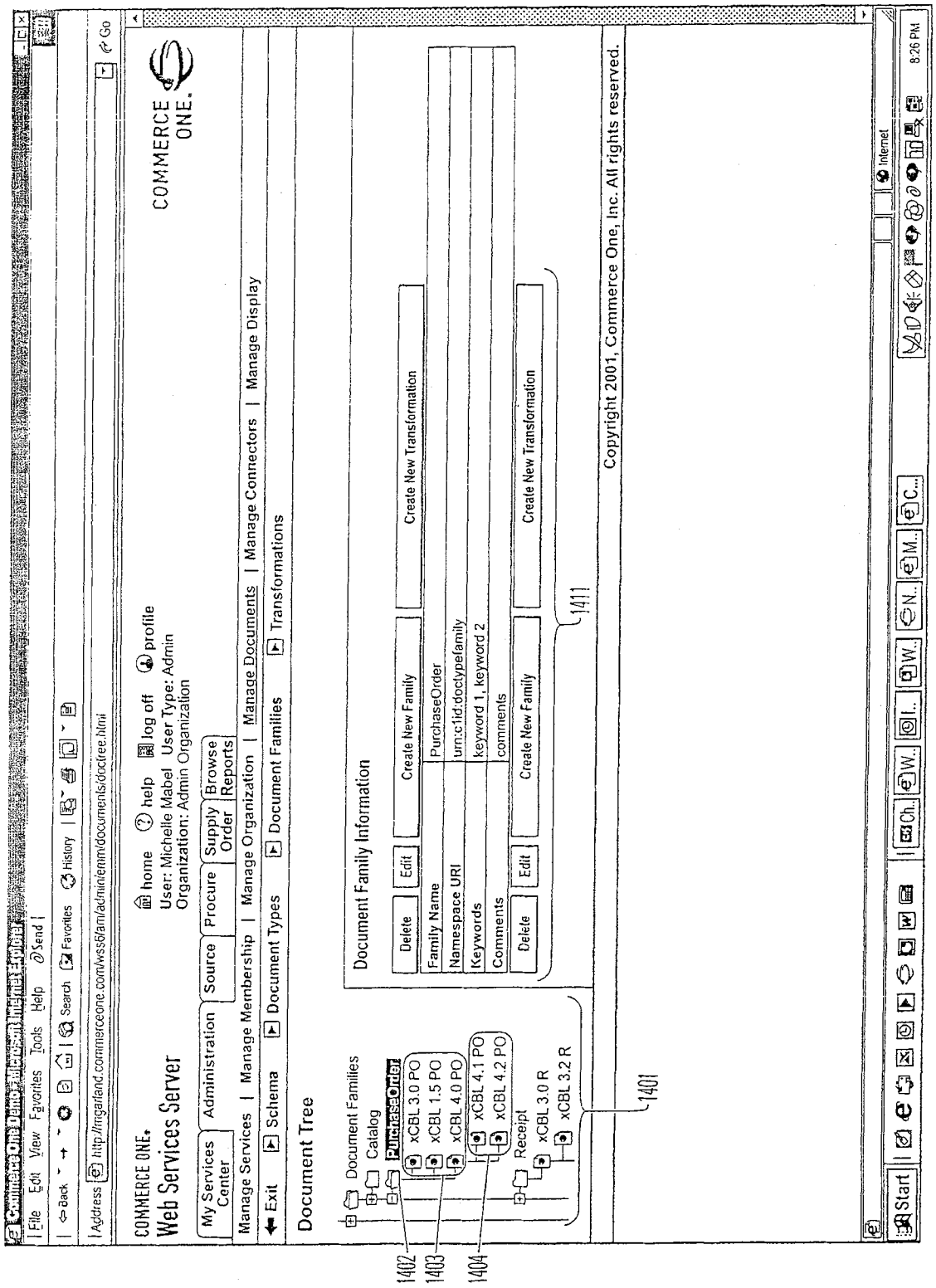

COMMERCE ONE
Web Services Server
home
help
log off
profile
User: Michelle Mabel User Type: Admin
Organization: Admin Organization
My Services Center
Administration
Source
Procure
Supply Order
Browse Reports
Manage Services | Manage Membership | Manage Organization | Manage Documents | Manage Connectors | Manage Display
Exit
Schema
Document Types
Document Families
Transformations
Document Tree
Document Families
Catalog
PurchaseOrder
xCBL 3.0 PO
xCBL 1.5 PO
xCBL 4.0 PO
xCBL 4.1 PO
xCBL 4.2 PO
Receipt
xCBL 3.0 R
xCBL 3.2 R
Document Family Information
Delete
Edit
Create New Family
Create New Transformation
Family Name
PurchaseOrder
Namespace URI
urn:c1id:doctypefamily
Keywords
keyword 1, keyword 2
Comments
comments
Delete
Edit
Create New Family
Create New Transformation
Copyright 2001, Commerce One, Inc. All rights reserved.
1401
1402
1403
1404
1411

图 14

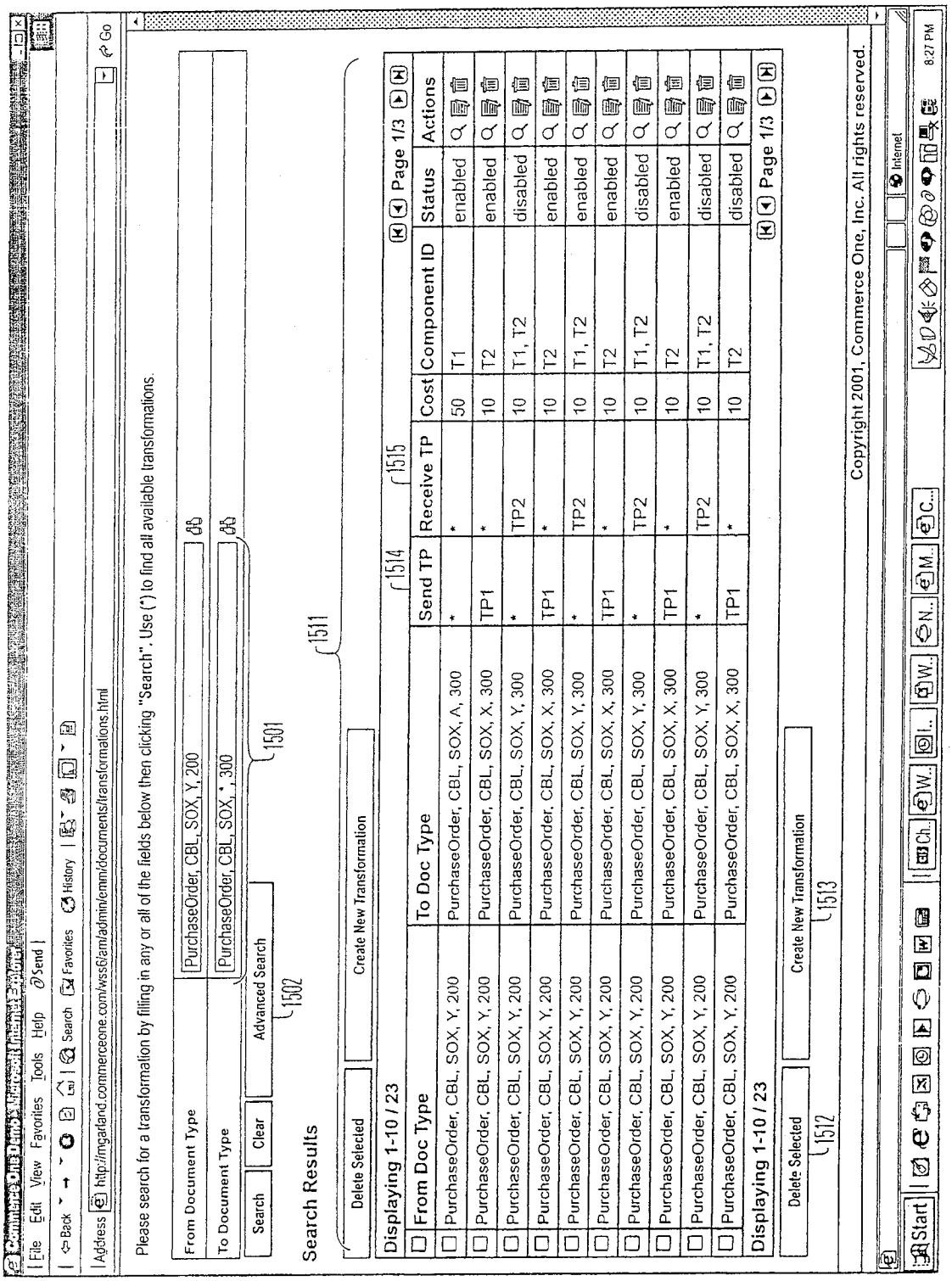

File Edit View Favorites Tools Help Send
Back Search Favorites History
Address http://mgarland.commerceone.com/wss6/am/admin/emm/documents/transformations.html
Go
Please search for a transformation by filling in any or all of the fields below then clicking "Search". Use (*) to find all available transformations.
From Document Type
PurchaseOrder, CBL, SOX, Y, 200
To Document Type
PurchaseOrder, CBL, SOX, *, 300
Search
Clear
Advanced Search
1501
1502
Search Results
1511
Delete Selected
Create New Transformation
Displaying 1-10 / 23
1514
1515
Page 1/3
From Doc Type
To Doc Type
Send TP
Receive TP
Cost
Component ID
Status
Actions
PurchaseOrder, CBL, SOX, Y, 200
PurchaseOrder, CBL, SOX, A, 300
PurchaseOrder, CBL, SOX, X, 300
PurchaseOrder, CBL, SOX, *, 300
50
10
T1
T2
T1, T2
enabled
disabled
Displaying 1-10 / 23
Page 1/3
Delete Selected
Create New Transformation
1512
1513
Copyright 2001, Commerce One, Inc. All rights reserved.
Start
Internet
8:27 PM

图 15