(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 101266666 B
(45)授权公告日 2012.08.22

(21)申请号 200810092714.4
(22)申请日 1999.10.08
(30)优先权数据
09/173,847 1998.10.16 US
09/173,854 1998.10.16 US
09/173,858 1998.10.16 US
(62)分案原申请数据
99802982.3 1999.10.08
(73)专利权人 开创网络有限公司
地址 美国特拉华州
(72)发明人 B·A·梅尔策 T·艾伦
M·D·富克斯 R·J·格鲁斯科
M·马洛尼 A·E·戴维德森
K·珀森 K·L·施瓦茨奥芙
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 钱慰民
(51)Int.Cl.
G06Q 10/06(2012.01)
(56)对比文件
JP 特開平10-162032 A,1998.06.19, 全文.
US 005737592 A,1998.04.07, 全文.
审查员 冯丹琼

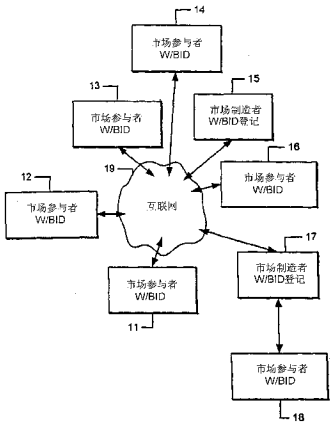
权利要求书 4 页 说明书 83 页 附图 16 页

(54)发明名称

贸易伙伴网络中的商务文档以及基于该文档的接口定义

(57)摘要

本发明涉及贸易伙伴网络中的商务文档以及基于该文档的接口定义。机器可读的文档将企业与客户、供应者和贸易伙伴联系在一起。诸如基于XML的文档等自定义电子文档很容易在贸易伙伴之间被理解。对这些电子商务文档的定义称为商业接口定义,商业接口定义公布在互联网上,或者与网络成员通信的其它地方。商业接口定义告诉潜在的贸易伙伴公司提供的服务以及当与这些服务通信时所用的文档。因此,一般的商业接口定义允许客户通过发购货定单来定购,或者允许供应者通过下载存货状态报告来检查可获得的量。另外,与公用商业库中的解释信息耦合,编制输入和输出文档,用接近并行于纸件企业运行的方式对事务编程。



CN 101266666 B

1. 一种为系统上执行的事务建立参与者接口的设备，其中所述参与者接口包括网络接口和数据处理资源，根据事务处理体系结构执行事务处理过程，其特征在于，所述设备包括：

用于提供为一特定事务参与者建立参与者接口定义的工具的装置，其中参与者接口定义包括参与者的输入文档定义和参与者的输出文档定义，输入和输出文档定义包括对各组存储单元以及各组存储单元之逻辑结构的机器可读的描述；和

用于响应输入和输出文档定义，编译与存储单元组以及输入和输出文档之逻辑结构相对应的数据结构，其中存储单元组以及输入和输出文档的逻辑结构遵从事务处理体系结构；编译系统可执行的指令，以便将输入文档翻译成相应的数据结构；以及编译系统可执行的指令，以便将事务处理过程的输出翻译成各组存储单元以及输出文档的逻辑结构的装置。

2. 如权利要求 1 所述的设备，其特征在于，用于建立参与者接口定义的工具包括用于从资源库中访问定义的元素的装置，其中所述资源库存储了由逻辑结构以及用来建立接口描述的逻辑结构解释信息所组成的库。

3. 如权利要求 2 所述的设备，其特征在于，所述资源库存储包含逻辑结构的文档定义。

4. 如权利要求 1 所述的设备，其特征在于，用于建立参与者接口定义的工具包括：

用于访问补充事务的另一个参与者接口定义的装置，被访问的定义包括补充事务的输入文档定义，以及补充事务的输出文档定义；并且

用于通过包括补充事务的输出文档定义作为正在建立的接口的输入文档的定义，来建立参与者接口定义的装置。

5. 如权利要求 4 所述的设备，其特征在于，用于建立参与者接口定义的工具包括：

用于通过包括补充事务的输入文档定义作为正在建立的接口的输出文档定义，来建立参与者接口定义的装置。

6. 如权利要求 1 所述的设备，其特征在于，用于建立参与者接口定义的工具包括用于建立一个遵从参与者接口定义的文档的装置，所述参与者接口定义包括这样的逻辑结构，它们用于存储一特定事务的标识符，并且存储关于该特定事务的输入和输出文档定义与对定义的参考两者中的至少一个。

7. 如权利要求 1 所述的设备，其特征在于，用于建立参与者接口定义的工具包括用于建立一个遵从参与者接口定义的文档的装置，所述参与者接口定义包括这样的逻辑结构，它们用于存储参与者接口的标识符，并且存储关于一个或多个由参与者接口支持的事务组成的事务组的规范说明与对规范说明的参考两者中的至少一种。

8. 如权利要求 7 所述的设备，其特征在于，遵从参与者接口定义的文档包括对机器可读的一特定事务规范说明的参考，并且该特定事务的规范说明包括一文档，该文档包括这样的逻辑结构，它们用于存储关于该特定事务的输入和输出文档定义与对定义的参考两者中的至少一个。

9. 如权利要求 8 所述的设备，其特征在于，存储单元包括作语法分析的数据。

10. 如权利要求 9 所述的设备，其特征在于，在输入和输出文档的至少一个文档中的作语法分析的数据包括：

字符数据，用于对输入和输出文档中的一个文档中的文本字符编码，和标记数据，用于

根据输入和输出文档中的一个文档的逻辑结构，识别存储单元组。

11. 如权利要求 10 所述的设备，其特征在于，至少一组存储单元对提供自然语言字的多个文本字符编码。

12. 如权利要求 10 所述的设备，其特征在于，规范说明包括用于至少一组存储单元的解释信息，解释信息对各组作语法分析的字符的定义进行编码，而存储单元由输入和输出文档中至少一个文档的逻辑结构来识别。

13. 如权利要求 9 所述的设备，其特征在于，存储单元包括未作语法分析的数据。

14. 如权利要求 1 所述的设备，其特征在于，根据不同的事务处理体系结构，对应于存储单元组以及输入和输出文档之逻辑结构的数据结构包括有包含许多变量和方法的编程对象。

15. 如权利要求 1 所述的设备，其特征在于，事务过程的不同事务处理体系结构包括一个遵从接口描述语言的过程。

16. 如权利要求 1 所述的设备，其特征在于，输入和输出文档定义包括遵从标准的可扩充标记语言 XML 的文档类型定义。

17. 如权利要求 3 所述的设备，其特征在于，输入和输出文档中的一个文档的定义包括对资源库中某个文档类型的参考。

18. 如权利要求 2 所述的设备，其特征在于，资源库包括用于规定事务的产品主题之度量的解释信息。

19. 如权利要求 2 所述的设备，其特征在于，资源库包括用于规定事务的产品主题之成本的解释信息。

20. 如权利要求 2 所述的设备，其特征在于，资源库包括用于规定事务的产品主题之特性的解释信息。

21. 如权利要求 2 所述的设备，其特征在于，资源库包括用于规定事务的金额事项的解释信息。

22. 如权利要求 2 所述的设备，其特征在于，资源库包括用于规定事务的产品主题之运送事项的解释信息。

23. 一种为系统上执行的事务建立参与者接口的方法，其中所述参与者接口包括网络接口和数据处理资源，根据事务处理体系结构执行事务处理过程，其特征在于，所述方法包括：

提供为一特定事务参与者建立参与者接口定义的工具，其中参与者接口定义包括参与者的输入文档定义和参与者的输出文档定义，输入和输出文档定义包括对各组存储单元以及各组存储单元之逻辑结构的机器可读的描述；和响应输入和输出文档定义，编译与存储单元组以及输入和输出文档之逻辑结构相对应的数据结构，其中存储单元组以及输入和输出文档的逻辑结构遵从事务处理体系结构；编译系统可执行的指令，以便将输入文档翻译成相应的数据结构；以及编译系统可执行的指令，以便将事务处理过程的输出翻译成各组存储单元以及输出文档的逻辑结构。

24. 如权利要求 23 所述的方法，其特征在于，建立参与者接口定义的工具用于从资源库中访问定义的元素，其中所述资源库存储了由逻辑结构以及用来建立接口描述的逻辑结构解释信息所组成的库。

25. 如权利要求 24 所述的方法，其特征在于，所述资源库存储包含逻辑结构的文档定义。

26. 如权利要求 23 所述的方法，其特征在于，用于建立参与者接口定义的工具用于：

访问补充事务的另一个参与者接口定义，被访问的定义包括补充事务的输入文档定义，以及补充事务的输出文档定义；并且

通过包括补充事务的输出文档定义作为正在建立的接口的输入文档的定义，来建立参与者接口定义。

27. 如权利要求 26 所述的方法，其特征在于，用于建立参与者接口定义的工具用于：

通过包括补充事务的输入文档定义作为正在建立的接口的输出文档定义，来建立参与者接口定义。

28. 如权利要求 23 所述的方法，其特征在于，用于建立参与者接口定义的工具用于建立一个遵从参与者接口定义的文档，所述参与者接口定义包括这样的逻辑结构，它们用于存储一特定事务的标识符，并且存储关于该特定事务的输入和输出文档定义与对定义的参考两者中的至少一个。

29. 如权利要求 23 所述的方法，其特征在于，用于建立参与者接口定义的工具用于建立一个遵从参与者接口定义的文档，所述参与者接口定义包括这样的逻辑结构，它们用于存储参与者接口的标识符，并且存储关于一个或多个由参与者接口支持的事务组成的事务组的规范说明与对规范说明的参考两者中的至少一种。

30. 如权利要求 29 所述的方法，其特征在于，遵从参与者接口定义的文档包括对机器可读的一特定事务规范说明的参考，并且该特定事务的规范说明包括一文档，该文档包括这样的逻辑结构，它们用于存储关于该特定事务的输入和输出文档定义与对定义的参考两者中的至少一个。

31. 如权利要求 30 所述的方法，其特征在于，存储单元包括作语法分析的数据。

32. 如权利要求 31 所述的方法，其特征在于，在输入和输出文档的至少一个文档中的作语法分析的数据包括：

字符数据，用于对输入和输出文档中的一个文档中的文本字符编码，和标记数据，用于根据输入和输出文档中的一个文档的逻辑结构，识别存储单元组。

33. 如权利要求 32 所述的方法，其特征在于，至少一组存储单元对提供自然语言字的多个文本字符编码。

34. 如权利要求 32 所述的方法，其特征在于，规范说明包括用于至少一组存储单元的解释信息，解释信息对各组作语法分析的字符的定义进行编码，而存储单元由输入和输出文档中至少一个文档的逻辑结构来识别。

35. 如权利要求 31 所述的方法，其特征在于，存储单元包括未作语法分析的数据。

36. 如权利要求 23 所述的方法，其特征在于，根据不同的事务处理体系结构，对应于存储单元组以及输入和输出文档之逻辑结构的数据结构包括有包含许多变量和方法的编程对象。

37. 如权利要求 23 所述的方法，其特征在于，事务过程的不同事务处理体系结构包括一个遵从接口描述语言的过程。

38. 如权利要求 23 所述的方法，其特征在于，输入和输出文档定义包括遵从标准的可

扩充标记语言 XML 的文档类型定义。

39. 如权利要求 25 所述的方法，其特征在于，输入和输出文档中的一个文档的定义包括对资源库中某个文档类型的参考。

40. 如权利要求 24 所述的方法，其特征在于，资源库包括用于规定事务的产品主题之度量的解释信息。

41. 如权利要求 24 所述的方法，其特征在于，资源库包括用于规定事务的产品主题之成本的解释信息。

42. 如权利要求 24 所述的方法，其特征在于，资源库包括用于规定事务的产品主题之特性的解释信息。

43. 如权利要求 24 所述的方法，其特征在于，资源库包括用于规定事务的金额事项的解释信息。

44. 如权利要求 24 所述的方法，其特征在于，资源库包括用于规定事务的产品主题之运送事项的解释信息。

贸易伙伴网络中的商务文档以及基于该文档的接口定义

[0001] 本申请是申请号为 99802982.3、国际申请日为 1999 年 10 月 8 日、发明名称为“贸易伙伴网络中的商务文档以及基于该文档的接口定义”的发明专利申请的分案申请。

[0002] 发明背景

[0003] 发明领域

[0004] 本发明涉及这样的系统和协议，它们支持与网络相连的不同客户之间的事务；尤其涉及这样的系统和协议，它们支持具有不同体系结构的平台之间的商务。

[0005] 相关技术的描述

[0006] 互联网和其它通信网在人和各计算机平台之间提供了通信的途径，这些途径正被广泛地用于各种事务，包括参与者买卖贷物和服务的商务活动。正在为方便互联网上的商务活动作各种努力。但是，面对许多竞争标准，为了执行一项事务，当事方必须预先对将要使用的协议达成一致，并且通常要求对平台体系结构进行定制集成，以支持这些事务。对某个与已达成一致的标准不相容的特定网点为内部的商业过程，可以要求进行实质的再工作，以便与其它网点集成。另外，当一家公司提交一种标准或另一个标准时，该公司会陷入一个关于交易当事方的给定标准群中，并排除其它的对象。

[0007] Tenenbaum 等人在“Eco 系统：互联网商务体系结构”中很好地概述了互联网商务发展所遇到的挑战，该论文发表在 1997 年 5 月的“计算机”杂志第 48-55 页上。

[0008] 为了在互联网上开展商务活动，需要对体系结构框架标准化。为支持这类商务框架而开发的平台包括 IBM 商务点、Microsoft 互联网商务框架、NetscapeONE（开放网络环境）、Oracle NCA（网络计算体系结构），和 Sun/JAVASoftJECF(JAVA 电子商务框架）。

[0009] 除了这些专利框架外，正在研究编程技术，诸如基于 CORBA IIOP 的公共分布式对象模型（公共对象请求中介器体系结构互联网 ORB 协议）。使用公共分布式对象模型的意图是为了简化下述的系统转移，即从企业系统转移至可以在电子商务的商业应用层面上协同工作的系统。但是，使用某个框架的用户或企业不能在另一个框架上进行交易。这限制了电子商务系统的成长。

[0010] 实施某个框架的公司将拥有一个应用程序设计接口 API，该 API 与支持其它框架的 API 不同。因此，如果不要求采用公共商务系统接口，那么公司很难访问其它的事务处理服务。在 API 层面上开发这类商务系统接口要求在在当事双方之间进行重要的合作，这通常是不现实的。

[0011] 因此，希望提供一种框架，它能够使通信网中不同平台之间的相互作用变得容易。这种系统应该方便交易伙伴间的自发贸易，不需要对全行业的标准定制集成或者达成在先协议。另外，这类系统应该鼓励增加商务自动化的途径，以避免了传统系统集成费时多，成本高及风险大的缺点。

[0012] 总的来说，希望提供一种电子商务系统，它用开放式市场代替了基于专利标准的封闭式贸易伙伴网络。

发明内容

[0013] 本发明提供了一种将企业与客户、供应者和贸易伙伴连接起来的基础结构。在本发明的基础结构下，公司用自定义的、机器可读的文档（诸如，基于可扩充标记语言 XML 的文档）交换信息和服务，而所述文档很容易在贸易伙伴之间被理解。在本文中，用于描述被交换文档的文档被称为商业接口定义 BID，这类文档公布在互联网上，或者与网络成员通信的其它地方。商业接口定义告诉潜在的贸易伙伴公司提供的服务，以及当与这类服务通信要使用的文档。因此，一般的商业接口定义允许客户通过发购货定单进行定购，其中所述购货定单遵从在某一方 BID 中公布的文档定义，以便接收此购货定单。允许供应者通过下载存货状态报告来检查可获得的量，其中所述存货状态报告遵从在用于管理存货数据的一个商务系统的 BID 中公布的文档定义。使用预定的、机器可读的商务文档为访问企业应用程序提供了更直观的、更灵活的方法。

[0014] 依照本发明，商务网络中的网点建立了一个事务接口，它包括机器可读的接口规范说明，以及机器可读的数据结构，其中机器可读的接口规范说明包括对机器可读的接口规范说明的解释信息。机器可读的接口规范说明包括对输入文档的定义以及对输出文档的定义，这些定义被网点起接口作用的事务处理过程所接受和产生。例如，按照基于标准 XML 的文档，输入和输出文档的定义包括对各组存储单元以及各组存储单元之逻辑结构的描述。根据本发明的各个方面，包括解释信息的机器可读的数据结构包括对输入和输出文档定义中逻辑结构的数据类型规范说明（例如，串，阵列等）、逻辑结构的内容模型（例如，可能值的清单），和 / 或将某一特定逻辑结构的预定存储单元组映像到一清单的各个条目以便提供逻辑结构之语义定义的数据结构（例如，将代码映像成产品名称）。

[0015] 根据本发明的其它方面，接口包括一个至少可以由网络中的一个网点访问的记忆资源库，它存储了一个由逻辑结构和逻辑结构的解释信息所组成的库。资源库可以扩充，以便包括由输入和输出文档定义组成的库、由接口规范说明组成的库，以及由网络中参与者接口网点之规范说明组成的库。

[0016] 因此，本发明事务框架中的参与者在网络的诸网点之间执行各项事务，其中所述网络包括多个网点，它们执行诸事务中所包含的过程。方法是为一事务存储机器可读的接口规范说明，而所述规范说明包括一输入文档的定义和一输出文档的定义。输入和输出文档的定义包括对各组存储单元以及各组存储单元之逻辑结构的描述。在一较佳系统中，定义的表述方式遵从标准的 XML 文档类型定义 DTD，可以通过对一些元素进行语义映像、内容模型化以及数据归类而得以扩充。事务参与者通过通信网接收包含文档的数据。参与者根据为事务存储的规范说明对文档进行语法分析，以便识别该事务的输入文档。在对文档作了语法分析之后，至少将一部分输入文档按机器可读的格式提供给一事务处理过程，以产生一输出。根据所存规范说明中输出文档的定义，形成一输出文档，该输出文档包括事务处理过程的输出。一般，在通信网上将输出文档发回输入文档的源，或者适合一特定事务类型需要的其它地方。

[0017] 因此，商业接口定义跨接了下述文档和程序之间的间隙，其中所述文档例如是根据 XML 规定的，而所述程序是在特定网点处的事务处理服务的前端上执行的。此类前端例如可以通过 JAVA 虚拟机或者通过在网络上提供系统互连的其它公共体系结构来实现。商业接口定义提供了一项技术，通过该技术，利用商业接口定义文档，可以设计事务协议。用文档型的详细形式规范说明可以建立事务协议程序。

[0018] 机器可读的事务接口规范说明可以被网络中的其它平台访问。参与者平台包括用于根据一互补网点上规定的事务接口来设计输入文档和输出文档的资源。因此，参与者网点包括用于存取互补接口的输入文档定义以及互补接口的输出文档定义的资源。通过在存储于规范说明中的接口的输入文档定义中至少包括一部分互补接口的输出文档定义，来建立关于访问参与者网点的所存储规范说明。

[0019] 根据本发明的另一方面，用于建立接口的所存储规范说明的过程包括对资源库存取机器可读的规范说明的元素。资源库存储了由逻辑结构、内容模型和逻辑结构的示意图所组成的库，以及文档的定义，其中所述文档包括用来建立接口描述的逻辑结构。可在网络中访问的资源库使得接口描述的建立变得容易，而接口描述很容易被共享。通过网络通信以及对公共逻辑结构认同很容易协商为一特定过程建立的输入文档与希望由互补过程返回的输出文档之间的任何差异，以便表达特殊的信息。

[0020] 根据本发明的一个方面，机器可读的事务接口规范说明包括一个遵从接口文档定义的文档，所述接口文档定义在网络的各成员之间共享。该接口文档定义包括这样的逻辑结构，它们用于存储一特定事务的标识符，以及关于该特定事务的输入和输出文档的定义与对定义的参考两者中的至少一种。也就是说，对一特定服务的接口描述实际上可以包含对输入和输出文档的定义。另一种方法是，包括指向资源库某个位置或者这种定义在网络中其它地方的指针。

[0021] 依照本发明的另一方面，机器可读的规范说明包括遵从接口文档定义的商业接口定义 BID 文档，而所述接口文档定义包括这样的逻辑结构，它们用于存储接口的标识符，并且存储关于一个或多个由接口支持的事务组成的事务组的规范说明与对规范说明的参考两者中的至少一种。对于每个被支持的事务，文档至少包括关于特定事务的输入和输出文档的定义和对定义的参考两者中的一种。

[0022] 依照本发明的另一方面，根据输入和输出文档的逻辑结构，在输入和输出文档定义中定义的存储单元包括作语法分析的数据，而这些数据又包括对文本字符编码的字符数据，以及用于识别存储单元组的标记数据。依照本发明的另一方面，至少一组存储单元对提供自然语言字的多个文本字符编码。这便于此类文档的人类阅读者和开发者使用输入和输出文档定义。

[0023] 依照本发明的另一方面，输入和输出规范说明包括逻辑结构所识别的多组存储单元中至少一组的解释信息。在一个例子中，解释信息对各组经分析字符的定义进行编码。在另一例中，解释信息提供了概念模型规范说明，诸如要求一具体的逻辑结构，以便携带代码清单中一个成员，其中所述代码被映像至目录的产品描述。在一些系统中，文档之逻辑结构中的存储单元包括许多组未分析的数据，这符合某种特定实施的需要。

[0024] 依照本发明的另一方面，在一特定平台中的事务处理过程具有一种事务处理体系结构，它是多种不同事务处理体系结构中的一种。因此，根据使用该信息的事务处理过程的不同事务处理体系结构，参与者网点包括用于将至少一部分输入文档翻译成一种可读格式的资源。根据事务处理过程的不同事务处理体系结构，将文档定义的诸元素翻译成包括许多变量和方法的编程对象。对于具有一个起事务处理过程前端作用的 JAVA 虚拟机的参与者来说，将文档中的特定字段翻译成 JAVA 对象，包括数据以及与 JAVA 对象相关的被获得和设置的函数。在公共对象请求中介器体系结构 CORBA、构件对象模型 COM、联机事务处理

OLTP 和电子数据交换 EDI 的意义上，本发明可支持的其它事务处理体系结构包括遵从一种接口描述语言的过程。

[0025] 依照本发明的其它方面，提供了一个资源库，它存储了在多项事务中使用的文档类型。机器可读的一特定事务的规范说明通过参考资源库的文档类型至少定义了输入文档和输出文档中的一个。依照另一方面，包含在资源库的文档类型包括用于识别网络参与者的文档类型。这类文档类型提供了用于识别参与者的结构、用于规定参与者支持的服务的结构，以及用于规定每种服务之输入和输出文档的结构。

[0026] 除了上述方法之外，本发明提供了一种用于管理网点间事务的设备，该设备包括网络接口；存储器，用于存储数据和指令程序，包括上述机器可读的事务接口规范说明；以及数据处理器，它与存储器和网络接口相连。存储在该设备中的指令程序包括逻辑电路，用以为事务参与者执行上述过程。

[0027] 本发明还提供了一种为系统上执行的事务建立参与者接口的设备，其中参与者接口包括网络接口和数据处理资源，根据事务处理体系结构执行事务处理过程。所述设备包括指令程序，它们可以由系统执行，并且存储在系统可访问的媒体上，指令程序提供了为一特定事务参与者建立参与者接口定义的工具。参与者接口定义包括输入文档定义和输出文档定义。输入和输出文档定义包括对各组存储单元以及各组存储单元之逻辑结构的机器可读的描述，依照本发明的一个方面，各组存储单元以及各组存储单元的逻辑结构遵从 XML 文档类型定义。

[0028] 依照本发明的这一方面，用于建立参与者接口的设备还包括以下指令程序，它们存储在数据处理系统可访问的媒体上，并且响应输入和输出文档定义，用于编译与存储单元组以及输入和输出文档之逻辑结构相对应的数据结构，其中存储单元组以及输入和输出文档的逻辑结构遵从事务处理体系结构；用于编译系统可执行的指令，以便将输入文档翻译成相应的数据结构；以及编译系统可执行的指令，以便将事务处理过程的输出翻译成各组存储单元和输出文档的逻辑结构。

[0029] 在一较佳实施例中，用于建立参与者接口定义的工具包括系统可执行的指令，以便从补充网点和 / 或资源库中存取定义的元素，其中所述资源库存储了由逻辑结构以及用来建立接口描述的逻辑结构解释信息组成的库。根据本发明的各个方面，资源库不仅包括逻辑结构库，而且包括包含逻辑结构的文档定义，以及用于规定参与者接口的格式。依照本发明的这一方面，根据上述技术，并结合对参与者网点的描述，为建立商业接口规范说明提供了工具。在较佳实施例的参与者网点中实现了用于建立接口的工具以及用于将接口编译成资源库的工具，其中资源库是根据本发明该方面与事务处理体系结构进行通信所需要的。并且，当网络的使用根据定义输入和输出文档的接口描述而增长时，可以用这些工具开发和优化接口描述。

[0030] 因此，本发明的另一方面提供一种设备，该设备包括存储器和数据处理器，其中数据处理器用于执行存储器中所存的指令，所述指令包括用于建立参与者接口定义的工具和用于完成上述功能的编译器。

[0031] 依照本发明的另一方面，使用参与者接口描述能够使市场制造者网点工作。在这样的网点上，提供了一种用于管理事务的方法，该方法包括存储机器可读的多个参与者接口的规范说明，所述规范说明可以识别由接口支持的事务，以及这些事务的输入和输出文

档。如上所述，输入和输出文档的定义包括对各组存储单元和各组存储单元之逻辑结构的描述，诸如根据 XML 标准。另外，事务的定义和参与者接口定义都包括根据一项技术规定的文档，所述技术遵从 XML 或其它标准化的文档表述语言。在这样的市场制造者网点上，通过通信网接口包含文档的数据。根据规范说明对文档进行语法分析，以便识别用于接受被识别输入文档的一个或多个事务中的输入文档。按机器可读的格式将至少一部分输入文档提供给与一个或多个被识别事务相关的事务处理过程。将至少一部分输入文档提供给事务处理过程的步骤包括在市场制造商网点上根据处理体系结构执行路由选择过程。根据一特定的处理体系结构，执行本发明一个方面的路由选择过程，并且将至少一部分输入文档翻译成路由选择过程的处理体系结构的格式。依照较佳实施例的翻译步骤包括根据路由选择过程的处理体系结构，产生包含许多变量的方法的编程对象。

[0032] 依照本发明的另一方面，市场制造者网点还支持资源库结构。因此，在市场制造者网点上提供一个过程，用于允许网络中的参与者能够访问存储在市场制造者网点上的资源库。如上所述，资源库包括逻辑结构的定义，解释信息，以及参与者网点用来建立事务接口文档的文档定义，并且资源库包括用来识别参与者的商业接口定义的实例，以及各参与者执行的事务。

[0033] 依照各种不同的情况，路由选择过程包括将输入文档翻译成文档将被发往的过程的不同处理体系结构，或者按原始文档格式通过网络将输入文档发送给一远程处理网点，或者发送给这类处理的组合。在另一种情况下，路由选择过程还可以包括下述技术，即将依照一个输入文档定义而定义的输入文档翻译成根据一个不同的过程文档规范说明而定义的不同文档，其中所述过程文档规范说明已经登记，以便等待该输入文档。

[0034] 另外，根据本发明将市场制造者网点提供为一种设备，该设备包括网络接口、存储器和数据处理器。其中存储器用于存储数据和指令程序，指令程序包括参与者接口的规范说明。以指令程序的形式为逻辑电路提供了数据处理器，或者如上所述执行市场制造者过程。

[0035] 因此，本发明根据共享输入和输出文档规范说明的原则，提供了电子事务的基础。文档为访问事务处理服务提供了直觉且灵活的方法，比编程 API 更容易实施。通过公司已在很大程序上达成一致的被交换的文档，而不是通过总是不同的商务系统接口，更容易使公司互连。另外，在较佳实施例中，用人可阅读的格式规定这类文档。根据本发明，在诸如 XML 文档等文档中规定商业接口，其中 XML 文档很容易被人或计算机解释。

[0036] 使用本发明的基于文档的事务体系结构，包括使用语法分析器，对于任何文档源，语法分析器的工作方式基本相同。这样做大大减少了对定制程序的需要，其中定制程序用于从网络的每个参与者那里抽取和集成信息。因此，通过以下步骤可以对由会计、购买、制造、运输和其它功能产生的企业信息进行集成。所述步骤是，首先根据本发明将每个源转换成具有一体系结构的文档，然后处理作语法分析的数据流。系统中参与市场的每个网点都只需要知道其内部系统的格式，以及为根据事务进行互换而规定的文档的格式。因此，不需要为希望参与电子商务网络的每个其它网点产生本地格式。

[0037] 为了完整的商业集成，本发明提供了一种具有标准化逻辑结构（类似 XML 元素）、属性或元数据的资源库，为特定的商业群体建立了语义语言。本发明还提供了一种在不同元数据描述之间映像的装置，以及用于处理文档并且调用合适应用程序和服务的服务器。

依照本发明，网络的基本构成块包括商业接口定义，它们告诉潜在的贸易伙伴公司提供怎样的联机服务，以及用哪些文档来调用服务；服务器，它们提供桥梁，以便将内部和外部的事务处理服务组束缚在一起，从而建立一个贸易群体。服务器的作用是对输入文档进行语法分析，并且调用合适的服务。另外，本发明的服务器承担翻译任务，将被接收和被发送的文档格式翻译成各主系统的格式。因此，贸易伙伴只需要对被交换的商务文档的结构、内容和序列达成一致，不需要对应用程序设计接口的细节达到一致。如何处理文档以及因接收文档而引起的行为严格地由提供服务的企业决定。这将集成从系统层次提高至企业层次。这使得企业可以将一清楚且稳定的接口呈现给它的对手，不管其内部技术实施、组织或过程如何变化。

[0038] 通过使用公共商业库或资源库，方便了建立商业接口定义并使服务器根据这些描述管理商业活动的整个过程，其中公共商业库或资源库包括一般商业概念的信息模型，这些模型包括诸如公司、服务和产品等商业描述原语，诸如目录、购货定单和发票等商业形式，以及包括时间和日期、位置以及货物分类等标准度量。

[0039] 通过附图、详细描述和后面的权利要求书，将清楚本发明的其它方面。

[0040] 附图概述

[0041] 图 1 是一简化图，示出了依照本发明的包括商业接口定义 BID 的电子商务网。

[0042] 图 2 是一简化图，示出了依照本发明的商业接口定义文档。

[0043] 图 3 是一概念性的方框图，示出了本发明网络中在一参与者网点处的服务器。

[0044] 图 4 是一流程图，示出了依照本发明的在一参与者网点处对接收文档的处理过程。

[0045] 图 5 是一方框图，示出了基于 XML 系统的语法分析器和事务处理过程前端。

[0046] 图 6 是一概念图，示出了分析函数的流程。

[0047] 图 7 是一简化图，示出了服务器的资源，这些资源用来建立本发明的商业接口定义。

[0048] 图 8 是一简化图，示出了依照本发明用来建立商业接口定义的资源库。

[0049] 图 9 是一流程图，示出了依照本发明建立商业接口定义的过程。

[0050] 图 10 提供了对本发明资源库的启发式观察。

[0051] 图 11 是一简化图，示出了服务器的资源，这些资源根据商业接口定义为本发明的网络提供了市场制造者的功能。

[0052] 图 12 是一流程图，示出了接收文档的市场制造者网点处理。

[0053] 图 13 是一流程图，示出了依照本发明在一市场制造者网点处登记参与者的过程。

[0054] 图 14 是一流程图，依照图 9 过程在一市场制造者网点处提供服务规范说明的过程。

[0055] 图 15 是一示意图，示出了依照本发明在一参与者或市场制造者网点处的操作序列。

[0056] 图 16 是一概念图，示出了依照本发明的、基于 BID 的商务网要素。

[0057] 详细描述

[0058] 针对各附图对本发明进行详细描述，其中图 1 示出了由市场参与者和市场制造者组成的网络，该网络基于使用商业接口定义，并且支持对依照这类接口描述规定的输入和

输出文档进行交易。网络包括多个网点 11-18，它们通过诸如互联网 19 或其它电信或数据通信网等通信网互联。每个网点 11-19 都包括一计算机，诸如便携式计算机、台式个人计算机、工作站、系统网络、或者其它数据处理资源。网点包括存储器，用于存储商业接口定义；处理器，用于执行与网络中其它网点进行商务交易的事务处理过程；计算机程序，它们由支持这些服务的的处理器执行。另外，每个网点包括一个网络接口，用于提供与互联网 19 或其余通信网的通信。

[0059] 在图 1 的环境中，网点 11、12、14、16 和 18 表示市场参与者。市场参与者包括要根据本发明建立的商务处理交易货品或服务的消费者或供应者。

[0060] 在本例中，网点 15 和 17 是市场制造者网点。市场制造者网点包括用于登记商业接口定义的资源，称为 BID 登记。参与者能够向市场制造者网点发送文档，而在市场制造者网点处，识别该文档并将文档传送给已登记可以接收这些文档作为输入的合适的参与者。市场制造者还保持标准形式的资源库，以方便商务网，这里标准形式的资源库构成一公共的商用库，用于建立商业接口定义。

[0061] 在本例中，市场参与者 18 直接与市场制造者 17 相连，而不通过互联网 19 连接。这种与市场制造者的直接连接表示支持商务交易的网络结构可以是各种各样的，包括诸如互联网 19 等公用网，以及诸如局域网的专用连接，或者如网点 17 和 18 之间所示的点对点连接。实际通信网变化很多，并且适用于本发明的商务交易网。

[0062] 图 2 是一启发图，示出了商业接口定义 BID 中的嵌套结构，其中商业接口定义 BID 是为本发明网络中的市场参与者建立的。图 2 所示的商业接口定义是一数据结构，它包括按照文档结构（诸如 XML 文档类型定义 DTD）的形式定义布置的逻辑结构和存储单元。图 2 的结构包括用于识别一方的第一逻辑结构 200。与逻辑结构 200 相关的是，嵌套了用于承载名称 201、物理地址 202、网络地址或单元 203、以及一组服务事务 204 的逻辑结构。对于服务组中的每件事务，提供一接口定义，包括事务 BID 205、事务 BID 206 和事务 BID 207。在诸如事务 BID 205 等每个事务 BID 内，提供这样的逻辑结构，它们用于包含名称 208、在网络上进行服务的位置 209、由服务执行的操作 210，以及一组由标签 211 表示的输入文档。另外，服务 BID 205 包括一组由标签 212 表示的输出文档。输入文档组 211 包括服务所响应的每个输入文档的商业接口定义，包括输入文档商业接口定义 213、214 和 215。每个输入文档商业接口定义包括名称 216、网络上可以找到文档描述的位置 217，以及如字段 218 表示的在文档中携带的模块。类似地，输出文档组 212 包括输出文档的接口定义，包括输出文档 BID 219、输出文档 BID 220 以及输出文档 BID 221。对于每个输出文档 BID，规定了名称 222、网络上或其它地方的位置 223、以及文档模块 224。图 2 所示的参与者的商业接口定义包括逻辑结构的实际定义，它们将用于各种服务的输入和输出文档，或者可以找到这些定义位置的的指针或其它参考。

[0063] 尽管可以使用其它文档定义结构，但在较佳系统中，图 2 的文档列在 XML 文档类型定义 DTD 中。另外，事务 BID、输入文档 BID 和输出文档 BID 都是根据 XML 文档类型定义来规定的。XML 型文档是一例基于分析数据的系统，所述分析数据包括标记数据和字符数据。标记数据识别文档内的逻辑结构，字符数据组识别逻辑结构的内容。另外，文档中还携带未分析的数据，用于各种目的。例如，参见 WC3XML 工作组在 WWW.W3.ORG/TR/1998/REC-XML-19980210 上公布的可扩充标记语言 XML1.0REC-XML-19980210 的规范说明。

[0064] 因此，在一例示的系统中，网络中的参与者网点通过将商业系统和服务与商业活动接受和产生的 XML 编码文档互连，建立了虚拟的公司。例如一特定服务的商业接口定义建立以下内容：如果接收到一个与请求目录输入的 BID 相匹配的文档，那么将返回一个与目录输入的 BID 相匹配的文档。另外，如果接收到一个与定单 BID 相匹配的文档，并且可被接收终端接受，那么将返回与发票相匹配的文档。在 XML 文档输入本地商业系统之前，网络中的网点处理这些文档，其中本地商业系统是根据网络中任何给定系统的各种事务处理结构而建立的。因此，系统拆开相关的文档组（诸如，经 MIMI 编码的 XML 文档组），分析它们，产生“标记消息”流。将消息传送到例如使用下述事件收听器模型的适合的应用程序和服务。

[0065] 用 XML 语言对商业服务之间交换的文档编码，其中 XML 语言由构成块的资源库建立，它是一种公用的商业语言，用该语言可以建立更复杂的文档定义。资源库存储了解释信息模块，着重于商业领域公共的功能和信息，包括诸如公司、服务和产品等商业描述原语；诸如目录、定单和发票等商业形式；诸如时间、日期的地址等标准度量；分类码；以及为 XML 文档中的逻辑结构提供解释信息的其它东西。

[0066] 商业接口定义是一种高级文档，它起模式作用，用于根据本发明设计交易文档的接口。因此，商业接口定义在 XML 规定的文档和在一特定网点的事务处理服务的前端上执行的程序之间的间隙上架起了一座桥梁。这些前端可以用 JAVA 虚拟机或者其它使系统在网络上互连的普通结构来实现。因此，商业接口定义提供了一项技术，通过该技术，利用商业接口定义文档可以对事务处理协议编程。用文档型的详细形式规范建立事务处理协议的程序。

[0067] 下面描述一例基于市场参与者文档的商业接口定义 BID，其中所述市场参与者文档符合 XML 格式。市场参与者 DTD 对市场参与者的商业信息分组，使联系和地址信息与一种服务和金额信息的描述相关联。这种商业信息包括名称、代码、地址、用于描述商业机构的专用分类机制，以及用于商业术语的指针。另外，由市场参与者 DTD 标识的服务将规定希望参与者对之响应并产生的输入和输出文档。因此，下面是用例举的公用商业语言为市场参与者 DTD、服务 DTD 和事务处理文档 DTD（按 XML 规定）制定模式的文档，包括说明性的评述。

[0068] 市场参与者实例

[0069] <！ DOCTYPE SCHEMA SYSTEM″ bidl.dtd″ >

[0070] <SCHEMA>

[0071] <H1>Market Participant Sample BID</H1>

[0072] <META

[0073] WHO.OWNS =″ Veo Systems″ WHO.COPYRIGHT =″ Veo Systems″

[0074] WHEN.COPYRIGHT =″ 1998″ DESCRIPTION =″ Sample BID″

[0075] WHO.CREATED =″ *″ WHEN.CREATED =″ *″

[0076] WHAT.VERSION =″ *″ WHO.MODIFIED =″ *″

[0077] WHEN.MODIFIED =″ *″ WHEN.EFFECTIVE =″ *″

[0078] WHEN.EXPIRES =″ *″ WHO.EFFECTIVE =″ *″

[0079] WHO.EXPIRES =″ *″ >

[0080] </META>

[0081] <PROLOG>

[0082] <XMLDECL STANDALONE =" no" ></XMLDECL>

[0083] <DOCTYPE NAME =" market.participant" >

[0084] <SYSTEM>markpart.dtd</SYSTEM></DOCTYPE>

[0085] </PROLOG>

[0086] <DTD NAME =" markpart.dtd" >

[0087] <H2>Market Participant</H2>

[0088] <H3>Market Participant</H3>

[0089] <ELEMENTTYPE NAME =" market.participant" >

[0090] <EXPLAIN><TITLE>A Market Participant</TITLE>

[0091] <SYNOPSIS>A business or person and its service interfaces.</SYNOPSIS>

[0092] <P>A market participant is a document definition that is created to describe a business and at least one

[0093] person with an email address, and it presents a set of pointers to service interfaces located on the network.

[0094] In this example, the pointers have been resolved and the complete BID is presented

[0095] here.</P></EXPLAIN>

[0096] <MODEL><CHOICE>

[0097] <ELEMENT NAME =" business" ></ELEMENT>

[0098] <ELEMENT NAME =" person" ></ELEMENT>

[0099] </CHOICE></MODEL></ELEMENTTYPE>

[0100] <H3>Party Prototype</H3>

[0101] <PROTOTYPE NAME =" party" >

[0102] <EXPLAIN><TITLE>The Party Prototype</TITLE></EXPLAIN>

[0103] <MODEL><SEQUENCE>

[0104] <ELEMENT NAME =" party.name" OCCURS =" +" ></ELEMENT>

[0105] <ELEMENT NAME =" address.set" ></ELEMENT>

[0106] </SEQUENCE></MODEL>

[0107] </PROTOTYPE>

[0108] <H3>Party Types</H3>

[0109] <ELEMENTTYPE NAME =" business" >

[0110] <EXPLAIN><TITLE>A Business</TITLE>

[0111] <SYNOPSIS>A business(party)with a business number attribute.</SYNOPSIS>

[0112] <P>This element inherits the content model of the party prototype and adds a business number

[0113] attribute,which serves as a key for database lookup.The business number may be used as a cross-reference

[0114] to/from customer id,credit limits,contacts lists,etc.</P></EXPLAIN>

[0115] <EXTENDS HREF =" party" >

[0116] <ATTDEF NAME =" business.number" ><REQUIRED></REQUIRED></ATTDEF>

[0117] </EXTENDS>

[0118] </ELEMENTTYPE>

[0119] <H3>Person Name</H3>

[0120] <ELEMENTTYPE NAME =" person" >

[0121] <EXPLAIN><TITLE>A Person</TITLE></EXPLAIN>

[0122] <EXTENDS HREF =" party" >

[0123] <ATTDEF NAME =" SSN" ><IMPLIED></IMPLIED></ATTDEF>

[0124] </EXTENDS>

[0125] </ELEMENTTYPE>

[0126] <H3>Party Name</H3>

[0127] <ELEMENTTYPE NAME =" party.name" >

[0128] <EXPLAIN><TITLE>A Party' s Name</TITLE>

[0129] <SYNOPSIS>A party' s name in a string of character.</SYNOPSIS></EXPLAIN>

[0130] <MODEL><STRING></STRING></MODEL>

[0131] </ELEMENTTYPE>

[0132] <H3>Address Set</H3>

[0133] <ELEMENTTYPE NAME =" address.set" >

[0134] <MODEL><SEQUENCE>

[0135] <ELEMENT NAME =" address.physical" ></ELEMENT>

[0136] <ELEMENT NAME =" telephone" OCCURS =" *" ></ELEMENT>

[0137] <ELEMENT NAME =" fax" OCCURS =" *" ></ELEMENT>

[0138] <ELEMENT NAME =" email" OCCURS =" *" ></ELEMENT>

[0139] <ELEMENT NAME =" internet" OCCURS =" *" ></ELEMENT>

[0140] </SEQUENCE></MODEL>

[0141] </ELEMENTTYPE>

[0142] <H3>Physical Address</H3>

[0143] <ELEMENTTYPE NAME =" address.physical" >

[0144] <EXPLAIN><TITLE>Physical Address</TITLE>

[0145] <SYNOPSIS>The street address,city,state,and zip code.</SYNOPSIS></EXPLAIN>

[0146] <MODEL><SEQUENCE>

[0147] <ELEMENT NAME =" street" ></ELEMENT>

[0148] <ELEMENT NAME =" city" ></ELEMENT>

[0149] <ELEMENT NAME ＝" state" ></ELEMENT>

[0150] <ELEMENT NAME ＝" postcode" OCCURS ＝" ? " ></ELEMENT>

[0151] <ELEMENT NAME ＝" country" ></ELEMENT>

[0152] </SEQUENCE></MODEL>

[0153] </ELEMENTTYPE>

[0154] <H3>Street</H3>

[0155] <ELEMENTTYPE NAME ＝" street" >

[0156] <EXPLAIN><TITLE>Street Address</TITLE>

[0157] <SYNOPSIS>Street or postal address.</SYNOPSIS></EXPLAIN>

[0158] <MODEL><STRING></STRING></MODEL>

[0159] </ELEMENTTYPE>

[0160] <H3>City</H3>

[0161] <ELEMENTTYPE NAME ＝" city" >

[0162] <EXPLAIN><TITLE>City Name or Code</TITLE>

[0163] <P>The city name or code is a string that contains sufficient information to identify a city within a

[0164] designated state.</P>

[0165] </EXPLAIN>

[0166] <MODEL><STRING></STRING></MODEL>

[0167] </ELEMENTTYPE>

[0168] <H3>State</H3>

[0169] <ELEMENTTYPE NAME ＝" state" >

[0170] <EXPLAIN><TITLE>State,Province or Prefecture Name or Code</TITLE>

[0171] <P>The state name or code contains sufficient information to identify a state within a designated

[0172] country.</P></EXPLAIN>

[0173] <MODEL><STRING DATATYPE ＝" COUNTRY.US.SUBENTITY " ></STRING></MODEL>

[0174] </ELEMENTTYPE>

[0175] <H3>Postal Code</H3>

[0176] <ELEMENTTYPE NAME ＝" postcode" >

[0177] <EXPLAIN><TITLE>Postal Code</TITLE>

[0178] <P>A postal code is an alphanumeric code,designated by an appropriate postal authority,that is used

[0179] to identify a location or region within the jurisdiction of that postal authority.Postal authorities include

[0180] designated national postal authorities.</P></EXPLAIN>

[0181] <MODEL><STRING DATATYPE ＝" string" ></STRING></MODEL>

[0182] </ELEMENTTYPE>

[0183] <H3>Country</H3>

[0184] <ELEMENTTYPE NAME ＝" country" >

[0185] <EXPLAIN><TITLE>Country Code</TITLE>

[0186] <P>A country code is a two-letter code, designated by ISO, that is used to uniquely identify a

[0187] country.</P></EXPLAIN>

[0188] <MODEL><STRING DATATYPE ＝" country" ></STRING></MODEL>

[0189] </ELEMENTTYPE>

[0190] <H3>Network Addresses</H3>

[0191] <ELEMENTTYPE NAME ＝" telephone" >

[0192] <EXPLAIN><TITLE>Telephone Number</TITLE>

[0193] <P>A telephone number is a string of alphanumerics and punctuation that uniquely identifies a

[0194] telephone service terminal, including extension number.</P></EXPLAIN>

[0195] <MODEL><STRING></STRING></MODEL>

[0196] </ELEMENTTYPE>

[0197] <H3>Fax</H3>

[0198] <ELEMENTTYPE NAME ＝" fax" >

[0199] <EXPLAIN><TITLE>Fax Number</TITLE>

[0200] <P>A fax number is a string of alphanumerics and punctuation that uniquely identifies a fax service

[0201] terminal.</P>

[0202] </EXPLAIN>

[0203] <MODEL><STRING></STRING></MODEL>

[0204] </ELEMENTTYPE>

[0205] <H3>Email</H3>

[0206] <ELEMENTTYPE NAME ＝" email" >

[0207] <EXPLAIN><TITLE>Email Address</TITLE>

[0208] <P>An email address is a datatype-constrained string that uniquely identifies a mailbox on a

[0209] server.</P></EXPLAIN>

[0210] <MODEL><STRING DATATYPE ＝" email" ></STRING></MODEL>

[0211] </ELEMENTTYPE>

[0212] <H3>Internet Address</H3>

[0213] <ELEMENTTYPE NAME ＝" internet" >

[0214] <EXPLAIN><TITLE>Internet Address</TITLE>

[0215] <P>An Internet address is a datatype-constrained string that uniquely identifies a resource on the

[0216] Internet by means of a URL.</P></EXPLAIN>

[0217] <MODEL><STRING DATATYPE =" url" ></STRING></MODEL>

[0218] </ELEMENTTYPE>

[0219] </DTD>

[0220] </SCHEMA>

[0221] 服务描述实例

[0222] <! DOCTYPE schema SYSTEM" bidl.dtd" >

[0223] <SCHEMA>

[0224] <H1>Service Description Sample BID</H1>

[0225] <META

[0226] WHO.OWNS =" Veo Systems" WHO.COPYRIGHT =" Veo Systems"

[0227] WHEN.COPYRIGHT =" 1998" DESCRIPTION =" Sample BID"

[0228] WHO CREATED =" *" WHEN.CREATED =" *"

[0229] WHAT.VERSION =" *" WHO.MODIFIED =" *"

[0230] WHEN.MODIFIED =" *" WHEN.EFFECTIVE =" *"

[0231] WHEN.EXPIRES =" *" WHO.EFFECTIVE =" *"

[0232] WHO.EXPIRES =" *" >

[0233] </META>

[0234] <PROLOG>

[0235] <XMLDECL STANDALONE =" no" ></XMLDECL>

[0236] <DOCTYPE NAME =" service" >

[0237] <SYSTEM>service.dtd</SYSTEM></DOCTYPE>

[0238] </PROLOG>

[0239] <DTD NAME =" service.dtd" >

[0240] <H2>Services</H2>

[0241] <H3>Includes</H3>

[0242] <! --INCLUDE><SYSTEM>comments.bim</SYSTEM></INCLUDE-->

[0243] <H3>Service Set</H3>

[0244] <ELEMENTTYPE NAME =" service.set" >

[0245] <EXPLAIN><TITLE>Service Set</TITLE>

[0246] <SYNOPSIS>A set of services.</SYNOPSIS></EXPLAIN>

[0247] <MODEL>

[0248] <ELEMENT NAME =" service" OCCURS =" +" ></ELEMENT>

[0249] </MODEL></ELEMENTTYPE>

[0250] <H3>Services Prototype</H3>

[0251] <PROTOTYPE NAME =" prototype.service" >

[0252] <EXPLAIN><TITLE>Service</TITLE></EXPLAIN>

[0253] <MODEL><SEQUENCE>

[0254] <ELEMENT NAME =" service.name" ></ELEMENT>

[0255] <ELEMENT NAME =" service.terms" OCCURS =" +" ></ELEMENT>

[0256] <ELEMENT NAME =" service.location" OCCURS =" +" ></ELEMENT>

[0257] <ELEMENT NAME =" service.operation" OCCURS =" +" ></ELEMENT>

[0258] </SEQUENCE></MODEL>

[0259] < ! --ATTGROUP><IMPLEMENTS

[0260] HREF =" common.attrib" ></IMPLEMENTS></ATTGROUP-->

[0261] </PROTOTYPE>

[0262] <H3>Service</H3>

[0263] <INTRO><P>A service is an addressable network resource that provides interfaces to specific

[0264] operations by way of input and output documents.</P></INTRO>

[0265] <ELEMENTTYPE NAME =" service" >

[0266] <EXPLAIN><TITLE>Service</TITLE>

[0267] <P>A service is defined in terms of its name, the location(s)at which the service is available, and the

[0268] operation(s)that the service performs.</P></EXPLAIN>

[0269] <MODEL><SEQUENCE>

[0270] <ELEMENT NAME =" service.name" ></ELEMENT>

[0271] <ELEMENT NAME =" service.location" ></ELEMENT>

[0272] <ELEMENT NAME =" service.operation" OCCURS =" +" ></ELEMENT>

[0273] <ELEMENT NAME =" service.terms" ></ELEMENT>

[0274] </SEQUENCE></MODEL>

[0275] </ELEMENTTYPE>

[0276] <H3>Service Name</H3>

[0277] <ELEMENTTYPE NAME =" service.name" >

[0278] <EXPLAIN><TITLE>Service Name</TITLE>

[0279] <P>The service name is a human-readable string that ascribes a moniker for a service.It may be

[0280] employed is user interfaces and documentation,or for other purposes.</P></EXPLAIN>

[0281] <MODEL><STRING></STRING></MODEL>

[0282] </ELEMENTTYPE>

[0283] <H3>Service Location</H3>

[0284] <ELEMENTTYPE NAME =" service.location" >

[0285] <EXPLAIN><TITLE>Service Location</TITLE>

[0286] <SYNOPSIS>A URI of a service.</SYNOPSIS>

[0287] <P>A service location is a datatype-constrained string that locates a service on the Internet by means

[0288] of a URI.</P></EXPLAIN>

[0289] <MODEL><STRING DATATYPE =" url" ></STRING></MODEL>

[0290] </ELEMENTTYPE>

[0291] <H3>Service Operations</H3>

[0292] <INTRO><P>A service operation consists of a name, location and its interface, as identified by the

[0293] type of input document that the service operation accepts and by the type of document that it will return as

[0294] a result.</P></INTRO>

[0295] <ELEMENTTYPE NAME ＝" service.operation" >

[0296] <EXPLAIN><TITLE>Service Operations</TITLE>

[0297] <P>A service operation must have a name, a location, and at least one document type as an input, with

[0298] one or more possible document types returned as a result of the operation.</P>

[0299] </EXPLAIN>

[0300] <MODEL><SEQUENCE>

[0301] <ELEMENT NAME ＝" service.operation.name" ></ELEMENT>

[0302] <ELEMENT NAME ＝" service.operation.location" ></ELEMENT>

[0303] <ELEMENT NAME ＝" service.operation.input" ></ELEMENT>

[0304] <ELEMENT NAME ＝" service.operation.output" ></ELEMENT>

[0305] </SEQUENCE></MODEL>

[0306] </ELEMENTTYPE>

[0307] <H3>Service Operation Name</H3>

[0308] <ELEMENTTYPE NAME ＝" service.operation.name" >

[0309] <EXPLAIN><TITLE>Service Operation Name</TITLE>

[0310] <P>The service operation name is a human-readable string that ascribes a moniker to a service

[0311] operation. It may be employed in user interfaces and documentation, or for other

[0312] purposes.</P></EXPLAIN>

[0313] <MODEL><STRING></STRING></MODEL>

[0314] </ELEMENTTYPE>

[0315] <H3>Service Operation Location</H3>

[0316] <INTRO><P>The service location is a network resource. That is to say, a URI.</P></INTRO>

[0317] <ELEMENTTYPE NAME ＝" service.operation.location" >

[0318] <EXPLAIN><TITLE>Service Operation Location</TITLE>

[0319] <SYNOPSIS>A URI of a service operation.</SYNOPSIS>

[0320] <P>A service operation location is a datatype-constrained string that locates a service operation on the

[0321] Internet by means of a URL.</P></EXPLAIN>

[0322] <MODEL><STRING DATATYPE =" url" ></STRING></MODEL>

[0323] </ELEMENTTYPE>

[0324] <H3>Service Operation Input Document</H3>

[0325] <INTRO><P>The input to a service operation is defined by its input document type.That is,the

[0326] service operation is invoked when the service operation location receives an input document whose type

[0327] corresponds to the document type specified by this element.</P>

[0328] <P>Rather than define the expected input and outPut document types in the market participant

[0329] document,this example provides pointers to externally-defined BIDs. This allows reuse of the same BID

[0330] as the input and/or output document type for multiple operations.In addition,it encourages parallel design

[0331] and implementation.</P></INTRO>

[0332] <ELEMENTTYPE NAME =" service.operation.input" >

[0333] <EXPLAIN><TITLE>Service Operation Input</TITLE>

[0334] <SYNOPSIS>Identifies the type of the service operation input document.</SYNOPSIS>

[0335] <P>Service location input is a datatype-constrained string that identifies a BID on the Internet by

[0336] means of a URI.</P>

[0337] </EXPLAIN>

[0338] <MODEL><STRING DATATYPE =" url" ></STRING></MODEL>

[0339] </ELEMENTTYPE>

[0340] <H3>Service Operation Output Document Type</H3>

[0341] <INTRO><P>The output of a service operation is defined by its output document type(s).That is,the

[0342] service operation is expected to emit a document whose type corresponds to the document type specified

[0343] by this element.</P></INTRO>

[0344] <ELEMENTTYPE NAME =" service.operation.output" >

[0345] <EXPLAIN><TITLE>Service Operation Output</TITLE>

[0346] <SYNOPSIS>Identifies the type of the service operation output document.</SYNOPSIS>

[0347] <P>Service location output is a datatype-constrained string that identifies a BID on the Internet by

[0348] means of a URI.</P>

[0349] </EXPLAIN>

[0350] <MODEL><STRING DATATYPE ＝" url" ></STRING></MODEL>

[0351] </ELEMENTTYPE>

[0352] <H3>Service Terms</H3>

[0353] <INTRO><P>This is a simple collection of string elements,describing the terms of an

[0354] agreement.</P></INTRO>

[0355] <ELEMENTTYPE NAME ＝" service.terms" >

[0356] <EXPLAIN><TITLE>Service Terms</TITLE>

[0357] <SYNOPSIS>Describes the terms of a given agreement.</SYNOPSIS>

[0358] </EXPLAIN>

[0359] <MODEL><STRING DATATYPE ＝" string" ></STRING></MODEL>

[0360] </ELEMENTTYPE>

[0361] </DTD>

[0362] </SCHEMA>

[0363] 可以如下所示,用公用商业语言资源库中的服务型元素扩充服务 DTD 模式。

[0364] <！ ELEMENT service.type EMPTY>

[0365] <！ ATTLIST service.type

[0366] service.type.name(

[0367] catalog.operator

[0368] |commercial.directory.operator

[0369] |eft.services.provider

[0370] |escrower

[0371] |fulfillment.service

[0372] |insurer

[0373] |manufacturer

[0374] |market.operator

[0375] |order.originator

[0376] |ordering.service

[0377] |personal.services.provider

[0378] |retailer

[0379] |retail.aggregator

[0380] |schema.resolution.service

[0381] |service.provider

[0382] |shipment.acceptor

[0383] |shipper

[0384] |van

[0385] |wholesale.aggregator

[0386])#REQUIRED

[0387] % common.attrib ;

[0388] >

[0389] 上述服务型元素说明了由商业接口定义携带的解释信息，在本例中，一种内容形式允许识别一有效服务清单中的任何一个。其它解释信息包括数据类型，例如元素<H3>Internet Address</H3>，它包括内容形式“url”并用数据类型“串”表示。还有其它解释信息包括代码至清单元素的映像，例如元素 <H3>State</H3>，它包括对文件“COUNTRY.US.SUBENTITY”中状态进行映像的代码。

[0390] 由市场参与者 DTD 提到的服务描述定义了当完成服务时该服务接受并产生的文档。下面将基本服务描述指定为 XML 文档 transact.dtd。

[0391] Transact.dtd 模拟诸如发票等事务处理描述，或者对值交换的描述。这种文档类型支持许多种使用，因此事务描述元素具有允许用户区分发票、性能、表示出售和请求报价等属性。交换可以在两个以上的对象之间发生，但只取出两个对象，发价人和计算方，这两个对象都用指针表示，指针指向与上述市场参与者 DTD 相符的文档。计算方的指针是可选择的，以便适应出售意向。交换描述在以下所列的模块 tranprim.mod 中描述，并且包括定价和小计。在交换描述之后，可以提供作为整体应用于事务处理的费用，并且必须提供总的费用。因此，以下给出了本例的事务描述概要文档 :

[0392] <! --transact.dtd Version:1.0-->

[0393] <! --Copyright 1998 Veo Systems, Inc.-->

[0394] <! ELEMENT transaction.description(meta ? , issuer.pointer,

[0395] counterparty.pointer ? , exhange.descrption+, general.charges ? ,

[0396] net.total ?)>

[0397] <! ATTLIST transaction.description

[0398] transaction.type(invoice|pro.forma|offer.to sell|order

[0399] |request.for.quote|request.for.bid

[0400] |request.for.proposal|response.to.request.for.quote

[0401] |response.to.request.for.bid

[0402] |response.to.request.for.proposal)" invoice"

[0403] % common.attrib ;

[0404] % altrep.attrib ;

[0405] % ttl.attrib ;

[0406] >

[0407] 以下是根据上述定义建立的代表性的市场参与者和服务 DTD :

[0408] 市场参与者 DTD

[0409] <! ELEMENT business(party.name+, address.set)>

[0410] <! ATTLIST business business.number CDATA #REQUIRED

[0411] >

[0412] <! ELEMENT party.name(#PCDATA)>

[0413] <! ELEMENT city(#PCDATA)>

[0414] <! ELEMENT internet(#PCDATA)>

[0415] <！ ELEMENT country(#PCDATA)>

[0416] <！ ELEMENT state(#PCDATA)>

[0417] <！ ELEMENT email(#PCDATA)>

[0418] <！ ELEMENT address.physical(street，city，state，postcode ？，country)>

[0419] <！ ELEMENT telephone(#PCDATA)>

[0420] <！ ELEMENT person(party.name+，address.set)>

[0421] <！ ATTLIST person SSN CDATA #IMPLIED

[0422] >

[0423] <！ ELEMENT fax(#PCDATA)>

[0424] <！ ELEMENT street(#PCDATA)>

[0425] <！ ELEMENT address.set(address.physical，telephone*，fax*，email*，internet*).>

[0426] <！ ELEMENT postcode(#PCDATA)>

[0427] <！ ELEMENT market.participant(business|person)>

[0428] 服务 DTD

[0429] <！ ELEMENT service.location(#PCDATA)>

[0430] <！ ELEMENT service.terms(#PCDATA)>

[0431] <！ ELEMENT service.operation.name(#PCDATA)>

[0432] <！ ELEMENT service.operation (service.operation.name，service.operation.location，

[0433] service.operation.input，service.operation.output)>

[0434] <！ ELEMENT service(service.name，service.location，service.operation+，service.terms)>

[0435] <！ ELEMENT service.operation.input(#PCDATA)>

[0436] <！ ELEMENT service.operation.location(#PCDATA)>

[0437] <！ ELEMENT service.name(#PCDATA)>

[0438] <！ ELEMENT service.set(service+)>

[0439] <！ ELEMENT service.operation.output(#PCDATA)>

[0440] 以下是根据 transact.dtd 产生的一例文档：

[0441] <？ xml version ＝″ 1.0″ ？ >

[0442] <！ --rorder.xml Version:1.0-->

[0443] <！ --Copyright 1998 Veo Systems，Inc.-->

[0444] <！ DOCTYPE transaction.desc ription SYSTEM″ urn:x-

[0445] veosystems:dtd:cbl:transact:1.0:>

[0446] <transaction.description transaction.type ＝″ order″ >

[0447] <meta>

[0448] <urn ？ urn:x-veosystems:doc:00023

[0449] </urn>

[0450] <thread.id party.assigned.by =" reqorg" >FRT876
[0451] </thread.id>
[0452] </meta>
[0453] <issuer.pointer>
[0454] <xll.locator urllink = " reqorg.xml" >Customer
[0455] Pointer
[0456] </xll.locator>
[0457] </issuer.pointer>
[0458] <counterparty.pointer>
[0459] <xll.locator urllink =" compu.xml" >Catalog entry owner
[0460] pointer
[0461] </xll.locator>
[0462] </counterparty.pointer>
[0463] <exchange.description>
[0464] <line.item>
[0465] <product.instance>
[0466] <product.description.pointer>
[0467] <xll.locator urllink =" cthink.xml" >Catalogue Entry Pointer
[0468] </xll.locator>
[0469] </product.desc ription.pointer>
[0470] <product.specifics>
[0471] <info.description.set>
[0472] <info.description>
[0473] <xml.descriptor>
[0474] <doctype>
[0475] <dtd system.id =" urn:x-veosystems:dtd:cbl:gprod:1.0" />
[0476] </doctype>
[0477] <xml.descriptor.details>
[0478] <xll.xptr.frag>DESCENDANT(ALL,os)STRING(" Windows 95")
[0479] </xll.xptr.frag>
[0480] <xll.xptr.frag>DECENDANT(ALL,p.speed)STRING(" 200")
[0481] </xll.xptr.frag>
[0482] <xll.xptr.frag>DESCENDANT(ALL,hard.disk.capacity)
[0483] STRING(" 4")
[0484] </xll.xptr.frag>
[0485] <xll.xptr.frag>DESCENDANT(ALL,d.size)STRING(" 14.1")

[0486] </xll.xptr.frag>
[0487] </xml.descriptor.details>
[0488] </xml.descriptor>
[0489] </info.description>
[0490] </info.description.set>
[0491] </product.specifics>
[0492] <quantity>1
[0493] </quantity>
[0494] </product.instance>
[0495] <shipment.coordinates.set>
[0496] <shipment.coordinates>
[0497] <shipment.destination>
[0498] <address.set>
[0499] <address.named>SW-1
[0500] </address.named>
[0501] <address.physical>
[0502] <building.sublocation>208C</building.sublocation>
[0503] <location.in.street>123
[0504] </location.in.street>
[0505] <street>Frontage Rd.
[0506] </street>
[0507] <city>Beltway
[0508] </city>
[0509] <country.subentity.us
[0510] country.subentity.us.name ＝" MD" />
[0511] <postcode>20000
[0512] </postcode>
[0513] </address.physical>
[0514] <telephone>
[0515] <telephone.number>617-666-2000
[0516] </telephone.number>
[0517] <telephone.extension>1201
[0518] </telephone.extension>
[0519] </telephone>
[0520] </address.set>
[0521] </shipment.destination>
[0522] <shipment.special>No deliveries after 4PM</shipment.special>
[0523] </shipment.coordinates>
[0524] </shipment.coordinates.set>

[0525] <payment.set>

[0526] <credit.card

[0527] issuer.name =" VISA"

[0528] instrument.number =" 3787-812345-67893"

[0529] expiry.date =" 12/97"

[0530] currency.code =" USD" />

[0531] <amount.group>

[0532] <amount.monetary currency.code =" USD" >3975

[0533] </amount.monetary>

[0534] </amount.group>

[0535] </payment.set>

[0536] </line.item>

[0537] </exhange.description>

[0538] </transaction.description>

[0539] 因此，本发明提供了一项技术，利用该技术，市场参与者可以识别其本身，并且识别输入文档的类型和输出文档的类型，利用这些识别结果市场参与者将进行商业活动。由交易其余方或本地方处理这些文档中所载内容的特定方式不包括在建立商业关系和进行事务处理的过程中。

[0540] 图 3 提供了一简化图，示出了依照本发明的网络中的参与者网点。图 3 所示的网点包括一网络接口 300，它在端口 301 上与通信网相连。网络接口与文档语法分析器 301 相连。语法分析器 301 将来自输入文档的逻辑结构提供给翻译器模块 302，翻译器模块将输入文档翻译成主交易系统可以使用的形式，并且反过来将主过程的输出翻译成与商业接口定义中输出文档形式相匹配的文档格式，以便传送到目的地。语法分析器 301 和翻译器 302 对参与者模块 303 中存储的商业接口定义作出响应。

[0541] 将来自翻译器 302 的输出数据结构和语法分析器 301 发出的信号事件一起提供给事务处理过程前端 304。一个实施例中的前端 304 包括 JAVA 虚拟机或其它类似的适于在网络不同网点之间通信的接口。事务处理前端 304 对语法分析器 301 和翻译器 302 表示的事件作出响应，以便将输入数据传送给企业系统和网络中与参与者相连的适当功能。因此图 3 示例中的事务处理过程前端 304 与商务功能 305、数据库功能 306、诸如会计和记账等其它企业功能 307，并且与特殊的事件收听器和处理器 308 相连，其中装置 308 被设计成响应语法分析器所表示的事件。

[0542] 语法分析器 301 获得类似上例中的定单，或者根据商业接口定义规定的其它文档，并且产生一组事件，所述事件用本地事务处理结构（诸如 JAVA 虚拟机的一组 JAVA 事件）来识出。

[0543] 本发明的语法分析器不与根据接收文档收听事件的程序相连。满足某些规范说明的接收文档或完整文档的各标记段起收听功能指令的作用，以便开始处理。因此，收听程序完成与文档信息相关的商业逻辑。例如，与地址元素相关的程序可以是通过检查数据库而使邮政编码生效的代码。这些收听器通过用文档路由选择器进行登记来订购事件，路由选择器将相关的事件发送给对它们感兴趣的所有用户。

[0544] 例如，可以用收听语法分析器产生之事件的程序来监视上述规定的定单，该程序会将文档或其内容与定单输入程序相连。接收购货定单内的产品说明将调用一程序，以检查存货。然后，接收购货定单内的地址信息将调用一程序，以检查交货服务的可用性。文档中的买方信息字段可以调用一些过程，以检查信用度的购货定单历史，或者在已知消费者身份的基础上提供提升或类似的处理。

[0545] 复杂的收听器可以对原始收听器进行配置而构成。例如，定单收听器可以包含和调用前一段给出的清单收听器，或者可以对其自身调用清单成员。注意，收听器运行的应用程序不可能是本机 XML 过程或本机 JAVA 过程。在这些情况下，可以将对象转换成接收转换应用程序所要求的格式。当应用程序完成处理时，将其输出转换回 XML 格式，以便传输给网络中的其它网点。

[0546] 可以看出，上述市场参与文档型描述和事务文档型描述包括一个示意性的用于文档中逻辑单元的映像，并且包括基于自然语言的标记语言。自然语言标记和 XML 的其它自然语言属性便于将 XML 型标记语言用于对商业接口定义的说明，服务描述以及对输入和输出文档的描述。

[0547] 参与者模块 303 除了存储商业接口定义，还包括一编译程序，它用来编译事务处理过程前端 304 将要使用的对象或其它数据结构，其中事务处理过程前端 304 对应于输入文档中的逻辑结构，并且还对翻译器 302 进行编译。因此，由于当参与者涉及的事务改变时，参与者会修改或更新商业接口定义，所以翻译器 302 和语法分析器 301 将自动保持最新。

[0548] 在一较佳实施例中，用对应于 grove 型 SGML 的编译程序产生一组 JAVA 事件，主要为标准元件结构信息组增加每个元件的“性能组”。国际标准 ISO/IEC10179 :1996(E)，信息技术 -- 处理语言 -- 文档型语义和说明语言 (DSSSL)。将 XML 文档变成一组全球事件，以便对照正常的分析模型，在正常分析模型中，语法分析器的输出保持为内部的数据结构。通过将 XML 文档的元素翻译成 JAVA 事件或其它适于各网点事务处理前端使用的编程结构，可以在使用被交易文档的网点处获得丰富的功能。

[0549] 因此，事务处理过程前端 304 能够公开操作，并且预订能够增加收听器程序而不用知道或影响系统中其它收听程序的结构。图 3 中的每个收听器 305、306、307 和 308 保持一队列，前端 304 按该队列引导事件。这可以使多个收听器处理按其自身步调并行处理事件。

[0550] 另外，依照本发明，收听器运行的应用程序不需要本机 XML 功能，或者与输入文档格式相匹配的本机功能。如果事务处理过程前端 304 是 JAVA 接口，那么这些收听器可以是 JAVA 功能，或者是根据唯一事务处理结构运行的功能。在这些情况下，可以将对象转换成接收应用程序所要求的格式。当收听器的应用程序结束时，将其输出转换回如模块 303 中商业接口定义所规定的文档格式。因此，翻译器 302 与网络接口 300 相连，直接用于提供组织好的文档，作为输出。

[0551] 与事务处理前端相连的收听器可以包括用于输入文档收听器、用于输入文档中具体元素的收听器，以及用于存储在输入文档特定元素中的属性的收听器。这可以对参与者网点处的事务处理过程进行不同的且灵活的实现，以便过滤和响应输入文档。

[0552] 图 4 示出了对于图 3 的系统接收和处理输入文档的过程。因此，过程开始是在网

络接口接收一文档（步骤400）。语法分析器响应于商业接口定义识别文档类型(401)。利用该商业接口定义（它存储了XML格式文档的DTD),对文档进行语法分析（步骤402)。接下来,将文档的元素和属性翻译成主格式（步骤403)。在本例中,将XML逻辑结构翻译成JAVA对象,该对象载有XML元素的数据以及与诸如获取和设置功能等数据相关的方法。接下来,将主对象传递给主事务处理前端（步骤404)。响应于语法分析器和翻译器所表示的事件,将这些对象提供给某些过程。执行用于接收文档元素的过程,并且产生一输出（步骤405)。将输出翻译成如商业接口定义所定义的输出文档格式（步骤406)。在本例中,翻译从JAVA对象的形式进行到XML文档的形式。最后,通过网络接口将输出文档传送到目的地。

[0553] 图5更详细地描述了图3所示系统的事件生成器/事件收听器机构。一般来说,图5所示的方法是对JAVA JDK 1.1事件模型的精练。在该模型中,考虑三种对象。第一种对象是事件对象,它包含事件发生的信息。可以有许多种事件对象,它们对应于所有可能发生的不同事件。第二种对象是事件生成器,它监视活动,并且当发生某些事情时产生事件对象。第三种是事件收听器,它收听由事件生成器产生的事件对象。事件收听器一般收听特殊的事件生成器,诸如用鼠标器在特定的窗口上点击。事件收听器在事件生成器上调用“增加事件收听器”。该模型适用于图3的环境,在图3的环境中,响应分析和走过诸如XML文档所表示的对象曲线,来产生各对象。

[0554] 图5所示的系统包括一个普通XML语法分析器500。这种语法分析器可以用标准的回调模型来实现。当发生分析事件时,语法分析器调用应用对象中的特定方法,传送参数中的合适信息。因此,单个应用程序501与语法分析器在一起。如方框502所表示的,应用程序对XML事件对象中语法分析器所提供的信息进行封装,并且将其发送给许多已识别其本身的事件收听器。事件组502对语法分析器是完全独立的。可以将事件502提供给任何数目机器上的任何多收听器和任何多线程(thread)。在另一种情况下,事件基于元素结构信息组ESIS。因此,它们包括由文档结构的重要方面构成的清单,诸如元素的开始和结束,或者对属性的辨认。XML(和SGML)语法分析器一般将ESIS结构用作一组缺省信息,供语法分析器返回其应用程序。

[0555] 将专用ESIS收听器503与一组事件502相连。该收听器503实现ESIS收听器API,并且收听来自一个或多个生成器的所有XML事件。元素事件生成器504是一种专用ESIS收听器,它也是一个XML事件生成器。它的收听器是只对特殊类型元素的事件感兴趣的对象。例如在HTML环境中,收听器只对定单感兴趣,这只是<OL>和</OL>标签之间的文档部分。在另一例子中,收听器根据公用商业语言,从以上例举的文档中收听“party.name”元素,或者“service.name”元素;处理事件,以保证元素携带与元素的示意性映像相匹配的数据;并且根据接收网点所需的过程起作用。

[0556] 这允许系统具有较小的对象,它们收听文档的特定部分,诸如只相加价格部分。由于收听器可以从生成器中增加和删除其本身,所以可以有一个收听器只收听使用HTML文档<首标>部分。由于这一原因以及XML文档的高度递归特性,所以可以写出高度目标化的代码,并且写出迸发的收听器。例如,<OL>收听器可以建立<LI>收听器,其方式完全独立于<UL>(未定购的清单)收听器建立其<LI>收听器的方式。另一种方法是,建立一个产生图形用户接口的收听器,建立另一个用同一输入搜索数字库的收听器。因此,将文档视作由收听器执行的程序,与应用程序每次检查一段的精练的数据结构相反。如果用这种方法

写应用程序，那么不必将整个文档放入内存中，以执行应用程序。

[0557] 与事件组 502 相连的下一个收听器是属性过滤器 505。属性过滤器 505 象元素过滤器 504 一样是根据 ESIS 收听器模型的属性事件生成器。用于属性过滤器的收听器规定了它感兴趣的属性，并且接收具有该规定属性的任何元素的事件。因此例如，字体管理器只接收具有字体属性的元素事件，诸如 <PFONT ＝“Times Roman”/p>。

[0558] 元素事件生成器 504 提供这类元素对象，以便使元素收听器 504A 专门化。

[0559] 属性事件生成器 505 将属性事件对象提供给属性收听器 505A。同样，在用属性从一种文档类型转换成另一种的 SGML/XML 变换的意义上，将属性对象提供给一种“结构”。因此，505B 的结构允许一特定的属性具有一个可区分的特定名称。只有具有该规定属性的元素才会变成输出文档的一部分，并且输出文档中的元素名称是输入文档中的属性值。例如，如果结构 505B 是 HTML，那么串 ：

[0560] <PURCHASES HTML ＝″ OL″ ><ITEM HTML ＝″ LI″ ><NAME

[0561] HTML ＝″ B″ >STUFF</NAME><PRICE

[0562] HTML ＝″ B″ >123</PRICE></ITEM></PURCHASES>

[0563] 翻译成 ：

[0564] <OL><LI><B>STUFF</B><B>123</B></LI></OL> 这是正确的 HTML。

[0565] 与事件组 502 相连的下一个模块是树构成器。树构成器取一个 XML 事件流，并且产生代表基础文档的树。树构成器 506 的一种较佳版本根据 W3C 的规范说明（参见，http://www.w3.org/TR/1998/WD-DOM-19980720)，产生文档对象模型 DOM 对象 507。但是，可以用事件流中的收听器处理大多数要求，树的版本适用于支持围绕文档的问询、网点的重排序、创建新的文档，并且支持可用来多次产生同一事件流的内存中的数据结构结构，例如象多次分析文档。可以将专用构成器 508 与树构成器 506 相连，以便为文档的各个部分构成特殊的子树，适应一特定的实施情况。

[0566] 除了响应输入文档之外，还可以提供 XML 事件的其它源。因此，通过走过 DOM 对象树并且再生对文档分析时所建立的原始事件流，来产生事件流 510。这允许系统表现出文档被分析了若干次。

[0567] 使对象走过树并产生事件流的想法可以推广到 DOM 对象树以外，推广到可以质询的任何对象树。因此，JAVA 行走器 512 可以是走过 JAVA 豆构件树 513 的应用程序。行走器走过所有可公开访问的字段和方法。行走器跟踪它已经访问过的对象，保证不进入死循环。JAVA 事件 514 是 JAVA 行走器 512 产生的事件类型。这通常包括可以从对象中导出的大多数类型的信息。这是 ESIS 的 JAVA 等效体，并且允许将相同的编程方法应用于一般将用于 JAVA 对象的 XML，尽管特别对于 JAVA 豆。

[0568] JAVA 至 XML 事件生成器 515 构造了 JAVA 收听器和 JAVA 事件生成器。它接收来自 JAVA 行走器 512 的事件流 514，并且翻译所选中的，以便将 JAVA 对象表示成 XML 文档。在一较佳实施例中，事件生成器 515 开发 JAVA 豆 API。每个被观察的对象变成一个元素，元素名称与类别名称相同。在元素内，每个嵌入方法也变成这样一个元素，其内容是通过调用方法而返回的值。如果它是一个对象或者一个对象阵列，那么对它们轮流行走。

[0569] 图 6 描绘了在图 5 框架上建立的特定应用程序。该应用程序在 XML 文档 600 中取得，并且将其应用于语法分析器 / 生成器 601。产生 ESIS 事件 602 并将其提供给属性生成器

603 和树构成器 604。属性生成器对应于图 5 的生成器 505。它将事件提供给“结构”505B，以便将 XML 输入翻译成例如 HTML 输出。如方框 605 所表示的，并行处理这些事件，并且用收听器进行处理。将收听器的输出提供给文档书写器 506，然后，翻译回到 XML 格式，进行输出。因此例如，图 6 所示的应用程序取 XML 文档，并且输出包括一种形式的 HTML 文档。然后，将该形式发送给浏览器，并且将结果转换回成 XML。对于这一练习，结构概念提供了从 XML 到 HTML 的映像。图 6 中包括三种结构，一种提供了 HTML 文档的结构，诸如表格和列表；第二种规定了要显示的文本，诸如游览器文档上的输入字段标志；第三种描述了输入字段本身。在 HTML 形式下，XML 文档中要求保持 XML 文档结构的元素变成不可见字段。这适用于由这样的信息重构 XML 文档，其中客户将所述信息译成 HTTP 邮寄消息，发回服务器。每个体系结构取得输入文档，并且将其转换成基于 HTML 子集的体系结构。收听这些事件的收听器输出 HTML 文档的事件，然后这些事件行至文档书写器对象。文档书写器对象收听 XML 事件，并且将它们转换回到 XML 文档。在本例中，对于所有收听体系结构的元素生成器来说，文档书写器对象是一个收听器。

[0570] 图 5 和图 6 所示处理模块的构造方式代表了图 3 系统中语法分析器和事务处理过程前端的一个实施例。由图可见，提供了一种非常灵活的接口，通过这一接口，响应于输入 XML 文档或其它构造的文档格式，执行各种事务处理过程。

[0571] 图 7 示出了与图 3 类似的网点，不同之处在于它包括商业接口定义构成器模块 700。因此，图 7 的系统包括网络接口 701、文档语法分析器 702 和文档翻译器 703。翻译器 703 将其它输出提供给事务处理前端 704，而事务处理前端 704 与诸如商务功能 705、数据库 706、企业功能 707 和其它普通收听器和处理器 708 等收听功能相连。如图 7 所示，商业接口定义构成器 700 包括用户接口、公用商业库 CBL 资源库、用于阅读补充商业接口定义的过程，以及编译程序。用户接口用来帮助企业依赖公用商业库资源库建立商业接口定义，并且阅读补充商业接口定义的能力。因此，可以将补充商业接口定义的输入文档指定为特定事务的输出文档，并且可以将补充商业接口定义的输出文档指定为这一事务处理过程的输入。用类似的方法，可以用从 CBL 资源库选出的构件组成事务商业接口定义。使用 CBL 资源库鼓励使用标准化的文档格式，诸如上述例举的模式 (bidl) 文档、在构成商业接口定义时的逻辑结构和解释信息，其中商业接口定义很容易被网络中的其它人采用。

[0572] 商业接口定义构成器模块 700 还包括一编译程序，它用来生成翻译器 703，翻译器根据主事务处理体系结构所产生的对象，并且管理分析功能 702。

[0573] 图 8 是一启发式的图，示出了存储在商业接口定义构成器 700 的资源库中的逻辑结构。因此，资源库存储代表当事方商业接口定义 800，包括例如消费者 BID 801、目录库 BID 802、仓库 BID 803 和拍卖库 BID 804。因此，联机市场上的新的参与者可以将一种最符合其商业活动的标准化 BID 选作基本接口描述。另外，资源库将存储一组服务商业接口定义 805。例如，可以存储购货定单进入 BID 806、购货定单跟踪 BID 807、购货定单履行 BID 808 和目录服务 BID 809。当市场中的新的参与者构成一个商业接口定义时，它可以选择存储在资源库中的标准化服务的商业接口定义。

[0574] 除了当事方和服务 BID 之外，如字段 810 所示将输入和输出文档 BID 存储在资源库中。因此，可以将购货定单 BID 811、发票 BID 812、请求报价 BID813、产品可用量报告 BID 814，以及购货定单状态 BID 815 存储在资源库中。

[0575] 除了在较佳实施例中根据 XML 指定为文档类型定义的商业接口定义之外，如字段 816 所示，资源库存储语义映像形式的解释信息。因此，可以将本例中用于规定重量 817、货币 818、尺寸 819、产品标识符 820 和产品特性 821 的语义映像存储在资源库中。另外，解释信息为文档逻辑结构内的数据结构提供类型。

[0576] 另外，如方框 822 所示，可以将组成商业接口定义时使用的逻辑结构存储在资源库中。因此，方框 825 可以提供用于提供地址信号的形式 823，提供报价信号 824 的形式以及提供契约关系各项形式。当网络扩大时，CBL 资源库也将扩大和标准化，以便新参与者的增加以及对商业接口定义的修改更方便。

[0577] 图 9 示出了用图 7 系统构成商业接口定义的过程。过程开始于向用户显示 BID 构成器图形接口（步骤 900）。系统接受用户输入，识别由图形接口产生的参与者、服务和文档信息（步骤 901）。

[0578] 接下来，响应通过图形用户接口的用户输入，从资源库中检索任何被引用的逻辑结构、解释信息和文档定义和 / 或服务定义（步骤 902）。在下一步骤中，由网络中通过用户输入选中的其它参与者，通过定制的搜索引擎、web 浏览器或类似手段，访问任何补充的商业接口定义或商业接口定义的构件（步骤 903）。用收集到的信息建立参与者的文档定义（步骤 904）。用编译程序建立文档至宿主和宿主至文档的翻译器（步骤 905）。用编译程序建立与定义对应的主体系结构数据结构（步骤 906），并且在网络上邮寄已建立的商业接口定义，诸如张贴在环球网站点或其它地方，使其对于网络中的其它网点可访问（步骤 907）。

[0579] 商业接口定义告诉潜在的贸易伙伴公司所提供的联机服务，以及用哪些文档来调用这些服务。因此，用它们接受和产生的文档将服务定义在商业接口定义中。在以下 XML 服务定义的片段中说明这一点。

[0580] <service>

[0581] <service.name>Order Service</service.name>

[0582] <service.location>www.veosystems.com/order</service.location>

[0583] <service.op>

[0584] <service.op.name>Submit Order</service.op.name>

[0585] <service.op.inputdoc>www.commerce.net/po.dtd</service.op.inputdoc>

[0586] <service.op.outputdoc>

[0587] www.veosystems.com/invoice.dtd</service.op.outputdoc>

[0588] </service.op>

[0589] <service.op>

[0590] <service.op.name>Track Order</service.op.name>

[0591] <service.op.inputdoc>www.commerce.net

[0592] /request.track.dtd<service.op.inputdoc>

[0593] <service.op.outputdoc>

[0594] www.veosystems.com/response.track.dtd<service.op.outputdoc>

[0595] </service.op>

[0596] </service>

[0597] 该 XML 片段定义了由两件事务组成的服务，一种事务是取得购货定单，另一种事

务用于跟踪它们。如果将有效请求发送给规定的 Weg 地址，那么每个定义表示一种履行服务的联系或承诺。这里购货定单服务需要一种输入文档，该文档符合资源库中标准“po.dtd”文档类型定义，它可以是本地的，或者存储在网络上的行业范围登记中。如果一个网点可以履行该购货定单，那么它将返回一个符合定制“invoice.dtd”的文档，其定义是本地的。事实上，公司在承诺与发出购货定单的任何人做生意，其中购货定单符合它要求的 XML 规范说明。不需要任何在先的安排。

[0598] DTD 是具有给定类型的文档的形式规范说明或语法；它描述了元素、其属性和它们必须出现的次序。例如，购货定单一般包括买方和卖方的名称和地址、一组产品描述，以及诸如价格和发货日期等术语和条件。例如在电子数据交换 EDI 中，X12 850 是一种常用的购货定单模型。

[0599] 资源库鼓励用对于许多商业领域公用的可重新使用的语义构件来开发 XML 文档模型。这类文档可以从它们的公用消息元素来理解，即使它们可以表现得非常难。这是公用商业库资源库的作用。

[0600] 公用商业库资源由一般商业概念的信息模型组成，所述概念包括：

[0601] ●诸如公司、服务和产品等商业描述原语；

[0602] ●诸如目录、购货定单和发票等商业形式；

[0603] ●标准度量、日期和时间、位置、分类码。

[0604] 将此信息表示成一组可扩充的、公开的 XML 构成块，公司可以定制和组装这些构成块，快速开发 XML 应用程序。原子 CBL 元素实现了行业的消息发送标准和公约，诸如用于国家、货币、地址和时间的标准 ISO 代码。低级的 CBL 语义也来自对互联网资源之推荐的元数据框架（诸如，Dublin 核）的分析。

[0605] 下一等级的元素用这些构成块实现基本的商业形式，诸如那些在 X12 EDI 事务处理中使用的形式以及那些在形成互联网标准（诸如公开交易协议 OTP 和在互联网上公开购买 OBI）时使用的形式。

[0606] CBL 的重点是对于所有商业领域公用的功能和信息（诸如公司、服务和产品等商业描述原语；诸如目录、购货定单和发票等商业形式；标准度量、时期和时间、位置、分类码）。CBL 在可能的语义标准或行业公约上构成（例如，在分立的 CBL 模型中对下述规则编码，即规定欧洲的“日期 / 月份 / 年份”对美国的“月份 / 日期 / 年份”）。

[0607] CBL 是一种用于设计应用程序的语言。它被设计成连接 XML“文档世界”和 JAVA“编程世界”或其它事务处理体系结构之间的差距。模式体现了一种“用文档编程”的哲学思想，在该方法中，文档类型的详细形式规范说明是主源，由该主源可以生成各种相关的表格。这些表格包括用于 CBL 的 XML DTD、JAVA 对象、用于将 XML 的实例和对应的 JAVA 对象相互转换并支持文档的程序。

[0608] CBL 建立单个源，编译程序通过该源可以自动生成系统的几乎所有片段。CBL 通过扩充 SGML/XML 来工作，而 SGML/XML 通常用来正式定义特定文档类型的结构，以便包括与每个信息元素和属性相关的语义的规范说明。可以扩充由 SGML/XML 大部分字符类型组成的有限组，以定义任何种数据类型。

[0609] 这里是来自 CBL 定义的关于“日期时间”模块的一个片段：

[0610] <！ --datetime.mod Version:1.0-->

[0611] <！ --Copyright 1998 Veo Systems，Inc.-->

[0612] <！ ELEMENT year(#PCDATA)>

[0613] <！ ATTLIST year

[0614] schema CDATA#FIXED″ urn:x-veosystems:stds:iso:8601:3.8″

[0615] >

[0616] <！ ELEMENT month(#PCDATA)>

[0617] <！ ATTLIST month

[0618] schema CDATA#FIXED″ urn:x-veosystems:stds:iso:8601:3.12″

[0619] >

[0620] …

[0621] 在该片段中，将元素“年”定义为字符数据，并且相关的“模式”属性也定义为字符数据，它将“年”的模式定义为 ISO 8601 标准的 3.8 节。

[0622] 此“日期时间”CBL 模块事实上被定义为模式 DTD 的实例。首先，定义模块名称。然后，将“日期时间”元素“年”限制到 ISO 8601 的语义。

[0623] <！ DOCTYPE SCHEMA SYSTEM″ schema.dtd″ >

[0624] <SCHEMA><H1>Date and Time Module</H1>

[0625] <ELEMNTTYPE NAME ＝″ year″ DATATYPE ＝″ YEAR″ ><MODEL>

[0626] <STRING

[0627] DATATYPE ＝″ YEAR″ ></STRING></MODEL>

[0628] <ATTDEF NAME ＝ :schema:iso8601″ DATATYPE ＝″ CDATA″ >

[0629] <FIXED>3.8

[0630] Gregorian calendar</FIXED></ATTDEF></ELEMENTTYPE>

[0631] 上述例举的市场参与者和服务模块也存储在 CBL 资源库中。

[0632] 在图 10 中，通过定制普通的购货定单 DTD 1001，增加更具体的关于运送重量的信息 1002，来定义航空帐单 1000。普通购货定单 1001 最初全部用有关地址、日期和时间、货币，以及卖主和产品描述的 CBL 模块来组装。因此，使用 CBL 明显加速了 XML 商务应用程序的开发和实施。更重要的是，CBL 使商务应用程序更容易互连。

[0633] 在 CBL 中，用一种模式扩充 XML。扩充为 XML 元素增加了强类型，致使很容易使内容有效。例如，可以将元素 <CPU_clock_speed> 定义为一个具有一组有效值为 {100，133，166，200，233，266Mhz} 的整数。模式还增加了类-子类的层次，致使很容易由类定义例示信息。例如，可以将膝上型计算机描述成具有有关显示类型和电池寿命等特征附加标签的一个计算机。这些和其它扩充方便了数据的输入，以及在 XML 与传统的面向对象的和关系数据模型间的自动翻译。

[0634] 因此，通过编译程序、JAVA 豆和转换代码运行完整的 BID，编译程序如上所述为参与者和服务的实际实例产生 DTD，JAVA 豆对应于 DTD 实例中的逻辑结构，而转换代码用于从 XML 至 JAVA 以及从 JAVA 至 XML 的转换。在另一些系统中，还生成文档，显示在用户接口上，或者由用户打印出来，以便于对象的使用。

[0635] 对于上述例举的市场参与者和服务 DTD，由编译程序生成的 JAVA 豆叙述如下（为简明起见，使用一些新版本）：

```
[0636]      import com.veo.vsp.doclet.meta.Document;
[0637]  public class AddressPhysical extends Document{
[0638]      public static finalString DOC_TYPE =" address.physical" ;
[0639]      String mStreet;
[0640]      String mCity;
[0641]      public final static int AK = 0;
[0642]      public final static int AL = 1;
[0643]      public final static int AR = 2;
[0644]      public final static int AZ = 3;
[0645]      public final static int CA = 4;
[0646]      public final static int WI = 48;
[0647]      public final static int WV = 49;
[0648]      public final static int WY = 50;
[0649]      int mS tate;
[0650]      String mPostcode;
[0651]      public final static int AD = 51;
[0652]      public final static int AE = 52;
[0653]      public final static int AF = 53;
[0654]      public final static int AG = 54;
[0655]      public final static int AI = 55;
[0656]      public final static int AM = 56;
[0657]      int mCountry;
[0658]      public AddressPhysical(){
[0659]              super(DOC_TYPE);
[0660]      mStreet = new String();
[0661]      mCity = new String();
[0662]             this.mS tate = -1;
[0663]      mPostcode = null;
[0664]             this.mCountry = -1;
[0665]      }
[0666]      public AddressPhysical(String doc_type){
[0667]             super(doc_type);
[0668]      mStreet = new String();
[0669]      mCity = new String();
[0670]             this.mState = -1;
[0671]      mPostcode = null;
[0672]             this.mCountry = -1;
[0673]      }
[0674]      static public AddressPhysical initAddressPhysical(String iStreet,
```

```
String  iCity,int iState,String
[0675]  iPostcode,int iCountry){
[0676]                  AddressPhysical obj = new AddressPhysical() ;
[0677]                    obj.initializeAll(iStreet, iCity, iState, iPostcode,
iCountry) ;
[0678]                  return obj ;
[0679]      }
[0680]      public void initializeAll(String iStreet,String iCity,int iState,
String iPostcode,int iCountry){
[0681]              mStreet = iStreet ;
[0682]              mCity = iCity ;
[0683]              mState = iState ;
[0684]              mPostcode = iPostcode ;
[0685]              mCountry = iCountry ;
[0686]      }
[0687]      public String getStreet(){
[0688]      return mStreet ;
[0689]      }
[0690]      public String getStreetToXML(){
[0691]              if(getStreet() == null)return null ;
[0692]              char[]c = getStreet().toCharArray() ;
[0693]              StringBuffer sb = new StringBuffer() ;
[0694]              for(int x = 0 ;x<c.length ;x++){
[0695]              switch(c[x]){
[0696]              case' >' :
[0697]              sb.append(" &gt ; " ) ;
[0698]              break ;
[0699]              case' <' :
[0700]      sb.append(" &lt ; " ) ;
[0701]      break ;
[0702]      case' &' :
[0703]      sb.append(" &amp ; " ) ;
[0704]      break ;
[0705]      case" " :
[0706]      sb.append(" &quot ; " ) ;
[0707]      break ;
[0708]      case' \" :
[0709]      sb.append(" &quot ; " ) ;
[0710]      break ;
```

```
[0711]      default:
[0712]      if(Character.isDefined(c[x]))
[0713]      sb.append(c[x]);
[0714]      }
[0715]      }
[0716]      return sb.toString();
[0717]  }
[0718]  public void setStreet(String inp){
[0719]          this.mStreet = inp;
[0720]  }
[0721]  public void streetFromXML(String n){
[0722]          setStreet(n);
[0723]  }
[0724]  public String getCity(){
[0725]  return mCity;
[0726]  }
[0727]  public String getCityToXML(){
[0728]          if(getCity() == null)return null;
[0729]          char[]c = getCity().toCharArray();
[0730]          StringBuffer sb = new StringBuffer();
[0731]          for(int x = 0;x<c.length;x++){
[0732]          switch(c[x]){
[0733]          case' >' :
[0734]          sb.append(" &gt;" );
[0735]      break;
[0736]      case' <' :
[0737]      sb.append(" &lt;" );
[0738]      break;
[0739]      case' &' :
[0740]      sb.append(" &amp;" );
[0741]      break;
[0742]      case" " :
[0743]      sb.append(" &quot;" );
[0744]      break;
[0745]      case' \" :
[0746]      sb.append(" &quot;" );
[0747]      break;
[0748]      default:
[0749]      if(Character.isDefined(c[x]))
```

```
[0750]        sb.append(c[x]) ;
[0751]        }
[0752]        }
[0753]        return sb.toString() ;
[0754]    }
[0755]    public void setCity(String inp){
[0756]        this.mCity = inp ;
[0757]    }
[0758]    public void cityFromXML(String n){
[0759]        setCity(n) ;
[0760]    }
[0761]    public int getState(){
[0762]    return mState ;
[0763]    }
[0764]    public String getStateToXML(){
[0765]            switch(mState){
[0766]                    case AK:return" AK" ;
[0767]                    case AL:return" AL" ;
[0768]                    case AR:return" AR" ;
[0769]                    case AZ:return" AZ" ;
[0770]        }
[0771]    return null ;
[0772]    }
[0773]    public void setState(int inp){
[0774]        this.mState = inp ;
[0775]    }
[0776]    public void stateFromXML(String s){
[0777]        if(s.equals(" AK" ))mState = AK ;
[0778]        else if(s.equals(" AL" ))mState = AL ;
[0779]        else if(s.equals(" AR" ))mState = AR ;
[0780]        else if(s.equals(" AZ" ))mState = AZ ;
[0781]    }
[0782]    public String getPostcode(){
[0783]    return mPostcode ;
[0784]    }
[0785]    public String getPostcodeToXML(){
[0786]            if(getPostcode() == null)return null ;
[0787]            char[]c = getPostcode().toCharArray() ;
[0788]            StringBuffer sb = new StringBuffer() ;
```

```
[0789]          for(int x = 0 ;x<c.length ;x++){
[0790]          switch(c[x]){
[0791]          case' >' :
[0792]          sb.append(" &gt ; " ) ;
[0793]          break ;
[0794]          case' <' :
[0795]          sb.append(" &lt ; " ) ;
[0796]          break ;
[0797]          case' &' :
[0798]          sb.append(" &amp ; " ) ;
[0799]          break ;
[0800]          case" " :
[0801]      sb.append(" &quot ; " ) ;
[0802]      break ;
[0803]      case' \" :
[0804]      sb.append(" &quot ; " ) ;
[0805]      break ;
[0806]      default:
[0807]      if(Character.isDefined(c[x]))
[0808]      sb.append(c[x]) ;
[0809]      }
[0810]      }
[0811]      return sb.toString() ;
[0812]  }
[0813]  public void setPostcode(String inp){
[0814]          this.mPostcode = inp ;
[0815]  }
[0816]  public void postcodeFromXML(String n){
[0817]         setPostcode(n) ;
[0818]  }
[0819]  public int getCountry(){
[0820]  return mCountry ;
[0821]  }
[0822]  public String getCountryToXML(){
[0823]          switch(mCountry){
[0824]                case AD:return" AD" ;
[0825]                case AE:return" AE" ;
[0826]                case AF:return" AF" ;
[0827]           }
```

```
[0828]  return null;
[0829]  }
[0830]  public void setCountry(int inp){
[0831]          this.mCountry = inp;
[0832]  }
[0833]  public void countryFromXML(String s){
[0834]      if(s.equals(" AD" ))mCountry = AD;
[0835]      else if(s.equals(" AE" ))mCountry = AE;
[0836]      else if(s.equals(" AF" ))mCountry = AF;
[0837]      else if(s.equals(" AG" ))mCountry = AG;
[0838]      else if(s.equals(" AI" ))mCountry = AI;
[0839]      }
[0840]  }
[0841]  package com.veo.xdk.dev.schema.test.blib;
[0842]  import com.veo.vsp.doclet.meta.Document;
[0843]  public class AddressSet extends Document{
[0844]      public static final String DOC_TYPE =" address.set" ;
[0845]      AddressPhysical mAddressPhysical;
[0846]      String[]mTelephone;
[0847]      String[]mFax;
[0848]      String[]mEmail;
[0849]      String[]mInternet;
[0850]      public AddressSet(){
[0851]      super(DOC_TYPE);
[0852]      this.mAddressPhysical = new AddressPhysical();
[0853]      mTelephone = null;
[0854]      mFax = null;
[0855]      mEmail = null;
[0856]      mInternet = null;
[0857]      }
[0858]      public AddressSet(String doc_type){
[0859]      super(doc_type);
[0860]      this.mAddressPhysical = new AddressPhysical();
[0861]      mTelephone = null;
[0862]      mFax = null;
[0863]      mEmail = null;
[0864]      mInternet = null;
[0865]      }
[0866]        static public AddressSet initAddressSet(AddressPhysical
```

```
iAddressPhysical,String[]
[0867]   iTelephone,String[]iFax,String[]iEmail,String[]iInternet){
[0868]                    AddressSet obj = new AddressSet() ;
[0869]                     obj.initializeAll(iAddressPhysical, iTelephone, iFax,
iEmail,iInternet) ;
[0870]                    return obj ;
[0871]       }
[0872]        public void initializeAll(AddressPhysical iAddressPhysical,
String[]iTelephone,String[]
[0873]   iFax,String[]iEmail,String[]iInternet){
[0874]                    mAddressPhysical = iAddressPhysical ;
[0875]                    mTelephone = iTelephone ;
[0876]                    mFax = iFax ;
[0877]                    mEmail = iEmail ;
[0878]                   mInternet = iInternet ;
[0879]       }
[0880]       public AddressPhysical getAddressPhysical(){
[0881]       return mAddressPhysical ;
[0882]       }
[0883]       public void setAddressPhysical(AddressPhysical inp){
[0884]               this.mAddressPhysical = inp ;
[0885]       }
[0886]       public String[]getTelephone(){
[0887]       return mTelephone ;
[0888]       }
[0889]       public String getTelephone(int index){
[0890]               if(this.mTelephone == null)
[0891]               return null ;
[0892]               if(index> = this.mTelephone.length)
[0893]               return null ;
[0894]               if(index<0&&-index>this.mTelephone.length)
[0895]               return null ;
[0896]               if(index> = 0)return this.mTelephone[index] ;
[0897]               return this.mTelephone[this.mTelephone.length+index] ;
[0898]       }
[0899]   public String[]getTelephoneToXML(){
[0900]       String[]valArr = getTelephone() ;
[0901]       if(valArr == null)return null ;
[0902]       String[]nvArr = new String[valArr.length] ;
```

```
[0903]      for(int z = 0 ;z<nvArr.length ;z++){
[0904]      char[]c = valArr[z].toCharArray() ;
[0905]      StringBuffer st = new StringBuffer() ;
[0906]      StringBuffer sb = new StringBuffer() ;
[0907]      for(int x = 0 ;x<c.length ;x++){
[0908]      switch(c[x]){
[0909]      case' >' :
[0910]      sb.append(" &gt ; " ) ;
[0911]      break ;
[0912]      case' <' :
[0913]      sb.append(" &lt ; " ) ;
[0914]      break ;
[0915]      case' &' :
[0916]      sb.append(" &amp ; " ) ;
[0917]      break ;
[0918]      case" " :
[0919]      sb.append(" &quot ; " ) ;
[0920]      break ;
[0921]      case" ' :
[0922]      sb.append(" &quot ; " ) ;
[0923]      break ;
[0924]      default:
[0925]      if(Character.isDefined(c[x]))
[0926]      sb.append(c[x]) ;
[0927]      }
[0928]      }
[0929]      nvArr[z] = sb.toString() ;
[0930]      }
[0931]      return nvArr ;
[0932]  }
[0933]  public void setTelephone(int index,String inp){
[0934]      if(this.mTelephone == null){
[0935]              if(index<0){
[0936]              this.mTelephone = new String[1] ;
[0937]              this.mTelephone[0] = inp ;
[0938]              }else{
[0939]              this.mTelephone = new String[index+1] ;
[0940]              this.mTelephone[index] = inp ;
[0941]              }
```

```
[0942]      }else if(index<0){
[0943]                                  String[]newTelephone = new String[this.
mTelephone.length+1];
[0944]                          java.lang.System.arraycopy((Object)mTelephone,0,
[0945]   (Object)newTelephone,0,this.mTelephone.length);
[0946]                          newTelephone[newTelephone.length-1] = inp;
[0947]                          mTelephone = newTelephone;
[0948]                }else if(index>=this.mTelephone.length){
[0949]                          String[]newTelephone = new String[index+1];
[0950]                            java.lang.System.arraycopy((Object)mTelephone,
0,
[0951]   (Objec t)newTelephone,0,this.mTelephone.length);
[0952]                          newTelephone[index] = inp;
[0953]                          mTelephone = newTelephone;
[0954]                    }else{
[0955]                          this.mTelephone[index] = inp;
[0956]                    }
[0957]      }
[0958]      public void setTelephone(String[]inp){
[0959]               this.mTelephone = inp;
[0960]      }
[0961]      public void telephoneFromXML(String n){
[0962]               setTelephone(-1,n);
[0963]      }
[0964]      public String[]getFax(){
[0965]      return mFax;
[0966]      }
[0967]      public String getFax(int index){
[0968]                if(this.mFax == null)
[0969]      return null;
[0970]      if(index>=this.mFax.length)
[0971]      return null;
[0972]      if(index<0&&-index>this.mFax.length)
[0973]      return null;
[0974]      if(index>=0)return this.mFax[index];
[0975]      return this.mFax[this.mFax.length+index];
[0976]  }
[0977]  public String[]getFaxToXML(){
[0978]          Strung[]valArr = getFax();
```

```
[0979]          if(valArr == null)return null;
[0980]          String[]nvArr = new String[valArr.length];
[0981]          for(int z = 0;z<nvArr.length;z++){
[0982]          char[]c = valArr[z].toCharArray();
[0983]          StringBuffer st = new StringBuffer();
[0984]          StringBuffer sb = new StringBuffer();
[0985]          for(int x = 0;x<c.length;x++){
[0986]          switch(c[x]){
[0987]          case' >' :
[0988]          sb.append(" &gt;" );
[0989]          break;
[0990]          case' <' :
[0991]          sb.append(" &lt;" );
[0992]          break;
[0993]          case' &' :
[0994]          sb.append(" &amp;" );
[0995]          break;
[0996]          case" " :
[0997]          sb.append(" &quot;" );
[0998]          break;
[0999]          case" ' :
[1000]          sb.append(" &quot;" );
[1001]          break;
[1002]          default:
[1003]          if(Character.isDefined(c[x]))
[1004]      sb.append(c[x]);
[1005]      }
[1006]      }
[1007]      nvArr[z] = sb.toString();
[1008]      }
[1009]      return nvArr;
[1010]  }
[1011]  public void setFax(int index,String inp){
[1012]  if(this.mFax == null){
[1013]          if(index<0){
[1014]          this.mFax = new String[1];
[1015]          this.mFax[0] = inp;
[1016]          }else{
[1017]          this.mFax = new String[index+1];
```

```
[1018]          this.mFax[index] = inp;
[1019]          }
[1020]  }else if(index<0){
[1021]                  String[]newFax = new String[this.mFax.length+1];
[1022]                     java.lang.Sys tem.arraycopy((Object)mFax,0, (Object)
newFax,0,
[1023]  this.mFax.length);
[1024]                  newFax[newFax.length-1] = inp;
[1025]                  mFax = newFax;
[1026]      }else if(index> = this.mFax.length){
[1027]                  String[]newFax = new String[index+1];
[1028]                       java.lang.System.arraycopy((Object)mFax,0, (Object)
newFax,0,
[1029]  this.mFax.length);
[1030]                  newFax[index] = inp;
[1031]                  mFax = newFax;
[1032]          }else{
[1033]                this.mFax[index] = inp;
[1034]          }
[1035]      }
[1036]      public void setFax(String[]inp){
[1037]                this.mFax = inp;
[1038]      }
[1039]  public void faxFromXML(String n){
[1040]          setFax(-1,n);
[1041]  }
[1042]  public String[]getEmail(){
[1043]  return mEmail;
[1044]  }
[1045]  public String getEmail(int index){
[1046]          if(this.mEmail == null)
[1047]          return null;
[1048]          if(index> = this.mEmail.length)
[1049]          return null;
[1050]          if(index<0&&-index>this.mEmail.length)
[1051]          return null;
[1052]          if(index> = 0)return this.mEmail[index];
[1053]          returnthis.mEmail[this.mEmail.length+index];
[1054]  }
```

```
[1055]  public String[]getEmailToXML(){
[1056]          String[]valArr = getEmail() ;
[1057]          if(valArr == null)return null ;
[1058]          String[]nvArr = new String[valArr.length] ;
[1059]          for(int z = 0 ;z<nvArr.length ;z++){
[1060]          char[]c = valArr[z].toCharArray() ;
[1061]          StringBuffer st = new StringBuffer() ;
[1062]          StringBuffer sb = new StringBuffer() ;
[1063]          for(int x = 0 ;x<c.length ;x++){
[1064]          switch(c[x]){
[1065]          case' >' :
[1066]          sb.append(" &gt ; " ) ;
[1067]          break ;
[1068]          case' <' :
[1069]          sb.append(" &lt ; " ) ;
[1070]          break ;
[1071]          case' &' :
[1072]          sb.append(" &amp ; " ) ;
[1073]          break ;
[1074]          case" " :
[1075]          sb.append(" &quot ; " ) ;
[1076]          break ;
[1077]          case" ' :
[1078]          sb.append(" &quo t ; " ) ;
[1079]          break ;
[1080]          default:
[1081]          if(Characte risDefined(c[x]))
[1082]          sb.append(c[x]) ;
[1083]          }
[1084]          }
[1085]          nvArr[z] = sb.toString() ;
[1086]          }
[1087]          return nvArr ;
[1088]  }
[1089]  public void setEmail(int index,String inp){
[1090]  if(this.mEmail == null){
[1091]          if(index<0){
[1092]          this.mEmail = new String[1] ;
[1093]          this.mEmail[0] = inp ;
```

```
[1094]          }else{
[1095]          this.mEmail = new String[index+1];
[1096]          this.mEmail[index] = inp;
[1097]          }
[1098]  }else if(index<0){
[1099]                  String[]newEmail = new String[this.mEmail.length+1];
[1100]                     java.lang.System.arraycopy((Object)mEmail,0, (Object)
newEmail,0,
[1101]  this.mEmail.length);
[1102]                  newEmail[newEmail.length-1] = inp;
[1103]                  mEmail = newEmail;
[1104]        }else if(index>= this.mEmail.length){
[1105]                  String[]newEmail = new String[index+1];
[1106]                     java.lang.System.arraycopy((Object)mEmail,0, (Object)
newEmail,0,
[1107]  this.mEmail.length);
[1108]                  newEmail[index] = inp;
[1109]             mEmail = newEmail;
[1110]      }else{
[1111]            this.mEmail[index] = inp;
[1112]      }
[1113]  }
[1114]  public void setEmail(String[]inp){
[1115]          this.mEmail = inp;
[1116]  }
[1117]  public void emailFromxML(String n){
[1118]          setEmail(-1,n);
[1119]  }
[1120]  public String[]getInternet(){
[1121]  return mInternet;
[1122]  }
[1123]  public String getInternet(int index){
[1124]          if(this.mInternet == null)
[1125]          return null;
[1126]          if(index>= this.mInternet.length)
[1127]          return null;
[1128]          if(index<0&&-index>this.mInternet.length)
[1129]          return null;
[1130]          if(index>= 0)return this.mInternet[index];
```

```
[1131]          return this.mInternet[this.mInternet.length+index];
[1132]      }
[1133]      public String[]getInternetToXML(){
[1134]              String[]valArr = getInternet();
[1135]              if(valArr == null)return null;
[1136]              String[]nvArr = new String[valArr.length];
[1137]              for(int z = 0;z<nvArr.length;z++){
[1138]              char[]c = valArr[z].toCharArray();
[1139]              StringBuffer st = new StringBuffer();
[1140]              StringBuffer sb = new StringBuffer();
[1141]              for(int x = 0;x<c.length;x++){
[1142]              switch(c[x]){
[1143]              case' >' :
[1144]      sb.append(" &gt; " );
[1145]      break;
[1146]      case' <' :
[1147]      sb.append(" &lt; " );
[1148]      break;
[1149]      case' &' :
[1150]      sb.append(" &amp; " );
[1151]      break;
[1152]      case" " :
[1153]      sb.append(" &quot; " );
[1154]      break;
[1155]      case" ' " :
[1156]      sb.append(" &quot; " );
[1157]      break;
[1158]      default:
[1159]      if(Character.isDefined(c[x]))
[1160]      sb.append(c[x]);
[1161]      }
[1162]      }
[1163]      nvA rr[z] = sb.toString();
[1164]      }
[1165]      return nvArr;
[1166]  }
[1167]  public void setInternet(int index,String inp){
[1168]  if(this.mInternet == null){
[1169]          if(index<0){
```

```
[1170]          this.mInternet = new String[1];
[1171]          this.mInternet[0] = inp;
[1172]          }else{
[1173]          this.mInternet = new String[index+1];
[1174]          this.mInternet[index] = inp;
[1175]          }
[1176]  }else if(index<0){
[1177]                      String[|newInternet = new String|this.mInternet.
length+1];
[1178]                          java.lang.System.arraycopy((Object)mInternet,
0,
[1179]  (Object)newInternet,0,this.mInternet.length);
[1180]                          newInternet[newInternet.length-1] = inp;
[1181]                          mInternet = newInternet;
[1182]                  }else if(index> = this.mInternet.length){
[1183]                          String[]newInternet = new String[index+1];
[1184]                          java.lang.System.arraycopy((Object)mInternet,
0,
[1185]  (Object)newInternet,0,this.mInternet.length);
[1186]                          newInternet[index] = inp;
[1187]                          mInternet = newInternet;
[1188]                    }else{
[1189]                          this.mInternet[index] = inp;
[1190]                }
[1191]      }
[1192]      public void setInternet(String[]inp){
[1193]              this.mInternet = inp;
[1194]      }
[1195]      public void internetFromXML(String n){
[1196]              setInternet(-1,n);
[1197]      }
[1198]  }
[1199]  package com.veo.xdk.dev.schema.test.blib;
[1200]  import com.veo.vsp.doclet.meta.Document;
[1201]  public class Business extends Party{
[1202]      public static final Srring DOC_TYPE =" business" ;
[1203]      String aBusinessNumber;
[1204]      public Business(){
[1205]              super(DOC_TYPE);
```

```
[1206]        aBusinessNumber = new String() ;
[1207]        }
[1208]        public Business(String doc_tyPe){
[1209]                super(doc_tyPe) ;
[1210]        aBusinessNumber = new String() ;
[1211]        }
[1212]          static public Business initBusiness(String iBusinessNumber,
String[]iPartyName,AddressSet
[1213]   iAddressSet){
[1214]        Business obj = new Business() ;
[1215]        obj.initializeAll(iBusinessNumber,iPartyName,iAddressSet) ;
[1216]        return obj ;
[1217]        }
[1218]          public void initializeAll(String iBusinessNumber,String[]
iPartyName,AddressSet iAddressSet){
[1219]                aBusinessNumber = iBusinessNumber ;
[1220]                super.initializeAll(iPartyName,iAddressSet) ;
[1221]        }
[1222]        public String getBusinessNumber(){
[1223]        return aBusinessNumber ;
[1224]        }
[1225]        public String getBusinessNumberToXML(){
[1226]                if(getBusinessNumber() == null)return null ;
[1227]                char[]c = getBusinessNumber().toCharArray() ;
[1228]                StringBuffer sb = new StringBuffer() ;
[1229]                for(int x = 0 ;x<c.length ;x++){
[1230]                switch(c[x]){
[1231]                case' >' :
[1232]                sb.append(" &gt ; " ) ;
[1233]                break ;
[1234]                case' <' :
[1235]                sb.append(" &lt ; " ) ;
[1236]                break ;
[1237]                case' &' :
[1238]                sb.append(" &amp ; " ) ;
[1239]                break ;
[1240]                case" " :
[1241]                sb.append(" &quot ; " ) ;
[1242]                break ;
```

```
[1243]              case ' \" :
[1244]          sb.append(" &quot ; " ) ;
[1245]          break ;
[1246]          default:
[1247]          if(Character.isDefined(c[x]))
[1248]          sb.append(c[x]) ;
[1249]          }
[1250]          }
[1251]          return sb.toString() ;
[1252]      }
[1253]      public void setBusinessNumber(String inp){
[1254]              this.aBusinessNumber = inp ;
[1255]      }
[1256]      public void businessNumberFromXML(String n){
[1257]              setBusinessNumber(n) ;
[1258]      }
[1259]  }
[1260]      import com.veo.vsp.doclet.meta.Document ;
[1261]      public class Party extends Document{
[1262]          public static final String DOC_TYPE =" party" ;
[1263]          String[]mPartyName ;
[1264]          AddressSet mAddressSet ;
[1265]          public Party(){
[1266]                  super(DOC_TYPE) ;
[1267]          mPartyName = new String[0] ;
[1268]                this.mAddressSet = new AddressSet() ;
[1269]      }
[1270]      public Party(String doc type){
[1271]              super(doc_tyPe) ;
[1272]      mPartyName = new String[0] ;
[1273]              this.mAddressSet = new AddressSet() ;
[1274]      }
[1275]        static public Party initParty(String[]iPartyName,AddressSet
iAddressSet){
[1276]              Party obj = new Party() ;
[1277]      obj.initializeAll(iPartyName,iAddressSet) ;
[1278]      return obj ;
[1279]  }
[1280]  public void initializeAll(String[]iPartyName,AddressSet iAddressSet){
```

```
[1281]      mPartyName = iPartyName;
[1282]      mAddressSet = iAddressSet;
[1283]  }
[1284]  public String[]getPartyName(){
[1285]  return mP artyName;
[1286]  }
[1287]  public String getPartyName(int index){
[1288]          if(this.mPartyName == null)
[1289]          return null;
[1290]          if(index> = this.mPartyName.length)
[1291]          return null;
[1292]          if(index<0&&-index>this.mPartyName.length)
[1293]          return null;
[1294]          if(index> = 0)return this.mPartyName[index];
[1295]          return this.mPartyName[this.mPartyName.length+index];
[1296]  }
[1297]  public String[]getPartyNameToXML(){
[1298]          String[]valArr = getPartyName();
[1299]          if(valArr == null)return null;
[1300]          String[]nvArr = new String[valArr.length];
[1301]          for(int z = 0;z<nvArr.length;z++){
[1302]          char[]c = valArr[z].toCharArray();
[1303]          StringBuffer st = new StringBuffer();
[1304]          StringBuffer sb = new StringBuffer();
[1305]          for(int x = 0;x<c.length;x++){
[1306]          switch(c[x]){
[1307]          case' >' :
[1308]          sb.append(" &gt;" );
[1309]          break;
[1310]          case' <' :
[1311]      sb.append(" &lt;" );
[1312]      break;
[1313]      case' &' :
[1314]      sb.append(" &amp;" );
[1315]      break;
[1316]      case" " :
[1317]      sb.append(" &quot;" );
[1318]      break;
[1319]      case" ' :
```

```
[1320]        sb.append(" &quot; " ) ;
[1321]        break ;
[1322]        default:
[1323]        if(Character.isDefined(c[x]))
[1324]        sb.append(c[x]) ;
[1325]        }
[1326]        }
[1327]        nvArr[z] = sb.toString() ;
[1328]        }
[1329]        return nvArr ;
[1330]    }
[1331]    public void setPartyName(int index,String inp){
[1332]    if(this.mPartyName == null){
[1333]            if(index<0){
[1334]            this.mPartyName = new String[1] ;
[1335]            this.mPartyName[0] = inp ;
[1336]            }else{
[1337]            this.mPartyName = new String[index+1] ;
[1338]            this.mPartyName[index] = inp ;
[1339]            }
[1340]        }else if(index<0){
[1341]                        String[]newPartyName = new String[this.mPartyName.
length+1] ;
[1342]                        java.lang.System.arraycopy((Object)mPartyName,0,
[1343]    (Object)newPartyName,0,this.mPartyName.length) ;
[1344]                         newPartyName[newPartyName.length-1] = inp ;
[1345]                         mPartyName = newPartyName ;
[1346]                         }else if(index> = this.mPartyName.length){
[1347]                         String[]newPartyName = new String[index+1] ;
[1348]                         java.lang.System.arraycopy((Object)mPartyName,0,
[1349]    (Object)newPartyName,0,this.mPartyName.length) ;
[1350]                          newPartyName[index] = inp ;
[1351]                          mPartyName = newPartyName ;
[1352]                    }else{
[1353]                          this.mPartyName[index] = inp ;
[1354]                    }
[1355]        }
[1356]        public void setPartyName(String[]inp){
[1357]                this.mPartyName = inp ;
```

```
[1358]      }
[1359]      public void partyNameFromXML(String n){
[1360]              setPartyName(-1,n) ;
[1361]      }
[1362]      public AddressSet getAddressSet(){
[1363]      return mAddressSet ;
[1364]      }
[1365]      public void setAddressSet(AddressSet inp){
[1366]              this.mAddressSet = inp ;
[1367]      }
[1368]  }
[1369]  package com.veo.xdk.dev.schema.test.blib ;
[1370]  import com.veo.vsp.doclet.meta.Document ;
[1371]  public class Person extends Party{
[1372]      public static final String DOC_TYPE =" person" ' ;
[1373]      String aSSN ;
[1374]      public Person(){
[1375]              super(DOC_TYPE) ;
[1376]      aSSN = null ;
[1377]      }
[1378]      public Person(String doc_type){
[1379]      super(doc_type) ;
[1380]  aSSN = null ;
[1381]  }
[1382]  static public Person initPerson(String iSSN,String[]iPartyName,
AddressSet iAddressSet){
[1383]          Person obj = new Person() ;
[1384]          obj.initializeAll(iSSN,iPartyName,iAddressSet) ;
[1385]          return obj ;
[1386]  }
[1387]  public void initializeAll(String iSSN,String[]iPartyName,AddressSet
iAddressSet){
[1388]          aSSN = iSSN ;
[1389]          super.initializeAll(iPartyName,iAddressSet) ;
[1390]  }
[1391]  public String getSSN(){
[1392]  return aSSN ;
[1393]  }
[1394]  public String getSSNToXML(){
```

```
[1395]          if(getSSN() == null)return null ;
[1396]          char[]c = getSSN().toCharArray() ;
[1397]          StringBuffer sb = new StringBuffer() ;
[1398]          for(int x = 0 ;x<c.length ;x++){
[1399]          switch(c[x]){
[1400]          case' >' :
[1401]          sb.append(" &gt ; " ) ;
[1402]          break ;
[1403]          case' <' :
[1404]          sb.append(" &lt ; " ) ;
[1405]          break ;
[1406]          case' &' :
[1407]          sb.append(" &amp ; " ) ;
[1408]          break ;
[1409]          case" " :
[1410]          sb.append(" &quot ; " ) ;
[1411]          break ;
[1412]      case' \" :
[1413]      sb.append(" &quot ; " ) ;
[1414]      break ;
[1415]      default:
[1416]      if(Character.isDefined(c[x]))
[1417]      sb.append(c[x]) ;
[1418]      }
[1419]      }
[1420]      return sb.toString() ;
[1421]  }
[1422]  public void setSSN(String inp){
[1423]          this.aSSN = inp ;
[1424]  }
[1425]  public void sSNFromXML(String n){
[1426]      setSSN(n) ;
[1427]  }
[1428]  }
[1429]  package com.veo.xdk.dev.schema.test.blib ;
[1430]  import com.veo.vsp.doclet.meta.Document ;
[1431]  public class PrototyPeService extends Document{
[1432]      public static final String DOC_TYPE =" prototyPe.service" ;
[1433]      String mServiceName ;
```

```
[1434]       String[]mServiceTerms;
[1435]       String[]mServiceLocation;
[1436]       ServiceOperation[]mServiceOperation;
[1437]       public PrototypeService(){
[1438]              super(DOC_TYPE);
[1439]       mServiceName = new String();
[1440]       mServiceTerms = new String[0];
[1441]       mServiceLocation = new String[0];
[1442]              this.mServiceOperation = new ServiceOperation[0];
[1443]       }
[1444]       public PrototypeService(String doc_type){
[1445]              super(doc_type);
[1446]       mServiceName = new String();
[1447]       mServiceTerms = new String[0];
[1448]       mServiceLocation = new String[0];
[1449]              this.mServiceOperation = new ServiceOperation[0];
[1450]       }
[1451]         static public PrototypeService initProto typeService(String
iServiceName,String[]
[1452]   iServiceTerms,String[]iServiceLocation,ServiceOpera tion[]
iServiceOperation){
[1453]                    PrototypeService obj = new PrototypeService();
[1454]                         obj.initializeAll(iServiceName,iServiceTerms,
iServiceLocation,iServiceOperation);
[1455]                    return obj;
[1456]       }
[1457]         public void initializeAll(String iServiceName,String[]
iServiceTerms,String[]
[1458]   ServiceLocation,ServiceOperation[]iServiceOperation){
[1459]                     mServiceName = iServiceName;
[1460]                     mServiceTerms = iServiceTerms;
[1461]                     mServiceLocation = iServiceLocation;
[1462]                     mServiceOperation = iServiceOperation;
[1463]       }
[1464]       public String getServiceName(){
[1465]       return mServiceName;
[1466]       }
[1467]       public String getServiceNameToXML(){
[1468]              if(getServiceName() == null)return null;
```

```
[1469]              char[]c = getServiceName().toCharArray() ;
[1470]              StringBuffer sb = new StringBuffer() ;
[1471]              for(int x = 0 ;x<c.length ;x++){
[1472]              switch(c[x]){
[1473]              case' >' :
[1474]              sb.append(" &gt ; " ) ;
[1475]              break ;
[1476]              case' <' :
[1477]              sb.append(" &lt ; " ) ;
[1478]      break ;
[1479]      case' &' :
[1480]      sb.append(" &amp ; " ) ;
[1481]      break ;
[1482]      case" " :
[1483]      sb.append(" &quot ; " ) ;
[1484]      break ;
[1485]      case' \" :
[1486]      sb.append(" &quot ; " ) ;
[1487]      break ;
[1488]      default:
[1489]      if(Character.isDefined(c[x]))
[1490]      sb.append(c[x]) ;
[1491]      }
[1492]      }
[1493]      return sb.toString() ;
[1494]  }
[1495]  public void setServiceName(String inp){
[1496]          this.mServiceName = inp ;
[1497]  }
[1498]  public void serviceNameFromXML(String n){
[1499]          setServiceName(n) ;
[1500]  }
[1501]  public String[]getServiceTerms(){
[1502]  return mServiceTerms ;
[1503]  }
[1504]  public String getServiceTerms(int index){
[1505]          if(this.mServiceTerms == null)
[1506]          return null ;
[1507]          if(index> = this.mServiceTerms.length)
```

```
[1508]          return null;
[1509]          if(index<0&&-index>this.mServiceTerms.length)
[1510]          return null;
[1511]          if(index>=0)return this.mServiceTerms[index];
[1512]          return this.mServiceTerms[this.mServiceTerms.length+index];
[1513]  }
[1514]  public String[]getServiceTermsToXML(){
[1515]      String[]valArr=getServiceTerms();
[1516]      if(valArr==null)return null;
[1517]      String[]nvArr=new String[valArr.length];
[1518]      for(int z=0;z<nvArr.length;z++){
[1519]      char[]c=valArr[z].toCharArray();
[1520]      StringBuffer st=new StringBuffer();
[1521]      StringBuffer sb=new StringBuffer();
[1522]      for(int x=0;x<c.length;x++){
[1523]      switch(c[x]){
[1524]      case'>':
[1525]      sb.append("&gt;");
[1526]      break;
[1527]      case'<':
[1528]      sb.append("&lt;");
[1529]      break;
[1530]      case'&':
[1531]      sb.append("&amp;");
[1532]      break;
[1533]      case"":
[1534]      sb.append("&quot;");
[1535]      break;
[1536]      case"'":
[1537]      sb.append("&quot;");
[1538]      break;
[1539]      default:
[1540]      if(Character.isDefined(c[x]))
[1541]      sb.append(c[x]);
[1542]      }
[1543]      }
[1544]      nvArr[z]=sb.toString();
[1545]      }
[1546]      return nvArr;
```

```
[1547]   }
[1548]       public void setServiceTerms(int index,String inp){
[1549]       if(this.mServiceTerms == null){
[1550]               if(index<0){
[1551]               this.mServiceTerms = new String[1];
[1552]               this.mServiceTerms[0] = inp;
[1553]               }else{
[1554]               this.mServiceTerms = new String[index+1];
[1555]               this.mServiceTerms[index] = inp;
[1556]               }
[1557]       }else if(index<0){
[1558]                               String[]newServiceTerms = new String[this.
mServiceTerms.length+1];
[1559]                        java.lang.System.arraycopy((Object)mServiceTerms,
0,
[1560]   (Object)newServiceTerms,0,this.mServiceTerms.length);
[1561]                       newServiceTerms[newServiceTerms.length-1] = inp;
[1562]                       mServiceTerms = newServiceTerms;
[1563]             }else if(index>=this.mServiceTerms.length){
[1564]                       String[]newServiceTerms = new String[index+1];
[1565]                        java.lang.System.arraycopy((Object)mServiceTerms,
0,
[1566]   (Object)newServiceTerms,0,this.mServiceTerms.length);
[1567]                       newServiceTerms[index] = inp;
[1568]                       mServiceTerms = newServiceTerms;
[1569]                }else{
[1570]                       this.mServiceTerms[index] = inp;
[1571]                }
[1572]       }
[1573]       public void setServiceTerms(String[]inp){
[1574]               this.mServiceTerms = inp;
[1575]       }
[1576]       public void serviceTermsFromXML(String n){
[1577]                setServiceTerms(-1,n);
[1578]       }
[1579]       public String[]getServiceLocation(){
[1580]       return mServiceLocation;
[1581]       }
[1582]   public String getServiceLocation(int index){
```

```
[1583]      if(this.mServiceLocation == null)
[1584]      return null;
[1585]      if(index> = this.mServiceLocation.length)
[1586]      return null;
[1587]      if(index<0&&-index>this.mServiceLocation.length)
[1588]      return null;
[1589]      if(index> = 0)return this.mServiceLocation[index];
[1590]      return this.mServiceLocation[this.mServiceLocation.length+index];
[1591]  }
[1592]  public String[]getServiceLocationToXML(){
[1593]      String[]valArr = getServiceLocation();
[1594]      if(valArr == null)return null;
[1595]      String[]nvArr = new String[valArr.length];
[1596]      for(int z = 0;z<nvArr.length;z++){
[1597]      char[]c = valArr[z].toCharArray();
[1598]      StringBuffer st = new StringBuffer();
[1599]      StringBuffer sb = new StringBuffer();
[1600]      for(int x = 0;x<c.length;x++){
[1601]      switch(c[x]){
[1602]      case' >' :
[1603]      sb.append(" &gt; " );
[1604]      break;
[1605]      case' <' :
[1606]      sb.append(" &lt; " );
[1607]      break;
[1608]      case' &' :
[1609]      sb.append(" &amp; " );
[1610]      break;
[1611]      case" " :
[1612]      sb.append(" &quot; " );
[1613]      break;
[1614]      case" ' :
[1615]      sb.append(" &quot; " );
[1616]      break;
[1617]              default:
[1618]              if(Character.isDefined(c[x]))
[1619]              sb.append(c[x]);
[1620]              }
[1621]              }
```

```
[1622]              nvArr[z] = sb.toString() ;
[1623]              }
[1624]              return nvArr ;
[1625]          }
[1626]          public void setServiceLocation(int index,String inp){
[1627]          if(this.mServiceLocation == null){
[1628]                  if(index<0){
[1629]                  this.mServiceLocation = new String[1] ;
[1630]                  this.mServiceLocation[0] = inp ;
[1631]                  }else{
[1632]                  this.mServiceLocation = new String[index+1] ;
[1633]                  this.mServiceLocation[index] = inp ;
[1634]                  }
[1635]          }else if(index<0){
[1636]                          String[]newServiceLocation = new
[1637]  String[this.mServiceLocation.length+1] ;
[1638]                                  java.lang.System.arraycopy((Object)
mServiceLocation,0,
[1639]  (Object)newServiceLocation,0,this.mServiceLocation.length) ;
[1640]                                  newServiceLocation[newServiceLocation.
length-1] = inp ;
[1641]                          mServiceLocation = newServiceLocation ;
[1642]                  }else if(index>= this.mServiceLocation.length){
[1643]                                   String[]newServiceLocation = new
String[index+1] ;
[1644]                                  java.lang.System.arraycopy((Object)
mServiceLocation,0,
[1645]  (Object)newServiceLocation,0,this.mServiceLocation.length) ;
[1646]                          newServiceLocation[index] = inp ;
[1647]                          mServiceLocation = newServiceLocation ;
[1648]                  }else{
[1649]                          this.mServiceLocation[index] = inp ;
[1650]                  }
[1651]      }
[1652]          public void setServiceLocation(String[]inp){
[1653]              this.mServiceLocation = inp ;
[1654]      }
[1655]      public void serviceLocationFromXML(String n){
[1656]              setServiceLocation(-1,n) ;
```

```
[1657]        }
[1658]        public ServiceOperation[]getServiceOperation(){
[1659]        return mServiceOperation;
[1660]        }
[1661]        public ServiceOperation getServiceOperation(int index){
[1662]                if(this.mServiceOperation == null)
[1663]                return null;
[1664]                if(index> = this.mServiceOperation.length)
[1665]                return null;
[1666]                if(index<0&&-index>this.mServiceOperation.length)
[1667]                return null;
[1668]                if(index> = 0)return this.mServiceOperation[index];
[1669]                    return  this.mServiceOperation[this.mServiceOperation.
length+index];
[1670]        }
[1671]        public void setServiceOperation(int index,ServiceOperation inp){
[1672]        if(this.mServiceOperation == null){
[1673]                if(index<0){
[1674]                this.mServiceOperation = new ServiceOperation[1];
[1675]                this.mServiceOperation[0] = inp;
[1676]                }else{
[1677]                this.mServiceOperation = new ServiceOperation[index+1];
[1678]                this.mServiceOperation[index] = inp;
[1679]                }
[1680]        }else if(index<0){
[1681]                            ServiceOperation[]newServiceOperation = new
[1682]   ServiceOperation[this.mServiceOperation.length+1];
[1683]                                    java.lang.System.arraycopy((Object)
mServiceOperation,0,
[1684]   (Object)newServiceOperation,0,this.mServiceOperation.length);
[1685]                                  newServiceOperation[newServiceOperation.
length-1] = inp;
[1686]                            mServiceOperation = newServiceOperation;
[1687]           }else if(index> = this.mServiceOperation.length){
[1688]                          ServiceOPeration[]newServiceOPeration = new
ServiceOperation[index+1];
[1689]                java.lang.System.arraycopy((Object)mServiceOperation,0,
[1690]        (Object)newServiceOperation,0,this.mServiceOperation.length);
[1691]                newServiceOperation[index] = inp;
```

```
[1692]                mServiceOperation = newServiceOperation;
[1693]           }else{
[1694]                this.mServiceOperation[index] = inp;
[1695]           }
[1696]       }
[1697]       public void setServiceOperation(ServiceOperation[]inp){
[1698]                this.mServiceOperation = inp;
[1699]       }
[1700]   }
[1701]   package com.veo.xdk.dev.schema.test.blib;
[1702]   import com.veo.vsp.doclet.meta.Document;
[1703]   public class Service extends Document{
[1704]       public static final String DOC_TYPE =" service " ;
[1705]       String mServiceName;
[1706]       String mServiceLocation;
[1707]       ServiceOperation[]mServiceOperation;
[1708]       String mServiceTerms;
[1709]       public Service(){
[1710]                super(DOC_TYPE);
[1711]       mServiceName = new String();
[1712]       mServiceLocation = new String();
[1713]                this.mServiceOperation = new ServiceOpera tion[0];
[1714]       mServiceTerms = new String();
[1715]       }
[1716]       public Service(String doc_type){
[1717]                super(doe_type);
[1718]       mServiceName = new String();
[1719]       mServiceLocation = new String();
[1720]                this.mServiceOperation = new ServiceOperation[0];
[1721]       mServiceTerms = new String();
[1722]       }
[1723]       static public Service initService(String iServiceName,String
[1724]   iServiceLocation,ServiceOperation[]iServiceOperation,String
iServiceTerms){
[1725]                 Service obj = new Service();
[1726]                      obj.initializeAll(iServiceName,iServiceLocation,
iServiceOperation,
[1727]   iServiceTerms);
[1728]                 return obj;
```

```
[1729]          }
[1730]          public void initializeAll(String iServiceName,String
[1731]  iServiceLocation,ServiceOperation[]iServiceOperation,String
iServiceTerms){
[1732]                  mServiceName = iServiceName;
[1733]                  mServiceLocation = iServiceLocation;
[1734]                  mServiceOperation = iServiceOperation;
[1735]                  mServiceTerms = iServiceTerms;
[1736]      }
[1737]      public String getServiceName(){
[1738]      return mServiceName;
[1739]      }
[1740]      public String getServiceNameToXML(){
[1741]              if(getServiceName() == null)return null;
[1742]              char[]c = getServiceName().toCharArray();
[1743]              StringBuffer sb = new StringBuffer();
[1744]              for(int x = 0;x<c.length;x++){
[1745]              switch(c[x]){
[1746]              case' >' :
[1747]              sb.append(" &gt; " );
[1748]              break;
[1749]              case' <' :
[1750]              sb.append(" &lt; " );
[1751]              break;
[1752]              case' &' :
[1753]      sb.append(" &amp; " );
[1754]      break;
[1755]      case" " :
[1756]      sb.append(" &quot; " );
[1757]      break;
[1758]      case' \" :
[1759]      sb.append(" &quot; " );
[1760]      break;
[1761]      default:
[1762]      if(Character.isDefined(c[x]))
[1763]      sb.append(c[x]);
[1764]      }
[1765]      }
[1766]      return sb.toString();
```

```
[1767]  }
[1768]  public void setServiceName(String inp){
[1769]      this.mServiceName = inp;
[1770]  }
[1771]  public void serviceNameFromXML(String n){
[1772]          setServiceName(n);
[1773]  }
[1774]  public String getServiceLocation(){
[1775]  return mServiceLocation;
[1776]  }
[1777]  public String getServiceLocationToXML(){
[1778]          if(getServiceLocation() == null)return null;
[1779]          char[]c = getServiceLocation().toCharArray();
[1780]          StringBuffer sb = new StringBuffer();
[1781]          for(int x = 0;x<c.length;x++){
[1782]          switch(c[x]){
[1783]          case' >' :
[1784]          sb.append(" &gt;" );
[1785]          break;
[1786]          case' <' :
[1787]          sb.append(" &lt;" );
[1788]      break;
[1789]      case' &' :
[1790]      sb.append(" &amp;" );
[1791]      break;
[1792]      case" " :
[1793]      sb.append(" &quot;" );
[1794]      break;
[1795]      case' \" :
[1796]      sb.append(" &quot;" );
[1797]      break;
[1798]      default:
[1799]      if(Character.isDefined(c[x]))
[1800]      sb.append(c[x]);
[1801]      }
[1802]      }
[1803]      return sb.toString();
[1804]  }
[1805]  public void setServiceLocation(String inp){
```

```
[1806]          this.mServiceLocation = inp;
[1807]  }
[1808]  public void serviceLocationFromXML(String n){
[1809]          setServiceLocation(n);
[1810]  }
[1811]  public ServiceOperation[]getServiceOperation(){
[1812]  return mServiceOperation;
[1813]  }
[1814]  public ServiceOperation getServiceOperation(int index){
[1815]          if(this.mServiceOperation == null)
[1816]          return null;
[1817]          if(index> = this.mServiceOperation.length)
[1818]          return null;
[1819]          if(index<0&&-index>this.mServiceOperation.length)
[1820]          return null;
[1821]          if(index> = 0)return this.mServiceOperation[index];
[1822]             return  this.mServiceOperation[this.mServiceOperation.
length+index];
[1823]      }
[1824]      public void setServiceOperation(int index,ServiceOperation inp){
[1825]      if(this.mServiceOperation == null){
[1826]              if(index<0){
[1827]              this.mServiceOperation = new ServiceOperation[1];
[1828]              this.mServiceOperation[0] = inp;
[1829]              }else{
[1830]              this.mServiceOperation = new ServiceOperation[index+1];
[1831]              this.mServiceOperation[index] = inp;
[1832]              }
[1833]      }else if(index<0){
[1834]                  ServiceOperation[]newServiceOperation = new
[1835]  ServiceOperation[this.mServiceOperation.length+1];
[1836]                                  java.lang.System.arraycopy((Object)
mServiceOperation,0,
[1837]  (Object)newServiceOperation,0,this.mServiceOperation.length);
[1838]                                  newServiceOperation[newServiceOperation.
length-1] = inp;
[1839]                          mServiceOperation = newServiceOperation;
[1840]                  }else if(index> = this.mServiceOperation.length){
[1841]                          ServiceOperation[]newServiceOperation = new
```

```
ServiceOperation[index+1];
[1842]                                    java.lang.System.arraycopy((Object)
mServiceOperation,0,
[1843]   (Object)newServiceOperation,0,this.mServiceOperation.length);
[1844]                          newServiceOperation[index] = inp;
[1845]                          mServiceOperation = newServiceOperation;
[1846]                  }else{
[1847]                       this.mServiceOperation[index] = inp;
[1848]                  }
[1849]       }
[1850]       public void setServiceOperation(ServiceOperation[]inp){
[1851]               this.mServiceOperation = inp;
[1852]       }
[1853]       public String getServiceTerms(){
[1854]       return mServiceTerms;
[1855]       }
[1856]       public String getServiceTermsToXML(){
[1857]           if(getServiceTerms() == null)return null;
[1858]           char[]c = getServiceTerms().toCharArray();
[1859]           StringBuffer sb = new StringBuffer();
[1860]           for(int x = 0;x<c.length;x++){
[1861]           switch(c[x]){
[1862]           case' >' :
[1863]           sb.append(" &gt;" );
[1864]           break;
[1865]           case' <' :
[1866]           sb.append(" &lt;" );
[1867]           break;
[1868]           case' &' :
[1869]           sb.append(" &amp;" );
[1870]           break;
[1871]           case" " :
[1872]           sb.append(" &quot;" );
[1873]           break;
[1874]           case' \" :
[1875]           sb.append(" &quot;" );
[1876]           break;
[1877]           default:
[1878]           if(Character.isDefined(c[x]))
```

```
[1879]            sb.append(c[x]) ;
[1880]            }
[1881]            }
[1882]            return sb.toString() ;
[1883]        }
[1884]        public void setServiceTerms(String inp){
[1885]                this.mServiceTerms = inp ;
[1886]        }
[1887]        public void serviceTermsFromXML(String n){
[1888]                setServiceTerms(n) ;
[1889]        }
[1890]    }
[1891]        package com.veo.xdk.dev.schema.test.blib ;
[1892]        import com.veo.vsp.doclet.meta.Document ;
[1893]        public class ServiceOperation extends Document{
[1894]           public static final String DOC_TYPE =" service.operation" ;
[1895]           Srring mServiceOperationName ;
[1896]           String mServiceOperationLocation ;
[1897]           Strnig mServiceOperationInput ;
[1898]           String mServiceOperationOutput ;
[1899]           public ServiceOperation(){
[1900]                   super(DOC_TYPE) ;
[1901]           mServiceOperationName = new String() ;
[1902]           mServiceOpera tionLocation = new String() ;
[1903]           mServiceOperationInput = new String() ;
[1904]           mServiceOperationOutput = new String() ;
[1905]           }
[1906]           public ServiceOperation(String doc_type){
[1907]                   super(doc_tyPe) ;
[1908]           mServiceOperationName = new String() ;
[1909]           mServiceOperationLocation = new String() ;
[1910]           mServiceOperationInput = new String() ;
[1911]           mServiceOperationOutput = new String() ;
[1912]           }
[1913]             static public ServiceOperation initServiceOperation(String
iServiceOperationName,String
[1914]    iServiceOperationLocation,String iServiceOperationInput,String
[1915]    iServiceOperationOutput){
[1916]                    ServiceOperation obj = new ServiceOperation() ;
```

```
[1917]                          obj.initializeAll(iServiceOperationName,
iServiceOperationLocation,
[1918]   iServiceOperationInput,iServiceOperationOutput) ;
[1919]                  return obj ;
[1920]       }
[1921]       public void initializeAll(String iServiceOperationName,String
[1922]   iServiceOperationLocation,String iServiceOperationInput,String
[1923]   iServiceOperationOutput){
[1924]                  mServiceOperationName = iServiceOperationName ;
[1925]       mServiceOperationLocation = iServiceOperationLocation ;
[1926]       mServiceOperationInput = iServiceOperationInput ;
[1927]       mServiceOperationOutput = iServiceOperationOutput ;
[1928]   }
[1929]   public String getServiceOperationName(){
[1930]   return mServiceOperationName ;
[1931]   }
[1932]   public String getServiceOperationNameToXML(){
[1933]           if(getServiceOperationName() == null)return null ;
[1934]           char[]c = getServiceOperationName().toCharArray() ;
[1935]           StringBuffer sb = new StringBuffer() ;
[1936]           for(int x = 0 ;x<c.length ;x++){
[1937]           switch(c[x]){
[1938]           case' >' :
[1939]           sb.append(" &gt ; " ) ;
[1940]           break ;
[1941]           case' <' :
[1942]           sb.append(" &lt ; " ) ;
[1943]           break ;
[1944]           case' &' :
[1945]           sb.append(" &amp ; " ) ;
[1946]           break ;
[1947]           case" " :
[1948]           sb.append(" &quot ; " ) ;
[1949]           break ;
[1950]           case' \" :
[1951]           sb.append(" &quot ; " ) ;
[1952]           break ;
[1953]           default:
[1954]           if(Character.isDefined(c[x]))
```

```
[1955]          sb.append(c[x]) ;
[1956]          }
[1957]          }
[1958]          return sb.toString() ;
[1959]      }
[1960]  public void setServiceOperationName(String inp){
[1961]          this.mServiceOperationName = inp ;
[1962]  }
[1963]  public void serviceOperationNameFromXML(String n){
[1964]          setServiceOperationName(n) ;
[1965]  }
[1966]      public String getServiceOperationLocation(){
[1967]  return mServiceOperationLocation ;
[1968]  }
[1969]  public String getServiceOperationLocationToXML(){
[1970]          if(getServiceOperationLocation() == null)return null ;
[1971]          char[]c = getServiceOperationLocation().toCharArray() ;
[1972]          StringBuffer sb = new StringBuffer() ;
[1973]          for(int x = 0 ;x<c.length ;x++){
[1974]          switch(c[x]){
[1975]          case' >' :
[1976]          sb.append(" &gt ; " ) ;
[1977]          break ;
[1978]          case' <' :
[1979]          sb.append(" &lt ; " ) ;
[1980]          break ;
[1981]          case' &' :
[1982]          sb.append(" &amp ; " ) ;
[1983]          break ;
[1984]          case" " :
[1985]          sb.append(" &quot ; " ) ;
[1986]          break ;
[1987]          case' \" :
[1988]          sb.append(" &quot ; " ) ;
[1989]          break ;
[1990]          default:
[1991]          if(Character.isDefined(c[x]))
[1992]          sb.append(c[x]) ;
[1993]          }
```

```
[1994]          }
[1995]              return sb.toString() ;
[1996]      }
[1997]      public void setServiceOperationLocation(String inp){
[1998]              this.mServiceOperationLocation = inp ;
[1999]      }
[2000]      public void serviceOperationLocationFromXML(String n){
[2001]              setServiceOperationLocation(n) ;
[2002]  }
[2003]  public String getServiceOperationInput(){
[2004]  return mServiceOperationInput ;
[2005]  }
[2006]  public String getServiceOperationInputToXML(){
[2007]          if(getServiceOperationInput() == null)return null ;
[2008]          char[]c = getServiceOperationInput().toCharArray() ;
[2009]          StringBuffer sb = new StringBuffer() ;
[2010]          for(int x = 0 ;x<c.length ;x++){
[2011]          switch(c[x]){
[2012]          case' >' :
[2013]          sb.append(" &gt ; " ) ;
[2014]          break ;
[2015]          case' <' :
[2016]          sb.append(" &lt ; " ) ;
[2017]          break ;
[2018]          case' &' :
[2019]          sb.append(" &amp ; " ) ;
[2020]          break ;
[2021]          case" " :
[2022]          sb.append(" &quot ; " ) ;
[2023]          break ;
[2024]          case' \" :
[2025]          sb.append(" &quot ; " ) ;
[2026]          break ;
[2027]          default:
[2028]          if(Character.isDefined(c[x]))
[2029]          sb.append(c[x]) ;
[2030]      }
[2031]      }
[2032]      return sb.toString() ;
```

```
[2033]  }
[2034]  public void setServiceOpera tionInput(String inp){
[2035]          this.mServiceOperationInput = inp ;
[2036]  }
[2037]  public void serviceOperationInputFromXML(String n){
[2038]          setServiceOperationInput(n) ;
[2039]  }
[2040]  public String getServiceOperationOutput(){
[2041]  return mServiceOperationOutput ;
[2042]  }
[2043]  public String getServiceOperationOutputToXML(){
[2044]         if(getServiceOperationOutput() == null)return null ;
[2045]          char[]c = getServiceOperationOutput().toCharArray() ;
[2046]          StringBuffer sb = new StringBuffer() ;
[2047]          for(int x = 0 ;x<c.length ;x++){
[2048]          switch(c[x]){
[2049]          case' >' :
[2050]          sb.append(" &gt ; " ) ;
[2051]          break ;
[2052]          case' <' :
[2053]          sb.append(" &lt ; " ) ;
[2054]          break ;
[2055]          case' &' :
[2056]          sb.append(" &amp ; " ) ;
[2057]          break ;
[2058]          case" " :
[2059]          sb.append(" &quot ; " ) ;
[2060]          break ;
[2061]          case' \" :
[2062]          sb.append(" &quot ; " ) ;
[2063]          break ;
[2064]          default:
[2065]      if(Character.isDefined(c[x]))
[2066]      sb.append(c[x]) ;
[2067]      }
[2068]      }
[2069]      return sb.toString() ;
[2070]      }
[2071]      public void setServiceOpera tionOutput(String inp){
```

```
[2072]              this.mServiceOperationOutput = inp;
[2073]      }
[2074]      public void serviceOperationOutputFromXML(String n){
[2075]              setServiceOperationOutput(n);
[2076]      }
[2077]  }
[2078]  package com.veo.xdk.dev.schema.test.blib;
[2079]  import com.veo.vsp.doclet.meta.Document;
[2080]  public class ServiceSet extends Document{
[2081]      public static final String DOC_TYPE =" service.set" ;
[2082]      Service[]mService;
[2083]      public ServiceSet(){
[2084]              super(DOC_TYPE);
[2085]              this.mService = new Service[0];
[2086]      }
[2087]      public ServiceSet(String doc_type){
[2088]              super(doc_type);
[2089]              this.mService = new Service[0];
[2090]      }
[2091]      static public ServiceSet initServiceSet(Service[]iService){
[2092]              ServiceSet obj = new ServiceSet();
[2093]              obj.initializeAll(iService);
[2094]              return obj;
[2095]      }
[2096]      public void initializeAll(Service[]iService){
[2097]              mService = iService;
[2098]      }
[2099]      public Service[]getService(){
[2100]      return mService;
[2101]      }
[2102]      public Service getService(int index){
[2103]               if(this.mService == null)
[2104]               return null;
[2105]               if(index> = this.mService.length)
[2106]               return null;
[2107]               if(index<0&&-index>this.mService.length)
[2108]               return null;
[2109]               if(index> = 0)return this.mService[index];
[2110]               return this.mService[this.mService.length+index];
```

```
[2111]            }
[2112]            public void setService(int index,Service inp){
[2113]            if(this.mService == null){
[2114]                    if(index<0){
[2115]                     this.mService = new Service[1];
[2116]                     this.mService[0] = inp;
[2117]                     }else{
[2118]                     this.mService = new Service[index+1];
[2119]                     this.mService[index] = inp;
[2120]                     }
[2121]            }else if(index<0){
[2122]                                Service[]newService = new Service[this.
mService.length+1];
[2123]                               java.lang.System.arraycopy((Object)mService,
0,
[2124]    (Object)newService,0,this.mService.length);
[2125]                              newService[newService.length-1] = inp;
[2126]                              mService = newService;
[2127]                    }else if(index> = this.mService.length){
[2128]                               Service[]newService = new Service[index+1];
[2129]                               java.lang.System.arraycopy((Object)mService,
0,
[2130]    (Object)newService,0,this.mService.length);
[2131]                              newService[index] = inp;
[2132]                 mService = newService;
[2133]           }else{
[2134]                 this.mService[index] = inp;
[2135]           }
[2136]       }
[2137]       public void setService(Service[]inp){
[2138]              this.mService = inp;
[2139]       }
[2140]   }
```

[2141] 除了上述 JAVA 豆之外,还如下所述产生转换代码,用于从 JAVA 至 XML 的翻译以及从 XML 至 JAVA 的翻译:

```
[2142]       Java to XML
[2143]   <! DOCTYPE tree SYSTEM" tree.dtd" >
[2144]   <tree source =" null" pass-through =" false" >
[2145]   <before>
```

[2146] <vardef name =" attribute.def" >

[2147] <element source =" ATTRIBUTE" class =" NAME" type =" 5" position =" -2" >

[2148] <parse>

[2149] <data class =" java.lang.String" position =" -2" />

[2150] </parse>

[2151] </element>

[2152] </vardef>

[2153] <vardef name =" pcdata.def" >

[2154] <element source =" PCDATA " class =" NAME " type =" 4 " position =" -2" >

[2155] <parse>

[2156] <data class =" 999" type =" 6" position =" -2" />

[2157] </parse>

[2158] </element>

[2159] </vardef>

[2160] <vardef name =" content.def" >

[2161] <element source =" PCDATA" >

[2162] <parse>

[2163] <data class =" 999" type =" 6" position =" -2" />

[2164] </parse>

[2165] </element>

[2166] </vardef>

[2167] <vardef name =" ServiceSet.var" >

[2168] <element source = " com.veo.xdk.dev.schema.test.blib.ServiceSet" class =" service.set" type =" 4"

[2169] position =" -2" >

[2170] <parse>

[2171] <callvar name =" Service.var" />

[2172] </parse>

[2173] </element>

[2174] </vardef>

[2175] <vardef name =" PrototypeService.var" >

[2176] <element source = " com.veo.xdk.dev.schema.test.blib.PrototypeService" class =" prototype.service"

[2177] type =" 4" position =" -2" >

[2178] <parse>

[2179] <callvar name =" pcdata.def" parms =" setSource ServiceNameToXML setGenerator service.name" />

[2180] <callvar name =" pcdata.def" parms =" setSource ServiceTermsToXML setGenerator service.terms" />

[2181] <callvar name = " pcdata.def " parms = " setSource ServiceLocationToXML setGenerator

[2182] service.location" />

[2183] <callvar name =" ServiceOperation.var" />

[2184] </parse>

[2185] </element>

[2186] </vardef>

[2187] <vardef name =" Service.var" >

[2188] <element source = " com.veo.xdk.dev.schema.test.blib. Service" class =" service" type =" 8" position =

[2189] " 0" >

[2190] <parse>

[2191] <callvar name =" pcdata.def" parms =" setSource ServiceNameToXML setGenerator service.name" />

[2192] <callvar name = " pcdata.def " parms = " setSource ServiceLocationToXML setGenerator

[2193] service.location" />

[2194] <callvar name =" ServiceOperation.var" />

[2195] <callvar uame =" pcdata.def" parms =" setSource ServiceTermsToXML setGenerator service.terms" />

[2196] </parse>

[2197] </element>

[2198] </vardef>

[2199] <vardef name =" ServiceOpera tion.var" >

[2200] <element source = " com.veo.xdk.dev.schema.test.blib. ServiceOperation" class =" service.operation"

[2201] type =" 4" position =" -2" >

[2202] <parse>

[2203] <callvar name = " pcdata.def " parms = " setSource ServiceOperationNameToXML setGenerator

[2204] service.operation.name" />

[2205] <callvar name = " pcdata.def " parms = " setSource ServiceOperationLocationToXML setGenerator

[2206] service.operation.location" />

[2207] <callvar name = " pcdata.def " parms = " setSource ServiceOperationInputToXML setGenerator

[2208] service.operation.input" />

```
[2209]        <callvar name = " pcdata.def " parms = " setSource
ServiceOperationOutputToXML setGenerator
[2210]   service.operation.output" />
[2211]       </parse>
[2212]       </element>
[2213]       </vardef>
[2214]       </before>
[2215]       <parse>
[2216]       <callvar name =" ServiceSet.var" />
[2217]       <callvar name =" PrototyPeService.var" />
[2218]       <callvar name =" Service.var" />
[2219]       <callvar name =" ServiceOperation.var" />
[2220]       </parse>
[2221]       </tree>
[2222]       XML to Java
[2223]       <! DOCTYPE tree SYSTEM" tree.dtd" >
[2224]       <tree source =" null" pass-through =" false" >
[2225]       <before>
[2226]       <vardef name =" business.var" >
[2227]       <element source =" business"
[2228]       class =" com.veo.xdk.dev.schema.test.blib.Business"
[2229]       type =" 7" setter =" setBusiness" >
[2230]       <before>
[2231]       <onattribute name =" business.number" >
[2232]       <actions>
[2233]       <callmeth name =" businessNumberFromXML" >
[2234]       <parms>
[2235]       <getattr name =" business.number" />
[2236]       </parms>
[2237]       </callmeth>
[2238]       </actions>
[2239]       </onattribute>
[2240]       </before>
[2241]       <parse>
[2242]       <callvar name =" party.name.var" parms =" setPosition-1" />
[2243]       <callvar name =" address.set.var" />
[2244]       </parse>
[2245]       </element>
[2246]       </vardef>
```

```
[2247]      <vardef name =" party.name.var" >
[2248]        <element source = " party.name " setter
=" partyNameFromXML" position =" -1" class =
[2249]  " java.lang.String" >
[2250]      <parse>
[2251]      <data class =" java.lang.String" position =" 0" />
[2252]      </parse>
[2253]      </element>
[2254]      </vardef>
[2255]      <vardef name =" city.var" >
[2256]       <element source = " city " setter = " cityFromXML " position
=" -1" class =" java.lang.String" >
[2257]      <parse>
[2258]      <data class =" java.lang.String" position =" 0" />
[2259]      </parse>
[2260]      </element>
[2261]  </vardef>
[2262]  <vardef name =" internet.var" >
[2263]  <element source =" internet" setter =" internetFromXML" position
=" -1" class =" java.lang.String" >
[2264]  <parse>
[2265]  <data class =" java.lang.String" position =" 0" />
[2266]  </parse>
[2267]  </element>
[2268]  </vardef>
[2269]  <vardef name =" country.var" >
[2270]  <element source = " country " setter = " countryFromXML " position
=" -1" class =" java.lang.String" >
[2271]  <parse>
[2272]  <data class =" java.lang.String" position =" 0" />
[2273]  </parse>
[2274]  </element>
[2275]  </vardef>
[2276]  <vardef name =" state.var" >
[2277]  <element source = " state " setter = " stateFromXML " position
=" -1" class =" java.lang.String" >
[2278]  <parse>
[2279]  <data class =" java.lang.String" position =" 0" />
[2280]  </parse>
```

[2281] </element>
[2282] </vardef>
[2283] <vardef name =" email.var" >
[2284] <element source = " email " setter = " emailFromXML " position =" -1" class =" java.lang.String" >
[2285] <parse>
[2286] <data class =" java.lang.String" position =" 0" />
[2287] </parse>
[2288] </element>
[2289] </vardef>
[2290] <vardef name =" address.physical.var" >
[2291] <element source =" address.physical"
[2292] class =" com.veo.xdk.dev.schema.test.blib.AddressPhysical"
[2293] type =" 7" setter =" setAddressPhysical" >
[2294] <before>
[2295] </before>
[2296] <parse>
[2297] <callvar name =" street.var" parms =" setPosition-1" />
[2298] <callvar name =" city.var" parms =" setPosition-1" />
[2299] <callvar name =" state.var" parms =" setPosition-1" />
[2300] <callvar name =" postcode.var" parms =" setPosition-1" />
[2301] <callvar name =" country.var" parms =" setPosition-1" />
[2302] </parse>
[2303] </element>
[2304] </vardef>
[2305] <vardef name =" telephone.var" >
[2306] <element source = " telephone " setter =" telephoneFromXML" position =" -1" class =
[2307] " java.lang.String" >
[2308] <parse>
[2309] <data class =" java.lang.String" position =" 0" />
[2310] </parse>
[2311] </element>
[2312] </vardef>
[2313] <vardef name =" person.var" >
[2314] <element source =" person"
[2315] class =" com.veo.xdk.dev.schema.test.blib.Person"
[2316] type =" 7" setter =" setPerson" >
[2317] <before>

[2318] <onattribute name =" SSN" >
[2319] <actions>
[2320] <callmeth name =" sSNFromXML" >
[2321] <parms>
[2322] <getattr name =" SSN" />
[2323] </parms>
[2324] </callmeth>
[2325] </actions>
[2326] </onattribute>
[2327] </before>
[2328] <parse>
[2329] <callvar name =" party.name.var" parms =" setPosition-1" />
[2330] <callvar name =" address.set.var" />
[2331] </parse>
[2332] </element>
[2333] </vardef>
[2334] <vardef name =" fax.var" >
[2335] <element source =" fax" setter =" faxFromXML" position =" -1" class =" java.lang.String" >
[2336] <parse>
[2337] <data class =" java.lang.String" position =" 0" />
[2338] </parse>
[2339] </element>
[2340] </vardef>
[2341] <vardef name =" street.var" >
[2342] <element source = " street " setter = " streetFromXML " position =" -1" class =" java.lang.String" >
[2343] <parse>
[2344] <data class =" java.lang.String" position =" 0" />
[2345] </parse>
[2346] </element>
[2347] </vardef>
[2348] <vardef name =" address.set.var" >
[2349] <element source =" address.set"
[2350] class =" com.veo.xdk.dev.schema.test.blib.AddressSet"
[2351] type =" 7" setter =" setAddressSet" >
[2352] <before>
[2353] </before>
[2354] <parse>

```
[2355]  <callvar name =" address.physical.var" />
[2356]  <callvar name =" telephone.var" parms =" setPosition-1" />
[2357]  <callvar name =" fax.var" parms =" setPosition-1" />
[2358]  <callvar name =" email.var" parms =" setPosition-1" />
[2359]  <callvar name =" internet.var" parms =" setPosition-1" />
[2360]  </parse>
[2361]  </element>
[2362]      </vardef>
[2363]      <vardef name =" postcode.var" >
[2364]          <element source =" postcode " setter
=" postcodeFromXML" position =" -1" class =
[2365]  " java.lang.String" >
[2366]      <parse>
[2367]      <data class =" java.lang.String" position =" 0" />
[2368]      </parse>
[2369]      </element>
[2370]      </vardef>
[2371]      <vardef name =" market.participant.var" >
[2372]      <element source =" market.participant"
[2373]      class =" com.veo.xdk.dev.schema.test.blib.MarketParticipant"
[2374]      type =" 7" position =" 0" >
[2375]      <before>
[2376]      </before>
[2377]      <parse>
[2378]      <callvar name =" business.var" />
[2379]      <callvar name =" person.var" />
[2380]      </parse>
[2381]      </element>
[2382]      </vardef>
[2383]      </before>
[2384]      <parse>
[2385]      <callvar name =" business.var" />
[2386]      <callvar name =" party.name.var" />
[2387]      <callvar name =" city.var" />
[2388]      <callvar name =" internet.var" />
[2389]      <callvar name =" country.var" />
[2390]      <callvar name =" state.var" />
[2391]      <callvar name =" email.var" />
[2392]      <callvar name =" address.physical.var" />
```

[2393] <callvar name =" telephone.var" />

[2394] <callvar name =" person.var" />

[2395] <callvar name =" fax.var" />

[2396] <callvar name =" street.var" />

[2397] <callvar name =" address.set.var" />

[2398] <callvar name =" postcode.var" />

[2399] <callvar name =" market.participant.var" />

[2400] </parse>

[2401] </tree>

[2402] Makefiles :

[2403] #

[2404] #this makefile was generated by bic version 0.0.05/02/1998

[2405] #

[2406] #

[2407] #

[2408] #

[2409] #get the package name from the package argument passed to SchemaGen

[2410] PACKAGE_NAME = com/veo/xdk/dev/schema/test/blib

[2411] JAVA_SOURCES+ = \

[2412] MarketParticipant.java\

[2413] Business.java\

[2414] Person.java\

[2415] Party.java\

[2416] AddressPhysical.java\

[2417] AddressSet.java\

[2418] MAKEFILE_MASTER_DIR = xxx

[2419] include$(MAKEFILE_MASTER_DIR)/Makefile.master

[2420] all::$(JAVA_CLASSES)

[2421] #

[2422] #this makefile was generated by bic version 0.0.05/02/1998

[2423] #

[2424] #

[2425] #

[2426] #

[2427] #get the package name from the package argument passed to SchemaGen

[2428] PACKAGE_NAME = com/veo/xdk/dev/schema/test/blib

[2429] JAVA_SOURCES+ = \

[2430] ServiceSet.java\

[2431] PrototypeService.java\

[2432] Service.java\

[2433] ServiceOperation.java\

[2434] MAKEFILE_MASTER_DIR = xxx

[2435] include$(MAKEFILE_MASTER_DIR)/Makefile.master

[2436] all::$(JAVA_CLASSES)

[2437] 最后，下面是作为一个例子，根据上述模型在运行时间生成的 XML 文档实例：

[2438] <! DOCTYPE market.participant SYSTEM" market.participant.dtd" >

[2439] <market.participant>

[2440] <business business.number =" 1234567890" >

[2441] <party.name>IBM</party.name>

[2442] <address.set>

[2443] <address.physical>

[2444] <street>1IBM Way</street>

[2445] <city>Palo Alto</city>

[2446] <state>CA</state>

[2447] <postcode>94304</postcode>

[2448] <country>USA</country>

[2449] </address.physical>

[2450] <telephone>123456-7890</telephone>

[2451] <fax>1234560987</fax>

[2452] <email>ibmec@ibm.com</email>

[2453] </address.set>

[2454] </business>

[2455] </market.participant>

[2456] <! DOCTYPE service SYSTEM" service.dtd" >

[2457] <service.set>

[2458] <service>

[2459] <service.name>Order Service</service.name>

[2460] <service.location>www.ibm.com/order</service.location>

[2461] <service.operation>

[2462] <service.operation.name>Submit Order</service.operation.name>

[2463] <service.operation.location>www.ibm.com/order/submit</service.location>

[2464] <service.operation.input>urn:x-

[2465] ibm:services:order:operations:po.dtd</service.operation.input>

[2466] <service.operation.output>urn:x-

[2467] ibm:services:order:operations:poack.dtd</service.operation.output>

[2468] </service.operation>

[2469] <service.operation>

[2470] <service.operation.name>Track Order</service.operation.name>

[2471] <service.operation.location>www.ibm.com/order/track</service.location>

[2472] <service.operation.input>urn:x-

[2473] ibm:services:order:operations:track.irequest.dtd</service.operation.input>

[2474] <service.operation.output>urn:x-

[2475] ibm:services:order:operations:track.iresponse.dtd</service.operation.output>

[2476] </service.operation>

[2477] </service>

[2478] </service.set>

[2479] 利用各种工具以及 BID 组合应用程序（提供拖曳，丢弃和形成编辑用户接口），开发器能够建立商业接口定义，并且以 XML 文档形式产生很好形成的、有效的商业接口定义。因此，例举性的运行时间实例是由 Ingram Micro 使用的、用于 IBM 定购服务以及从 IBM 定购膝上型计算机的其它定购服务的商业接口定义。（申请人与 IBM 或 Ingram Micro 之间没有关系）。利用这些过程，用户能够建立一种系统允许用根据本发明定义的文档对商业接口编程。

[2480] 通过以下对购货定单处理的说明，可以进一步理解本发明 CBL 和 BID 处理器在 XML/JAVA 环境下的作用。

[2481] 公司 A 用一个可见的编程环境定义其购货定单文档类型，其中所述编程环境包含一个由 CBL DTD 和模块组成的库，而 CBL DTD 和模块都是用公用商业语言元素定义的，因此它们包括数据类型和其它解释信息。公司 A 的 PO 可以就提供 CBL 库的更一般的“事务文档”范围仅仅包含最小的客户化，或者它可以完全由地址、时期和时间、货币等 CBL 模块来构造。

[2482] 用于一般“事务文档”规范说明的文档（诸如上述 transact.dtd）代表了由模块建立的并且与其它 CBL DTD 互链的 CBL 规范说明的方式。

[2483] 编译程序取得购货定单定义，并且生成几个不同的目标形式。所有这些目标形式可以通过原始规范说明的“树到树”转换来导出。该例子最重要的地方是：

[2484] (a) 用于购货定单的 XML DTD

[2485] (b) JAVA 豆，它封装购货定单的数据结构（建立 JAVA 类、形参、数据类型、方法和例外的结构，对应于购货定单模式定义中的信息）。

[2486] (c) “引渡”程序，它将符合购货定单 DTD 的购货定单转换成购货定单 JAVA 豆，或者将它们加载入数据库，或者建立 HTML（或者 XSL 格式页），用于将购货定单显示在浏览器中。

[2487] (d) “非引渡”程序，它从购货定单 JAVA 豆中抽取数据值，并且将它们转换成与购货定单 DTD 符合的 XML 文档。

[2488] 现在返回脚本。购买应用程序产生一购货定单，该购货定单符合为接受购货定单的供应商指定为服务接口的 DTD。

[2489] 语法分析器用购货定单 DTD 将购货定单实例分解成有关其所含元素和属性值的信息流。然后，通过用 JAVA 代码包裹这些“特性组”，将它们转换成对应的 JAVA 事件对象。事实上，这一转换将标记 XML 文档的片段视作定制编程语言中的指令，其语法由 DTD 定义。现在可以用编译程序生成的引渡应用程序处理这些 JAVA 事件，以便“加载”JAVA 豆数据结构。

[2490] 将 XML 文档转换成一组 JAVA 应用程序的事件以便处理，这不象将语法分析器输出作为内部数据结构保持的正常分析模型，并且处理直到分析结束后才开始。响应 BID 定义的、基于事件的处理是使处理器具有更多功能的关键，因为它允许一发出第一事件就开始处理进发的文档应用程序。

[2491] 用于“收听”各种类型事件的 JAVA 程序由那些事件的模式定义生成。这些收听器是用来完成与 CBL 中 XML 定义相关的商业逻辑的程序，例如与“地址”元素相关的可以是通过检查数据库而使邮政编码有效的代码。这些收听器通过向文档路由器登记来“订购”事件，其中文档路由器将相关的事件引导给所有对它们感兴趣的用户。

[2492] 这种公布和预订体系结构表示可以增加新的收听器程序，而不用知道或不影响现有的收听器程序。每个收听器都有一个队列，路由器将其事件引入该队列，并且该队列可使多个收听器以其自身的步调并行处理事件。

[2493] 对于这里例举的购货定单，可以存在具有以下用途的收听器：

[2494] ●购货定单，它可以与定单输入程序相连，

[2495] ●产品描述，它可以检查存货情况，

[2496] ●地址信息，它可以检查 Fed Ex 或者其它邮递服务，

[2497] ●买方信息，它可以检查定购历史（关于信用度，或者提升，或者根据已知客户是谁来进行类似的处理）。

[2498] 可以通过对原始收听器的配置创建复杂的收听器（例如，购货定单收听器可以包含和调用这里的这些收听器，或者它们可以被自身调用）。

[2499] 图 11 示出了图 1 网络中的市场制造者网点。市场制造者网点包括图 3 系统的基本结构，包括网络接口 1101，文档语法分析器 1102、文档至宿主和宿主至文档的翻译器 1103，以及前端 1104，在本例中称之为路由器。本例中的市场制造者模块 1105 包括一组商业接口定义、或者其它足以为市场中的参与者支持市场制造者功能的标识符、CBL 资源库，以及编译程序，所有这些都为市场参与者服务。路由器 1104 包括参与者登记和文档过滤器，它们对应于翻译器输出处生成的事件并由语法分析器根据参与者登记和收听器之间的元素和属性过滤器来路由选择输入文档，将它们提供给 XML 事件生成器。因此，市场中的某些参与者可以登记接收满足预定参数的文档。例如，可以用下述输入文档在路由器 1104 处过滤文档，所述输入文档根据特定 DTD，并且包括诸如产品预购数目大于阈值，或者预购文档请求的最高价格等属性。然后，只将那些在路由器 1104 处与参与者登记中所登记的信息相一致的文档传送给登记过的参与者。

[2500] 路由器 1104 还可以提供本地主服务 1105 和 1106，因此起市场中参与者以及市场制造者的作用。一般来说，遍历路由器 1104 接收到的文档，以便确定这些文档应送达的目的地，这里如果必须可以再次通过翻译器 1103 传回，并且传出网络接口 1101，到达各自的目的地。

[2501] 市场制造者是一个服务器，它将一组内部的和外部的商业服务捆绑在一起，以产生一个虚似的企业或交易团体。服务器通过将产品数据请求切换至目录服务器，或者将购货定单发送给 ERP 系统，来对输入文档进行语法分析，并且调用适当的服务。服务器还处理翻译任务，将来自公司 XML 文档的信息映像到交易当事方所用的文档格式以及其传统系统所要求的数据格式。

[2502] 对于上述服务定义，当公司发出一购货定单时，服务器中的 XML 语法分析器用购货定单 DTD 将购货定单实例转换成信息事件流。然后，将这些事件传送给这样的应用程序，这些应用程序被编制成处理具有某个类型的事件；在一些情况下，通过互联网将信息完整地发送给不同的企业。在购货定单的例子中，若干应用程序可以对来自语法分析器的信息起作用：

[2503] ●定单输入程序将购货定单处理成一个完整的消息；

[2504] ● ERP 系统检查购货定单中所述产品的存货；

[2505] ●客户数据库验证或更新客户的地址；

[2506] ●运输公司使用地址信息安排递送的日程；

[2507] ●银行用信用卡信息给交易授权。

[2508] 交易当事方只需要就他们交换的商业文档的结构、内容和序列达成一致，不需要对 API 的细节达成一致。如何处理文档以及产生什么动作是严格由提供服务的企业来决定的。这提高了从系统级到企业级的集成。这使得企业表现出与其它生意伙伴的清楚且稳定的接口，尽管其内部技术实施、组织或处理中存在变化。

[2509] 图 12、13 和 14 示出了在图 11 系统中，在市场制造者网点处执行的过程。在图 12 中，在网络接口处，接收来自原发参与者网点的输入文档（步骤 1200）。对文档进行语法分析（步骤 1201）。将文档翻译成宿主的格式，例如从 XML 至 JAVA（步骤 1202）。然后将宿主格式化的事件和对象传送给路由器服务（步骤 1203）。识别被登记用来根据文档类型和文档内容来接收文档的服务（步骤 1204）。将文档或文档的一部分传送给被识别的服务（步骤 1205）。响应于文档内容进行服务（步骤 1206）。产生服务的输出数据（步骤 1207）。将输出转换成文档格式，例如从 JAVA 格式转换成 XML 格式（步骤 1208）。最后，将输出文档发送给参与者网点（步骤 1209）。

[2510] 登记服务是一种由路由器管理的功能。因此，如图 13 所示，在网络接口处接受市场参与者文档（步骤 1300）。将市场参与者文档存储在市场制造者网点的商业接口定义资源库（步骤 1301）中。另外，对文档进行语法分析（步骤 1302）。将分析过的文档翻译成宿主的格式（步骤 1303）。接下来，将文档传送给路由器服务（步骤 1304）。路由器服务包括一个收听器，它根据文档类型和内容，将登记服务识别为文档的目的地（步骤 1305）。将文档或文档的元素传送给登记服务 (1306)。在登记服务中，根据商业接口定义检索所需的服务规范说明（步骤 1307）。如果在步骤 1308 集合服务规范说明，那么根据商业接口定义和服务规范说明设定路由器服务过滤器（步骤 1309）。产生登记确认数据（步骤 1310）。登记确认数据转换成文档格式（步骤 1311）。最后，将确认文档发送给参与者网点，向参与者表示已向市场制造者作了成功的登记（步骤 1312）。

[2511] 图 14 例示了步骤 1307 中集合所需服务规范说明的过程。该过程的开始是，对市场参与者所支持的服务商业接口定义进行定位（步骤 1400）。例如，通过电子邮件交易或

者对资源库网点进行网络访问，来检索服务定义（步骤 1401）。将服务规范说明存储在 BID 资源库中（步骤 1402）。对服务商业接口定义文档进行语法分析（步骤 1403）。将分析过的文档翻译成主格式（步骤 1404）。将主对象传送给路由器服务（步骤 1405）。根据文档类型和内容识别登记服务（步骤 1406）。最后，根据图 13 的过程，将服务商业接口定义文档中的信息传送给登记服务（步骤 1407），供使用。

[2512] 图 15 示出了处理器、构件以及根据本发明在市场制造者网点处处理输入数据的序列。市场制造者网点包括在网络接口处的通信代理 1500。通信代理与 XML 语法分析器 1501 相连，而 XML 语法分析器将事件提供给 XML 处理器 1502。XML 处理器将事件提供给文档路由器。文档路由器提供一文档服务 1504，而文档服务 1504 又提供一个接口，用于将接收到的文档提供给主系统中的企业解答软件 1505。通信代理 1500 是互联网接口，它包括支持诸如 HTTP、SMTP、FTP 或其它协议的适当协议堆栈。因此，输入数据可以按 XML 语法、ASCII 数据语法或其它适合特定通信信道的语法而到来。将所有以非 XML 语法接收到的文档翻译成 XML，并且传送给 XML 语法分析器。用翻译表 1506 支持从非 XML 形式至 XML 形式的翻译。

[2513] 将转换后的文档提供给语法分析器 1501。XML 语法分析器根据与其匹配的文档类型定义对接收到的 XML 文档进行语法分析。如果发现错误，那么语法分析器将文档发回通信代理 1500。商业接口定义编译器 BIDC 1507 起商业接口定义数据编译器的作用。通过编译 BID 数据，建立 XML 语法分析器的 DTD 文件、对应于 DTD 文件的 JAVA 豆以及将 DTD 文件翻译成 JAVA 豆的翻译规则。通过参考这些工具，将 XML 实例翻译成 JAVA 实例。因此，BID 编译器 1507 存储 DTD 文档 1508，并产生对应的 JAVA 文档 1509。将 XML 文档传送给将它们翻译成 JAVA 格式的处理器 1502。在一较佳系统中，产生状态与按 XML 格式接收到的文档类型定义相同的 JAVA 文档。将 JAVA 豆传送给文档路由器 1503。文档路由器 1503 接收 JAVA 豆，并且用登记程序（例如用上述事件收听器体系结构）将接收到的类传送给合适的文档服务。从路由器 1503 接收 JAVA 豆形式的文档的文档服务 1504 起企业解答软件之接口的作用。这包括登记服务 1510，XML 事件的收听器通过该登记服务与输入数据流耦合，并且还包括服务管理器 1511，用于管理将输入文档传送给合适的服务。文档服务管理器 1511 对登记服务进行管理，并且保持文档一致性。

[2514] 文档服务用任何专用 API 或者用诸如 CORBA/COM 接口或其它体系结构等更普通的形式，与后端系统通信。

[2515] 图 16 提供了本发明市场制造者和市场参与者结构的启发图。因此，如图 16 所示，可以有逻辑地组织本发明的电子商务市场。在组织的顶部，建立市场制造者网点 1600。市场制造者网点包括用于建立市场 1601 的资源。这类资源包括市场登记服务等。企业 1602 通过公布商业接口定义在市场 1601 中登记。商业接口定义定义了企业将参与的商务交易的服务 1603。事务 1604 和服务 1603 用文档 1605 定义输入和输出，并且划定了交易中参与者之间的贸易关系。文档包括内容 1606，而内容 1606 载有每宗交易的参与者。市场参与者和市场制造者处理内容所用的方式完全独立于依照本发明建立的基于文档的电子商务网。总的来说，提供了一种稳健的、规模可变的、直观的结构，它能够提供通信网上的电子商务。

[2516] 因此，本发明在一例举的系统中提供了一种基于 XML 处理器的平台，并且将 XML 文档用作松散连接的商业系统之间的接口。文档在企业之间传递，并且在进入公司商业系统

之前由参与者网点处理。因此，平台能够使电子商务应用程序在企业之间工作，每个商业系统通过指定一组公用的商业文档和形式，用不同的内部商务平台、过程和语义进行工作。

[2517] 依照本发明，通过互连商业系统和服务建立了虚拟的企业，并且主要用企业接受并产生的文档（XML 编码的）对虚拟企业定义：

[2518] ●“如果你发给我一个目录请求，我将发给你一个目录”

[2519] ●“如果你发给我一份购货定单，并且我可以接受，那么我将发给你发票”。

[2520] 为了说明和描述的目的，提供了上述对本发明较佳实施例的描述。并不试图将本发明穷尽或限制到所揭示的精确形式。显然，对于本领域的技术人员来说，可以进行许多变化和改变。本发明的范围由以下权利要求书及其等效权利来限制。

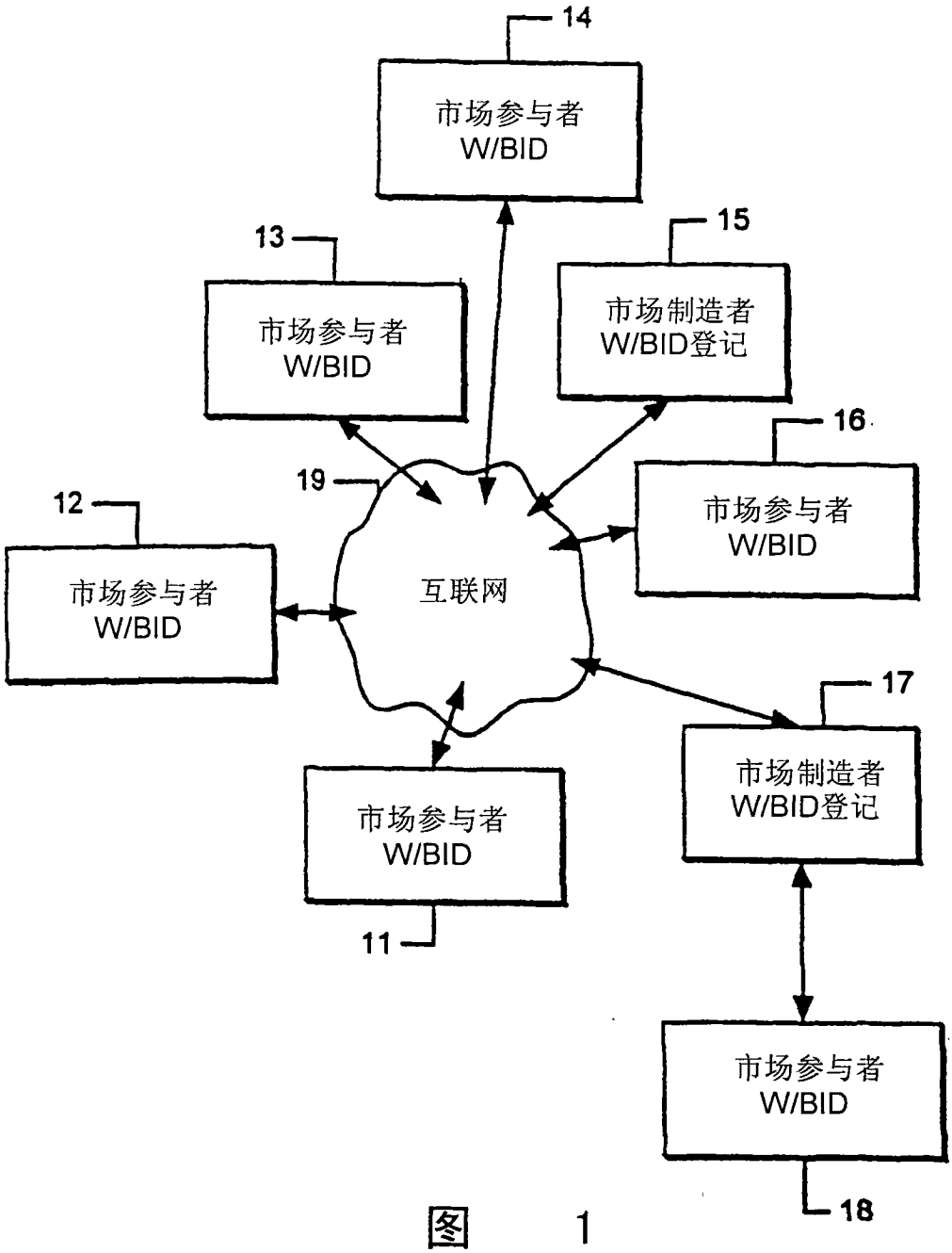

14
市场参与者
W/BID
13
市场参与者
W/BID
15
市场制造者
W/BID登记
16
市场参与者
W/BID
19
互联网
12
市场参与者
W/BID
17
市场制造者
W/BID登记
市场参与者
W/BID
11
市场参与者
W/BID
18

图 1

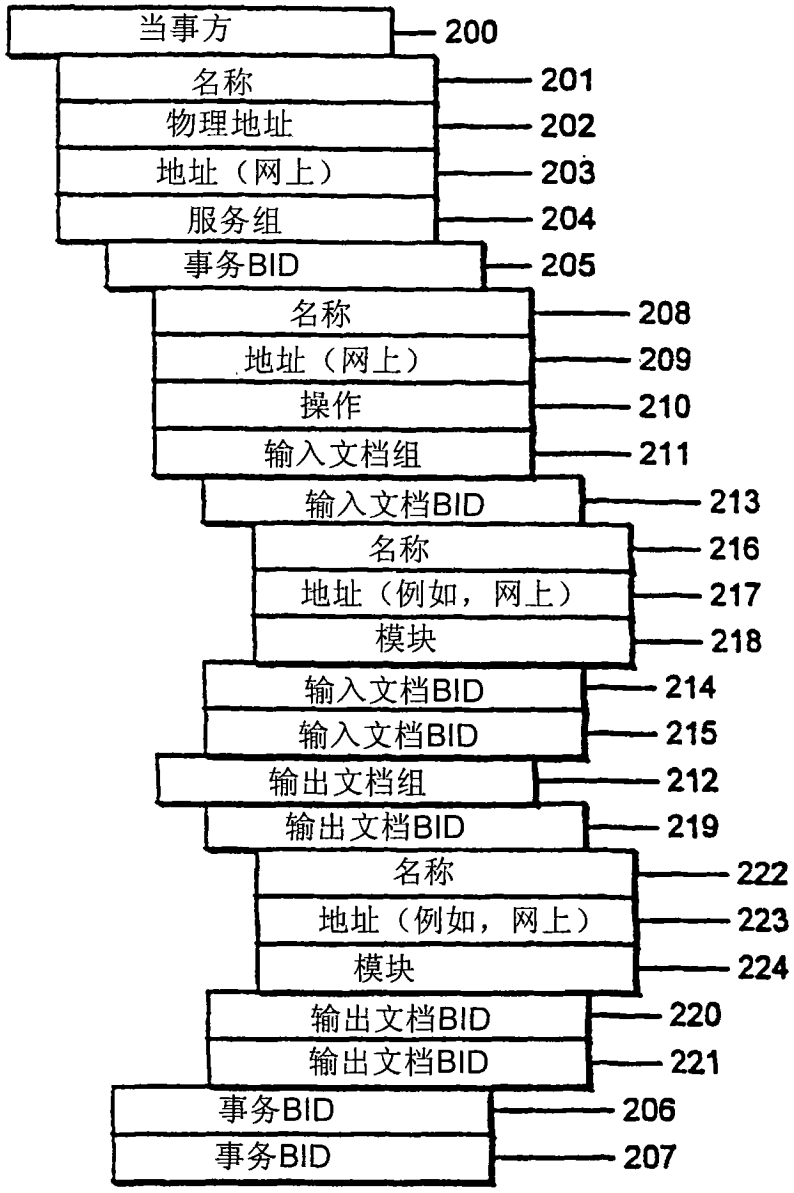

当事方
200
名称
201
物理地址
202
地址（网上）
203
服务组
204
事务BID
205
名称
208
地址（网上）
209
操作
210
输入文档组
211
输入文档BID
213
名称
216
地址（例如，网上）
217
模块
218
输入文档BID
214
输入文档BID
215
输出文档组
212
输出文档BID
219
名称
222
地址（例如，网上）
223
模块
224
输出文档BID
220
输出文档BID
221
事务BID
206
事务BID
207

图 2

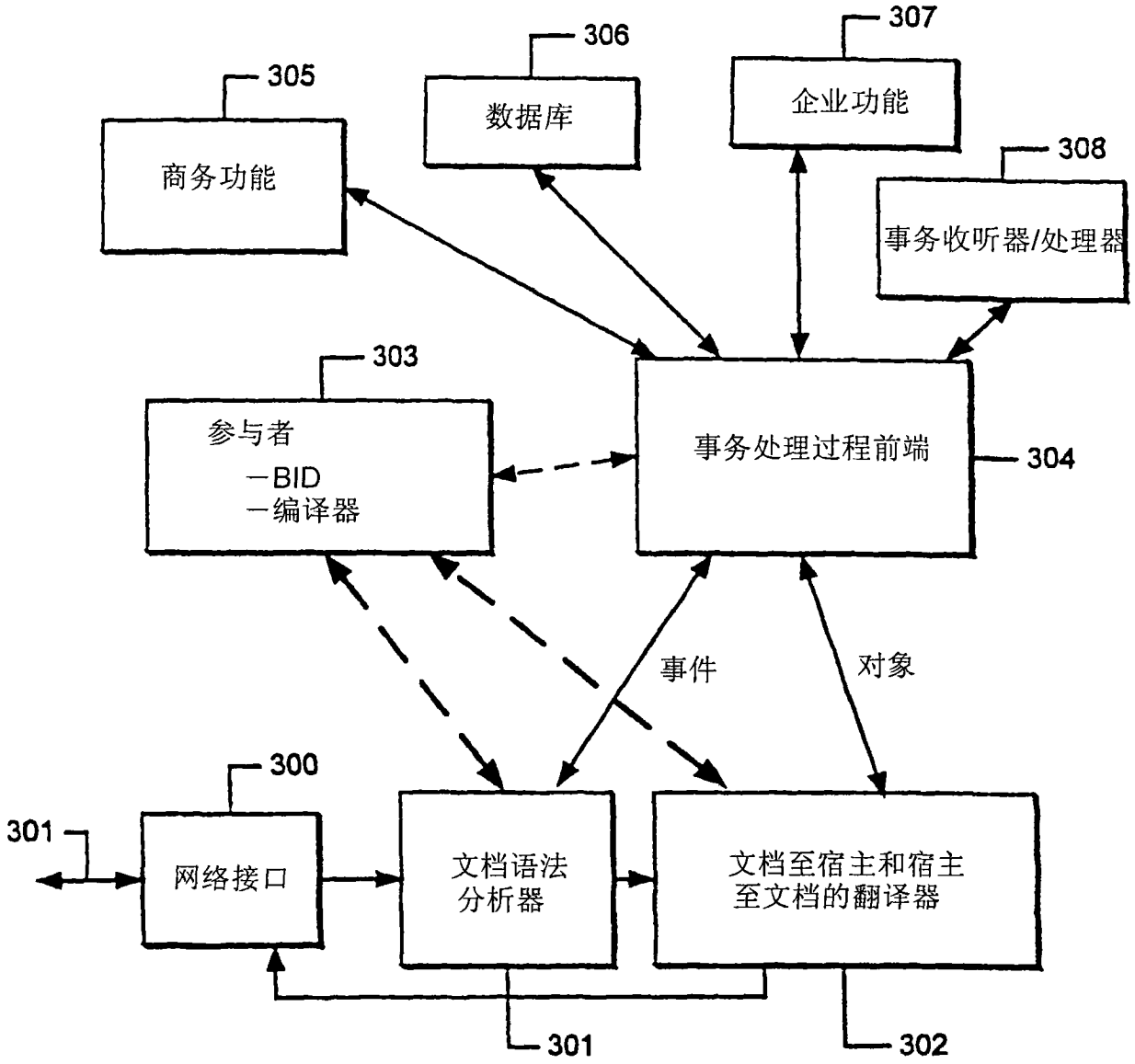

305
商务功能
306
数据库
307
企业功能
308
事务收听器/处理器
303
参与者
—BID
—编译器
事务处理过程前端
304
事件
对象
300
301
网络接口
文档语法
分析器
文档至宿主和宿主
至文档的翻译器
301
302

图 3

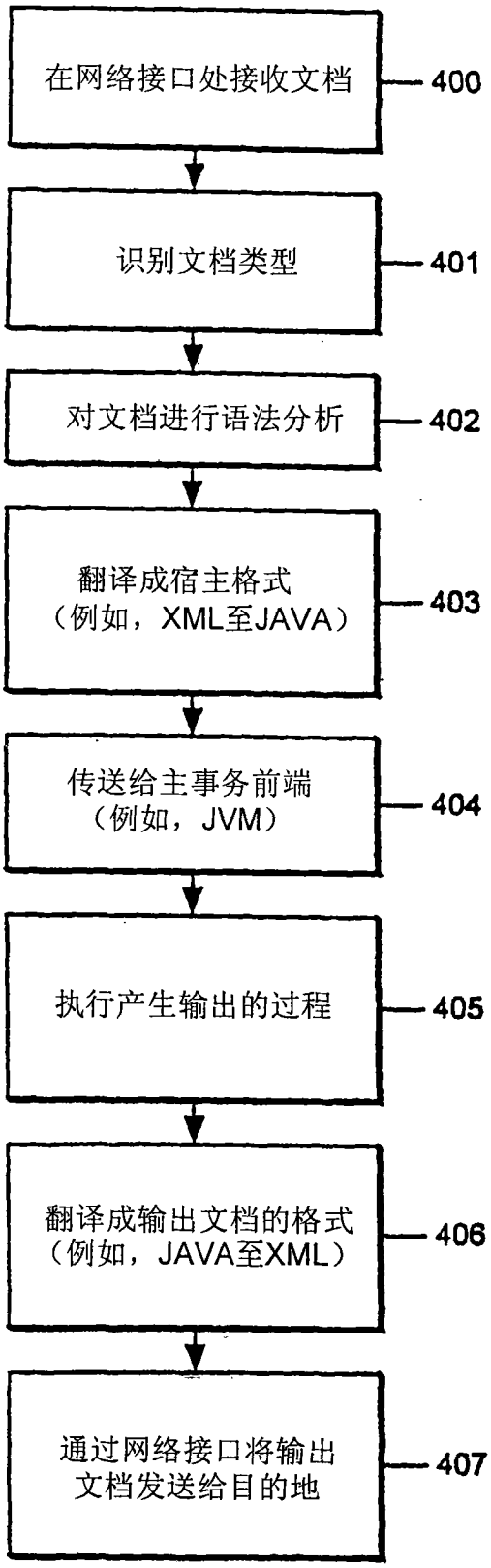

在网络接口处接收文档
400
识别文档类型
401
对文档进行语法分析
402
翻译成宿主格式
（例如，XML至JAVA）
403
传送给主事务前端
（例如，JVM）
404
执行产生输出的过程
405
翻译成输出文档的格式
（例如，JAVA至XML）
406
通过网络接口将输出
文档发送给目的地
407

图 4

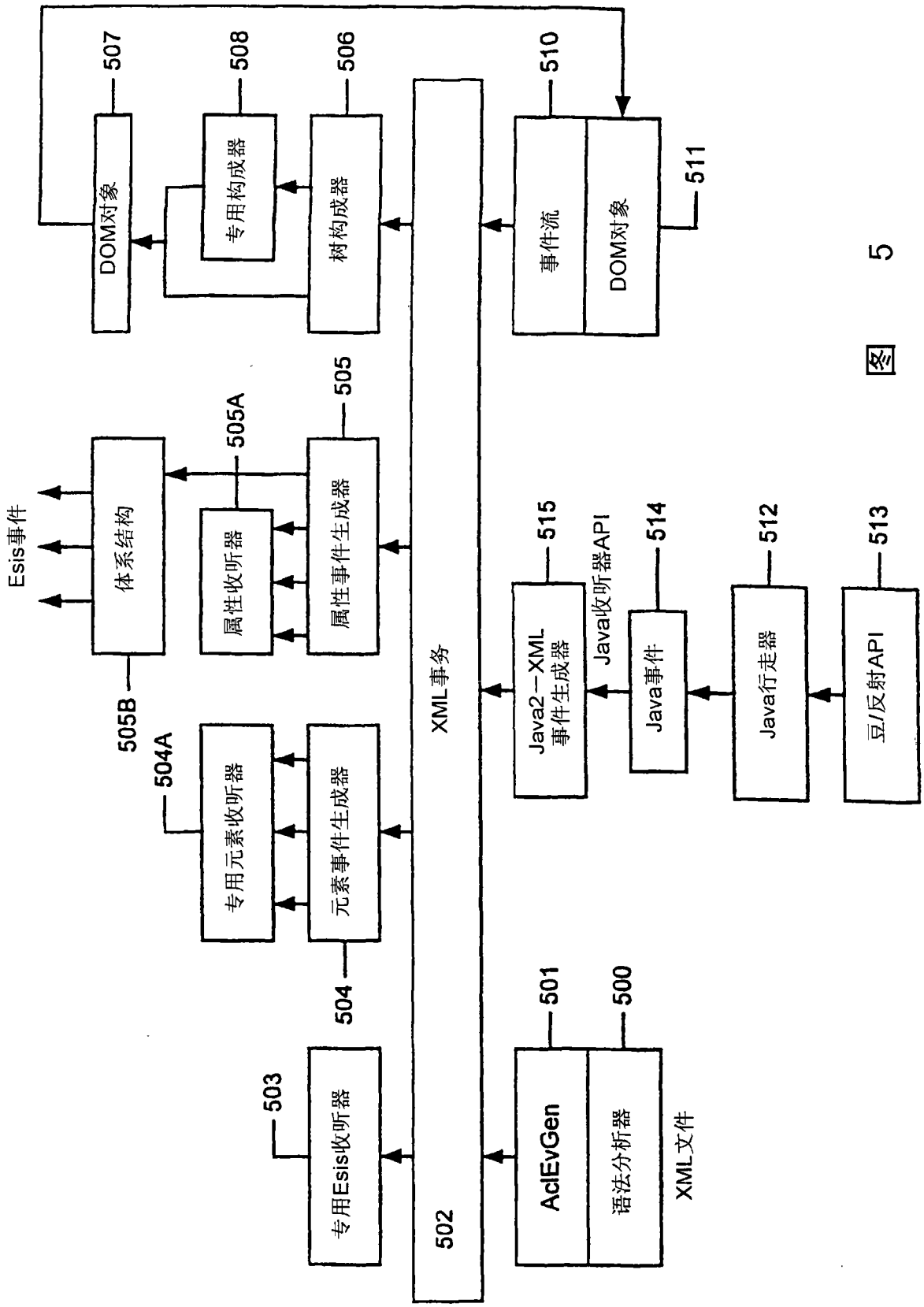

DOM对象
507
专用构成器
508
树构成器
506
事件流
510
DOM对象
511
Esis事件
体系结构
505B
505A
属性收听器
属性事件生成器
505
504A
专用元素收听器
元素事件生成器
504
XML事务
Java2—XML
事件生成器
515
Java收听器API
Java事件
514
Java行走器
512
豆/反射API
513
专用Esis收听器
503
AclEvGen
501
语法分析器
500
XML文件
502

图 5

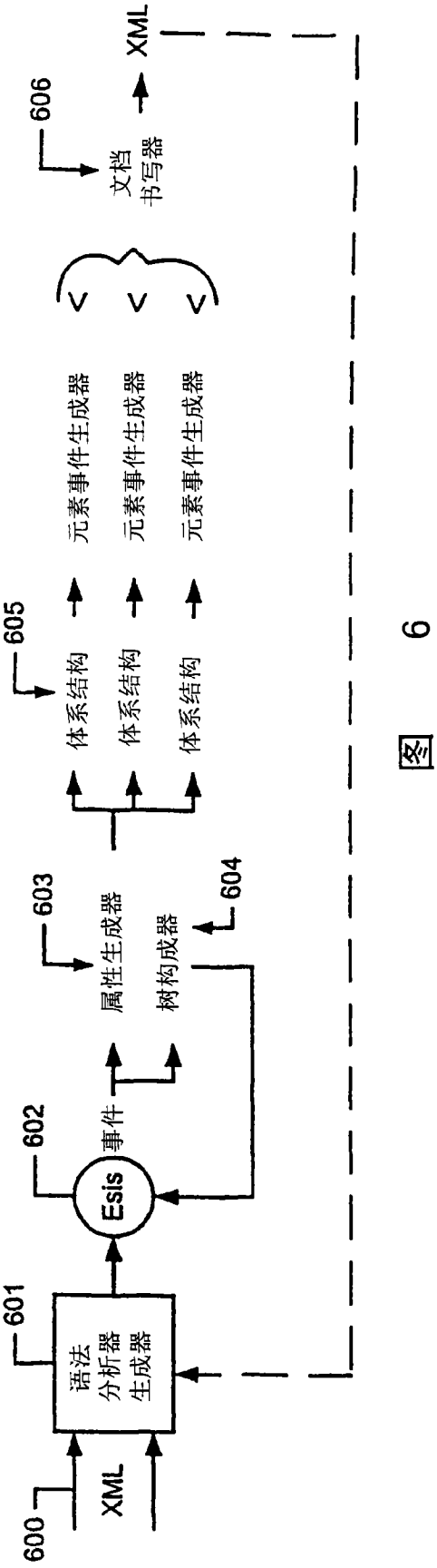

600
XML
601
语法
分析器
生成器
602
Esis
事件
603
属性生成器
604
树构成器
605
体系结构
体系结构
体系结构
元素事件生成器
元素事件生成器
元素事件生成器
606
文档
书写器
XML

图 6

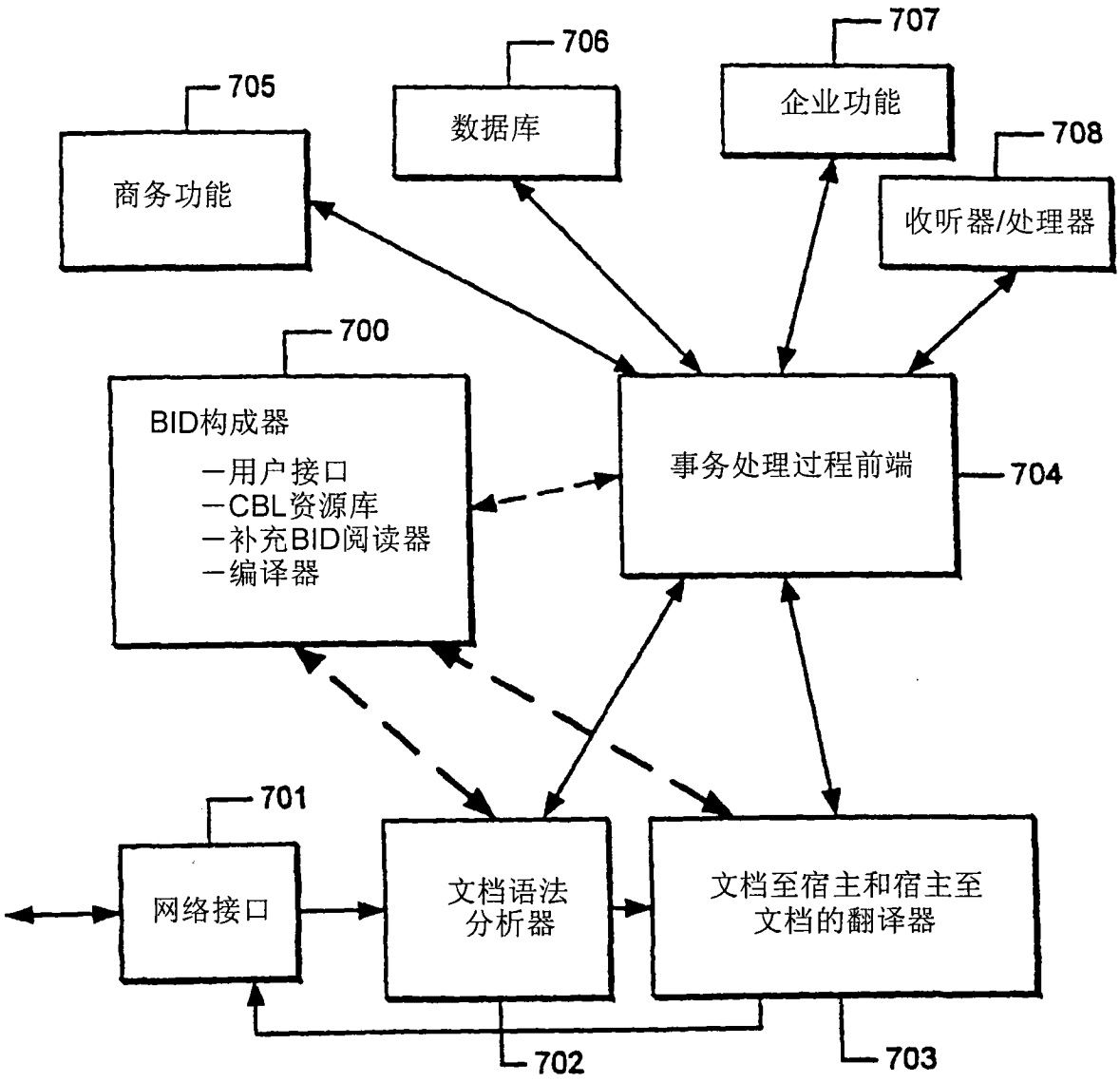

705
商务功能
706
数据库
707
企业功能
708
收听器/处理器
700
BID构成器
—用户接口
—CBL资源库
—补充BID阅读器
—编译器
事务处理过程前端
704
701
网络接口
文档语法
分析器
文档至宿主和宿主至
文档的翻译器
702
703

图 7

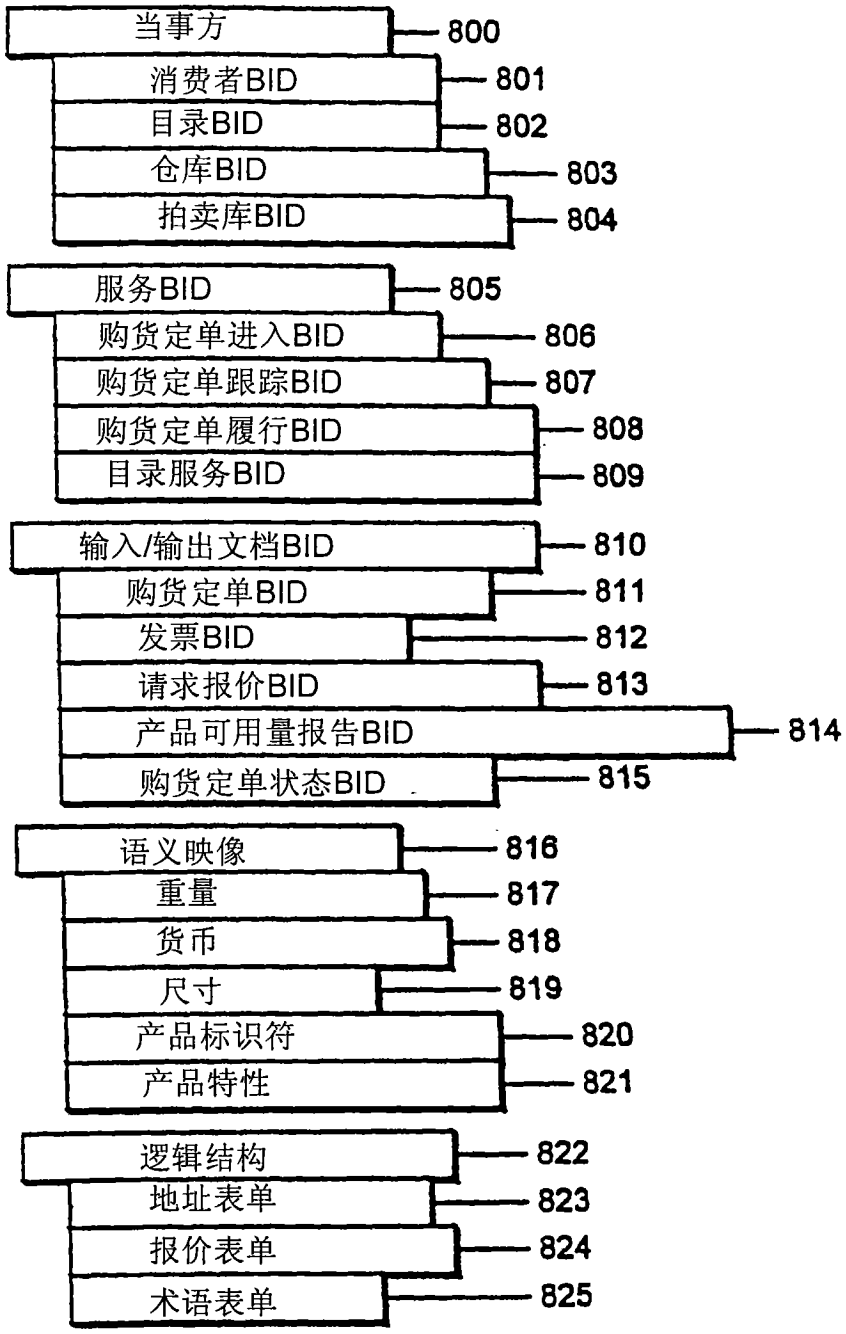

当事方
800
消费者BID
801
目录BID
802
仓库BID
803
拍卖库BID
804
服务BID
805
购货定单进入BID
806
购货定单跟踪BID
807
购货定单履行BID
808
目录服务BID
809
输入/输出文档BID
810
购货定单BID
811
发票BID
812
请求报价BID
813
产品可用量报告BID
814
购货定单状态BID
815
语义映像
816
重量
817
货币
818
尺寸
819
产品标识符
820
产品特性
821
逻辑结构
822
地址表单
823
报价表单
824
术语表单
825

图 8

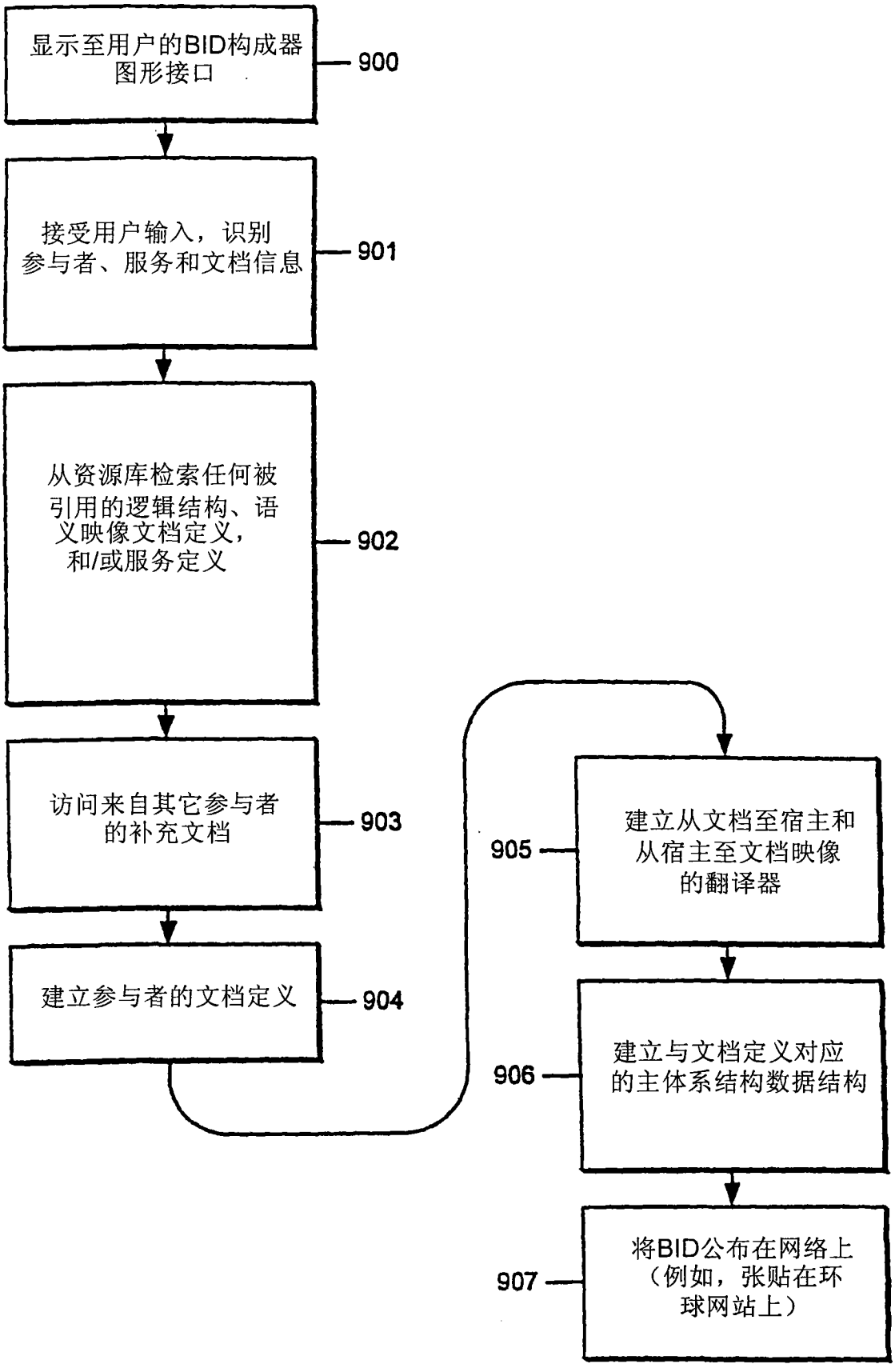

显示至用户的BID构成器图形接口
900
接受用户输入，识别参与者、服务和文档信息
901
从资源库检索任何被引用的逻辑结构、语义映像文档定义，和/或服务定义
902
访问来自其它参与者的补充文档
903
建立参与者的文档定义
904
905
建立从文档至宿主和从宿主至文档映像的翻译器
906
建立与文档定义对应的主体系结构数据结构
907
将BID公布在网络上（例如，张贴在环球网站上）

图 9

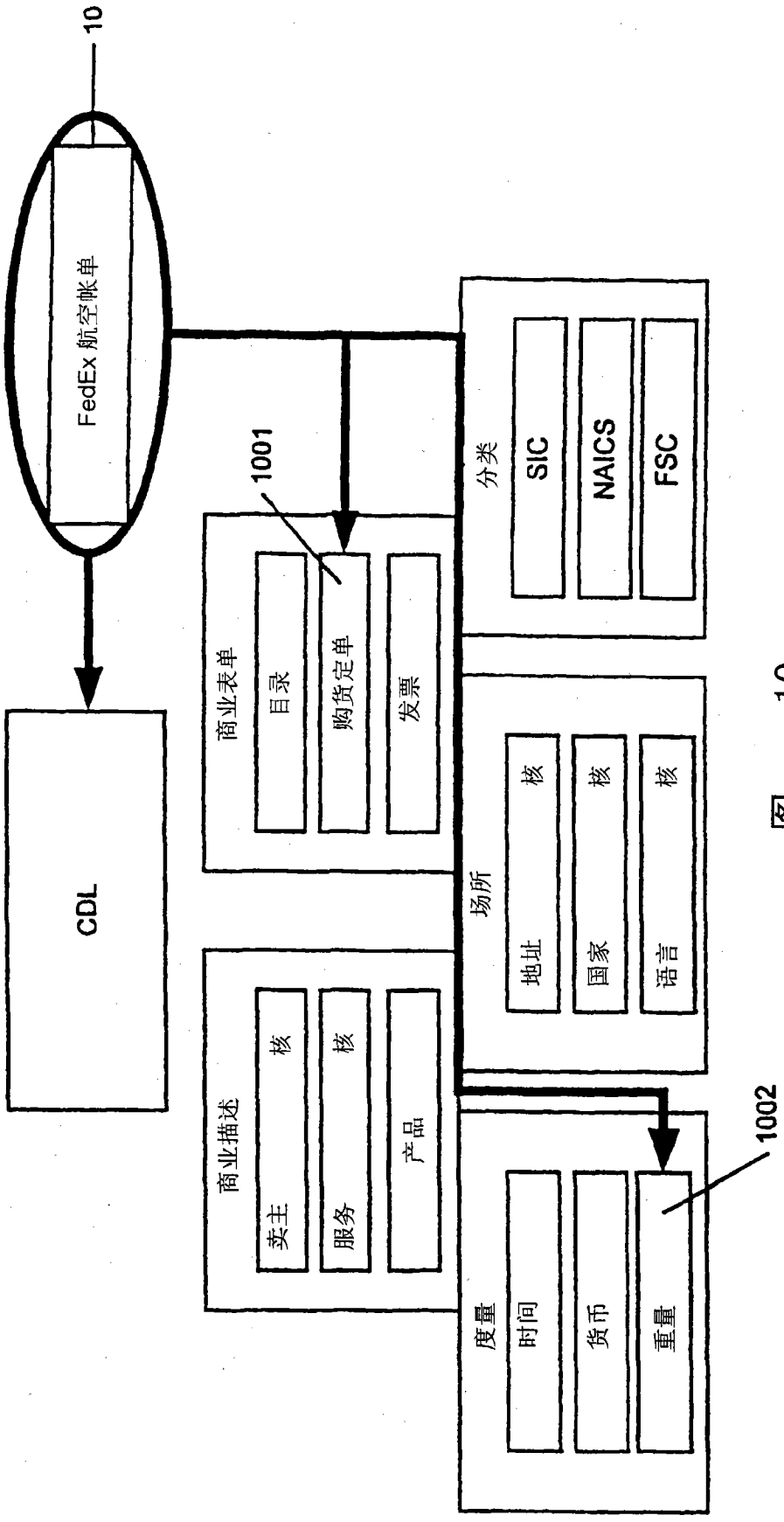

10
FedEx 航空帐单
CDL
1001
商业表单
目录
购货定单
发票
商业描述
英主
核
服务
核
产品
分类
SIC
NAICS
FSC
场所
地址
核
国家
核
语言
核
度量
时间
货币
重量
1002

图 10

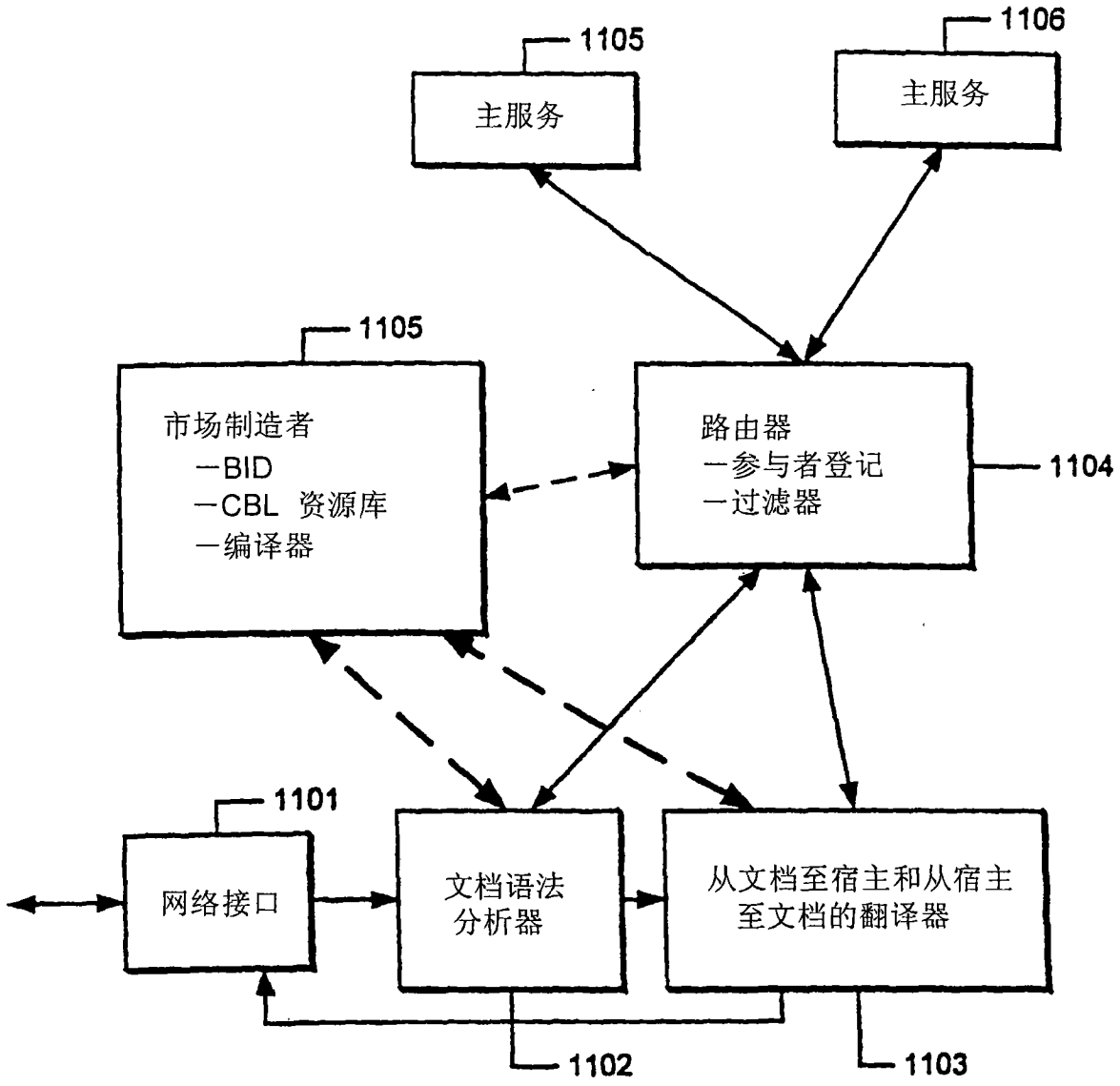

1105
主服务
1106
主服务
1105
市场制造者
—BID
—CBL 资源库
—编译器
路由器
—参与者登记
—过滤器
1104
1101
网络接口
文档语法
分析器
从文档至宿主和从宿主
至文档的翻译器
1102
1103

图 11

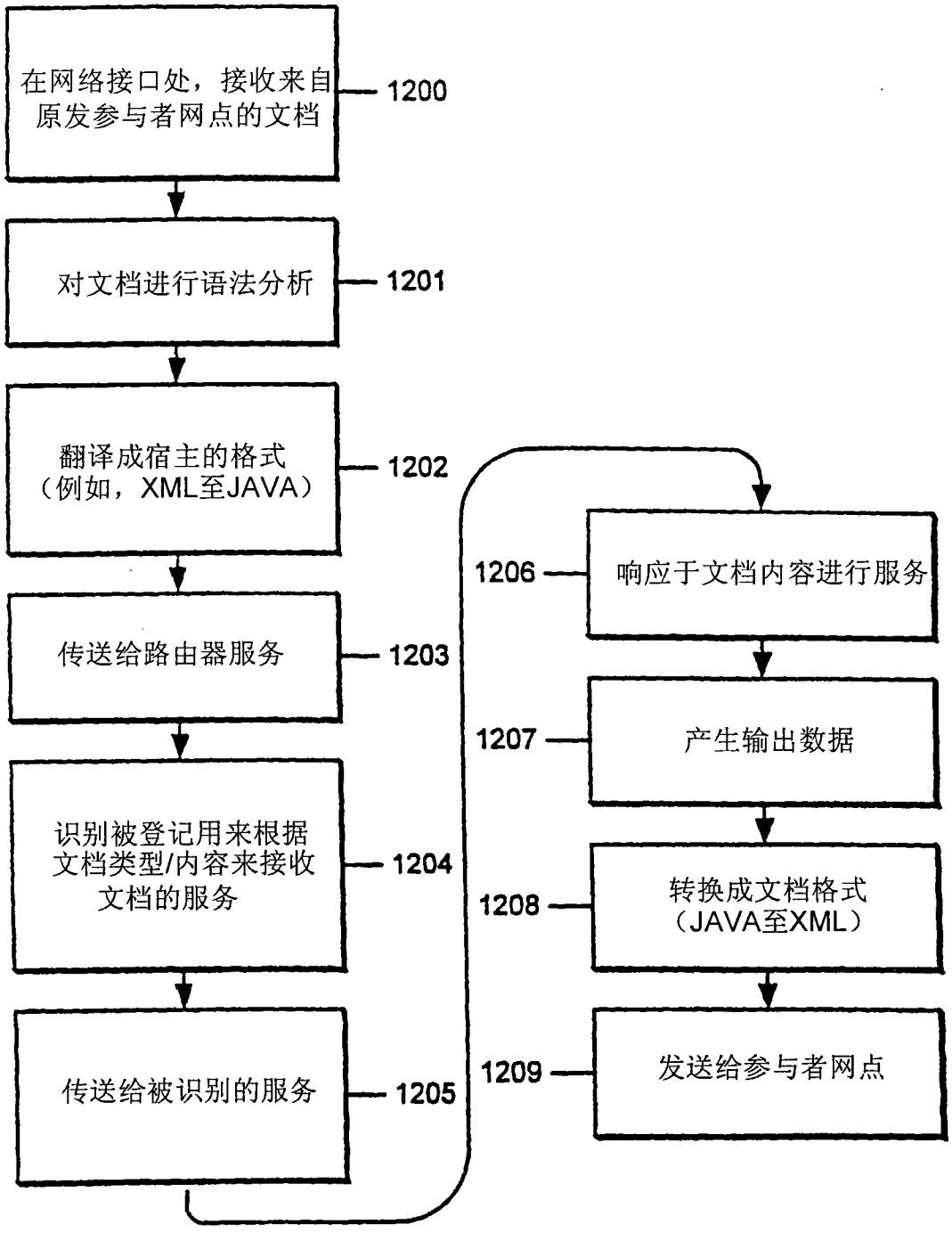

在网络接口处，接收来自原发参与者网点的文档
1200
对文档进行语法分析
1201
翻译成宿主的格式（例如，XML至JAVA）
1202
传送给路由器服务
1203
识别被登记用来根据文档类型/内容来接收文档的服务
1204
传送给被识别的服务
1205
1206
响应于文档内容进行服务
1207
产生输出数据
1208
转换成文档格式（JAVA至XML）
1209
发送给参与者网点

图 12

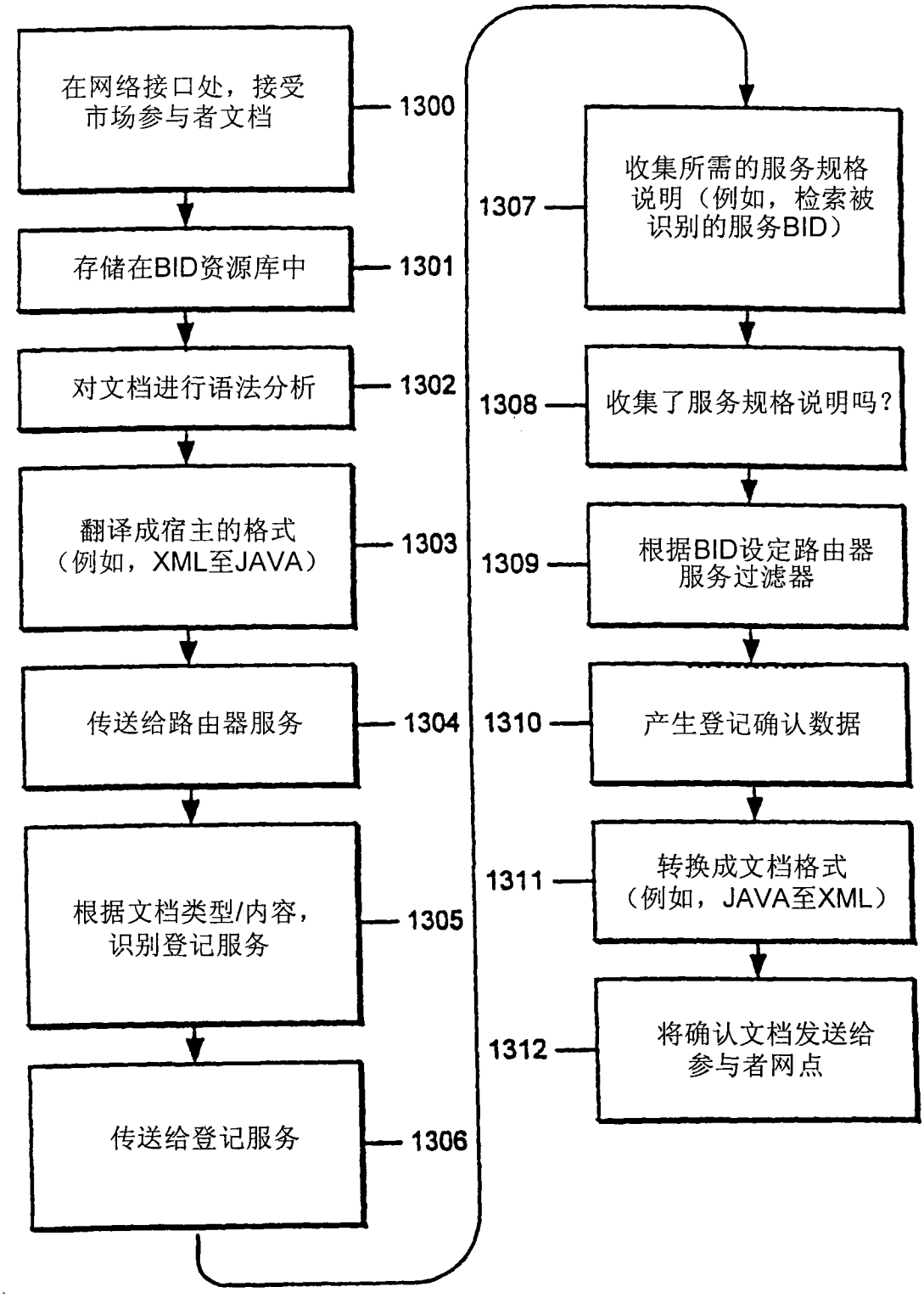

在网络接口处，接受市场参与者文档
1300
存储在BID资源库中
1301
对文档进行语法分析
1302
翻译成宿主的格式（例如，XML至JAVA）
1303
传送给路由器服务
1304
根据文档类型/内容，识别登记服务
1305
传送给登记服务
1306
收集所需的服务规格说明（例如，检索被识别的服务BID）
1307
收集了服务规格说明吗？
1308
根据BID设定路由器服务过滤器
1309
产生登记确认数据
1310
转换成文档格式（例如，JAVA至XML）
1311
将确认文档发送给参与者网点
1312

图 13

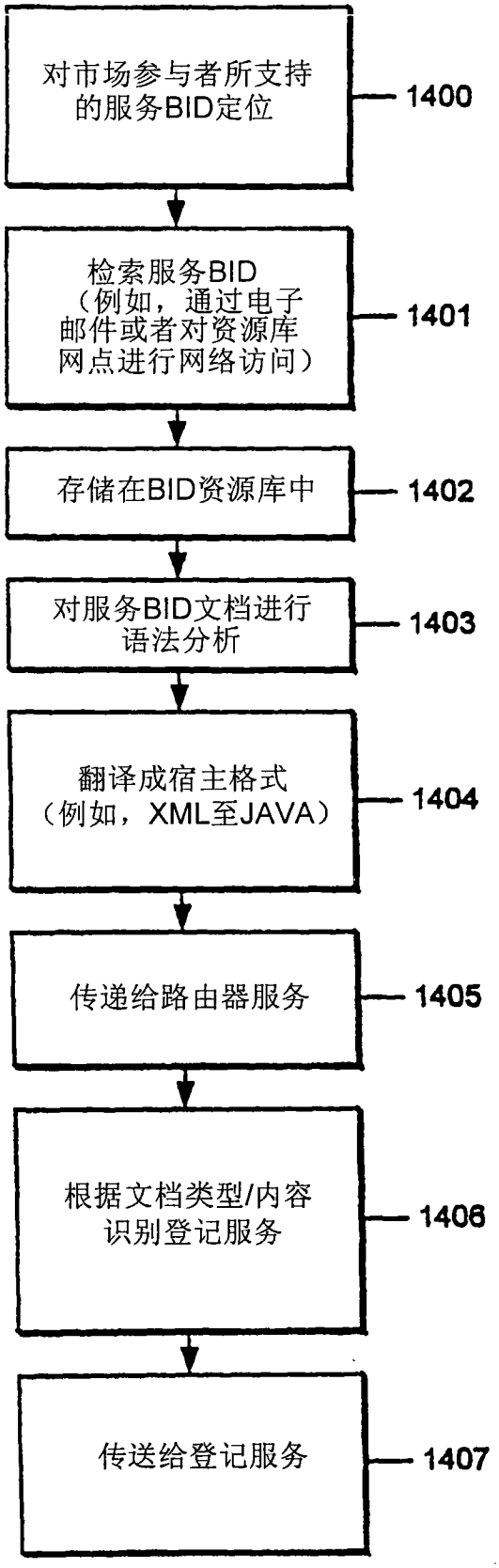

对市场参与者所支持的服务BID定位
1400
检索服务BID（例如，通过电子邮件或者对资源库网点进行网络访问）
1401
存储在BID资源库中
1402
对服务BID文档进行语法分析
1403
翻译成宿主格式（例如，XML至JAVA）
1404
传递给路由器服务
1405
根据文档类型/内容识别登记服务
1406
传送给登记服务
1407

图 14

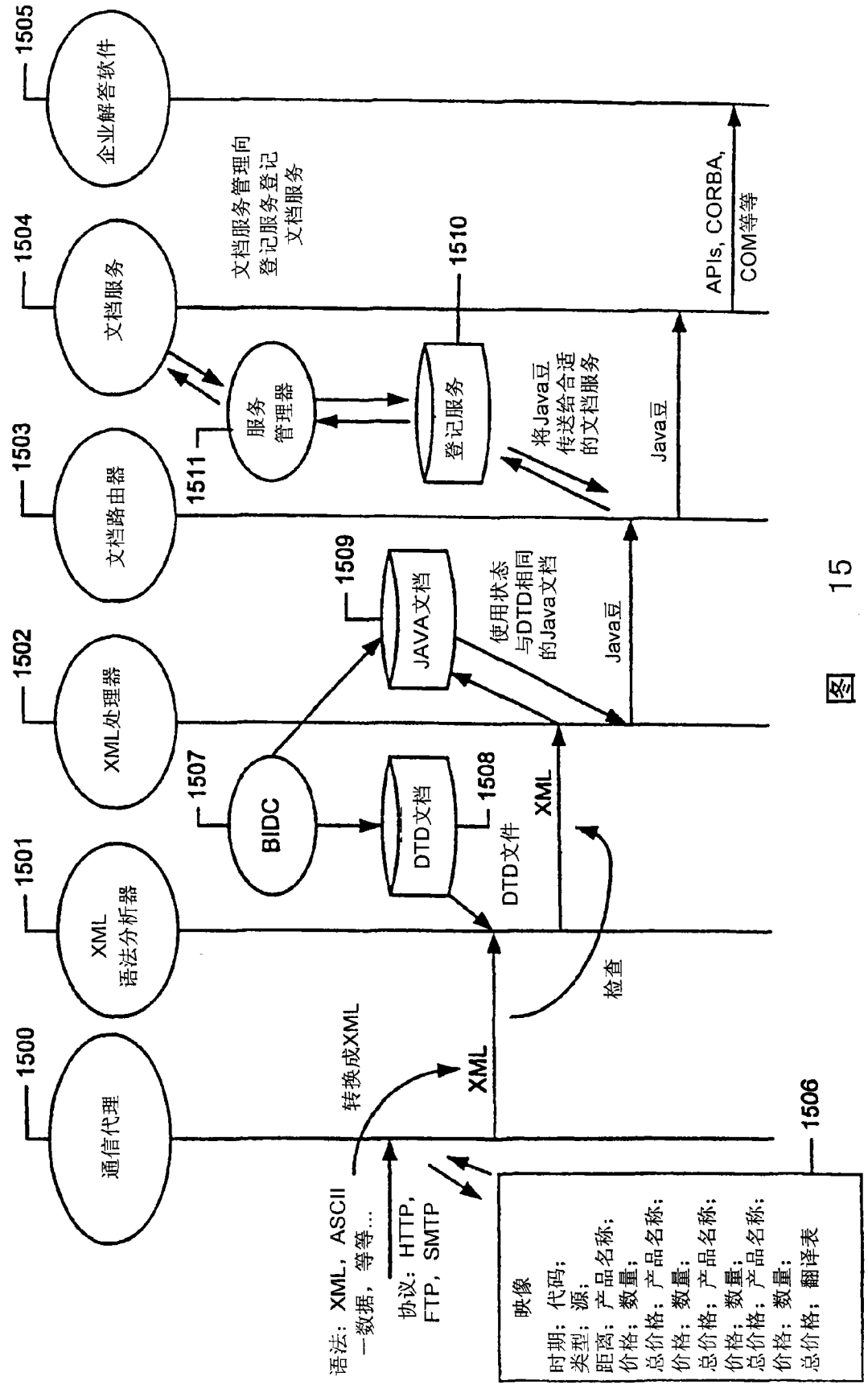

1500
通信代理
1501
XML
语法分析器
1502
XML处理器
1503
文档路由器
1504
文档服务
1505
企业解答软件
1506
1507
BIDC
1508
DTD文档
DTD文件
1509
JAVA文档
1510
1511
服务
管理器
登记服务
文档服务管理向
登记服务登记
文档服务
将Java豆
传送给合适
的文档服务
使用状态
与DTD相同
的Java文档
Java豆
Java豆
APIs, CORBA,
COM等等
XML
XML
转换成XML
检查
语法：XML，ASCII
—数据，等等...
协议：HTTP,
FTP，SMTP
映像
时期：代码；
类型：源；
距离：产品名称；
价格：数量；
总价格：产品名称；
价格：数量；
总价格：产品名称；
价格：数量；
总价格：产品名称；
价格：数量；
总价格：翻译表

图 15

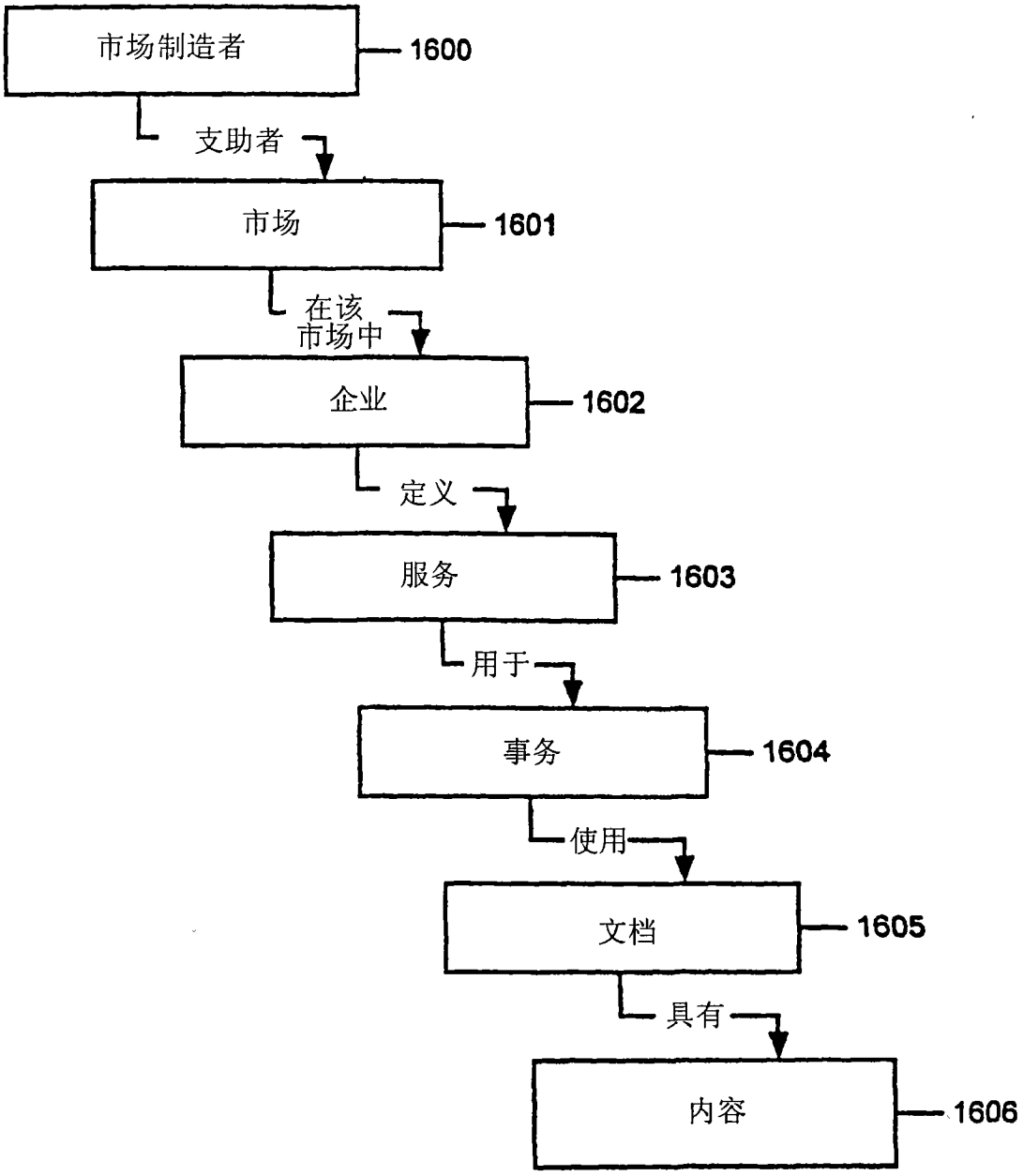

市场制造者
1600
支助者
市场
1601
在该
市场中
企业
1602
定义
服务
1603
用于
事务
1604
使用
文档
1605
具有
内容
1606

图 16