

[19] 中华人民共和国国家知识产权局

[51] *Int. Cl.*⁷

G06F 17/27

G06F 17/21 G06F 17/22



[12] 发明专利申请公开说明书

[21] 申请号 01800133.5

[43] 公开日 2003 年 1 月 22 日

[11] 公开号 CN 1392986A

[22] 申请日 2001. 1. 8 [21] 申请号 01800133. 5

[30] 优先权

[32] 2000. 1. 31 [33] US [31] 60/179,330

[86] 国际申请 PCT/US01/00631 2001. 1. 8

[87] 国际公布 WO01/55900 英 2001. 8. 2

[85] 进入国家阶段日期 2001. 9. 28

[71] 申请人 XML 都市公司

地址 美国加利福尼亚州

[72] 发明人 E·S·黄

[74] 专利代理机构 北京纪凯知识产权代理有限公司

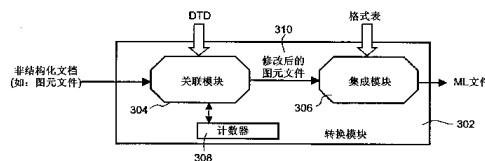
代理人 沙捷 彭益群

权利要求书 6 页 说明书 14 页 附图 17 页

[54] 发明名称 用于产生各种呈现的结构化文档的方法和装置

[57] 摘要

本发明提供了一种使用用户定义的文档类型定义中的标识符来将非结构化的文档转换结构化的文档。所述用户定义的文档类型定义中的标识符,用于关联非结构化文档中选定的对象或组对象,这样选定的对象或组对象的关联信息,可以便于利用用于在各种媒质上呈现的标识语言产生文件。



ISSN 1008-4274

1. 一种用于产生结构化文档的方法，该方法包括：
接收包括文档类型定义（ DTD ）在内的一个定义文件；
与该定义文件一起显示一个图元文件，该图元文件包括若干可显示对象和关于每一可显示对象的各自装饰属性；以及
把定义文件中的至少一个定义与一个可显示对象相关联。
2. 如权利要求 1所述的方法，还包括：
产生包括可显示对象的一个修改的图元文件，每一个可显示对象与定义文件中的至少一个定义相关。
3. 如权利要求 2所述方法，还包括根据一组映射规则来把修改的图元文件转换为一个标识语言文件。
4. 如权利要求 1所述的方法，其中，定义文件包括文档要素的结构，每一个文档要素相应于图元文件中的可显示对象之一。
5. 如权利要求 4所述的方法，其中，一些文档要素包括另一层次文档要素，每一个次文档要素对应于图元文件中的一个可显示对象。
6. 如权利要求 4所述的方法，其中，至少一些文档要素分别包括若干标识符，每一标识符被分配给至少一些文档要素之一。
7. 如权利要求 6所述的方法，其中，一些标识符是一个或多个数字和字母。
8. 如权利要求 6所述的方法，其中，从由字体类型、颜色名称、大小、格式和效果组成的一个分组中选择一些标识符。
9. 如权利要求 6所述的方法，其中，定义文件中的至少一个定义

的关联包括：

选择可显示对象之一；以及
把一个标识符分配给选择的显示对象。

10. 如权利要求 9所述的方法，其中，一个标识符是一个数字或者一个字母。

11. 如权利要求 10所述的方法，其中，一个标识符是（ i ）字体类型、（ ii ）颜色、（ iii ）大小、（ iv ）格式和（ v ）效果中的一个或多个。

12. 如权利要求 1所述的方法，其中，图元文件是或者产生自一个非结构化文档，其通过一个编辑工具来组成、编辑或者管理。

13. 如权利要求 12所述的方法，其中，一些可显示对象是各自的字符分组。

14. 如权利要求 13所述的方法，其中，对于每一字符分组，装饰属性包括至少位置、字体颜色、字体大小、字体类型、格式、和效果。

15. 一种用于产生结构化文档的方法，该方法包括：

激活包括第一显示和第二显示的一个环境，该第一显示显示一个图元文件而该第二显示显示包括文档类型定义（ DTD ）的一个定义文件，其中，图元文件包括若干可显示对象和关于每一可显示对象的各自的装饰属性，并且其中每一文档类型定义包括一个标识符；

把若干组对象分组，每一组对象包括若干可显示对象；以及
把每一组对象与在一个文档类型定义中的标识符相关联。

16. 如权利要求 15所述的方法，还包括产生一个修改的图元文件，其包括与在一个文档类型定义中的标识符相关的每一个组对象的信息。

17. 如权利要求 16所述方法, 还包括根据一组映射规则来把修改的图元文件转换为一个标识语言文件。

18. 如权利要求 17所述的方法, 其中, 标识语言文件适于在一个所选择的媒质上呈现。

19. 如权利要求 18所述的方法, 其中, 所选择的媒质是互联网上的一种 web呈现。

20. 如权利要求 18所述的方法, 其中, 标识语言文件是以从如下分组中选择的一种标识语言, 该分组由超级文本标识语言 (HTML)、简明的超级文本标识语言 (cHTML)、可扩展标识语言 (XML)、标准通用标识语言 (SGML)或者无线标识语言 (WML)组成。

21. 如权利要求 15所述的方法, 其中, 对于每一字符分组, 装饰属性包括至少位置、字体类型、颜色、大小、格式、和效果。

22. 如权利要求 21所述的方法, 其中, 一些可显示对象是各自的字符分组。

23. 如权利要求 22所述的方法, 其中, 一些标识符是一个或多个数字和字母。

24. 如权利要求 23所述的方法, 其中, 标识符是 (i)字体类型、(ii)颜色、(iii)大小、(iv)格式和 (v)效果中的一个或多个。

25. 一种包括由处理器执行的指令的机器可读媒质, 该指令, 当被处理器执行时, 使处理器产生结构化文档, 该机器可读媒质包括: 程序代码, 用于接收包括文档类型定义 (DTD)在内的一个定义

文件；

程序代码，用于与该定义文件一起显示一个图元文件，该图元文件包括若干可显示对象和关于每一可显示对象的各自装饰属性；以及

程序代码，用于把定义文件中的至少一个定义与一个可显示对象相关联。

26. 如权利要求 25所述的机器可读媒质，还包括：程序代码，用于产生包括可显示对象的一个修改的图元文件，其每一个可显示对象与定义文件中的至少一个定义相关。

27. 如权利要求 25所述的机器可读媒质，还包括：程序代码，用于根据一组映射规则来把修改的图元文件转换为一个标识语言文件。

28. 如权利要求 25所述的机器可读媒质，其中，定义文件包括文档要素的结构，每一个文档要素相应于图元文件中的可显示对象之一。

29. 如权利要求 28所述的机器可读媒质，其中，一些文档要素包括另一层次文档要素，每一个次文档要素对应于图元文件中的一个可显示对象。

30. 如权利要求 28所述的机器可读媒质，其中，至少一些文档要素分别包括若干标识符，每一标识符被分配给至少一些文档要素之一。

31. 如权利要求 30所述的机器可读媒质，其中，一些标识符是数字或者字母之一。

32. 如权利要求 30所述的机器可读媒质，其中，从由字体类型、颜色名称、大小、格式和效果组成的一个分组中选择一些标识符。

33. 如权利要求 30所述的机器可读媒质，其中，定义文件中的至少一个定义的关联包括：

程序代码，用于选择可显示对象之一；以及
程序代码，用于把一个标识符分配给选择的显示对象。

34. 如权利要求 33所述的机器可读媒质，其中，一个标识符是一个或多个数字以及字母。

35. 如权利要求 34所述的机器可读媒质，其中，一个标识符是 (i) 字体类型、 (ii) 颜色、 (iii) 大小、 (iv) 格式和 (v) 效果中的一个或多个。

36. 如权利要求 25所述的机器可读媒质，其中，图元文件是或者产生自一个非结构化文档，其通过一个编辑工具来组成、编辑或者管理。

37. 如权利要求 36所述的机器可读媒质，其中，一些可显示对象是各自的字符分组。

38. 如权利要求 37所述的机器可读媒质，其中，对于每一字符分组，装饰属性包括至少位置、字体颜色、字体大小、字体类型、格式、和效果。

39. 一种包括由处理器执行的指令的机器可读媒质，该指令，当被处理器执行时，使处理器产生结构化文档，该机器可读媒质包括：

程序代码，用于激活包括第一显示和第二显示的一个环境，该第一显示显示一个图元文件而该第二显示显示包括文档类型定义 (DTD) 的一个定义文件，其中，图元文件包括若干可显示对象和关于每一可显示对象的各自的装饰属性，并且其中每一文档类型定义包括一个标识符；

程序代码，用于把若干组对象分组，每一组对象包括若干可显示对象；以及

程序代码，用于把每一组对象与在一个文档类型定义中的标识符

相关联。

40. 如权利要求 39所述的机器可读媒质, 还包括: 程序代码, 用于产生一个修改的图元文件, 其包括与在一个文档类型定义中的标识符相关的每一个组对象的信息。

41. 如权利要求 40所述的机器可读媒质, 还包括: 程序代码, 用于根据一组映射规则来把修改的图元文件转换为一个标识语言文件。

42. 如权利要求 39 所述的方法, 其中, 对于每一字符分组, 装饰属性包括至少位置、字体类型、颜色、大小、格式、和效果, 并且其中一些可显示对象是各自的字符分组。

用于产生各种呈现的结构化文档的方法和装置

相关申请的交叉引用

本申请要求 1/31/2000 申请的标题为 “ Method and Apparatus for Generating Structured Documents with User-defined Document Type Definitions Using Structure-based Font Information ”（使用基于结构的字体信息用于产生具有自定义文档类型定义的结构化文档的方法和装置）的临时申请 No. 60/179, 330 的权利，其因此为所有目的被引用结合。

发明领域

本发明通常涉及文件处理和电子出版系统领域，更特别涉及一种用于产生具有自定义文档类型定义的结构化文档的方法和装置。本发明还涉及一种机制，其提供给用户以便使用该方法和装置来为各种呈现转换非结构化文档，其中，非结构化文档被定义为通过一个编辑应用程序（例如文字处理）组成的、编辑或管理的文档。

发明背景

互联网是世界上一个快速发展的互连计算机的通信网络。同时，这些数百万的连接计算机形成一个巨大的超级链接信息的仓库，从任何地方在任何时候通过任意的连接计算机可以很容易访问之。每年随着数百万的 web 页面产生并增加到这个巨大的仓库中，存在一个极大的需要来快速并容易地把诸如呈现、数据表或手册之类的文档转换为可由互联网上的另外一个应用程序或计算机呈现或访问的一种格式。

所熟知的是，可呈现到 web 浏览应用程序（例如一个浏览器）上的一种较好的格式是以标识语言的形式，比如超级文本标识语言（HTML）、可扩展标识语言（XML）、标准通用标识语言（SGML）或者无线标识语言（WML）之类的。为了 web 浏览应用程序而如此组成、

编辑或管理的文件或文档通常被称为结构化文件或文档。在所有结构化文档的好处当中，提供自定义文档类型定义（DTD）或文档概要定义的能力开启了一种新的信息交换或存储的范例。可是，挑战是如何产生具有任意自定义 DTD 的结构化文档。

一个具有特定 DTD 的非结构化文档可以从一个非结构化文档中产生或者从具有其他 DTD 类型的结构化文档中转换而来。这里有若干用于产生结构化文档的编辑器。典型的编辑器包括 Adobe FrameMaker、Arbortext Epic 和 SoftQuad XMetal。这些编辑器通常提供一个结构视图以及一个文字处理视图，在此，文字处理视图类似传统的非结构化文档的文字处理环境同时结构视图包含定义在某个 DTD 中的数据要素的文件结构。为了从这些编辑器中的刻线（scratch）中产生一个结构化文档，用户通常需要在文字处理视图中产生一个非结构化文档。随着一个期望的 DTD 载入，用户根据定义在 DTD 中的文档要素来在结构视图中来构造一棵文件结构树。通常，用户忙于通过把数据要素从产生的文档中拷贝与粘贴或者拖与落入文件结构树的过程中。

为了在这些编辑器中把具有一个 DTD 的结构化文档转换为另外一个 DTD，人们需要加载在结构化文档中，以便把文档要素的标记和属性从一个 DTD 修改为另一个，并以便拖曳该数据要素或者在新的 DTD 中解析与再定义的文档要素相关的新建数据要素。

在上述的过程中，在数据要素和文档要素之间的关联是一个关键性的和努力性的数据处理，用于创建一个非结构化文档或结构化文档或者将其转换为具有特定 DTD 的一个结构化文档。已经建议了若干方法来将数据要素和文档要素关联以便简化结构化文档的产生。作为例子，一种关键字提取方法从非结构化文档中提取表示文件结构的关键字并且把关键字 / 正文对使用作为文档要素和数据要素之间的关联。一种坐标方法通过对坐标文件的坐标进行分类来把数据要素与文档要素中的标识语言标记关联。一种逻辑结构方法通过匹配预确定模式分析文件结构并根据分析的文档要素来解析数据要素。尽管如此，上面没有一种方法已经考虑使用标识符（例如字体信息）来把数据要素和文档要素关联起来。因此，需要一种一般的方法来使用自定义文档类型定义中的标志符信息以便把用于产生结构化文档的数据要素和文档

元件关联。

另外，典型的编辑器所需要的程序某种程度上是冗长而繁重的，并且天性是高成本的。经常，有许多文档要转换的一个企业不得不外购该过程——由于与转换过程相关的低效率和缓慢性。在另一端，服务提供者处理的转换过程难以被量化，因为它主要涉及取决于文档复杂性的人工的和重复的过程。因此，另一个需要是一种机制，以可确定费用的方式用于对各种呈现的非结构化文档到结构化文档的转换进行量化。

发明内容

考虑到上述问题已构成了本发明并且本发明需要并且具有通过互联网的呈现的特定应用。本发明的特征之一是使用 DTD文件中的标识符以便关联选定对象或组对象以使选定对象或组对象的关联信息可以促进以适于各种媒质上的呈现的标识语言的形式产生文件。

本发明可被实现为一种方法、一种系统、一种产品或另一种实际的形式。根据一个实施例，本发明是一种方法。该方法接收一个定义文件并和该定义文件一起显示一个图元文件，该定义文件包括文档类型定义（DTD），该图元文件包括若干可显示的对象和每一可显示对象的各个装饰（decoration）属性。该定义文件包括一个文档要素的结构，每一个相应于图元文件中的可显示对象之一。某些文档要素包括许多标识符，每一标识符被分配给一个文档要素。在一个实施例中，标识符是数字和/或字母。在另外一个实施例中，标识符是字体名称、颜色名称、大小、字体类型、颜色、格式、各种效果或其它符号中的一个或多个。该方法把至少一个标识符与一个可显示对象相关联。

根据另外一个实施，本发明被实现为一种用于提供文档转换过程的方法，该方法包括激活一个具有记数系统的计数器，把非结构化文档转换为图元文件，其中，图元文件包括若干可显示的对象和每一可显示对象的各个装饰（decoration）属性，接收包括与非结构化文档有关的文档类型定义（DTD）在内的一个定义文件；产生一个修改的图元文件，其包括与定义文件中的一个定义相关的至少一个可显示对象的关联信息；以及只要修改的图元文件要被存储就使计数器加1。

在下列说明和产生的附图中说明的实施例中，本发明的运用中获得的上述的目的和优点。

附图说明

关于下列说明、附加的权利要求和附图，本发明的这些和其他特点、方面和优点将变得更好理解，附图中：

图 1A表示根据本发明优选实施例可以实现本发明的一种基本系统结构；

图 1B表示一种系统的内部结构，其中可以实现本发明从而达到在被本发明中期望的结果；

图 2A阐明了一个非结构化文档的例子，其可以通过一个编辑工具被组成、编辑或管理；

图 2B是文档类型定义（DTD）的例子；

图 2C表示以图 2B中的文档类型定义（DTD）为基础如图 2A所示的非结构化文档的结构化文档；

图 3A说明了如本发明一个实施例所述的功能图；

图 3B表示如本发明一个实施例所述的实现转换模块的一种可视环境；

图 3C表示关于如图 3B所示的图元文件中可显示对象以 XML格式设计的格式表（style sheet）的例子；

图 3D表示来自 dongle（道尔）的一则典型消息；

图 3E表示如本发明一个实施例所述使用包括转换模块的实施的在一个产品的流程图；

图 4说明数据处理装置的框图，该数据处理装置使用基于结构的字体信息来把未结构化或结构化文档输入、编辑并转换到具有自定义 DTD的结构化文档；

图 5是在图 2B中的 DTD中定义的文档要素的关联表和字体属性；

图 6表示图 1中的非结构化文档的一个编辑结果，在此，基于图 5中的关联表，每个解析的数据要素已经被分配了字体属性；

图 7表示一个转换过程，它把图 6中的解析的数据要素转换为具有图 2B中典型 DTD的期望的结构化文档；

图 8是一个中间的结构化文档，它包含解析的数据要素，该解析的数据要素具有与字体文档要素列表相关的分配字体 ID；

图 9表示一个转换过程，它把图 6中的中间的结构化文档转换为具有图 2B中 DTD的期望的结构化文档；以及

图 10表示使用转换的可扩展格式语言（XSLT）的在图 9中的转换过程的一种实施。

最佳实施方式

在本发明的下列详细说明中，阐明了很多特殊的细节以便提供对本发明的全面理解。可是，对本领域的普通技术人员来说很明显，没有这些特殊的细节也可以实践本发明。在其他实例中，为了避免使本发明的各方面不必要地变得不明朗，没有详细描述熟知的方法、程序、组件以及整机线路。详细的说明以程序、逻辑块、处理的形式和直接或间接地类似耦合到网络上的数据处理设备操作的其它符号表示而大量存在。这些处理描述和表述是由本领域有经验的或者精通的技术人员使用的方法以便把他们工作的实质最有效地传送给本领域的其他技术人员。

在此，关于“一个实施例”或“一实施例”表示有关于该实施例描述的特定的特点、结构或特征可以被包括在本发明的至少一个实施例中。在说明书中的各个位置的短语“在一个实施例中”出现不一定是指同一实施例，也不是互斥的其他实施例的单独或可选实施例。此外，表示本发明的一个或多个实施例的流程图或图表中的模块顺序不是固有地表示任何特定的顺序也不是意味着本发明的任何限制。

现在参见附图，其中，整个若干视图中，类似数字是指类似的部分。图 1A表示根据本发明优选实施例可以实现本发明的一种基本系统结构。利用在计算机 100上执行的一个编辑工具可能创建诸如产品说明、功能列表和价格预定之类的非结构化文档。通过一个编辑工具创建的文件或文档被称为非结构化文档。典型的编辑工具可以包括 Microsoft Office（例如，Microsoft Word、Microsoft PowerPoint 和 Microsoft Excel），Adobe FrameMaker和 Adobe Photoshop。非结构化文档可以被上载到可担任中央仓库的计算设备 102上。计算设

备 102可以是来自 Sun Microsystems (www.sun.com) 的一个服务站或者是用实现本发明的一个实施例的遵照和链接版本装载的一台式计算机。

在一个设置中, 计算机 100和计算设备 102是不可分离的并且执行文档转换过程和产生可以用诸如 XML或 HTML之类的标识语言格式最终表示的结构化文档。一个应用中, 以 XML的形式表示的结构化文档被转换成 HTML格式并通过一专用网 110到一个服务服务器 104变成可利用的, 该服务服务器 104主管通常称为一个 www (环球信息网) 站点。

在一种情况中, 用户使用一台式计算机 106, 该台式计算机 106运行了浏览应用程序并且连接到数据网络 108上以便访问服务服务器 104上的文件。在计算机 102中通过结构化文档表示的这些文件可以表示通过一个编辑工具最初组成的最近的产品信息。

正如在下面将解释的, 本发明不局限于互联网应用程序。它可在个人计算机中实践, 其中, 用户经常以文字处理格式产生文档, 比如 FrameMaker或 Microsoft Word之类的。本发明可被应用来把文档转换为一种标识表示, 而不管实际的文字处理格式。

图 1B表示一种系统 118的内部结构, 其中可以实现本发明从而达到在被本发明中期望的结果。系统 118可以与客户机设备 (例如计算机 100、 102或 106)或者服务器设备 (例如服务器 104)通信。如图 1B所示, 系统 108包括与数据总线 120接口的中央处理器 (CPU)和设备接口 124。为了同步操作 CPU 122执行某些指令来管理所有的设备和连接到数据总线 120上的接口并且设备接口 124可连接到诸如计算机 102之类的一个外部设备上, 因此由此而来的文档通过数据总线 120被接收到存储器或者储存器中。显示接口 126、网络接口 128、打印机接口 130和软盘驱动器接口 138也与数据总线 120接口。通常, 本发明一个实施例的编译的和链接的版本通过软盘驱动器接口 138、网络接口 128、设备接口 124或连接到数据总线 120上的其他接口而装载到储存器 136中。

诸如随机访问存储器 (RAM)之类的主存储器 132也与数据总线 120接口以便向 CPU 122提供指令并访问存储器存储 136的数据和其它

指令。特别地，当执行储存的应用程序（例如本发明的遵照和链接的版本）时，使 CPU 122操作数据来达到本发明预期的结果。提供 ROM（只读存储器）134用于存储诸如键盘 140、显示 126和鼠标 142操作的基础输入输出操作系统（BIOS）之类的常量指令序列，如果有的话。

图 2A阐明了一个非结构化文档 200的例子，其可以通过一个编辑工具被组成、编辑或管理。在一个非结构化文档中，通常以序列的形式表示数据，该序列通常遵循一种读取顺序（例如自上而下和从左到右）。此序列可被解析为数据要素段，在此，每个数据要素 102被分配装饰属性（decoration attributes）或诸如位置、字体颜色、字体大小、字体类型、格式和各种效果等等之类的信息。当包含数据要素的一个文件由编辑工具显示在显示屏上时，装饰信息基本上是为了适当的布置和呈现目的。

按照一个实施例，一个非结构化文档被打印到包含装饰信息的一种图元文件格式。图元文件格式的一个例子是通常使用的可移植数据格式（PDF）。图元文件格式的一个优点是它独立于编辑工具以及或许独立于计算机的独立性以使可以在许多不同的环境中同样打开或读取图元文件格式。

诸如 SGML和 XML之类的一个结构化文档以文档类型定义（DTD）开始。图 2B说明了“制法类型”（recipe - type）文档的 DTD 208的例子，其中文档要被分解成为文档要素的结构。一个特定的文档要素 210可以包含其它的文档要素和属性。文档要素 212的另外一个例子只包含解析的字符数据。

图 2C表示相对于图 2B中的 DTD 208从图 2A中的非结构化文档 200相应转换而来的结构化文档 220。如此图所示，在非结构化文档中的数据序列被解析成为与结构化文件的以 DTD形式定义的文档要素相关的数据要素。

不同于非结构化文档，结构化文档可以通过文档要素容易地访问某些信息。一个结构化文档的呈现通常以分开的格式表的形式来定义，例如，写入级联格式表（CSS）或者可扩展格式语言用于对对象进行格式化（XSL - FO），其通译每个文档要素的布置。这个特点允许对于不同的媒质通过不同的格式表让一个结构化文档呈现于不同的布置

中。通常，诸如非结构化文档中的字体信息之类的装饰信息或者格式化属性，除非在 DTD 中定义为文档要素的属性，否则在一个非结构化文档被转换成一个相应的结构化文档之后被抛弃。格式化信息的进一步修改通常将不影响已转换的结构化文档。

图 3A 说明了如本发明一个实施例所述的功能图 300。转换模块 302 包括关联模块 304 和集成模块 306。关联模块 304 接收一个非结构化文档，最好以图元文件的格式。同时，关联模块 304 还接收被称为定义文件的一个文件，包括预定义的 DTD。通常，根据非结构化文档的特性或用途定义 DTD。例如，非结构化文档是在收到的一个类别中，例如图 2A 中的文档 200，按照“收到类型”（receipt - type）文档设计如图 2B 所示的定义文件中的 DTD。

为了更进一步地理解关联模块 304，图 3B 表示如本发明一个实施例所述的实现转换模块 302 的环境 320。环境 320 包括为用户提供的两个显示 322 和 324 以便执行把非结构化文档转换为标识语言形式的文件（称为标识语言文件）。显示 322 用于显示该非结构化文档。在一个优选实施例中，装载该非结构化文档的图元文件版本用于显示。一个图元文件，涉及该非结构化文档或者它的打印版本，通常包含许多可显示的对象。每个对象是一类或一组字符或单词或者一种图示。如图显示 322 中所示，每个字或一个孤立的数字是一个可显示对象，其在图元文件中固有地被转入。换言之，每个对象通过包括（但不限制于此）对象的类型、大小、颜色和位置在内的若干属性或装饰信息来定义，如此以致它可以是正确地“打印”。若干对象可以依据它们的含义或用途由用户手动地归类。例如，组对象 326 包括三个字符类型对象“Green”（绿）、“Chili”（红辣椒）和“Salsa”（酱汁）。自然，这三个字符类型对象形成一个名称作为一个组对象 326。至于显示 322 中其余的被显示图元文件也可执行对象分组。

显示 324 用于显示为显示 322 中的图元文件准备的一个定义文件。为了易于关联模块 304 的操作，定义文件被图形地显示为“DTD Pool”328。例如，图 2B 中的 DTD 208 的图解表示被使用于显示 324 中以便说明文档要素之中的层次关系。

因此，对于最终将图元文件转换为 XML 文件的一个实施例，从“

DTD Pool " 328中产生一个辅助的 XML树 330。辅助的 XML树 330还表示文档要素中的层次关系。此外，把每一文档要素分配给一个标识符，其可以包括（但是不限制于此）数字、名称、字体、类型名称或者颜色。在一个实施例中，标志符在每一文档要素的“数据”中。为了把组对象 326与文档要素“标题” 332关联，在选择组对象 326被选择之后激活“数据” 334。本发明中的特征之一是一个基础的关联，其把组对象 326与“数据” 334中的标志符相关联。特别是在一个实施例中，如果“数据” 334中的标志符是颜色，“green”（绿色），则组对象 326以绿色被高亮显示以便表示这个组对象已经与 DTD相关。如果标志符是字体，“Anal”（解析），则组对象 326以格式解析的形式被高亮显示以便表示这个组对象已经与 DTD相关。

同样地，在“成分”之下，组对象 340可以与数据 342中的标识符相关，在“成分”的“数量”之下，组对象 344可以与数据 346中的标志符相关等等。结果，显示 322中的图元文件已经被分段并且在其中的可显示对象分别被归类然后通过一个标识符把每一组对象与加载的 DTD中的文档要素相关。显示 322现在具有一个修改的图元文件 310，在下面将说明它的一个例子。

现在回来参见图 3A，把修改的图元文件 310输入给另外接收格式表的一个集成模块 306。按照来自图元文件中的对象将显示在其上的介质，通常把一个格式表配置来包括映射规则。一个典型的介质是可通过浏览器（例如来自微软的 Internet Explore）访问的一种文件的 web呈现。因此，该文件是诸如 HTML或 XML之类的标识语言的，称为一种标识语言文件。

图 3C表示这样的关于图元文件中可显示对象以 XML格式设计的格式表（style sheet）的例子。通常，一个格式表分别被设计为每一对象的位置、颜色或者大小以便对于一个特定的介质可以达到一个适当的和注意的呈现。图 3C中的例子是设计用于显示“收到类型”文件并当装载时使修改的图元文件产生一个适当的 XML。换言之，集成模块 306根据格式表从修改的图元文件中产生 XML文件。在此给出说明，应当指出，不必把格式表输入到集成模块 306。在一个实施中，利用 DTD文件可以载入映射规则以使集成模块 306根据装载的映射规

则执行从修改的图元文件到一个标识语言文件的映射。

按照一个实施例，以软件的形式实现转换模块 302并且可将其作为一种应用程序发送给用户或服务提供者。不言而喻，从非结构化文档到标识语言文件的转换过程难以用可确定成本的方式来量化。计数器 308包括在转换模块 302中。在一个实施例中，计数器 308被配置来对要转换的图元文件中的页数进行计数。每次，在一个显示（即，一个页面显示）中的全部对象与一个 DTD文件中的文档要素相关并且被另存为一个相应的修改的图元文件，计数器 308增加。图 3D表示保持在一个 dongle中的计数结果的例子。一个 dongle（明确的 DONG - uh1）是一个机制，用于确保只有特许用户可以拷贝或使用一个特定的软件应用程序，特别是非常昂贵的程序。一个 dongle的普通实现包括一个硬件密钥，其插入在计算机上的并行或串行端口中并且在继续运行以前一个软件应用程序访问它用于检验；以类似的方式访问的特殊密钥软盘；以及登记数，其在工厂或在一个系统建立期间被装载入某种只读存储器的形式。

当 dongle需要被重置时，可以以可确定成本的方式估计转换过程。按照一个实施例，包括转换模块 302的实现在内的一种产品的所有者可以免费地或者以非常低的成本向用户分布该产品。典型情况下，用户需要从通过各种编辑工具组成、编辑或管理的非结构化文档中产生 web页面的容量。对于用户以这样的方式接收产品的一个好处是在使用它以前不必为了获得该产品而付出很多资金。用户可以为产品的使用付款。因此，使用具有转换模块 302的一个用途是管理它的使用。结果，该产品的所有者可以通过控制包含使用信息的 dongle来控制该产品的使用。

图 3E表示如本发明一个实施例所述使用包括转换模块 302的实现在内的一个产品的流程图 370。有时，产品被一个用户或者一个业务所租借。另一时间，该产品被一个服务提供者使用，该服务提供者向需要把非结构化文档转换为不同媒质呈现（在一个web站点上的呈现）的结构化文档的业务提供服务。

过程 370在 372处从编辑文档中产生图元文件开始。通常，编辑文档可能已经准备使用一个或多个编辑工具。如上所述，最好优先地

从编辑文档中获得图元文件以使不必为不同编辑工具的每一个分别地配置转换模块 302。可是，应当指出，图元文件的优先选择不是对本发明的一个固有限制而是使产品或转换模块 302更有效地工作。本领域的技术人员理解可以配置一个转换接口或一个打印驱动器来适应任何类型的编辑文档或者产生图元文件。

一旦获得图元文件，则立刻把它们装载到一个可视的环境中，其中可以分别显示图元文件。图 3B的环境 320可适用的以使可以逐一地装载每一图元文件的页面用于显示。

在过程 370允许用户更进一步继续以前，启动一个授权过程 378来确保用户正在操作一个授权的产品。如上所述，一个典型的授权方法是通过由提供 /拥有 /控制该产品的企业或经销商预先设置的 dongle。如果授权过程 378表示过程 370没有被授权，则通常向用户显示关于在哪里该产品可以被授权的一个显示。在设置授权 376中的程序之一涉及转换或存储的页数的允许数量的购买。

按照一个实施例，一个 dongle被用于耦合到执行过程 370的一台计算机上。dongle包括第一和第二号码。第一号码是一个开始号码，例如，“10”，而第二号码是一个限制号码，例如，“1000”，其是指有 1000页的转换文档可以被过程 370处理和保存。

一旦过程 370被允许继续到 380，立刻允许用户或许按照它们的含义或者它们的用途并考虑到装载的 DTD文件和在附近的显示来把若干可显示对象分别归类到组对象。在 382处，组对象可以分别与 DTD中的定义相关。至少某种定义具有许多标识符，优先地，每个标识符与一个定义相关或者指定给一个定义。

如上所述，在选定对象和定义之间动关联要被保存在修改的图元文件中。在过程 370允许如此保存之前，在 386处要检查一个计数器。在一个实施例中，比较 dongle中的第一和第二号码。当第一号码基本上接近于第二号码时，例如两个号码是相同的，则过程 370将询问允许使用的一个补充。通常，用户不得不使 dongle被现在可以根据 dongle中的信息征收费用的企业或者经销商进行重置或者重新架构。在 384处，数字已经被重置现在允许过程 370继续。

在 386处，可以操作一个保存步骤。依靠一个精确的实施，修改

的图元文件或一个标识语言文件可以被保存在存储空间中。根据一个预定义媒质呈现的一个格式表，从修改的图元文件中形成标识语言文件。在 388处，计数器被增加。

在图 3E中，应当指出在 386处已检查计数器，特别是在 384后。实际上，对本领域的技术人员来说已经很明显，实际上可以在沿着过程 370的任何地方检查或者查阅计数器。在此处使用一个计数器的一个目的是使企业易于控制和确定过程 370的应用以便可以确定成本并可以收费。

图 4表示如本发明一个实施例所述的数据处理模块 404的原理框图 400。包括在集成模块 306中的数据处理模块 404包括一个输入模块 406、一个编辑模块 410和一个转换或过滤模块 414。数据处理模块 404执行的一个功能是把非结构化文档或者具有不同 DTD的结构化文件转换成为具有预定义的或特定的 DTD的相应结构化文档。

输入模块 406从一个文档数据库 402（其可以相应于图 1A的计算设备 102中的仓库）中加载文档或者输入文档。替代地，输入模块 406可以开始一个新建文档 408。应当指出，加载的或输入的文档，在某些情况下可以是未结构化的（例如，图元文件）或者结构化的并且可以已经包含预先产生的基于结构化的字体信息。

编辑模块 410与输入模块 406联系并为输入资料创建 /编辑基于结构化的字体信息。此模块允许输入文档的数据要素的选择并提供一个编辑环境来改变所选择的数据要素的诸如字体类型、字体格式、字体颜色、字体大小和字体效果之类的字体属性。把输入文档解析成为数据要素以及分配字体属性的办法是以定义在期望的 DTD中的文档要素的一个关联表和相关的字体属性 412为基础的。DTD 412的一个典型的关联表 500在图 5中给出，它包含文档要素 502、要素属性 504、字体类型 506、字体格式 508、字体颜色 510、字体大小 512和字体效果 514等各字段。

图 6表示图 2A的非结构化文档 200的编辑结果 600。每一个解析的数据要素或合并的对象 602、604、606、608、610、612和 614基于图 5中的关联表已经被指定了字体属性并以相关的字体形式分别被显示。在解析期间，这个模块根据输入文档 602的读取顺序来提供

数据要素的序列选择以便编辑它们的字体信息。这个模块还允许数据要素的区域分组以便编辑它们的字体信息。这个模块还可以提供关联表的辅助视图。

转换或过滤模块 414使用基于结构化的字体信息来把加载的文档转换成为具有自定义文档类型定义 (DTD)的结构化文档。转换 416的基于文档要素和字体字体的映射规则被输入或设计在这个模块中。

图 7说明了把编辑文档 602转换成为图 2C的结构化文档 220的映射规则的一个例子 700。特别地,702开始而 718结束“文档”要素,704开始而 714结束“制法”文档,706形成“成分”要素,708形成“过程”要素,710形成“呈现”要素,以及 712形成“起源”要素。在 704、706、708、710和 712中,基于结构化的字体信息用于定位数据要素并且该定位的数据要素被指定作为文档要素的属性或解析的字符数据。这些映射规则可以被实现为(但不限制于此)诸如 Java、JavaScript之类的程序设计语言、用于转换的可扩展格式语言1 (XSLT), C/C++,.. 等等, 或者任何内建的或者可编程的硬件设备。转换文档可以被另存为一个文件文档或者被导出到文件数据库 418中。

除了直接转换成期望的文档之外,转换模块还可以把编辑文档作为包含基于结构化的字体信息的中间的结构化文档而输出。该中间的结构化文档可以被再装载用于另外的编辑或批量转换。

编辑文档 600的中间的结构化文档 808的一个例子在图 8中给出,在此,802包含具有特定的字体属性的“字体”要素而 804包含有关联字体信息的“font_ID”属性的解析数据要素。由于具有相同字体属性的解析数据要素已经依据中间的结构化文档中相同的“font_ID”而被分组,可以基于该分组的字体信息设计转换的映射规则。

图 9说明了把中间的文档 800转换成为结构化文档 220的映射规则的一个例子。特别地,902开始而 918结束“文档”要素,904开始而 914结束“制法”文档,906形成“成分”要素,908形成“过程”要素,910形成“呈现”要素,以及 912形成“起源”要素。在 904、906、908、910和 912中,分组的字体信息用于定位数据要素并且该定位的数据要素被指定作为文档要素的属性或解析的字符数据。

图 10表示使用转换的可扩展格式语言 (XSLT)的在图 9中给出的映射规则实现的一种例子。

上述的发明最好优先地以软件、硬件或者二者组合的形式来实现。本发明的至少一些部分可以被具体化为计算机可读媒质上的计算机可读代码。计算机可读媒质是可以存储此后可以被一个计算设备读出的数据的任何数据存储设备。计算机可读媒质的例子包括只读存储器、随机访问存储器、磁盘驱动器、软磁盘、 CD-ROM、 DVD、磁带、光学数据存储设备、载波。计算机可读媒质还可以通过耦合计算机系统的网络来分布以使用一种分布的形式来储存并执行计算机可读代码。

已经以一定程度的特性足够详细地描述了本发明。本领域的技术人员应理解，只是已经用实例来进行实施例的公开，并且不偏离所要求的本发明的精神和范围，也可凭借在设备和各部分的组合中的许多替换。同时在此讨论的实施例依据格式和配置可以存在包括关于信息单元的呈现的某些限制，本发明具有超过这样的实施例的良好适用性，这可以被本领域的技术人员所估计到。因此，本发明的范围通过附加的权利要求而不是前述实施例的说明来定义。

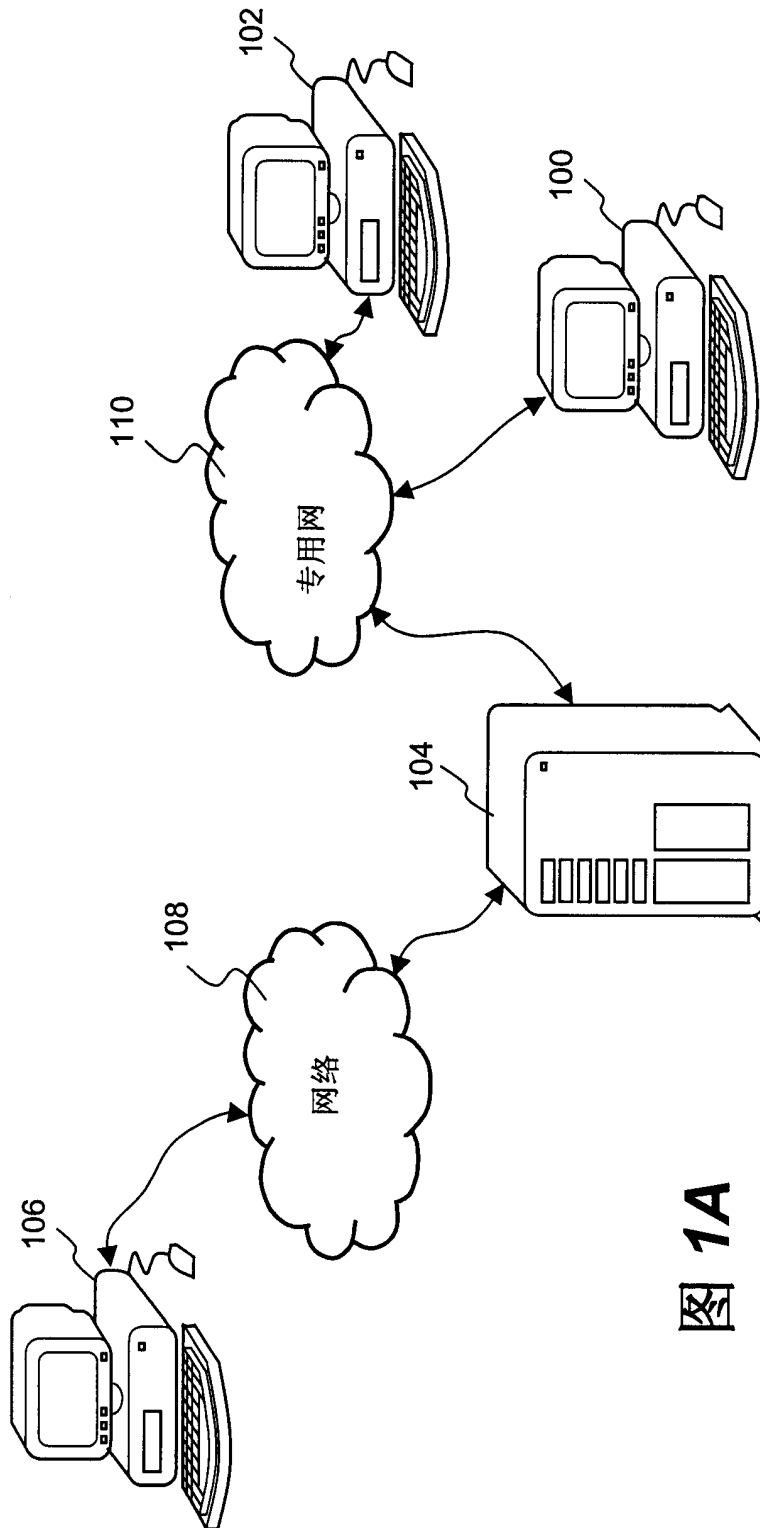


图 1A

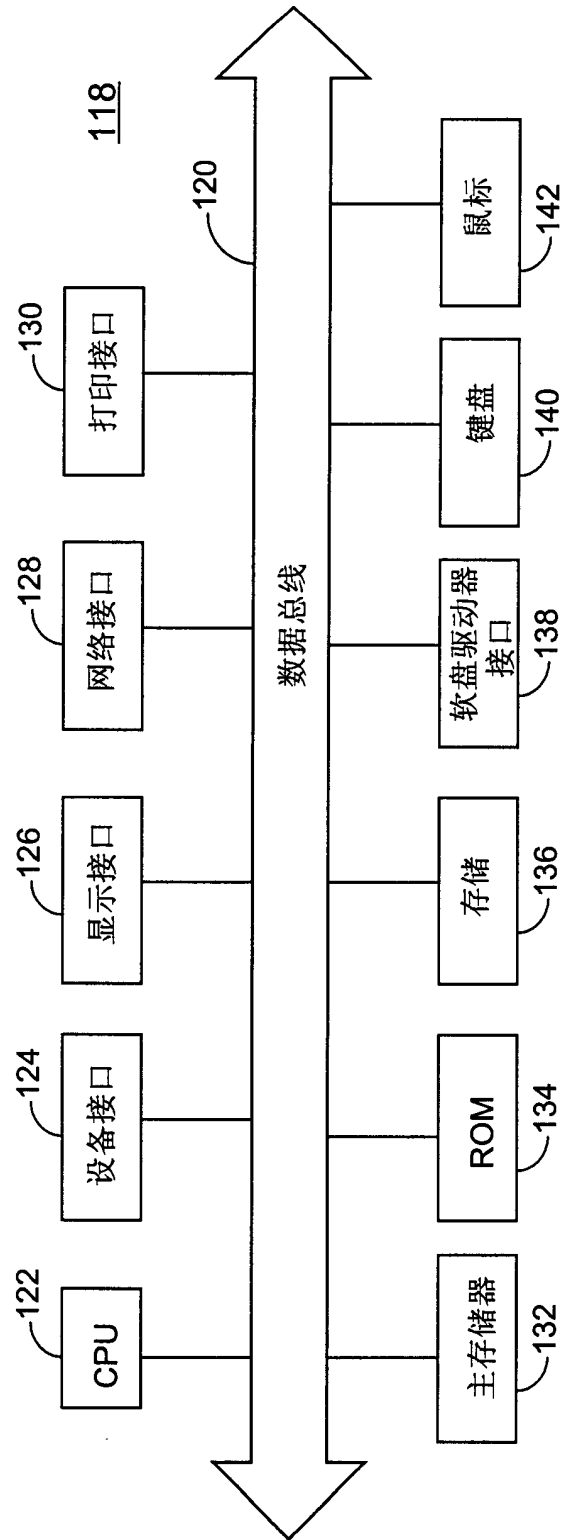


图 1B

200

202

Green Chili Salsa

Chopped Tomatoes	1 16-ounce Can
Chopped Green Chile	3 Tablespoons
Diced Onions	3 Tablespoons
Chopped Garlic	1 Tablespoon
Salt	¼ Teaspoon
Lime Juice	½ Lime
Chopped Cilantro	1 Tablespoon

In a medium bowl, mix together all ingredients. Use as a dip, or over grilled vegetables, or toss with bowtie noodles for an appetizer course.

www.VeggieLife.com November 1997



208

```

<!--..... Document Type Definition (DTD) .....-->
<!-- document element -->
<!ELEMENT document (recipe+)>

<!--..... recipe element .....-->
<!ELEMENT recipe (ingredient+, procedure+, presentation, originate)>
<ATTLIST recipe title CDATA #REQUIRED>

```

```

<!--..... ingredient element .....-->
<!ELEMENT ingredient (#PCDATA)>
<ATTLIST ingredient amount CDATA #REQUIRED>

```

210

```

<!--..... procedure element .....-->
<!ELEMENT procedure (#PCDATA)>
<ATTLIST procedure step CDATA #IMPLIED>

```

```

<!--..... presentation element .....-->
<!ELEMENT presentation (#PCDATA)>

```

212

```

<!--..... originate element .....-->
<!ELEMENT originate EMPTY>
<ATTLIST originate name CDATA #REQUIRED>
<ATTLIST originate website CDATA #IMPLIED>
<ATTLIST originate date CDATA #REQUIRED>
<!--..... END of DTD .....-->

```

图 2B

```
<document>
<recipe title="Green Chile Salsa">
  <ingredient amount="1 16-ounce Can">Chopped Tomatoes</ingredient>
  <ingredient amount="3 Tablespoons">Chopped Green Chile</ingredient>
  <ingredient amount="3 Tablespoons">Diced Onions</ingredient>
  <ingredient amount="1 Tablespoon">Chopped Garlic</ingredient>
  <ingredient amount="1/4 Teaspoon">Salt</ingredient>
  <ingredient amount="1/2 Lime">Lime Juice</ingredient>
  <ingredient amount="1 Tablespoon">Chopped Cilantro</ingredient>
  <procedure>In a medium bowl, mix together all ingredients.</procedure>
  <presentation>Use as dip, or over grilled vegetables, or toss with bowtie noodles for an appetizer course.</presentation>
  <originate name="VeggieLife" website="www.VeggieLife.com" date="November 1997" />
</recipe>
</document>
```

图2C

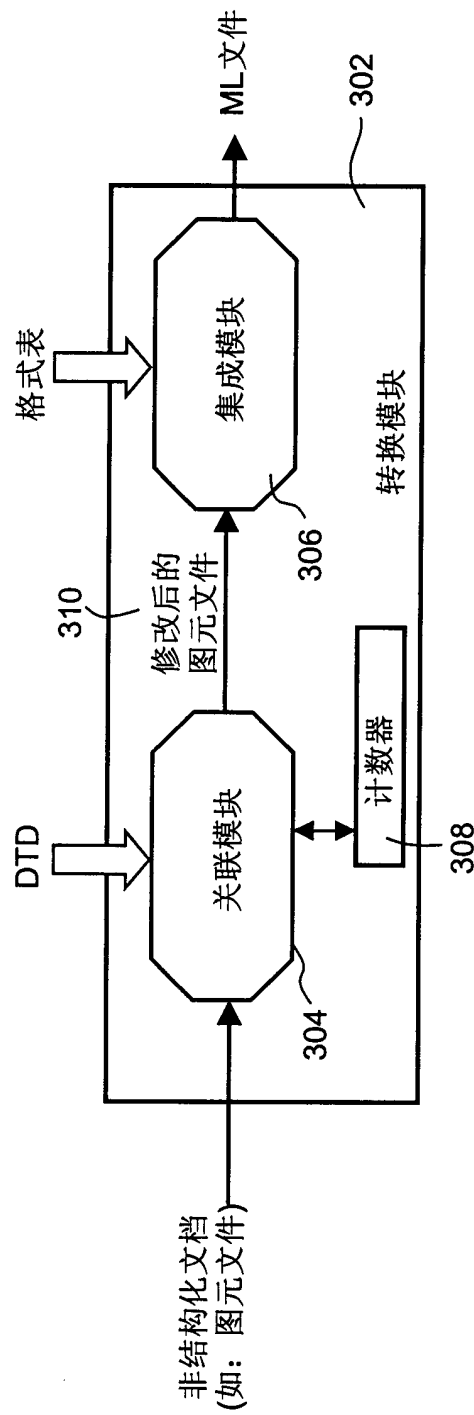
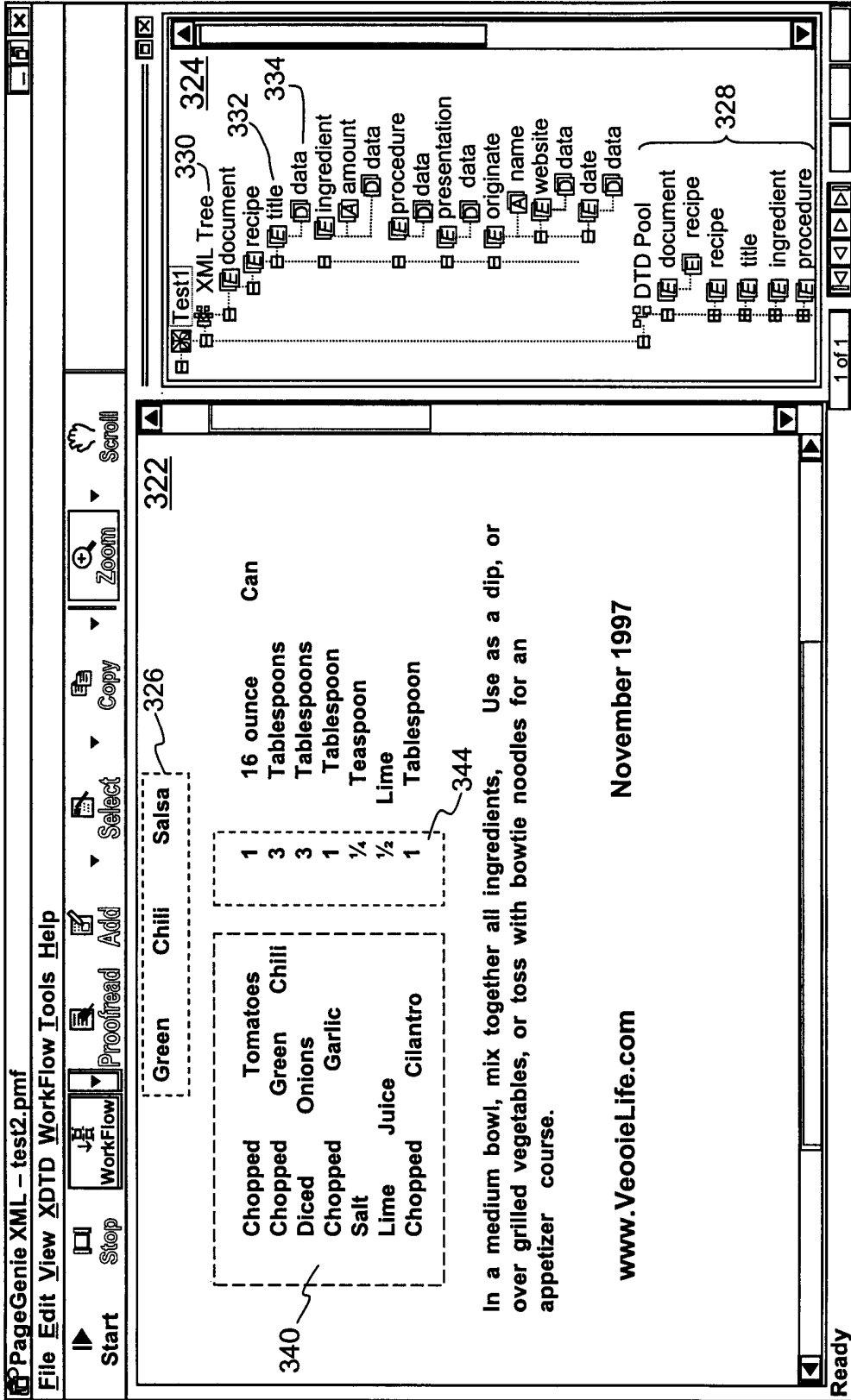


图3A

图 3B

320



```

<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
  <xsl:template match="/" >
    <html>
      <body>
        <xsl:apply-templates />
      </body>
    </html>
  </xsl:template>

  <xsl:template match="document">
    <xsl:apply-templates />
  </xsl:template>

  <xsl:template match="recipe">
    <h2><center><font color="teal">Title: </font><font color="red">
      <xsl:value-of select="title"/>
    </font></center></h2>
    <h3><center><font color="teal">From: </font><font color="gray">
      <xsl:for-each select="originate" >
        <xsl:value-of select="@name" />
      </xsl:for-each>
    </font></center></h3>
    <p><center><table bgcolor="ffffc0" cellspacing="12"><tr><td>
      <xsl:for-each select="image">
        <IMG SRC="recipe.jpg" HEIGHT="200" WIDTH="220"/></xsl:for-each></td><td><table>
          <tr><td><font color="teal" size="5">Ingredient:</font></td></tr>
          <xsl:for-each select="ingredient">
            <tr><td><xsl:value-of select="text()" /></td>
            <td><font color="blue"><xsl:value-of select="amount"/></font></td></tr>
          </xsl:for-each>
        </table></td></tr></table></center></p>

    <p><center><table><tr><td><font color="teal" size="5">Procedure:</font></td></tr>
      <xsl:for-each select="procedure">
        <tr><td><xsl:value-of select="text()" /> </td></tr>
      </xsl:for-each>
      <tr><td></td></tr>
      <tr><td><font color="teal" size="5">Presentation:</font></td></tr>
      <xsl:for-each select="presentation">
        <tr><td><xsl:value-of select="text()" /> </td></tr>
      </xsl:for-each>
    </table></center></p>

    <p><center><table bgcolor="c0ffff" width="80%">
      <xsl:for-each select="originate">
        <tr><td align="left"><xsl:value-of select="website" /></td>
        <td align="right"><xsl:value-of select="date"/></td></tr>
      </xsl:for-each>
    </table></center></p>
  </xsl:template>
</xsl:stylesheet>

```

图3C

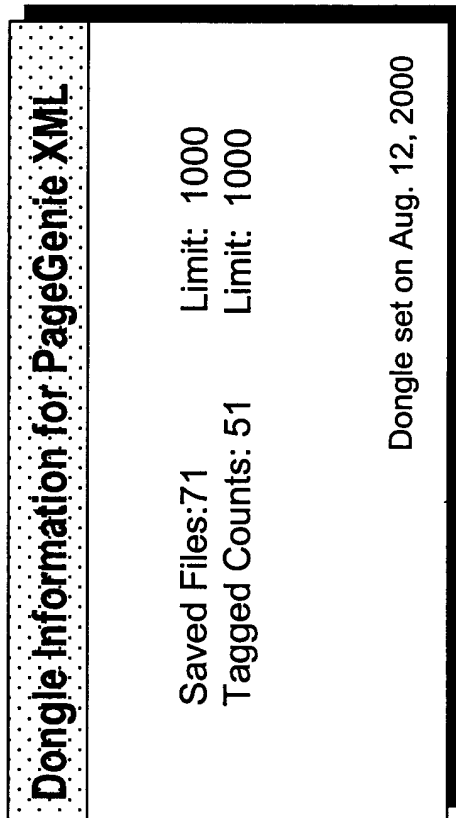


图 3D

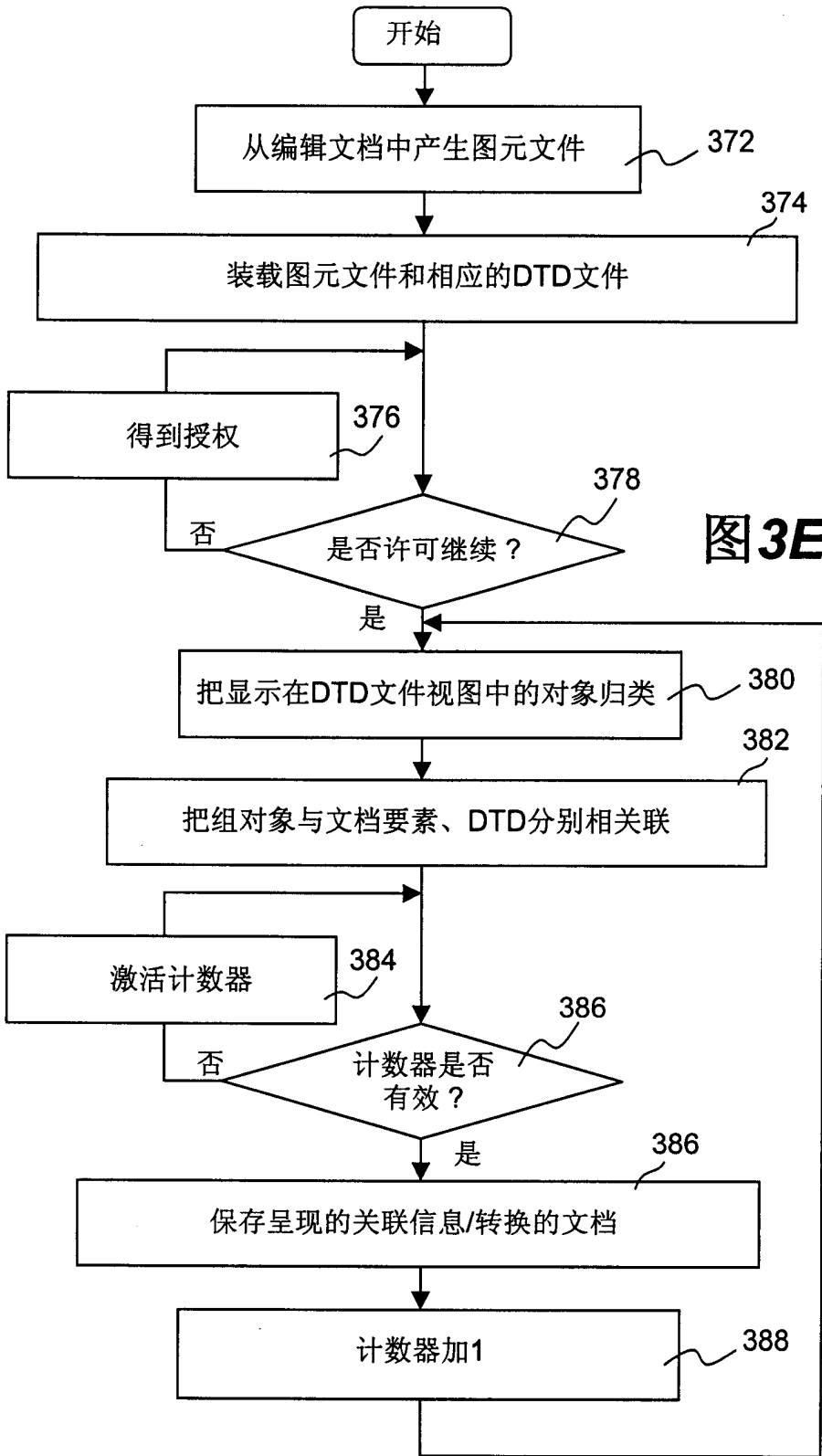
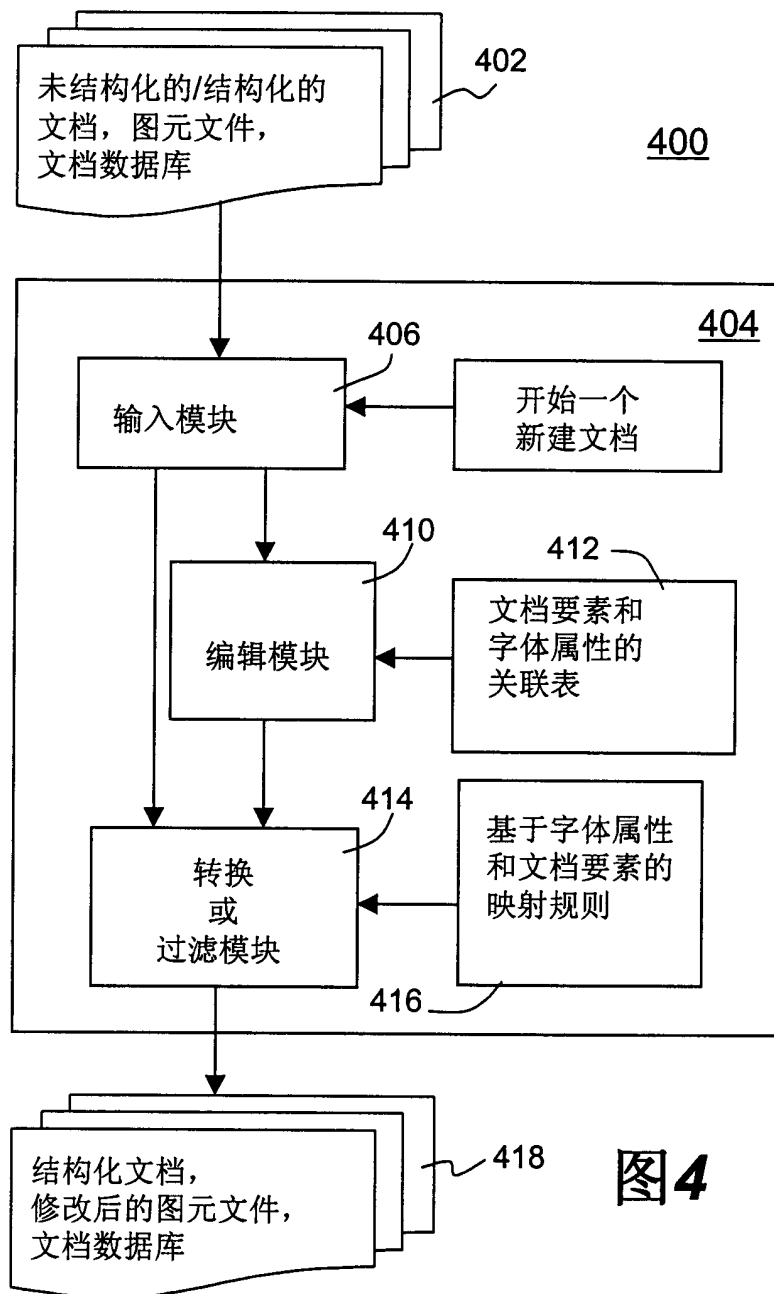


图3E



文件要素	要素属性	字体类型	字体格式	字体颜色	字体大小	字体效果
recipe	title	Courier New	Bold	Black (0x000000)	12	Underline
ingredient		Courier New	Normal	Magenta (0xFF00FF)	10	Normal
ingredient	amount	Courier New	Italic	Red (0x0000FF)	10	Normal
procedure		Arial	Italic	Blue (0xFF0000)	10	Normal
presentation		Arial	Normal	Cyan (0xFFFF00)	10	Normal
originate	name	Times New Roman	Bold	Green (0x00FF00)	10	Normal
originate	website	Times New Roman	Bold + Italic	Yellow (0x00FFFF)	10	Normal
originate	date	Times New Roman	Normal	Gray (0x808080)	10	Normal

图5

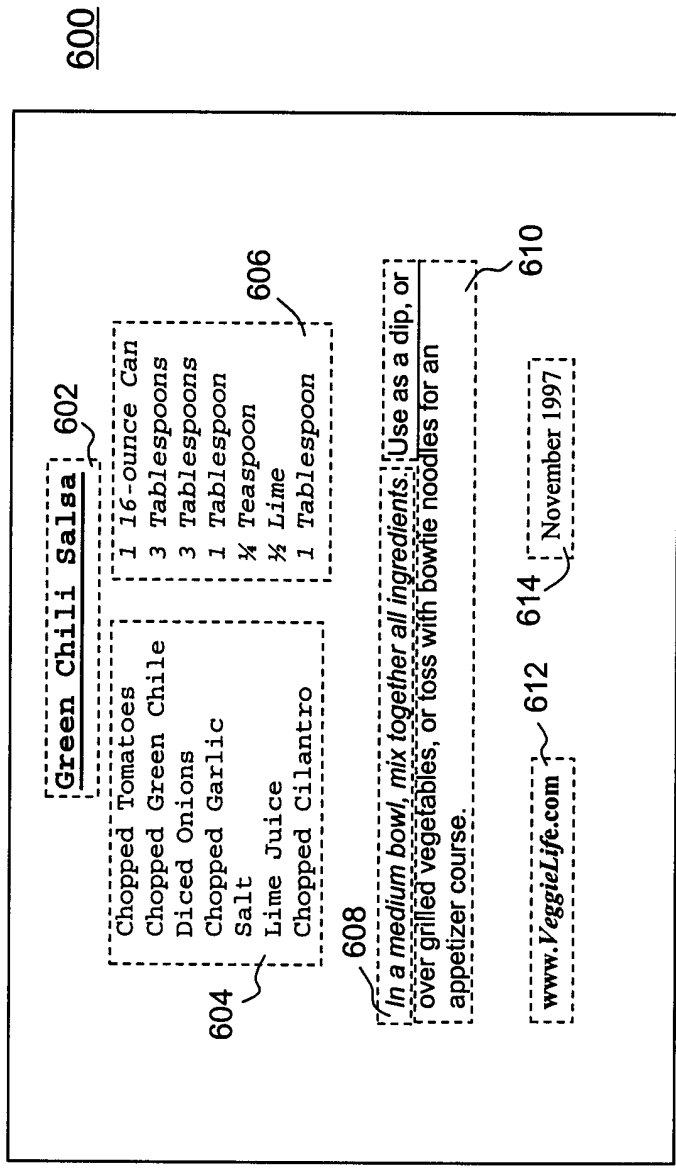
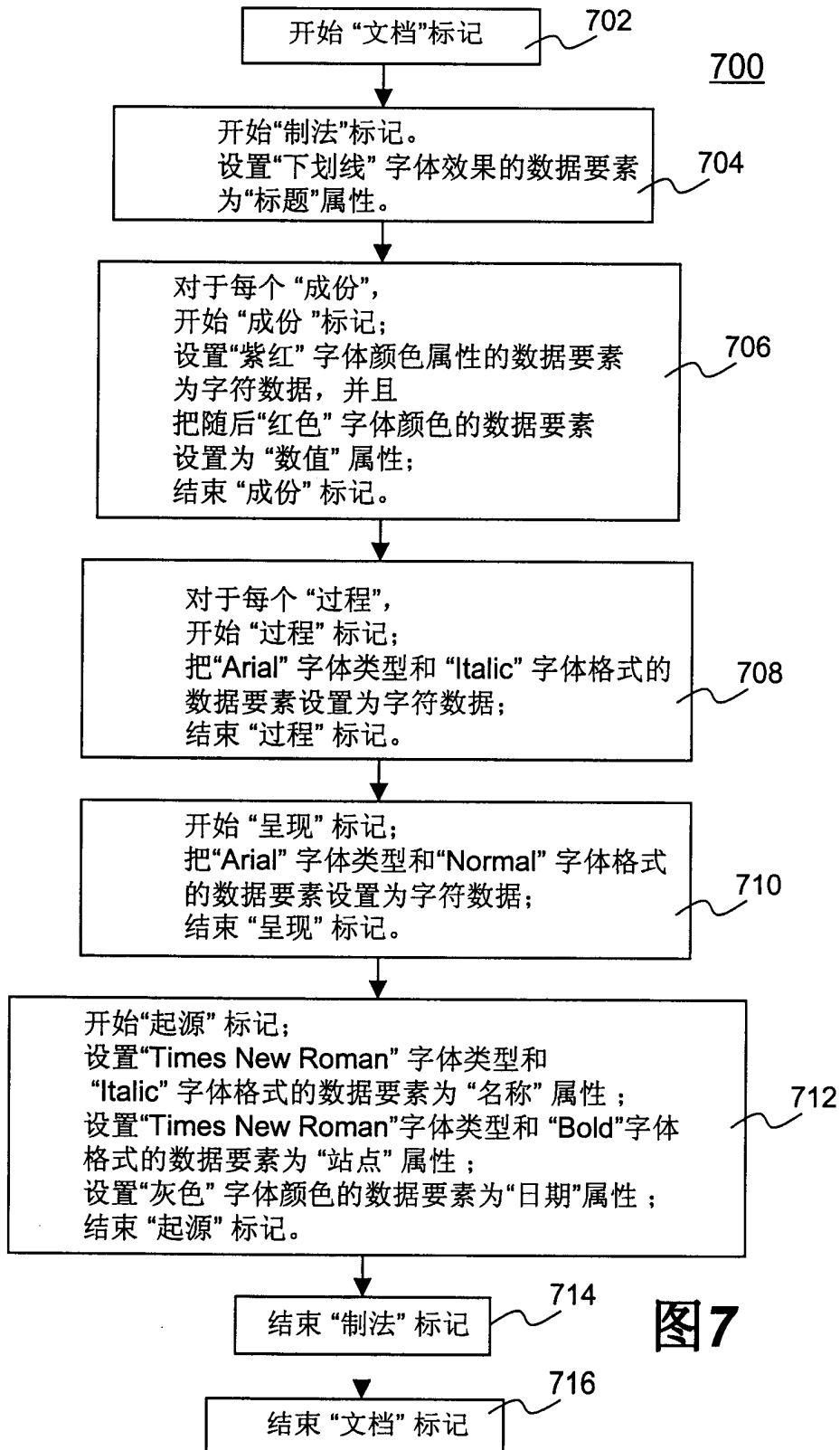
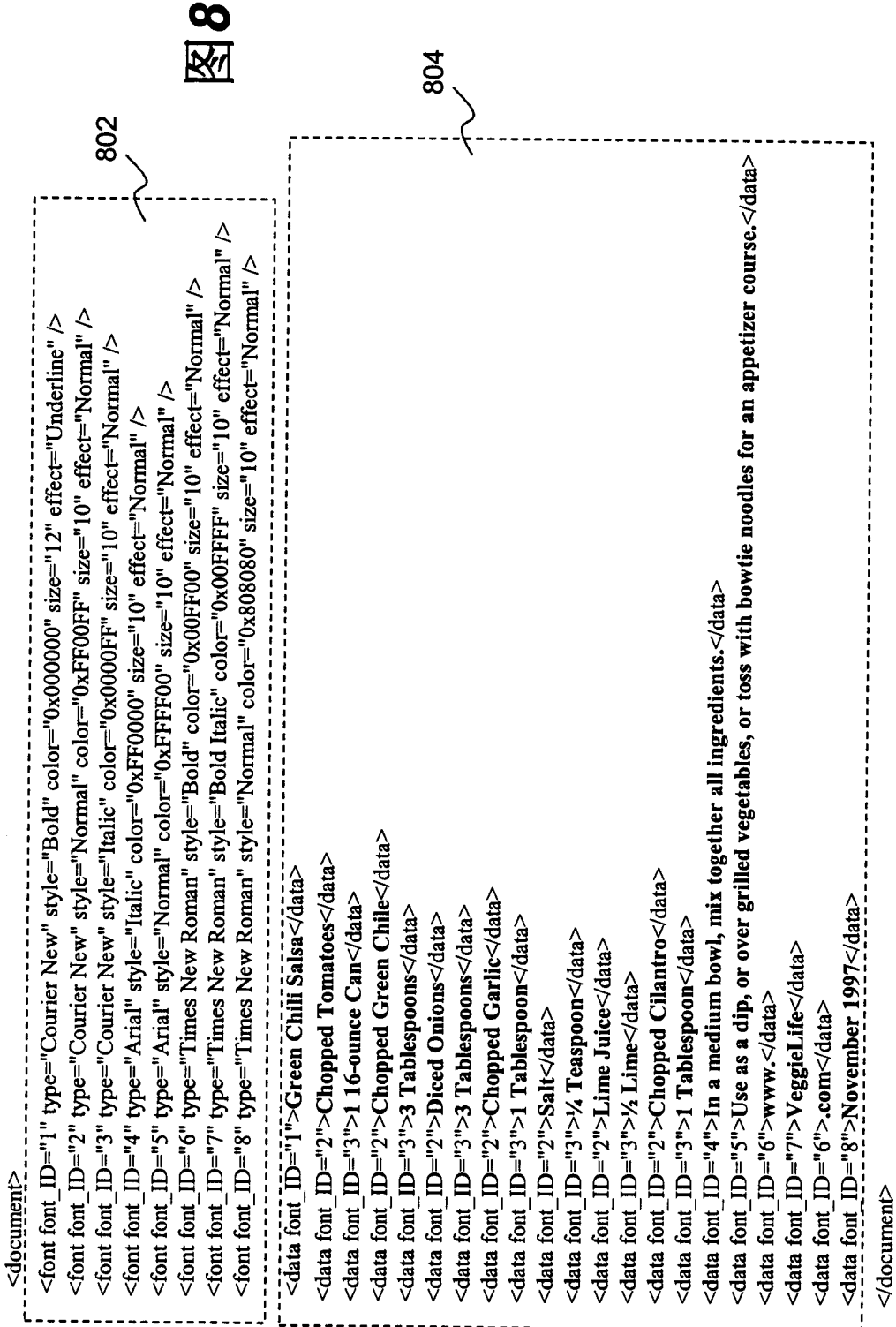
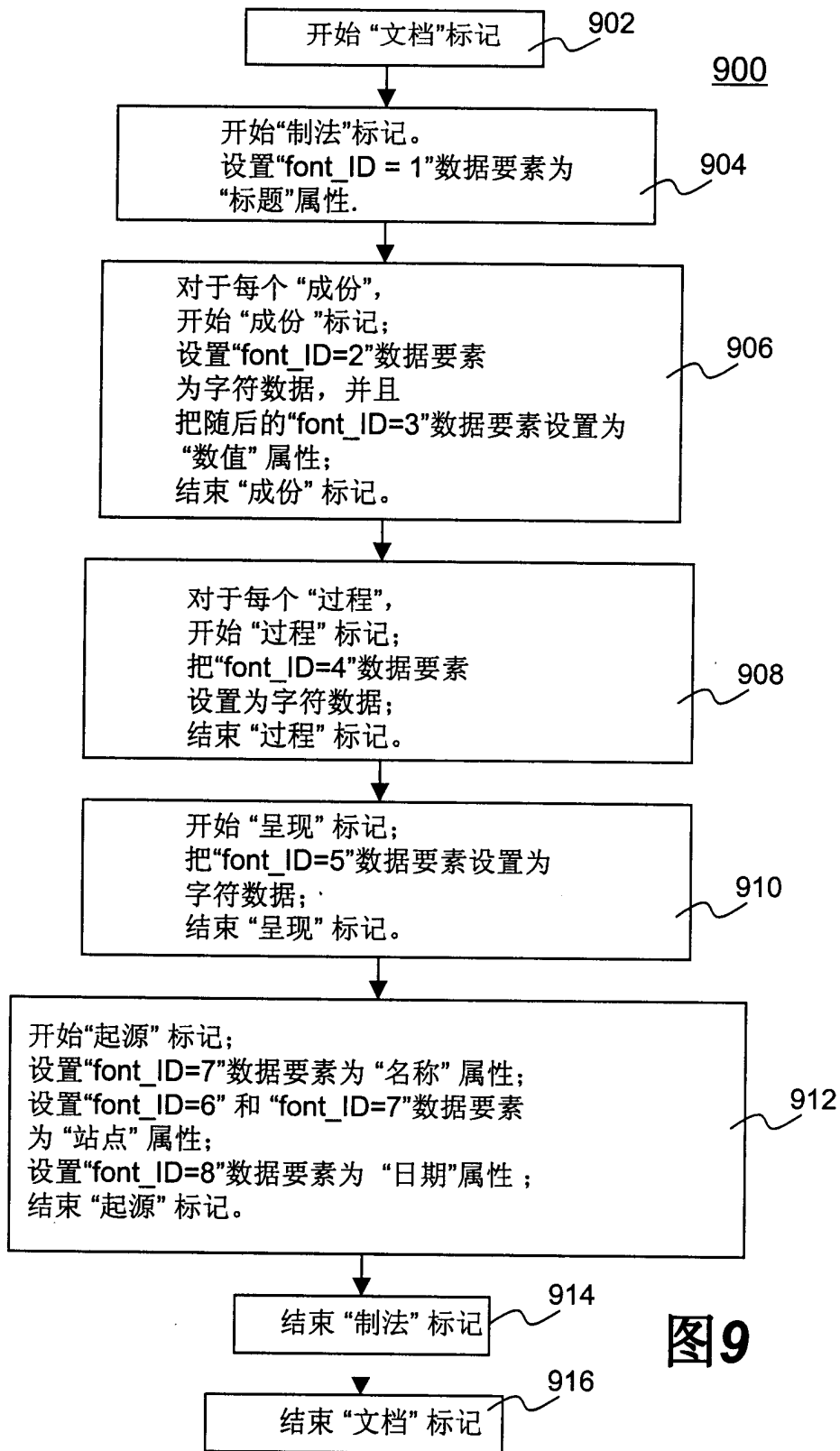


图 6







```

<?xml version="1.0" standalone="yes"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="/">
    <document>
      <xsl:apply-templates/>
    </document>
  </xsl:template>
  <xsl:template match="document">
    <recipe title="{data[@font_ID='1']}">
      <xsl:for-each select="data[@font_ID='2']">
        <ingredient amount="{following-sibling::data[@font_ID='3']}">
          <xsl:value-of select="."/ >
        </ingredient>
      </xsl:for-each>
      <xsl:for-each select="data[@font_ID='4']">
        <procedure step="{position()}">
          <xsl:value-of select="."/ >
        </procedure>
      </xsl:for-each>
      <presentation>
        <xsl:value-of select="data[@font_ID='5']" />
      </presentation>
      <xsl:for-each select="data[@font_ID='7']">
        <originate name="{." website="{preceding-sibling::data[@font_ID='6']}">
          <following-sibling::data[@font_ID='8']" />
        </xsl:for-each>
      </recipe>
    </xsl:template>
  </xsl:stylesheet>

```

图10