



(12) 发明专利

(10) 授权公告号 CN 1602463 B

(45) 授权公告日 2012.03.14

(21) 申请号 02824527.X

(51) Int. Cl.

(22) 申请日 2002.12.05

G06F 7/00(2006.01)

G06F 7/06(2006.01)

(30) 优先权数据

审查员 王丽

10/004,447 2001.12.05 US

(85) PCT申请进入国家阶段日

2004.06.08

(86) PCT申请的申请数据

PCT/US2002/038840 2002.12.05

(87) PCT申请的公布数据

W003/048922 EN 2003.06.12

(73) 专利权人 佳能株式会社

地址 日本东京

(72) 发明人 理查德·K·雅都米安

洛伦·A·伍德

克里斯托弗·J·卡西拉诺

(74) 专利代理机构 中国国际贸易促进委员会专

利商标事务所 11038

代理人 康建忠

权利要求书 1 页 说明书 12 页 附图 6 页

(54) 发明名称

多页 SVG 文档的目录

(57) 摘要

包括多页文本的基于 XML 的文档还包括目录信息，目录信息保存所述多页中每页的位置指针。最好，基于 XML 的文档是遵守可缩放矢量图形 (SVG) 标准的文档。还公开了创作工具和浏览器。

1. 一种用于呈现多页可缩放的矢量图形文档和以基于 XML 的文档为基础的文本中的至少一个的浏览器，所述基于 XML 的文档包括多页文本，其中，每页文本由多个元素类型界定，每个元素类型具有共同遵守定义所述多个元素类型的规则的文档类型定义的开标记和对应的关标记，所述浏览器包括：

用于接收当前页的用户选择的装置；

用于从该基于 XML 的文档中的页信息元素获得与当前页相对应的基于 XML 的文档中的文本的位置的装置，其中所述页信息元素包括当前页在基于 XML 的文档中的位置；

用于解析在与当前页相对应的位置处的基于 XML 的文档中的文本的装置；和

用于根据解析的文本呈现当前页的图像的装置。

2. 按照权利要求 1 所述的浏览器，其中所述用于呈现当前页的图像的装置渐进地呈现当前页的图像和与当前页相邻的各页的图像。

3. 按照权利要求 1 所述的浏览器，其中浏览器根据正在查看的当前页，并根据页信息文本解析基于 XML 的文档，以便呈现并显示当前页。

4. 按照权利要求 1 所述的浏览器，其中渐进地显示当前页和与当前页相邻的各页。

多页 SVG 文档的目录

技术领域

[0001] 本发明涉及多页 SVG 格式文档（这里“SVG”指的是可缩放的矢量图形），更具体地说，涉及定义包括每页的位置信息的文档目录的新元素类型（element type）。本发明还涉及这种多页文档的创作工具，以及查看这种文档的浏览器。

[0002] 背景技术

[0003] 可缩放的矢量图形（“SVG”）是用可扩展置标语言（“XML”）描述二维图形的语言。SVG 允许三种图形对象：矢量图形形状，图像和文本。SVG 文档是明文文档，其包括图形对象和它们的属性的明文描述，例如描述对象位于何处（“URI”）或如何绘制该对象的文本。文本由总称为“元素类型”的开标记和附随的关标记定界。例如，“title”元素类型可用开标记<title>和关标记</title>划定文本界限，不过一些元素类型可在一行中包括开标记和关标记。

[0004] SVG 语言由万维网协会（W3C）主张和定义，在 W3C 的网站 www.w3c.org 可找到 SVG 最新语法的细节。SVG 的最新定义由 W3C 在“Scalable Vector Graphics (SVG) 1.0 specification :W3C Recommendation 04 September 2001”（可在上述网站得到）中定义，该规范作为参考包含于此。该规范包括关于 SVG 的文档类型定义。“文档类型定义”定义多个元素类型遵守的规则和语法。

[0005] 如同 W3G 设想的那样，SVG 定义可在文档内的图形窗口中查看的图形对象。但是最近已考虑把 SVG 的概念扩展到文档本身，包括多页文档。

[0006] 本发明的受让人在 2000 年 9 月 13 日申请的申请 No. 09/661387，“A Scalable Vector Graphics Print Driver”提出了这样的一个例子。根据该申请（其内容作为参考包含于此），打印驱动程序接受来自应用程序的打印输出，并且不是产生打印机的信息，而是产生 SVG 文档。文档可包括多页。所得到的 SVG 文档之后可被用于任意适当的用途，例如贴在因特网上，或者包含在网页中。此外，由于 SVG 文档是明文，并且 SVG 语法与平台无关，因此可跨越计算平台，把 SVG 文档转移给用户，从而 SVG 文档用作独立于平台的文档格式。

[0007] 通过上述申请 No. 09/661387 中定义的打印驱动程序的应用，支持打印的任意应用程序能够输出 SVG 格式化文档，从而实现任意应用程序的打印输出的平台无关性。

[0008] 开发输出 SVG 格式化文档的打印驱动程序的另一种努力由 Software Mechanics Party Ltd. of Brisbane, Australia 开发的 SVGmaker™ 提供，在其网站 www.svgmaker.com 提供了关于其的信息。

[0009] 多页 SVG 格式化文档最近遇到的一个问题涉及呈现和查看这种文档。由于 SVG 最初被设想为单一窗口中图形的格式，为了呈现文档，以便查看该文档，则必须从头至尾解析整个文档。但是，对于多页文档来说，解析整个文档耗时，尤其是如果文档包括许多页。此外，由于在完成解析之前，不能完整地呈现或查看文档，因此即使当只查看文档的一页，或者从一页移动到另一页时，也会遇到较大的延迟。

[0010] 发明内容

[0011] 本发明的目的是提供定义文档目录结构的元素类型，文档目录结构至少包括和 SVG 文档中各页的位置相关的信息。由于能够确定 SVG 文档中各页的位置，因此浏览器只需解析呈现当前页的视图所需的位置。

[0012] 从而，在一个方面，本发明涉及基于 XML 的文档，例如包括由一个以上元素类型定界的文本的 SVG，所述一个以上元素类型至少分别定义矢量图形形状，图像和文本，每个元素类型具有共同遵守定义所述一个以上元素类型的规则的文档类型定义的开标记和对应的关标记。基于 XML 的文档包括多页的文本以及页信息文本，其中页信息文本由页信息元素类型定界，页信息元素类型定义所述多页中每页的文本在文档中的位置。最好，基于 XML 的文档还包括由文档资源元素类型定界的文档资源文本，文档资源元素类型定义在多页内，或者在整个基于 XML 的文档内适用的资源（例如字体定义等）。

[0013] 由于页信息文本的缘故，不必要求呈现基于 XML 的文档的浏览器解析整个基于 XML 的文档。相反，只需要解析文档中由页信息文本识别的那些位置。此外，由于页信息文本由页信息元素类型定界，因此浏览器能够快速确定页信息文本的位置。

[0014] 此外，由于类似地由文档资源元素类型定界的文档资源文本的缘故，浏览器能够快速定位并处理在多页内，或者在整个基于 XML 的文档内适用的资源。

[0015] 在优选实施例中，基于 XML 的文档包括位于文档起点或其附近的 位置指针，位置指针指向由目录表元素类型定界的目录表文本，目录表元素类型直接或间接地定义页信息元素类型的位置。借助这些位置指针，浏览器更快地解析文档，因为能够准确查明文档内，关于任意特定页需要解析的位置。

[0016] 最好，目录表直接或间接包括关于文档资源文本的信息，以及关于文本各页的缩略图及诸如作者和创建日期之类文本属性的信息。

[0017] 本发明的其它方面涉及创作根据本发明的基于 XML 的文档的创作工具，例如打印驱动程序，以及解析、呈现和查看这种基于 XML 的文档的浏览工具。

[0018] 为了实现上述目的，本发明提供了一种用于呈现多页可缩放的矢量图形文档和以基于 XML 的文档为基础的文本中的至少一个的浏览器，所述基于 XML 的文档包括多页文本，其中，每页文本由多个元素类型定界，每个元素类型具有共同遵守定义所述多个元素类型的规则的文档类型定义的开标记和对应的关标记，所述浏览器包括：用于接收当前页的用户选择的装置；用于从该基于 XML 的文档中的页信息元素获得与当前页相对应的基于 XML 的文档中的文本的位置的装置，其中所述页信息元素包括当前页在基于 XML 的文档中的位置；用于解析在与当前页相对应的位置处的基于 XML 的文档中的文本的装置；和用于根据解析的文本呈现当前页的图像的装置。

[0019] 附图说明

[0020] 上面给出了本发明的概要，从而能够很快理解本发明的本质。结合附图，参考本发明的优选实施例的下述详细说明，能够更完整地理解本发明。

[0021] 图 1 图解说明了具体体现本发明的计算设备的外观。

[0022] 图 2 是图 1 中所示计算设备的详细方框图。

[0023] 图 3 是说明根据本发明的文档浏览器的流程图。

[0024] 图 4 图解说明了文档浏览器的多页文档的显示。

[0025] 图 5 是说明本发明的第二实施例的流程图。

[0026] 具体实施方式

[0027] 图 1 表示了包括根据本发明的基于图像的全域映射的典型计算设备 10 的外观。计算设备 10 包括主处理器 11, 主处理器 11 包含个人计算机 (下面称为“PC”), 个人计算机最好具有诸如 Microsoft Windows、Xwindows 或 MacIntosh 操作系统之类视窗操作系统。计算设备 10 配有包括显示屏 14 的彩色监视器 12, 输入文本数据和用户命令的键盘 15, 和点击设备 16。点击设备 16 最好包括鼠标, 用于点击、选择和操作在显示屏 14 上显示的对象。

[0028] 计算设备 10 包括计算机可读存储媒体, 例如硬盘 17 和 / 或软盘驱动器 19 和 / 或 CDROM 驱动器 20。这样的计算机可读存储媒体允许计算设备 10 访问保存在可换的及不可换的存储媒体上的信息, 例如图像数据, 计算机可执行进程步骤, 应用程序等。另外, 网络接入 21 允许计算设备 10 从其它来源, 例如局域网或因特网, 或者从数码相机或数码摄像机获得信息、图像和应用程序。

[0029] 图 2 是表示 PC 11 的内部结构的详细方框图。如图 2 中所示, PC 11 包括与计算机总线 26 连接的中央处理器 (“CPU”) 25。与计算机总线 26 连接的还有硬盘 17, 用于网络接入 21 的网络接口 27, 用作主存储器的随机存取存储器 (“RAM”) 29, 只读存储器 (“ROM”) 30, 软盘接口 31, CDROM 接口 32, 监视器 12 的显示接口 34, 键盘 15 的键盘接口 36, 指示器 16 的鼠标接口 37, 和打印机 24 的打印机接口 39。

[0030] 主存储器 29 与计算机总线 26 连接, 以便在诸如操作系统, 应用程序和设备驱动器之类软件程序的执行过程中, 向 CPU 25 提供 RAM 存储。更具体地说, CPU 25 把计算机可执行进程步骤从硬盘 17 或其它存储媒体装入主存储器 29 的某一区中, 之后从主存储器 29 执行保存的进程步骤, 以便执行软件程序。

[0031] 另外如图 2 中所示, 硬盘 17 包括视窗操作系统 41, 应用程序 42, 例如获得、操作和打印多页文档的应用程序, 多页 SVG 文档 43, 诸如打印机驱动程序 45 之类的设备驱动程序 44, 创作多页 SVG 文档的 SVG 创作 (authoring) 工具 46, 和读取包括多页 SVG 文件的 SGV 数据文件, 并在监视器 12 上显示这些文件的 SVG 浏览器 47。

[0032] 根据本发明的多页 SVG 文档的创作最好被实现成打印机驱动程序 45 的一部分, 但是也可被实现成独立的软件应用程序。也可把创作工具实现成动态链接库 (“DLL”), 或者实现成其它应用程序, 例如 Microsoft Corporation 的 WordTM 或 Quark, Inc. 的 QuarkXPressTM 的插件。同样地, SVG 浏览器 47 最好被实现成 DLL 或者诸如 Microsoft Corporation 的 Internet ExplorerTM 之类另一程序的插件。

[0033] 创作工具 46 被安排成输出基于 XML 的文档, 该基于 XML 的文档包括至少分别定义矢量图形形状、图像和文本的多个文本定界元素类型。每个元素类型具有共同遵守定义所述多个文档类型的规则和语法的文档类型定义的开标记和相应的关标记。最好, 基于 XML 的文档是符合 W3C 标准的 SVG 格式化文档, 这种情况下, 文档类型定义 (下面称为 “DTD”) 是由 W3C 定义, 并由下面描述的 DTD 扩充的文档类型定义。

[0034] 基于 SVG 的文档包括多于一页的文本。文本不必是在呈现的文档中实际出现的文本, 相反可包括表示文本可能位于的基于 web 的 URI 的文本, 或者包括定义文档各页的各个方面其它基于 SVG 的元素类型的文本。最好, 文档各页的文本最好和该页的其它方面聚集在一起, 但是该页的文本也可被分离并分散到整个 SVG 文档中, 只要该页位置信息 (下面

说明) 正确识别任意一页中所有元素的位置。

[0035] 在输出基于 XML 文档的文本的过程中,创作工具 46 还输出定义一页或多页使用的,或者所有页全体使用的资源的文本。这样的资源例如包括字体定义,形状,剪辑区等。

[0036] 创作工具 46 还输出由目录存在 (directory-exists) 元素类型定界的目录存在文本,目录存在元素类型表示 SVG 文档包含目录信息,目录信息直接或间接地指示多页文档中每页的位置。目录存在文本最好位置 SVG 文档的起点或者位于其附近,从而解析 SVG 文档的浏览器 (viewer) 将及早遇见目录存在文本,并且能够定位并取回文本每页的位置信息,而不必解析整个 SVG 文档。这种目录信息最好由创作工具 46 写在 SVG 文档的结尾,以便简化浏览器进行的解析。

[0037] 本发明中,目录存在文本指向目录表元素,目录表元素间接提供多页文档中每页的位置信息。间接信息的原因在于在整个目录结构中,允许额外多层的抽取 (abstraction),从而实现文档的更普遍并且可扩展的目录描述。

[0038] 目录表元素包括由目录表元素类型定界的目录表文本,并且只包括关于表清单 (list) 文本的附加位置指针。这种程度的间接性是合乎需要的,因为它确保在根据本发明的所有多页 SVG 文档中,目录表文本总是位于相同位置 (相对于 SVG 文档的尾部)。

[0039] 目录表清单文本包括由目录表清单元素类型定界的文本,目录表清单元素类型定义跟随有零个或多个 doc 资源元素的至少一个目录定义元素。doc 资源元素包括指定 SVG 文档中,在一页或多页内适用的资源 (例如字体定义等) 的位置指针的文本。doc 资源文本由 doc 资源元素类型定界。

[0040] 目录定义文本由目录定义元素类型定界,指定目录表信息元素的位置。目录信息元素又包括指定文档中页数的文本,以及至少一个页目录元素。目录信息文本还可包括零个或多个缩略图目录元素和 doc 属性元素。

[0041] 页目录文本元素包含 SVG 文档中每页的一个页信息元素。每个页信息元素又包括指定对应于多页文档的每页的文本在 SVG 文档中的位置的文本。最好,SVG 文档的每页被聚集在一起,从而每页只需要一个页信息元素,不过在每页的文本分散在整个多页文档内的情况下,也可包括一个以上的页信息元素。

[0042] 返回目录信息元素,该元素还可包括零个或多个缩略图 - 目录元素和零个或多个文档属性元素。缩略图信息元素包含代表多页文档中每页的缩略图的定义。文档属性元素包括诸如作者和创建日期之类的文档属性。

[0043] [文档类型定义 (“DTD”)]

[0044] 上面提供了关于文档目录的元素的简要描述。优选的 DTD 的完整副本如下,并被用于扩充 W3C 提供的关于 SVG 文档的 DTD :

[0045] <! ELEMENT docDirectoryExists EMPTY>

[0046] <! ATTLIST docDirectoryExists

offset% Integer ;#REQUIRED>

[0048] <! ELEMENT pageStart EMPTY>

[0049] <! ATTLIST pageStart>

[0050] <! ELEMENT pageEnd EMPTY>

[0051] <! ATTLIST pageEnd>

```
[0052]      <! ELEMENT docDirectoryTable EMPTY>
[0053]      <! ATTLIST docDirectoryTable
[0054]          offset% Integer ;#REQUIRED
[0055]          byteCount% Integer ;#REQUIRED>
[0056]      <! ELEMENT docDirectoryDefs EMPTY>
[0057]      <! ATTLIST docDirectoryDefs
[0058]          offset% Integer ;#REQUIRED
[0059]          byteCount% Integer ;#REQUIRED>
[0060]      <! ELEMENT docResources EMPTY>
[0061]      <! ATTLIST docResources
[0062]          offset% Integer ;#REQUIRED
[0063]          byteCount% Integer ;#REQUIRED>
[0064]      <! ELEMENT directoryTableList(docDirectoryDefs,
docResources*)>
[0065]      <! ATTLIST directoryTableList>
[0066]      <! ELEMENT docPageDir(docPageInfo+)>
[0067]      <! ATTLIST docPageDir>
[0068]      <! ELEMENT docThumbnailDir(docThumbnailInfo*)>
[0069]      <! ATTLIST docThumbnailDir>
[0070]      <! ELEMENT docPageInfo EMPTY>
[0071]      <! ATTLIST docPageInfo
[0072]          pageNumber% Integer ;#REQUIRED
[0073]          offset% Integer ;#REQUIRED
[0074]          byteCount% Integer ;#REQUIRED
[0075]          width% Length ;#REQUIRED
[0076]          height% Length ;#REQUIRED
[0077]          color(true|false)#IMPLIED>
[0078]      <! ELEMENT docThumbnailInfo(image)>
[0079]      <! ATTLIST docThumbnailInfo
[0080]          pageNumber% Integer ;#REQUIRED>
[0081]      <! ELEMENT docAttributes EMPTY>
[0082]      <! ATTLIST docAttributes
[0083]          author          CDATA    #IMPLIED
[0084]          creationDate   CDATA    #IMPLIED
[0085]          modifiedDate   CDATA    #IMPLIED
[0086]          title          CDATA    #IMPLIED
[0087]          subject         CDATA    #IMPLIED
[0088]          lastSavedBy    CDATA    #IMPLIED
[0089]          revisionNumber CDATA    #IMPLIED
```

```

[0090]           applicationName    CDATA   #IMPLIED
[0091]           companyName     CDATA   #IMPLIED
[0092]           lastViewedDate  CDATA   #IMPLIED
[0093]           lastViewedBy   CDATA   #IMPLIED
[0094]           keywords        CDATA   #IMPLIED
[0095]           originalFileName CDATA   #IMPLIED>
[0096]           <! ELEMENT docDirectoryInfo(docPageDir, docThumbnailDir,
[0097]                         docAttributes)>
[0098]           <! ATTLIST docDirectoryInfo
[0099]                         pageCount% Integer ;#REQUIRED>

[0100] [元素类型的定义]
[0101] 下面提供元素的更完整定义：
[0102] [docDirectoryExists 元素]
[0103] docDirectoryExists 元素指示 SVG 文档文件包含目录信息，并指定 docDirectoryTable 元素的位置（到文件结尾的字节偏移量）
[0104] <! Element docDirectoryExists EMPTY>
[0105] <! ATTLIST docDirectoryExists
[0106]               offset% Integer ;#REQUIRED>
[0107] 属性定义：
[0108] offset = “<integer>”:从文档文件的结尾到 docDirectoryTable 元素的字节偏移量。
[0109] [docDirectoryTable 元素]
[0110] docDirectoryTable 元素规定 directoryTableList 元素的位置（自文件结尾的字节偏移量）和大小（字节数）。
[0111] <! Element docDirectoryTable EMPTY>
[0112] <! ATTLIST docDirectoryTable
[0113]               offset% Integer ;#REQUIRED
[0114]               byteCount% Integer ;#REQUIRED>
[0115] 属性定义：
[0116] offset = “<integer>”:从文档文件的结尾到 directoryTableList 元素的字节偏移量。数值串由左侧填充 0 的四个数字组成。
[0117] byteCount = “<integer>”:用字节表示的 directoryTableList 元素的长度。数值串由左侧填充 0 的四个数字组成。
[0118] [directoryTableList 元素]
[0119] directoryTableList 元素包含跟随有 0 或多个 docResource 元素的一个 docDirectoryDefs 元素。
[0120] <! Element directoryTableList(docDirectoryDefs, docResources*)>
[0121] <! ATTLIST directoryTableList>
[0122] 属性定义：

```

- [0123] 无
- [0124] [docDirectoryDefs 元素]
- [0125] docDirectoryDefs 元素规定 directoryTableInfo 元素的位置（自文件起点的字节偏移量）和大小（字节数）。
- [0126] <! Element docDirectoryDefs EMPTY>
- [0127] <! ATTLIST docDirectoryDefs
- [0128] offset% Integer ;#REQUIRED
- [0129] byteCount% Integer ;#REQUIRED>
- [0130] 属性定义：
- [0131] offset =“<integer>”:从文档文件的起点到 docDirectoryInfo 元素的字节偏移量。
- [0132] byteCount =“<integer>”:用字节表示的 directoryDefinitions 元素的长度。
- [0133] [docResources 元素]
- [0134] docResources 元素规定多页上 SVG 定义（嵌入字体定义,defs 元素等）参考的位置（自文件起点的字节偏移量）和大小（字节数）。
- [0135] <! Element docResources EMPTY>
- [0136] <! ATTLIST docResources
- [0137] offset% Integer ;#REQUIRED
- [0138] byteCount% Integer ;#REQUIRED>
- [0139] 属性定义：
- [0140] offset =“<integer>”:从文档文件的起点到共享资源定义的起点的字节偏移量。
- [0141] byteCount =“<integer>”:用字节表示的共享资源定义的长度。
- [0142] [docDirectoryInfo 元素]
- [0143] docDirectoryInfo 元素规定 SVG 文档中的页数，并包含所需的 docPageDir 元素。该元素还可包括可选的 docThumbnailDir 和 docAttributes 元素。
- [0144] <! Element docDirectoryInfo(docPageDir, docThumbnailDir ? ,
[0145] docAttributes ?)>
- [0146] <! ATTLIST docDirectoryInfo
- [0147] pageCount% Integer ;#REQUIRED>
- [0148] 属性定义：
- [0149] pageCount =“<integer>”:SVG 文档中的页数。
- [0150] [docAttributes 元素]
- [0151] docAttributes 元素规定 SVG 文档信息关键字的值。
- [0152] <! Element docAttributes EMPTY>
- [0153] <! ATTLIST docAttributes
- [0154] author CDATA #IMPLIED
- [0155] creationDate CDATA #IMPLIED
- [0156] modifiedDate CDATA #IMPLIED
- [0157] title CDATA #IMPLIED

[0158]	subject	CDATA	#IMPLIED
[0159]	lastSavedBy	CDATA	#IMPLIED
[0160]	revisionNumber	CDATA	#IMPLIED
[0161]	applicationName	CDATA	#IMPLIED
[0162]	companyName	CDATA	#IMPLIED
[0163]	lastViewedDate	CDATA	#IMPLIED
[0164]	lastViewedBy	CDATA	#IMPLIED
[0165]	keywords	CDATA	#IMPLIED
[0166]	originalFileName	CDATA	#IMPLIED>

[0167] 属性定义：

[0168] author = “<cdata>”: 文档作者的姓名。

[0169] creationDate = “<cdata>”: 利用格式“年 / 月 / 日 小时 : 分钟 : 秒”, 产生文档的日期。

[0170] modifiedDate = “<cdata>”: 最后修改文档的日期。

[0171] title = “<cdata>”: 文档的题目。

[0172] subject = “<cdata>”: 文档的主题。

[0173] lastSavedBy = “<cdata>”: 最后修改文档的个人的姓名。

[0174] revisionNumber = “<cdata>”: 文档修改次数。

[0175] applicationName = “<cdata>”: 产生该 SVG 文档的应用程序的名称。

[0176] companyName = “<cdata>”: 产生该文档的公司的名称。

[0177] lastViewerDate = “<cdata>”: 最后查看该文档的日期。

[0178] lastViewedBy = “<cdata>”: 最后查看该文档的个人的姓名。

[0179] keywords = “<cdata>”: 文档搜索关键字。

[0180] originalFileName = “<cdata>”: 文档的初始文件名。

[0181] [docPageDir 元素]

[0182] docPageDir 元素包含 SVG 文档中每页的一个 docPageInfo 元素。

[0183] <! Element docPageDir(docPageInfo+)>

[0184] <! ATTLIST docPageDir>

[0185] 属性定义：

[0186] 无

[0187] [docThumbnailDir 元素]

[0188] docThumbnailDir 元素包含存在于 SVG 文档中的每页缩略图的一个 docThumbnailInfo 元素。

[0189] <! Element docThumbnailDir(docThumbnailInfo*)>

[0190] <! ATTLIST docThumbnailDir>

[0191] 属性定义：

[0192] 无

[0193] [docPageInfo 元素]

[0194] docPageInfo 元素规定 SVG 文档中某一页的页码, 位置 (自文件起点的字节偏移

量) 和大小(字节量)。可选的颜色属性规定该页是否使用除白、黑或灰之外的颜色(如果未指定,则默认为假)。

- [0195] <! Element docPageInfo EMPTY>
- [0196] <! ATTLIST docPageInfo
 - [0197] pageNumber% Integer ;#REQUIRED
 - [0198] offset% Integer ;#REQUIRED
 - [0199] byteCount% Integer ;#REQUIRED
 - [0200] width% Length ;#REQUIRED
 - [0201] height% Length ;#REQUIRED
 - [0202] color(true|false) #IMPLIED>
- [0203] 属性定义:
 - [0204] pageNumber = “<integer>”:本元素描述页码。第一页为 1。
 - [0205] offset = “<integer>”:从文档文件的起点到 pageStart 元素的字节偏移量。
 - [0206] byteCount = “<integer>”:用字节表示的页定义元素的长度。
 - [0207] width = “<length>”:可选地带有标准单位标识符,例如 pt, cm 或 in 的实数格式的页宽。
 - [0208] height = “<length>”:可选地带有标准单位标识符,例如 pt, cm 或 in 的实数格式的页高。
 - [0209] color = “<ture/false>”:指示该页是否使用除白、黑或灰之外的颜色。
 - [0210] [docThumbnailInfo 元素]
- [0211] docThumbnailInfo 元素包含页缩略图的图形的定义。所需的 pageNumber 属性规定缩略图代表的 SVG 文档页的页码。
- [0212] <! Element docThumbnailInfo(image)>
- [0213] <! ATTLIST docThumbnailInfo
 - [0214] pageNumber% Integer ;#REQUIRED>
- [0215] 属性定义:
 - [0216] pageNumber = “<integer>”:本元素定义缩略图代表的 SVG 文档页的页码。第一页为 1。
 - [0217] [例子]
- [0218] 下面提供一页以上的 SVG 文档的一个简单例子。确切地说存在两页。第一页仅由文本“Page one”组成,第二页仅由文本“Page two”组成。该 SGV 文档包括呈剪辑路径形式的全局资源,并包括根据本发明的目录信息。
- [0219] 本例还强调 SVG 文档不必包括对元素类型遵守的 DTD 的明确参考。相反,即使不明确给出对 DTD 的参考,对于元素类型来说,遵守 DTD 就足够了:
- [0220] <? xml version = " 1.0 " standalone = " no " ? >
- [0221] <! --Canon SVG Driver Copyright(C) 2001 Canon Inc.-->
- [0222] <svg width = " 612pt " height = " 792pt " >
- [0223] <docDirectoryExists offset = " 59 " />
- [0224] <defs><! --SVG :commonly used clippath-->

```
[0225] <clipPath id = " clipPath0" >
[0226]   <rect x = " 21.3" y = " 482.4" width = " 18.25" height = " 7.57" />
[0227] </clipPath>
[0228] </defs>
[0229] <text x = " 15.0" y = " 16.6" >Page one</text>
[0230] ...
[0231] <text x = " 15.0" y = " 16.6" >Page two</text>
[0232] ...
[0233] <docDirectoryInfo pageCount = " 2" >
[0234]   <docPageDir>
[0235]     <docPageInfo pageNumber = " 1" offset = " 308"
[0236]       byteCount = " 1235" width = " 612.000pt"
[0237]       height = " 792.000pt" />
[0238]     <docPageInfo pageNumber = " 2" offset = " 1543"
[0239]       byteCount = " 2357" width = " 8.5in" height
= " 11in"
[0240]       color = " true" />
[0241]   </docPageDir>
[0242]   <docThumbnailDir>
[0243]     <docThumbnailInfo pageNumber = " 1" >
[0244]       <image x = " 0" y = " 0" width
= " 18" height = " 23"
[0245]         xlink :href = " data:image/png ;
base64,....." />
[0246]     </docThumbnailInfo>
[0247]   </docThumbnailDir>
[0248]   <docAttributes author = " Loren Wood"
[0249]     creationDate = " 2001/11/28 17:05:43"
[0250]     title = " SVG Document Directory Element
Definitions"
[0251]     revisionNumber = " 0.10"
[0252]   ./>
[0253] </docDirectoryInfo>
[0254] <directoryTableList>
[0255]   <docDirectoryDefs offset = " 3900" byteCount = " 423" />
[0256]   <docResources offset = " 163" byteCount = " 145" />
[0257] </directoryTableList>
[0258] <docDirectoryTable offset = " 0204" byteCount = " 0145" />
[0259] </svg>
```

[0260] [SVG 浏览器 47]

[0261] 图 3 是图解说明 SVG 浏览器 47 解析、呈现和显示包括具有根据本发明的目录信息的多页 SVG 文档在内的 SVG 文档的操作的流程图。

[0262] 简单地说,根据图 3,SVG 浏览器 47 确定多页 SVG 文档的文档目录信息是否存在,如果存在这种信息,则 SVG 浏览器 47 获得和多页文档中每页的文本对应的位置信息。SVG 浏览器 47 还获得一页或多页使用的,或者整个文档整体使用的文档资源的位置信息。根据该信息以及根据正在查看的当前页,SVG 浏览器 47 解析当前页的文本,并利用当前页所需的任意文档资源呈现当前页。之后在监视器 12 上向用户显示呈现的页面,以及包含在文档目录中的任意缩略图或文档属性信息。

[0263] 更具体地说,在步骤 S301 中,SVG 浏览器 47 解析目录存在元素。如果未找到目录存在元素(步骤 S302),则流程转到步骤 S304,在步骤 S304 中,浏览器 47 解析整个 SVG 文档,呈现整个 SVG 文档(步骤 S305),并显示整个 SVG 文档(步骤 S306),所有这些步骤和解析、呈现并显示不包含多页信息的 SVG 文档的现有技术相一致。

[0264] 如果找到目录存在元素,则流程进入步骤 S308,在步骤 S308 中,浏览器 47 跳转到目录表元素,之后根据目录表元素中的位置指针,跳转到目录表清单元素(步骤 S309)。如果目录表清单元素包含文档资源元素(步骤 S310),则流程转到步骤 S311,其中浏览器 47 解析位于文档资源元素中的文本所指位置的文档资源。在任意一种情况下,流程随后进入步骤 S313,其中浏览器 47 根据包含在目录表清单元素的目录定义元素中的位置指针,跳转到目录信息元素。

[0265] 在步骤 S314,浏览器 47 从目录信息元素获得页数。对于当前页,步骤 S315 的执行导致浏览器 47 从对应的一个或多个页信息元素(取决于在多页 SVG 文档内,每页的信息是否聚集在一起),获得页位置信息。

[0266] 在步骤 S316 中,浏览器 47 对位于当前页的位置的文本,解析 SVG 文档。一般,浏览器 47 将排除其它页,只对当前页解析 SVG 文档,从而和在不存在文档目录的情况下,将根据步骤 S304-S306 进行的解析和呈现过程相比,这种解析和呈现过程较快。

[0267] 如果存在缩略图目录元素(步骤 S317),则浏览器 47 获得对应的缩略图信息(步骤 S318)。同样地,如果文档属性元素存在(步骤 S320),则浏览器 47 获得对应的文档属性信息(步骤 S321)。

[0268] 在步骤 S323 中,浏览器 47 利用在步骤 S311 中获得的任意所需的文档资源,呈现当前一页(或多页)。之后,在步骤 S325 中,浏览器 47 显示当前页及缩略图和文档属性(如果存在缩略图和文档属性)。如果收到显示新的一页的请求(步骤 S326),则流程返回步骤 S315,对新请求的一页重复该过程。

[0269] 依据上述例子中提供的 SVG 文档的处理,图 4 图解说明浏览器 47 的输出。如图 4 中所示,浏览器 47 在监视器 12 上形成显示窗口 81,显示窗口 81 包括页显示区 82,缩略图显示区 83 和属性区 84。页显示区 82 包括根据步骤 S323 和 S325 一个或多个当前页的完整呈现。缩略图区 83 显示在步骤 S317 和 S318 中显示的任意缩略图,而在步骤 S320 和 S321 中获得的任意文档属性显示在属性区 84 中。

[0270] 对于对应于步骤 S315-S325 的当前显示页(图 4 中为“Page 2”)来说,在 85 突出显示缩略图图像,以便在相对于整个文档的当前页的位置,向用户提供导航反馈。

[0271] 图 5 是图解说明根据第二实施例的操作的流程图, 第二实施例以页显示区 82 中, 多页的渐进显示为特征。图 5 中, 步骤 S501-S514 和图 3 中的对应步骤 S301-S314 相似, 为了简洁起见, 不再赘述。

[0272] 在步骤 S515 中, 浏览器只获得当前选择的页面的信息。具体地说, 浏览器根据在多页 SVG 文档内, 各页的信息是否被聚集在一起, 从对应的一个或多个页信息元素获得页位置信息。

[0273] 在步骤 S516 中, 浏览器对位于当前页的位置的文本, 解析 SVG 文档。鉴于本实施例的渐进本质, 浏览器排除其它各页 (尤其是排除和当前页相邻的各页), 只对当前页解析 SVG 文档。

[0274] 如果存在缩略图目录元素 (步骤 S517), 则对于当前页, 浏览器获得对应的缩略图信息 (步骤 S518)。同样地, 如果存在当前页的文档属性元素 (步骤 S520), 则浏览器获得对应的文档属性信息 (步骤 S512)。

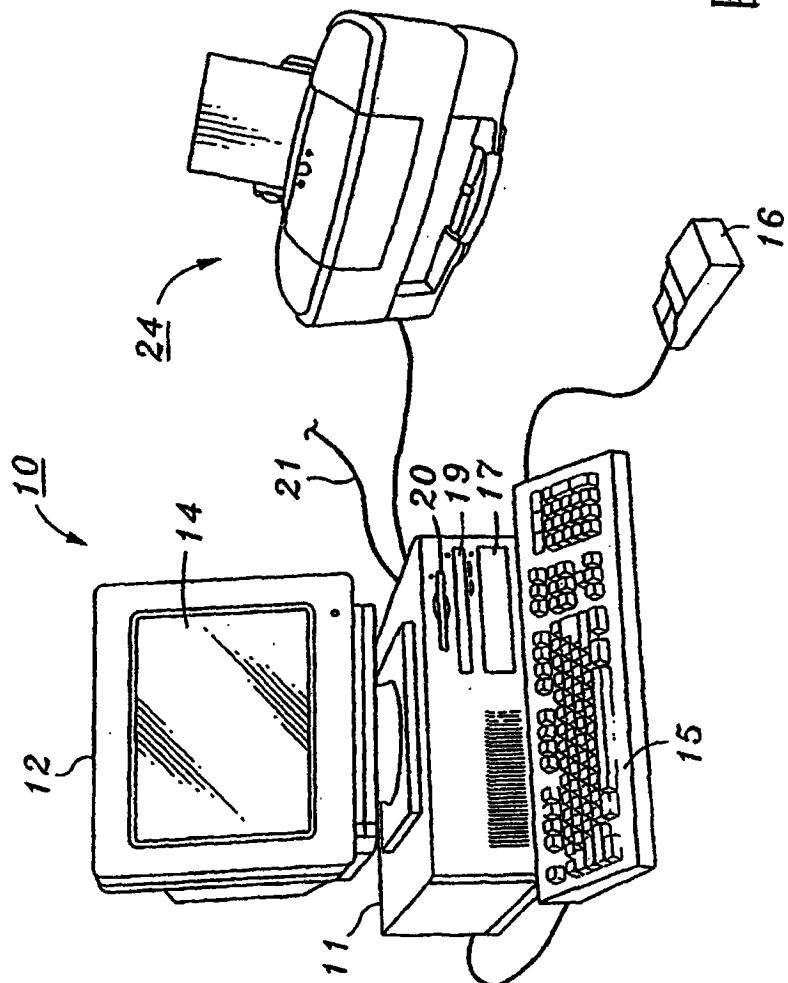
[0275] 在步骤 S523 中, 浏览器利用迄今获得的任意所需文档资源, 只呈现当前页。之后, 在步骤 S525 中, 浏览器显示当前页以及缩略图和文档属性。当前页显示在显示区 82 中。与当前页相邻的各页仍然不显示 (即, 到执行步骤 S525 时为止不被显示), 但是最好对与当前页相邻的各页, 显示占位符图像 (例如虚像轮廓)。

[0276] 由于对于与当前页相邻的各页, 只显示占位符图像, 因此更快并且渐进地显示当前页, 从而加快了当前选择的页面的显示。之后, 呈现 (步骤 S526) 并显示 (步骤 S527) 与当前页相邻的各页。这提供其中首先显示当前页, 随后显示与当前页相邻各页的渐进显示。这种渐进显示提高了显示当前页的速度, 同时保持了用户前进到不同页面的导航提示。

[0277] 在步骤 S528 中, 如果收到把新的一页显示成当前页的请求, 则流程返回步骤 S515, 对最新请求的当前页重复上述过程。在该显示中重新使用任意已呈现的页面。例如, 如果用户选择某一相邻页作为新的当前页, 则显示在先当前页, 而不必重新呈现所述在先当前页。

[0278] 参考特定的例证实施例, 说明了本发明。显然本发明并不局限于上述实施例, 在不脱离本发明的精神和范围的情况下, 本领域的普通技术人员可做出各种变化和修改。

图 1



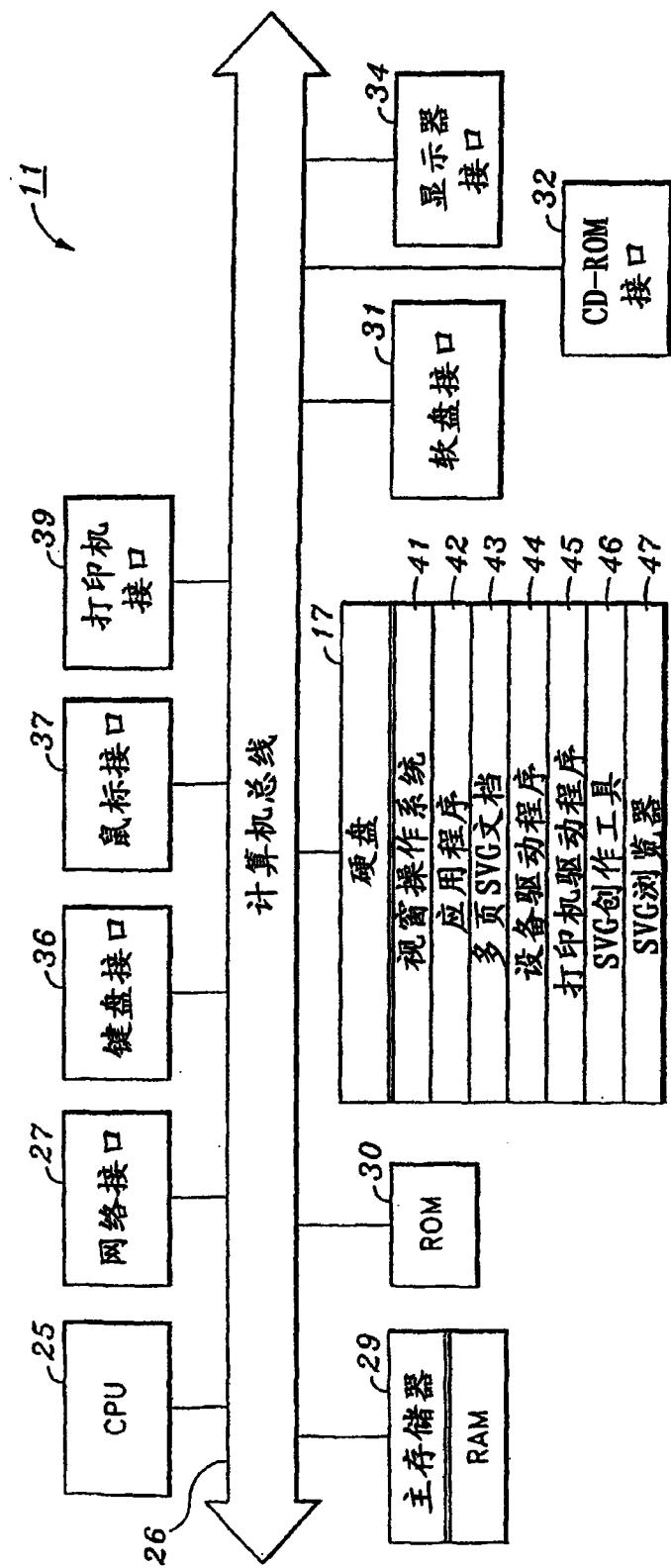


图 2

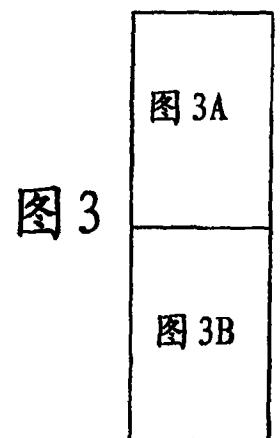
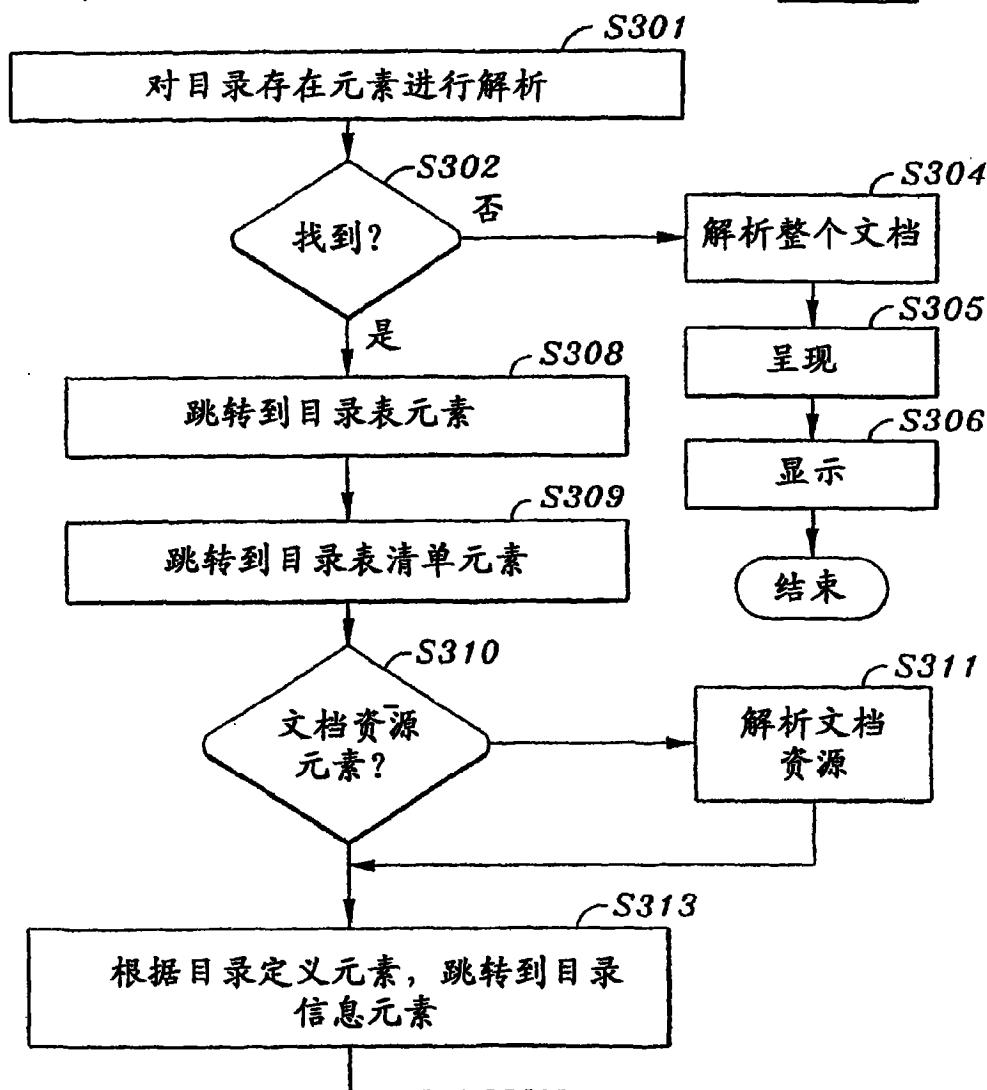


图 3A



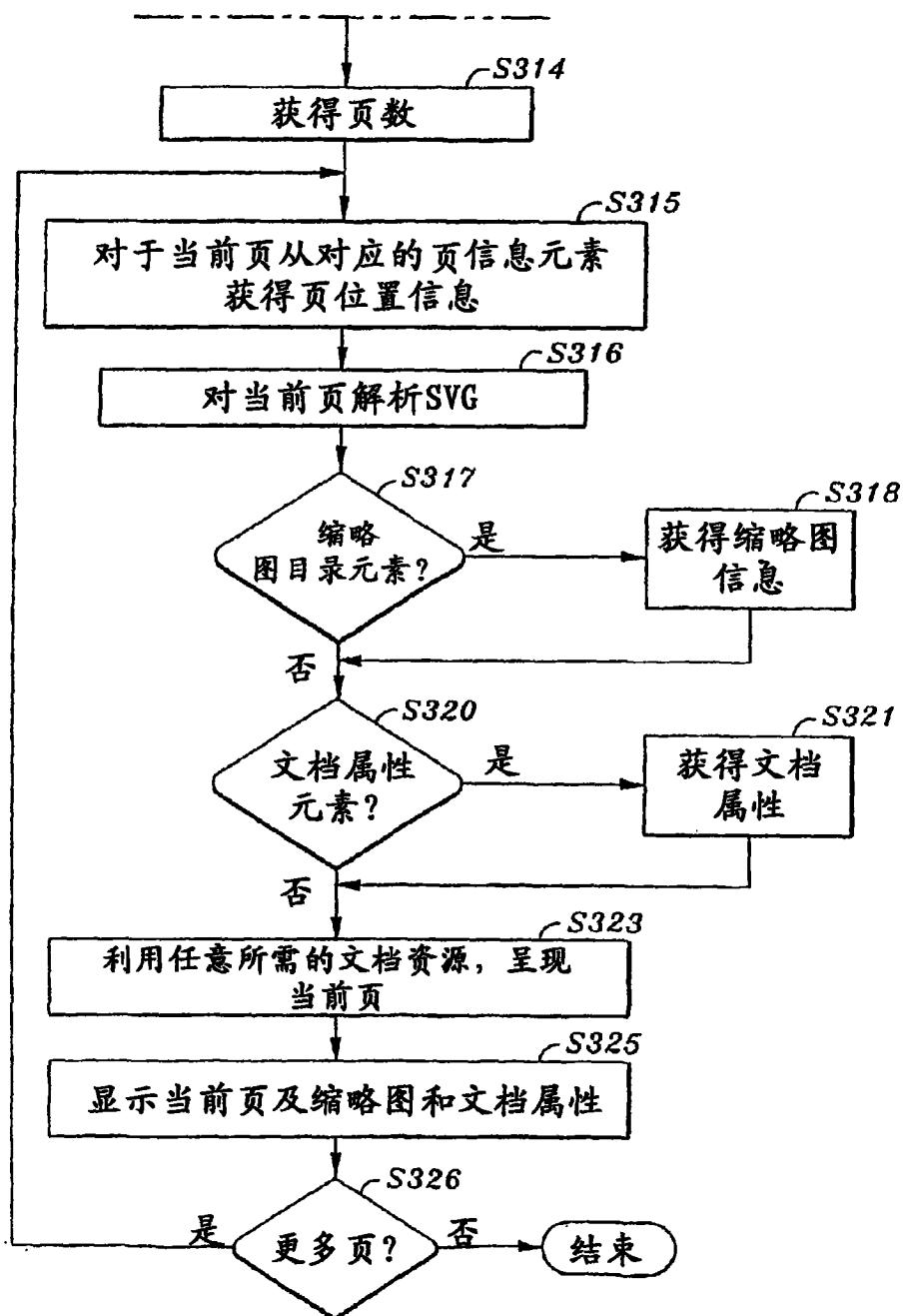
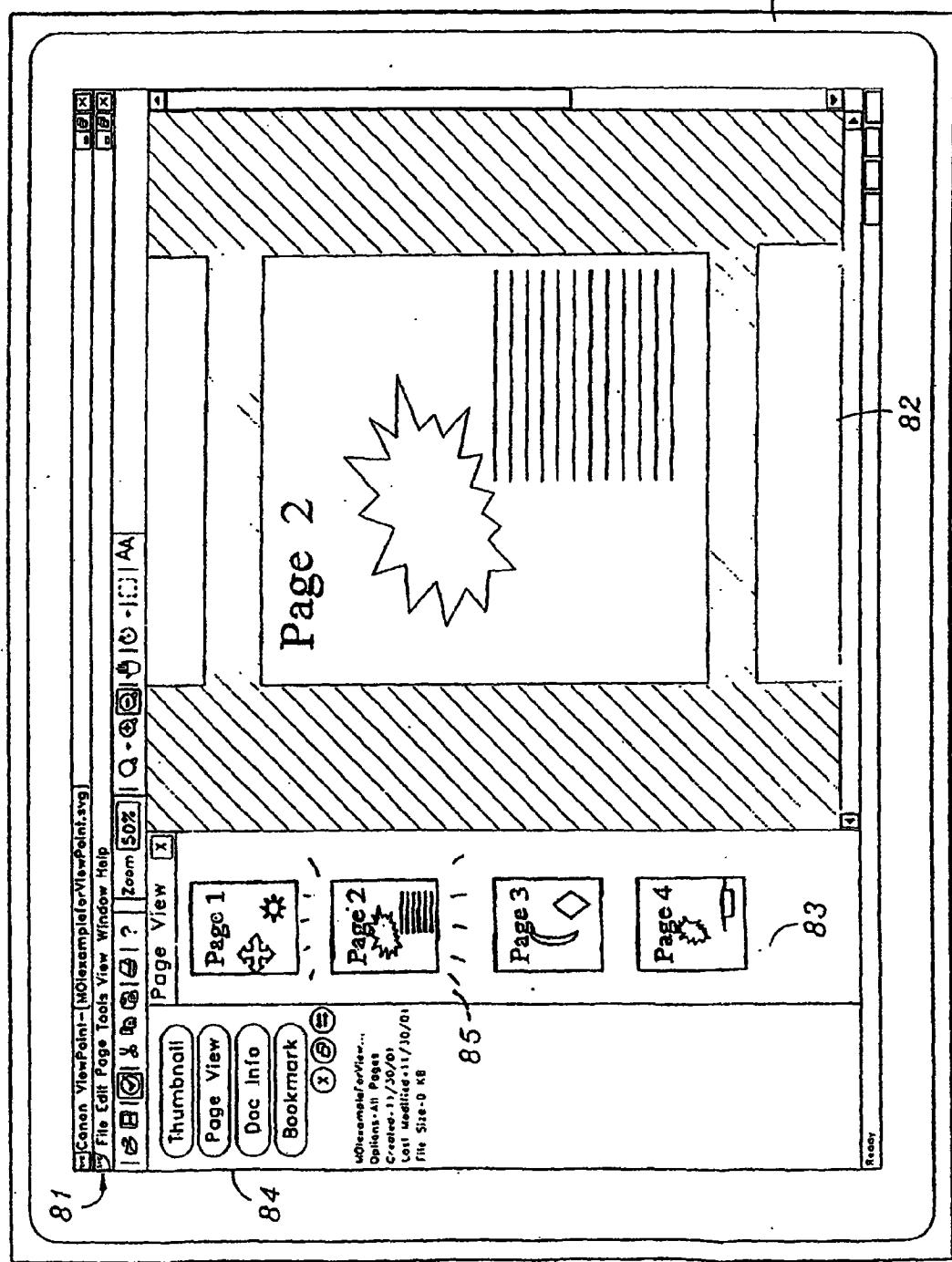


图 3B

图 4



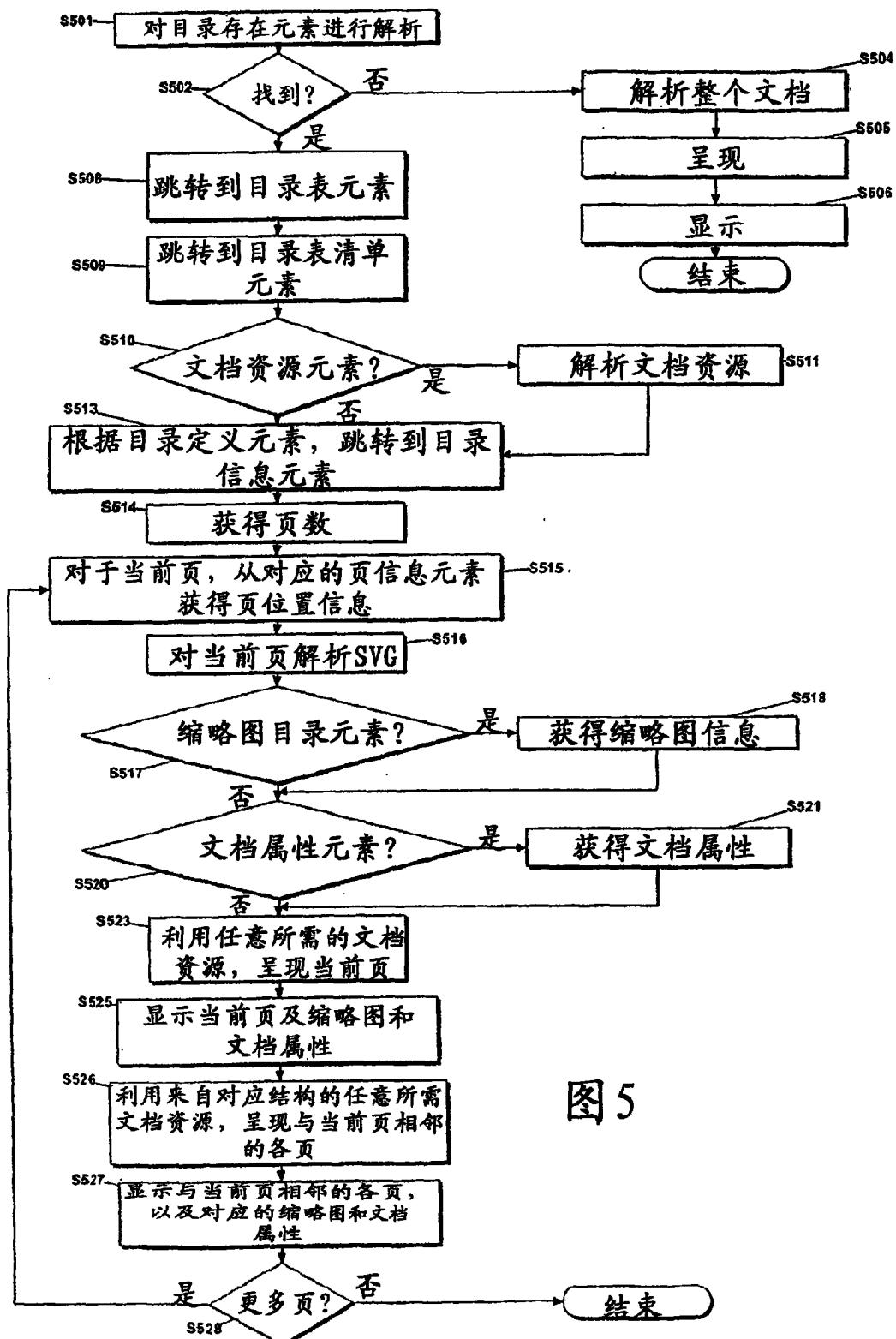


图 5