(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 104956362 B
(45)授权公告日 2017.10.24

(21)申请号 201380071708.X
(22)申请日 2013.01.29
(65)同一申请的已公布的文献号
申请公布号 CN 104956362 A
(43)申请公布日 2015.09.30
(85)PCT国际申请进入国家阶段日
2015.07.29
(86)PCT国际申请的申请数据
PCT/US2013/023655 2013.01.29
(87)PCT国际申请的公布数据
WO2014/120128 EN 2014.08.07
(73)专利权人 慧与发展有限责任合伙企业
地址 美国德克萨斯州
(72)发明人 肖恩·摩根·森普森
基里尔·缅杰列维
戴维·斯科特·蒂勒
(74)专利代理机构 北京德琦知识产权代理有限公司 11018
代理人 于会玲 康泉
(51)Int.Cl.
G06F 17/00(2006.01)
G06F 17/21(2006.01)
G06F 9/44(2006.01)
(56)对比文件
EP 1235144 A2,2002.08.28,
US 2012023487 A1,2012.01.26,
US 2004111727 A1,2004.06.10,
US 2011015917 A1,2011.01.20,
US 2011159179 A1,2011.06.30,
CN 101689200 A,2010.03.31,
CN 101529362 A,2009.09.09,
CN 101742031 A,2010.06.16,
CN 101808093 A,2010.08.18,
审查员 王高云

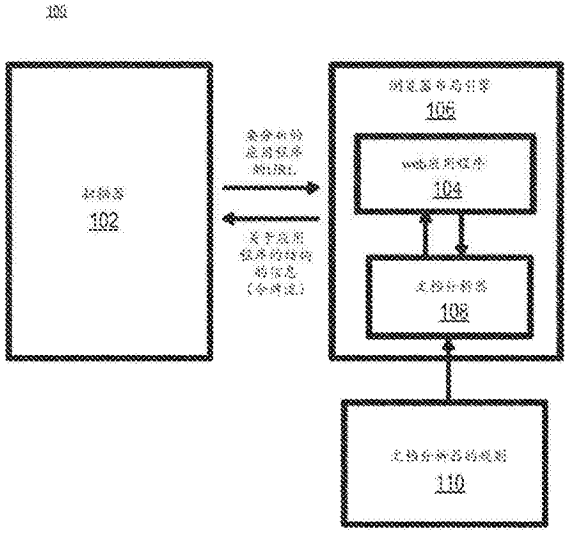
权利要求书2页 说明书6页 附图3页

(54)发明名称

分析web应用程序的结构

(57)摘要

本发明公开的示例实施例涉及分析web应用程序。web应用程序被加载。在该web应用程序的用户界面元素上模拟用户动作。根据规则遍历该web应用程序的结构以确定一组可操作的令牌。所述各可操作的令牌包括该web应用程序的、能够改变基于该web应用程序而被呈现的用户界面的部分。



CN 104956362 B

1.一种用于分析web应用程序的结构的系统，包括：

浏览器布局引擎，用于加载web应用程序；

扫描器，用于在该web应用程序的用户界面元素上模拟用户动作；和

文档分析器，用于根据规则遍历该web应用程序的结构和将该web应用程序的复杂文档对象模型转换为一组可操作的令牌，

其中所述模拟被用来帮助所述文档分析器确定要激活所述规则来产生可操作的令牌，

其中各个可操作的令牌包括该web应用程序的、能够改变基于该web应用程序而被呈现的用户界面的部分。

2.根据权利要求1所述的系统，其中该部分包括下列中至少一个的有效目标：键盘事件、点击事件和JavaScript对象。

3.根据权利要求1所述的系统，其中所述规则包括至少一个选择器，并且该至少一个选择器返回所述可操作的令牌中的至少一个可操作的令牌。

4.根据权利要求3所述的系统，其中该文档分析器根据所述可操作的令牌中的一个可操作的令牌确定一组选择器是相关的，并且根据该相关性确定启用该组选择器，并且其中该相关性确定至少部分地基于该浏览器布局引擎的缓存。

5.根据权利要求1所述的系统，其中所述各个可操作的令牌包括定位器和一组允许动作。

6.根据权利要求5所述的系统，其中该扫描器或其它扫描器包括用于消耗所述可操作的令牌的事件处理程序，其中该扫描器或该其它扫描器使用所述可操作的令牌确定要在该web应用程序上执行的一组测试。

7.根据权利要求6所述的系统，其中该扫描器或该其它扫描器通过为所述可操作的令牌中的每一个可操作的令牌都执行根据各个定位器定位的测试和在该定位器处的相应的允许动作，来根据所述可操作的令牌在该web应用程序中执行所述测试。

8.根据权利要求5所述的系统，其中该文档分析器确定特定类型的代码在该web应用程序中可执行，并且其中该文档分析器根据该特定类型确定所述允许动作中的至少一个允许动作。

9.一种非暂时性机器可读存储介质，该存储介质用于存储指令，所述指令如果被计算系统的至少一个处理器执行，致使该计算系统：

加载web应用程序；

在该web应用程序的用户界面元素上模拟用户动作；和

根据规则和所述用户动作来遍历web应用程序的结构，并分析该web应用程序的复杂文档对象模型DOM以确定一组可操作的令牌，

其中所述模拟被用来帮助确定要激活所述规则来产生可操作的令牌，

其中各个可操作的令牌包括该web应用程序的、能够改变基于该web应用程序而被呈现的用户界面的部分，并且

其中所述各个可操作的令牌包括定位器和一组允许动作。

10.根据权利要求9所述的非暂时性机器可读存储介质，进一步包括指令，所述指令如果由该至少一个处理器执行，致使该计算系统：

消耗所述可操作的令牌以确定要在该web应用程序中执行的一组测试；和

为所述可操作的令牌中的每一个可操作的令牌执行根据各个定位器定位的测试和与该定位器有关的相应的允许动作。

11.根据权利要求9所述的非暂时性机器可读存储介质，进一步包括指令，所述指令如果由该至少一个处理器执行，致使该计算系统：

根据所述可操作的令牌中的一个可操作的令牌确定一组选择器是相关的；

根据该相关性确定，启用该组选择器；和

根据所述选择器进一步分析该DOM，以产生另外多组所述可操作的令牌。

12.根据权利要求9所述的非暂时性机器可读存储介质，其中该部分包括JavaScript对象的有效目标。

13.一种用于分析web应用程序的结构的方法，包括：

加载web应用程序；

在该web应用程序的用户界面元素上模拟用户动作；和

根据规则来遍历该web应用程序的结构，并分析该web应用程序的复杂文档对象模型DOM以确定一组可操作的令牌，

其中所述模拟被用来帮助确定要激活所述规则来产生可操作的令牌，

其中各个可操作的令牌包括该web应用程序的、能够改变基于该web应用程序而被呈现的用户界面的部分，并且

其中所述可操作的令牌中的每一个可操作的令牌包括相应的定位器和相应的一组允许动作。

14.根据权利要求13所述的方法，进一步包括：

消耗所述可操作的令牌来为所述可操作的令牌中的每一个可操作的令牌确定要在该web应用程序中执行的一组测试，和

为所述可操作的令牌中的每一个可操作的令牌执行基于所述相应的定位器定位的相应一组测试和与所述相应的定位器有关的相应的允许动作。

15.根据权利要求13所述的方法，其中该部分包括键盘事件和点击事件中的至少一个事件的有效目标。

分析web应用程序的结构

背景技术

[0001] 使用软件安全测试来确定诸如web应用程序的应用程序中的漏洞。基于网络的软件工作的传统的黑盒安全测试通过利用伪装成攻击者的、通常被称为扫描器的安全测试应用程序，而发挥作用。扫描器通过发出HTTP请求并分析HTTP响应或缺乏HTTP响应来探究被测应用程序(AUT)，以便发现所有AUT接受输入的URL。AUT接受输入的URL可以被称为AUT的受攻击面。然后，扫描器根据受攻击面和可能的漏洞类别来创建攻击。扫描器应用程序攻击来通过评估程序的HTTP响应以诊断漏洞的存在或不存在。

附图说明

[0002] 下面的详细说明参照附图，附图中：

[0003] 图1为根据一个示例的用于分析web应用程序的结构的计算系统的框图；

[0004] 图2A和图2B为根据各种示例的规则和令牌的框图；

[0005] 图3为根据一个示例的用于分析web应用程序的结构的方法的流程图；和

[0006] 图4为根据一个示例的能够分析web应用程序的结构的装置的框图。

具体实施方式

[0007] 本发明描述的实施例提供用于对诸如web应用程序的应用程序进行测试的技术。当公司想要知道公司所拥有的生产中的或将要投产的web应用程序有多安全时，公司经常采用诸如渗透测试解决方案(例如，使用扫描器)、模糊测试、漏洞测试、软件安全测试、网站安全测试、它们的组合等的安全测试解决方案。公司可能希望使用生产中的应用程序的副本作为被测应用程序(AUT)。

[0008] 自动动态web应用程序安全扫描器在攻击AUT之前，先探究AUT。这一过程可以被称为“爬行”。爬行AUT可以通过分析web应用程序的超文本标记语言(HTML)和在诸如嵌入到web应用程序扫描器中的网络浏览器布局引擎的网络浏览器布局引擎的受控环境中执行AUT的代码来完成。

[0009] 由于使用Web 2.0应用程序的增长，它们复杂的结构妨碍了HTML页面分析并使得浏览器级别的处理变得复杂。许多Web2.0应用程序具有最小引导的HTML代码，最小引导的HTML代码下载主代码，例如，直接运行于浏览器文档对象模型(DOM)和JavaScript(JS)结构上的JS代码、异步JS和XML(AJAX)调用、动态创建的链接、DOM事件等等。有时，由于运行了JS代码，因此其它HTML文档不在应用程序的周期内取出。

[0010] 一种爬行这样严重地基于JS的应用程序的方法是允许在标准浏览器引擎中执行应用程序，然后通过把鼠标和键盘事件发送到Web应用程序的用户界面(UI)元素来模拟用户动作，征求对JS代码的评价，这可能会改变Web应用程序的状态并可能提供有关Web应用程序的攻击媒介的信息。但是，这种方法是非常耗时、不可靠的，而且容易“扫描失控”。一个“扫描失控”的示例是扫描在页面上发现的“日历”控件：尽管日历中所有日期都是扫描器生成事件的有效目标，但是对可能的日历日期的尝试点击会导致基本上是无限的扫描时间。

因此，高级JS框架可以广泛地利用“事件冒泡”来巩固事件处理程序，防止系统找到合适的事件目标，并且最复杂的部件(例如日历、时间表、树、表格等)可能呈现对于爬虫在可管理的时间中进行遍历来说，太多的可操作元素。

[0011] 此外，通过与所有是用户事件的潜在目标的UI部件交互，以进行穷尽的UI级的爬行是低效率的。应用程序内的某些UI元素，由于事件处理程序属于应用程序的结构内的其它元素(例如，自下而上检测或现代浏览器内的DOM事件的冒泡)不能被识别为事件的有效目标。进一步地，通过发送鼠标和键盘事件的应用程序的DOM元素的完全饱和是很耗时的，可能会破坏应用程序逻辑，因此是行不通的。此外，缺少关于应用程序的逻辑的认识不允许与复杂UI控件的正确交互。如此，这些挑战阻止了通过基于UI的爬行而正确发现应用程序面，这大大削弱了自动动态web应用程序扫描的质量。

[0012] 因此，本发明中公开的各种实施例涉及遵照一套区别处理以具体方式写成的应用程序的预定义规则，简化应用程序的结构以列出(例如，图或树)可操作的元件。利用这种方法，可以提供对附加的框架(例如，jQuery、Dojo等)的支持，从而实现对应用程序的更高质量的扫描并发现先前被隐藏的应用程序状态中的漏洞。

[0013] 该方案的优点包括创建应用程序的结构的简化但是准确的表示。使用用于描述对特定类型的应用程序和具体的JS框架的分析过程的良好定义的规则语言来实现对应用程序的高覆盖。

[0014] 为了发现应用程序受攻击面，web应用程序扫描器在web应用程序的UI元素上模拟用户动作。DOM分析器利用图2A中进一步详细说明的规则来遍历应用程序的结构，并把复杂的DOM转换为图2B中进一步详细说明的简明列表可操作的令牌。在某些示例中，令牌表示为鼠标或键盘事件的有效目标的DOM元素，或可能被调用、评价、或以任何其它方式修改的JS实体。

[0015] 每条规则可以包括用于检查该规则是否应该在web应用程序的具体框架上激活的规则相关属性，或由其组成。在一些示例中，规则可以是JS框架专用的，包括所使用的JS的版本。在一个示例中，如果规则是活跃的，那么该规则的所有选择器都被调用。然后，每个选择器都被用在对web应用程序的DOM和JS结构的遍历中，以返回令牌的列表。在另一示例中，如果规则是活跃的，那么可以根据对web应用程序的进一步分析(例如，根据web应用程序的缓存)来确定选择器。在这里详细说明上述进一步分析。

[0016] 在一些示例中，选择器是用来识别感兴趣的DOM中的元素的技术或机制。在一些示例中，选择器可以被内置到用于描述应用程序的语言中，例如层叠样式表(CSS)具有内置的选择器。在CSS的示例中，“.”可以用来选择类，而“#”可以用来选择id。利用这一方法，“.role”可以用来选择具有类“role”的DOM中的每一个元素。在其它示例中，选择器可以由规则产生。例如，规则可以编写JavaScript或其它脚本/代码以返回作为令牌的、一组选定的元素。

[0017] 在某些示例中，令牌表示为鼠标或键盘事件的有效目标的DOM元素，或可能被调用、评价、或以任何其它方式修改的JS实体。令牌包括可以被用来检索元件和DOM或JS实体的允许动作的列表的定位器(例如，定位器可以是简单的XPath、TruClient特定的定位器等)。

[0018] 可以识别有关的规则，然后可以启用相关的选择器。于是，可以利用选择器分析

web应用程序的DOM以产生令牌。这扩大了应用程序的抓取范围，并在应用程序中发现较短的业务流程路径。令牌可以被发送给扫描器以扫描web应用程序。在扫描过程中，令牌可以由web应用程序扫描器使用来定位有关DOM对象/JS实体并激活它们。这可以被用来确定漏洞和关于web应用程序的其它消息。

[0019] 图1为根据一个示例的用于分析web应用程序的结构的计算系统的框图。在此示例中，计算系统100可以包括用于在web应用程序104的用户界面元素上模拟用户动作的扫描器102。web应用程序104可以被加载到浏览器布局引擎106中，并由文档分析器108根据一组规则110进行分析。计算系统100的一个或多个部件可以利用至少一个处理器和存储器实现。进一步地，一个或多个计算机可以被用来实现部件中的每一个部件，或者部件中的每一个部件都可以被实现在单个计算系统中。

[0020] 扫描器102可以发送web应用程序104的定位器或标识符(例如，统一资源定位符(URL))给浏览器布局引擎106。浏览器布局引擎106可以加载web应用程序104，例如，根据URL加载web应用程序104。在一些示例中，浏览器布局引擎106可以是被配置来作为扫描器计算系统100的部分而起作用的、网络浏览器或修改的浏览器。web应用程序104的示例包括日历应用程序、电子邮件界面、新闻网页、诸如视频流的其它内容资源、生产应用程序等等。

[0021] 然后，扫描器102可以在web应用程序104的用户界面元素上模拟用户动作。模拟可以是随机的，或基于预定的组。模拟可以当文档分析器108遍历web应用程序104的结构时发生。模拟可以被用来帮助文档分析器108确定要激活一条或多条规则来产生可操作的令牌。如此，文档分析器108根据规则遍历web应用程序104的结构，以把web应用程序104的复杂DOM转换为一组可操作的令牌。各可操作的令牌可以包括web应用程序的、能够改变基于web应用程序104而被呈现的用户界面的部分。web应用程序的、能够改变用户界面的部分的示例可以包括键盘事件的目标、点击事件的目标、其它可操作的元素等等。在一些示例中，可操作的元素可以包括JavaScript对象。

[0022] 在一个示例中，文档分析器108可以根据模拟的动作确定web应用程序104中存在特定类型的框架。在一个示例中，框架是其中提供通用功能软件可以被用户代码选择性地改变，以提供应用程序特定的软件的抽象。这可以通过利用模拟的动作来确定存在与框架有关的选择器而发生。根据这一方法，可以确定多个框架。web框架的示例包括jQuery、YUI Lobrary、Dojo工具箱、MooTool、原型JS框架、Ajax、网络无障碍倡议(Web Accessibility Initiative，WAI)－无障碍丰富互联网应用程序(Accessible Rich Internet Applications，ARIA)和Flash。框架可以被进一步粒化成版本，因为版本的变化可能影响相关的功能和选择器。

[0023] 可以为每一个识别的框架启用一组选择器。在一个示例中，也可以启用一些默认的选择器，在其它示例中，当识别了框架时，所有有关的选择器都被启用。在又一示例中，当与框架有关的选择器被识别时，该框架被识别。

[0024] 此外，选择器可以被基于web应用程序104可以被搜索的内容而过滤。例如，知道jQuery存储关于选择器的信息的方式，文档分析器108就可以通过查询jQuery的缓存结构来获取选择器的列表。缓存结构可以根据jQuery的版本变化。

[0025] 在一个示例中，为了找到相关的选择器，文档分析器可以分析jQuery.cache JS代码[window.document[jQuery.expando]]。结果可能是包含为鼠标和/或键盘事件的有效目

标的DOM元素的选择器的JS对象的阵列。这一示例可以基于jQuery 1.6。在jQuery1.7中，内部结构发生了变化，因此可以编写更复杂的JS代码以获得相关元素。这一类型的代码可以具体地为具体类型的框架编写。在某些示例中，可以把jQuery称为框架，而1.6和1.7被称为框架的版本或类型。当发现特定的框架时，可以基于框架的具体细节分析web应用程序104以获得可操作的令牌。诸如处理web应用程序代码的其它方法可以被用来确定特定框架的存在。此外，可以实现文档分析器108的规则来自动地检测一个或多个框架并获取相关元素的列表。

[0026] 当分析web应用程序104时，越来越多的选择器被确定。如所指出的，规则可以被用来确定通用的，框架特定的，或实现特定的(例如，那些从缓存抽出的)的选择器。然后，文档分析器108可以使用符号化功能来尝试寻找被激活的选择器的作用和位置。利用选择器，文档分析器108和/或扫描器102可以在web应用程序104上查找令牌。如上所述，在某些示例中，令牌表示为鼠标或键盘事件的有效目标的DOM元素，或可能被调用、评价、或以任何其它方式修改的JS实体。可以为选择器中的每一个选择器找到一组令牌。在一些示例中，组可以是空的。令牌可以包括令牌的定位器和可以对/由令牌进行的一个或多个功能或动作。

[0027] 因此，当被实现时，规则可以激活至少一个选择器，至少一个选择器可以返回至少一个可操作的令牌。此外，各可操作的令牌可以包括相关的定位器和一组与各可操作的令牌有关的允许的动作。进一步地，当文档分析器108确定特定类型的代码在web应用程序中可执行时(例如，确定了框架)，文档分析器108可以根据特定类型的代码(例如，框架)确定允许的动作中的至少一个动作。例如，这可以通过启用相关的选择器、然后根据选择器尝试寻找令牌和元素，而发生。

[0028] 如所示出地，一组令牌可以被发送给扫描器102以扫描web应用程序104。在一个示例中，扫描器102可以是用于模拟用户动作的同一扫描器。在另一示例中，扫描器可以是另一扫描器。扫描器可以包括用于消耗可操作的令牌的事件处理程序。进一步地，扫描器可以使用可操作的令牌来确定要在web应用程序104中执行的一组测试。扫描器可以通过为可操作的令牌中的每一个可操作的令牌都执行根据各定位器定位的测试和在定位器处的相应的允许动作，来根据可操作的令牌在web应用程序中执行测试。

[0029] 适于检索和执行指令的处理器，例如中央处理单元(CPU)或微处理器，和/或电子电路可以被配置来执行本发明所述的任何部件的功能。在某些情况下，指令和/或其它信息，如令牌、web应用程序、规则等，可以被包括在存储器中。每个部件都可以包括，例如包括用于实现本发明所述的功能性的电子电路的硬件设备。除此之外或作为替代，每个部件都可被实现为被编码在机器可读存储介质上并且可由处理器执行的一系列指令。应该注意到，在一些实施例中，一些模块被实现为硬件设备，而其它模块则被实现为可执行指令。

[0030] 图2A和图2B为根据各种示例的规则和令牌的框图。图2A示出规则对象200。规则对象200可以是结构、类等等。每条规则都可以与相关属性202和选择器204相关联。选择器204可以包括一组一个或多个选择器206a－206n。选择器204、206a－206n检查DOM并返回可操作的DOM实体的序列。不是所有的选择器206a－206n都需要与单个规则相关联。相关属性202被用来激活或禁用规则。在一个示例中，当文档分析器正在web应用程序上执行时，来自扫描器的爬行可以被用来确定规则是否是相关的。在一些示例中，在web应用程序中使用的一个或多个框架可能留下指纹。这可以被用来确定哪些规则与应用程序有关。在其它示例

中，扫描器可以爬行应用程序并随机地选择应用程序中的对象。对象可以被分析以用来确定规则是否相关。选择器的示例包括CSS选择器206a、JavaScript选择器206b和jQuery选择器206n。在一个示例中，规则可以被与特定类型的jQuery版本相关联。如此，当规则被认为是相关的，可以检查jQuery缓存来决定要启用的选择器。

[0031] 图2B示出令牌对象250。令牌对象250可以是结构、类等等。令牌对象250可以包括一个或多个动作252，以及定位器254。令牌对象250可以定义应该是扫描器的动作的目标的DOM实体。可以通过各种机制，例如XPath260、属性262(例如，ARIA)、TruClient264等，进行位置确定。操作或动作可能根据DOM元素而不同。例如，可操作的令牌可以被与鼠标事件256、键盘事件258或其它用户界面改变事件(例如，执行JS代码)相关联。

[0032] 图3为根据一个示例的用于分析web应用程序的结构的方法的流程图。尽管下面参照计算系统100描述方法300的执行，但是可以使用执行方法300的其它合适的部件(例如，计算设备400)。此外，用于执行方法300的部件可以在多个设备间被展开。方法300可以被实现为被存储在诸如存储介质420的机器可读存储介质上的可执行指令和/或电子电路的形式。

[0033] 方法300可以在302，以web应用程序104被加载到浏览器布局引擎106中，而开始。在304，扫描器或其它设备/模块可以在web应用程序104的用户界面元素上模拟用户的动作。在一些示例中，用户界面元素是能够交互和/或定义界面的外观的结构。用户界面元素的示例包括窗口、菜单、图标、域、控件、标签、光标、指针等。

[0034] 在306，文档分析器108可以根据规则遍历web应用程序104的结构，并分析web应用程序104的复杂DOM以确定一组可操作的令牌。如上所述，各可操作的令牌可以包括web应用程序104的、能够改变基于web应用程序104而被呈现的用户界面的部分。该部分可以包括键盘事件、点击事件、可能被调用、评价或以任何其它方式修改的JS实体、或其组合的有效的目标。进一步地，可操作的令牌中的每一个可操作的令牌都可以包括各自的定位器和各自的一组允许动作。

[0035] 如上所述，文档分析器108可以通过确定什么规则应该被启用和根据启用的规则确定选择器，来确定可操作的令牌。然后，文档分析器108可以使用选择器来确定令牌。

[0036] 在某些示例中，令牌然后可以被提供给能够消耗令牌的扫描器。扫描器消耗可操作的令牌来为可操作的令牌中的每一个可操作的令牌确定要在web应用程序中执行的一组测试。然后，扫描器为可操作的令牌中的每一个可操作的令牌执行基于各定位器定位的各组测试和与各定位器有关的相应的允许动作。如此，当进行测试时，扫描器不需要消耗额外的时间来尝试与各选择器不兼容的动作。

[0037] 图4为根据一个示例的能够分析web应用程序的结构的装置的框图。计算设备400包括例如处理器410和包括用于分析web应用程序的结构的指令422、424、426的机器可读存储介质420。计算设备400可以是例如笔记本计算机、平板计算设备、服务器、工作站、台式计算机、或任何其它计算设备。

[0038] 处理器410可以是至少一个中央处理单元(CPU)、至少一个基于半导体的微处理器、至少一个图形处理单元(GPU)、其它适于检索和执行存储在机器可读存储媒体420中的指令的硬件设备，或者其组合。例如，处理器410可以包括芯片上的多个核，跨多个芯片的多个核，跨多个设备(例如，如果计算设备400包括多个节点设备)的多个核、或其组合。处理器

410可以获取、译码和执行指令422、424、426来实施分析DOM结构，例如，如在方法300中实现的那样实施分析DOM结构。作为检索和执行指令的替代或者附加于检索和执行指令，处理器310可以包括：包括用于执行指令422、424和426的功能的若干电子部件的至少一个集成电路(IC)、其它控制逻辑、其它电子电路、或其组合。

[0039] 机器可读存储媒体420可以是任何包含或存储可执行指令的电子的、磁的、光的或其它物理存储设备。因此，机器可读存储介质可以是，例如随机存取存储器(RAM)、电可擦除可编程只读存储器(EEPROM)、存储驱动器、紧致盘只读存储器(CD-ROM)等等。如此，机器可读存储介质可以是非暂时的。如在本发明中详细说明地，机器可读存储介质320可以以用于分析DOM结构的一系列可执行指令被编码。

[0040] 网页指令422可以由处理器410执行，以把web应用程序加载到浏览器布局引擎中。然后，扫描指令424可以由处理器410执行，来在web应用程序的用户界面元素上模拟用户动作。可以与爬行网页类似地，来完成用户动作，并且用户动作可以是随机的或基于算法。

[0041] 在模拟用户动作的过程中，分析器指令426可以被执行来确定与web应用程序有关的一个或多个选择器。分析器可以根据规则和模拟的用户动作来遍历web应用程序的结构，并分析web应用程序的复杂DOM以确定一组可操作的令牌。各可操作的令牌可以包括该web应用程序的、能够改变基于该web应用程序而被呈现的用户界面的部分。该部分的示例可以包括键盘事件、点击事件、JS事件、或其组合的目标。

[0042] 进一步地，各可操作的令牌包括定位器和一组允许动作。定位器可以基于各种方法(例如，XPath、Attribute、TruClient等)中的一个方法。允许动作可以基于所用框架的类型和/或与令牌有关的对象的类型/作用。

[0043] 一旦产生了令牌，那么令牌可以被发送给扫描器。扫描指令424可以被执行来消耗可操作的令牌来确定要在web应用程序中执行的一组测试。扫描器可以为可操作的令牌中的每一个可操作的令牌执行根据各定位器定位的测试和根据与定位器有关的相应的允许动作的可操作。

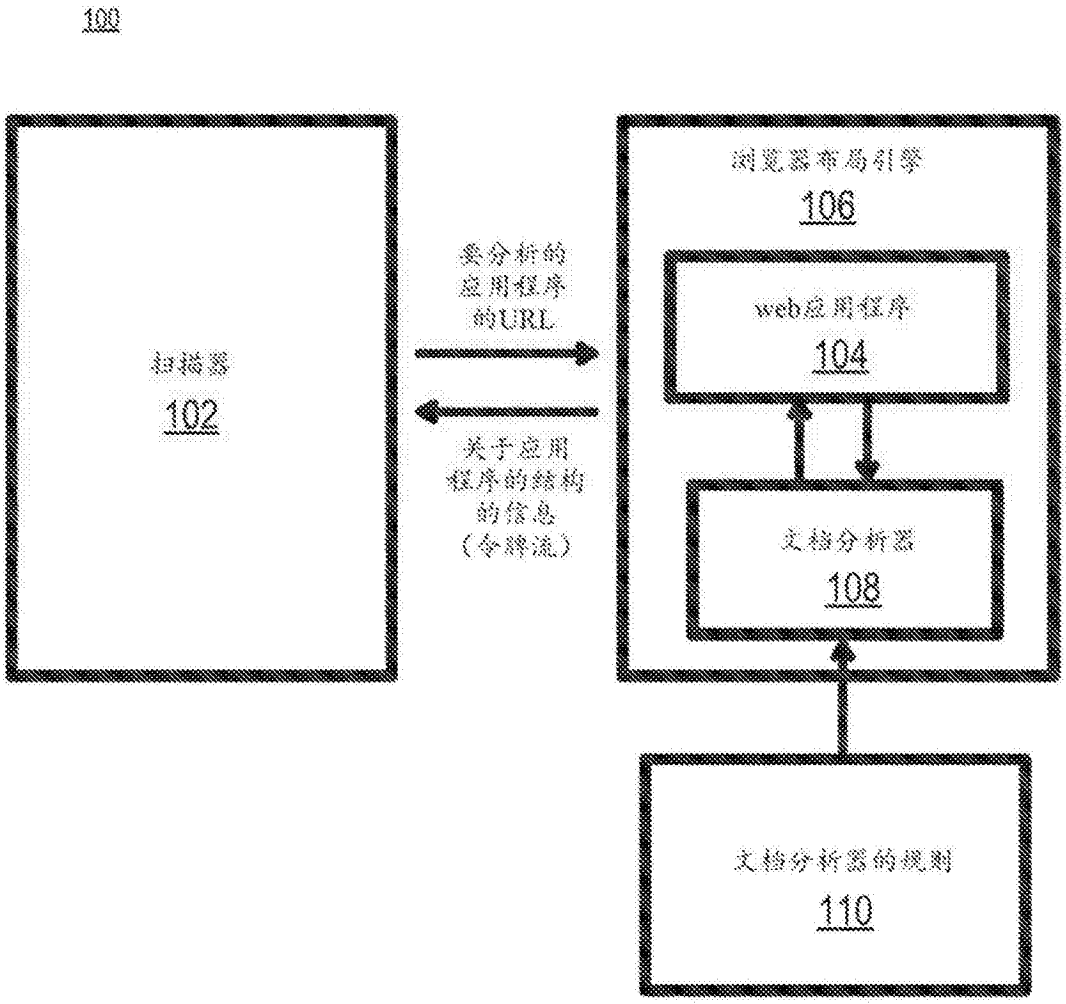

100
扫描器
102
要分析的
应用程序
的URL
关于应用
程序的结构
的信息
（令牌流）
浏览器布局引擎
106
web应用程序
104
文档分析器
108
文档分析器的规则
110

图1

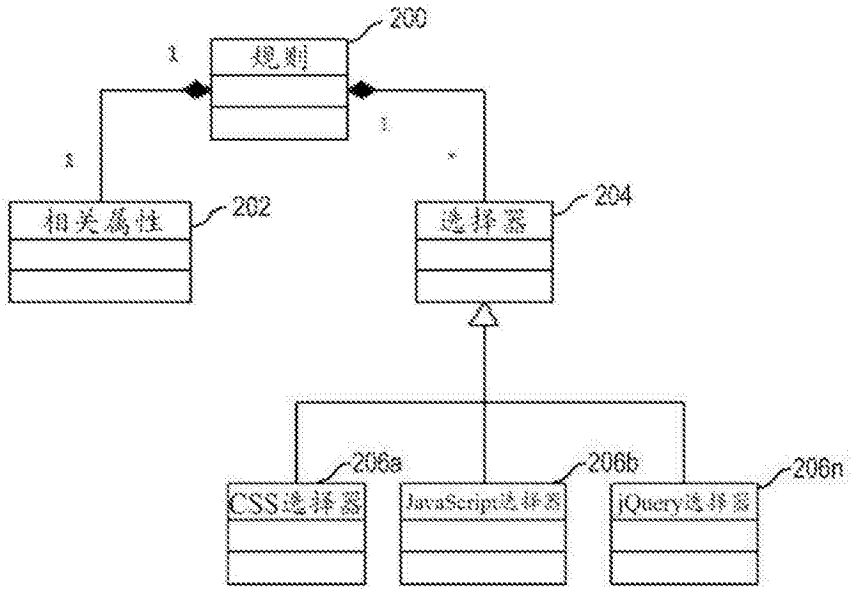

200
规则
1
1
1
*
相关属性
202
选择器
204
206a
CSS选择器
206b
JavaScript选择器
206n
jQuery选择器

图2A

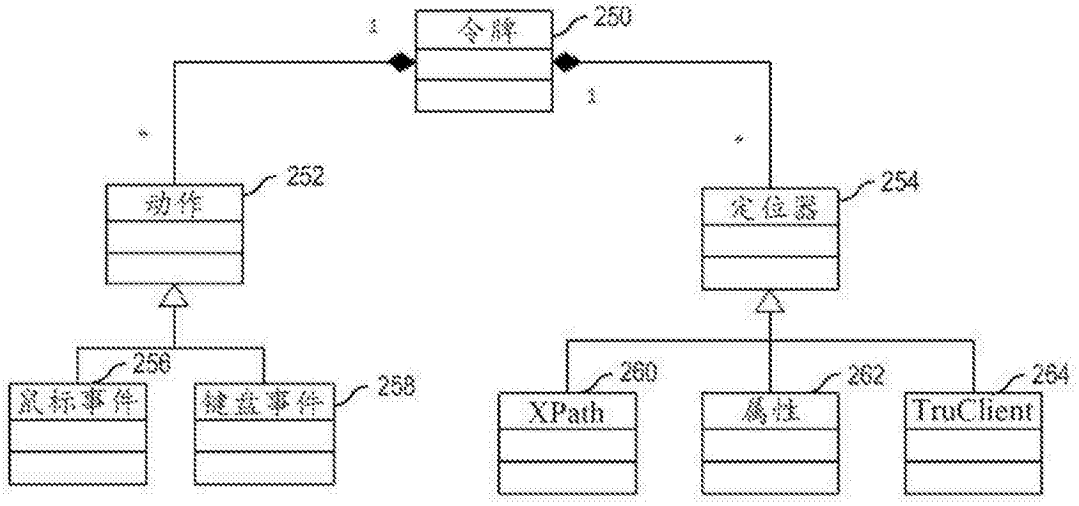

令牌
250
动作
252
定位器
254
鼠标事件
256
键盘事件
258
XPath
260
属性
262
TruClient
264

图2B

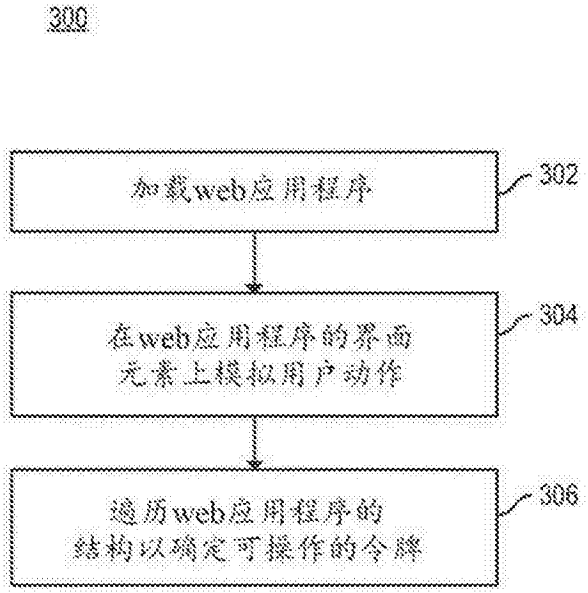

300
加载web应用程序
302
在web应用程序的界面
元素上模拟用户动作
304
遍历web应用程序的
结构以确定可操作的令牌
306

图3

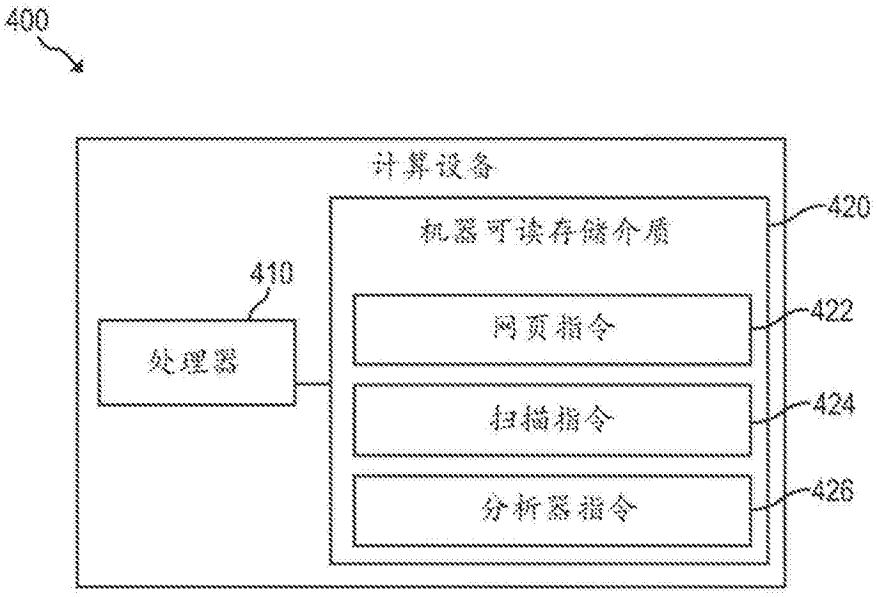


图4