(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 106844191 A
(43)申请公布日 2017.06.13

(21)申请号 201610141869.7

(22)申请日 2016.03.14

(30)优先权数据
2015-237213 2015.12.04 JP

(71)申请人 株式会社日立制作所
地址 日本东京都

(72)发明人 是木玄太 前冈淳 中村秀树

(74)专利代理机构 北京银龙知识产权代理有限公司 11243
代理人 范胜杰 王立杰

(51)Int.Cl.
G06F 11/36(2006.01)

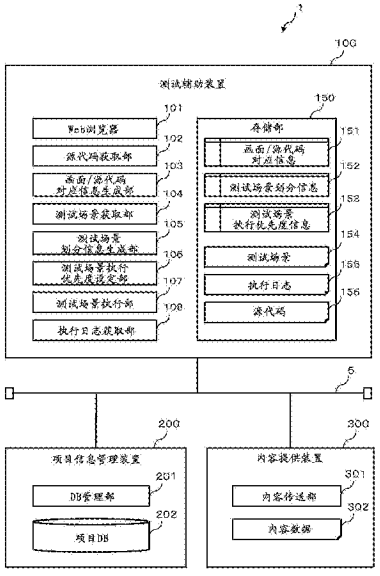
权利要求书2页 说明书10页 附图13页

(54)发明名称

测试辅助装置以及测试辅助方法

(57)摘要

本发明提供一种测试辅助装置以及测试辅助方法，高效地进行软件的测试。测试辅助装置(100)存储包含与画面的迁移有关的描述的多个测试场景(154)、构成软件的源代码(156)以及将画面和与画面的生成有关的源代码对应起来的信息即画面/源代码对应信息(151)，生成包含将测试场景(154)的描述划分为以画面为单位的多个块的结果的信息即测试场景划分信息(152)，并根据生成的测试场景划分信息(152)对多个测试场景(154)中的每一个场景设定执行优先度，根据执行优先度来依次执行测试场景(154)。测试辅助装置(100)可按照画面和每个视点，对多个测试场景中的每一个场景设定执行优先度。



CN 106844191 A

1.一种测试辅助装置,其是用于辅助软件测试的信息处理装置,其特征在于,

所述测试辅助装置具备:

存储部,其存储包含与画面的迁移有关的描述的多个测试场景、构成所述软件的源代码、以及将所述画面和与所述画面的生成有关的所述源代码对应起来的信息即画面/源代码对应信息;以及

测试场景划分部,其生成包含将所述测试场景的描述划分为以所述画面为单位的多个块的结果的信息即测试场景划分信息;以及

测试场景执行优先度设定部,其根据所述测试场景划分信息对所述多个测试场景中的每一个场景设定执行优先度。

2.根据权利要求1所述的测试辅助装置,其特征在于,

所述测试场景执行优先度设定部按照各所述画面和每个视点,对所述多个测试场景中的每一个场景设定所述执行优先度。

3.根据权利要求2所述的测试辅助装置,其特征在于,

所述测试场景执行优先度设定部基于使较早开始特定的所述画面的测试的所述测试场景比其它所述测试场景优先执行的视点,对所述多个测试场景中的每一个场景设定所述执行优先度。

4.根据权利要求3所述的测试辅助装置,其特征在于,

所述存储部还存储所述多个测试场景中的每一个场景的执行日志,

所述测试场景执行优先度设定部针对所述测试场景中的每一个场景,从所述执行日志中获取用于确定所述特定画面的测试开始定时的信息,并根据获取到的所述定时对所述多个测试场景中的每一个场景设定所述执行优先度。

5.根据权利要求2所述的测试辅助装置,其特征在于,

所述存储部还存储所述多个测试场景中的每一个场景的执行日志,

所述测试场景执行优先度设定部针对所述测试场景中的每一个场景,从所述执行日志中获取与关于特定的所述画面的不良现象的发生频率相关的信息,并基于使关于所述特定的画面的不良现象的发生频率较高的所述测试场景比其它所述测试场景优先执行的所述视点,来设定所述执行优先度。

6.根据权利要求1～5中任一项所述的测试辅助装置,其特征在于,

所述存储部以文件的形式存储所述源代码,

所述测试辅助装置还具备:画面/源代码对应信息生成部,其从包含用于定义特定的所述画面的描述的所述源代码中,获取关于特定的所述画面的源代码的文件名,将使特定的所述画面和所述文件名对应起来的信息生成为所述画面/源代码对应信息。

7.根据权利要求1～5中任一项所述的测试辅助装置,其特征在于,

所述测试场景划分部根据所述测试场景中描述的产生画面迁移的描述来划分测试场景。

8.根据权利要求1～5中任一项所述的测试辅助装置,其特征在于,

所述测试辅助装置还具备:测试场景执行部,其根据所述执行优先度依次执行所述多个测试场景。

9.根据权利要求8所述的测试辅助装置,其特征在于,

所述测试场景执行部在所述特定的画面的所述块的执行完成的时刻停止所述测试场景的执行。

10.根据权利要求8所述的测试辅助装置，其特征在于，

在所述测试场景的所述画面迁移模式与其它所述测试场景的画面迁移模式全部或部分重叠时，所述测试场景执行部选择要执行的所述测试场景从而不重复执行所述模式。

11.一种测试辅助方法，其是用于辅助软件测试的方法，其特征在于，

使信息处理装置执行以下步骤：

存储包含与画面的迁移有关的描述的多个测试场景、构成所述软件的源代码以及将所述画面和与所述画面的生成有关的所述源代码对应起来的信息即画面/源代码对应信息的步骤；

生成包含将所述测试场景的描述划分为以所述画面为单位的多个块的结果的信息即测试场景划分信息的步骤；以及

根据所述测试场景划分信息对所述多个测试场景中的每一个场景设定执行优先度的步骤。

12.根据权利要求11所述的测试辅助方法，其特征在于，

所述信息处理装置根据各所述画面和每个视点，对所述多个测试场景中的每一个场景设定所述执行优先度。

13.根据权利要求12所述的测试辅助方法，其特征在于，

所述信息处理装置基于使较早开始特定的所述画面的测试的所述测试场景比其它所述测试场景优先执行的视点，对所述多个测试场景中的每一个场景设定所述执行优先度。

14.根据权利要求13所述的测试辅助方法，其特征在于，

所述信息处理装置进行如下处理：

还存储所述多个测试场景中的每一个场景的执行日志，

针对所述测试场景中的每一个场景，从所述执行日志中获取用于确定所述特定的画面的测试开始定时的信息，并根据获取到的所述定时对所述多个测试场景中的每一个场景设定所述执行优先度。

测试辅助装置以及测试辅助方法

技术领域

[0001] 本发明涉及一种测试辅助装置以及测试辅助方法。

背景技术

[0002] 专利文献1中记载了“在进行回归测试时，为了避免测试项目过于庞大，可以根据变更了的场所自动地优先提取应该测试的部分，从而使回归测试能够高效、自动地进行”，“构成一种回归系统，该系统将与过去进行的测试例有关的信息进行保存，并通过将该信息和对对象程序进行解析而得到的依存关系加以组合来进行测试”，“信息处理装置参照存储单元中存储的软件程序中的各个Statement的依存关系，提取出与变更后的Statement有关的Statement，并根据存储单元中存储的与过去所执行的测试例和软件程序中的各个Statement之间的对应有关的信息，来检测与提取出的Statement相关联的测试例，并对检测出的测试例予以执行”。

[0003] 专利文献1：日本特开2008-204405号公报

发明内容

[0004] 发明所要解决的问题

[0005] 在软件的开发实践中，为了提高质量和开发效率，以功能单位重复进行源代码的安装和测试的开发形式已趋于稳定。此外，Web应用程序等应用软件有时会通过多个画面来实现一个功能，这种软件的测试大多是通过执行将包含画面迁移操作(按钮或链接操作等)的用户操作描述为一系列序列的测试场景来进行的。

[0006] 然而，在使用了测试场景的测试中，获取画面与测试场景的描述部分之间的对应或画面与源代码之间的对应未必容易。因此，尤其在多数开发人员所从事的大规模信息处理系统的开发项目中，会在发现不良现象或者原因的确定上花费大量精力和时间，成为导致软件开发效率低下的重要原因。

[0007] 本发明是鉴于这一背景而提出的，其目的在于提供一种能够高效地进行软件测试的测试辅助装置、测试辅助方法以及程序。

[0008] 用于解决问题的手段

[0009] 用于实现上述目的的本发明之一为辅助软件测试的测试辅助装置(信息处理装置)，其具备：存储部，其存储包含与画面的迁移有关的描述的多个测试场景、构成所述软件的源代码以及将所述画面和与所述画面的生成相关的所述源代码对应起来的信息即画面/源代码对应信息；测试场景划分部，其生成测试场景划分信息，该测试场景划分信息为包含将所述测试场景的描述划分为以所述画面为单位的多个块的结果的信息；以及测试场景执行优先度设定部，其根据所述测试场景划分信息对所述多个测试场景中的每一个场景设定执行优先度。

[0010] 发明效果

[0011] 根据本发明，能够高效地进行软件的测试。

附图说明

[0012] 图1是信息处理系统1的结构及功能示意图。

[0013] 图2是作为项目信息管理装置200以及内容提供装置300的硬件的一个例子的信息处理装置10的框图。

[0014] 图3是画面/源代码对应信息151的一个例子。

[0015] 图4是与图3中画面/源代码对应信息151相对应的画面布局。

[0016] 图5是测试场景划分信息152的一个例子。

[0017] 图6是画面跳数(hop)示意图。

[0018] 图7是测试场景执行优先度信息153的一个例子。

[0019] 图8是说明测试场景执行处理S800的流程图。

[0020] 图9是用于说明图8中画面/源代码对应信息生成处理S811的细节的流程图。

[0021] 图10的(a)是画面定义用脚本的一个例子、(b)是(a)中画面定义用脚本所描述的文件名为“top.html”的源文件(HTML文件)的描述例、(c)是文件名为“top.js”的脚本文件的描述例。

[0022] 图11是用于说明图8中测试场景划分信息生成处理S812的细节的流程图。

[0023] 图12的(a)是测试场景154的一个例子、(b)是与(a)中的测试场景154相关联的画面的例子。

[0024] 图13是图12的(b)中各画面的源代码的例子。

[0025] 图14是将图12的(a)所示的测试场景154的描述以画面单位进行划分后的结果的图。

[0026] 符号说明

[0027] 5通信网络、100测试辅助装置、101Web浏览器、102源代码获取部、103画面/源代码对应信息生成部、104测试场景获取部、105测试场景划分信息生成部、106测试场景执行优先度设定部、107测试场景执行部、108执行日志获取部、150存储部、151画面/源代码对应信息、152测试场景划分信息、153测试场景执行优先度信息、154测试场景、155执行日志、156源代码、200项目信息管理装置、201DB管理部、202项目DB、300内容提供装置、301内容传送部、302内容数据、S800测试场景执行处理、S811画面/源代码对应信息生成处理、S812测试场景划分信息生成处理

具体实施方式

[0028] 下面将参照附图对实施方式进行说明。在下述说明中,有时会将“数据库”简称为“DB”。

[0029] 图1表示作为实施方式而要说明的信息处理系统1的概要的结构。如图1所示,信息处理系统1包括测试辅助装置100、项目信息管理装置200以及内容提供装置300。这些均为信息处理装置(计算机),且彼此之间经由通信网络5可通信地连接。通信网络5可以是LAN(局域网:Local Area Network)、WAN(广域网:Wide Area Network)、因特网等。

[0030] 测试辅助装置100用于辅助用户所进行的软件测试。在本实施方式中,以成为测试辅助装置100的测试对象的软件是在因特网上提供Web服务的软件的情况为例进行说明。此

外，在下述说明中，假设的测试辅助装置100的用户例如为利用测试辅助装置100所输出的信息的用户或成为测试对象的软件的开发人员或维修人员等。

[0031] 例如当用户在执行包含使画面迁移的描述的多个测试场景的同时进行软件的测试时，测试辅助装置100通过控制测试场景的执行使得用户能够高效地进行测试。例如，测试辅助装置100在对用户正在开发中的指定源代码(例如，Web描述语言(HTML(HyperText Markup Language)，XHTML(eXtensible HyperText Markup Language)，XML(eXtensible Markup Language)等)、各种脚本(JavaScript(注册商标)脚本等)所描述的画面(以下称为开发画面)进行测试时，对多个测试场景中的每一个场景设定执行优先度，并以基于所设定的执行优先度的顺序执行各测试场景。例如，当成为测试对象的软件规模较大，测试场景较长的时候，从开始执行测试场景到执行用户所要验证的开发画面为止有时会需要一些时间，而通过设定上述执行优先度以使根据开发画面而较早到达的测试场景被优先执行，能够使执行开发画面的测试的时间提前，从而用户能够迅速地获取开发画面的测试结果。

[0032] 测试辅助装置100根据由用户指定的视点来设定上述执行优先度。例如，测试辅助装置100设定上述执行优先度以使开发画面的测试较早开始。此外，测试辅助装置100从过去执行测试场景时所获取的执行日志中获取与开发画面不良现象的发生频率有关的信息，并设定上述执行优先度以使不良现象的发生频率较高的测试场景优先执行。这样，由于测试辅助装置100根据由用户来指定的视点设定上述执行优先度，所以能够以适应用户需求的形式来进行开发画面的测试。另外，测试辅助装置100在设定上述执行优先度时，对测试场景的描述以画面单位来进行划分，从而能够以画面单位来高效地把握测试场景的描述。

[0033] 图2是作为实现测试辅助装置100、项目信息管理装置200以及内容提供装置300的硬件的一个例子的信息处理装置10(计算机)的框图。信息处理处理装置10例如可以是作为网络存储装置而发挥作用的装置。此外，虽然在图1中描述了测试辅助装置100、项目信息管理装置200以及内容提供装置300分别被构成为独立的硬件，但也可以将它们中的两个以上通过共同的硬件(平台)来实现。此外，还可以将它们中的一个以上用虚拟的信息处理装置(例如，通过云系统来提供的云服务等)来实现。

[0034] 如图2所示，信息处理装置10具备处理器11、主存储装置12、辅助存储装置13、输入装置14、输出装置15以及通信装置16。它们彼此之间通过未图示的总线可通信地连接。

[0035] 处理器11使用例如CPU(Central Processing Unit)或MPU(Micro Processing Unit)来构成。处理器11通过读取并执行主存储装置12中所存储的程序，来实现信息处理装置10的各种功能。主存储装置12是用来存储程序或数据的装置，其可以是ROM(Read Only Memory)、RAM(Random Access Memory)、NVRAM(Non Volatile RAM)等。辅助存储装置13可以是硬盘驱动器、SSD(Solid State Drive)、光学式存储装置、记录介质的读取/写入装置等。辅助存储装置13中存储的程序或数据随时被加载到主存储装置12中。

[0036] 输入装置14可以是键盘、鼠标、触摸板等。输出装置15例如可以是液晶监视器、LCD(Liquid Crystal Display)、图形卡、扬声器等。通信装置16是经由通信网络5与其它装置进行通信的通信接口，例如可以是NIC(Network Interface Card)、无线通信模块等。输入装置14或输出装置15并不一定是必要的构成要素，信息处理装置10也可以构成为通过通信装置16来进行与其它装置之间的信息的输入或输出。

[0037] 接下来，结合图1对测试辅助装置100所具备的功能以及测试辅助装置100所管理

的信息(数据)进行说明。如图1所示,测试辅助装置100具备Web浏览器101、源代码获取部102、画面/源代码对应信息生成部103、测试场景获取部104、测试场景划分信息生成部105、测试场景执行优先度设定部106、测试场景执行部107、执行日志获取部108以及存储部150。这些功能通过测试辅助装置100的处理器11读取并执行主存储装置12中存储的程序来实现。这些功能的全部或一部分也可以作为硬件来实现。在测试辅助装置100中,除了这些功能以外,进行各功能的起动/管理或基于文件系统的数据管理等的操作系统(Operating System)或控制程序等软件也可以工作。

[0038] 如图1所示,存储部150对画面/源代码对应信息151、测试场景划分信息152、测试场景执行优先度信息153、测试场景154、执行日志155以及源代码156进行存储。

[0039] 在图1所示功能中,Web浏览器101可根据测试场景154来读取源代码156、进行记载有基于源代码156的内容的画面(Web网页)的生成或显示、内容的播放等处理。在进行上述处理时,Web浏览器101向内容提供装置300发送适当内容的获取请求并从内容提供装置300中获取内容数据302。

[0040] 源代码获取部102经由通信网络5从项目信息管理装置200中获取适当源代码156。源代码156例如可以是Web描述语言或各种脚本所描述的数据。测试辅助装置100将这些数据例如以由文件系统进行处理的文件为单位来管理。

[0041] 画面/源代码对应信息生成部103根据存储部150所存储的源代码156(源文件),生成画面/源代码对应信息151。有关画面/源代码对应信息151的细节将在后文中进行阐述。

[0042] 测试场景获取部104经由通信网络5从项目信息管理装置200中获取适当测试场景154。测试场景154例如包含用自然语言描述的多个步骤(行)。测试场景154根据预定的业务场景进行Web网页的操作(画面)并产生画面的迁移。测试场景154包括表示对用来产生画面迁移的描述或画面进行怎样的测试的描述。

[0043] 测试场景划分信息生成部105将测试场景154划分为以根据源代码156生成的画面为单位的多个块,并生成记载有划分结果的测试场景划分信息152。有关测试场景划分信息152的细节将在后文中进行阐述。

[0044] 测试场景执行优先度设定部106对测试场景获取部104所获取的多个测试场景154中的每一个场景设定执行优先度,并将设定的结果存储为测试场景执行优先度信息153。有关执行优先度的具体设定方法以及测试场景执行优先度信息153将在后文中进行详细阐述。

[0045] 测试场景执行部107根据测试场景154所设定的执行优先度依次执行各测试场景154。在执行测试场景154时,测试场景执行部107例如可以控制Web浏览器101,来自动执行基于测试场景154的操作输入的步骤。作为实现测试场景执行部107的功能的软件,例如可以是“Cucumber(注册商标)”、“Turnip(注册商标)”等。测试场景执行部107例如可以是与在内容提供装置300中动作的Web服务器进行协同动作的单元。

[0046] 执行日志获取部108获取在执行测试场景154时测试场景执行部107所生成的信息(各步骤的执行开始/结束时间、有无错误产生、确定错误产生场所的信息、产生的错误内容、执行履历等),并将获取到的信息存储为执行日志155。

[0047] 项目信息管理装置200将与用户所从事的项目(开发、维修、设计等项目)有关的信息进行综合累积/管理。项目信息管理装置200具有管理与项目相关信息的项目DB202和管

理项目DB202的DB管理部201。DB管理部201例如可具有DBMS(DataBase Management System)或分散型版本管理系统等功能。项目信息管理装置200例如在项目DB202中管理将主数据(规格书、设计书、模型、图解、信息操作规则等)与被作为一个存储器来管理的软件或数据项目对应起来的信息。

[0048] 内容提供装置300具有存储内容数据302并对内容数据302进行传送的内容传送部301。内容数据302例如是用预定数据格式来描述的数据(文字数据、音频数据、图像数据、视频数据等)。内容提供装置300在接收到前述内容的获取请求时,向测试辅助装置100发送相应的内容数据302。

[0049] 此外,测试场景154、执行日志155、源代码156、项目DB202以及内容数据302并非一定要以上文所述方式来进行配置,这些各数据的配置方式均可以根据信息处理系统1的构成来进行适当变更。

[0050] <画面/源代码对应信息>

[0051] 在图3表示画面/源代码对应信息151的一个例子,图4表示与图3的画面/源代码对应信息151相对应的画面布局。如图3所示,画面/源代码对应信息151包含表示画面ID1511和源代码ID1512之间的对应的信息。在该例中,将用户用作画面识别符的画面名作为画面ID1511,另外,将源代码所描述的文件的文件名作为源代码ID1512来使用。另外,画面ID1511和源代码ID1512之间的对应根据项目的形式、信息处理系统1的状态等而多种多样。此外,还有如图4的“样本画面”那样,一个画面由多个源代码(源文件)构成的情况,画面ID1511和源代码ID1512之间的对应并不一定是一对一的关系。

[0052] 如图3～图4所示,在这一示例中,作为与画面ID1511“样本画面”相对应的源文件(源代码),存在有“sample.html”、“sample.js”、“menu.html”、“sidebar.html”、“contents.html”以及“contents.js”。此外,如图4所示,在“样本画面”中,在“sample.html”的内部使用了“menu.html”、“sidebar.html”以及“contents.html”。在这个示例中,“sample.html”所描述的单个或多个画面部件的行为在“sample.js”中描述,而“contents.html”所描述的单个或多个画面部件的行为在“contents.js”中描述。

[0053] 另外,作为与画面ID1511“顶部画面”相对应的源文件(源代码),存在有“top.html”和“top.js”。“top.html”所描述的单个或多个画面部件的行为在“top.js”中描述。

[0054] 此外,作为与画面ID1511“确认画面”相对应的源文件(源代码),存在有“confirm.html”和“confirm.js”。“confirm.html”所描述的单个或多个画面部件的表现在“confirm.js”中描述。

[0055] 此外,作为与画面ID1511“门户画面”相对应的源文件,存在有“portal.html”。“portal.html”所描述的单个或多个画面部件无需定义其行为,不存在脚本文件。

[0056] <测试场景划分信息>

[0057] 图5表示测试场景划分信息152的一个例子。在图5中,测试场景ID1521中设定有对每一个测试场景154赋予的识别符即测试场景ID。此外,在画面跳数1522中设定有表示将测试场景154的描述以画面单位进行划分而得到的各个块在该测试场景154中位于第几个的信息。在图5中,上述各个块通过对每个画面跳数“1”、“2”、“3”……设置的方框中所记载的画面名来确定。

[0058] 图5的测试场景划分信息152表示测试场景ID为“1”的测试场景154的描述被划分为三个块，即，关于“顶部画面”的块、关于“确认画面”的块以及关于“门户画面”的块。此外，该测试场景划分信息152还表示了在执行测试场景ID为“1”的测试场景154时，首先进行“顶部画面”的测试，然后在画面由“顶部画面”迁移到“确认画面”后进行“确认画面”的测试，最后在画面由“确认画面”迁移为“门户画面”后进行“门户画面”的测试。

[0059] 下面结合图6对画面跳数进行具体说明。在图6中，空心箭头表示画面的迁移。在该例的情况下，在测试场景(1)中，按照A画面→B画面→C画面的顺序执行画面测试，在测试场景(2)中，按照B画面→C画面的顺序执行画面测试，在测试场景(3)中，按照E画面→F画面→G画面→A画面→B画面的顺序执行画面测试，测试场景(4)中，按照F画面→G画面的顺序执行画面测试。

[0060] 画面跳数表示在根据测试场景154进行测试时，对成为测试对象的画面从测试开始时数起的顺序。例如，在图6中，在测试场景(1)中，B画面的画面跳数为“2”，测试场景(2)中，B画面的画面跳数为“1”，测试场景(3)中，B画面的画面跳数为“5”。此外，在测试场景(4)中，B画面并未成为测试对象，在这种情况下画面跳数为不明确(无)。

[0061] 另外，当成为测试对象的画面为B画面，并且以画面跳数为视点设定各测试场景的执行优先度时，在图6中，按照测试场景(2)→测试场景(1)→测试场景(3)的顺序来设定执行优先度。

[0062] ＜测试场景执行优先度信息＞

[0063] 在测试场景执行优先度信息153中，设定有由测试场景执行优先度设定部106设定的、按照画面类别且每个视点的各测试场景154的执行优先度。

[0064] 图7表示测试场景执行优先度信息153的一个例子。测试场景执行优先度信息153包括表示根据画面类别且每个视点设定的测试场景154的执行优先度的信息。图7是对“顶部画面”基于3个视点(“画面跳数”、“到达时间”以及“不良现象发生次数”)中的每一个视点设定执行优先度的情况。

[0065] 在图7中，以“画面跳数”为视点的执行优先度中设定有“顶部画面”的画面跳数。

[0066] 此外，在图7中，以“到达时间”为视点的执行优先度由测试场景执行优先度设定部106根据过去执行各测试场景154时所获取的执行日志155来设定。例如，测试场景执行优先度设定部106设定以“到达时间”为视点的执行优先度，以使从执行日志155掌握的到达该画面为止的时间(例如，从开始执行测试场景154到开始执行该画面的块为止的时间(开始时间))越短执行优先度越高。

[0067] 此外，在图7中，以“不良现象发生次数”为视点的执行优先度由测试场景执行优先度设定部106根据过去执行各测试场景154时所获取的执行日志155来设定。例如，测试场景执行优先度设定部106设定以“不良现象发生次数”为视点的执行优先度，以使从执行日志155掌握的该画面的不良现象发生次数越多执行优先度越高。

[0068] 在此，例如在希望尽早开始特定开发画面的测试时，用户将“画面跳数”或“到达时间”选择为在设定测试场景154的执行优先度时的视点。此外，例如希望尽早发现特定开发画面的不良现象时，用户选择“不良现象发生次数”为在设定测试场景154的执行优先度时的视点。

[0069] 以上列举的视点仅为一个例子，也可以设定除此之外的其他视点。另外，例如，也

可以通过对从多个视点中的每一个视点求出的执行优先顺序进行综合判断来设定测试场景154的执行优先度(例如,当基于两个视点中每一个视点的执行优先度反转时,优先采用基于其中任一个视点的执行优先度等)。此外,也可以将不包含与成为对象的画面有关的描述的测试场景154从执行对象中排除。这种情况下,例如在测试场景执行优先度信息153的相应测试场景栏中设定“—”等预定符号。

[0070] ＜处理说明＞

[0071] 接下来,对在上述结构构成的信息处理系统1中所进行的处理进行说明。

[0072] 图8是说明对软件显示的画面进行测试时由测试辅助装置100进行的处理(以下称为测试场景执行处理S800)的流程图。

[0073] 首先,测试辅助装置100进行生成画面/源代码对应信息151的处理(以下称为画面/源代码对应信息生成处理S811)和生成测试场景划分信息152的处理(以下称为测试场景划分信息生成处理S812)(S811、S812)。有关这些处理的细节将在后文中进行阐述。

[0074] 接着,测试辅助装置100从用户接收用户ID(例如,与开发画面的开发有关的开发人员名等)以及视点的指定(S813)。用户可以指定单个或多个视点。

[0075] 然后,测试辅助装置100从项目信息管理装置200的项目DB202中获取与在S813中接收的用户ID相对应的源代码,并存储为源代码156(S814)。此外,虽然在本示例中没有进行描述,但也可以根据从用户接收的开发日期等信息将从项目DB202获取的源代码的范围缩小。

[0076] 接下来,测试辅助装置100确定存储有获取到的源代码的源文件(S815),根据画面/源代码对应信息151来确定与该源文件相对应的画面(以下称为开发画面)(S816)。

[0077] 而后,测试辅助装置100参照测试场景划分信息152,获取包含与在S816中确定的开发画面有关的描述的测试场景154(S817)。此外,在本示例中,假设测试辅助装置100获取多个测试场景154。

[0078] 接着,测试辅助装置100根据在S813中从用户接收的视点,对在S817中获取的多个测试场景154中的每一个场景设定执行优先度,并将其结果存储为测试场景执行优先度信息153(S818)。

[0079] 然后,测试辅助装置100执行尚未执行的测试场景154中执行优先度最高的场景(S819)。

[0080] 然后,测试辅助装置100对与开发画面有关的测试是否成功执行进行判定(S820)。判定为是时(S820:是),继续进行S821的处理。判定为否时(S820:否),进行S822的处理,在测试辅助装置100进行了错误信息的输出等处理后,结束测试场景154的执行。另外,既可以这样在与开发画面有关的测试完成的阶段结束测试场景154的执行,也可以将测试场景一直执行到最后。如前者那样在与开发画面有关的测试完成的阶段结束测试场景154的执行时,用户可以尽早获取测试结果,能够缩短发现不良现象所需的时间。

[0081] 在S821中,测试辅助装置100判定在与开发画面有关的测试场景154中是否有尚未执行的测试场景154。判定为是时(S821:是),使处理返回S819,执行尚未执行的测试场景154。判定为否时(S821:否),测试辅助装置100结束测试场景执行处理S800。

[0082] 此外,在上述说明的测试场景执行处理S800中,当某一测试场景154的画面迁移模式与其它测试场景154的画面迁移模式全部或部分重叠时,为了提高测试效率,可以通过对

要执行的测试场景154进行选择来避免对重复模式进行重复执行。例如，在图6中，测试场景(1)的画面迁移模式为A画面→B画面→C画面，测试场景(2)的画面迁移模式为B画面→C画面，二者均包含B画面→C画面这一画面迁移模式，在这种情况下，测试辅助装置100可不执行测试场景(2)。

[0083] 图9是详细描述图8的画面/源代码对应信息生成处理S811的流程图。以下将结合该图9进行说明。

[0084] 首先，测试辅助装置100从所存储的源代码156(源文件)中获取进行了画面定义的源代码(以下称为画面定义用脚本文件)。例如，测试辅助装置100从源代码156(源文件)中获取描述有画面名与源文件名(例如，HTML文件名、HTML文件ID、脚本文件名、脚本ID等)之间的组的画面定义用脚本文件(S911)。

[0085] 接着，测试辅助装置100从在S911中获取的画面定义用脚本文件中获取与画面名相对应的源文件名(S912)，并将画面名与源文件名的组存储为画面/源代码对应信息151(S913)。

[0086] 图10的(a)表示画面定义用脚本的一个例子。该画面定义用脚本中，将“顶部画面”描述为画面名，另外将“top.html”描述为源文件名(HTML文件名)。此外，在该画面定义用脚本中，将“TopCtrl”描述为与“顶部画面”相对应的脚本ID。另外，这样在画面定义用脚本中描述脚本ID而不是脚本文件名的情况下，会在后续处理中根据脚本ID来确定脚本文件名。

[0087] 接着回到图9，测试辅助装置100对画面/源代码对应信息151所记载的源文件进行解析来获取脚本ID(S914)。

[0088] 图10的(b)表示(a)中的画面定义用脚本所描述的文件名为“top.html”的源文件(HTML文件)的描述例。在该例中，对“controller”属性设定“TopCtrl”。此外，测试辅助装置100例如通过预先存储可指定脚本ID的属性，能够从源文件中高效地获取脚本ID。此外，也可以根据属性之外的信息来获取脚本ID。

[0089] 接着回到图9，测试辅助装置100将在S912和S914中获取的脚本ID进行列表化(S915)。

[0090] 然后，测试辅助装置100从在S915列表化了的脚本ID中选择尚未选择的脚本ID(S916)。

[0091] 然后，测试辅助装置100从所存储的源代码156(源文件)中，对选择中的脚本ID所描述的脚本文件进行检索，并获取检索出的脚本文件的名称(脚本文件名)(S917)。而后，测试辅助装置100将获取到的脚本文件名和画面定义用脚本文件中描述的画面名的组存储为画面/源代码对应信息151(S918)。

[0092] 图10的(c)表示包含脚本ID的(所定义的)脚本文件的一个例子。在该脚本文件“top.js”中，将“TopCtrl”定义为脚本ID。

[0093] 接着回到图9，测试辅助装置100判定在S915中列表化了的脚本ID中是否存在尚未在S916中选择的ID(S919)。判定为是时(S919:是)，处理返回S916。判定为否时(S919:否)，进行S920的处理。

[0094] 在S920中，测试辅助装置100判定所存储的源文件214中是否存在尚未在S911中获取的画面定义用脚本文件。判定为是时(S920:是)，使处理返回S911，获取其他画面定义用脚本文件，并对新获取到的画面定义用脚本文件重复同样的处理。判定为否时(S920:否)，

进行图8中S912的处理。

[0095] 图11是说明图8的测试场景划分信息生成处理S812的细节的流程图。另外，图12的(a)是用来对测试场景划分信息生成处理S812进行说明的测试场景154的例子，图12的(b)是与图12的(a)的测试场景154有关的画面的例子。另外，图13是与图12的(b)中各画面相对应的源代码的例子。

[0096] 如图12的(a)所示，在列举出的测试场景154中描述了下述A)～G)的内容。

[0097] A)显示top.html

[0098] B)用户名输入(id＝“hitachi”)

[0099] C)密码输入(pass＝“pass”)

[0100] D)按下登录(id＝“login”)按钮

[0101] E)在文本区域显示“上一次登录时间”

[0102] F)按下(id＝“next”)按钮

[0103] G)在登录名中显示“hitachi”

[0104] 该测试场景154是在进行了顶部画面的测试后，从顶部画面迁移到确认画面，接着在进行了确认画面的测试后，从确认画面迁移到门户画面，最后进行门户画面的测试的场景。此外，如该例所示，在测试场景154的描述中并没有包含表示每个画面的分隔的信息。因此，无法直接确定某一画面的测试从哪一行开始到哪一行结束。

[0105] 下面结合图11对测试场景划分信息生成处理S812进行说明。如图11所示，首先，测试辅助装置100从所存储的测试场景154中获取一个尚未获取的场景(S1111)。

[0106] 然后，测试辅助装置100从获取到的测试场景154中，确定与在执行该测试场景154时最先成为测试对象的画面有关的描述(S1112)。在此，当测试场景154为图12的(a)示例的测试场景154时，测试辅助装置100，A)根据“显示top.html”中的“top.html”的描述以及画面/源代码对应信息151，判定为开始了“顶部画面”的测试。此外，测试辅助装置100判定测试场景154的执行内容的方法不一定受限，也可以根据例如“选择包含……按钮的窗口”、“选择……的标题的窗口”等描述判定为测试开始。

[0107] 接着，测试辅助装置100判定后续行中是否存在产生画面迁移的描述(S1113)。判定为是时(S1113:是)，进行S1114的处理。判定为否时(S1113:否)，进行S1116的处理。

[0108] 例如，当测试场景154为图12的(a)示例的测试场景时，测试辅助装置100对于“D)按下登录(id＝“login”)按钮”这一行，根据“顶部画面”的源文件即“top.html”的描述，按下“id＝“login””的按钮时调用函数“goLogin”，并根据“顶部画面”的源文件即“top.js”这一描述解释为迁移到“confirm.html”，且判定为该行是产生画面迁移的描述。此时，测试辅助装置100根据画面/源代码对应信息151，确定上述按钮被按下时的迁移目的画面为“确认画面”。

[0109] 在S1114中，测试辅助装置100将测试场景154的描述划分为画面单位的块，并将划分后的结果存储为测试场景划分信息152。

[0110] 在S1115中，测试辅助装置100判定获取中的测试场景154中是否有后续行。判定为是时(S1115:是)，处理返回S1113，对后续行重复进行来自S1113的处理。判定为否时(S1115:否)，进行S1116的处理。

[0111] 在图12的(a)示例的测试场景154中，测试辅助装置100对于后续行中的“F)按下向

下一个(id＝"next")按钮",判定该行是产生画面迁移的描述。此外,测试辅助装置100根据图13所示"确认画面"的源文件"confirm.html"的描述,在"id＝"next""的按钮被按下时调用函数"goNext",根据"确认画面"的源文件"confirm.js"的描述,确定迁移目的为"门户画面(portal.html)"。

[0112] 在S1116中,测试辅助装置100判定是否存在尚未获取的测试场景154。判定为是时(S1116:是),处理返回S1111。判定为否时(S1116:否),进行图8中S813的处理。

[0113] 此外,通过测试场景划分信息生成处理S812,图12的(a)所示的测试场景154例如如图14所示划分为三个块,即,与"顶部画面"有关的A)～D)块、与"确认画面"有关的E)～F)块和与"门户画面"有关的G)块。

[0114] 另外,虽然在上述说明中,对测试场景154进行划分时,参照了与画面的组成部件即按钮有关的描述,但也可以参照文字或图像等连接部件等的其他描述。

[0115] 如以上详细说明所述,本实施方式的测试辅助装置100生成以画面为单位将测试场景154的描述划分为多个块后得到的测试场景划分信息152,因此能够以画面单位掌握测试场景154的描述,例如,能够确定测试场景154的描述中与开发画面相对应的部分。因此,例如能够进行为了尽早获取开发画面的测试结果优先执行哪个测试场景154较好、为了尽早获取与开发画面的不良现象有关的信息优先执行哪个测试场景154较好等的判定。然后通过根据判定结果对多个测试场景中的每一个场景设定执行优先度,能够迅速且高效地提供用户所要求的信息。

[0116] 此外,例如对于图6所示的四个测试场景154,当用户要获取测试结果的开发画面是B画面时,如果以"画面跳数"为视点来设定执行优先度,则顺序为测试场景(2)→测试场景(1)→测试场景(3)。如果将其例如与按照测试场景(4)→测试场景(1)→测试场景(2)→测试场景(3)的顺序执行测试场景的情况相比较,则首先,由于不执行测试场景(4),因此可以减少测试场景(4)的执行时间。另外,通过先于执行测试场景(1)执行测试场景(2),减少了测试场景(1)的A画面的测试时间,使得用户能够尽早获取B画面的测试结果。

[0117] 以上基于实施方式对本发明人提出的发明进行了具体说明,但本发明并不仅限于上述实施方式,当然可以在不脱离其主旨的范围内进行各种变更。例如,上述实施方式是为了易于理解本发明而进行详细说明的方式,其并不一定仅限于具备所说明的所有结构的方式。此外,对于上述实施方式的部分结构,可以添加、删除或替换为其他结构。

[0118] 例如,本发明能够适用于例如在对通过软件所显示的画面进行回顾时想要迅速地显示特定的画面时等各种目的。

[0119] 此外,对于上述各结构、功能、处理部、处理单元等,可以将其一部分或全部例如通过集成电路设计等由硬件来实现。此外,上述各结构、功能等也可以通过处理器对实现每一功能的程序进行解释并执行,由软件来实现。实现各功能的程序、表格、文件等信息可预先存入存储器或硬盘、SSD(Solid State Drive)等记录装置,或者IC卡、SD卡、DVD等记录介质中。

[0120] 此外,在上述各图中,对于控制线或信息线,示出了进行说明所必须的部分,而并不限定示出安装上的所有控制线或信息线。实际上可以认为几乎所有的结构都互相连接。

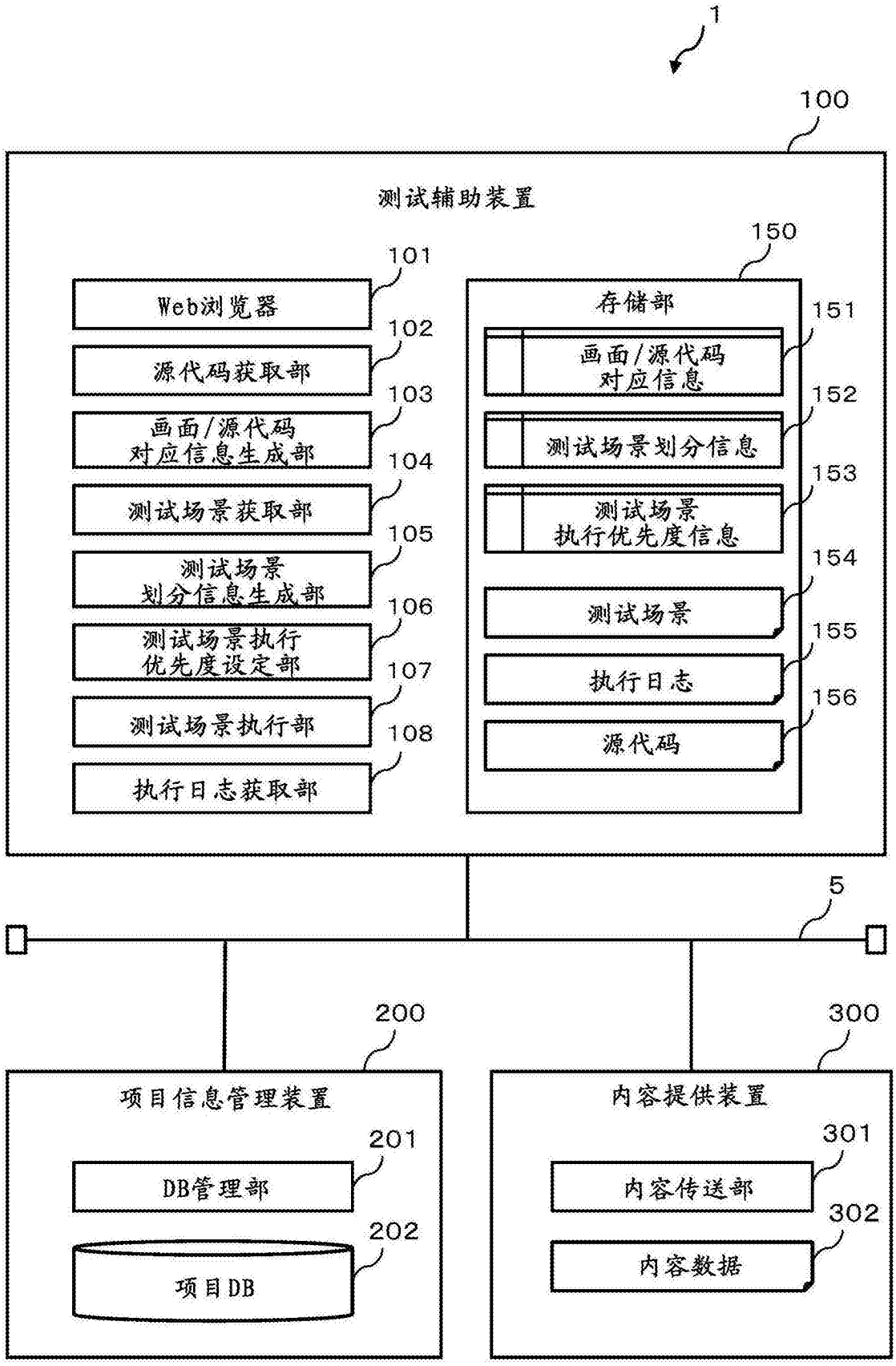

1
100
测试辅助装置
101
Web浏览器
102
源代码获取部
103
画面/源代码
对应信息生成部
104
测试场景获取部
105
测试场景
划分信息生成部
106
测试场景执行
优先度设定部
107
测试场景执行部
108
执行日志获取部
150
存储部
151
画面/源代码
对应信息
152
测试场景划分信息
153
测试场景
执行优先度信息
154
测试场景
155
执行日志
156
源代码
5
200
项目信息管理装置
201
DB管理部
202
项目DB
300
内容提供装置
301
内容传送部
302
内容数据

图1

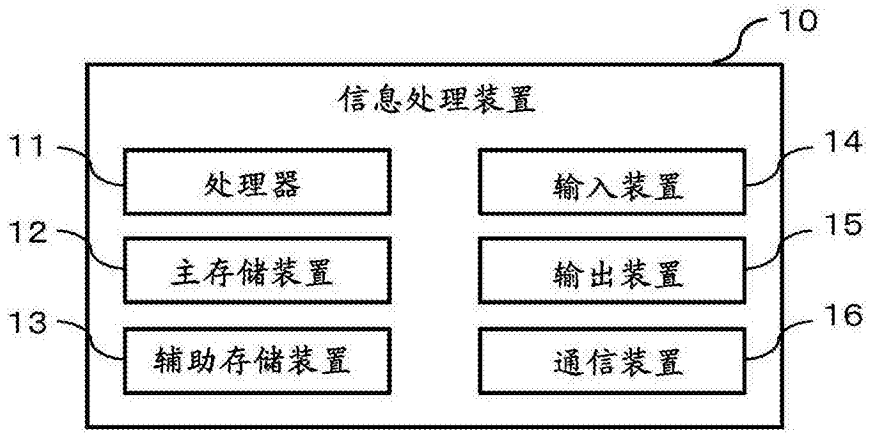


图2

151

画面/源代码对应信息

1511 画面ID	1512 源代码ID
样本画面	sample.html
	sample.js
	menu.html
	sidebar.html
	contents.html
	contents.js
顶部画面	top.html
	top.js
确认画面	confirm.html
	confirm.js
门户画面	portal.html
:	:

图3

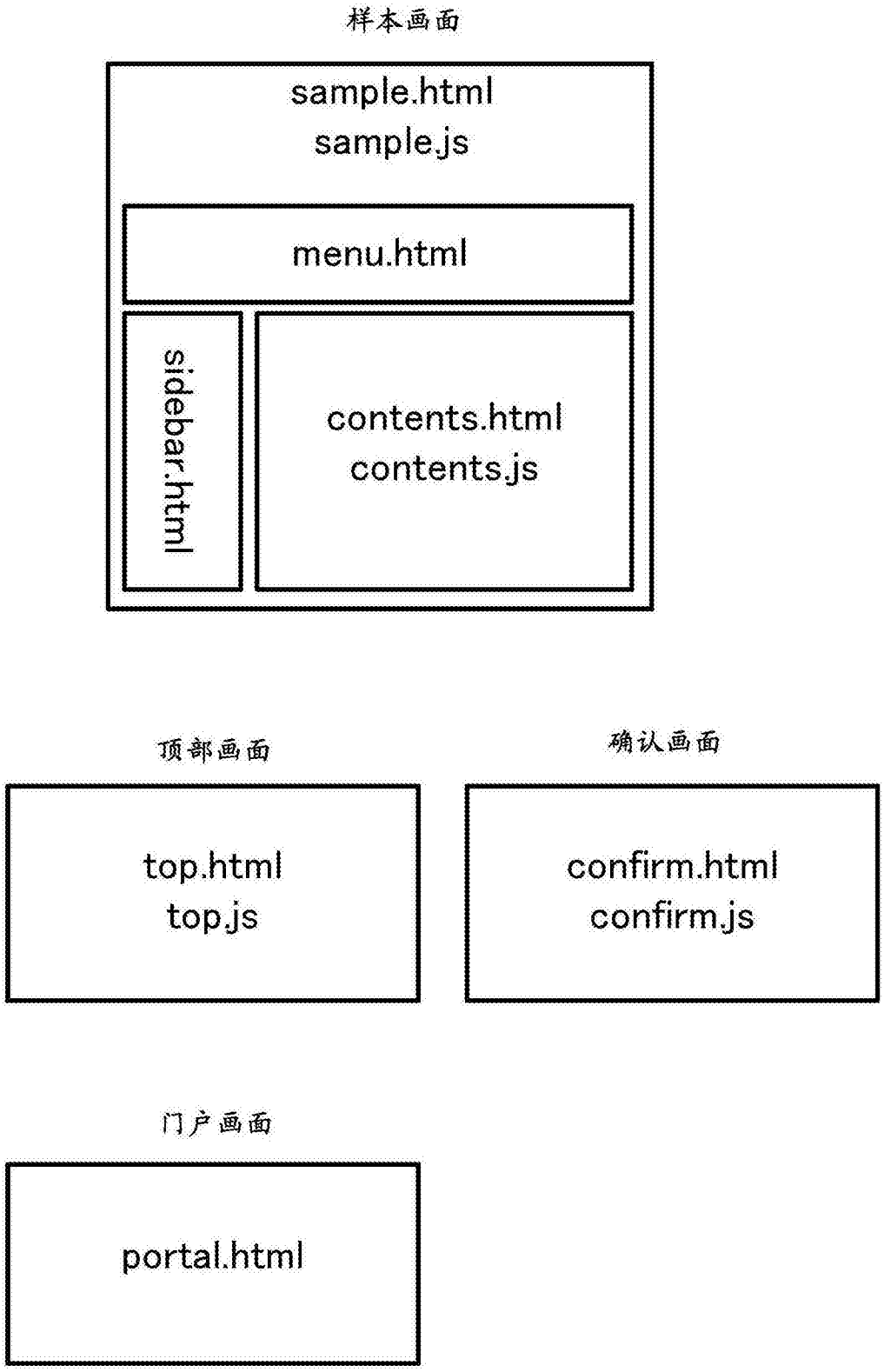

样本画面
sample.html
sample.js
menu.html
sidebar.html
contents.html
contents.js
顶部画面
top.html
top.js
确认画面
confirm.html
confirm.js
门户画面
portal.html

图4

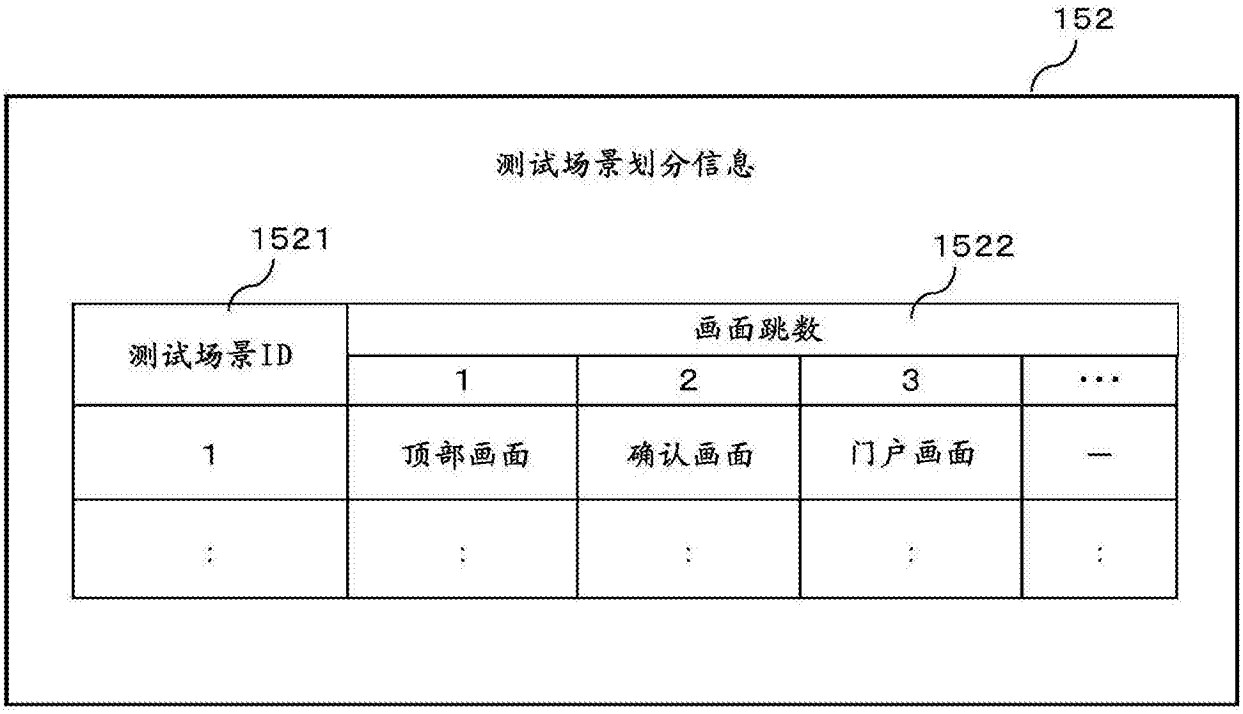


图5

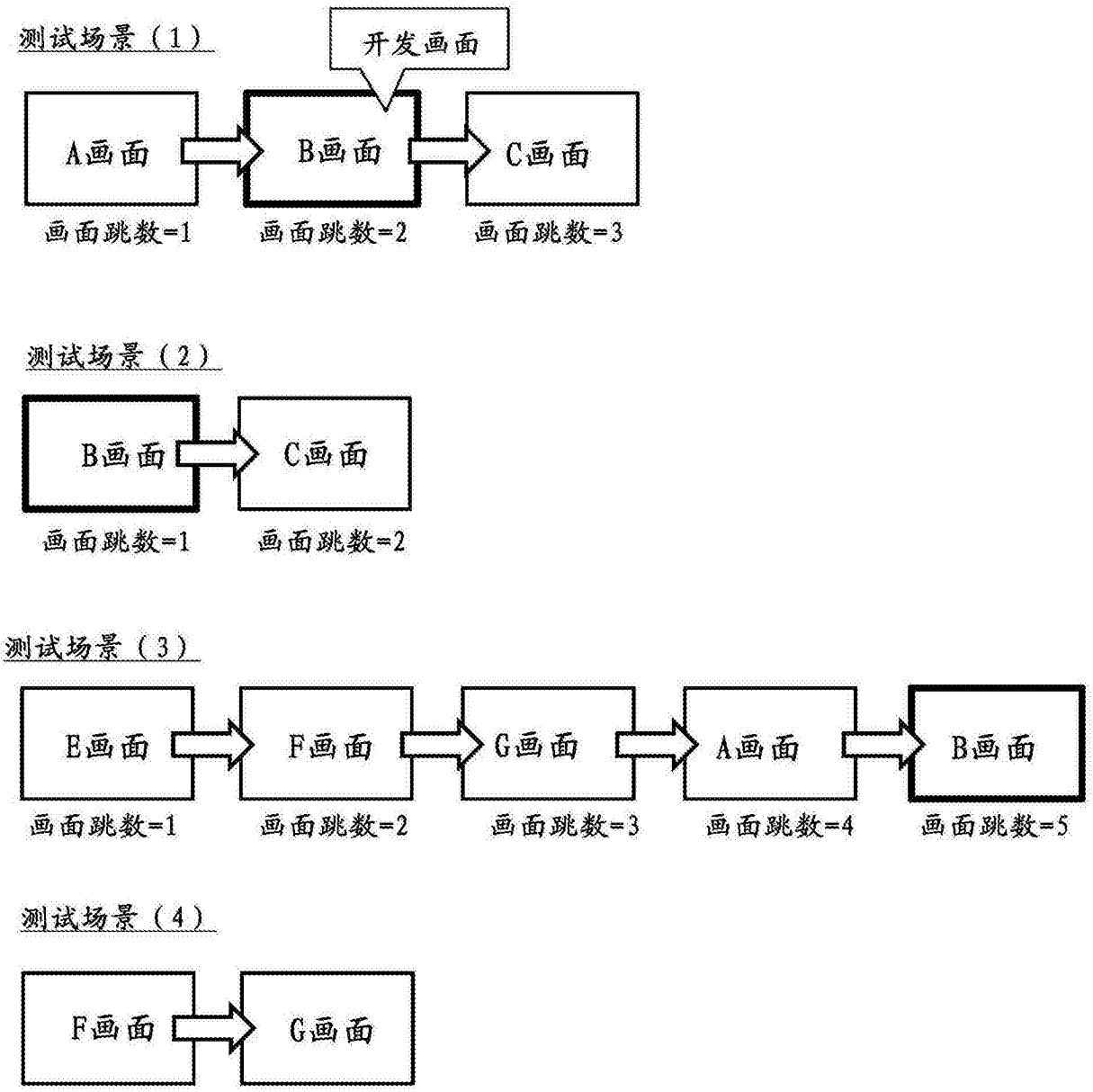

测试场景（1）
开发画面
A画面
B画面
C画面
画面跳数=1
画面跳数=2
画面跳数=3
测试场景（2）
B画面
C画面
画面跳数=1
画面跳数=2
测试场景（3）
E画面
F画面
G画面
A画面
B画面
画面跳数=1
画面跳数=2
画面跳数=3
画面跳数=4
画面跳数=5
测试场景（4）
F画面
G画面

图6

153

测试场景执行优先度信息

测试场景编号	顶部画面				···
	画面跳数	到达时间	不良现象发生次数	···	···
1	2	1	1	···	···
2	1	2	3	···	···
3	3	3	2	···	···
⋮	⋮	⋮	⋮	⋮	⋮

图7

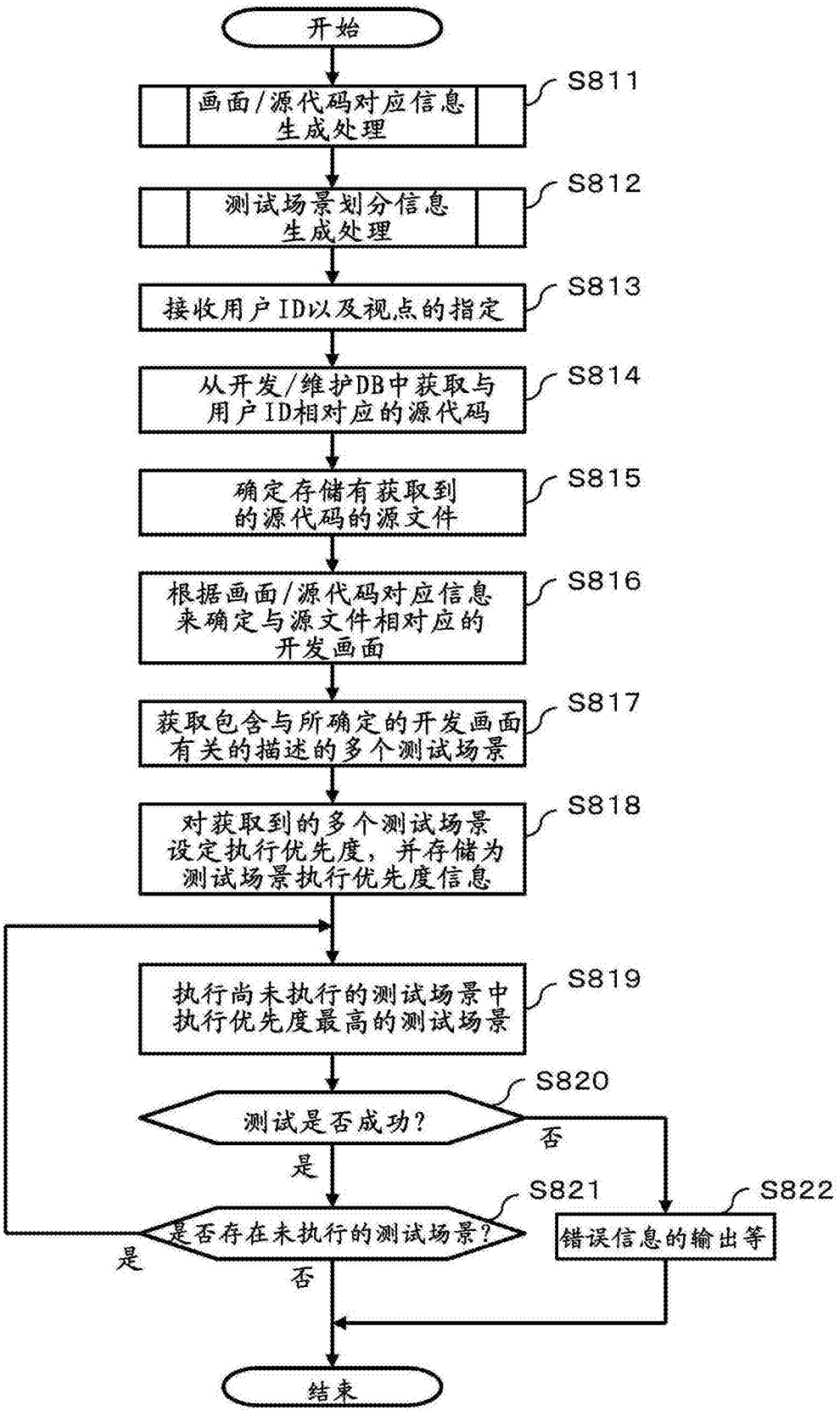

开始
画面/源代码对应信息生成处理
S811
测试场景划分信息生成处理
S812
接收用户ID以及视点的指定
S813
从开发/维护DB中获取与用户ID相对应的源代码
S814
确定存储有获取到的源代码的源文件
S815
根据画面/源代码对应信息来确定与源文件相对应的开发画面
S816
获取包含与所确定的开发画面有关的描述的多个测试场景
S817
对获取到的多个测试场景设定执行优先度，并存储为测试场景执行优先度信息
S818
执行尚未执行的测试场景中执行优先度最高的测试场景
S819
测试是否成功？
S820
否
是
是否存在未执行的测试场景？
S821
是
否
错误信息的输出等
S822
结束

图8

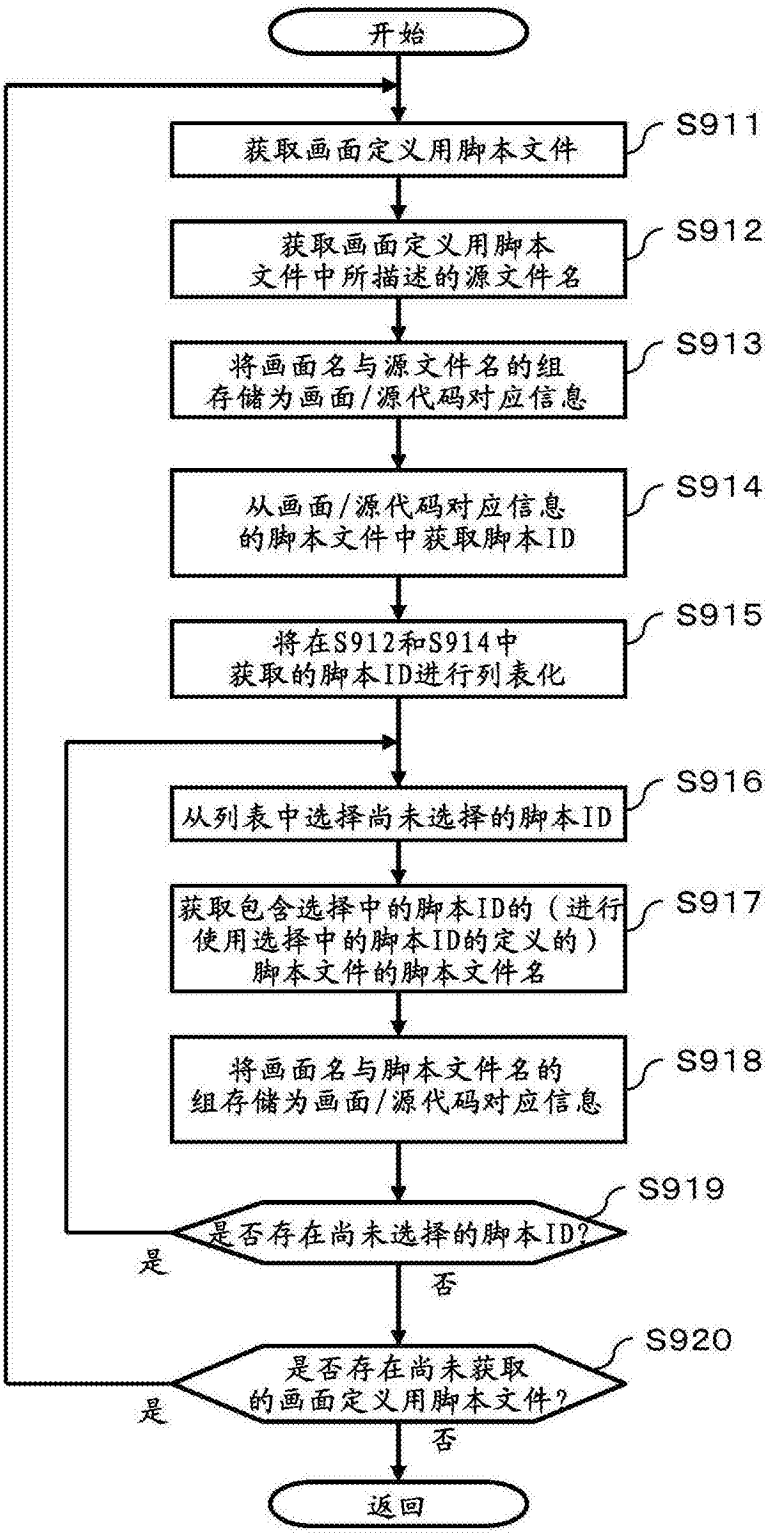

开始
获取画面定义用脚本文件
S911
获取画面定义用脚本文件中所描述的源文件名
S912
将画面名与源文件名的组存储为画面/源代码对应信息
S913
从画面/源代码对应信息的脚本文件中获取脚本ID
S914
将在S912和S914中获取的脚本ID进行列表化
S915
从列表中选择尚未选择的脚本ID
S916
获取包含选择中的脚本ID的（进行使用选择中的脚本ID的定义的）脚本文件的脚本文件名
S917
将画面名与脚本文件名的组存储为画面/源代码对应信息
S918
是否存在尚未选择的脚本ID?
S919
是
否
是否存在尚未获取的画面定义用脚本文件?
S920
是
否
返回

图9

```
顶部画面:{
  htmlFileName:'top.html',
  scriptID:'TopCtrl'
}
```

(a)

```
                :
<div controller='TopCtrl'>
                :
```

(b)

```
                :
module
  .controller('TopCtrl',
                :
```

(c)

图10

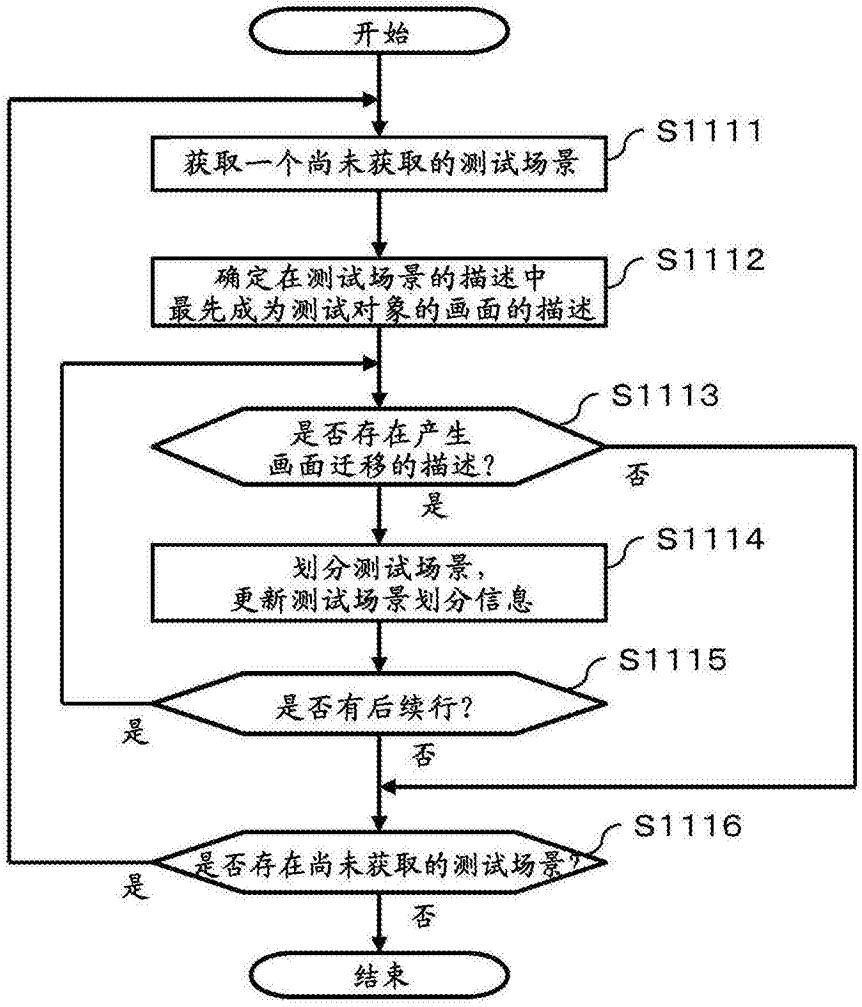
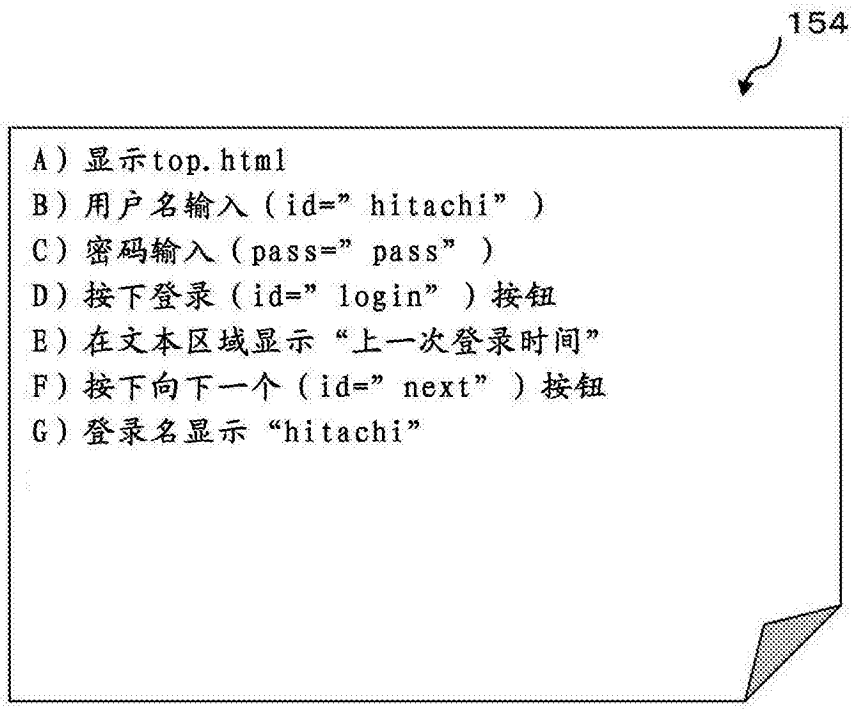
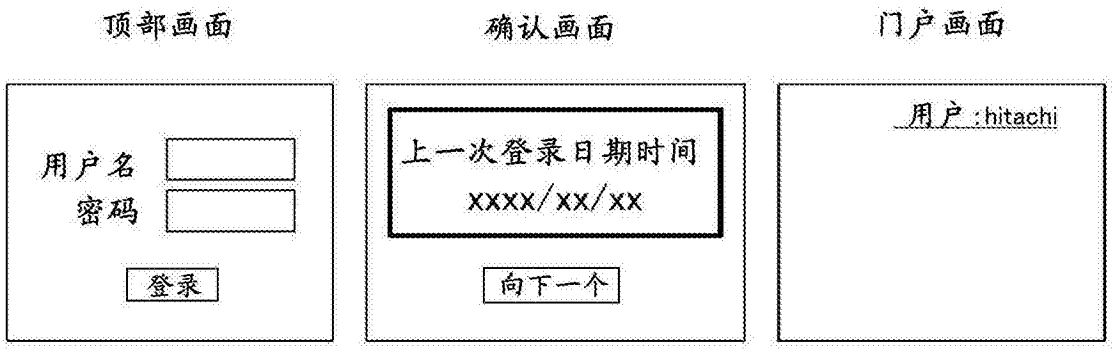

开始
获取一个尚未获取的测试场景
S1111
确定在测试场景的描述中
最先成为测试对象的画面的描述
S1112
是否存在产生
画面迁移的描述?
S1113
否
是
划分测试场景，
更新测试场景划分信息
S1114
是否有后续行?
S1115
是
否
是否存在尚未获取的测试场景?
S1116
是
否
结束

图11


154
A）显示top.html
B）用户名输入（id=" hitachi" ）
C）密码输入（pass=" pass" ）
D）按下登录（id=" login" ）按钮
E）在文本区域显示“上一次登录时间”
F）按下向下一个（id=" next" ）按钮
G）登录名显示“hitachi”

(a)


顶部画面
确认画面
门户画面
用户名
密码
登录
上一次登录日期时间
xxxx/xx/xx
向下一个
用户 :hitachi

(b)

图12

顶部画面

top.html

```
<html>
..
<form method="post">
    <input type="text" name="userid">
    <input type="password" name="password">
</form>
..

<input type="button" id="login" onclick="goLogin()">
..
</html>
```

top.js

```
..

function goLogin() {
    location.href="confirm.html";
}


..
```

确认画面

confirm.html

```
..
<input type="button" id="next" onclick="goNext()">
..
```

confirm.js

```
..
function goNext() {
    location.href="portal.html";
}
..
```

图13

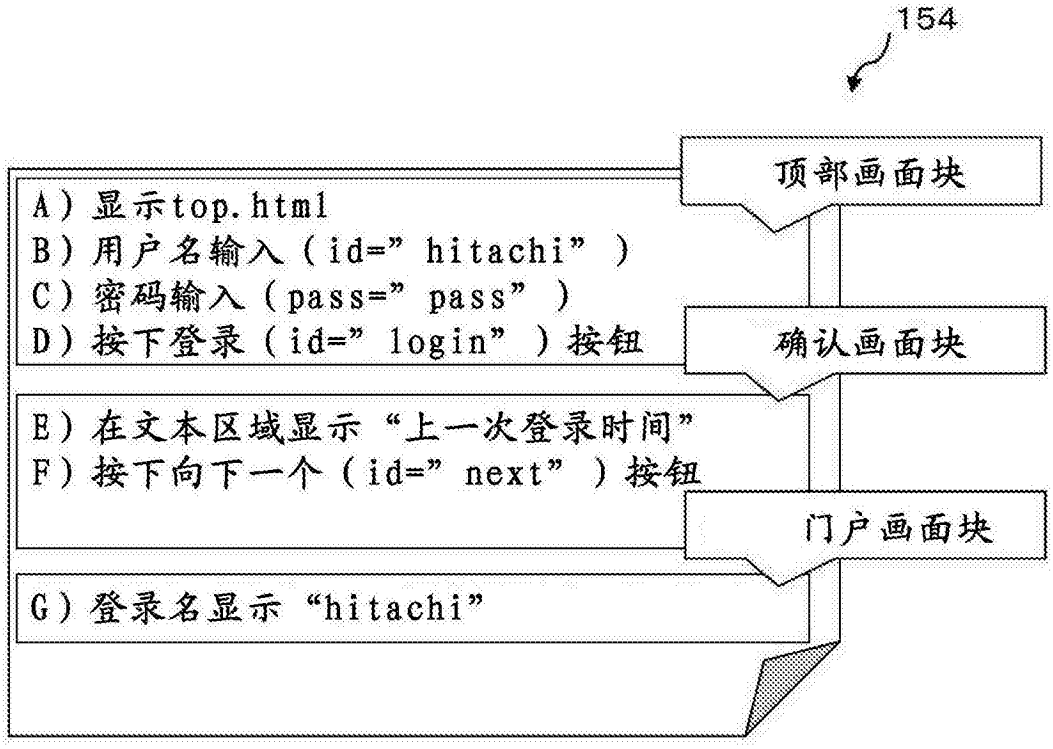

154
顶部画面块
A）显示top.html
B）用户名输入（id=” hitachi” ）
C）密码输入（pass=” pass” ）
D）按下登录（id=” login” ）按钮
确认画面块
E）在文本区域显示“上一次登录时间”
F）按下向下一个（id=” next” ）按钮
门户画面块
G）登录名显示“hitachi”

图14