

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第6441786号
(P6441786)

(45) 発行日 平成30年12月19日(2018.12.19)

(24) 登録日 平成30年11月30日(2018.11.30)

(51) Int.Cl. F I
G O 6 F 11/36 (2006.01) G O 6 F 11/36 1 8 8

請求項の数 15 (全 20 頁)

(21) 出願番号	特願2015-237213 (P2015-237213)	(73) 特許権者	000005108
(22) 出願日	平成27年12月4日(2015.12.4)		株式会社日立製作所
(65) 公開番号	特開2017-102807 (P2017-102807A)		東京都千代田区丸の内一丁目6番6号
(43) 公開日	平成29年6月8日(2017.6.8)	(74) 代理人	110000176
審査請求日	平成30年2月13日(2018.2.13)		一色国際特許業務法人
		(72) 発明者	是木 玄太
			東京都千代田区丸の内一丁目6番6号 株
			式会社日立製作所内
		(72) 発明者	前岡 淳
			東京都千代田区丸の内一丁目6番6号 株
			式会社日立製作所内
		(72) 発明者	中村 秀樹
			東京都千代田区丸の内一丁目6番6号 株
			式会社日立製作所内

最終頁に続く

(54) 【発明の名称】 テスト支援装置、テスト支援方法、及びプログラム

(57) 【特許請求の範囲】

【請求項1】

ソフトウェアのテストを支援する情報処理装置であって、
画面の遷移に関する記述を含む複数のテストシナリオと、
前記ソフトウェアを構成するソースコードと、
前記画面と前記画面の生成に関わる前記ソースコードとを対応付けた情報である画面／
ソースコード対応情報と、
を記憶する記憶部、
前記テストシナリオの記述を、前記画面を単位とした複数のブロックに分割した結果を
含む情報であるテストシナリオ分割情報を生成するテストシナリオ分割部、及び、
前記テストシナリオ分割情報に基づき、前記複数のテストシナリオの夫々に実行優先度
を設定するテストシナリオ実行優先度設定部、
を備える、
テスト支援装置。

10

【請求項2】

請求項1に記載のテスト支援装置であって、
前記テストシナリオ実行優先度設定部は、前記画面別かつ観点毎に、前記複数のテスト
シナリオの夫々に前記実行優先度を設定する、
テスト支援装置。

【請求項3】

20

請求項 2 に記載のテスト支援装置であって、

前記テストシナリオ実行優先度設定部は、特定の前記画面のテストが早期に開始される前記テストシナリオが他の前記テストシナリオよりも優先して実行されるようにするという前記観点に基づき、前記複数のテストシナリオの夫々に前記実行優先度を設定する、
テスト支援装置。

【請求項 4】

請求項 3 に記載のテスト支援装置であって、

前記記憶部は、前記複数のテストシナリオの夫々の実行ログをさらに記憶し、

前記テストシナリオ実行優先度設定部は、前記テストシナリオの夫々について、前記実行ログから前記特定の画面のテストが開始されるタイミングを特定する情報を取得し、取得した前記タイミングに基づき前記複数のテストシナリオの夫々に前記実行優先度を設定する、

テスト支援装置。

【請求項 5】

請求項 2 に記載のテスト支援装置であって、

前記記憶部は、前記複数のテストシナリオの夫々の実行ログをさらに記憶し、

前記テストシナリオ実行優先度設定部は、前記テストシナリオの夫々について、前記実行ログから、特定の前記画面についての不具合の発生頻度に関する情報を取得し、前記特定の画面についての不具合の発生頻度が多い前記テストシナリオが他の前記テストシナリオよりも優先して実行されるようにするという前記観点に基づき、前記実行優先度を設定する、

テスト支援装置。

【請求項 6】

請求項 1 乃至 5 のいずれか一項に記載のテスト支援装置であって、

前記記憶部は、前記ソースコードをファイルの形式で記憶し、

画面を定義する記述を含む前記ソースコードから、前記画面に関する記述を含むソースコードのファイル名を取得し、前記画面と前記ファイル名とを対応づけた情報を、前記画面／ソースコード対応情報として生成する、画面／ソースコード対応情報生成部をさらに備える、

テスト支援装置。

【請求項 7】

請求項 1 乃至 5 のいずれか一項に記載のテスト支援装置であって、

前記テストシナリオ分割部は、前記テストシナリオに記述されている、画面の遷移を生じさせる記述に基づきテストシナリオを分割する、

テスト支援装置。

【請求項 8】

請求項 1 乃至 5 のいずれか一項に記載のテスト支援装置であって、

前記実行優先度に従って前記複数のテストシナリオを順に実行するテストシナリオ実行部をさらに備える、

テスト支援装置。

【請求項 9】

請求項 8 に記載のテスト支援装置であって、

前記テストシナリオ実行部は、特定の前記画面の前記ブロックの実行が完了した時点で前記テストシナリオの実行を停止する、

テスト支援装置。

【請求項 10】

請求項 8 に記載のテスト支援装置であって、

前記テストシナリオ実行部は、前記テストシナリオの画面の遷移のパターンが他の前記テストシナリオの画面の遷移のパターンの全部又は一部と重複する場合、実行する前記テストシナリオを選択して前記パターンを重複して実行しないようにする、

テスト支援装置。

【請求項 1 1】

ソフトウェアのテストを支援する方法であって、
情報処理装置が、
画面の遷移に関する記述を含む複数のテストシナリオと、
前記ソフトウェアを構成するソースコードと、
前記画面と前記画面の生成に関わる前記ソースコードとを対応付けた情報である画面／
ソースコード対応情報と、
を記憶するステップと、
前記テストシナリオの記述を、前記画面を単位とした複数のブロックに分割した結果を
含む情報であるテストシナリオ分割情報を生成するステップと、
前記テストシナリオ分割情報に基づき、前記複数のテストシナリオの夫々に実行優先度
を設定するステップと、
を実行する、
テスト支援方法。

10

【請求項 1 2】

請求項 1 1 に記載のテスト支援方法であって、
前記情報処理装置は、前記画面別かつ観点毎に、前記複数のテストシナリオの夫々に前
記実行優先度を設定する、
テスト支援方法。

20

【請求項 1 3】

請求項 1 2 に記載のテスト支援方法であって、
前記情報処理装置は、特定の前記画面のテストが早期に開始される前記テストシナリオ
が他の前記テストシナリオよりも優先して実行されるようにするという観点に基づき、前
記複数のテストシナリオの夫々に前記実行優先度を設定する、
テスト支援方法。

【請求項 1 4】

請求項 1 3 に記載のテスト支援方法であって、
前記情報処理装置は、
前記複数のテストシナリオの夫々の実行ログをさらに記憶し、
前記テストシナリオの夫々について、前記実行ログから前記特定の画面のテストが開始
されるタイミングを特定する情報を取得し、取得した前記タイミングに基づき前記複数の
テストシナリオの夫々に前記実行優先度を設定する、
テスト支援方法。

30

【請求項 1 5】

情報処理装置に、
画面の遷移に関する記述を含む複数のテストシナリオと、
ソフトウェアを構成するソースコードと、
前記画面と前記画面の生成に関わる前記ソースコードとを対応付けた情報である画面／
ソースコード対応情報と、
を記憶する機能、
前記テストシナリオの記述を、前記画面を単位とした複数のブロックに分割した結果を
含む情報であるテストシナリオ分割情報を生成する機能、及び
前記テストシナリオ分割情報に基づき、前記複数のテストシナリオの夫々に実行優先度
を設定する機能、
を実現するためのプログラム。

40

【発明の詳細な説明】

【技術分野】

【0001】

50

本発明は、テスト支援装置、テスト支援方法、及びプログラム

【背景技術】

【0002】

特許文献1には、「リグレッションテストを行う場合に、テスト項目が膨大になってしまわないように、変更した箇所に応じて優先的にテストすべき部分を自動的に抽出し、リグレッションテストを効率的、自動的に行うことを可能にする」、「過去に行われたテストのテストケースについての情報を保存しておき、その情報と対象プログラムを解析して得られた依存関係を組み合わせることによりテストを行うリグレッションシステムを構成する」、「情報処理装置が、格納手段に格納された、ソフトウェアプログラムにおける各Statementの依存関係を参照し、変更されたStatementと関係あるStatementを抽出し、抽出したStatementと関連付けられたテストケースを、格納手段に格納された、過去に実行されたテストケースとソフトウェアプログラムにおける各Statementとの対応についての情報から、抽出したStatementと関連のあるテストケースを検出し、検出されたテストケースの実行を行う」と記載されている。

10

【特許文献1】特開2008-204405号公報

【発明の開示】

【発明が解決しようとする課題】

【0003】

ソフトウェアの開発現場においては、品質並びに開発効率の向上を図るべく、ソースコードの実装並びにテストを機能単位で繰り返し行う開発スタイルが定着している。またWebアプリケーション等のアプリケーションソフトウェアは、複数の画面によって一つの機能を実現していることがあり、こうしたソフトウェアのテストは、画面を遷移させる操作（ボタンやリンクの操作等）を含むユーザの操作を一連のシーケンスとして記述したテストシナリオを実行することにより行われることが多い。

20

【0004】

しかしテストシナリオを用いたテストにおいては、画面とテストシナリオの記述部分との対応や、画面とソースコードとの対応をとることが必ずしも容易ではない。そのため、とくに多数の開発者が携わるような大規模な情報処理システムの開発プロジェクトにおいては、不具合の発見や原因の特定に多大な労力と時間を費やしており、ソフトウェアの開発効率を低下させる要因となっている。

30

【0005】

本発明は、こうした背景に鑑みてなされたもので、ソフトウェアのテストを効率よく行うことが可能な、テスト支援装置、テスト支援方法、及びプログラムを提供することを目的としている。

【課題を解決するための手段】

【0006】

上記目的を達成するための本発明の一つは、ソフトウェアのテストを支援するテスト支援装置（情報処理装置）であって、画面の遷移に関する記述を含む複数のテストシナリオと、前記ソフトウェアを構成するソースコードと、前記画面と前記画面の生成に関わる前記ソースコードとを対応付けた情報である画面／ソースコード対応情報と、を記憶する記憶部、前記テストシナリオの記述を、前記画面を単位とした複数のブロックに分割した結果を含む情報であるテストシナリオ分割情報を生成するテストシナリオ分割部、及び、前記テストシナリオ分割情報に基づき、前記複数のテストシナリオの夫々に実行優先度を設定するテストシナリオ実行優先度設定部、を備える。

40

【発明の効果】

【0007】

本発明によれば、ソフトウェアのテストを効率よく行うことができる。

【図面の簡単な説明】

【0008】

50

【図 1】 情報処理システム 1 の概略的な構成並びに機能を示す図である。

【図 2】 プロジェクト情報管理装置 200 及びコンテンツ提供装置 300 のハードウェアの一例として示す情報処理装置 10 のブロック図である。

【図 3】 画面／ソースコード対応情報 151 の一例である。

【図 4】 図 3 の画面／ソースコード対応情報 151 に対応する画面レイアウトである。

【図 5】 テストシナリオ分割情報 152 の一例である。

【図 6】 画面ホップ数を説明する図である。

【図 7】 テストシナリオ実行優先度情報 153 の一例である。

【図 8】 テストシナリオ実行処理 S800 を説明するフローチャートである。

【図 9】 図 8 の画面／ソースコード対応情報生成処理 S811 の詳細を説明するフローチャートである。 10

【図 10】 (a) は画面定義用スクリプトの一例であり、(b) は (a) の画面定義用スクリプトに記述されているファイル名が「top.html」のソースファイル (HTML ファイル) の記述例であり、(c) はファイル名が「top.js」のスクリプトファイルの記述例である。

【図 11】 図 8 のテストシナリオ分割情報生成処理 S812 の詳細を説明するフローチャートである。

【図 12】 (a) はテストシナリオ 154 の一例であり、(b) は (a) のテストシナリオ 154 に関連する画面の例である。

【図 13】 図 12 (b) の各画面のソースコードの例である。 20

【図 14】 図 12 (a) に示したテストシナリオ 154 の記述を画面単位で分割した結果を説明する図である。

【発明を実施するための形態】

【0009】

以下、実施形態につき図面を参照しつつ説明する。尚、以下の説明において「データベース」のことを「DB」と略記する場合がある。

【0010】

図 1 に実施形態として説明する情報処理システム 1 の概略的な構成を示している。同図に示すように、情報処理システム 1 は、テスト支援装置 100、プロジェクト情報管理装置 200、及びコンテンツ提供装置 300 を含む。これらはいずれも情報処理装置 (コンピュータ) であり、通信ネットワーク 5 を介して互いに通信可能に接続している。通信ネットワーク 5 は、例えば、LAN (Local Area Network)、WAN (Wide Area Network)、インターネット等である。 30

【0011】

テスト支援装置 100 は、ユーザが行うソフトウェアのテストを支援する。本実施形態では、テスト支援装置 100 によるテストの対象となるソフトウェアが、インターネット上で Web サービスを提供するソフトウェアである場合を例として説明する。また以下の説明において想定するテスト支援装置 100 のユーザは、例えば、テスト支援装置 100 が出力する情報を利用する者や、テストの対象となるソフトウェアの開発者や保守担当者等である。 40

【0012】

テスト支援装置 100 は、例えば、ユーザが画面を遷移させる記述を含んだ複数のテストシナリオを実行しつつソフトウェアのテストを行うに際し、テストシナリオの実行を制御してユーザが効率よくテストを行うことができるようにする。例えば、テスト支援装置 100 は、ユーザが現在開発中の所定のソースコード (例えば、Web 記述言語 (HTML (HyperText Markup Language)、XHTML (eXtensible HyperText Markup Language)、XML (eXtensible Markup Language) 等)、各種スクリプト (JavaScript (登録商標) スクリプト等)) で記述された画面 (以下、開発画面と称する。) についてのテストを行うに際し、複数のテストシナリオの夫々に実行優先度を設定し、設定した実行優先度に従った順序で各テストシナリオを実行する。例えば、テストの対象となる 50

ソフトウェアの規模が大きく、テストシナリオが長大である場合には、テストシナリオの実行を開始してからユーザが検証しようとしている開発画面が実行されるまでに時間を要することがあるが、開発画面により早く到達するテストシナリオが優先して実行されるように上記実行優先度を設定することで、開発画面のテストが実行されるタイミングを早めることができ、ユーザは開発画面のテスト結果を迅速に取得することができる。

【0013】

テスト支援装置100は、上記実行優先度をユーザが指定する観点に従って設定する。例えば、テスト支援装置100は、開発画面のテストが早期に開始されるように上記実行優先度を設定する。またテスト支援装置100は、過去にテストシナリオを実行した際に取得した実行ログから開発画面の不具合の発生頻度に関する情報を取得し、不具合の発生頻度が多いテストシナリオが優先して実行されるように上記実行優先度を設定する。このようにテスト支援装置100は、上記実行優先度をユーザが指定する観点に従って設定するので、ユーザニーズに応じた形で開発画面のテストを行うことができる。尚、テスト支援装置100は、上記実行優先度の設定にあたり、テストシナリオの記述を画面単位で分割し、テストシナリオの記述を画面単位で効率よく把握できるようにする。

10

【0014】

図2は、テスト支援装置100、プロジェクト情報管理装置200、及びコンテンツ提供装置300を実現するハードウェアの一例として示す情報処理装置10（コンピュータ）のブロック図である。情報処理装置10は、例えば、ネットワークストレージとして機能するものであってもよい。尚、図1では、テスト支援装置100、プロジェクト情報管理装置200、及びコンテンツ提供装置300が、夫々、独立したハードウェアとして構成されるように描いているが、これらのうちの2つ以上が共通のハードウェア（プラットフォーム）によって実現されていてもよい。またこれらのうちの一つ以上を、仮想的な情報処理装置（例えば、クラウドシステムによって提供されるクラウドサーバ等）を用いて実現してもよい。

20

【0015】

同図に示すように、情報処理装置10は、プロセッサ11、主記憶装置12、補助記憶装置13、入力装置14、出力装置15、及び通信装置16を備える。これらは図示しないバス等の通信手段を介して互いに通信可能に接続されている。

【0016】

プロセッサ11は、例えば、CPU（Central Processing Unit）やMPU（Micro Processing Unit）を用いて構成されている。プロセッサ11が、主記憶装置12に格納されているプログラムを読み出して実行することにより、情報処理装置10の様々な機能が実現される。主記憶装置12は、プログラムやデータを記憶する装置であり、例えば、ROM（Read Only Memory）、RAM（Random Access Memory）、NVRAM（Non Volatile RAM）等である。補助記憶装置13は、例えば、ハードディスクドライブ、SSD（Solid State Drive）、光学式記憶装置、記録媒体の読取／書込装置等である。補助記憶装置13に格納されているプログラムやデータは主記憶装置12に随時ロードされる。

30

【0017】

入力装置14は、例えば、キーボード、マウス、タッチパネル等である。出力装置15は、例えば、液晶モニタ、LCD（Liquid Crystal Display）、グラフィックカード、スピーカ等である。通信装置16は、通信ネットワーク5を介して他の装置と通信する通信インタフェースであり、例えば、NIC（Network Interface Card）、無線通信モジュール等である。入力装置14や出力装置15は必ずしも必須の構成ではなく、例えば、情報処理装置10が通信装置16を介して他の装置との間で情報の入力や出力を行う構成としてもよい。

40

【0018】

続いて、図1とともに、テスト支援装置100が備える機能、及びテスト支援装置100が管理する情報（データ）について説明する。同図に示すように、テスト支援装置100は、Webブラウザ101、ソースコード取得部102、画面／ソースコード対応情報

50

生成部103、テストシナリオ取得部104、テストシナリオ分割情報生成部105、テストシナリオ実行優先度設定部106、テストシナリオ実行部107、実行ログ取得部108、及び記憶部150を備える。これらの機能は、テスト支援装置100のプロセッサ11が、主記憶装置12に格納されているプログラムを読み出して実行することにより実現される。これらの機能の全部又は一部をハードウェアとして実現してもよい。これらの機能に加え、テスト支援装置100において、各機能の起動／管理やファイルシステムによるデータの管理等を行う、オペレーティングシステム(Operating System)や制御プログラム等のソフトウェアが動作していてもよい。

【0019】

同図に示すように、記憶部150は、画面／ソースコード対応情報151、テストシナリオ分割情報152、テストシナリオ実行優先度情報153、テストシナリオ154、実行ログ155、及びソースコード156を記憶する。

10

【0020】

同図に示す機能のうち、Webブラウザ101は、例えば、テストシナリオ154に従ってソースコード156を読み込み、ソースコード156に基づくコンテンツを記載した画面(Webページ)の生成や表示、コンテンツの再生等の処理を行う。上記処理に際し、Webブラウザ101は、コンテンツ提供装置300に適宜コンテンツの取得要求を送信してコンテンツ提供装置300からコンテンツデータ302を取得する。

【0021】

ソースコード取得部102は、通信ネットワーク5を介してプロジェクト情報管理装置200から適宜ソースコード156を取得する。尚、ソースコード156は、例えば、Web記述言語や各種スクリプトで記述されたデータである。テスト支援装置100は、これらのデータを、例えば、ファイルシステムにより取り扱われるファイルを単位として管理する。

20

【0022】

画面／ソースコード対応情報生成部103は、記憶部150が記憶しているソースコード156(ソースファイル)に基づき、画面／ソースコード対応情報151を生成する。画面／ソースコード対応情報151の詳細については後述する。

【0023】

テストシナリオ取得部104は、通信ネットワーク5を介してプロジェクト情報管理装置200から適宜テストシナリオ154を取得する。テストシナリオ154は、例えば、自然言語で記述された複数のステップ(行)を含む。テストシナリオ154は、所定の業務のシナリオに沿ってWebページの操作(画面)を行い画面遷移を発生させる。テストシナリオ154は、画面遷移を発生させるための記述や画面についてどのようなテストを行うかを示す記述を含む。

30

【0024】

テストシナリオ分割情報生成部105は、テストシナリオ154を、ソースコード156に基づき生成される画面を単位とした複数のブロックに分割し、分割した結果を記載したテストシナリオ分割情報152を生成する。テストシナリオ分割情報152の詳細については後述する。

40

【0025】

テストシナリオ実行優先度設定部106は、テストシナリオ取得部104が取得した複数のテストシナリオ154の夫々に実行優先度を設定し、設定した結果をテストシナリオ実行優先度情報153として記憶する。実行優先度の具体的な設定方法やテストシナリオ実行優先度情報153の詳細については後述する。

【0026】

テストシナリオ実行部107は、テストシナリオ154に設定されている実行優先度に従って各テストシナリオ154を順に実行する。テストシナリオ154の実行に際し、テストシナリオ実行部107は、例えば、Webブラウザ101を制御して、テストシナリオ154に従った操作入力の手順を自動的に実行する。テストシナリオ実行部107の機

50

能を実現するソフトウェアとして、例えば、「C u c u m b e r（登録商標）」、「T u r n i p（登録商標）」等がある。テストシナリオ実行部107は、例えば、コンテンツ提供装置300において動作するWebサーバと連携して動作するものであってもよい。

【0027】

実行ログ取得部108は、テストシナリオ154の実行に際しテストシナリオ実行部107が生成する情報（各ステップの実行開始／終了時刻、エラーの発生有無、エラー発生箇所を特定する情報、発生したエラーの内容、実行履歴等）を取得し、取得した情報を実行ログ155として記憶する。

【0028】

プロジェクト情報管理装置200は、ユーザが携わるプロジェクト（開発、保守、設計等のプロジェクト）に関する情報を統合的に蓄積／管理する。プロジェクト情報管理装置200は、プロジェクトに関する情報が管理されるプロジェクトDB202と、プロジェクトDB202を管理するDB管理部201とを有する。DB管理部201は、例えば、DBMS（DataBase Management System）や分散型バージョン管理システム等の機能を有する。プロジェクト情報管理装置200は、例えば、リポジトリとして管理されるソフトウェアやデータ項目にメタデータ（仕様書、設計書、モデル、ダイアグラム、情報操作ルール等）を対応づけた情報をプロジェクトDB202に管理する。

【0029】

コンテンツ提供装置300は、コンテンツデータ302を記憶し、コンテンツデータ302を配信するコンテンツ配信部301を有する。コンテンツデータ302は、例えば、所定のデータフォーマットで記述されたデータ（文字データ、音声データ、画像データ、動画データ等）である。コンテンツ提供装置300は、前述したコンテンツの取得要求を受信すると、該当するコンテンツデータ302をテスト支援装置100に送信する。

【0030】

尚、テストシナリオ154、実行ログ155、ソースコード156、プロジェクトDB202、及びコンテンツデータ302は、必ずしも以上に示した形態で配置されている必要はなく、これら各データの配置形態は情報処理システム1の構成に応じて適宜変更可能である。

【0031】

<画面／ソースコード対応情報>

図3に画面／ソースコード対応情報151の一例を、また図4に図3の画面／ソースコード対応情報151に対応する画面レイアウトを示す。図3に示すように、画面／ソースコード対応情報151は、画面ID1511とソースコードID1512との対応を示す情報を含む。この例では、画面ID1511として、ユーザが画面の識別子として用いている画面名を、またソースコードID1512として、ソースコードが記述されているファイルのファイル名を用いている。尚、画面ID1511とソースコードID1512との対応は、プロジェクトの形態や情報処理システム1の態様等に応じて様々である。また図4の「サンプル画面」のように、一つの画面が複数のソースコード（ソースファイル）で構成されている場合もあり、画面ID1511とソースコードID1512との対応は、必ずしも1対1の関係であるとは限らない。

【0032】

図3乃至図4に示すように、この例では、画面ID1511「サンプル画面」に対応するソースファイル（ソースコード）として、「sample.html」、「sample.js」、「menu.html」、「sidebar.html」、「contents.html」、及び「contents.js」が存在する。また、図4に示すように、「サンプル画面」では、「sample.html」の内部で「menu.html」、「sidebar.html」、及びcontents.htmlを用いている。この例では、「sample.html」に記述されている単数または複数の画面部品の挙動は「sample.js」に、「contents.html」に記述されている単数または複数の画面部品の挙動は「contents.js」に、夫々記述されている。

【0033】

また画面 I D 1 5 1 1 「トップ画面」に対応するソースファイル（ソースコード）として、「top.html」及び「top.js」が存在する。「top.html」に記述されている単数または複数の画面部品の挙動は「top.js」に記述されている。

【0034】

また画面 I D 1 5 1 1 「確認画面」に対応するソースファイル（ソースコード）として、「confirm.html」及び「confirm.js」が存在する。「confirm.html」に記述されている単数または複数の画面部品の挙動は「confirm.js」に記述されている。

【0035】

また画面 I D 1 5 1 1 「ポータル画面」に対応するソースファイルとして、「portal.html」が存在する。「portal.html」に記述されている単数または複数の画面部品は挙動を定義する必要がなく、スクリプトファイルは存在しない。

【0036】

<テストシナリオ分割情報>

図5にテストシナリオ分割情報152の一例を示す。同図において、テストシナリオ I D 1 5 2 1 には、テストシナリオ154ごとに付与される識別子であるテストシナリオ I D が設定される。また画面ホップ数1522には、テストシナリオ154の記述を画面単位で分割して得られる各ブロックが、当該テストシナリオ154の中で何番目に位置しているかを示す情報が設定される。同図において、上記各ブロックは、画面ホップ数「1」、「2」、「3」・・・ごとに設けられた枠内に記載されている画面名で特定される。

【0037】

同図に示したテストシナリオ分割情報152は、テストシナリオ I D が「1」のテストシナリオ154の記述が、「トップ画面」に関するブロック、「確認画面」に関するブロック、及び「ポータル画面」に関するブロックの3つのブロックに分割されていることを示している。またこのテストシナリオ分割情報152は、テストシナリオ I D が「1」のテストシナリオ154を実行した場合、「トップ画面」のテストが行われ、続いて「トップ画面」から「確認画面」へ画面遷移した上で「確認画面」のテストが行われ、最後に「確認画面」から「ポータル画面」へ画面遷移した上で「ポータル画面」のテストが行われる事を示している。

【0038】

図6とともに画面ホップ数について具体的に説明する。同図において、白抜き矢線は画面の遷移を表している。この例の場合、テストシナリオ（1）では、A画面→B画面→C画面の順に、テストシナリオ（2）では、B画面→C画面の順に、テストシナリオ（3）では、E画面→F画面→G画面→A画面→B画面の順に、テストシナリオ（4）では、F画面→G画面の順に画面のテストが実行される。

【0039】

画面ホップ数は、テストシナリオ154に従ってテストを行った場合に、テストの対象となる画面をテスト開始時から数えた順序を表わす。例えば、同図において、テストシナリオ（1）では、B画面の画面ホップ数は「2」となり、テストシナリオ（2）では、B画面の画面ホップ数は「1」となり、テストシナリオ（3）では、B画面の画面ホップ数は「5」となる。尚、テストシナリオ（4）では、B画面はテストの対象になっておらず、この場合の画面ホップ数は不定（無し）となる。

【0040】

尚、テスト対象となる画面がB画面であり、各テストシナリオの実行優先度を画面ホップ数を観点として設定する場合、図6の例では、テストシナリオ（2）→テストシナリオ（1）→テストシナリオ（3）の順に実行優先度が設定される。

【0041】

<テストシナリオ実行優先度情報>

テストシナリオ実行優先度情報153には、テストシナリオ実行優先度設定部106が設定した、画面別かつ観点毎の、各テストシナリオ154の実行優先度が設定される。

【0042】

10

20

30

40

50

図7に、テストシナリオ実行優先度情報153の一例を示す。テストシナリオ実行優先度情報153は、画面別かつ観点毎に設定されたテストシナリオ154の実行優先度を示す情報を含む。同図は、「トップ画面」について3つの観点（「画面ホップ数」、「到達時間」、及び「不具合発生回数」）毎に実行優先度を設定した場合である。

【0043】

同図において、「画面ホップ数」を観点とした実行優先度には「トップ画面」の画面ホップ数が設定される。

【0044】

また同図において、「到達時間」を観点とした実行優先度は、テストシナリオ実行優先度設定部106が、各テストシナリオ154の過去の実行時に取得された実行ログ155に基づき設定する。例えば、テストシナリオ実行優先度設定部106は、実行ログ155から把握される、該当画面に到達するまでの時間（例えば、テストシナリオ154の実行が開始されてから該当画面のブロックの実行が開始されるまでの時間（開始タイミング））が短いものほど実行優先度が高くなるように、「到達時間」を観点とした実行優先度を設定する。

10

【0045】

また同図において、「不具合発生回数」を観点とした実行優先度は、テストシナリオ実行優先度設定部106が、各テストシナリオ154の過去の実行時に取得された実行ログ155に基づき設定する。例えば、テストシナリオ実行優先度設定部106は、実行ログ155から把握される、該当画面の不具合発生回数が多いものほど実行優先度が高くなるように、「不具合発生回数」を観点とした実行優先度を設定する。

20

【0046】

ここで例えば、特定の開発画面のテストを早期に開始させたい場合、ユーザは、テストシナリオ154の実行優先度を設定する際の観点として「画面ホップ数」又は「到達時間」を選択する。また例えば、特定の開発画面の不具合を早期に発見したい場合、ユーザは、テストシナリオ154の実行優先度を設定する際の観点として「不具合発生回数」を選択する。

【0047】

尚、以上に例示した観点は一例に過ぎず、これら以外の観点を設定してもよい。また例えば、複数の観点の夫々から求まる実行優先順位を総合的に判断してテストシナリオ154の実行優先度を設定するようにしてもよい（例えば、2つの観点の夫々に基づく実行優先度が逆転している場合にいずれか一方の観点に基づく実行優先度を優先して採用する等）。また対象となる画面に関する記述を含んでいないテストシナリオ154を実行の対象から除外するようにしてもよい。その場合、例えば、テストシナリオ実行優先度情報153の該当するテストシナリオの欄に「－」等の所定の記号を設定する。

30

【0048】

<処理説明>

続いて、以上の構成からなる情報処理システム1において行われる処理について説明する。

【0049】

40

図8は、ソフトウェアが表示する画面のテストに際してテスト支援装置100が行う処理（以下、テストシナリオ実行処理S800と称する。）を説明するフローチャートである。

【0050】

まずテスト支援装置100は、画面／ソースコード対応情報151を生成する処理（以下、画面／ソースコード対応情報生成処理S811と称する。）、及びテストシナリオ分割情報152を生成する処理（以下、テストシナリオ分割情報生成処理S812と称する。）を行う（S811、S812）。これらの処理の詳細については後述する。

【0051】

続いて、テスト支援装置100は、ユーザから、ユーザID（例えば、開発画面の開発

50

に關係する開発者名等)及び観点の指定を受け付ける(S813)。ユーザは単数又は複数の観点を指定することができる。

【0052】

続いて、テスト支援装置100は、S813で受け付けたユーザIDに対応づけられているソースコードを、プロジェクト情報管理装置200のプロジェクトDB202から取得し、ソースコード156として記憶する(S814)。尚、本例では記述していないが、例えば、ユーザから受け付けた開発日時等の情報に基づき、プロジェクトDB202から取得するソースコードを絞り込むようにしてもよい。

【0053】

続いて、テスト支援装置100は、取得したソースコードが格納されているソースファイルを特定し(S815)、画面/ソースコード対応情報151から、当該ソースファイルに対応する画面(以下、開発画面と称する。)を特定する(S816)。

【0054】

続いて、テスト支援装置100は、テストシナリオ分割情報152を参照し、S816で特定した開発画面に関する記述を含むテストシナリオ154を取得する(S817)。尚、本例では、テスト支援装置100が複数のテストシナリオ154を取得するものとする。

【0055】

続いて、テスト支援装置100は、S813でユーザから受け付けた観点に基づき、S817で取得した複数のテストシナリオ154の夫々に実行優先度を設定し、その結果をテストシナリオ実行優先度情報153として記憶する(S818)。

【0056】

続いて、テスト支援装置100は、未実行のテストシナリオ154のうち実行優先度の最も高いものを実行する(S819)。

【0057】

続いて、テスト支援装置100は、開発画面に関するテストの実行が成功したか否かを判定する(S820)。判定が真であれば(S820:YES)、処理はS821に進む。判定が偽であれば(S820:NO)、処理はS822に進み、テスト支援装置100は、エラー情報の出力等の処理を行った後、テストシナリオ154の実行を終了する。尚、このように開発画面に関するテストが終了した段階でテストシナリオ154の実行を終了させてもよいし、そのままテストシナリオを最後まで実行するようにしてもよい。前者のように開発画面に関するテストが終了した段階でテストシナリオ154の実行を終了させる場合、ユーザは早期にテストの結果を取得することができ、不具合の発見に要する時間を短縮することができる。

【0058】

S821では、テスト支援装置100は、開発画面に関するテストシナリオ154のうち未実行のテストシナリオ154があるか否かを判定する。判定が真であれば(S821:YES)、処理はS819に戻り、未実行のテストシナリオ154を実行する。判定が偽であれば(S821:NO)、テスト支援装置100は、テストシナリオ実行処理S800を終了する。

【0059】

尚、以上に説明したテストシナリオ実行処理S800において、あるテストシナリオ154の画面遷移のパターンが他のテストシナリオ154の画面遷移のパターンの全部又は一部と重複している場合には、テストの効率を向上すべく、実行するテストシナリオ154を選択することにより重複するパターンについて重複して実行しないようにしてもよい。例えば、図6では、テストシナリオ(1)の画面遷移のパターンはA画面→B画面→C画面であり、テストシナリオ(2)の画面遷移のパターンはB画面→C画面であり、いずれもB画面→C画面という画面遷移のパターンを含んでいるが、この場合、例えば、テスト支援装置100がテストシナリオ(2)を実行しないようにする。

【0060】

図9は、図8の画面／ソースコード対応情報生成処理S811の詳細を説明するフローチャートである。以下、同図とともに説明する。

【0061】

まずテスト支援装置100は、記憶しているソースコード156（ソースファイル）の中から、画面の定義を行っているソースコード（以下、画面定義用スクリプトファイルと称する。）を取得する。例えば、テスト支援装置100は、ソースコード156（ソースファイル）の中から、画面名とソースファイル名（例えば、HTMLファイル名、HTMLファイルID、スクリプトファイル名、スクリプトID等）との組が記述されている画面定義用スクリプトファイルを取得する（S911）。

【0062】

続いて、テスト支援装置100は、S911で取得した画面定義用スクリプトファイルから、画面名に対応づけられているソースファイル名を取得し（S912）、画面名とソースファイル名との組を画面／ソースコード対応情報151として記憶する（S913）。

【0063】

図10（a）に画面定義用スクリプトの一例を示す。この画面定義用スクリプトには、画面名として「トップ画面」が、またソースファイル名（HTMLファイル名）として「top.html」が記述されている。またこの画面定義用スクリプトには、「トップ画面」に対応するスクリプトIDとして「TopCtrl」が記述されている。尚、このように画面定義用スクリプトにスクリプトファイル名ではなくスクリプトIDが記述されている場合には、後の処理にてスクリプトIDからスクリプトファイル名を特定する。

【0064】

図9に戻り、続いて、テスト支援装置100は、画面／ソースコード対応情報151に記載されているソースファイルを解析してスクリプトIDを取得する（S914）。

【0065】

図10（b）に（a）の画面定義用スクリプトに記述されているファイル名が「top.html」のソースファイル（HTMLファイル）の記述例を示す。この例では「controller」属性に「TopCtrl」が設定されている。尚、テスト支援装置100が、例えば、スクリプトIDを指定することが可能な属性を予め記憶しておくことで、ソースファイルからスクリプトIDを効率よく取得することができる。また属性以外の情報に基づきスクリプトIDを取得してもよい。

【0066】

図9に戻り、続いて、テスト支援装置100は、S912及びS914で取得したスクリプトIDをリスト化する（S915）。

【0067】

続いて、テスト支援装置100は、S915でリスト化したスクリプトIDのうち、未選択のスクリプトIDを選択する（S916）。

【0068】

続いて、テスト支援装置100は、記憶しているソースコード156（ソースファイル）から、選択中のスクリプトIDが記述されているスクリプトファイルを検索し、検索されたスクリプトファイルの名称（スクリプトファイル名）を取得する（S917）。そしてテスト支援装置100は、取得したスクリプトファイル名と、画面定義用スクリプトファイルに記述されている画面名との組を、画面／ソースコード対応情報151として記憶する（S918）。

【0069】

図10（c）に、スクリプトIDを含む（定義している）スクリプトファイルの一例を示す。このスクリプトファイル「top.js」では、スクリプトIDとして「TopCtrl」が定義されている。

【0070】

図9に戻り、続いて、テスト支援装置100は、S915でリスト化したスクリプトI

10

20

30

40

50

Dのうち、S 9 1 6で未選択のものがあるか否かを判定する（S 9 1 9）。判定が真であれば（S 9 1 9：YES）、処理はS 9 1 6に戻る。判定が偽であれば（S 9 1 9：NO）、処理はS 9 2 0に進む。

【0071】

S 9 2 0では、テスト支援装置100は、記憶しているソースファイル214にS 9 1 1で未取得の画面定義用スクリプトファイルが存在するか否かを判定する。判定が真であれば（S 9 2 0：YES）、処理はS 9 1 1に戻り、他の画面定義用スクリプトファイルを取得し、新たに取得した画面定義用スクリプトファイルについて同様の処理を繰り返す。判定が偽であれば（S 9 2 0：NO）、処理は図8のS 9 1 2に進む。

【0072】

図11は、図8のテストシナリオ分割情報生成処理S 8 1 2の詳細を説明するフローチャートである。また図12（a）は、テストシナリオ分割情報生成処理S 8 1 2の説明に用いるテストシナリオ154、図12（b）は、図12（a）のテストシナリオ154に関連する画面の例である。また図13は、図12（b）の各画面に対応するソースコードの例である。

【0073】

図12（a）に示すように、例示したテストシナリオ154には、次のA）～G）の内容が記述されている。

A）top.htmlを表示

B）ユーザ名入力(id="hitachi")

C）パスワード入力(pass="pass")

D）ログイン(id="login")ボタンを押す

E）テキストエリアに"前回ログイン時間"が表示されていること

F）(id="next")ボタンを押す

G）ログイン名に"hitachi"が表示されていること

【0074】

このテストシナリオ154は、トップ画面のテストを行った後、トップ画面から確認画面に遷移し、続いて確認画面のテストを行った後、確認画面からポータル画面に遷移し、最後にポータル画面のテストを行うものである。尚、この例に示すように、テストシナリオ154の記述には画面毎の区切りを示す情報が含まれていない。このため、ある画面のテストがいずれの行から始まりいずれの行で終わるのかを直接に特定することができない。

【0075】

以下、図11とともにテストシナリオ分割情報生成処理S 8 1 2について説明する。同図に示すように、まずテスト支援装置100は、記憶しているテストシナリオ154のうち未取得のものを一つを取得する（S 1 1 1 1）。

【0076】

続いて、テスト支援装置100は、取得したテストシナリオ154から、当該テストシナリオ154を実行した際に最初にテストの対象になる画面に関する記述を特定する（S 1 1 1 2）。ここでテストシナリオ154が図12（a）に例示したテストシナリオ154である場合、テスト支援装置100は、A）「top.htmlを表示」の「top.html」という記述と画面／ソースコード対応情報151とに基づき、「トップ画面」のテストが開始されると判定する。尚、テスト支援装置100が、テストシナリオ154の実行内容を判定する方法は必ずしも限定されず、例えば、「・・・ボタンを含むウィンドウを選択する」、「・・・というタイトルのウィンドウを選択する」等の記述に基づきテストが開始されると判定してもよい。

【0077】

続いて、テスト支援装置100は、後続行に画面の遷移を生じさせる記述があるか否かを判定する（S 1 1 1 3）。判定が真であれば（S 1 1 1 3：YES）、処理はS 1 1 1 4に進む。判定が偽であれば（S 1 1 1 3：NO）、処理はS 1 1 1 6に進む。

【0078】

例えば、テストシナリオ154が図12(a)に例示したテストシナリオである場合、テスト支援装置100は、「D) ログイン(id="login")ボタンを押す」の行について、「トップ画面」のソースファイルである「top.html」の記述から、「id="login"」のボタンが押されると「goLogin」関数が呼ばれ、また「トップ画面」のソースファイルである「top.js」の記述から「confirm.html」に遷移すると解釈し、当該行が画面の遷移を生じさせる記述であると判定する。またこのとき、テスト支援装置100は、画面／ソースコード対応情報151から、上記ボタンが押された場合の遷移先の画面が「確認画面」であることを特定する。

【0079】

S1114では、テスト支援装置100は、テストシナリオ154の記述を画面単位ブロックに分割し、分割した結果をテストシナリオ分割情報152として記憶する。

【0080】

S1115では、テスト支援装置100は、取得中のテストシナリオ154に後続行があるか否かを判定する。判定が真であれば(S1115: YES)、処理はS1113に戻り、後続行についてS1113からの処理を繰り返す。判定が偽であれば(S1115: NO)、処理はS1116に進む。

【0081】

図12(a)に例示したテストシナリオ154では、テスト支援装置100は、後続行の「F) 次へ(id="next")ボタンを押す」について、当該行が画面の遷移を生じさせる記述であると判定する。またテスト支援装置100は、図13に示す「確認画面」のソースファイル「confirm.html」の記述から、「id="next"」のボタンが押されると「goNext」関数が呼ばれ、また「確認画面」のソースファイル「confirm.js」の記述から、遷移先が「ポータル画面(portal.html)」であることを特定する。

【0082】

S1116では、テスト支援装置100は、未取得のテストシナリオ154があるか否かを判定する。判定が真であれば(S1116: YES)、処理はS1111に戻る。判定が偽であれば(S1116: NO)、処理は図8のS813に進む。

【0083】

尚、テストシナリオ分割情報生成処理S812により、図12(a)に示したテストシナリオ154は、例えば、図14に示すように、即ち、「トップ画面」に関するA)～D)のブロック、「確認画面」に関するE)～F)のブロック、「ポータル画面」に関するG)のブロック、の3つのブロックに分割される。

【0084】

尚、以上の説明では、テストシナリオ154の分割に際し、画面の構成部品であるボタンに関する記述を参照したが、文字や画像等のリンク部品等の他の記述を参照してもよい。

【0085】

以上、詳細に説明したように、本実施形態のテスト支援装置100は、テストシナリオ154の記述を画面を単位として複数のブロックに分割したテストシナリオ分割情報152を生成するので、テストシナリオ154の記述を画面単位で把握することが可能となり、例えば、テストシナリオ154の記述のうち開発画面に対応する部分を特定することが可能となる。このため、例えば、開発画面のテストの結果を早期に取得するにはいずれのテストシナリオ154を優先して実行すればよいか、開発画面の不具合に関する情報を早期に取得するにはいずれのテストシナリオ154を優先して実行すればよいか等の判定が可能となる。そして判定の結果に応じて複数のテストシナリオの夫々に実行優先度を設定することで、ユーザが求める情報を迅速かつ効率よく提供することができる。

【0086】

尚、例えば、図6に示した4つのテストシナリオ154について、ユーザがテスト結果を取得しようとしている開発画面がB画面である場合に、「画面ホップ数」を観点として

10

20

30

40

50

実行優先度を設定すれば、テストシナリオ（２）→テストシナリオ（１）→テストシナリオ（３）の順となる。これを例えば、テストシナリオ（４）→テストシナリオ（１）→テストシナリオ（２）→テストシナリオ（３）の順にテストシナリオが実行される場合と比較すると、まずテストシナリオ（４）を実行しないことでテストシナリオ（４）の実行時間が削減される。またテストシナリオ（２）をテストシナリオ（１）よりも先に実行することで、テストシナリオ（１）のＡ画面のテストの時間分が削減されることとなり、ユーザは早期にＢ画面のテスト結果を取得することができる。

【００８７】

以上、本発明者によってなされた発明を実施の形態に基づき具体的に説明したが、本発明は上記の実施の形態に限定されるものではなく、その要旨を逸脱しない範囲で種々変更可能であることはいうまでもない。例えば、上記の実施の形態は本発明を分かりやすく説明するために詳細に説明したものであり、必ずしも説明した全ての構成を備えるものに限定されるものではない。また、上記実施形態の構成の一部について、他の構成の追加・削除・置換をすることが可能である。

10

【００８８】

例えば、本発明は、例えば、ソフトウェアによって表示される画面のレビューに際し特定の画面を迅速に表示させたい場合等、様々な目的に適用することができる。

【００８９】

また、上記の各構成、機能、処理部、処理手段等は、それらの一部または全部を、例えば、集積回路で設計する等によりハードウェアで実現してもよい。また、上記の各構成、機能等は、プロセッサがそれぞれの機能を実現するプログラムを解釈し、実行することによりソフトウェアで実現してもよい。各機能を実現するプログラム、テーブル、ファイル等の情報は、メモリやハードディスク、ＳＳＤ（Solid State Drive）等の記録装置、またはＩＣカード、ＳＤカード、ＤＶＤ等の記録媒体に置くことができる。

20

【００９０】

また、上記の各図において、制御線や情報線は説明上必要と考えられるものを示しており、必ずしも実装上の全ての制御線や情報線を示しているとは限らない。実際にはほとんど全ての構成が相互に接続されていると考えてもよい。

【符号の説明】

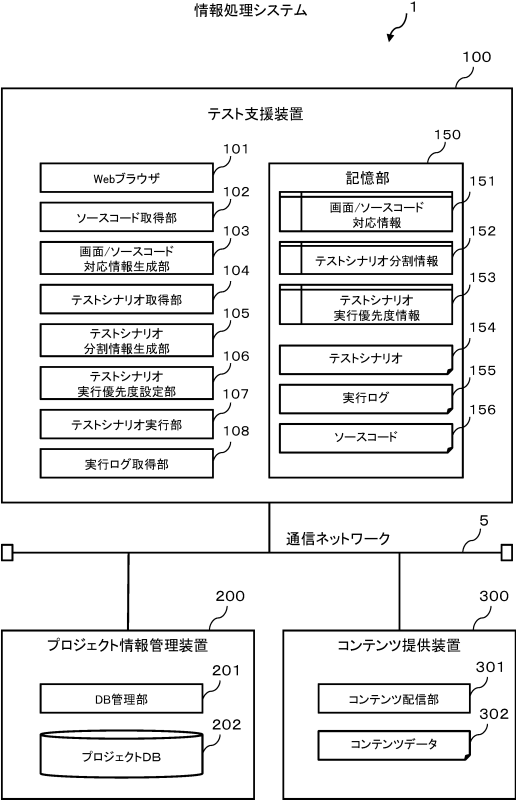
【００９１】

５ 通信ネットワーク、１００ テスト支援装置、１０１ Ｗｅｂブラウザ、１０２ ソースコード取得部、１０３ 画面／ソースコード対応情報生成部、１０４ テストシナリオ取得部、１０５ テストシナリオ分割情報生成部、１０６ テストシナリオ実行優先度設定部、１０７ テストシナリオ実行部、１０８ 実行ログ取得部、１５０ 記憶部、１５１ 画面／ソースコード対応情報、１５２ テストシナリオ分割情報、１５３ テストシナリオ実行優先度情報、１５４ テストシナリオ、１５５ 実行ログ、１５６ ソースコード、２００ プロジェクト情報管理装置、２０１ ＤＢ管理部、２０２ プロジェクトＤＢ、３００ コンテンツ提供装置、３０１ コンテンツ配信部、３０２ コンテンツデータ、Ｓ８００ テストシナリオ実行処理、Ｓ８１１ 画面／ソースコード対応情報生成処理、Ｓ８１２ テストシナリオ分割情報生成処理

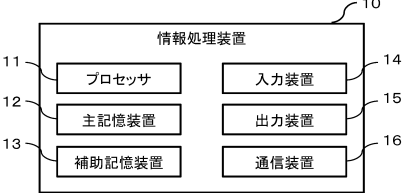
30

40

【図 1】



【図 2】

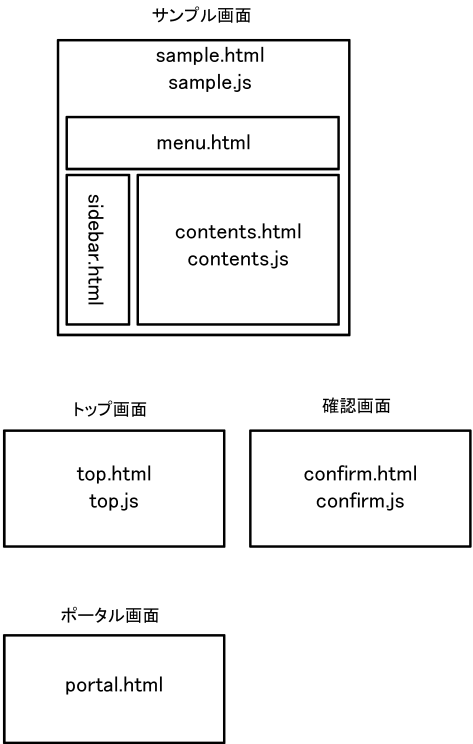


【図 3】

151 画面/ソースコード対応情報

画面ID 1511	ソースコードID 1512
サンプル画面	sample.html
	sample.js
	menu.html
	sidebar.html
	contents.html
	contents.js
トップ画面	top.html
	top.js
確認画面	confirm.html
	confirm.js
ポータル画面	portal.html
:	:

【図 4】

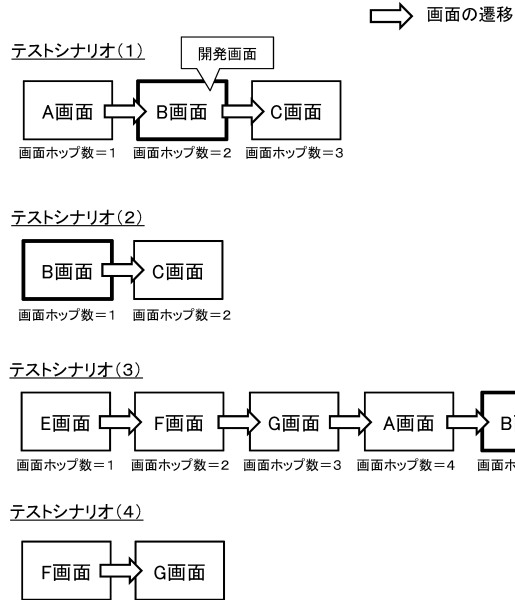


【図 5】

152 テストシナリオ分割情報

テストシナリオID 1521	画面ホップ数 1522			
	1	2	3	...
1	トップ画面	確認画面	ポータル画面	—
:	:	:	:	:

【図 6】



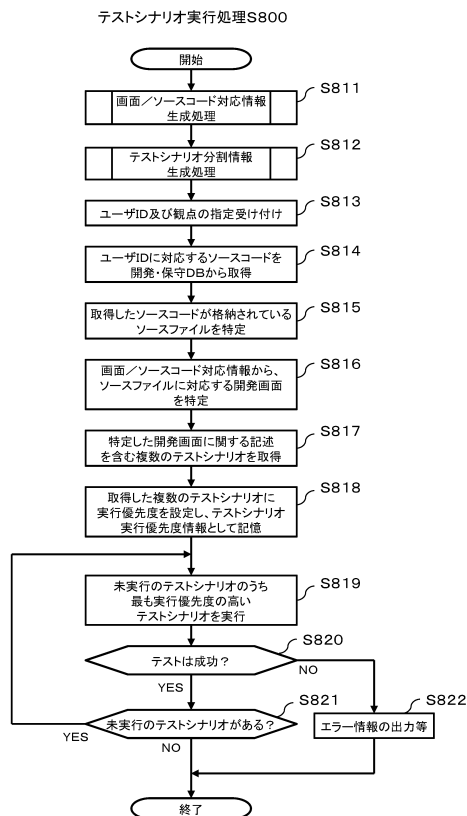
【図 7】

153

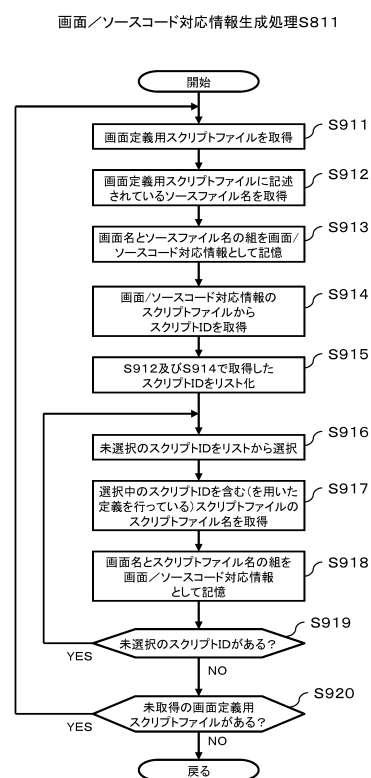
テストシナリオ実行優先度情報

テスト シナリオ 項番	トップ画面				...
	画面 ホップ数	到達時間	不具合 発生回数	...	
1	2	1	1	...	...
2	1	2	3	...	...
3	3	3	2	...	...
⋮	⋮	⋮	⋮	⋮	⋮

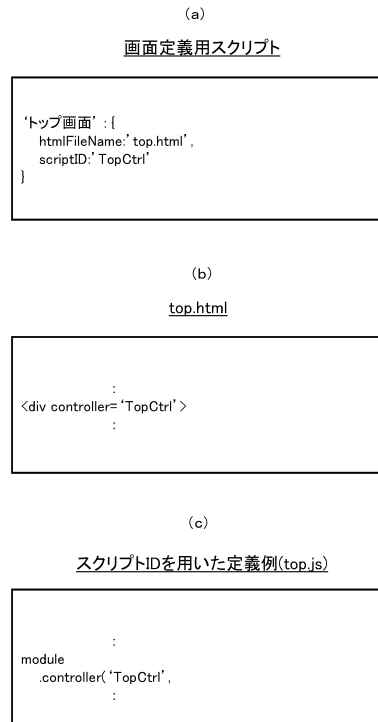
【図 8】



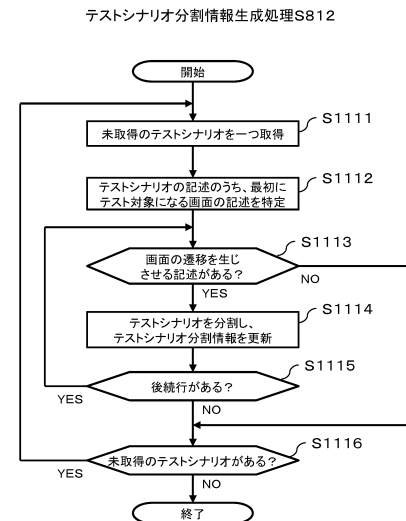
【図 9】



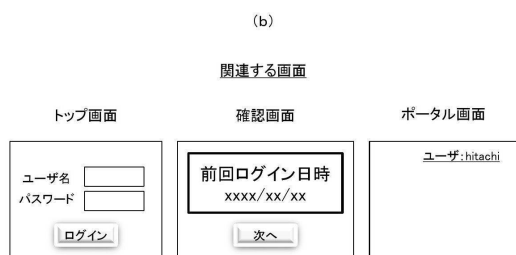
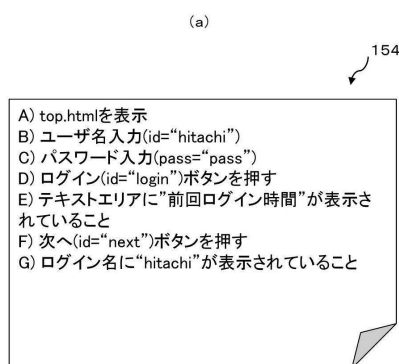
【図 10】



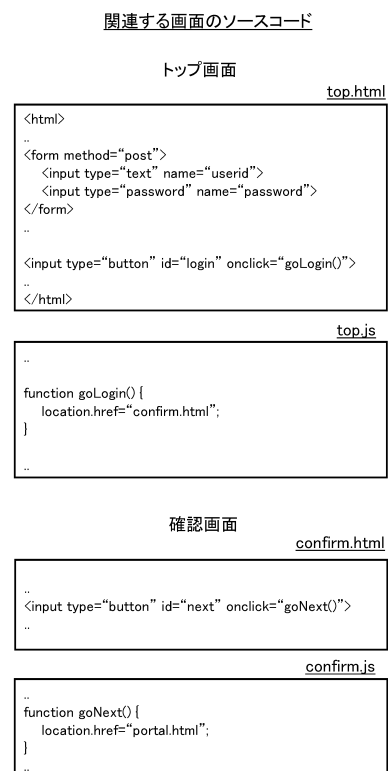
【図 11】



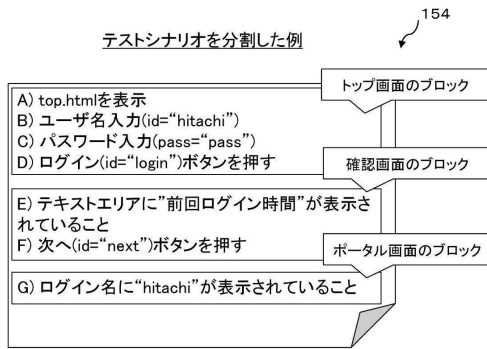
【図 12】



【図 13】



【図 1 4】



フロントページの続き

審査官 多賀 実

(56)参考文献 特開2013-077159 (JP, A)

(58)調査した分野(Int.Cl., DB名)
G06F 11/36