

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2004-342105

(P2004-342105A)

(43) 公開日 平成16年12月2日(2004.12.2)

(51) Int.Cl.⁷

G06F 12/00

G06F 17/21

F I

G06F 12/00

5 4 6 R

G06F 17/21

5 3 0 T

テーマコード (参考)

5 B 0 0 9

5 B 0 8 2

審査請求 有 請求項の数 20 O L (全 13 頁)

(21) 出願番号 特願2004-141156 (P2004-141156)
 (22) 出願日 平成16年5月11日 (2004.5.11)
 (31) 優先権主張番号 10/439867
 (32) 優先日 平成15年5月15日 (2003.5.15)
 (33) 優先権主張国 米国 (US)

(71) 出願人 390009531
 インターナショナル・ビジネス・マシー
 ズ・コーポレーション
 INTERNATIONAL BUSIN
 ESS MACHINES CORPO
 RATION
 アメリカ合衆国10504 ニューヨーク
 州 アーモンク ニュー オーチャード
 ロード
 (74) 代理人 100086243
 弁理士 坂口 博
 (74) 代理人 100091568
 弁理士 市位 嘉宏
 (74) 代理人 100108501
 弁理士 上野 剛史

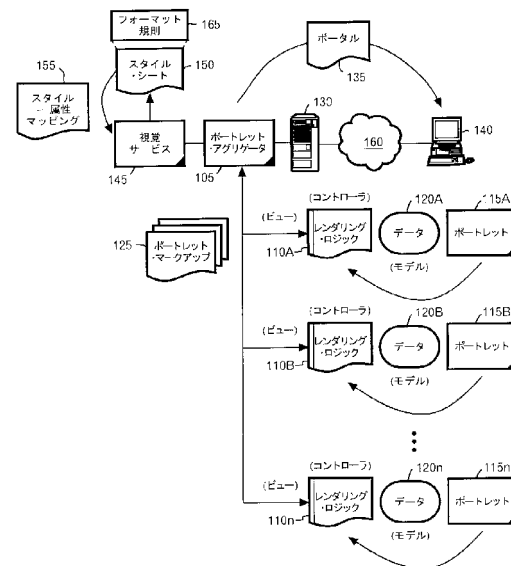
最終頁に続く

(54) 【発明の名称】 パーベイスプ・エージェントにおけるポートレット・スタイル適合

(57) 【要約】

【課題】ポータル・サーバ・システムおよび方法を提供すること。

【解決手段】ポータル・サーバ・システムは、1つまたは複数のポートレットに結合されたポータルを備えることができ、各ポートレットは関連するポートレット・レンダリング・ロジックを有する。このシステムはまた、ポートレット・レンダリング・ロジックと通信可能にリンクされたポートレット・アグリゲータも備えることができる。最後に、ポートレット・アグリゲータに対する視覚サービス・エクステンションを備えることができる。視覚サービス・エクステンションは、ポートレット・レンダリング・ロジックを処理してポートレット・レンダリング・ロジック中の視覚スタイル属性をマークアップ言語タグに変換するようにプログラムすることができ、これらのマークアップ言語タグは、指定のタイプのパーベイスプ・エージェント中で表示されるようにレンダリングすることができる。いずれにしても、好ましくは、ポートレット・レンダリング・ロジックはJava (R) サーバ・ページ (JSP) とすることができる。



【特許請求の範囲】

【請求項 1】

ポータル・ビューをパーベイスブ・エージェント中で表示するためにポータル・スタイルに従ってレンダリングする方法であって、

ポータルのためのテーマを定義するスタイルシートをロードするステップと、

前記定義されたテーマに関する視覚レンダリング属性を、特定タイプのパーベイスブ・エージェントに特有のマークアップ言語タグにマッピングするステップと、

指定のポートレットのためのポートレット・レンダリング・ロジックを解析して、埋め込まれたスタイル属性参照を識別するステップと、

前記ポートレット・レンダリング・ロジック中の前記埋め込まれたスタイル属性参照のうちの選択された参照を、前記マークアップ言語タグのうちのマッピングされるタグで置き換えるステップと、

前記ポートレット・レンダリング・ロジックを、前記指定のポートレットについてのビューを生成する際に使用されるようにコンパイルするステップと

を含む、方法。

【請求項 2】

サポートされないタグを前記ポートレット・レンダリング・ロジックから除去するステップをさらに含む、請求項 1 に記載の方法。

【請求項 3】

前記除去するステップが、前記サポートされないタグをサポートされるタグで置き換えるステップを含む、請求項 2 に記載の方法。

【請求項 4】

前記除去するステップが、前記サポートされないタグを削除するステップを含む、請求項 2 に記載の方法。

【請求項 5】

前記ロードするステップが、

前記ポートレットについてのユーザ・エージェントと、前記ポートレットについてのマークアップ言語タイプとのうちの一方を識別するステップと、

前記ユーザ・エージェントと前記マークアップ言語タイプとのうちの前記識別された一方に対応するスタイルシートをロードするステップと

を含む、請求項 1 に記載の方法。

【請求項 6】

前記置き換えるステップがさらに、前記ポートレット・レンダリング・ロジック中の前記スタイル属性参照のうちの選択された参照を、前記視覚レンダリング属性のうちのマッピングされる属性に従って、一連のネストされたスタイル属性参照から複合タグを再帰的に分解するように構成されたスクリプトレットで置き換えるステップを含む、請求項 1 に記載の方法。

【請求項 7】

関連するポートレット・レンダリング・ロジックをそれぞれが有する複数のポートレットに結合されたポータルと、

前記ポートレット・レンダリング・ロジックに通信可能にリンクされたポートレット・アグリゲータと、

前記ポートレット・アグリゲータに対する視覚サービス・エクステンションとを備え、

前記視覚サービス・エクステンションは、前記ポートレット・レンダリング・ロジックを処理して前記ポートレット・レンダリング・ロジック中の視覚スタイル属性をマークアップ言語タグに変換するようにプログラムされ、前記マークアップ言語タグは、指定のタイプのパーベイスブ・エージェント中で表示されるようにレンダリングすることができる、

ポータル・サーバ・システム。

【請求項 8】

10

20

30

40

50

前記ポートレット・レンダリング・ロジックが、J a v aサーバ・ページ（J S P）である、請求項7に記載のポータル・サーバ・システム。

【請求項9】

前記視覚サービス・エクステンションが、視覚スタイル属性からマークアップ言語タグへのマッピングを含む、請求項7に記載のポータル・サーバ・システム。

【請求項10】

前記パーベイスブ・エージェントの特定の特性に対して前記マッピングを選択的に指定するフォーマット規則のセットをさらに備える、請求項7に記載のポータル・サーバ・システム。

【請求項11】

前記特定の特性が、ユーザ・エージェントおよびマークアップ言語タイプからなるグループから選択される特性である、請求項10に記載のポータル・サーバ・システム。

【請求項12】

ポートレット・ビューを単一のポータル・ビューに集約するように構成されたポートレット・アグリゲータと、

前記ポータル・アグリゲータに対する視覚サービス・エクステンションとを備え、

前記視覚サービス・エクステンションは、前記ポートレット・ビューのうちの選択されたポートレット・ビューのためのポートレット・レンダリング・ロジックを処理して、前記ポートレット・レンダリング・ロジック中の視覚スタイル属性をマークアップ言語タグに変換するようにプログラムされ、前記マークアップ言語タグは、指定のタイプのパーベイスブ・エージェント中で表示されるようにレンダリングする、

ポータル・サーバ。

【請求項13】

前記ポートレット・レンダリング・ロジックが、J a v aサーバ・ページ（J S P）である、請求項12に記載のポータル・サーバ。

【請求項14】

前記視覚サービス・エクステンションが、視覚スタイル属性からマークアップ言語タグへのマッピングを含む、請求項12に記載のポータル・サーバ。

【請求項15】

ポータル・ビューをパーベイスブ・エージェント中で表示するためにポータル・スタイルに従ってレンダリングするためのコンピュータ・プログラムをストアしたマシン可読記憶装置であって、前記コンピュータ・プログラムは、

ポータルのためのテーマを定義するスタイルシートをロードするステップと、

前記定義されたテーマに関する視覚レンダリング属性を、特定タイプのパーベイスブ・エージェントに特有のマークアップ言語タグにマッピングするステップと、

指定のポートレットのためのポートレット・レンダリング・ロジックを解析して、埋め込まれたスタイル属性参照を識別するステップと、

前記ポートレット・レンダリング・ロジック中の前記埋め込まれたスタイル属性参照のうちの選択された参照を、前記マークアップ言語タグのうちのマッピングされるタグで置き換えるステップと、

前記ポートレット・レンダリング・ロジックを、前記指定のポートレットについてのビューを生成する際に使用されるようにコンパイルするステップと

をマシンに実施させるための命令のルーチン・セットを含む、マシン可読記憶装置。

【請求項16】

サポートされないタグを前記ポートレット・レンダリング・ロジックから除去するステップをさらに含む、請求項15に記載のマシン可読記憶装置。

【請求項17】

前記除去するステップが、前記サポートされないタグをサポートされるタグで置き換えるステップを含む、請求項16に記載のマシン可読記憶装置。

【請求項18】

10

20

30

40

50

前記除去するステップが、前記サポートされないタグを削除するステップを含む、請求項 16 に記載のマシン可読記憶装置。

【請求項 19】

前記ロードするステップが、

前記ポートレットについてのユーザ・エージェントと、前記ポートレットについてのマークアップ言語タイプとのうちの一方を識別するステップと、

前記ユーザ・エージェントと前記マークアップ言語タイプとのうちの前記識別された一方に対応するスタイルシートをロードするステップと

を含む、請求項 15 に記載のマシン可読記憶装置。

【請求項 20】

前記置き換えるステップがさらに、前記ポートレット・レンダリング・ロジック中の前記スタイル属性参照のうちの選択された参照を、前記視覚レンダリング属性のうちのマッピングされる属性に従って、一連のネストされたスタイル属性参照から複合タグを再帰的に分解するように構成されたスクリプトレットで置き換えるステップを含む、請求項 15 に記載のマシン可読記憶装置。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、ポータルおよびポートレットの分野に関し、より詳細には、パーベイスブ・エージェントの表示におけるポータル・ビューのスタイリングに関する。

【背景技術】

【0002】

大規模なコンピュータ通信ネットワーク中でコンテンツを配信することには、困難がつきものである。特に、コンピュータ通信ネットワーク内で配信するのに利用可能なコンテンツの量は、コンピュータ通信ネットワークのサイズに比例して変わることが多い。極端な場合、インターネットは、ほとんどのエンドユーザからは容易にアクセスできない多量のコンテンツをホストする。ポータルは、ネットワークアドレス可能な単一場所のチャネル・パラダイムを介して、コンテンツを集約する問題に対する賢明な解決法を表すものである。このためポータルは、コンテンツ配信において人気のあるものになっている。

【0003】

ポートレットは、ポータル・ページの一部として含まれる、目に見えるアクティブ・コンポーネントである。ウィンドウ環境のオペレーティング・システムのグラフィカル・ウィンドウ・パラダイムと同様、ポータル中の各ポートレットがポータル・ページの一部を占め、ポートレットは、ポータル・ページを介してポートレット・チャネルからの関連コンテンツを表示することができる。ポートレットは、電子メール・クライアントなどの単純なアプリケーションと、顧客関係管理システムからの予測出力などのより複雑なアプリケーションとの両方を含むことが知られている。典型的なポートレットは、ポータル・サーバを介して実行されるサーバ側スクリプトとして実装することができる。

【0004】

エンドユーザから見ると、ポートレットは、エンドユーザが加入することのできるコンテンツ・チャネルまたはアプリケーションである。これに比べて、コンテンツ・プロバイダから見ると、ポートレットは、個人に特化した形でコンテンツを加入エンドユーザに配信することのできる手段である。最後にポータルの視点からは、ポートレットは単に、ポータル・ページ内にレンダリングすることのできるコンポーネントである。いずれにしても、個々に選択可能かつ構成可能な 1 つまたは複数のポートレットをポータル内に提供することにより、ポータル・プロバイダは、統一されたインターフェースを介して、エンドユーザの好みに応じて個人に特化した形でコンテンツおよびアプリケーションを配信することができる。

【0005】

従来、ポータルへのアクセスは、デスクトップ・ブラウザ・アプリケーションを介して

10

20

30

40

50

行うことができる。ブラウザは、コンテンツにスタイルシートを適用してアプリケーション間の視覚体裁を適合させる能力を含めて、強力なレンダリング機能を提供することができるので、「リッチ・クライアント」と呼ばれてきた。しかし最近、ハンドヘルド・コンピュータや携帯電話などのパーベイスブ・デバイスが、消費者および企業の市場にかなり浸透し始めている。さらに、パーベイスブ・デバイスは、好まれるユーザ・インターフェースになってきた。したがって、資源の限られたこれらのデバイスにとって、ポータル・ソフトウェアがますますターゲットになっていくであろう。

【0006】

今日、カスケード・スタイルシート（CSS）などのスタイルシートが、デスクトップ・ポータル用の異種アプリケーションにまたがってカラーリングおよびスタイリングの方式を適合させるための統一機構を提供している。さらに、様々なスタイルシートを選択することにより、新しいテーマを動的にポータル・インターフェースに適用することができる。より具体的には、「スタイル・ガイドライン」を定義することによって新しいテーマを動的にポータル・インターフェースに適用することができ、スタイル・ガイドラインでは、ポートレット・ビューに含まれるハイパーテキスト・マークアップ言語（HTML）マークアップ・タグが、名前を特定のスタイリング方式に関連付けるクラス属性を含む。

【発明の開示】

【発明が解決しようとする課題】

【0007】

このようなマークアップ・タグの例として、`<pclass="myTextStyle">My Text</p>`を挙げることができる。スタイルシート定義「`.myTextStyle{color:#000000;font-weight:bold;font-size:12pt;}`」をクライアント側で適用することができ、テキスト「My Text」を12サイズの黒色ボールド・フォントとして表示することができる。しかし、従来のハンドヘルド・コンピュータなど、表示資源の限られたパーベイスブ・デバイスは、このスタイルシート・レンダリング機能をサポートしない。したがって、ポータルがパーベイスブ・デバイスをサポートする方向に移行するのに伴って、アプリケーションにまたがる一貫した視覚的集約を維持するため、ならびに様々なスタイリング・テーマに動的に適合する能力をサポートするために、これらの制限されたスタイリング機能に適応する方法が必要とされている。

【0008】

従来のポートレットは、モデル・ビュー・コントローラ方式で使用されるように構成されている。モデル・ビュー・コントローラ方式では、モデルがポートレットに関する基礎データを管理することができ、ビューがインターフェースを介してデータを提示することができ、コントローラがビューにおけるデータの使用および提示を管理することができる。モデル・ビュー・コントローラを用いてパーベイスブ・デバイスをサポートするには、コントローラが、パーベイスブ・デバイスの固有のタイプごとに適したビューを生成することが必要になる。しかし、固有のデバイス・タイプごとに複数のコントローラを生成することは、スケーラブルだとはとても思えない。別法として、要求元デバイスのユーザ・エージェントを、サポートされるデバイスのリストと比較することもできる。各デバイス・タイプごとにJava（R）サーバ・ページ（JSP）などの固有のレンダリング・ロジックを含め、それに従ってコンテンツをレンダリングすることもできる。しかしやはり、レンダリング・ロジックの複数のセットをポートレット・コードに組み込むことは、厄介であり、保守およびアップグレードが困難である。

【課題を解決するための手段】

【0009】

本発明は、ハンドヘルド・コンピューティング・デバイスを含めた異種パーベイスブ・エージェントにまたがってポートレットのスタイルを適合させるためのシステム、方法、および装置である。本発明によれば、レンダリング・ロジックの内容を、例えばサーバ・ページ・カスタム・タグを使用してコントローラによってコンパイルされる前にインターセプトすることができる。このレンダリング・ロジックを、特定のデバイスに対して記述

10

20

30

40

50

された別指定の規則のセットに基づいて修正することができる。修正されたレンダリング・ロジックは、その後、実行されるようにポータル集約ロジックにサブミットすることができる。このようにして、レンダリング・ロジックに手動変更を加える必要なしに、また異種の各パーベイス・デバイス用に新しいコントローラを生み出す必要なしに、各ターゲット・パーベイス・デバイスの特殊化された固有の閲覧要件に対応することができる。

【0010】

ポータル・ビューをパーベイス・エージェント中で表示するためにポータル・スタイルに従ってレンダリングする方法は、ポータルのためのテーマを定義するスタイルシートをロードすることを含むことができる。定義されたテーマに関する視覚レンダリング属性を、特定タイプのパーベイス・エージェントに特有のマークアップ言語タグにマッピングすることができる。指定のポートレットのためのポートレット・レンダリング・ロジックを解析して、埋め込まれたスタイル属性参照を識別することができる。ポートレット・レンダリング・ロジック中の埋め込まれたスタイル属性参照のうちの選択された参照を、マークアップ言語タグのうちのマッピングされるタグで置き換えることができる。最後に、ポートレット・レンダリング・ロジックを、指定のポートレットのためのビューを生成する際に使用されるようにコンパイルすることができる。

10

【0011】

特に、本発明の好ましい一態様では、サポートされないタグをポートレット・レンダリング・ロジックから除去することができる。これに関して、除去するステップは、サポートされないタグをサポートされるタグで置き換えるステップを含むことができる。あるいは、除去するステップは、サポートされないタグを削除するステップを含むことができる。本発明の別の好ましい態様では、ロードするステップは、ポートレットについてのユーザ・エージェントと、ポートレットについてのマークアップ言語タイプとの方の一方を識別することを含むことができる。その後、ユーザ・エージェントとマークアップ言語タイプとの方の一方の識別された一方に対応するスタイルシートをロードすることができる。最後に、本発明の別の好ましい態様では、置き換えるステップはさらに、ポートレット・レンダリング・ロジック中のスタイル属性参照のうちの選択された参照をスクリプトレットで置き換えるステップを含むことができ、スクリプトレットは、視覚レンダリング属性のうちのマッピングされる属性に従って、一連のネストされたスタイル属性参照から複合タグを再帰的に分解するように構成される。

20

30

【0012】

この発明の構成によれば、ポータル・サーバ・システムを提供することもできる。ポータル・サーバ・システムは、1つまたは複数のポートレットに結合されたポータルを備えることができ、各ポートレットは関連するポートレット・レンダリング・ロジックを有する。このシステムはまた、ポートレット・レンダリング・ロジックと通信可能にリンクされたポートレット・アグリゲータも備えることができる。最後に、ポートレット・アグリゲータに対する視覚サービス・エクステンションを備えることができる。視覚サービス・エクステンションは、ポートレット・レンダリング・ロジックを処理してポートレット・レンダリング・ロジック中の視覚スタイル属性をマークアップ言語タグに変換するようにプログラムすることができ、これらのマークアップ言語タグは、指定のタイプのパーベイス・エージェント中で表示されるようにレンダリングすることができる。いずれにしても、好ましくは、ポートレット・レンダリング・ロジックはJSPとすることができる。

40

【0013】

最後に、本発明によれば、ポータル・サーバを提供することができる。ポータル・サーバは、ポートレット・ビューを単一のポータル・ビューに集約するように構成されたポートレット・アグリゲータを備えることができる。さらに、このポータル・アグリゲータに対する視覚サービス・エクステンションも備えることができる。視覚サービス・エクステンションは、ポートレット・ビューのうちの選択されたポートレット・ビューのためのJSPなどのポートレット・レンダリング・ロジックを処理して、ポートレット・レンダリ

50

ング・ロジック中の視覚スタイル属性をマークアップ言語タグに変換するようにプログラムすることができ、これらのマークアップ言語タグは、指定のタイプのパーベিশブ・エージェント中で表示されるようにレンダリングすることができる。この目的で、視覚サービス・エクステンションは、視覚スタイル属性からマークアップ言語タグへのマッピングを含むことができる。例えば、

```
<styleTag:stylestylebean="bean">  
<p class="myTextStyle">MyText</p>  
</styleTag:style>
```

このマークアップでは、スタイル・ビーンの"bean"がコンパイルの直後に制御権を得ることができ、コンパイルから、動的スタイル定義に従ってデバイス特有のマークアップを生成することができる。 10

【0014】

現時点で好ましい実施形態を図面に示すが、本発明は、図示するとおりの構成および手段に限定されないことを理解されたい。

【発明を実施するための最良の形態】

【0015】

本発明は、ポータル・ビューを1つまたは複数のパーベিশブ・エージェント中で表示するために指定のスタイルに従ってレンダリングするためのシステム、方法、および装置である。本発明によれば、個々のポートレットのためのレンダリング・ロジックが、ポータル・サーバ内のポータル・アグリゲータ中の視覚サービスにアクセスすることができる。視覚サービスは、ポータルのためのスタイルシートをロードすることができ、関連するフォーマット規則に従って、視覚レンダリング属性を特定タイプのパーベিশブ・エージェントにマッピングすることができる。各ポートレットのためのレンダリング・ロジックは、ポートレットについてのビューを生成するときこのマッピングを利用することができる。このようにして、ターゲット・パーベিশブ・エージェントの性質が様々であるにもかかわらず、ポータルのスケーラビリティおよび管理容易性を犠牲にする必要なしに、コンテンツをポータル中で適切に一貫してレンダリングすることができる。 20

【0016】

図1は、ポートレットのコンテンツをパーベিশブ・エージェント中で表示するためにレンダリングするように構成されたポータル・サーバ・システムの概略図である。ポータル・サーバ・システムはポータル135を含むことができ、ポータル135は、ポータル・サーバ130を介して、精選されたポートレット115A、115B、115nに通信可能に結合される。各ポートレット115A、115B、115nは、対応するレンダリング・ロジック110A、110B、110nを介して、ポートレット・データ120A、120B、120nに基づいてポートレット・マークアップ125の形でビューを生成することができる。とりわけ、レンダリング・ロジック110A、110B、110nは、JSPなどのアクティブ・マークアップとすることができ、JSPでは、論理スクリプトレットを埋め込んで特定のマークアップ言語タグを生成することができる。 30

【0017】

ポートレット・マークアップ125を受け取り、ポートレット・マークアップをポータル135中のビューに集約するために、ポートレット・アグリゲータ105を各ポートレット115A、115B、115nに結合することができる。これに関して、ポータル135はサーバ130に配置されているものとすることができ、クライアント・デバイス140が、サーバ130から、ローカル・コンピュータ通信ネットワーク、例えばイントラネット、やグローバル・コンピュータ通信ネットワーク、例えばインターネットなどのコンピュータ通信ネットワーク160を介してポータル135にアクセスすることができる。とりわけ、クライアント・デバイス140には、従来のように構成された表示資源と、限られた表示資源を有するパーベিশブ・デバイスとの両方を含めることができる。 40

【0018】

本発明によれば、本明細書で視覚サービス145と呼ぶプログラマティック・スタイリ 50

ング・サービスをポートレット・アグリゲータ105に結合することができ、個々のレンダリング・ロジック110A、110B、110nからアクセスすることができる。より具体的には、ポートレット・アグリゲータ105を拡張して、例えばスタイル・ビーンやスタイルシート・オブジェクトなどの視覚サービス145を、まだセッションが存在しない場合のHttpSessionやHttpRequestオブジェクトなどのポータル要求オブジェクトに挿入することができる。(スタイルシート・オブジェクトがスタイル・ビーンの代わりに挿入された場合、レンダリング・ロジック110A、110B、110nは、スタイルシート・オブジェクトに基づいてスタイル・ビーンを生成することができる。)それにより、ポートレット115A、115B、115nのレンダリング・ロジック110A、110B、110nは、視覚サービス145にアクセスすることができる。 10

【0019】

視覚サービス145はさらに、ポータル135に関する1つまたは複数の視覚テーマを定義する1つまたは複数のスタイルシート150に結合させることができる。視覚サービス145は、スタイルシート150をロードすると、ポータル135に関する現在のスタイリング属性を決定し、この構造から、内部スタイル属性マッピング155を生成することができる。とりわけ、スタイルシート150が解析されるとき、スタイルシート150内で定義された各クラスは、複数のタグに当てはめることができる属性を含むことができる。例えば、クラス定義「.myText{background-color#EBEBEB; color:#666666;font-family:sans-serif; font-size:12pt;}」の場合に、タグが生成されるものとする 20
ことができる。この定義のcolor、font-family、font-sizeの各要素は、タグを介して指定される属性にマッピングするが、複数のタグがなおbackground-color属性を使用することができる。

【0020】

スタイルシート中のどの属性を特定のマークアップ・タグと共に使用すべきか決定するために、スタイルシート150をフォーマット規則プロパティ・ファイル165で増強することができる。フォーマット規則プロパティ・ファイル165は、例えばキーと値の対によって定義された、新しい出力を生成するための規則のセットを含むことができる。キーは、この例ではタグ名とすることができ、値は、そのタグおよびレンダリングすべき任意の子孫タグと共に使用されるスタイルシート属性とすることができ。フォーマット規則プロパティ・ファイル165は、一度生成すれば、特定のマークアップDTDについて 30
頻繁に変更する必要はない。

【0021】

スタイル属性マッピング155内で、スタイルシート属性名とマークアップ属性名との間のマッピングは、以下のような形をとる。

「style_sheet_attribute_name=html_attribute_name(s)」

したがって、例えばタグの場合のマッピングは、以下のマッピングを含むことができる。

「font=font-family=face, color=color,font-size=size」

ここでfont-family、color、font-sizeの各要素は、タグに当てはまる属性である。また、属性タグが生成されるとき、face要素は、スタイルシート中の要素font-familyの値に関する属性名を含むことができ、color要素は、スタイルシート中のcolor要素の値に関する属性名を含むことができ、size要素は、スタイルシート中のfont-size要素の値に関する属性名を含むことができる。 40

【0022】

フォーマット規則プロパティ・ファイル165が前述の例に限定されないことは、当業者には理解されるであろう。そうではなく、ポータル135がレンダリングされることになるデバイスのタイプに従って、その他の規則を定義することもできる。例えば別の例としては、フォーマット規則プロパティ・ファイル165の中の規則は、テーブル行タグ、例えば「<td>.:td=background-color=bgcolor; descendants=font」をどのように処理す 50

るかを識別することができる。それにより、「<tdclass="myText">」などの単一のタグを、C S S 使用不可デバイスに対して<tdbgcolor=#EBEBEB"> のように解釈することができることに留意されたい。

【0023】

したがって、種々のデバイスに対する種々の規則を処理するために、まずフォーマット規則プロパティ・ファイル165中で、特定のユーザ・エージェントのサブストリングがサフィックスに付いた名前を検索することができる。これに関して、「_&」で始まるサフィックスは、ユーザ・エージェントのサブストリングを識別することができる。サブストリング内のスペースはアンダースコアで表すことができる。例えば、「Windows (R) CE」というフレーズをそのユーザ・エージェント・フィールド内に含む任意のデバイスに、name_&Windows_CE.propertiesを使用することができる。指定のユーザ・エージェントに対する特定のフォーマット規則をフォーマット規則プロパティ・ファイル165内で識別することができない場合は、マークアップ特有プロパティ・ファイルの検索を実施することができる。この結果、マークアップ特有プロパティ・ファイルのサフィックスにマークアップ名を付けることができる。

10

【0024】

例えば、特定のユーザ・エージェントに関するデバイス・プロパティが定義されていないか任意のHTMLデバイスに、name_HTML.propertiesを使用することができる。どちらの定義もフォーマット規則プロパティ・ファイル165中になかった場合は、サフィックスなしのプロパティ定義、例えばname.propertiesを使用することができる。したがって、フォーマット規則プロパティ・ファイル165に「tag map」という名前が付いている場合、フォーマット規則の検索順は、以下の順番を含むことができる。

20

- 1) tagmap_&useragentsubstring.properties
- 2) tagmap_markup.properties
- 3) tagmap.properties

この結果、デバイス特有からマークアップ特有、ポータル全体まで、フォーマット規則に幅広い細分性をもたらすことができる。これらの規則はまた、画像や色などサポートされないエンティティをどのように処理するかをポータルに指示することもできる。

【0025】

図2は、図1のポータル・サーバ・システム内で、ポートレット・コンテンツをパーベイスブ・エージェント中で表示するためにレンダリングするプロセスを示すフロー・チャートである。ブロック210で開始し、ポートレット・アグリゲータと協同して、視覚サービスのインスタンスを生成することができる。具体的には、ポータル・ビューのレンダリングにおけるポートレット集約段階の間に、集約スタイル・オブジェクトをインスタンス化することができる。スタイル・オブジェクトがインスタンス化されると、ポートレットのレンダリング・ロジックが、ポートレット要求オブジェクトを介してスタイル・オブジェクトにアクセスすることができる。いずれにしてもブロック220で、ポートレット・アグリゲータは、ポータル・テーマ・スタイルシートをロードして、ポータルについての現在のスタイリング属性を決定することができる。ブロック230で、アグリゲータは、この構造からスタイル属性マッピングを構築することができる。

30

40

【0026】

各ポートレットごとに、パーベイスブ・エージェントなどの選択されたクライアント中でポートレット・ビューをポータル・テーマ・スタイルシートに従ってレンダリングするのに適した属性およびロジックで、レンダリング・ロジックをコンパイル前に修正することができる。具体的には、まず決定ブロック240で、サポートされない視覚要素への参照をレンダリング・ロジックから除去するかどうかを決定することができる。除去する場合は、ブロック250で、レンダリング・ロジックの内容を解析することができ、サポートされない要素をサポートされる要素に変更するか、または完全に削除することができる。

【0027】

50

例えば、や<i>タグなど、任意のインライン・スタイリング・タグをレンダリング・ロジックから削除することができる。HTMLのようなマークアップ・ビューは、すでに従来のブラウザ用にオーサリングされているが、このようにしてマークアップ・ビューをサポートすることができる。さらに、開発者がレンダリング・ロジックに手動変更を行う必要なしに、従来のマークアップ・ビューをサポートすることができる。最後に、新たに生成されるビューは、最適化のために、XHTML strict など、よりスタイリング制限的なマークアップ言語に適合することができる。

【0028】

最後に、ブロック250でレンダリング・ロジックにスクラブ処理を施したかどうかにかかわらず、ブロック260で、ポータル・スタイル抽象化をビューに適用することができる。具体的には、前述のテーブル行(<td>)の例のように、レンダリング・ロジック内の単一のタグを複数のタグに拡張することができる。ポータル・スタイルシートのスタイリング属性をレンダリング・ロジック中にインラインするだけでは、スタイルシート・テーマを動的に選択できないことは、当業者には理解されるであろう。具体的には、レンダリング・ロジックを解析し修正した後、レンダリング・ロジックをサブレットにコンパイルしてサーバ上にキャッシュすることができる。この結果、後でテーマ変更があっても、キャッシュされたサブレットには適用されないことになる。

【0029】

複数タグの拡張と動的なスタイルシート・テーマ選択との両方を考慮するためには、前述のテーブル行の例に関して以下に述べるように、スタイル抽象化を再帰的にビューに適用することができる。前述の例では、JSPステートメント<tr class="myText">をスクリプトレット<%=styleBean.tag("tr","myText")%>で置き換えて、スタイル・ビーン集約サービスに影響を与えることができる。この抽象化により、JSPをサブレットにコンパイルした後には動的な決定を行うことができる。

【0030】

いずれにしても、すべてのタグ属性およびクラス属性を見つけて、前述のスクリプトレットで置き換えることができる。JSPがコンパイルされると、スタイル・ビーンは、動的スタイリング能力を維持するために以下の再帰的な3段階プロセスを実施する。より具体的には、3段階プロセスは第1に、初期マークアップ・タグ・フラグメントをスクリプトレット呼出しの代わりに挿入することを含むことができる。第2に、挿入されたフラグメントに、スタイル・マップによって指定されるようにタグ属性を付加することができる。最後に、挿入されたタグを完成させるためにスタイル子孫を生み出すことができる。

【0031】

例えば、まずスクリプトレット呼出しをマークアップ・タグの始まりの部分で置き換えることができる。「myText」の例では、マークアップ・タグ・フラグメント「<tr」を挿入することができる。その後、スタイル・マップに定義されるように、現在のスタイルシート・クラス名からの値で、現在のマークアップ・タグに属性を追加することができる。やはり「myText」の例では、マークアップ・タグ・フラグメント「bgcolor='EBEBEB'>」を挿入することができる。最後に、スタイル・マップによって定義されるように次のスタイル子孫を決定することができる。すべての子孫が尽きるまで、次のスタイル子孫について前述の各ステップを再帰的に再処理することができる。

【0032】

例えば、「myText」の例では、挿入されたタグ・フラグメントに付加される次のタグ・フラグメントは「font」を含むものとすることができる。したがって、「font」スタイル子孫について3段階プロセスを繰り返す結果、「<tr bgcolor='EBEBEB'> 」となる。このタイプの再帰的アルゴリズムにより、スタイル・ビーン実装に変更を加えなくても、あるいはスタイル・ビーン実装の内部に特殊化されたコードがなくても、例えばCHTMLなどの新しいマークアップ・タイプをプラグインすることができるという、包括的な解決法が可能になる。

【0033】

要約すると、ほとんどのパーベシブ・デバイスでは、スタイルシート・レンダリングはサポートされない。しかしポータル技術は、アプリケーション間で一貫性のある視覚体裁を提供するために、スタイルシート・レンダリングを当然のこととして広く組み込んでいる。このため、パーベシブ・デバイスのためのポータル・コンテンツのレンダリングに一貫性がなくなる。本発明のスタイル・ビーン・レンダリング技術を適用すれば、JSPのコンテンツを動的に変更することができ、それにより、ポータルのスタイル設定に忠実であり続けながらも、選択されたパーベシブ・デバイスに互換性のあるスタイリング・タグを生成することができる。したがって本発明を用いることにより、ポータル開発者は、従来のクライアント用に1度だけポートレット・コードを書けばよく、コードによってパーベシブ・デバイス中で正しいポータル体裁が生み出されるようにすることができる。 10

【0034】

本発明は、ハードウェア、ソフトウェア、またはハードウェアとソフトウェアの組合せにおいて実現することができる。本発明の実装形態は、1つのコンピュータ・システム内で中央化方式で実現することもでき、相互接続されたいくつかのコンピュータ・システムにわたって各種要素が散在する分散方式で実現することもできる。本明細書に述べた方法を実施するように適合されたものならどんな種類のコンピュータ・システムまたはその他の装置も、本明細書に述べた機能を実施するのに適している。

【0035】

ハードウェアとソフトウェアの典型的な組合せの1つは、コンピュータ・プログラムを備えた汎用コンピュータ・システムとすることができ、コンピュータ・プログラムは、ロードされて実行されると、本明細書に述べた方法を実施するようにコンピュータ・システムを制御する。本発明はコンピュータ・プログラム製品に組み込むこともでき、このコンピュータ・プログラム製品は、本明細書に述べた方法の実施を可能にするすべての特徴を備え、また、コンピュータ・システムにロードされると、これらの方法を実施することができる。 20

【0036】

このコンテキストにおけるコンピュータ・プログラムまたはアプリケーションとは、情報処理機能を有するシステムに特定の機能を実施させるように意図された命令セットを任意の言語、コード、または表記で任意に表現したものを意味し、命令セットは、特定の機能を直接に実施させるか、a)別の言語、コード、または表記に変換された後、あるいはb)異なる材料形式に再生された後、の一方または両方で実施させる。重要なことだが、本発明は、その趣旨または本質的な属性を逸脱することなく他の特定の形に組み入れることもでき、したがって、本発明の範囲を示すものとしては、前述の明細書ではなく添付の特許請求の範囲を参照すべきである。 30

【図面の簡単な説明】

【0037】

【図1】ポートレット・コンテンツをパーベシブ・エージェント中で表示するためにレンダリングするように構成されたポータル・サーバ・システムの概略図である。

【図2】ポートレット・コンテンツをパーベシブ・エージェント中で表示するためにレンダリングするプロセスを示すフロー・チャートである。 40

【符号の説明】

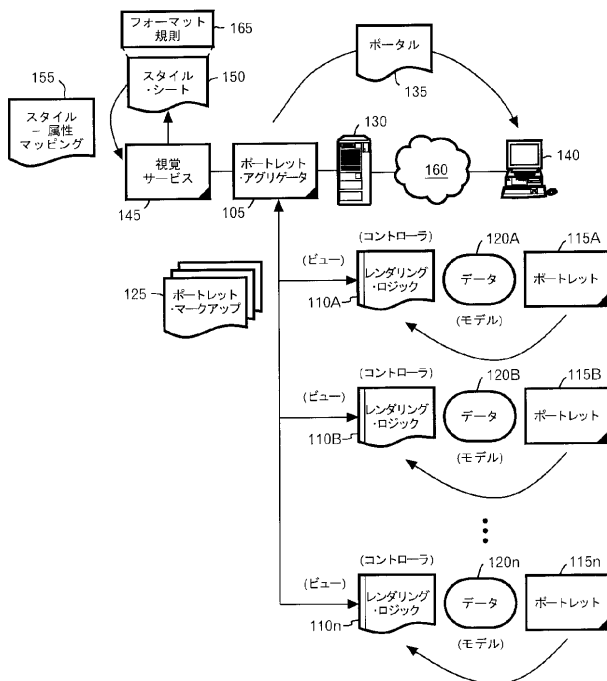
【0038】

- 105 ポートレット・アグリゲータ
- 110A レンダリング・ロジック
- 110B レンダリング・ロジック
- 110n レンダリング・ロジック
- 115A ポートレット
- 115B ポートレット
- 115n ポートレット

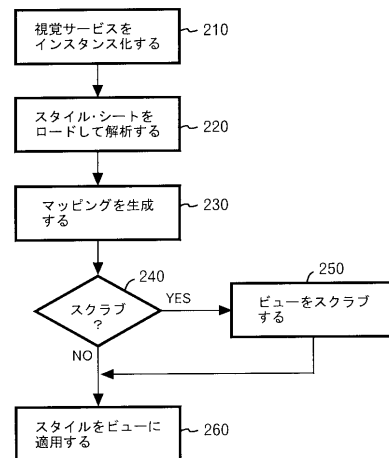
- 1 2 0 A データ
- 1 2 0 B データ
- 1 2 0 n データ
- 1 2 5 ポートレット・マークアップ
- 1 3 0 サーバ
- 1 3 5 ポータル
- 1 4 0 クライアント・デバイス
- 1 4 5 視覚サービス
- 1 5 0 スタイルシート
- 1 5 5 スタイル属性マッピング
- 1 6 0 コンピュータ通信ネットワーク
- 1 6 5 フォーマット規則プロパティ・ファイル

10

【図 1】



【図 2】



フロントページの続き

(72)発明者 ジェームズ・ラッセル・ゴッドウィン

アメリカ合衆国27613 ノースカロライナ州ラーレー グラーリン・ロード 9904

(72)発明者 マイケル・シー・ワンダースキー

アメリカ合衆国27613 ノースカロライナ州ダーラム ヘイヴェンツリー・ロード 1105

Fターム(参考) 5B009 NA04 QA06 RB01 TA11

5B082 GA00 GC02

【要約の続き】

【選択図】 図1