

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号
特許第4698599号
(P4698599)

(45) 発行日 平成23年6月8日 (2011.6.8)

(24) 登録日 平成23年3月11日 (2011.3.11)

(51) Int. Cl.

F I

G O 6 F 3/041 (2006.01)

G O 6 T 11/80 (2006.01)

G O 6 T 13/00 (2011.01)

G O 6 T 13/80 (2011.01)

G O 6 F 3/041 3 8 0 Q

G O 6 T 11/80 A

G O 6 T 13/00 B

請求項の数 5 (全 63 頁)

(21) 出願番号	特願2006-536576 (P2006-536576)	(73) 特許権者	500046438
(86) (22) 出願日	平成16年7月28日 (2004.7.28)		マイクロソフト コーポレーション
(65) 公表番号	特表2007-509413 (P2007-509413A)		アメリカ合衆国 ワシントン州 9805
(43) 公表日	平成19年4月12日 (2007.4.12)		2-6399 レッドモンド ワン マイ
(86) 国際出願番号	PCT/US2004/024194		クロソフト ウェイ
(87) 国際公開番号	W02005/045574	(74) 代理人	100077481
(87) 国際公開日	平成17年5月19日 (2005.5.19)		弁理士 谷 義一
審査請求日	平成19年7月26日 (2007.7.26)	(74) 代理人	100088915
(31) 優先権主張番号	60/513,591		弁理士 阿部 和夫
(32) 優先日	平成15年10月24日 (2003.10.24)	(72) 発明者	スティーブ ダッジ
(33) 優先権主張国	米国 (US)		アメリカ合衆国 98052 ワシントン
(31) 優先権主張番号	10/879,527		州 レッドモンド ワン マイクロソフト
(32) 優先日	平成16年6月30日 (2004.6.30)		ウェイ マイクロソフト コーポレーシ
(33) 優先権主張国	米国 (US)		ョン内

最終頁に続く

(54) 【発明の名称】 リアルタイムインキング

(57) 【特許請求の範囲】

【請求項 1】

構成要素に接続されるような動的描写オブジェクトを使用してコンピュータにインクを描写させる方法であって、前記同期描写オブジェクトは、インターフェースを有し、前記構成要素は、入力インターフェース、同期インターフェース、および非同期インターフェースを有し、前記方法は、

どのようにしてデータを処理するかを指示する前記構成要素からインクを受け取る工程であって、前記インクは、前記構成要素の前記同期インターフェースに接続される前記動的描写オブジェクトのインターフェースを介して受け取られる工程と、

表示装置の後続の表示のために前記インクを描写する工程であって、前記描写は、前記インクを前記コンピュータが受け取ることに同期して発生する工程と、

前記動的描写オブジェクトから前記インクを伝送して、描写された前記インクを一時的にキャッシュする工程と、

別の描写オブジェクトから通知を受け取る工程であって、前記別の描写オブジェクトは、前記構成要素の前記非同期インターフェースに接続されており、前記通知は、前記動的描写オブジェクトに前記別の描写オブジェクトが前記構成要素から前記インクに関する情報を受け取ったことを示す工程と、

前記通知に応答して、描写された前記インクのキャッシュをクリアする工程とを含むことを特徴とする方法。

【請求項 2】

前記描写工程は、スタイラスから流れる連続的なインクの概観をユーザに提示すること
を特徴とする請求項1に記載の方法。

【請求項3】

前記動的描写オブジェクトは、前記構成要素に接続された同期プラグイン集合の一部で
あることを特徴とする請求項1に記載の方法。

【請求項4】

前記描写オブジェクトに接続された追加オブジェクトにデータを送る工程をさらに含む
ことを特徴とする請求項1に記載の方法。

【請求項5】

請求項1乃至4の何れかに記載の方法を実行するためのプログラムを格納したコンピュ
ータ可読媒体。

10

【発明の詳細な説明】

【技術分野】

【0001】

本発明の態様は、画像処理および情報処理に関する。より詳細には、本発明の態様は、
電子インク情報を受け取って、処理することに関する。

【背景技術】

【0002】

本出願は、その内容が参照により本明細書に明確に組み込まれている、2003年10
月24日に出願された「Tablet Platform Controls and APIs」という名称の米国出願第
60/513,591号の優先権を主張するものである。

20

【0003】

典型的なコンピュータシステム、特にマイクロソフトWindows（登録商標）の如
きグラフィカルユーザインターフェース（GUI）システムを使用するコンピュータシ
ステムは、テキストを入力するためのキーボードの如き1つまたは複数の個別的な入力デバ
イス、およびユーザインターフェースを駆動するための1つまたは複数のボタンを備えた
マウスの如きポインティングデバイスからユーザ入力を受けつけるように最適化される。
汎用的なキーボードおよびマウスインターフェースは、文書、スプレッドシート、データ
ベースフィールド、図および写真等の高速作成および修正に対応する。しかし、コンピュ
ータを使用しない（すなわち標準の）ペンおよび紙に比べると、キーボードおよびマウス
インターフェースが与える柔軟性に大きなギャップがある。標準のペンおよび紙では、ユ
ーザは、文書を編集し、縁に注釈を書き、絵および他の形状等を描く。場合によっては、
ユーザは、キーボードおよびマウスインターフェースの範囲外で自由に注釈を加えること
ができるという理由から、画面上で文書を見直すより、ペンを使用して、文書にマーク付
けするのを好むこともある。

30

【0004】

ユーザは、ペンで書くときは、ペン先端からインクが流れるのを期待する。電子インク
の領域でも、同様の目標が存在する。電子インクがペンまたはスタイラスから流れるのを
妨げる1つの問題は、電子インクの扱い方である。

【0005】

40

従来のアプローチは、インキングのすべての側面を同時に扱うことを意図していた。従
来のアプローチは、インク情報が存在すれば、インクを描写し、ジェスチャを追跡し、イン
ク情報を他のインク情報と組み合わせ、インクを認識し、インクの外観を滑らかにする
等の処理を行うことを意図していた。これらの処理は、ユーザに対してインクを表示す
るのを遅らせることによって、より多くのインクを導入、より重要なことは、電子スタイ
ラスの先端からインクが流れるという幻想を低減または排除するのにユーザを待たせるもの
であった。

【発明の開示】

【発明が解決しようとする課題】

【0006】

50

本発明の態様は、上述の問題の1つまたは複数の問題に対処することにより、スタイラスからのインクの流れを提供する。

【課題を解決するための手段】

【0007】

動的描写オブジェクトをプロセス流れて移動させて、ユーザの動きと電子インクの表示との間の遅れを低減することができる。

【0008】

これらの態様および他の態様を、図面および関連説明に関連づけながら述べる。

【発明を実施するための最良の形態】

【0009】

本発明の態様は、スタイラスイベントを効率的に処理して、電子インクをユーザに対して迅速に描写することに関する。

【0010】

本文献は、読者に役立つようにいくつかのセクションに分割されている。これらのセクションは、インクの特性、用語定義、汎用的コンピューティング環境、リアルタイムインキングの概観、オブジェクトモデル、動的描写および湿式インク、ジェスチャ認識、同期および非同期プロセス、カスケードイング、動的プラグイン集合改良、誤差伝搬、管理／非管理実例、データ集合および流れ、データ同期、およびアプリケーションプログラミングインターフェースを含む。

【0011】

インクの特性

インクペンを使用するユーザに知られているように、物理的インク（インク溜を備えたペンを使用して紙上に書き込まれる種類）は、線分によって接続された一連の座標より多くの情報を伝達できる。例えば、物理的インクは、ペン圧（インクの濃さによる）、ペン角度（直線または曲線分の形状、および個別点付近のインクの挙動による）、およびペンの尖端の速度（真直性、線幅、および直線または曲線のコースに対する線幅変化による）を反映することができる。さらなる例は、インクが付着する紙または他の表面の繊維にインクが吸収される様式を含む。これらの些細な特性も、上記の特性を付与するに役立つ。これらの追加的な特性により、転換の均一な線幅を用いるより、情緒、個性および強調等をより即時的に付与することが可能である。

【0012】

電子インク（またはインク）は、ユーザがスタイラスをベースとした入力デバイスを使用するときに取り込まれる電子情報の取込みおよび表示に関する。電子インクは、筆画の配列または任意の集合を指し、各筆画は、点の配列で構成される。筆画は、同時に描画または収集されるか、または個別的な時刻および箇所、および個別的な理由で描画または収集されうる。デカルト座標（X、Y）、極座標（r、 θ ）、および当該技術分野で知られている他の技術を含む、知られている様々技術を用いて、それぞれの点を表現することができる。電子インクは、圧力、角度、速度、色、スタイラスサイズおよびインク透明性を含む実際のインクの特性の表示を含むことができる。電子インクは、ページにインクが付着された秩序（たいていの西洋言語では、左から右へ、そして下に続けるラスタパターン）、タイムスタンプ（インクが付着されたときを示す）、インクの作者の指標、および発祥デバイス（インクが引き出された機械の指標、またはインクを付着させるのに使用されたペンの指標）等の情報をさらに含むことができる。

【0013】

用語定義

【0014】

10

20

30

40

【表 1 - 1】

インク

特性を有する筆画の配列または集合。筆画の配列は、秩序的な形態の筆画を含むことができる。取り込まれた時刻によって、またはページ上で筆画が現れる箇所によって、またはインクの作者による協同的状况で配列を秩序化することができる。他の秩序も可能である。筆画の集合は、筆画の配列、または無秩序の筆画、またはそれらの任意の組合せを含むことができる。さらに、いくつかの特性（例えば、圧力、速度および角度等）は、各筆画または筆画における点に固有のものであってもよい。これらの特性は、インクレベルではなく、筆画または点のレベルで格納されうる。

10

インクオブジェクト

特性を有するインクまたは有さないインクを格納するデータ構造。

筆画

取り込まれた点の配列または集合。例えば、描写するときに、点の配列を線で結ぶことができる。あるいは、筆画を、点、および次の点の方向のベクトルとして表現することもできる。つまり、筆画は、点の基礎表現、および／または点を結んでいるものにかかわらず、インクに関する点または部分の任意の表現を包括することを意図する。

20

点

空間内の箇所を定める情報。例えば、点を、取込み空間（例えばデジタイザ上の点）、仮想のインク空間（取り込まれたインクが配置される空間の座標）、および／または表示空間（表示デバイスの点または画素）に対して相対的に定めることができる。

30

【 0 0 1 5】

【表 1 - 2】

文書

可視的な表現および内容を有する任意の電子ファイル。文書は、ウェブページ、ワードプロセッシング文書、ノートページまたはパッド、スプレッドシート、視覚的表現、データベース記録、画像ファイル、およびそれらの組合せを含むことができる。

リアルタイムスタイラス

リアルタイムスタイラスは、所定のウィンドウ入力枠内での所定のウィンドウ処理に対するリアルタイムスタイラスイベントを提供するオブジェクトである。リアルタイムスタイラスは、追加的な機能処理するプラグインオブジェクトが追加されることになる枠組と見なすことも可能である。リアルタイムスタイラスの所望の機能が変化することによって、プラグインオブジェクトを追加または除去することができる。リアルタイムスタイラスに対するAPIは、RTSと称することができる。RTSは、デベロッパが他の機能を結びつけることができる中核のAPI構成要素でありうる。RTSは、ペンサービスからの生データを一括し、そのデータを第1のプラグインオブジェクト（存在する場合）に送る。リアルタイムスタイラスは、1つまたは複数のインターフェースを有することができる。2つのインターフェースの場合は、これらは同期インターフェースおよび非同期インターフェースであってもよい。これらのインターフェースは、プラグインをリアルタイムスタイラスに接続できる場所を提供する。これらのインターフェースは、例示を目的としたものにすぎない。他のインターフェースを用いることもできる。

10

20

30

【0016】

【表 1－3】

リアルタイムインク

リアルタイムインクは、インクがスタイラスの先端から流れているという幻想である。従来のアプローチは、重要な処理工程をインクの描写と組み合わせることによって、新たに受け取られた筆画の表示を減速させようとするものであった。リアルタイムインクは、新たに受け取られる筆画を受け取りながら表示させ、ともに動作するリアルタイム処理工程とリアルタイム表示を分離することによって、インクをより速くユーザに表示しようとするものである。

10

プラグイン

プラグインは、リアルタイムスタイラスオブジェクトに追加できる機能的構成要素である。プラグインは、リアルタイムスタイラスオブジェクトの同期インターフェースに取り付けられる場合は、同期プラグインと称することができる。プラグインは、リアルタイムスタイラスオブジェクトの非同期インターフェースに取り付けられる場合は、非同期プラグインと称することができる。

ペンサービス

デジタイザハードウェアドライバと対話し、タブレット機能（例えばTabletAdded/Removed、StylusButtonUp/Down等）に関係するインク集合に無関係な他の通知とともに、標準「パケット」および関連構造に予め翻訳された生「パケット」データを提供するシステムサービス構成要素。ペンサービスは、プラグイン集合によってパケットを処理するための一次的な機構を提供する。

20

【0 0 1 7】

30

【表 1 - 4】

プラグイン集合

RTSに取りつけられる1つまたは複数のグループのプラグインの集合。2つの集合が存在する場合は、それぞれRTSの同期および非同期インターフェースに対応づけることができる。各集合を順次実行することができる。2つまたはそれ以上の集合が存在する場合は、それらをRTS（または複数のRTS）の異なるインターフェースに取りつけることができるため、互いに独立して実行することができる。データは、各プラグインによって処理された後にRTSに戻る。集合におけるプラグインの順序は、集合の出力に影響を与えることがある。

10

プラグインチェーン

プラグインの各グループが従属接続されたプラグインのチェーン。ここでは、データは、RTSに戻る前に、チェーンにおける各プラグインを通じてそれぞれの機能を果たす。チェーンは、プラグインの「デージチェーン」と称することもできる。チェーンまたはデージチェーンにおけるプラグインの順序は、チェーンの出力に影響を与えることがある。

20

入力キュー

同期プラグイン集合またはプラグインチェーンによる再処理のために、同期プラグイン集合またはプラグインチェーンによって生成されるパケットまたはオブジェクトに対する一次的な保留待ち行列。つまり、同期プラグイン集合またはプラグインチェーンは、オブジェクトを入力キューに押し込むことが可能である。

30

【 0 0 1 8 】

【表 1－5】

動的描写	ペンが画面に触れている間にインクを描画する漸進的プロセス。ペンがデジタイザを移動するに従って、「インク」の尾が画面の背後に残る。その効果は、インクが、描画されながらペンから流れているかのように現れる。インクは、データが受け取られるにつれて漸次に描画される。描写は、アンチエイリアシング等の追加的な効果を適用できる。	
静的描写	全インク筆画を同時に描写するプロセス。データは、インクを描画し、全筆画を単一単位として描画する前に既に利用可能である。静的描写は、Bezierスムーシングおよびアンチエイリアシングのような追加的な効果を適用できる。	10
動的描写	デベロッパは、場合によって、動的描写をインスタンス化して、リアルタイムパケットデータの高効率描写をGDI描写面に自動的に与えることができる。本発明の態様の実装形態の一部として提供されるGDIセントリックDynamicRendererと同じインターフェース規定を用いたDirectXまたはプロッタのような他の描写面に対する動的描写も可能である。	20
【0019】		
【表 1－6】		
ジェスチャレコグナイザ	デベロッパは、場合によって、ジェスチャレコグナイザをインスタンス化して、筆画のリアルタイム認識を行い、デベロッパが興味を示したジェスチャを1つまたは複数の筆画が構成するときを報告することができる。ジェスチャレコグナイザは、使用される場合は、同期または非同期集合またはチェーンに配置されうる。	30
出力キュー	パケットデータは、オブジェクトの集合またはチェーンを流れる（そして潜在的に改造される）と、RTSがそれを非同期プラグインまたはチェーンの集合に送る出力キューに置かれる。	
インク収集オブジェクト	リアルタイムスタイラスによって提供されるインクデータを蓄積・記憶する同期プラグインオブジェクトを示す。	40
IC0	非同期集合またはチェーンに存在しうるインク集合オブジェクト。出力キューからデータを受け取る。	

【0020】

汎用コンピューティング環境

図1Aは、本発明を実施できる好適なコンピューティングシステム環境100の例を示

50

す図である。コンピューティングシステム環境100は、好適なコンピューティング環境の一例にすぎず、本発明の用途または機能の範囲に関して何らかの制限を示唆することを意図するものではない。コンピューティングシステム環境100は、例示的なコンピューティングシステム環境100に示されている任意の1つの構成要素またはその組合せに関する何らかの依存性または要件を有するものとして解釈されるべきではない。

【0021】

本発明は、他の多くの汎用または特殊用途のコンピューティングシステム環境または構成で動作可能である。本発明への使用に好適でありうるよく知られているコンピューティングシステム、環境および構成の例としては、パーソナルコンピュータ、サーバコンピュータ、スマートフォン、スマートウォッチおよびパーソナルデータアシスタントを含むハンドヘルドまたはラップトップデバイス、マルチプロセッサシステム、マイクロプロセッサをベースとしたシステム、セットトップボックス、プログラム可能大衆消費電子製品、ネットワークPC、ミニコンピュータ、大型コンピュータ、および上記システムまたはデバイスのいずれかを含む分散コンピューティング環境等が挙げられるが、それらに限定されない。

10

【0022】

コンピュータによって実行されるプログラムモジュールの如きコンピュータ実行可能命令の全体的な脈絡で本発明を説明することができる。一般には、プログラムモジュールは、特定の作業を実施するか、または特定の抽象データタイプを実装するルーチン、プログラム、オブジェクト、構成要素およびデータ構造等を含む。通信ネットワークを通じて互いにリンクされる遠隔処理デバイスによってタスクが実施される分散コンピューティング環境で本発明を実施することもできる。分散コンピューティング環境では、メモリ記憶デバイスを含むローカルおよび遠隔コンピュータ記憶媒体にプログラムモジュールを配置することができる。

20

【0023】

図1Aを参照すると、本発明を実施するための例示的なシステムは、コンピュータ110の形の汎用コンピューティングデバイスを含む。コンピュータ110の構成要素としては、処理装置120、システムメモリ130、およびシステムメモリを含む様々なシステム構成要素を処理装置120に結合するシステムバス121を挙げることができるが、それらに限定されない。システムバス121は、メモリバスまたはメモリコントローラ、周辺バス、および様々なバスアーキテクチャのいずれかを使用したローカルバスを含むいくつかのバス構造体のいずれかであってもよい。限定ではなく、例として、当該アーキテクチャは、工業規格アーキテクチャ（ISA）バス、マイクロチャネルアーキテクチャ（MCA）バス、拡張ISA（EISA）バス、ビデオ電子装置規格化協会（VESA）ローカルバス、およびメザニンバスとしても知られている周辺装置相互接続（PCI）を含む。

30

【0024】

コンピュータ110は、典型的には、様々なコンピュータ読取り可能媒体を含む。コンピュータ読取り可能媒体は、コンピュータ110によってアクセスすることが可能である任意の利用可能な媒体で、揮発性媒体、不揮発性媒体、取外し可能な媒体および取外し不可能な媒体を含む。限定ではなく、例として、コンピュータ読取り可能媒体は、コンピュータ記憶媒体および通信媒体を含むことができる。コンピュータ記憶媒体は、コンピュータ読取り可能命令、データ構造、プログラムモジュールまたは他のデータの如き情報を記憶するための任意の方法または技術で実装される揮発性媒体、不揮発性媒体、取外し可能な媒体および取外し不可能な媒体を含む。コンピュータ記憶媒体としては、ランダムアクセスメモリ（RAM）、読取り専用メモリ（ROM）、電子的消去書込可能読取り専用メモリ（EEPROM）、フラッシュメモリまたは他のメモリ技術、CD-ROM、デジタルビデオディスク（DVD）または他の光ディスク記憶機構、磁気カセット、磁気テープ、磁気ディスク記憶機構または他の磁気記憶デバイス、あるいは所望の情報を記憶するのに使用でき、コンピュータ110によってアクセスできる任意の他の媒体が挙げられるが

40

50

、それらに限定されない。通信媒体は、典型的には、コンピュータ読取り可能命令、データ構造、プログラムモジュール、または搬送波の如き変調データ信号もしくは他の伝送機構内の他のデータを具現化し、任意の情報配信媒体を含む。「変調データ信号」という用語は、その1つまたは複数の特性が信号内の情報をコード化するように設定または変化した信号を意味する。限定ではなく、例として、通信媒体は、有線ネットワークまたは直線有線接続の如き有線媒体、ならびに音響、RF、赤外および他の無線媒体の如き無線媒体を含む。上記のいずれかの組合せもコンピュータ読取り可能媒体の範囲に含まれるものとする。

【0025】

システムメモリ130は、ROM131およびRAM132の如き揮発性および／または不揮発性メモリの形のコンピュータ記憶媒体を含む。起動時などにコンピュータ110内の要素間で情報を伝達するのに役立つ基本ルーチンを含む基本入出力システム133（BIOS）は、典型的にはROM131に記憶される。RAM132は、典型的には、すぐにアクセス可能で、かつ／または処理装置120によって現在動作されているデータおよび／またはプログラムモジュールを含む。限定ではなく、例として、図1Aは、オペレーティングシステム134、アプリケーションプログラム135、他のプログラムモジュール136およびプログラムデータ137を示す図である。

【0026】

コンピュータ110は、他の取外し可能／取外し不可能、および揮発性／不揮発性コンピュータ記憶媒体を含むこともできる。例示のみを目的とし、図1Aは、取外し不可能で不揮発性の磁気媒体に対して読取りまたは書込を行うハードディスクドライブ141、取外し可能で不揮発性の磁気ディスク152に対して読取りまたは書込みを行う磁気ディスクドライブ151、およびCD-ROM、または他の光媒体の如き取外し可能で不揮発性の光ディスク156に対して読取りまたは書込みを行う光ディスクドライブ155を示す図である。例示的な動作環境で使用できる他の取外し可能／取外し不可能および揮発性／不揮発性コンピュータ記憶媒体としては、磁気テープカセット、フラッシュメモリカード、デジタルビデオディスク、デジタルビデオテープ、固体RAMおよび固体ROM等が挙げられるが、それらに限定されない。ハードディスクドライブ141は、典型的には、インターフェース140の如き取外し不可能なメモリインターフェースを通じてシステムバス121に接続され、磁気ディスクドライブ151および光ディスクドライブ155は、典型的には、インターフェース150の如き取外し可能なメモリインターフェースによってシステムバス121に接続される。

【0027】

上記に説明し、図1Aに示したドライブ、およびそれらに伴うコンピュータ記憶媒体は、コンピュータ読取り可能命令、データ構造、プログラムモジュール、およびコンピュータ110のための他のデータの記憶機構を提供する。図1Aにおいて、例えば、ハードディスクドライブ141は、オペレーティングシステム144、アプリケーションプログラム145、他のプログラムモジュール146およびプログラムデータ147を記憶するものとして示されている。これらの構成要素は、オペレーティングシステム134、アプリケーションプログラム135、他のプログラムモジュール136およびプログラムデータ137と同じであっても異なってもよい。ここでは、オペレーティングシステム144、アプリケーションプログラム145、他のプログラムモジュール146およびプログラムデータ147にそれぞれ異なる番号を割り当てて、最低限、それらが異なるコピーであることを示す。ユーザは、デジタルカメラ163、キーボード162、および一般的にマウスまたはタッチパッドと称するポインティングデバイス161の如き入力デバイスを通じて、コンピュータ110に指令および情報を入力することができる。他の入力デバイス（不図示）としては、マイクロフォン、ジョイスティック、ゲームパッド、サテライトディッシュまたはスキャナ等を挙げることができる。これらの入力デバイスおよび他の入力デバイスは、システムバス121に結合されるユーザ入力インターフェース160を通じて処理装置120にしばしば接続されるが、並列ポート、ゲームポートまたはユニバー

10

20

30

40

50

サルシリアルバス（USB）によって接続されうる。モニタ191または他のタイプの表示デバイスも、ビデオインターフェース190の如きインターフェースを介してシステムバス121に接続される。モニタに加えて、コンピュータは、出力周辺インターフェース195を通じて接続できるスピーカ197およびプリンタ196の如き他の周辺出力デバイスを含むこともできる。

【0028】

コンピュータ110は、遠隔コンピュータ180の如き1つまたは複数の遠隔コンピュータEの論理接続を用いたネットワーク化環境で動作できる。遠隔コンピュータ180は、パーソナルコンピュータ、サーバ、ルータ、ネットワークPC、同位デバイスまたは他の共通ネットワークノードであってもよく、コンピュータ110に対して上述した要素の多くまたは全てを含むが、図1Aにはメモリ記憶デバイス181のみを示した。図1Aに示されている論理接続は、ローカルエリアネットワーク（LAN）171およびワイドエリアネットワーク（WAN）173を含むが、他のネットワークを含むこともできる。当該ネットワーク環境は、オフィス、企業内コンピュータネットワーク、イントラネットおよびインターネットに普及している。

10

【0029】

コンピュータ110は、LANネットワーク環境で使用されるときは、ネットワークインターフェースまたはアダプタ170を通じてLAN171に接続される。コンピュータ110は、WANネットワーク環境で使用されるときは、典型的には、モデム172、またはインターネットの如きWAN173に通信を確立するための他の手段を含む。内部または外部モデムであってもよいモデム172は、ユーザ入力インターフェース160または他の適切な機構を介してシステムバス121に接続されうる。ネットワーク化環境では、コンピュータ110またはその部分に対して示されているプログラムモジュールを遠隔メモリ記憶デバイスに記憶することができる。限定ではなく、例として、図1Aは、メモリデバイス181上に存在する遠隔アプリケーションプログラム185を示す図である。示されているネットワーク接続は、例示的なものであり、コンピュータの間に通信リンクを確立する他の手段を使用することが理解されるであろう。

20

【0030】

示されているネットワーク接続は、例示的なものであり、コンピュータの間に通信リンクを確立する他の手段を使用することが可能であることが理解されるであろう。TCP/IP、イーサネット（登録商標）、FTPおよびHTTP等のよく知られている様々なプロトコルのいずれかの存在が想定され、システムをクライアントサーバ構成で動作させて、ユーザがウェブをベースとしたサーバからウェブページを検索することを可能にすることができる。様々な従来のウェブブラウザのいずれかを使用して、ウェブページ上にデータを表示・操作することが可能である。

30

【0031】

プログラミングインターフェース（または、より簡潔にはインターフェース）を、コードの1つまたは複数のセグメントが、コードの1つまたは複数の他のセグメントによって提供される機能と通信するか、またはそれにアクセスすることを可能にするための任意の機構、プロセス、プロトコルと見なすことができる。あるいは、プログラミングインターフェースを、他の構成要素の1つまたは複数の機構、方法、機能、呼出およびモジュール等と通信結合することが可能なシステムの構成要素の1つまたは複数の機構、方法、機能、呼出およびモジュール等と見なすことができる。先述の文における「コードのセグメント」という用語は、コードの1つまたは複数の命令またはラインを含むことを意図し、例えば適用される用語、あるいはコードセグメントが個別に編集されるかどうか、あるいはコードセグメントが原コードとして提供されるか、中間コードとして提供されるか、または目的コードとして提供される、あるいはコードセグメントがルーチンシステムまたはプロセスで利用されるか、あるいはそれらが同じまたは異なる機械に配置されるか、または複数の機械に分配されるか、あるいはコードのセグメントによって表される機能が全てソフトウェアで実施されるか、全てハードウェアで実施されるか、ハードウェアとソフトウ

40

50

ェアの組合せで実施されるかにかかわらず、例えばコードモジュール、オブジェクト、サブルーチンおよび機能等を含む。

【0032】

概念的には、図1Bまたは図1Cに示されるように、プログラミングインターフェースを包括的に捉えることができる。図1Bは、第1および第2のコードセグメントが、それを通じて通信する導管としてのインターフェース、すなわちインターフェース1を示す図である。図1Cは、システムの第1および第2のコードセグメントが、媒体Mを介して通信することを可能にする、(第1および第2のコードセグメントの一部であってもなくてもよい) インターフェースオブジェクトI1およびI2を備えるインターフェースを示す図である。図1Cを見ると、インターフェースオブジェクトI1およびI2を同一システムの個別的なインターフェースと見なすことができるとともに、オブジェクトI1およびI2+媒体Mはインターフェースを備えると考えられることができる。図1Bおよび1Cは、双方向の流れと、流れの各々の側のインターフェースとを示しているが、特定の実装形態は、一方向にのみ情報の流れを有することができ(または、以下に示すように情報の流れを有さなくてもよく)、あるいは一方の側にのみインターフェースを有することができる。限定ではなく、例として、アプリケーションプログラミングインターフェース(API)、入口点、方法、機能、サブルーチン、遠隔手順呼出および構成要素オブジェクトモデル(COM) インターフェースのような用語は、プログラミングインターフェースの定義に包括される。

【0033】

当該プログラミングインターフェースの態様は、第1のコードセグメントが、情報(ただし、「情報」は最も広い意味で用いられ、データ、指令および要求等を含む)を第2のコードセグメントに伝送する方法と、第2のコードセグメントが情報を受け取る方法と、情報の構造、配列、シンタックス、構成、スキーム、タイミングおよび内容とを含むことができる。この点で、基礎を成す伝達媒体そのものは、情報がインターフェースによって定められるように伝達される限り、媒体が有線であっても、無線であっても、両者の組合せであっても、インターフェースの動作にとって重要ではないといえる。一定の状況において、情報伝達が他の機構を介するものであっても(例えば、コードセグメントの間の情報の流れから分離された緩衝域、ファイル等に搭載された情報)、1つのコードセグメントが第2のコードセグメントによって果たされる機能に単にアクセスするとき等は存在しなくてもよい。情報、従来の意味で一方向または双方向に送られないこともある。これらの態様のいずれかまたは全ては、所定の状況において、例えばコードセグメントが疎結合構成または緊密結合構成のシステムの一部であるかどうかに応じて重要でありうるため、このリストは例示的で、非制限的である。

【0034】

このプログラミングインターフェースの概念は、当業者に知られており、本発明の先述の詳細な説明から明らかである。しかし、プログラミングインターフェースを実装するための他の方法も存在し、明らかに排除されるものでなければ、これらも、本明細書の前に記載されている請求項に包括されるものとする。そのような他の方法は、図1Bおよび1Cの簡潔な図に比べて、より精巧または複雑に見えることもあるが、それでも同様の機能を果たして、同じ包括的結果を達成する。次に、プログラミングインターフェースのいくつかの例示的な代替的な実装形態を簡単に説明する。

【0035】

A. ファクタリング

通信を多数の個別的な通信に分割することによって、1つのコードセグメントから他のコードセグメントへの通信を間接的に達成できる。これは、図1Dおよび1Eに概略的に示されている。ここに示されるように、分割可能な機能の集合体という観点でいくつかのインターフェースを説明することが可能である。したがって、ちょうど数学的に 2×4 、または $2 \times 2 \times 3 \times 2$ を与えることができるように、図1Bおよび1Cのインターフェース機能をファクタリングして、同じ結果を達成することができる。よって、図1Dに示され

るように、インターフェース、すなわちインターフェース 1 によって提供される機能を細分割して、インターフェースの通信を多数のインターフェース、すなわちインターフェース 1 A、インターフェース 1 B、インターフェース 1 C 等に変換しながら、同じ結果を達成することができる。図 1 E に示されるように、インターフェース I 1 によって提供される機能を多数のインターフェース I 1 a、I 1 b、I 1 c 等に細分割しながら、同じ結果を達成することができる。同様に、第 1 のコードセグメントから情報を受け取る第 2 のコードセグメントのインターフェース I 2 を多数のインターフェース I 2 a、I 2 b、I 2 c 等にファクタリングすることができる。ファクタリングするときは、第 1 のコードセグメントに含まれるインターフェースの数が、第 2 のコードセグメントに含まれるインターフェースの数と一致する必要はない。図 1 D および 1 E の場合のいずれも、インターフェース、すなわちインターフェース 1 およびインターフェース I 1 の機能的意味は、それぞれ図 1 B および 1 C の場合と同じである。インターフェースのファクタリングは、ファクタリングの認識が困難になりうるように、結合特性、交換特性および他の数学的特性を追従することもある。例えば、動作の順序は重要でなく、その結果、インターフェースによって実施される昨日は、コードまたはインターフェースの他の部分によって、インターフェースに到達する前に十分に実施されるか、またはシステムの個別的な構成要素によって実施されうる。さらに、同じ結果を達成する異なる機能呼出の様々な方法が存在することを、プログラミング技術分野の当業者なら理解できる。

10

【0036】

B. 再定義

20

場合によっては、プログラミングインターフェースの一定の態様（例えばパラメータ）を無視、追加または再定義しながら、目的とする結果を遂行することが可能である。これは、図 1 F および 1 G に示されている。例えば、インターフェース、すなわち図 1 のインターフェース 1 は、機能呼出平方（入力、精度、出力）、すなわち入力、精度および出力の 3 つのパラメータを含み、第 1 のコードセグメントから第 2 のコードセグメントに発せられる呼出を含むものと仮定する。中央のパラメータである精度が、所定のシナリオに関係ない場合は、図 1 F に示されるように、無視するか、または（この状況において）無意味なパラメータと交換することも可能である。関係のないさらなるパラメータを追加することもできる。いずれの場合も、入力が第 2 のコードセグメントによって平方された後に出力が戻されるのであれば、平方の機能を達成することが可能である。精度は、コンピューティングシステムの下流または他の部分にとって十分に意味のあるパラメータでありうるが、精度が、平方を計算するという狭い目的に必要ないと認識されれば、交換または無視することができる。例えば、有効な精度値を通す代わりに、結果に悪影響を与えることなく誕生日のような無意味な値を通すことが可能である。同様に、図 1 G に示されるように、インターフェース I 1 を、パラメータを無視するか、またはインターフェースに追加するように再定義されたインターフェース I 1 ' と交換する。インターフェース I 2 は、同様に、不要なパラメータ、または他の場所で処理できるパラメータを無視するように再定義されたインターフェース I 2 ' として再定義されうる。ここでのポイントは、いくつかの場合において、プログラミングインターフェースは、ある目的に必要なとされないパラメータの如き態様を含むことができるため、無視または再定義されるか、あるいは他の目的のために他の場所で処理されうることである。

30

40

【0037】

C. インラインコード化

2 つの個別的なコードモジュールの機能の一部または全てを、それらの間の「インターフェース」が形を変えるように結合することも実現可能である。例えば、図 1 B および 1 C の機能をそれぞれ図 1 H および 1 I に変換できる。図 1 H において、図 1 B の先の第 1 および第 2 のコードセグメントを結合して、その両方を含むモジュールにする。この場合、コードセグメントは、まだ互いに通信しているが、単一モジュールにより好適な形に適合されうる。したがって、例えば、フォーマット呼出および復帰命令はもう必要ではないが、インターフェース、すなわちインターフェース 1 に準拠する処理または応答はまだ有

50

効でありうる。同様に、図 1 I に示されるように、図 1 C によるインターフェース I 2 の一部（または全て）をインターフェース I 1 にインラインで書き込んで、インターフェース I 1 ' ' を形成することができる。示されるように、インターフェース I 2 は、I 2 a と I 2 b に分割され、インターフェース I 2 a は、インターフェース I 1 によりインラインでコード化されて、インターフェース I 1 ' ' を形成する。具体的な例としては、第 2 のコードセグメントによって（入力の平方を計算するために）入力とともに通された値を処理した後で、平方結果を出力とともに戻すインターフェース I 2 によって受け取られる機能呼出平方（入力、出力）を実施する図 1 のインターフェイス I 1 を考える。そのような場合は、（入力を平方する）第 2 のコードセグメントによって実施された処理を、インターフェースに対する呼出を伴わずに第 1 のコードセグメントによって実施することができる。

10

【0038】

D. 分離

1 つのコードセグメントから他のコードセグメントへの通信は、通信を多数の個別的な通信に分割することによって間接的に達成されうる。これは、図 1 J および 1 K に概略的に示されている。図 1 J に示されるように、コードの 1 つまたは複数の部分（機能および・またはインターフェース機能を本来のインターフェースから分離するため分離インターフェースである）は、第 1 のインターフェース、すなわちインターフェース 1 上の通信を変換して、それらを異なるインターフェース、この場合はインターフェース、すなわちインターフェース 2 A、インターフェース 2 B およびインターフェース 2 C に適合させるように設けられる。これは、例えばインターフェース 1 プロトコルに従って所謂オペレーティングシステムと通信するように設計されたアプリケーションのベースがインストールされているが、その後オペレーティングシステムが、異なるインターフェース、この場合はインターフェース、すなわちインターフェース 2 A、インターフェース 2 B およびインターフェース 2 C を使用するように変化される場合に実施されうる。ポイントは、第 2 のコードセグメントによって使用される本来のインターフェースが、第 1 のコードセグメントによって使用されるインターフェースと適合できなくなるように変化されるため、仲介物を使用して、古いインターフェースと新しいインターフェースを適合可能にすることである。同様に、図 1 K に示されるように、第 3 のコードセグメントを分離インターフェース D I 1 とともに導入して、インターフェース I 1 からの通信を受け、分離インターフェース D I 2 とともに導入して、インターフェース機能を、例えば、D I 2 と協働するが、同じ機能結果を提供するように再設計されたインターフェース I 2 a および I 2 b に送信することが可能である。同様に、D I 1 および D I 2 は協働して、図 1 C のインターフェース I 1 および I 2 の機能を新たなオペレーティングシステムに合わせて翻訳しながら、同一または同様の機能的結果を提供することができる。

20

30

【0039】

E. 書換え

さらに他の可能な変形は、コードを動的に書き換えて、インターフェース機能を他の機能と交換するものであるが、それも同じ包括的結果を達成する。例えば、中間言語（例えばマイクロソフト IL、Java（登録商標）Byte Code 等）が、実行環境（.NET フレームワークによって提供される環境、Java（登録商標）ルーチン環境、他の同様のルーチン型環境等）におけるジャストインタイトム（JIT）コンパイラまたはインタプリタに提供される。第 1 のコードセグメントから第 2 のコードセグメントへの通信を動的に変換させ、すなわち第 2 のコードセグメント（本来の第 2 のコードセグメントまたは異なる第 2 のコードセグメント）が必要としうる異なるインターフェースにそれらを適合させるように、JIT コンパイラを書き込むことができる。これは、図 1 L および 1 M に示されている。図 1 L に見られるように、このアプローチは上記の分離シナリオに類似している。それは、例えば、インストールされたアプリケーションのベースが、インターフェース 1 のプロトコルに従ってオペレーティングシステムと通信するように設計されるが、後に異なるインターフェースを使用するようにオペレーティングシステムが変化され

40

50

る場合に実施されうる。JITコンパイラを使用して、インストールされたベースアプリケーションからの実行中の通信をオペレーティングシステムの新たなインターフェースに適合させることが可能である。図1Mに示されるように、動的にインターフェースを書き換えるこのアプローチを適用して、インターフェースを動的にファクタライズするか、あるいはインターフェースを変えることもできる。

【0040】

代替的な実施形態を介してインターフェースとして同一または同様の結果を達成するための上記シナリオを様々な様式で、縦列および／または並列に、または他の介在コードで組み合わせることができるという点にも注目される。したがって、上述の代替的な実施形態は、相互に排他的なものではなく、それらを混合し、整合し、組み合わせて、図1Bおよび1Cに示される一般的なシナリオと同一または同等のシナリオを生成することができる。たいていのプログラミング構造と同様に、ここに記載することができないが、それでも本発明の主旨および範囲によって表される同一または同様のインターフェースの機能を達成する他の同様の方法が存在するという点にも注目される。すなわち、それは、少なくとも部分的に、インターフェースの値の基礎を成すインターフェースによって表される機能、およびそれによって可能になる有利な結果であるという点に注目される。

【0041】

図2は、本発明の様々な態様に従って使用することが可能であるタブレットPC201を示す図である。図1のシステムにおける特徴、サブシステムおよび機能のいずれかまたは全てを図2のコンピュータに含めることができる。タブレットPC201は、複数のウィンドウ203が表示される大きな表示面202、例えばデジタル化フラットパネルディスプレイ、好ましくは液晶表示(LCD)画面を含む。ユーザは、スタイラス204を使用して、デジタル化表示面202上で選択、強調および／または書込みを行うことができる。好適なデジタル化表示面202の例としては、MutohまたはWacomペンデジタイザの如き磁気ペンデジタイザが挙げられる。他のタイプのペンデジタイザ、例えば光デジタイザを使用することもできる。タブレットPC201は、データを処理し、テキストを入力し、図を作成し、かつ／またはスプレッドシートおよびワードプロセッシングプログラム等の従来のコンピュータアプリケーション作業を実行するために、スタイラス204を使用してジェスチャを翻訳する。

【0042】

スタイラス204には、その選択機能を拡大するために、1つまたは複数のボタンまたは他の特徴を装備することができる。一実施形態において、一端が執筆部を構成し、他端が「消しゴム」端を構成し、ディスプレイを移動するときに、ディスプレイの部分が消去されることを示す「鉛筆」または「ペン」としてスタイラス204を実装することが可能である。マウスまたはトラックボール等の他のタイプの入力デバイスを使用することが可能である。また、ユーザ自身の指をスタイラス204とし、接触式または近接式ディスプレイ上の表示画像の部分を選択または指示するのに使用することも可能である。これにより、本明細書で用いられている「ユーザ入力デバイス」という用語は、広い定義を有することを意図しており、スタイラス204の如きよく知られている入力デバイスに対する多くの変形を包括する。領域205は、スタイラス204が表示面202のどこに接触したかをユーザが判断することを可能にするフィードバック領域または接触領域を示す。

【0043】

様々な実施形態において、そのシステムは、インクを取り込み、処理し、保管するのにアプリケーションが使用できるCOM(構成要素オブジェクトモデル)サービスの集合としてインクプラットフォームを提供する。1つのサービスは、アプリケーションが、開示されたインクの表現を用いてインクの読取りおよび書込みを行うことを可能にする。インクプラットフォームは、拡張可能マーク付け言語(XML)のような言語を含むマーク付け言語を含むこともできる。さらに、システムは、他の実装形態としてDCOMを使用できる。マイクロソフトコーポレーションによるWin32プログラミングモデルおよびNetプログラミングモデルを含むさらなる実装形態を使用することができる。

【0044】

リアルタイムインキングの概要

ユーザは、スタイラスまたはペンを使用して、電子インクを作成する。インクは、さらなるプロセスが完了するまで待つのではなく、ユーザがインクを作成する時刻により近い時刻にインクを表示することを可能にするシステムによって処理される。

【0045】

図3は、本発明の態様による電子インクを処理するための例示的なシステムを示す図である。入力マネージャは、入力マネージャ301においてインクを受け取る。ペンおよびデジタル化タブレット、およびWacomタブレット等を含む様々な入力マネージャが当該技術分野に存在する。インクデータの存在は、スタイラス入力イベント302と呼ばれる。次に、スタイラス入力イベント302は、インク収集器303によって処理される。インク集合オブジェクト303は、入力マネージャ130からの情報の最初の処理を実施する。次いで、システムは、ディスプレイ305上に出力用インク304を動的に描写する。他の構成要素は、インクのより複雑な処理を行うことができる。例えば、筆画を既存のインクオブジェクト306に加え（またはインク筆画を含む新たなインクオブジェクトを作製し）、受け取ったインク（データと称する）を1つまたは複数の特性と関連づけることができる。これは、インクオブジェクト特性を有する構成要素307によって示される。次いで、ディスプレイ305上の表示のために、インクオブジェクトの細描写（スムーシングが望まれる場合）308を行うことができる。

【0046】

図4は、図3のアプローチの代替的なアプローチを示す図である。図4は、入力内部スタイラス入力源401、入力マネージャ402（入力キューを有しても有さなくてもよい）、インク集合オブジェクト403、インクオブジェクト特性を有する要素405、リアルタイムインキングオブジェクト404およびディスプレイ406を含む。

【0047】

2つのタイプの入力の集合、すなわちスタイラスとデジタイザの間の接触に起因するデータと、デジタイザ上の移動に起因するデータとを処理することができる。デジタイザに接触しないデジタイザ上の移動は、「空間スタイラス」入力と呼ばれる。内部スタイラス入力源は、2つの入力の集合を分離し、それに応じてそれらを追跡する。図4に示される様々な活動を以下に列記する。

- A) 空中スタイラス入力イベントを入力マネージャ402の入力キューに追加する。
- B) 入力マネージャ402は、空中スタイラス入力をインク集合403に送る。
- C) 入力マネージャ402は、焦点変化が生じたかどうかを判断するための処理に向けて空中スタイラス入力を出力する。リアルタイムインキングオブジェクト404（リアルタイムスタイラスコントロールとも称する）にも通知される。リアルタイムインキングオブジェクト404は、何らかの必要なデータ（インク色等）を要求することができる。
- D) 空中スタイラスイベントは、通常の処理を続行し、インクオブジェクト特性を有する要素405に続く。
- E) スタイラス「ダウン」イベントを受け取って、リアルタイムインキングオブジェクト405に送る。
- F) リアルタイムインキングオブジェクト405は、点を受け取りながら描画する。これを動的描写と呼ぶことができる。
- G) リアルタイムインキングオブジェクト405は、描画したイベントを入力マネージャ402に送って、ユーザインターフェーススレッドが利用可能になる後の時刻に送らせる。
- H) 筆画を集め、次いでインクプロジェクトに追加する。
- I) 該要素は、リアルタイムインキングオブジェクト405に、動的に描画された筆画を除去するように通知し、次いで新たな筆画を再描画する。この動作は、筆画毎に行ってもよいし、いくつかのインク筆画に対して適用してもよい。
- J) 全てのインキングが完了した後にインクを描写する。

【0048】

図4に示される上記アプローチは、インクラグが「スタイラスダウン」イベントに対してのみ致命的であるという利点を提供する。インクラグは、スタイラスが活動的に描画している間のみ認知可能である。マイナスのユーザ影響を与えることなく、他の全てのスタイラスイベントを遅らせることが可能である。

【0049】

図4のアプローチは、マルチスレッド感知を入力マネージャキューおよびRTIオブジェクトにおいて局在化させる。また、焦点が確立されると、ラグまたは遅れが生じないことを保証する。

【0050】

オブジェクトモデル

図5は、様々な度合いで組み合わせることができる5つの部分を含む本発明の態様と併用することができる様々なオブジェクトを示す図である。以下の説明に用いられる様々な用語を以下に列記する。

【0051】

第1の部分は、電子インクの収集をサポートするペンサービスアプリケーション501である。例としては、マイクロソフトコーポレーションから提供され、Windows（登録商標）XPタブレットエディションで使用されるwisptis.exeがある。

【0052】

第2に、リアルタイムスタイラスサービス503（プロセス1 502と対応づけて示されている）は、スタイラスデータをペンサービス501から適切な収集用ウィンドウに送るアプリケーションである。リアルタイムスタイラスサービス503は、無数のオブジェクトを処理することができ、または過使用を最小限にするために制限されうる。例えば、制限される場合は、スレッド毎のオブジェクトの例示的な数を16、32、64および128等とすることができる。勿論、他の値を適用することもできる。

【0053】

第3に、リアルタイムスタイラス504および505をプロセス1に示す。リアルタイムスタイラス504および505は、スタイラスデータをリアルタイムスタイラスサービス503から受け取ることができる。各リアルタイムスタイラスオブジェクトは、ウィンドウの所定の部分または（対応付けられたウィンドウに基づく）領域またはそのリアルタイムスタイラスオブジェクトに対するスタイラスデータを受け取ることができる。

【0054】

多数のプロセスを以下にして同時に実施できるかを示すために、プロセス2 506も示す。リアルタイムスタイラスサービス507は、また、ペンサービス501からスタイラスデータを受け取ることができ、この情報をリアルタイムスタイラス508および509に送る。

【0055】

リアルタイムスタイラスオブジェクト504、505、508および509を、図6により詳細に示す。ペンサービス構成要素601は、データ流Aにおいて、データをリアルタイムスタイラスサービス602に送る。次に、リアルタイムスタイラスサービス602は、データ流Bにおいて、スタイラスデータを1つまたは複数のリアルタイムスタイラス構成要素603に送る。あるいは、点線で示されるリアルタイムスタイラスサービス602によって示されるように、このサービスを省略し、スタイラスデータをリアルタイムスタイラス構成要素603に直接流すことができる。

【0056】

リアルタイムスタイラス構成要素603で受け取られたスタイラスデータをプラグ可能構成要素606～608に直接供給することができる。あるいは、受け取ったスタイラスデータを、順序処理に向けてその入力キュー604に供給することもできる。次に、リアルタイムスタイラス構成要素603は、スタイラスデータを1つまたは複数のプラグ可能構成要素に送る。これらの構成要素606～608は、現在描写されているスタイラスデ

10

20

30

40

50

ータを記憶する視覚的記憶機構を有する動的描写体 608 を含むことができる。データ流は、データ流 C および D を含む。

【0057】

動的描写体 608 は、1 つまたは複数の筆画のパケットデータを蓄積し、視覚的記憶機構 609 に記憶することができる。これは、例えば、筆画がエッジの上に伸び、次いでスタイラスがまだダウンしている間にインク可能領域に再度入るときに有利である。ペンがダウンしている間に表示ウィンドウが再塗装要求を受け取ると、記憶機構 609 は、そのウィンドウを迅速に再塗装するためにその情報を提供する。動的描写体が記憶機構 609 にスタイラスデータを記憶しなかった場合は、システムは、データがそれを出力キューから出して、インク集合オブジェクト 611 に入れるまで、ウィンドウを再塗装し、最も新しい 1 つまたは複数の筆画を描写するのを待たなければならないことになる。

10

【0058】

プラグ可能構成要素 606 ~ 608 は、インターフェースを通じて一定の呼出に応答して、スタイラスデータを処理できるクラスオブジェクトである。使用できるインターフェースの一例としては、I S t y l u s S y n c p l u g i n が挙げられる。プラグ可能構成要素 606 ~ 608 の組合せは、デベロッパが、プラグイン内のパケット流からのデータを完全に制御し、修正し、そしてさらに消去することを可能にする。次いで、データ流 B からの修正されたパケット流を出力キュー 605 に記憶する。

【0059】

出力キュー 605 の出力は、データ流 F および G を通じて、プラグ可能構成要素 610 および 611 の他の集合に流れる。これらの構成要素は、ジェスチャレコグナイザ 610 と、インク収蔵庫 612 を有するインク集合オブジェクト 611 とを含むことができる。様々な追加的プラグインオブジェクトをデータ流 F および G によって引き出すことができる。

20

【0060】

プラグ可能構成要素 606 ~ 608 をプラグ可能構成要素 610 および 611 から分離するための決定を様々な基準に基づいて行うことができる。例えば、構成要素 606 ~ 608 を同期プラグインとし、構成要素 610 および 611 を非同期プラグインとすることができる。あるいは、スタイラスデータ処理時間がより短いプラグインをデータ流 C / D で処理することができ、より長いデータ処理時間はデータ流 F / G によって対処することができる。1 つのスレッド C / D からの構成要素をスレッド F / G の構成要素と交換することができる。

30

【0061】

プラグ可能構成要素 606 ~ 608 と 610 ~ 611 の 2 つの集合を分離する利点は、プラグ可能構成要素が異なるスレッドによって処理されることである。ここで、同期プラグインと非同期プラグインの違いは、それらが実行されるスレッド、および呼出順序にある（リアルタイムスタイラス 603 が実行されるスレッドによって同期プラグインを呼び出すことができ、非同期プラグインは、パケット流が同期プラグインによって処理され、出力キュー 605 に記憶された後で、通常はユーザインターフェースアプリケーションスレッドによって呼び出される）。

40

【0062】

ある場合は、リアルタイムスタイラス構成要素 603 からペンサービス 601 またはリアルタイムスタイラスサービス 602 に戻る公共通信が存在しうる。他の場合は、リアルタイムスタイラス構成要素 603 からペンサービス 601 またはリアルタイムスタイラスサービス 602 に戻る公共通信が存在しない。通信を防止することで、これらの構成要素からのデータの流れを容易にすることができる。

【0063】

ある状況において、リアルタイムスタイラス構成要素 603 は、所定の順序で呼出をプラグインに送ることによってデータを検索するときをプラグインに通知することができる。これらの通知を検索するプラグインの順序およびタイプは、デベロッパによって制御さ

50

れうる。イベントにおけるパケットデータをこれらのプラグイン606～608ならびに610～611によって監視することができる。

【0064】

本発明の態様は、ペンサービスからのパケットデータ、ディスプレイ、タブレットおよびペン等に関する変化の通知、ならびにリアルタイムスタイラスによって処理できる他のデータ集合を含む様々なデータタイプを扱う。以下の説明は、ペンサービスからのパケットデータの使用について記載しているが、これは、リアルタイムスタイラス603に使用できる多くのデータのタイプの1つにすぎない。以下の説明では、パケットデータは、RTSによって処理されるデータタイプについての説明例として用いられているが、RTSによって処理できるより一般的なデータを示すものとして理解されるべきである。

10

【0065】

リアルタイムスタイラス構成要素603は、キュー604および605を含むこともできる。出力キュー605は、時間スタイラス構成要素603が処理するパケットデータの全てを維持する。パケットデータがプラグインから戻されると、パケットデータはデータ流Eからの出力キュー605に追加される。次いで、出力キュー605は、プラグイン（例えば、非同期で、一般にインク集合オブジェクト611を含む）によって使用されうる。これは、データを（データ流Fから）抽出し、データ流Gにおける出力キュー605内に保持されたデータからインクオブジェクトを構築することによって行うことができる。

【0066】

出力キュー605のサイズは一定であってもなくてもよい。一定であれば、キュー605が満たされた後に、続いて受け取られるデータパケットが失われることがある。一定でなければ、キュー605のサイズが拡大して、追加的なデータパケットを受け取ることができる。一定サイズのキューを維持する利点は、データのバックログを合理的な時間で処理できるバックログに限定することである。例えば、エンドユーザがシステムと相互作用し、システムが応答しなくなった場合は、エンドユーザは、システムが再び応答するまで休止することによって、キューがデータを失うことなくバックログを処理することを可能にするのが一般的である。また、何らかの理由で、大容量のスタイラスデータが作成された場合は、キュー605は、一定サイズを有することによってデータの一部を除去するのに役立つことができる。

20

【0067】

データを順次受け取る際に出力キュー605を固定することができる。あるいは、以下に記載するように、データを順不同で出力キュー605に配置して、リアルタイム描写との同期を維持することができる。

30

【0068】

入力キュー604は、データ流B（または、リアルタイムスタイラスサービス602が存在しない場合は、データ流A）における情報を受け取る。入力キューは、データをプラグイン606～608に入力するプロセスを提供する。あるいは、スタイラスデータは、プラグイン606～608に直接流れることができる。データ流BとCの間の仲介物（ここではデータ流Z）として入力キュー604を有することに対する利点は、作成されたスタイラスデータを何も存在しないところに挿入できることである。

40

【0069】

高レベル制御の流れを以下に記載する。

a. リアルタイムスタイラス603は、プラグイン606～608に通されたパケットデータの処理を終了する。

b. リアルタイムスタイラス603は、処理したデータを出力キュー605に記憶する。

c. リアルタイムスタイラス603は、入力キュー604に保留中のパケットデータが存在するかどうかを調べる。データが存在すれば、入力キューからのそのパケットを取り出し、上記の工程aで処理する。

d. リアルタイムスタイラス603は、ペンサービス601に保留中のパケットデータ

50

が存在するかどうかを調べる。存在すれば、ペンサービス 601 からのそのパケットデータを取り出し、上記の工程 a で処理する。

e. このプロセスは、工程 c で繰り返される。

【0070】

リアルタイムスタイラスオブジェクトが以下にしてタブレットペンデータを処理できるかに対する代替的なアプローチを以下に記載する。図 8、11 および 13 に対する小さな改造によって以下のことを適用することができ、その改造は、同期スレッド上のプラグインの配列（デジチェーンされた各プラグインからリアルタイムスタイラスオブジェクトによって個別に呼び出された各プラグインまで）であることが理解される。

a. リアルタイムスタイラスオブジェクト 603 は、まずその入力キュー 604 のプラグインデータオブジェクトを、次いでペンデータ流 B からのプラグインデータオブジェクトを調べる。

b. リアルタイムスタイラスオブジェクト 603 は、C の 1 つのプラグインデータオブジェクトをその同期プラグイン集合におけるオブジェクト 606 ~ 608 に送る。各同期プラグイン 606 ~ 608 は、入力キュー 604 または出力キュー 605 にデータを追加することが可能である。

c. プラグインデータオブジェクトが同期プラグイン集合の全てのメンバーに送られると、プラグインデータオブジェクトをリアルタイムスタイラスオブジェクトの出力キュー 605 に配置する。

d. 次いで、リアルタイムスタイラスオブジェクト 603 は、（入力キュー 604 またはデータ流 B からの）処理する次のプラグインデータオブジェクトを調べる。

e. リアルタイムスタイラスオブジェクトの出力キュー 605 がデータを含んでいる間に、リアルタイムスタイラスオブジェクト 603 は、1 つのプラグインデータオブジェクトを出力キュー 605 からその非同期プラグイン集合 610 ~ 611 におけるオブジェクトに送る。各非同期プラグイン 610 ~ 611 は、入力キュー 604 または出力キュー 605 にデータを追加することができる。しかし、非同期プラグインは UI スレッド上で実行できるため、キュー 604 / 605 に追加されたデータは、タブレットペンデータ流 B におけるデータの残り、またはリアルタイムスタイラスオブジェクトの入力 604 および出力キュー 605 に対して、設定された関係を有さない。

【0071】

第 4 に、システムは、非同期プラグイン 611（ここではインク集合オブジェクトとして示される）を含む。ここで、インク集合オブジェクトは、1 つまたは複数のプラグインオブジェクトを表すことができる。インク収集および保存は、UI または非同期スレッドで行われる様々な活動の 1 つでありうる。パケットデータ（または修正パケットデータ）は、同期プラグイン 606 ~ 608 から戻ると、リアルタイムスタイラス 603 の出力キュー 605 に配置される。次いで、リアルタイムスタイラス 603 は、データを出力キュー 605 に配置する。次いで、データは、次のプラグイン 610 ~ 611 の集合（集合またはチェーン）に送られる。これは、非同期スレッド上の様々なプラグインによって指定されるようにデータを破壊／削除／再循環／解放できるインク収集オブジェクト 611 にデータを送ることを含むことができる。

【0072】

第 5 に、システムは、基本的な機能をシステムに提供する（同期または非同期スレッドのいずれかまたは両方のスレッド上で動作するように設計されうる）標準的なプラグインを含むことができる。標準的なプラグインは、動的描写体 608 と、ジェスチャレコグナイザ 610 とを含むことができる。勿論、要望に応じて、標準的なプラグインを他のプラグインと交換することができる。さらに、本発明の態様のいくつかの実装形態において、リアルタイムスタイラスには、最初から標準的なプラグインを含めなくてもよい。むしろ、デベロッパは、第 3 の供給者から選定し、かつ／または必要に応じて独自のプラグインを作成するのがよい。

【0073】

図7および8は、図6の構造の代替的なバージョンを提示している。図7は、スタイラスデータをリアルタイムスタイラス702に提供するペンサービス701を示す図である。入力キュー703（使用してもしなくてもよい）は、スタイラスデータを同期プラグイン801～805に提供する。ここで、各プラグインは、プラグイン集合の一部である。あるいは、プラグインは、プラグインのチェーンの一部であってもよい。最後の同期プラグイン（リアルタイムスタイラス同期プラグインN805）の後に、スタイラスデータ（または修正スタイラスデータ）は、出力キュー709に配置され、次いで後続のリアルタイムスタイラスイベント非同期プラグイン710および808によって処理される。

【0074】

図6および7のデータに関しては、RTSを通じて送信されるデータ集合は、単一データ集合であってもよいし、デジタイザは非常に高速でサンプリングするために効率性を目的として束ねられたデータ集合の束であってもよい。これらのデータ集合は、様々なイベントの通知を提供し、RTSを通じて新たな情報を送る。ある状況では、データ集合を削除することができる。データ集合がデータ集合の束である他の状況では、その束における単一のデータ集合を削除しながら、他のデータ集合を保持することができる。実際には、データ集合の束が修正される。次いで、リアルタイムスタイラス702は、私的なメッセージを、それが添付されるウィンドウに通知し、入力キューにおける次のデータ集合に進むことができる（存在しない場合は、ペンクライアントインターフェース上のペンサービス701によって呼び出される機能から戻る）。

【0075】

図8Aおよび8Bは、様々な動作のデータ流を示す図である。図8Aは、ペンサービス701で受け取られている色変化を示す図である。リアルタイムスタイラス702は、同期プラグイン集合809および非同期プラグイン集合810を含む。同期プラグイン集合は、同期プラグイン1801、動的描写体804および同期プラグイン3805を含む。同期プラグイン集合809からの出力は、出力キュー709に送られる。非同期プラグイン集合810は、出力キュー709からデータ集合を受け取り、非同期プラグイン1～3811、812および813で処理する。

【0076】

図8Aおよび8Bについての以下の説明では、データパケットが使用される。しかし、情報を伝達するのに他のデータ集合も使用できることが理解される。データパケットは、使用できるデータ集合のタイプの一例にすぎない。ここで、リアルタイムスタイラス702は、出力キュー709に2つのデータパケットAおよびBを有する。データパケットCは、色変化CCがペンサービス701によって受け取られるときに、動的描写体804によって処理されている。色変化CCをすぐに非同期プラグインの1つに送り、それによって処理することができる。しかし、そのようにすると、データパケットA、BおよびCが色変化の前に作成される当惑的状况が発生しうる。よって、非同期プラグイン集合810によるデータパケットA、BおよびCの最終的な処理および最終的な描写の後でのみ色変化を処理することが望まれることもある。

【0077】

色変化CCの処理を遅らせるために、プラグイン集合809における同期プラグインの1つは、データオブジェクトCC₁を作成し、入力キュー703に押し込むことができる。次に、そのデータオブジェクトを入力キューによってオンラインで処理することができ、その結果は、色変化が出力キュー709においてCC₂で表される。このアプローチを用いて、筆画の色を変える命令を既に受け取られたデータパケットで適性に指示することができる。

【0078】

図8Aは、リアルタイムスタイラス構成要素702によって処理されている色変化の例を示す図である。しかし、入力キューまたは出力キューによる処理のためのオブジェクトを作成するこのアプローチは、他のイベントを処理するのに役立つ。

【0079】

10

20

30

40

50

いくつかのプラグイン（動的描写体を含むが、それに限定されない）は、2つの範疇の特性、すなわち、変化が即時効果を示す特性と、次のスタイラスダウン（筆画開始）から効果を示す特性とを有することができる。それら2値の範疇の間の区別は、技術的要素、およびユーザの経験およびユーザの予想についての問題等を含む様々な理由によって発生しうる。例えば、筆画およびイベントがユーザの観点から発生した指示を保持するように、筆画の色変化を遅らせることができる。

【0080】

図8Bは、データパケットを処理し、同時にジェスチャ認識オブジェクトを処理するリアルタイムスタイラス構成要素702を示す図である。出力キュー709は、スタイラスダウンイベント（SD）、データパケット（P）およびスタイラスアップイベント（SU）を含むいくつかのデータオブジェクトを含む。スタイラスアップイベントSUが同期プラグイン集合809でペンサービス701から受け取られると、ジェスチャレコグナイザは、1つまたは複数の前のデータパケットPからジェスチャを認識しようと試みる。データパケットが、ジェスチャとして認識された場合は、ジェスチャレコグナイザ814は、ジェスチャ認識オブジェクトGRを生成し、入力キュー703に配置する。次いで、オブジェクトのジェスチャ認識GRは、同期プラグインを通して、出力キュー709に送られる。出力キュー709から、ジェスチャ認識オブジェクトGRは、同期プラグインを通して、次いで破壊／消去／再循環／解放される。ジェスチャ認識オブジェクトを作成し、入力プラグイン集合809を通じて戻す利点は、その処理により、受け取られたジェスチャ認識オブジェクト出力キュー709に送る前に、修正し、かつ／または消去することが可能になることである。さらに、ジェスチャ認識オブジェクトGRを同期プラグイン集合809および非同期プラグイン集合810によって処理させることによって、スタイラスダウンイベントとスタイラスアップイベントの間のデータパケットは、ジェスチャオブジェクトGRの存在に取って代わられるため、それらを除去することができる。あるいは、ジェスチャが対応するデータのイベントを無視してよいが、それは下流の他のプラグインに関係することもある。概して、システムは、挙動がデベロッパの所望の結果と符合するように、すなわち、無視されているデータが、データ構造に配置された後にそこから実際に除去されるように、ジェスチャレコグナイザが存在するときどのプラグインが存在するかを管理する。

【0081】

例えば、（パケットのテーブルを構築することができ、次いでSUにインク筆画を作成する）インク収集オブジェクトプラグインは、GRがそこに達する時まで既に筆画を作成することができる。そのため、GRを受け取ると、インク収集オブジェクトは、前に作成された筆画を単に消去することができる。次いで、他のプラグインは、GR上の適切な活動をトリガする役割を担うことができる。

【0082】

同期および非同期プラグイン集合内のデータの処理は、データに対する様々な処理技術に対応する。あるデータがいくつかのプラグインにのみ関係していた場合は、他のプラグインはそのデータを必要に応じて、無視するか、または送ることができる。例えば、第1のプラグインは、データパケットがディスプレイのインク不可能領域に関連していることを判断することができる。第2のプラグインは、以下の方法を含む様々な方法でデータパケットを処理することができる。

a. 第1のパケットからの通知を無視する。

b. データパケットが後続のプラグインに送られないように通知を吸収し、破棄する。
または、

c. （例えば、第2のプラグインがこれらのタイプのパケットに持つ関心の程度を示す特性を設定することによって）それが通知に関心がないことを明記する。ここで、それが、データパケットに関心がないことを明示すると、リアルタイムスタイラス構成要素は、第2のプラグインを省く。

【0083】

この第3のアプローチCは、機能呼出のオーバーヘッドを、必要とされないときに除去することによって、性能便益を提供する。

【0084】

「カスタムスタイラスデータをキューへ追加」法を呼び出すことによってカスタムスタイラスデータをリアルタイムスタイラスオブジェクトに追加することが可能である。「カスタムスタイラスデータをキューへ追加」法の呼出が、その「I スタイラス同期プラグイン」法の1つの呼出に応答して同期プラグインからなされる場合は、カスタムスタイラスデータを確定的にタブレットペンデータ流に追加する。さもなければ、非確定的に追加する。「カスタムスタイラスデータをキューへ追加」法は、「リアルタイムスタイラス」オブジェクトが使用不可の場合は、例外をもたらす。

10

【0085】

3つの場合のいずれかにおいて、カスタムスタイラスデータをリアルタイムスタイラスオブジェクトのキューに追加することができる。

a. キューパラメータが「入力」に設定されるときは、タブレットペンデータ流からの新たなデータの前に、カスタムデータをリアルタイムスタイラスオブジェクトの入力キューに追加し、同期プラグイン集合に送る。

b. キューパラメータが「出力」に設定されるときは、同期プラグイン集合によって現在処理されているデータの後に、カスタムデータをリアルタイムスタイラスオブジェクトの出力キューに追加する。

c. キューパラメータが「即時出力」に設定されるときは、同期プラグイン集合によって現在処理されているデータの前に、カスタムデータをリアルタイムスタイラスオブジェクトの出力キューに追加する。

20

【0086】

上記の各々の場合において、同期プラグイン集合における後続のプラグインによって追加されたデータは、直前のプラグインによって追加されたデータの後に追加される。

【0087】

カスタムスタイラスデータは、カスタムスタイラスデータオブジェクトとしてキューに追加され、プラグインは、「I スタイラス同期プラグイン、カスタムスタイラスデータ追加」または「I スタイラス非同期プラグイン、カスタムスタイラスデータ追加」法を通じてこのデータを受け取る。

30

【0088】

動的描写体およびジェスチャレコグナイザオブジェクトは、カスタムスタイラスデータをキューに追加することができる。

【0089】

リアルタイムスタイラスオブジェクトは、それが、その「カスタムスタイラスデータをキューへ追加」法の呼出を受ける元となるスレッド上で、「I スタイラス同期プラグイン、カスタムスタイラスデータ追加」法を呼び出す。

【0090】

リアルタイムスタイラスオブジェクトは、インクを収集するように構成されていても、いなくてもよい。それがインクを収集する場合は、リアルタイムスタイラスを使用して、インクデータをインク集合オブジェクトに送ることが可能である。インク集合オブジェクトは、リアルタイムスタイラスの出力に差し込むインク集合プラグインの形をとることができる。

40

【0091】

インク分析または手書き認識は、リアルタイムスタイラスオブジェクトの機能ではない。インク集合プラグインがインクを収集・作成すると、またはインクを認識したいときは、インクを「レコグナイザコンテキスト」または「ディバイダ」オブジェクトにコピーすることが可能である。

【0092】

上述したように、インクを複数回表示することができる。最初は、インクを動的描写体

50

で表示することができる。二度目は、インクがインク集合オブジェクトに収集された後で、インクを静的描写体で表示することができる。これにより、他のタイプの描写体を使用して、インクがインク集合オブジェクトから受け取られたときにインクを描写することが可能になる。例えば、多色インクを静的描写体の一部として有することができる。この効果の集合を動的描写体に押しつけることは、同期スレッドには負担が大きすぎることもある。そこで、描写体を作成し、それを非同期スレッドに接続することができる。あるいは、デフォルトの動的描写体を無視し、独自の動的描写体を作成し、それを同期スレッドに差し込むこともできる。オブジェクト指向のアプローチで用いられるように、多型、凝集または封込めを通じて標準的な動的描写体を内部的に再使用する動的描写体を作成することもできる。

10

【0093】

非同期スレッド上のこの新しい描写体を静的描写体として再使用してもよいし、この目的に合わせて新しい描写体を作成してもよい。例えば、インクを、木炭または他の先端で描画されているかのように描写したいことがある。さらに、物理的インクが乾燥する様子を表現するために、インクが（急速または徐々に）経時的に絶えず変化しているかのようにインクを表示する描写体を作成することができる。さらに、インクを強調表示するために、（虹のような）循環色でインクを表示する描写体を作成することができる。具体的には、スタイルスダウン、パケットおよびスタイルスアップ通知に従う同期プラグインを作成することによって、動的描写体プラグインを作成することができる。次いで、プラグインは、筆画を描かれているように描写することができる。新しい描写体は、インキングならびに様々な選択機構を処理することができる。

20

【0094】

動的描写体プラグインは、タブレットペンデータがリアルタイムスタイルオブジェクトによって処理されているときに、それをリアルタイムで表示するオブジェクトである。後に、フォーム再生のようなイベントに対して、動的描写体プラグインまたはインク集合プラグインは、インクを再描画することができる。

【0095】

動的描写体オブジェクトは、「Iスタイル同期プラグイン」インターフェースを実装する。動的描写体は、また、非同期インターフェースを実装することもできる。動的描写体オブジェクトは、インクを描画されているようにリアルタイムで描写する。特に指定がなければ、再生法が呼び出されると、動的描写体オブジェクトは、（「動的描写体のキャッシュドデータ」に対応する）既に収集された任意の他の筆画とともに、現在収集されている筆画を再描画する。例えば、インクが所定の形状の外に描写されないように、クリッピング領域によって描画の挙動を拘束するためのパラメータを含む再生挙動の他のオーバーロードが可能である。

30

【0096】

これは、（動的描写体が一時的にインクデータを保存することができる）様々な図に示されている。動的描写体オブジェクトは、その「Iスタイル同期プラグイン、スタイルスアップ」法の呼出を受け取ると、筆画データを保存し、その筆画に対する「スタイルスアップデータ」オブジェクトに応答してカスタムスタイルデータを入力キューに追加する。「カスタムスタイルデータ」オブジェクトの「カスタムデータId」特性は、「ダイナミック描写体キャッシュドデータGuid」値に設定され、「カスタムスタイルデータ」オブジェクトのデータ特性は、「ダイナミック描写体キャッシュドデータ」オブジェクトを含む。次いで、インクが下流で描写されると、対応づけられたデータの動的描写体のキャッシュをクリアすることができる。あるいは、インク筆画は、まだインク収集オブジェクトを通して、それによって記憶されていないため、インクを再生する（原稿の筆画および「キャッシュドデータ」に記憶されたデータを再描画する）と、必ずインク筆画の動的描写体のキャッシュをクリアされるわけではない。

40

【0097】

図9および10は、本システムの態様を確立するためのプロセスを示す図である。図9

50

では、工程 901 において、リアルタイムスタイラス構成要素をインスタンス化する。工程 902 において、同期および非同期プラグインをインスタンス化する。これは、動的描写体および／またはジェスチャレコグナイザ 903 を含んでいてもいなくてもよい。工程 904 において、プラグインをプラグイン集合に追加する。工程 905 において、リアルタイムスタイラスを使用可能にする。工程 906 において、リアルタイムスタイラスを通じてペンデータを受け取る。

【0098】

図 10 は、リアルタイムスタイラスを使用するためのプロセスを示す図である。工程 1001 において、RTS によってパケットを受け取る。これは、入力キューの動作を含んでいてもいなくてもよい。入力キューは、任意のデータの挿入を、それが同期集合を通るように何時でもサポートする。本例では、RTS のペンクライアントインターフェースの実装に対する機能呼び出す結果として、データを入力キューに挿入する唯一の構成要素は、ペンサービスである。

【0099】

工程 1002 において、パケットは、同期プラグイン集合に入る。図 8 のシステムのようなシステムに対しては、この工程は、パケットをプラグインの第 1 のプラグインに送ることを意味することになる。リアルタイムスタイラスは、集合（またはチェーンが存在する場合はチェーン）における第 1 のプラグインにおける特定のパケットデータに適した機能呼び出す。多くの場合、この構成要素は、データを修正するか、またはそのままにしておくかの選択肢を有する。構成要素は、その動作を終了すると、次のプラグインオブジェクトについて同じ機能呼び出す等である。

【0100】

工程 1003 において、パケットは、同期プラグイン集合（またはチェーン）を進む。これは、工程 1004 に示されるように、パケットをジェスチャレコグナイザまたは動的描写体に送ることを含んでいてもいなくてもよい。動的描写体を使用する場合は、カーソルダウンイベント（ペンダウンイベントとも称する）のデータを記憶機構に蓄積し、それらを画面に描写すること（動的描写体の動作）を開始することができる。カーソルアップイベント（またはペンアップイベント）において、動的描写体は、次のカーソルダウンイベントのためにその記憶機構をクリアすることができる。

【0101】

ジェスチャレコグナイザを使用する場合は、ジェスチャレコグナイザは、カーソルダウンイベントを検出し、それ自体の記憶機構（例えば記憶機構 807）へのデータの蓄積を開始することができる。筆画が蓄積するにつれて、ジェスチャレコグナイザは、筆画をレコグナイザに導き、筆画が所望のパターン（またはジェスチャ）に対応する場合は、a) ジェスチャデータを出力キューに追加し、b) 記憶機構を空にし、次のカーソルダウンイベントをキューに通して蓄積を再び開始する。ここで、ジェスチャは、1 つまたは複数の筆画を含むことができることを注記しておく。以下により詳細に説明する図 14 は、図 10 のプロセスに関して、多数の筆画ジェスチャをどのように処理できるかを示す図である。

【0102】

プラグインがどの程度前の筆画を把握すべきか、または把握できるかは、プラグインの目的に左右される。

a. 1 つの例示的なシナリオにおいて、ジェスチャを検出しているプラグインは、ジェスチャに応答して活動を行う役割を担うことができる。それは、筆画に関して何かを把握する必要はない。例えば、ウィンドウ内の「サークル」ジェスチャが、そのアプリケーションに対する「削除」を意味する場合は、ジェスチャを受け取るプラグインは、単にそのウィンドウに「削除」キー入力を送ることになる。この例示的なシナリオでは、インクがウィンドウに収集されないことがある。それは、テキストを含むウィンドウであってもよく、この場合、筆画データが単に通過して回収されないように、RTS 枠組を構成することができる。したがって、プラグインは、筆画を保存する必要はない。

i. 代替的な場合において、上記の「インタープレタ」プラグインは、ジェスチャが翻訳されたため、その筆画を持続しないことを下流のプラグインが把握するように、独自の通知を生成することができる。

b. 他の例示的なシナリオにおいて、プラグインは、インク収集を行い、動作を編集することができる。この規範的な例は、全ての筆画が潜在的にジェスチャであり、それらがジェスチャとして扱われる場合は、筆画が実質的に削除されるインクコレクタオブジェクトのモード（インクおよびジェスチャモード）である。したがって、プラグインは、筆画がジェスチャとして翻訳されてしまった場合に、筆画を見つけて削除できるようにするために、筆画を保存する。

【0103】

これらの例示的なシナリオのいずれにおいても、実際には、単一の筆画ジェスチャと多数の筆画ジェスチャとの違いは問題ではない。アプリケーションシナリオは、多数の筆画ジェスチャ（例えば矢印）を必要とすることがあり、その場合は、上記の例示的なプラグインのいずれもが不要な多数の筆画ジェスチャを認識することになる。

【0104】

次に、工程1005において、パケットは、同期プラグインの集合によって処理された後に出力キューに入る。先述したように、プラグインの集合は、そこに送られたオブジェクトの各々を処理するのに際して、データパケットを修正、削除、かつ／または交換することができる。同期プラグインのいずれもがパケットを削除しなかったとすると、工程1005を実行する。

【0105】

工程1006において、パケットは、非同期プラグイン集合を進む。

【0106】

動的描写および湿式インク

システムは、スタイラスデータを迅速かつ効率的に処理する能力を提供する。電子インクを真のインクのように挙動させるために、電子インクは、スタイラスの先端から、真のインクと同じくらい容易に流れるように見えることが必要である。インクの流れにおける小さな破損および遅れは、電子インク体験にとって有害である。電子インクが、スタイラスからスムーズに流れるように見えるようにするための様々なアプローチを以下に記載する。「湿式」インクは、動的描写体が、インク収集後に、静的描写をどのくらい長く保持するかを制御する機能と見なされる。インクの寿命の管理を扱いやすくするために、通知（ここでは動的描写体キャッシュドデータと称する）を利用して、インク収集プラグインが、キャッシュド（または湿式）インクを動的描写体から解放することを可能にする。

【0107】

様々な異なるタイプの動的描写体を作成することができる。例えば、動的描写は、微尖端またはチゼル尖端ペンによって生成されたようなインクを描写する代わりに、半透明水性塗料（「水性色」とも称する）によって生成されたようなインクを描写することができる。また、木炭スタイラスによって生成されたようなインクを描写する動的描写体を作成することができる。さらに、動的描写体は、インクが固定されずに、定期的に色を変化させ、かつ／または移動するようにインクを描写することができる。例えば、合体して所望のインク形態になるインクまたはダストの小滴、究極的に移動して所望のインクを形成する小さなウォーム、または色および形状を変化させるインクのように見えるインクを描写することができる。

【0108】

動的描写体は、RTSの非同期スレッドに対応づけられたオブジェクトによって後に描写されるのと同じ形態でインクを表示することができる。あるいは、動的描写体によって描写されたインクと、RTSの非同期スレッドに対応づけられたインクの最終的な表示との間に関係が存在していてもよい。例えば、動的描写体は、非同期スレッド上のプラグインによって後に第2の色調または形状に修正されることになる第1の色調または形状でインクを示すことができる。これは、ユーザにとっては、スタイラスから流れているインク

10

20

30

40

50

または塗料が乾燥するように見えることがある。

【0109】

動的描写および湿式インクについての以下の説明を図6および7に適用することができる。説明を目的として、図6を参照する。図7の構成も同様に適用可能で、以下の説明は、図7をも包括するのが望ましい。

【0110】

図6を参照すると、デジタイザ入力は、ペンサービス601から進入し、リアルタイムスタイラス603に到達する。リアルタイムスタイラスは、デジタイザ入力を複数の同期プラグインオブジェクト606～608に通し、その結果を出力キュー605に記憶する。オブジェクト606～608は、オブジェクト606～608の間にリンクを描くインターフェース、例えば「I スタイラス同期プラグイン」を通じて接続されうる。次いで、オブジェクト610および611の非同期プラグイン集合は、出力キュー605から出力を受け取り、その処理を開始する。例えば、インク収集オブジェクト611は、出力キューからデジタイザ入力を抽出し、それをインクとして記憶することができる。デジタイザ入力の動的描写は、リアルタイムスタイラス構成要素603が、デジタイザ入力（修正または未修正）を動的描写体608に通すときに行われる。デジタイザ入力（修正または未修正）の静的描写は、デジタイザ入力（修正または未修正）がインク収集オブジェクト611の内側のインク記憶機構に記憶されているときに行われる。

【0111】

動的描写608の後で、デジタイザ入力（修正または未修正）がインク収集オブジェクト611の内側に記憶される前にインキング表面が無効になる場合に1つの問題が発生しうる。得られるユーザー体験は、デジタイザ入力（修正または未修正）がインク収集オブジェクト611に到達するまで表示インクが消失することになる。ユーザは、インクを再度作成する必要があるものと考え、そうすることに時間を費やすため、結果的には重複したインクが記憶される。この視覚的問題は、インクがすぐにユーザに示されないこと、または遅れの後に示されることに起因する。

【0112】

このインク消失シナリオを回避するために、動的描写体608とインク収集オブジェクト611の間に通信プロトコルを確立することができる。動的描写体608は、（例えば、デジタイザ入力（修正または未修正）がインク記憶機構612に記憶されたときに）インク収集オブジェクト611によりデジタイザ入力を解除できることを指示されるまで、デジタイザ入力を視覚的記憶機構609に継続して蓄積することができる。

【0113】

ここでの目的に応じて、動的描写体608におけるデジタイザ入力の一時的キャッシュ609を現行ではキャッシュデータと呼ぶ。湿式インクは、システムが、デジタイザ入力をインク集合オブジェクト611に記憶する前にインクを描写するための機能である。

【0114】

図11は、どのようにしてインク集合オブジェクトが、そのキャッシュデータを解放するように動的描写体に警告するかを示す図である。動的描写体1201は、視覚的記憶機構1202を含む。インク収集オブジェクトは、インク記憶機構1204を含む。動的描写体1201は、スタイラスデータを視覚的記憶機構1202に記憶すると、オブジェクトを入力キュー1205に出力する。リアルタイムスタイラス（例えば702）は、ペンサービス701を検索する前に処理する次のデータのために入力キューを探索する。ここでの目的に応じて、入力キューに挿入されたオブジェクトを「ダイナミック描写体キャッシュデータ」と呼ぶ。次いで、オブジェクトをリアルタイムスタイラスによって取り出し（追加的な処理を伴う、または伴わない）、それを出力キュー1206に出力する。その間、インク収集オブジェクト1203は、出力キュー1206に現れる順番に、受け取られたスタイラス入力パケットを処理する。インク収集オブジェクト1203は、動的描写体1201からのオブジェクト（ここでは「動的描写体キャッシュデータオブジェクト」）、以下のことを実施する。

- a. 動的描写体によって描写された筆画と一致する適正な相関筆画を作成する。

b. 動的描写体筆画上に存在する描画属性を、描写される比較に適用する。

c. 動的描写体 1 2 0 1 に解除通知を送る。この通知は、動的描写体に関する方法を実行し、パラメータを通すことを含むことができる。

【0 1 1 5】

上記の工程 B は、全ての筆画が出力キューから外に移動される前に、描画属性改良の同期がとられたか否かに適正に基づいて、実施してもしなくてもよい。図 8 A は、描画属性の変化を先に受け取られたデータパケットと同期させるためのプロセスに関する。

【0 1 1 6】

描画属性をデベロッパの要望に従ってシステムに設定してもよいし、他のアプローチでは、インク収集オブジェクト 1 2 0 3 に送られるオブジェクトに設けてもよい。解除通知は、動的描写体 1 2 0 1 によって提供されるインターフェースに対する呼出であってもよい。ここでは、通知は、リリースキャッシュドデータであってもよい。

【0 1 1 7】

図 1 2 は、湿式インクを処理する代替的なアプローチを示す図である。図 1 2 は、スタイラスデータをリアルタイムスタイラス 7 0 2 に送るペンサービス入力 7 0 1 を含む。入力キュー 7 0 3 は、リアルタイムスタイラスイベントシンクを出力キュー 7 0 9 に配置した結果を伴う入力を処理し、待ち行列に入れることができる。図 1 2 は、追加的なキュー 1 1 0 1 を含む。このキューは、他のリアルタイムスタイラスイベントシンク A 1 1 0 3 および B 1 1 0 4 から分離された動的描写体 1 1 0 2 からの結果を処理する。ここで、このアプローチは、リアルタイムスタイラスイベントシンク A 1 1 0 3 および／または B 1 1 0 4 がスタイラスから流れるインクの出現を遅らせるという、発生しうる問題に対処する。図 1 2 のシステムは、他のイベントシンク A 1 1 0 3 および B 1 1 0 4 を処理する前にスタイラスデータの動的描写を処理する。イベントシンク A 1 1 0 3 および B 1 1 0 4 が、スタイラスデータがインク集合オブジェクト 1 1 0 5 に送られる過程でその処理を遅らせたとしても、動的描写 1 1 0 2 イベントシンクは、スタイラスデータが受け取られているときにその処理を続けることができる。スタイラスデータが動的描写体 1 1 0 2 によって受け取られるや否や、出力が追加的なキュー 1 1 0 1 に送られ、そこでリアルタイムイベントシンク A 1 1 0 3 および B 1 1 0 4 によってアクセスされる。リアルタイムイベントシンク A 1 1 0 3 および B 1 1 0 4 がどのようにアクセスされるか（一方が他方を呼び出すか、または両方が追加的なキュー 1 1 0 1 によって呼び出される）についてのプロセスは、いずれかのアプローチを通じて実施されうること留意されたい。

【0 1 1 8】

図 1 3 は、湿式インクがスタイラスから流れることをどのようにして保証するかに関するさらに他の態様を示す図である。ここで、個別的なスレッドに二次的なリアルタイムスタイラスオブジェクトが存在する。ペンサービス 7 0 1 は、スタイラスデータをリアルタイムスタイラス構成要素 1 3 0 1 に出力する。入力キュー 1 3 0 2 は、スタイラスデータを動的描写体 1 3 0 3 に提供する。他の同期プラグインをリアルタイムスタイラス構成要素 1 3 0 1 に対応づけても、対応づけなくてもよい。オプションの同期プラグインは、ジェスチャレコグナイザ 1 3 0 4 および他の同期プラグイン 1 3 0 5 を含む。これらの同期プラグインからの出力は、出力キュー 1 3 0 6 に送られる。ここで、リアルタイムスタイラス構成要素 1 3 0 7 は、リアルタイムスタイラス構成要素 1 3 0 1 からの出力キュー 1 3 0 6 に接続される。リアルタイムスタイラス構成要素 1 3 0 7、ならびにそれに対応づけられた同期プラグイン 1 3 0 9 ～ 1 3 1 1 および非同期プラグイン 1 3 1 2 ～ 1 3 1 3 は、リアルタイムスタイラス構成要素 1 3 0 1 の観点から、非同期プラグインとして働く。データパケット（ただし、リアルタイムスタイラス構成要素 1 3 0 1 の同期プラグインによって修正されている）は、リアルタイムスタイラス構成要素 1 3 0 7 に対する同期プラグイン集合に送られる。例えば、リアルタイムスタイラス構成要素 1 3 0 7 に対する同期プラグイン集合は、同期プラグイン A 1 3 0 9、同期プラグイン構成要素 B、1 3 1 0、および（既に使用されていない場合は）ジェスチャレコグナイザ 1 3 1 3 を含む。ジェスチャレコグナイザ 1 3 1 3 は、任意のリアルタイムスタイラス構成要素に対する同期ス

10

20

30

40

50

レッドまたは非同期スレッドに対応づけることができるタイプのプラグインであることが理解される。ジェスチャレコグナイザ1313をリアルタイムスタイラス構成要素1307に対する同期プラグイン集合に対応づけることは、例示のみを目的としている。

【0119】

リアルタイムスタイラス構成要素1307の同期プラグイン集合の出力は、出力キュー1311に送られる。次いで、非同期プラグインオブジェクト1312および1313は、出力キュー1313におけるデータパケットを処理することができる。ここでも、インクは動的に描写され、アプリケーションが中断されてもペンからスムーズに流れる。

【0120】

参照のため、様々なプラグイン構成要素を、プラグイン集合において順次送るのではなく、互いに連鎖させることができることを注記しておく。ここで、同期プラグイン構成要素A1309をプラグイン構成要素B1310に連鎖した例を図13に示す。勿論、本発明の態様に従ってプラグイン構成要素を扱うために、プラグイン構成要素を互いに連鎖させ、またはそれらを集合に収集するアプローチを互いに代替的または追加的に用いることができる。

【0121】

ジェスチャ認識

ジェスチャ認識は、ジェスチャが作られ、それが適切に処理されているかどうかを判断することを目的とする。例えば、ジェスチャレコグナイザ610は、デジタイザ入力を検出し、ジェスチャ認識結果を動的描写体608と同様の入力キューに挿入する役割を担う。図8Bは、同期および非同期プラグイン集合でジェスチャ認識をどのように実施できるかを説明する図である。その説明は、ジェスチャ認識オブジェクトの動作に対応している。

【0122】

ジェスチャレコグナイザ610は、アルゴリズムを使用することができる。ここで、ジェスチャレコグナイザは、交換アルゴリズムを使用して、多筆画ジェスチャを認識することができる。例えば、最大数の筆画に関する特性が設定されると、ジェスチャレコグナイザは、多くの筆画を繰り返る。例えば、その特性が2に設定された場合は、ジェスチャレコグナイザは、最近の2つの筆画を繰り返って、それらをジェスチャとして認識しようとする。これは、多数の認識呼出をもたらすことがある。しかし、このアプローチは、ジェスチャレコグナイザが、最後の筆画の後で処理を開始するのを待つための一定の遅れを排除する。

【0123】

図14は、ジェスチャレコグナイザが、ジェスチャの部分を得るために、どのくらい後まで繰り返るかについての例を示す図である。結果を（ここでは、例えば「ジェスチャ認識データ」と名付けられる）オブジェクトに記憶することができる。その結果をリアルタイムスタイラス入力キューに挿入することができる。

【0124】

インク収集オブジェクト611は、ジェスチャオブジェクト（ここでは「ジェスチャ認識データ」）を受け取ると、インク記憶機構612からそのジェスチャに対する筆画を除去し、そのジェスチャに応じて適切な活動を実施する。

【0125】

上述したように、ジェスチャ認識を実施するために、システムは、タップジェスチャに対する「スタイラスアップデータ」オブジェクトの如き、ジェスチャを完成するデータに応じて「システムジェスチャデータ」オブジェクトを入力キューに追加することができる。

【0126】

ジェスチャレコグナイザを様々なインターフェースによるオブジェクトとして実装することができる。例えば、ジェスチャレコグナイザオブジェクトは、「Iスタイラス同期プラグイン」および「Iスタイラス非同期プラグイン」インターフェースを実装することが

できる。

【0127】

ジェスチャレコグナイザオブジェクトは、ジェスチャを認識すると、筆画に対する「スタイラスアップデータ」オブジェクトに応じて、カスタムスタイラスデータを入力キューに追加する。「カスタムスタイラスデータ」オブジェクトの「カスタムデータId」特性が「ジェスチャ認識データGuid」値に設定され、「カスタムスタイラスデータ」オブジェクトのデータ特性は、「ジェスチャ認識データ」オブジェクトを含む。

【0128】

特に指定がなければ、ジェスチャレコグナイザオブジェクトは、単一筆画ジェスチャを認識するにすぎない。ジェスチャレコグナイザオブジェクトは、多筆画ジェスチャを認識するように設定されうる（例えば図14を参照）。多筆画ジェスチャについては、「カスタムスタイラスデータ」オブジェクトが、ジェスチャの最終筆画に対する「スタイラスアップデータ」オブジェクトに応じて、入力キューに追加される。多筆画ジェスチャを認識すると、重複する筆画の集合に対する通知を受け取ることができる。例えば、第1および第2の筆画とともに1つのジェスチャとして認識することができ、第2の筆画そのものをジェスチャとして認識することができ、第2および第3の筆画とともに他のジェスチャとして認識することができる。

10

【0129】

多筆画ジェスチャ認識のためにジェスチャレコグナイザオブジェクトを使用している場合は、リアルタイムスレッドでの遅れを低減しながらも、ジェスチャ認識がユーザインターフェースでの遅れの影響を受けるのを防ぐために、縦続リアルタイムスタイラスモデルを使用し、ジェスチャレコグナイザオブジェクトを二次的なリアルタイムスタイラスオブジェクトに接続することができる。

20

【0130】

さらに、筆跡、ジェスチャまたは他のオブジェクトを以下の3つの方法で認識するカスタムジェスチャレコグナイザプラグインを作成することができる。

a. 筆画情報を既存のレコグナイザオブジェクトに送り、「カスタムスタイラスデータをキューへ追加」法を用いて、その結果をペンデータ流に追加する。

b. カスタムプラグイン内で認識を行い、「カスタムスタイラスデータをキューへ追加」法を用いて、その結果をペンデータ流に追加する。

30

c. 標準的なジェスチャレコグナイザプラグインを、ページチェーン様式で、その標準的なプラグインを呼び出すカスタムプラグインに重ねる。このように、デベロッパは、インエアパケットをパケットに、「カーソルインレンジ」を「スタイラスダウン」に、「カーソルアウトオブレンジ」を「スタイラスアップ」に翻訳することによって、インエアジェスチャ認識を実施することができる。

【0131】

リアルタイムスタイラスオブジェクトは、システムジェスチャがシステムによって認識されているときに、それらに関するデータを受け取る。以下のテーブルは、「スタイラスジェスチャデータ」オブジェクトがペンデータ流のどこで発生するかを、他のペンデータに関連して説明するものである。以下のリストは、例示的なものであって、網羅的なものではない。本発明の範囲を逸脱することなく、他のジェスチャを本発明の態様と併用することができる。

40

【0132】

【表 2】

システムジェスチャ タップ	摘要 スタイラスダウンデータオブジェクトの後で、スタイラスアップデータオブジェクトの前	
ダブルタップ	スタイラスダウンデータオブジェクト、タップシステムジェスチャに対するシステムジェスチャデータオブジェクト、およびスタイラスアップデータオブジェクトの後で、第2のスタイラスダウンデータオブジェクトの前	
ライトタップ	スタイラスダウンデータオブジェクト、およびホールドエンターシステムジェスチャに対するシステムジェスチャデータオブジェクトの後で、スタイラスアップデータオブジェクトの前	10
ドラッグ	スタイラスダウンデータオブジェクトの後で、スタイラスアップデータオブジェクトの前	
ライトドラッグ	スタイラスダウンデータオブジェクトの後で、スタイラスアップデータオブジェクトの前	
ホールドエンター	スタイラスダウンデータオブジェクトの後で、スタイラスアップデータオブジェクトの前。このシステムジェスチャは、ユーザがドラッグまたはライトドラッグシステムジェスチャを開始すると、認識されない。	20
ホールドリリース	オプション	
ホバーエンター	平均速度が小さいいくつかのインエアパケットデータオブジェクトの後。ホバーエンターシステムジェスチャを受け取る前に顕著な遅れが生じることがある。リアルタイムスタイラスオブジェクトが、システムジェスチャのときにペンの真下のウィンドウまたはコントロールに接続されると、リアルタイムスタイラスオブジェクトはこのデータを受け取る。	
ホバーリリース	ホバーエンターシステムジェスチャに対するシステムジェスチャデータオブジェクト、および平均速度が十分ないいくつかのインエアパケットデータオブジェクトの後。ホバーリリースシステムジェスチャを受け取る前に顕著な遅れが生じることがある。リアルタイムスタイラスオブジェクトが、システムジェスチャのときにペンの真下のウィンドウまたはコントロールに接続されると、リアルタイムスタイラスオブジェクトはこのデータを受け取る。	30

【0133】

同期および非同期プロセス

40

リアルタイムスタイラスインスタンスがインスタンス化されると、実行スレッドをインスタンス化することができる。実行スレッドを同期オブジェクトと非同期オブジェクトに分離することができる。同期オブジェクトは、一般には、ペンサービス構成要素から発するパケットと同期して動作する。非同期オブジェクトは、一般には、インク筆画、および／またはペンサービス構成要素から発する他のデータパケットと常に同期して実行されなくてもよいオブジェクトに分類される。これら2つのスレッドの分離は、動的描写体オブジェクトが、データパケットの受取りと同期してデータパケットを迅速に処理することを可能にし、まだ重要である間に、オリジナルデータパケットを受け取ってからわずかな遅れで正確に処理することができる他のプラグインオブジェクトを可能にする。

【0134】

50

同期プラグインの収集は、リアルタイムスタイラスインスタンスに対してインスタンス化されたこの新しいスレッド上で実行される。非同期プラグインの収集は、ユーザインターフェーススレッドで実行されうる。

【0135】

リアルタイムスタイラススレッドは、インキング（パケットキューイングおよび動的描写）が全体的に反動的でスムーズになるように、計算的に軽い活動を目的とするものであってもよい。したがって、一般には、同期プラグインに対するスレッドには、計算的に軽い処理のみを実施すべきである。

【0136】

（通常は出力キューの出力に対応づけられる）ユーザインターフェーススレッドは、通常は、計算的に重い活動を必要とする非同期プラグインによって使用される。そのために、リアルタイムスタイラススレッドとユーザインターフェーススレッドの間にキューアーキテクチャを設けて、ユーザインターフェース（UI）スレッドにおける一時的な妨害に対して強くする。UIスレッドは、パケット、およびリアルタイムスタイラススレッドに収集された他のデータの最終的な宛先であってもなくてもよい。代替的なアーキテクチャも計算的に重い活動をより良好に処理できることにより、ペンスレッドの妨害とユーザインターフェーススレッドの妨害との間でデベロッパに容易に妥協させないことを注記しておく。例えば、ここに記載されている縦続設計は、スレッドの分離によって異なる活動をより良好に処理することを可能にする。

【0137】

ジェスチャレコグナイザを同期スレッド上に実装してもよいし、ジェスチャレコグナイザが、見込まれる重いジェスチャ認識活動（例えば、筆画が比較されることになる多数の容認ジェスチャ）から弱まるものと判断または想定される場合は、非同期スレッド上に実装してもよい。ジェスチャレコグナイザオブジェクトは、同期プラグインインターフェースと非同期プラグインインターフェースの両方を有することで、いずれかまたは両方のスレッドからのオブジェクトを利用する柔軟性をデベロッパに与えることができる。この機能は、ジェスチャレコグナイザに限定されない。動的描写体または任意の他のプラグインも、場合によっては、いずれかまたは両方のキューにおける使用をサポートすることができる。デベロッパがプラグインの良好な位置をどこであると想定するかに応じて、プラグインがサポートする場合は、デベロッパがプラグインを同期集合または非同期集合またはその両方に配置することを可能にするために、全てのプラグインが、同期インターフェース、非同期インターフェース、またはその両方を有していてもいなくてもよい。

【0138】

リアルタイムスタイラスオブジェクトは、タブレットペンデータ流へのリアルタイムのアクセスを提供し、それ自体のスレッド上で動作することができる。同期プラグインは、一般には、リアルタイムスタイラスオブジェクトのスレッド上で動作してもしなくてもよく、非同期プラグインは、一般には、アプリケーションのユーザインターフェース（UI）スレッド上で動作してもしなくてもよい。動的描写の如き、データ流へのリアルタイムのアクセスを必要とし、計算的に軽い作業のためのプラグインを同期スレッド上に配置することによって、同期スレッドを非同期スレッドから分離することができる。インク収集の如き、データ流へのリアルタイムのアクセスを必要としない作業のためのプラグインを同期スレッド上に配置することができる。

【0139】

特定の作業は、計算的に重い作業であっても、多筆画ジェスチャ認識のように、タブレットペンデータ流へのリアルタイムアクセスに近づくことを必要とすることもある。これらのタスクを扱うためのプラグインを一方または他方のスレッド上で分類してもよいし、図12および13に示されているように、縦続リアルタイムスタイラスオブジェクトの一部とすることができる。

【0140】

縦続

10

20

30

40

50

リアルタイムスタイラスインスタンスは、多くの面で、拡張性および方策を包含するスレッドホストである。図13は、どのようにして多数の同期スレッドがともに機能できるかの例を示す図である。ジェスチャレコグナイザを位置1304の同期スレッド上に配置してもよいし、位置1313の同期スレッド上に配置してもよい。

【0141】

デベロッパが多重糸状に接続したい多機能が存在していれば、デベロッパは、多数のリアルタイムスタイラスインスタンスを縦続し、それらの機能を同期プラグインとして挿入することが可能である。

【0142】

この特徴を示唆する具体的なシナリオは、「完全無中断」インキングを実現することである。ジェスチャレコグナイザは、デベロッパが多筆画ジェスチャに特に関心があるときに、認識結果を得るために時々著しい待ち時間をもたらすことがある。したがって、ジェスチャレコグナイザは、動的描写体を妨害する可能性を有するため、動的描写体が配置されるリアルタイムスタイラスの同期集合上に存在しないことがある。ジェスチャレコグナイザは、非同期プラグインを有するインク集合オブジェクトとともに配置されることもある。

【0143】

図13に示されるシステムは、計算的に重い動作をジェスチャレコグナイザ、またはデータベースへのアクセス、ウェブサーバへの書込み、および画面の再生等を含む動作のための他のスレッドから分離したい場合に有益である。ジェスチャ認識が、「完全非遮断」体験の不可欠の部分である場合は、ジェスチャレコグナイザを独自のスレッド、すなわち第2のリアルタイムスタイラススレッド上で動作させることができる。

【0144】

リアルタイムスタイラスは、非同期インターフェースを実装して、単なる非同期プラグインインスタンスとして挙動することを可能にすることによって、この使用をサポートすることができる。どのようにしてリアルタイムスタイラスを他のリアルタイムスタイラスに追加できるかを以下に説明する。実施してもしなくてもよいいくつかのオプションを以下に提示する。

1. リアルタイムスタイラスを非同期プラグインの集合に追加するときは、システムは、他の非同期プラグインインターフェースを追加できるように関係を制限することができる。

2. 様々な数のリアルタイムスタイラスオブジェクトを多次元に沿って縦続できる。例えば、多数の「リアルタイムスタイラス」オブジェクトを単一の「親」リアルタイムスタイラスに直接接続するか、あるいは多数の「リアルタイムスタイラス」を、各々が隣に縦続するように直列に接続するか。あるいは「リアルタイムスタイラス」オブジェクトを上記の構成の組合せで接続することが可能である。実行者は、簡潔性または信頼性に応じてこれらの組合せを限定するように選択することが可能である。

3. 同一のリアルタイムスタイラスを多数の親リアルタイムスタイラスから多数の非同期プラグインの集合体に追加するときは、縦続リアルタイムスタイラスは、通常通り機能できる。あるいは、縦続（子）リアルタイムスタイラスインスタンスは、機能を停止し、「エラーデータ」オブジェクトをキューに挿入することができる。「エラーデータ」オブジェクトは、データにエラーがあるか、または前のプラグインが、データを処理している間にエラーに遭遇したことを下流のオブジェクトに示す指標でありうる。勿論、これは、エラーを処理する1つのアプローチにすぎない。他のアプローチを用いてもよい。

【0145】

一実装形態は、縦続のための集合点として非同期プラグイン集合を再使用できるのに対して、他の実装形態は、縦続RTSを接続するための専用接続点を有することによって、縦続の各RTSが、同期および非同期プラグインの独自の独立集合を有することを可能にできることを注記しておく。

【0146】

「エラーデータ」オブジェクトの存在は、縦続から独立していてもよいことが理解される。そのため、単一のリアルタイムスタイル構成要素、ならびに縦続リアルタイムスタイル構成要素に対して、「エラーデータ」オブジェクトが生成されうる。

【0147】

ダイナミックプラグイン集合の修正

性能便益のために、リアルタイムスタイルを使用不可とし、次いで再度使用可能とすることなしに、リアルタイムスタイル上の同期および非同期集合を修正できる。したがって、リアルタイムスタイルの範囲で実施される「使用可能」または「使用不可」動作が存在する保証はない。

【0148】

プラグインが初期化&浄化できるタイミングを確保するために、「人工的な」（使用可能&使用不可に対する真のクライアントコード呼出に起因しないことを意味する）「リアルタイムスタイルスイネーブルド」&「リアルタイムスタイルディスエーブルド」呼出を、プラグイン集合に動的に挿入され、またはそこから除去されるプラグインに対して行うことができる。

【0149】

プラグインを追加または除去するときに起こりうる挙動のいくつかを以下に記載する。

- ・プラグインを使用可能リアルタイムスタイルインスタンスに追加するときは、人工的な「リアルタイムスタイルスイネーブルド」呼出をこのプラグインに対して行うことができる。

- ・プラグインを使用不可リアルタイムスタイルインスタンスに追加するときは、何も行われない。

- ・プラグインを使用可能リアルタイムスタイルインスタンスから除去するときは、人工的な「リアルタイムスタイルディスエーブルド」呼出をこのプラグインに対して行うことができる。

- ・プラグインを使用不可リアルタイムスタイルインスタンスから除去するときは、何も行われない。

【0150】

これらの関係を維持する方法は、リアルタイムスタイル使用可能呼出を受ける毎にカウンタをインクリメントすることを含んでも含まなくてもよい。このカウンタは、リアルタイムスタイル使用不可呼出を受ける毎に、デクリメントされる。カウンタが1のときは、このプラグインは機能する。カウンタが1より大きくなると、このプラグインは例外を生じ、その機能を抑制する。カウンタが1に戻っても、このプラグインは使用不可状態を維持する。カウンタが再び0に戻らなければ、プラグインは、再度使用可能にならない。他のアプローチを用いることもできる。

【0151】

エラー伝搬

設計時

開発環境にある間は（例えばマイクロソフトコーポレーションによるVisual Studio .NET）、デベロッパは、トライキャッチでキャッチされるか否かにかかわらず生じる任意の例外で中断することがある。したがって、エラー報告は、「リアルタイムスタイル」枠組の無効な構成を検出することを目的として、端的なものとなる。

【0152】

実行時

エラー報告／伝搬の問題は、リアルタイムスタイルおよび同期プラグインが、UIスレッド（アプリケーションスレッド）と異なるリアルタイムスタイルスレッド（インクスレッド）上で実行されているため、実行時エラーの場合は困難になる。同期プラグインが例外を生じるのであれば、それをリアルタイムスタイルでキャッチすることが可能であるが、例外は、リアルタイムスタイルスレッド（インクスレッド）上に存在し、制御フローの起源はリアルタイムスタイルであるため、リアルタイムスタイル以外に行く

10

20

30

40

50

ところはない。エラーをユーザインターフェーススレッドに伝搬し、クライアントコードが優雅にエラーに対処することを可能にすることは困難であるといえる。以下に解決策を記載する。

【0153】

NETの標準的なエラー処理は、例外が生じたときにプラグインからイベントを発し、UIスレッドにおけるデベロッパのエラー処理コードにこのイベントを検出させる。しかし、これは、プラグインがイベントを発すると、その例外を生じたデータがキューイングのためUIスレッドに到達していない可能性があるため、リアルタイムスタイラスに対しては機能しない。例外がエラー処理コードに対して生じた、コンテキスト（すなわち、不良データならびに先行および後続データ）を利用できない優雅なエラー処理を実施することは困難である。このアプローチは、エラーダイアログを立ち上げ、アプリケーションを終了するといった単純なエラー処理に対してのみ好適である。

10

【0154】

プラグインは、デベロッパの制御下にある唯一のコードブロックであるため、リアルタイムスタイラス枠組内でエラー処理を実施するための適正な場所は、プラグイン（同期または非同期）にある。

【0155】

UIスレッド（アプリケーションスレッド）においてエラー処理を実施することが困難になる（例えば、キューはエラー処理に対してそのままの状態を維持する必要がある）ため、リアルタイムスタイラススレッド（インクスレッド）は、同時プラグインから生じた例外に際して直ちに機能を停止してはならない。また、1つのプラグインが無効な入力であると見なすものは、他のプラグインにとって完全に有効なことがある。

20

【0156】

図15は、可能な解決策を示す図である。ここで、例外情報は、エラーデータとして、再びキューに通される。プラグインにおいてエラー処理を実施することができる。プラグインに対して生成されたデータ流、すなわちキューを通じて例外情報を処理することができる。

【0157】

図15に示されるように、リアルタイムスタイラス603は、プラグイン606～608からの全ての例外をキャッチし、エラーデータオブジェクトを作成する。以下に、エラーを識別するためのプロセスを示す。

30

a. 例外が、同期または非同期プラグインに対するインターフェース上でエラー方法に生じた場合は、例外発生プラグインへの呼び戻しによる無限のループを回避するために、エラーデータオブジェクトを作成しない。

i. 代替的なアプローチは、例外を生じた実際のデータの前に、エラーデータオブジェクトをすぐに集合における残りのプラグインに通すことである。

b. 同期または非同期プラグイン法に対するインターフェース上で他の方法に例外が生じた場合は、まず例外発生プラグインにエラーデータオブジェクトをすぐに送り、次いで、例外を生じた実際のデータの前に、集合における残りのプラグインに通すことができる。

40

c. 例外が同期プラグインに生じた場合は、エラーデータを最終的に出力キューに入れ、または非同期プラグインに送る。

d. エラーデータが集合を通った後に、例外を生じたデータが、集合における残りのプラグインを通る。

【0158】

専用の方法（例えば「Iスタイラス同期プラグイン．エラー」法または「Iスタイラス非同期プラグイン．エラー」法）を通じて、エラーデータオブジェクトをプラグイン集合に通すことができる。

【0159】

より具体的には、プラグインが例外を投げると、データの正常な流れが妨害される。リ

50

リアルタイムスタイルオブジェクトは、エラーデータオブジェクトを生成し、例外を投じたプラグインの「I スタイル同期プラグイン、エラー」または「I スタイル非同期プラグイン、エラー」法、ならびにその集合における残りのプラグインの「I スタイル同期プラグイン、エラー」または「I スタイル非同期プラグイン、エラー」法を呼び出す。例外を投じたプラグインが同期プラグインである場合は、エラーデータオブジェクトを出力キューに追加する。次いで、リアルタイムスタイルオブジェクトは、オリジナルデータの処理を再開する。

【0160】

プラグインがそのエラー法から例外を投じる場合は、リアルタイムスタイルオブジェクトは、その例外をキャッチするが、新たなエラーデータオブジェクトを生成しない。あるいは、実装形態またはデベロッパの要望に応じて、新たなエラーデータオブジェクトを生成することができる。この代替的なアプローチにおいて、リアルタイムスタイルは、次いで、反復数を制限することによって、任意の生成ループを低減することもできるし、またはそのリスクを全く無視することもできる。

【0161】

同期プラグインによって追加されたカスタムスタイルデータに関して、エラーデータは、エラーデータを作成した例外の前に「即時出力」キューに追加される任意のカスタムスタイルデータの後で、同期プラグイン集合における後続のプラグインによって「即時出力」キューに追加される任意のカスタムスタイルデータの前に、出力キューに追加される。同期プラグインのいずれかが、エラーデータに対応して、カスタムスタイルデータを「即時出力」キューに追加する場合は、データは、エラーデータの直前に追加される。同期プラグインのいずれかが、エラーデータに対応して、カスタムスタイルデータを出力キューに追加する場合は、データは、エラーデータの直後に追加される。

【0162】

リアルタイムスタイルオブジェクトは、そこから例外が投げられるスレッド上で、「I スタイル同期プラグイン、エラー」法を呼び出す。

【0163】

管理／非管理実例

図16は、本発明の態様を挿入する様々なアプローチを示す図である。該システムは、C#管理オブジェクトの集合によって包まれたCOMオブジェクトを含むことができる。あるいは、Java（登録商標）、C++等を含む任意のオブジェクト指向言語を使用してもよい。

【0164】

図16は、管理または非管理空間に存在するリアルタイムスタイル構成要素を示す図である。図16は、動的描写体、非同期プラグインY、非同期プラグインおよびジェスチャレコグナイザを含む。

【0165】

ペンサービス1601は、非管理コード領域にその入力キュー1603および出力キュー1604を有するリアルタイムスタイル構成要素1602にスタイルデータを送る。スタイルデータは、管理コード領域におけるリアルタイムスタイル構成要素1605に送られる。

【0166】

動的描写体は、リアルタイムスタイル構成要素に接続された最初の同期プラグインである。非管理空間において、そのデータキャッシュ1616を有する動的描写体1608は、リアルタイムスタイル構成要素1602に接続される。同様に、管理空間において、そのデータキャッシュ1617を有する動的描写体1615は、リアルタイムスタイル構成要素1605に対する同期プラグイン集合の一部である。同期プラグイン集合における次の同期プラグインは、同期プラグイン1607である。同期プラグイン1607は、リアルタイムスタイル構成要素1605に対する同期プラグイン集合における動的描写体の後に続く。同期プラグインY1607は、管理空間に存在するにすぎないため、同

期プラグインラッパー 1618 は、非管理「リアルタイムスタイル」構成要素 1602 が、管理／非管理境界を越えて同期プラグイン 1607 にアクセスすることを可能にする。

【0167】

図 16 は、非同期プラグインの集合をも示す図である。非同期プラグインは、同期プラグイン 1612 およびジェスチャレコグナイザ 1609 を含む。非同期プラグイン 1612 は、管理空間における「リアルタイムスタイル」構成要素 1605 に接続された非同期プラグイン集合における最初の非同期プラグインである。非同期プラグイン 1612 は管理空間に存在するため、非同期プラグインラッパー 1619 を使用して、非管理「リアルタイムスタイル」構成要素 1602 からそこへのアクセスを可能にすることができる。ジェスチャレコグナイザは、管理空間と非管理空間の両方に存在する。そのデータキャッシュ 1610 を有する管理ジェスチャレコグナイザ 1609 は、非同期プラグイン 1612 の後にアクセスされる次のプラグインである。ジェスチャレコグナイザ 1609 は、（そのデータキャッシュ 1614 を有する）ジェスチャレコグナイザ 1613 の非管理バージョンと情報を交換することができる。

10

【0168】

図 16 の管理側と非管理側の間を横切るデータを、非管理空間で使用する構造と、管理空間で使用する空間との間で変換または「配列」することが必要とされることがある。マイクロソフトコーポレーションによる、NET フレームワークは、この配列の多くを自動的に行うインターオペラビリティ層を提供する。この追加的なデータ処理は、黙視的な性能ペナルティを招くため、図 16 に示される設計は、インターオペラビリティ層の交差を最小限にするように調整される。

20

【0169】

RTS 1605 に対する管理ラッパーと非管理 RTS 1602 との関係は、ネイティブ（非管理 RTS 1602）にとって、管理 RTS 1605 が他の RTS イベントシンクのように見えることである。そのコンストラクタにおける動的描写体 1615 でインスタンス化されると、RTS 1605 は、対応する非管理動的描写体 1608 にアクセスし、その後の同期プラグイン集合にフックする。

【0170】

管理ジェスチャレコグナイザオブジェクト 1609 は、非同期プラグインの完全な実装形態でありうる。それは、パケットデータを管理側に蓄積し、それをインターオペラビリティ層に沿って配列し、非管理コード実装 1613 に送り込む。戻りコードは、パケットがジェスチャに対応しているか否かについてのものである。

30

【0171】

図 16 には、管理および非管理動的描写体オブジェクト 1615 および 1608 も示されている。管理動的描写体オブジェクトは、非管理動的描写体 1608 の特性に対する薄いラッパーであってもよい。管理動的描写体 1615 はオプションである。ここで動的描写体 1615 がインスタンス化されない場合は、それを他の同期プラグインの 1 つとしてインスタンス化することができる。

【0172】

図 16 のシステムを作成するためのプロセスを以下に示す。

40

a. 第 1 に、デベロッパは、管理非同期プラグイン 1612 および管理動的描写体 1615 をインスタンス化する。内的には、動的描写体 1615 は、特性設定を下に送るために、非管理動的描写体をインスタンス化する。

b. 第 2 に、デベロッパは、動的描写体 1615 に特性を設定する（描画属性等）。

c. 第 3 に、デベロッパは、リアルタイムスタイル 1602 をインスタンス化して、リアルタイムスタイルコンストラクタにおける動的描写体 1615 を処理する。いくつかの事柄が発生しうる。

i. 管理リアルタイムスタイル 1605 は、（公用密封アクセス機構を介して）非管理動的描写体 1608 のアドレスを問い合わせる。

50

i i. 管理リアルタイムスタイラスは、それ自体をペンサービス1601にフックするネイティブリアルタイムスタイラス1602をインスタンス化する。

i i i. 管理リアルタイムスタイラス1605は、リアルタイムスタイラスイベントシンクとして、それ自体をネイティブリアルタイムスタイラス1602にフックする。

i v. 管理リアルタイムスタイラス1605は、ネイティブ動的描写体1608をその同期出力にフックする。

v. 管理リアルタイムスタイラス1605は、それ自体をネイティブリアルタイムスタイラス1602の同期プラグインスレッドにフックする。

d. 第4に、デベロッパは、場合によっては管理ジェスチャレコグナイザ1609を含む1つまたは複数の管理同期および／または非同期プラグインオブジェクトをインスタンス化する。

e. 第5に、デベロッパは、管理プラグインオブジェクトを集めて、様々なプラグイン集合にする（これは、プラグインにアクセスする順番を「リアルタイムスタイラス」構成要素に通知することによって実施できる。あるいは、プラグインを互いに連鎖させるために、特性を設定し、ここでは第1のプラグインを参照することができる）。

f. 第6に、デベロッパは、管理非同期プラグインオブジェクト1612を管理リアルタイムスタイラス1605の「インク集合オブジェクト」特性に対応づける（したがって、非同期プラグインオブジェクト1612を非同期プラグイン集合にフックする）。

g. 第7に、デベロッパは、「R T S. イネーブルド」を「真」に設定する。これにより、管理リアルタイムスタイラス1605に、ネイティブリアルタイムスタイラス1602上で「Iリアルタイムスタイラス>イネーブルド」を「真」に設定させることもできる。

h. 第8に、イベントが、リアルタイムスタイラス1602を流れ始める。

【0173】

工程(c)を代替的な構成で実現することができる。例えば、APIは、R T Sコンストラクタ上にプラグイン特有のアコモデーションを有することができない。その代わりに、R T Sの様々なオーバーロードが、R T S自体の使用法（例えば、ウィンドウまたは制御機構に接続する、1つまたは多くのデジタイザに対応づける、マウスでペンの挙動を表現する等）を決定づける。インターオペラビリティ境界を複数回横切るデータの性能により「動的描写体」パラメータを使用することができる。その結果、「動的描写体」を構築的に使用して、プラグイン集合またはチェーンにおける最初または最後の同期プラグインとすることができる。

【0174】

図16は、インターオペラビリティ交差を最小限にし、「動的描写体」がプラグインの順序リストのいずれかに存在することを可能にする構造のアプローチを示す図である。

【0175】

図16のシステムを使用するためのプロセスを以下に記載する。

a. ペンサービス1601は、R T S1602上に機能呼びだし、「リアルタイムスタイラス」構成要素1602に対応づけられた同期プラグイン集合に蓄積されるデータを送る。場合によっては、入力キュー1603からのデータをプラグイン集合に送り込むこともできる。

【0176】

b. 新たなデータが入力キュー1603に出現したとき（または、それがペンサービス1601に接触させるインターフェース状に適切な方法が呼び出されたとき）に、ネイティブR T S1602は以下のことを実施する。

i. パケットをデジタイザ空間からH I M E T R I C（「インク空間」）に変換する。

i i. 逆変換（H I M E T R I C→デジタイザ）をパケットデータに埋め込む。

i i i. データに対する適切な機能を同期プラグイン集合インターフェース（この場

10

20

30

40

50

合は管理 R T S) 上に呼び出すことによって、変換されたパケットデータをプラグイン集合の第 1 の同期プラグインに送り込む。

【0177】

c. 管理 R T S 1605 は以下のことを実施する。

i. 適切な管理データ構造を作成して、管理に適した様式でデータを保持する。

i i. 収集された管理同期プラグイン上に適切な機能呼び出す。

i i i. 各プラグインがデータを処理し、次のプラグインを呼び出すことができるように「リアルタイムスタイル」構成要素 1605 を戻すことを確認する。

i v. ジェスチャレコグナイザ 1609 上に対応する機能呼び出すまで繰り返す (ジェスチャレコグナイザ 1609 が使用される場合)。

10

【0178】

d. 動的描写体 1615 を「リアルタイムスタイル」1605 によって呼び出すことができる。動的描写体 1615 は、新たなパケットを描写するのを完了すると、戻り、非管理 R T S 1602 は、パケットデータが修正されたかどうかを検出し、適切なデータをキューに入れる。追加すべき何らかの「非即時的」ユーザデータ項目が存在した場合は、ここでそれらを追加する。

【0179】

e. 次いで、ネイティブ R T S 1602 は、接続されたウィンドウに私的なメッセージを貼りつける。

【0180】

20

f. 次いで、ネイティブ R T S 1602 は、入力キュー 1603 を調べて、何らかのリアルタイムデータが追加されたかどうかを確認し、追加された場合は、同期プラグイン集合を再び呼び出し、そのデータを送る。データが追加されなかった場合は、ネイティブ R T S 1602 は戻り、ペンサービス 1601 が、前進し、新たなデジタイザデータとともに再び R T S 1602 を呼び出すことを可能にする。

【0181】

g. 私的メッセージは、データをキュー 1603 から検索し、適切な機能を接続された同期プラグインインターフェース上に呼び出す R T S 1602 上のサブクラスプロクによって確認される。

【0182】

30

h. この場合、管理 R T S 1605 は、それ自体を接続し、呼び出される。次いで、管理 R T S 1605 は、非同期プラグイン 1612 上に対応する機能呼び出す。これを非同期プラグインラッパー 1619 に通すことができる。

【0183】

i. 次いで、管理非同期プラグイン 1612 は、データを非同期的に処理し、適宜描写または蓄積する。

【0184】

j. ジェスチャレコグナイザ 1609 を「リアルタイムスタイル」構成要素からの非同期プラグイン集合の一部として呼び出すことができる。ジェスチャレコグナイザ 1609 は、「カーソルアップ」に遭遇し、その時点で蓄積パケットが、認識のためにインターオペラビリティ境界を通過してネイティブジェスチャレコグナイザ 1613 に送られるまで、内部にパケットを蓄積する。

40

i. この動作は、どのジェスチャが認識されたかに関する情報、およびどれだけの筆画が可能なジェスチャの各々に対応するかに関する情報を戻す。

i i. 次いで、ジェスチャレコグナイザ 1609 は、(「即時」が偽に設定された) 管理 R T S 1605 上で「ユーザデータをキューに追加」法を用いて、ジェスチャ認識の結果を非同期的消費のためにキュー 1604 に入れる (これにより、新たなジェスチャデータが、すぐにインターオペラビリティ境界 (またはインターオブ) を横切り、ネイティブプラグイン集合が完成するまで他のキューにとどまる)。

i i i. また、従来の管理イベントをあらゆる聞き手に発して、今ジェスチャが行

50

われたことをリアルタイムで世界に通知する。

i v. 次いで、ジェスチャレコグナイザ1609によって現行のデータをプラグイン集合におけるあらゆる追加的プラグインに対する「リアルタイムスタイラス」構成要素1605に戻す。

【0185】

データ集合および流れ

イベントシンクハンドラ、およびそれらに伴う情報を以下に列記する。

・ソース I リアルタイムペン

このバックポインタは、イベントシンク実装に原始「リアルタイムスタイラス」へのバックリファレンスを与える。このポインタは、2つのこと、すなわち (a) スタイラスインスタンスに関するコンテキスト情報にアクセスすること (例えば、動的描写体1615は、デジタイザ座標と画素の間で単位変換ができることが必要である)、および (b) 多数の RTS イベントチェーンの間で共用することを可能にする。

・タブレットコンテキスト ID

リアルタイムデータの消費者が、効率的にデータを蓄積し、タブレット ID をデータ集合における指標として用いることを可能にする。

・パケットデスクリプション

パケット呼出におけるデータのレイアウトを描く。異なるデジタイザが、(例えば) x、y、圧力、角度を含む異なる量の情報をサポートする。「パケットデスクリプション」は、パケットの受取側に、最終的に到達する整数の平坦なアレイにどのようにしてデータを配置するかを示す。

・カーソル ID

リアルタイムデータの消費者が、効率的にデータを蓄積し、カーソル ID をデータ集合における指標として用いることを可能にする。

・スタイラスインフォ

タブレットコンテキスト ID と、カーソル ID と、反転状態 (これは、関連性、およびカーソルがペンの「消しゴム」端に対応するか、「執筆」端に対応するかを示す3値状態である) と、ボタン状態 (32までのボタンの各々が押されるかどうか) と、必要に応じて他の関連情報とを含む便利な情報を集約した単純な構造。

・プロパティカウントパーパケット

デベロッパが、API 呼出を用いずにパケットデータのレイアウトを推定することを可能にする妥協パラメータ。(X、Y) は、第1になり、その順番になるように保証され、カウントは、x y データのみに関心を持つデベロッパが、あらゆる n 番目の要素を飛ばしながらリストを閲覧することを可能にする。

・カウントオブパケット

効率のために「結束された」パケットの数 (ペンサービスおよび／またはデジタイザデバイスドライバは、いつ、そしてどれだけのパケットがこのようにして束ねられるかを判断する)。

・パケットデータ

これは、パケットデータの読取り専用コピーである。

・カウントオブパケットレフ

プラグインデベロッパが割り当てることができる修正パケットデータ集合におけるパケットの数。

・パケットデータレフ

これは、リアルタイムデータ操作を含むシナリオを達成するために、パケットデータを修正する機能をプラグインオブジェクトのデベロッパに与えるパケットデータのリファレンスである。

【0186】

一実装形態において、プラグイン集合実装が、プラグイン集合に入る前にパケットデータが修正されたかどうかを「把握」することを可能にすることができない。あるいは、パ

10

20

30

40

50

ケットデータが修正されたかどうかをプラグイン集合に知らせることができる。例えば、データ上のフラグ、またはデータから独立したフラグを設定することができる。プラグイン集合が、パケットデータが修正されたかどうかを把握することを可能にする1つの利点は、警告のない修正パケットデータが無視する機能をプラグイン集合に提供することである。他方で、このアプローチは、デベロッパが誤りを犯すさらなるオプションを提供する。より重要なことは、プラグインはデータの起源に対して認知不能ではないため、設計パターンの簡潔性とオブジェクト指向の性質を混同させる。プラグインがオリジナルデータと修正データの両方を調べることができれば、前のプラグインに属する動作を恐らく実施している。

【0187】

性能／効率のために、デベロッパがデータを修正することを意図しなければ、メモリを割り当てずにすむように、2つのデータを内部に維持することができる。

【0188】

データ同期

図17Aに示すように、同期を確保するために、どのようにしてデータをキューに追加できるかを以下に説明する。

【0189】

ICOは、UIスレッド上に到来する他の要求と、インキングスレッドを通じて到来する保留データとを同期させるために、シグナリング機構として入力および出力キューを用いることができる。

【0190】

図17Aは、システムが情報を処理しながら行われるモード切換えを示す図である。システムは、1701における（において「実施」された）第1の処理の結果を示す図である。時刻1702において、新たなスタイルデータを受け取り、時刻1702でプラグイン1によって、時刻1704でプラグイン2によってその処理を開始し、時刻1705で完了する。時刻1703においてモード切換えが行われる。例えば、アプリケーションのユーザが、「編集モード」を「インク」から「消しゴム」に変更し、アプリケーションがビジーである場合は、ユーザモードが変化してからある時間にわたって、キューから流れるパケットデータがまだ「インク」であるという合理的な可能性がある。

【0191】

しかし、このときにモード切替えイベントをキュー1706に追加する代わりに、スタイルデータに対して、時刻1705で現行の処理が完了された後まで遅らせる。これを2通りの方法で行うことができる。第1に、時刻1702でプラグインにおいて開始するスタイルデータが完成されるまでモード切換え動作イベントを遅らせることができる。第2に、システムは、時刻1705におけるプラグイン2からの結果をキュー1706におけるその前に配置するための十分な余地を残しながら、モード切換えイベントをキュー1706に入力することができる。そこで、この場合、ICOは、ユーザがモードを変えた瞬間に、「マーカ」を出力キューに挿入することができる。ある時間が経過した後に、マーカは、（リアルタイムスタイルプラグイン集合インターフェース上の「カスタムデータ追加」法を介して）ICOに戻され、その時点で、ICOが、入力パケットデータを「消しゴム」要求と解釈しはじめることが可能になる。

【0192】

図17Bは、図17Aの一般的なバージョンを示す図である。第1のスレッドに対するプロセスの終了がイベント1707によって示されている。それはキュー1712に配置される。次の時刻1708において、第1のスレッドに対する他のプロセスが開始する。時刻1709において、第2のスレッド上にイベントが発生する。第1のスレッドに対するプロセスが点1708で開始したため、プロセスは、時刻1709でのイベントの前に扱われるものと期待される。しかし、第1のスレッドに対するプロセスは、時刻1710で続行し、次いで時刻1711で終了するため、そのプロセスの開始時刻1708に対する時刻1709におけるイベントの挿入のタイミングは、時刻1711でプロセスが実施

10

20

30

40

50

される前に時刻1709におけるイベントがキューに配置されることになるという点において、好ましくない。ここで、図17Bは、キュー1712における少なくとも1つの箇所を飛ばすことにより、点1707で実施されたプロセスから生成されたイベントからキュー1712の下流に距離をおいて挿入されている時刻1711におけるイベントを示す図である。キュー1712においてある場所を利用可能に維持することにより、プロセス1708、1710、1711の完成をキュー1712における1709のイベントの前方の箇所に正しく配置することを可能にして、キュー1712におけるパケット／イベントの読取りを期待される順序で行うことを可能にする。ここで、その順序を、それらのプロセスが開始された、またはイベントが生じた順序でパケット／イベントが生じるものとして要約することができる。この順序は、点1713（プロセスの開始）、1714（プロセス続行）および1715（プロセス完了）によって示される次のプロセスについても継続する。

10

【0193】

図18および19は、本発明の態様による、ペンデータを処理するための様々なシステムを示す図である。

【0194】

図18を参照すると、ペンデバイス1801は、データをペンサービス1803に送る。マウスは、情報を生成し、それをユーザ32（ユーザ32、dllとしても知られる）1804に送ることもできる。アルペンデータ（例えばボタンをクリックする）は、マウスイベントを表現することができ、ユーザ32 1804に送られて、マウスイベントとして処理される。同様に、いくつかのマウスイベントは、インクを表現することができ、1814において送られて、ペン入力として処理される。次いで、マウスイベントは、ウィンドウメッセージポンプ1805、次いでHWNDD1806、HWNDDマウスデバイス1812、入力キュー1810、次いで入力マネージャ1811に送られる。ペン入力（例えば、ペンが潜在的なインキングサービスの範囲に移動するとき）、またはスタイラスイベントは、ペン入力マネージャ1807／1808（非管理および管理リアルタイムスタイラスサービス）に送られる。ペン入力マネージャ1807／1808は、ペンサービス1803とアプリケーションの間の全ての通信を処理する。この処理は、通常の優先スレッド、または高優先ステータススレッド上で実施されうる。ペンサービス1803は、様々なイベントを生成することができる。ペンイベントをペン入力マネージャ1807／1808に送って、標準メッセージ通信システム1804～1806および1812をバイパスすることができる。

20

30

【0195】

ペンサービス1803は、以下のイベント、すなわちスタイラスインレンジ（スタイラスがデジタイザの範囲に入った場合）、スタイラスアウトオブレンジ（スタイラスが、デジタイザによって検出されなくなる場合）、スタイラスパケット（デジタイザからの生の点データーペンはデジタイザに接触してもしなくてもよい）、スタイラスインエアポイント、タブレット追加／除去、およびシステムジェスチャイベント（例えば、「押下げおよび保持」ならびにホーバイイベント）を生成することができる。

【0196】

スタイラスは、生のプリビューイベントを生成することができる。これらの生のプリビューイベントをリアルタイムインキングサービス1901によって処理することができる（図19より）。他のサービスもこれらのイベントを検索することができる。リアルタイムインキングサービス1901は、生のプリビューイベントに対して以下の工程を含むいくつかの工程を実施することができる。

40

a. イベントがダウン「パケット」イベントであるかどうかを調べる。そうでない場合は、処理を停止し、ナルに戻る。

b. イベントが、そのキャッシュドレイアウト情報に基づいて、インキング要素上に存在したかどうかを調べる。そうでない場合は、処理を停止し、ナルに戻る。

c. このパケットは、インキング部の内部の「ダウン」イベントに対するものであるた

50

め、増進的に筆画を描く。

d. 最後に、その要素上に筆画が描かれたペン入力マネージャ 1807 / 1808 に戻る。

【0197】

生のプリビュイベントからの戻り値は、そのイベントを「目標」とするペン入力マネージャ 1807 / 1808 によって使用される。ナルでない目標が戻されると、イベントをその特定の目標とともに入力キューに配置することができる。これにより、要素に描かれたインクが、入力キューにおける他のイベントの結果として何らかの他の要素に送られるのではなく、実際にその要素に送られることになる。

【0198】

ここで、スタイラスイベントは、入力イベントになる準備が整っている。しかし、たいていの場合、各スタイラスイベントは、メッセージシステムを流れる対応するマウスメッセージをも有する。ペン入力マネージャ 1807 / 1808 がスタイラスイベントを入力イベントに変換する前に、まずイベントを対応するマウスメッセージと突き合わせる必要がある。必要な場合は、ペン入力マネージャ 1807 / 1808 は、マウスメッセージが到着するのを待つことが可能である。

【0199】

ペン入力マネージャ 1807 / 1808 が、マウスメッセージとスタイラスイベントの両方を有すると、それら 2 つを組み合わせて、適切な入力レポートとし、そのレポートを入力キュー 1810 に配置する。

【0200】

図 20 は、本発明の態様によるキューを示す図である。図 20 は、キュー 2001 を含む。簡単にするために、キューを円として示す。線形キュー、および当該技術分野で知られている他の様々な形を含む代替的なキューの形を用いてもよい。キュー 2001 は、開始ポインタ 2002 および終了ポインタ 2008 を含む。キュー 2001 は、情報を有するいくつかの場所 (2003、2004 および 2009) をも含む。場所 2005 および 2007 は空である。場所 2006 はロックされている。場所 2006 は、異なる供給源からのイベントおよびデータを同期するために場所ホルダとしてロックされていてもよい (その例が図 17A および 17B に示されている)。また、その内部に含まれるデータがプラグイン集合またはチェーンに通されているときに、場所 2006 をロックしてもよい。場所 2010 および 2011 も空である。空の空間の数を必要に応じて増減することができる。例えば、キューが一杯になった場合に、追加的なデータを記憶するために追加的な空間を追加することができる。あるいは、キューに保持されうるデータが破棄されるように、キューのサイズを固定することもできる。これは、システムが他のプロセスに高度に関わっており、キューに対する新たなデータの作成を遅くすることが有益であるということを示す指標をユーザに提示するという点において、有用な利点を提供することができる。情報の破棄は、システムがロックアップされており、システムが再び処理を続けるまで追加的なデータを入力しない方がよいということをユーザに示すこともできる。

【0201】

アプリケーションプログラミングインターフェース

本発明の態様によって使用できる様々なアプリケーションプログラミングインターフェースの概要を以下に提示する。

【0202】

リアルタイムスタイラスは、リアルタイムスタイラス入力クラスの一部であってもよい。インスタンス化されたリアルタイムスタイラスオブジェクトをウィンドウハンドルまたはコントロールに接続することができる。リアルタイムスタイラスは、既定コンストラクタを含むことができる。既定コンストラクタを使用すると、他のリアルタイムスタイラスオブジェクトからの入力のみを受け付けることが可能になる。

【0203】

プラグインが、ペンデータ流と相互作用することを可能にするために、リアルタイムス

10

20

30

40

50

タイラスオブジェクトは、2つのプラグイン集合を維持することができる。それらの集合をリアルタイムスタイラスオブジェクトの特性に規定することができる（例えば、同期プラグインに対する「同期プラグイン集合」および非同期プラグインに対する「非同期プラグイン集合」）。追加法（例えば、プラグインを同期集合に追加するための「スタイラス同期プラグイン集合追加」、またはプラグインを非同期集合に追加するための「スタイラス非同期プラグイン集合追加」）を適切な特性上に呼び出すことによって、プラグインをいずれかの集合に追加することができる。

【0204】

同期プラグインは、あるインターフェース（例えば「I スタイラス同期プラグイン」）を実装することができ、非同期プラグインは、異なるインターフェース（例えば、「I スタイラス非同期プラグイン」）を実装することができる。各プラグインは、そのデータ対象（例えば、「I スタイラス同期プラグインデータ対象」または「I スタイラス非同期プラグインデータ対象」）を規定する特性を有することができる。リアルタイムスタイラスオブジェクトは、プラグインが利用した方法に対する通知方法と呼び出すことができる。

10

【0205】

リアルタイムスタイラスオブジェクトは、「I スタイラス非同期プラグイン」インターフェースを実装する。他のリアルタイムスタイラスから入力を受け付けるリアルタイムスタイラスをインスタンス化することは、既定コンストラクタを使用することを含むことができる。さらなる代替的な機構を使用して、子または縦続RTSを親に対応づけることができる。他の機構は、親RTSに対応づけられた専用ポインタを使用して、子RTSへのリファレンスを保持することができる。ここで、親RTSは、非同期プラグインインターフェースを実装する必要はない。

20

【0206】

リアルタイムスタイラスオブジェクトは、ペンデータを転送する2つまたはそれ以上の内部キューを有することができる。これらは、入力キューおよび出力キューを含む。これらの2つのキューの間に他のキューを追加することができる。また、データが、ペンデータ流から同期プラグインを入力することだけを目的としているときは、入力キューを取り除いてもよい。これらのキューがペンデータを処理するのに用いるプロセスは、少なくとも図6、8、11および13に関連して上述されている。

【0207】

インクを収集する形式上でリアルタイムスタイラスオブジェクトを使用するための例示的なシナリオを以下に記載する。

30

a. 「I スタイラス非同期プラグイン」インターフェースを実装する形式を作成する。
b. その形式上にコントロールに接続されたリアルタイムスタイラスオブジェクトを作成する。
c. その形式が関与するデータに関する形式の特性（例えばデータ対象特性）において、表面に接触するスタイラスに関する通知（スタイラスダウンイベント）、データパケット、およびスタイラスアップ（例えばスタイラスアップイベント）通知に関与するようにその形式を設定する。

d. その形式の「I スタイラス非同期プラグイン. タイラスダウン」、「I スタイラス非同期プラグイン. パケット」および「I スタイラス非同期プラグイン. スタイラスアップ」法において、その形式のリアルタイムスタイラスオブジェクトから送られるスタイラスダウン、パケットおよびスタイラスアップ通知を処理するためのコードを追加する。

40

【0208】

各リアルタイムスタイラスオブジェクトは、それが関与できるオブジェクトに対する一意的識別子のリストを維持する。リアルタイムスタイラスオブジェクトは、一意的識別子とタブレットオブジェクトとの間の翻訳を行うためのいくつかの方法を有することができる。これらは、「タブレットからタブレットコンテキストIDを取得」および「タブレットコンテキストIDからタブレットを取得」法を含むことができる。

【0209】

50

ペンデータは、それ自体の名前空間（例えば、「スタイラス入力、プラグインデータ」名前空間）を占有することができる。

【0210】

タブレット特性記述オブジェクト（例えば「タブレット特性記述」オブジェクト）は、特性包括的一意的識別子（GUID）、および特定のタブレットについての範囲、解像度、および特性の単位を示すタブレット特性計量（例えば「タブレット特性計量」）を含むことができる。

【0211】

一意的タブレット識別子を取得し、タブレットによってサポートされる特性記述オブジェクトの集合を戻す方法が存在しうる。例えば、リアルタイムスタイラスオブジェクトの「タブレット特性記述集合を取得」法は、一意的タブレット識別子を取得し、タブレットによってサポートされる「タブレット特性記述」オブジェクトの集合を戻す。リアルタイムスタイラスオブジェクトの「所望のパケット記述を取得」法は、リアルタイムスタイラスオブジェクトがそのプラグインに送ることになるパケット特性に対するGUIDのアレイを戻す。

【0212】

所定のタブレットに対するプラグインに送られるパケット特性の集合は、「タブレット特性記述集合を取得」法によって戻される集合と、「所望のパケット記述を取得」法によって戻される集合との共通部分でありうる。リアルタイムスタイラスがそのプラグインに送ることになるパケット特性の集合を修正するために、リアルタイムスタイラスオブジェクトの「所望のパケット記述を設定」法を呼び出すことができる。

【0213】

リアルタイムスタイラスオブジェクトが使用可能になると、各プラグインは、「I スタイラス同期プラグイン、リアルタイムスタイラスイネーブルド」または「I スタイラス非同期プラグイン、リアルタイムスタイラスイネーブルド」法の呼出を受け取る。通知で送られた「リアルタイムスタイラスイネーブルドデータ」オブジェクトは、「リアルタイムスタイラス」オブジェクトが使用可能であるときに利用可能なタブレットに対するコンテキスト識別子の集合を含む。「リアルタイムスタイラス」オブジェクトが使用可能である間に、「リアルタイムスタイラス」オブジェクトが使用できるタブレットが、ペン使用可能コンピューティングシステムに追加、またはそこから除去されると、「リアルタイムスタイラスオブジェクト」は、タブレットが追加または除去されたことをそのプラグインに通知する。

【0214】

スタイラスは、追加的なインターフェースを有することができる。各スタイラスは、スタイラスに対応づけられたパケットを示す、スタイラスに対応づけられた情報を有することができる。リアルタイムスタイラスオブジェクトは、スタイラスに関する情報をいくつかの通知方法でプラグインに送る。ペンまたはスタイラスに関する情報をスタイラスオブジェクトによって表現することができる。オブジェクトは、データが収集されたときのタブレットペンの状態のスナップショットである。プラグインは、タブレットペンデータ流の一部としてタブレットペンデータを受け取っているため、プラグインは、カーソルクラスを通じて特定のタブレットペンの現行の状態を調べる代わりに、スタイラスオブジェクトにおける情報を用いる必要がある。

【0215】

プラグインおよび「リアルタイムスタイラス」クラスについて以下に説明する。プラグイン、「I スタイラス同期プラグイン」または「I スタイラス非同期プラグイン」インターフェースを実装するオブジェクトをリアルタイムスタイラスオブジェクトに追加することが可能である。

【0216】

「I スタイラス同期プラグイン」および「I スタイラス非同期プラグイン」インターフェースは、同じ方法を定める。これらの方法は、リアルタイムスタイラスオブジェクトが

10

20

30

40

50

、ペンデータを各プラグインに送ることを可能にする。{I スタイラス同期プラグイン、データ対象} および「I スタイラス非同期プラグイン、データ対象」特性は、各プラグインが、タブレットペンデータ流における特定のデータを利用することを可能にする。プラグインは、潜在的な性能上の問題を最小限に抑える、その作業を実施するのに必要なデータを利用するだけでよい。あるいは、追加的な利用はプラグインに含まれてもよい。

【0217】

リアルタイムスタイラスオブジェクトは、「スタイラス入力、プラグインデータ」名前空間におけるオブジェクトを使用して、ペンデータをそのプラグインに送ることができる。リアルタイムスタイラスは、また、プラグインによって投げられた例外をキャッチする。そうする場合は、リアルタイムスタイラスは、「I スタイラス同期プラグイン、エラー」または「I スタイラス非同期プラグイン、エラー」法を呼び出すことによってプラグインに通知できる。

10

【0218】

プラグインデータおよびリアルタイムスタイラスクラスについて以下に説明する。

【0219】

リアルタイムスタイラスに対するプラグインは、「I スタイラス同期プラグイン」または「I スタイラス非同期プラグイン」インターフェースのいずれかを実装することができる。リアルタイムスタイラスにおける全ての方法を実施することができる。

【0220】

インターフェース上に定められた方法は、「スタイラス入力、プラグインデータ」名前空間におけるオブジェクトを使用して、ペンデータをプラグインに送ることができる。以下の表には、通知方法におけるパラメータであるデータオブジェクトが示されているとともに、その通知に対応づけられた「データ対象マスク」値が列記されている。

20

【0221】

【表 3 - 1】

プラグインデータ	データインタレスト マスク値	摘要	
カスタムスタイラスデータ	カスタムスタイラスデータ追加	プラグインによって追加されるカスタムアプリケーションデータ	
エラーデータ	エラー	リアルタイムスタイラスオブジェクトが、そのプラグインの1つの未処理の例外に応じて追加するエラー情報	
インエアパケットデータ	インエアパケット	スタイラスがデジタイザの空中に存在するときのスタイラスの動きに対するパケット情報	10
インエアパケットデータ	パケット	スタイラスがデジタイザに接触しているときのスタイラスの動きに対するパケット情報	
リアルタイムスタイラスディスエーブルドデータ	リアルタイムスタイラスディスエーブルド	リアルタイムスタイラスオブジェクトが、使用不可であるときに追加する情報	
リアルタイムスタイラスイネーブルドデータ	リアルタイムスタイラスイネーブルド	リアルタイムスタイラスオブジェクトが、使用可能であるときに追加する情報	20
スタイラスボタンダウンデータ	スタイラスボタンダウン	押されている特定のスタイラスボタンに関する情報	
スタイラスボタンアップデータ	スタイラスボタンアップ	解除されている特定のスタイラスボタンに関する情報	

【0 2 2 2】

【表 3 - 2】

スタイラスダウンデータ	スタイラスダウン	スタイラスがデジタイザと接触するときのスタイラスに対するパケット情報	
スタイラスインレンジデータ	スタイラスインレンジ	リアルタイムスタイラスオブジェクトの入力部、またはリアルタイムスタイラスオブジェクトの入力部の情報のデジタイザの検出範囲に入る特定のスタイラスに関する情報	10
スタイラスアウトオブレンジデータ	スタイラスアウトオブレンジ	リアルタイムスタイラスオブジェクトの入力部、またはリアルタイムスタイラス使用不可データオブジェクトの入力部の情報のデジタイザの検出範囲を出る特定のスタイラスに関する情報	
スタイラスアップデータ	スタイラスアップ	スタイラスがデジタイザから引き上げられているときのスタイラスに対するパケット情報	
システムジェスチャデータ	システムジェスチャ	リアルタイムスタイラスオブジェクトが、システムジェスチャを検出するときに追加する情報	20
タブレット追加データ	タブレット追加	追加されているタブレットに関する情報	
タブレット除去データ	タブレット除去	除去されているタブレットに関する情報	

【0 2 2 3】

リアルタイムスタイラスオブジェクトは、タブレットペンに関する情報をいくつかの通知方法でそのプラグインに送る。タブレットペンに関する情報は、スタイラスオブジェクトによって表現される。このオブジェクトは、データが収集されたときのタブレットペンの状態のスナップショットである。プラグインは、タブレットペンデータ流の一部としてタブレットペンデータを受け取っているため、プラグインは、カーソルクラスを通じて特定のタブレットペンの現行の状態を調べる代わりに、スタイラスオブジェクトにおける情報を用いる必要がある。各スタイラスオブジェクトは、データを収集したタブレットに対するタブレットコンテキスト識別子を含む。

【0 2 2 4】

同期および非同期インターフェース上で以下の 1 3 の機能を実装することができる。表は、様々なイベントを通じて送られる情報を比較したものである。イベントおよびパラメータのリストは例示的なものであって、網羅的なものではない。以下のイベントおよびパラメータとともに、またはそれらの代わりに他のイベントおよびパラメータを用いてもよい。

【0 2 2 5】

同期パラメータの多くは、受取り側が、キューを通るデータを修正することを可能にするという点においてのみそれらの同期パラメータと異なる。あるいは、両方のパラメータの集合が、キューを通るデータを修正することを可能にすることができる。

【0 2 2 6】

【表 4－1】

機能	目的	同期パラメータ摘要	非同期パラメータ摘要	
コンテキスト 作成（RTSイネーブルドとしても知られる）	イベントチェーンを使用可能にできるように、タブレットコンテキストが作成されたときを判断するのに用いられる。このイベントの消費者は、これを用いて、タブレット初期化によりRTSが完成され、データのクライアントコードへの発信を開始する準備が整うときを把握することができる。	ペンの供給源（ペンIDを含むが、それに限定されない）、タブレット識別、およびデータパケットの種別を含む。	ペンの供給源（ペンIDを含むが、それに限定されない）、タブレット識別、およびデータパケットの種別を含む。	10
コンテキスト 消去（RTSディスエーブルドとしても知られる）	RTSがそのオブジェクトを一掃できるように、タブレットコンテキストが消去されたときを判断するのに用いられる。このイベントの消費者は、これを用いて、（恐らくRTSイネーブルドが偽であるため）RTSがタブレットコンテキストを解放し、任意のパケット種別データを、それが利用できなくなる前に必要に応じて挿入しようとするときを把握することができる。	ペンの供給源（ペンIDを含むが、それに限定されない）およびタブレット識別を含む。	ペンの供給源（ペンIDを含むが、それに限定されない）およびタブレット識別を含む。	20
				30

【0 2 2 7】

【表 4 - 2】

カーソルニュー ー	デジタイザによって新 たなスタイラスが推奨 されたという通知	ペンの供給源（ペンID を含むが、それに限定さ れない）、およびスタイ ラスに関する情報（活性 化されたスタイラスの ボタン等を含むが、それ に限定されない）を含む 。	ペンの供給源（ペンID を含むが、それに限定さ れない）、およびスタイ ラスに関する情報（活性 化されたスタイラスの ボタン等を含むが、それ に限定されない）を含む 。
カーソルイン レンジ	スタイラスがデジタイ ザの範囲内に移動した という通知	ペンの供給源（ペンID を含むが、それに限定さ れない）、およびスタイ ラスに関する情報（活性 化されたスタイラスの ボタン等を含むが、それ に限定されない）を含む 。	ペンの供給源（ペンID を含むが、それに限定さ れない）、およびスタイ ラスに関する情報（活性 化されたスタイラスの ボタン等を含むが、それ に限定されない）を含む 。

10

20

【0 2 2 8】

【表 4－3】

カーソルアウトブレンジ	スタイラスがデジタルの範囲外に移動したという通知	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む
カーソルダウン	スタイラス先端がデジタル表面に「接触」しているという通知	ソースリアルタイムペン、スタイラスインフォ、プロパティカウントパーパケット、パケットデータレフ	ソースリアルタイムペン、スタイラスインフォ、プロパティカウントパーパケット、パケットデータ
カーソルアップ	スタイラス先端がデジタル表面に接触していないという通知	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、データパケット当たりの特性カウント、およびパケットデータタイプのリファレンスを含むことができる。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、データパケット当たりの特性カウント、およびパケットデータタイプのリファレンスを含むことができる。

10

20

30

【0 2 2 9】

【表 4 - 4】

インエアパケット	スタイラスがデジタル表面の上方に移動したという通知	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、データパケット当たりの特性カウント、およびパケットデータタイプのリファレンスを含むことができる。これは、データパケット緩衝域の長さ、およびデータパケットのカウントを含むこともできる。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、データパケット当たりの特性カウント、およびパケットデータタイプのリファレンスを含むことができる。これは、データパケット緩衝域の長さ、およびデータパケットのカウントを含むこともできる。	10
パケット	スタイラスが、デジタル表面に接触しながら移動しているという通知。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、データパケット当たりの特性カウント、およびパケットデータタイプのリファレンスを含むことができる。これは、データパケット緩衝域の長さ、およびデータパケットのカウントを含むこともできる。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、データパケット当たりの特性カウント、およびパケットデータタイプのリファレンスを含むことができる。これは、データパケット緩衝域の長さ、およびデータパケットのカウントを含むこともできる。	20
				30

【表 4－5】

システムイベント	システムイベントまたはジェスチャの通知。例としては、「タップ」、「ダブルタップ」、「プレス&ホールド」および「シェーク」が挙げられる。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、システムイベント情報、およびシステムイベントに関するデータを含むこともできる。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびスタイラスに関する情報（活性化されたスタイラスのボタン等を含むが、それに限定されない）を含む。これは、システムイベント情報、およびシステムイベントに関するデータを含むこともできる。	10
タブレット追加	システムに接続された新しいデジタイザ（典型的には外部USBデバイス）の通知	ペンの供給源（ペンIDを含むが、それに限定されない）、およびタブレットに関する情報を含む。	ペンの供給源（ペンIDを含むが、それに限定されない）、およびタブレットに関する情報を含む。	20

【0 2 3 1】

【表 4－6】

タブレット除去	システムから切り離されたデジタイザ（典型的には外部USBデバイス）の通知	ペンの供給源（ペンIDを含むが、それに限定されない）、および除去されたタブレットに関する情報を含む。	ペンの供給源（ペンIDを含むが、それに限定されない）、および除去されたタブレットに関する情報を含む。	
ユーザデータ	（例えばGUIDによって）一意的に識別された任意のユーザデータ。これは、キューを通じて情報を下流にインク集合オブジェクトまで送り、それが、上述のリアルタイムデータの残りに対して正しい順序になることを保証しようとする特殊プラグインによって用いられる。	n/a	ペンの供給源（ペンIDを含むが、それに限定されない）、ユーザのデータに対応づけられたGUID、ユーザのデータのサイズの指標、およびユーザのデータの内容を含む。	30
				40

【0 2 3 2】

本発明の態様をその例示的な実施形態の観点で説明した。この開示内容を吟味すれば、添付の請求項の範囲および主旨に含まれる多くの実施形態、改造形態および変更が当業者理解されるであろう。

【図面の簡単な説明】

【0233】

【図1】本発明のある態様を実施することができる汎用デジタルコンピューティング環境の概略図を示す図である。

【図1B】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1C】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1D】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1E】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。 10

【図1F】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1G】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1H】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1I】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1J】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。 20

【図1K】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1L】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図1M】本発明の1つまたは複数の態様をサポートする汎用コンピュータ環境を示す図である。

【図2】本発明の態様によるスタイラスをベースとした入力システムのためのディスプレイを示す図である。

【図3】本発明の態様による電子インクの流れを処理するためのシステムを示す図である。 30

【図4】本発明の態様による電子インクの流れを処理するための代替的アプローチを示す図である。

【図5】本発明の態様をサポートすることができるオブジェクトモードを示す図である。

【図6】本発明の態様による電子インクを処理するためのシステムを示す図である。

【図7】本発明の態様による電子インクを処理するためのシステムを示す図である。

【図8A】本発明の態様による入力キューおよび出力キューを使用するための様々なアプローチを示す図である。

【図8B】本発明の態様による入力キューおよび出力キューを使用するための様々なアプローチを示す図である。 40

【図9】本発明の態様によるインクを処理するためのシステムを形成・使用するための方法を示す図である。

【図10】本発明の態様によるインクを処理するためのシステムを形成・使用するための方法を示す図である。

【図11】本発明の態様によるインク集合オブジェクトを警報するためのプロトコルを備えたシステムを示す図である。

【図12】本発明の態様による追加的なキューを備えたシステムを示す図である。

【図13】本発明の態様による分離されたリアルタイムスタイラス構成要素を備えたシステムを示す図である。

【図14】本発明の態様による、ジェスチャ認識装置がジェスチャを追跡できる深さを示 50

す図である。

【図15】本発明の態様による例外処理のためのシステムを示す図である。

【図16】本発明の態様による管理および非管理コードを示す図である。

【図17A】本発明の態様による、同期を確保するためにキューにおける異なる箇所にとのようしてデータを配置できるかを示す図である。

【図17B】本発明の態様による、同期を確保するためにキューにおける異なる箇所にとのようしてデータを配置できるかを示す図である。

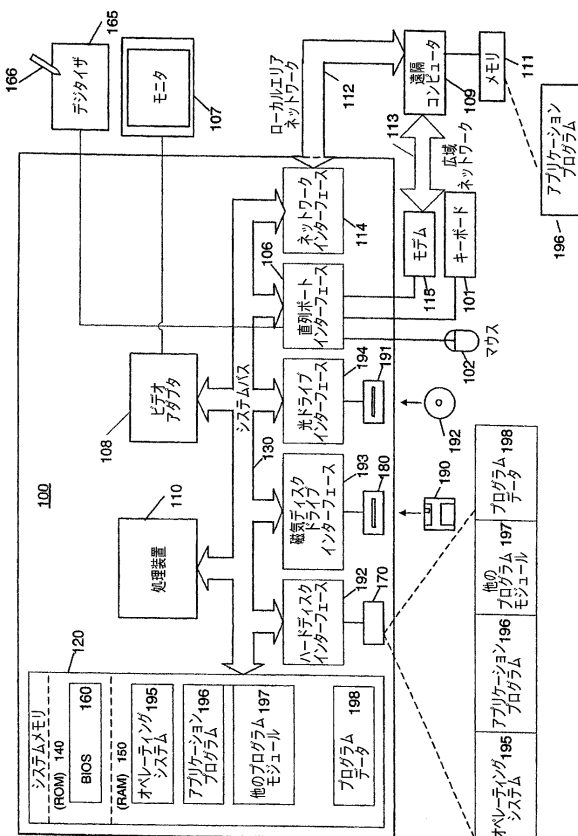
【図18】本発明の態様によるペンデータを処理するための様々なシステムを示す図である。

【図19】本発明の態様によるペンデータを処理するための様々なシステムを示す図である。

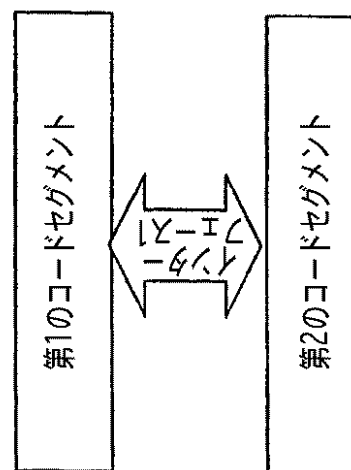
10

【図20】本発明の態様によるキューを示す図である。

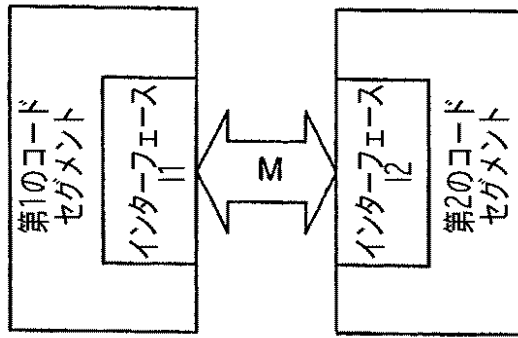
【図1】



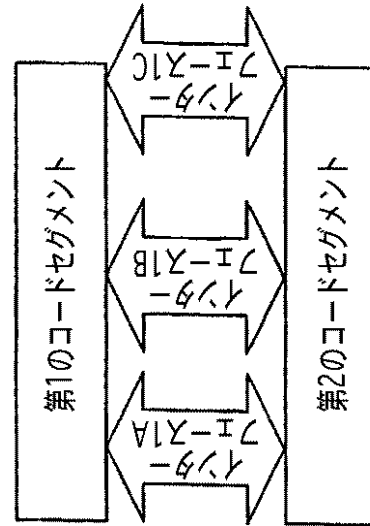
【図1B】



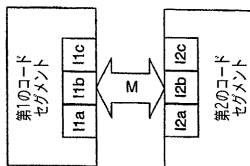
【図 1 C】



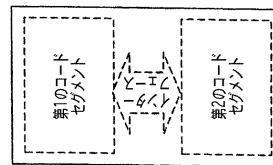
【図 1 D】



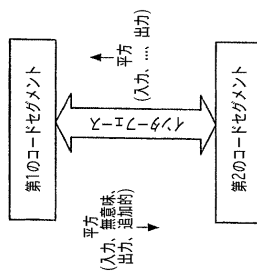
【図 1 E】



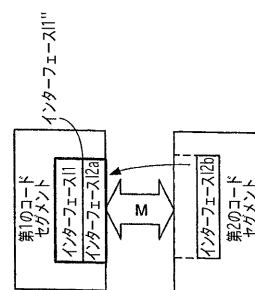
【図 1 H】



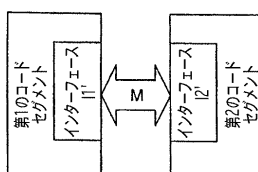
【図 1 F】



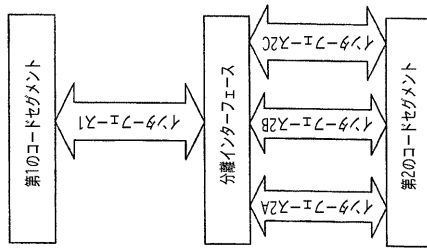
【図 1 I】



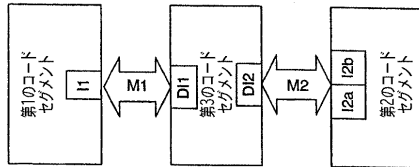
【図 1 G】



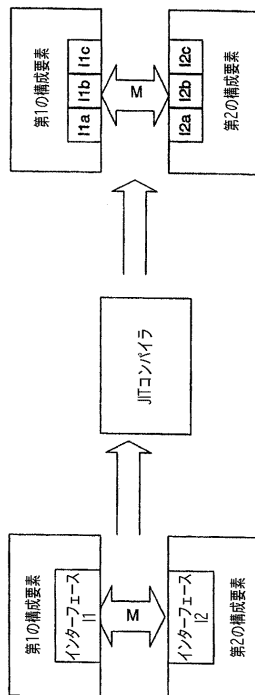
【図 1 J】



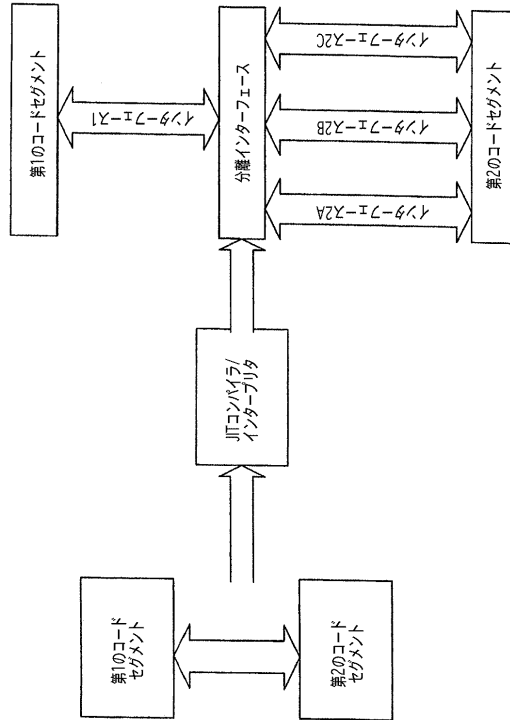
【図 1 K】



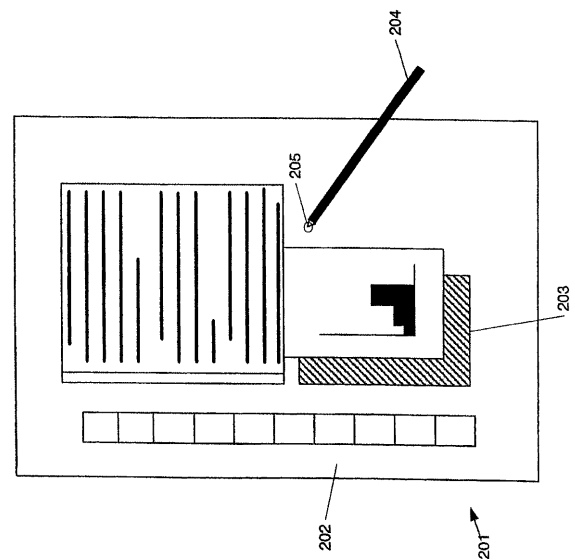
【図 1 M】



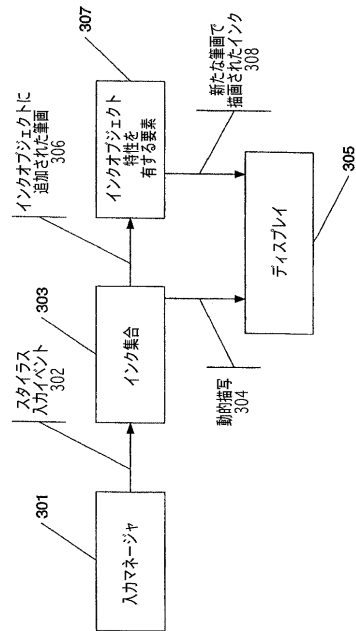
【図 1 L】



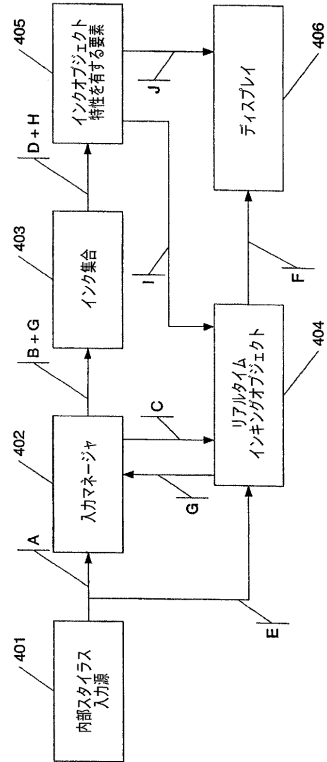
【图 2】



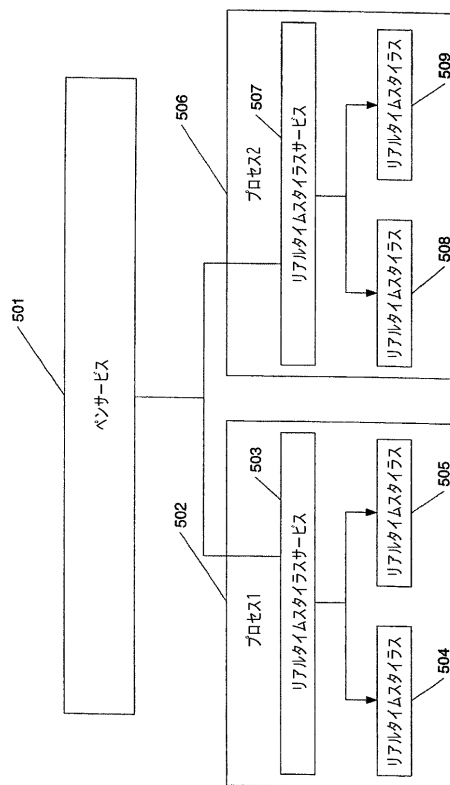
【図 3】



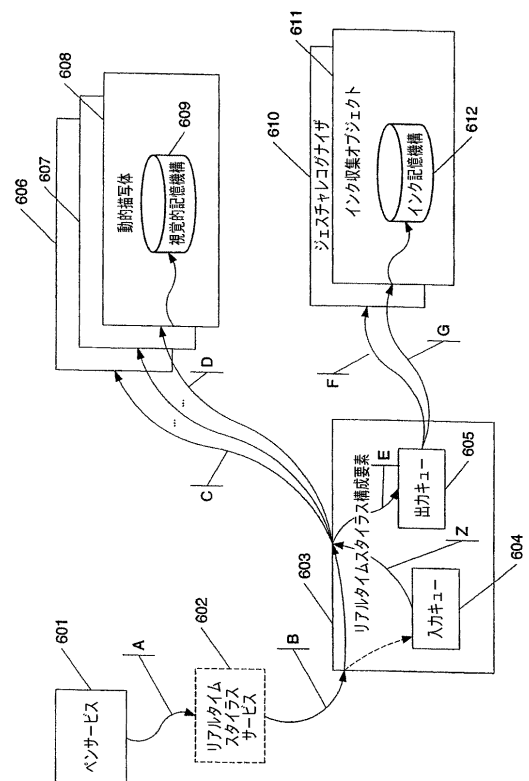
【図 4】



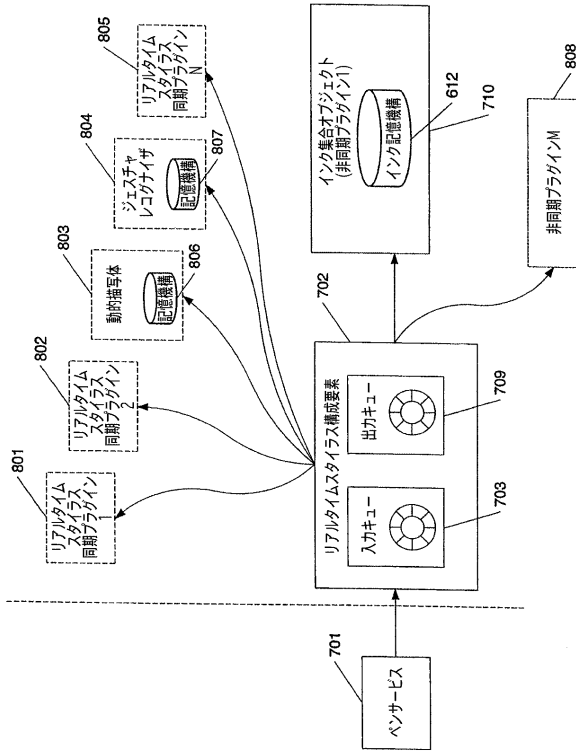
【図 5】



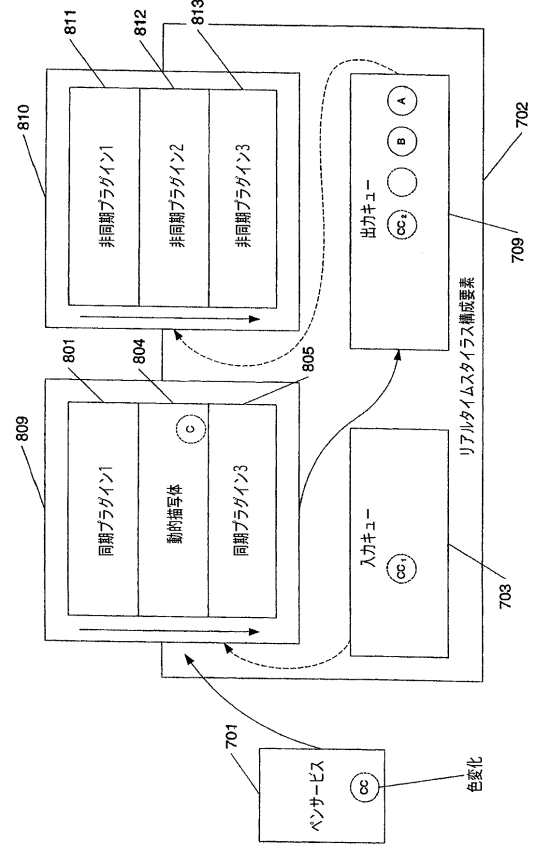
【图 6】



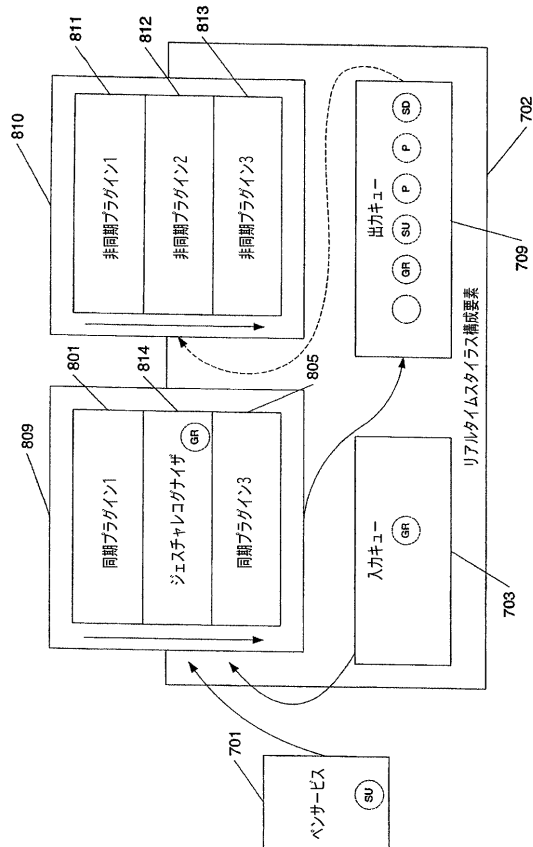
【図 7】



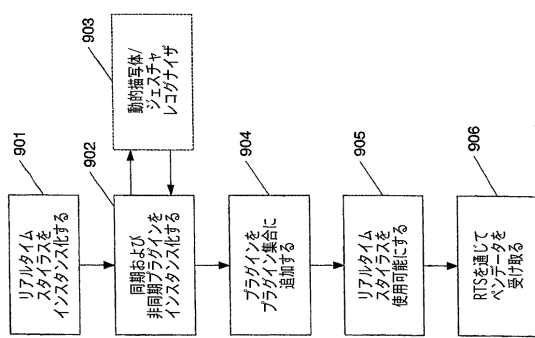
【図 8 A】



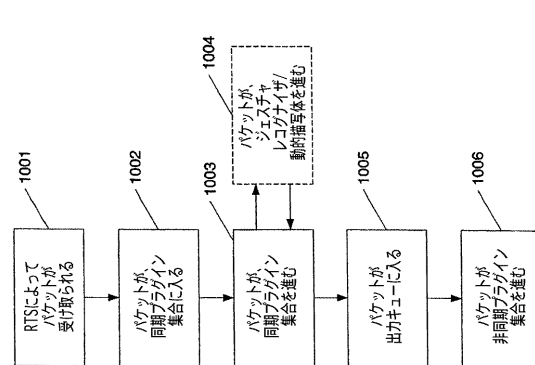
【図 8 B】



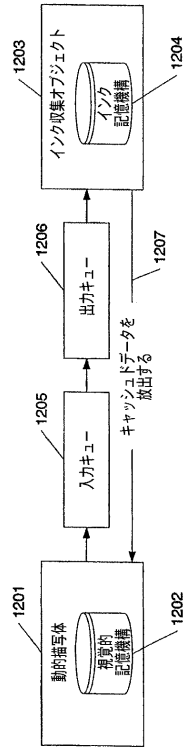
【図 9】



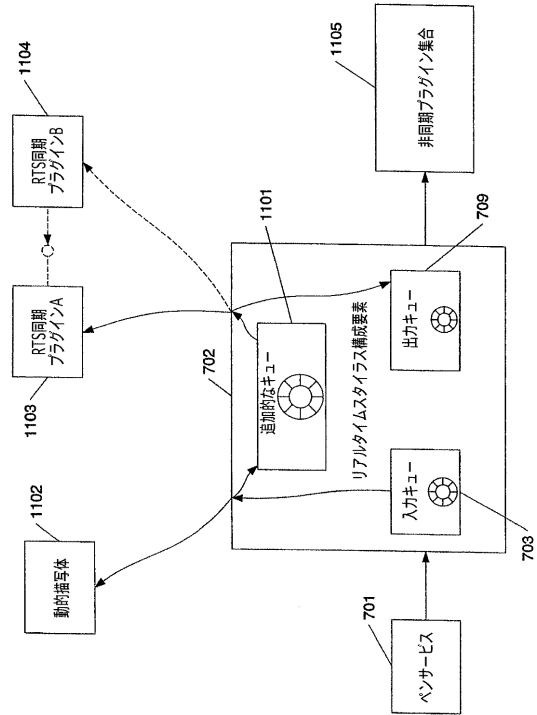
【図 10】



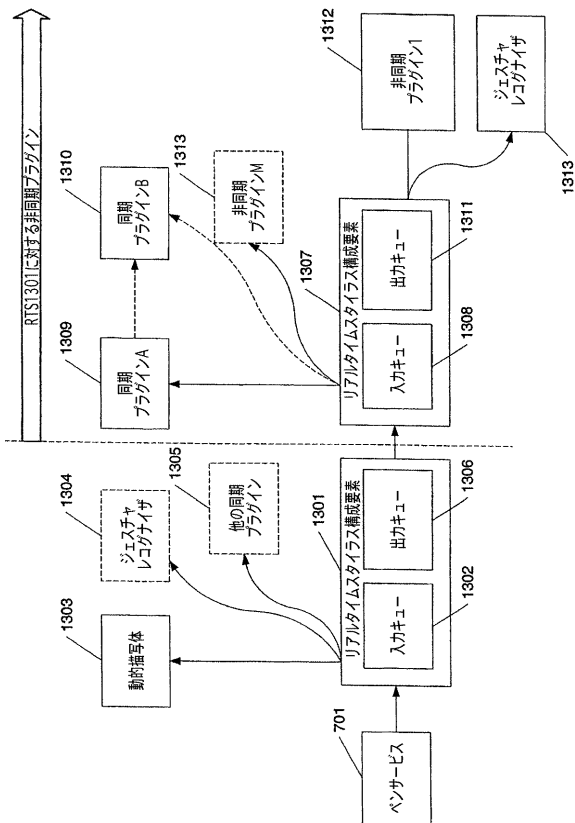
【図 1 1】



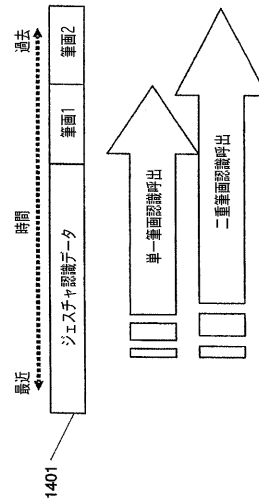
【図 1 2】



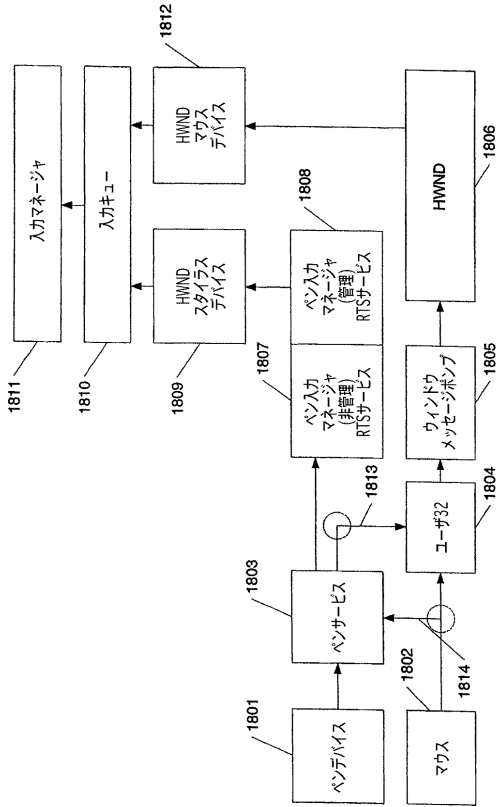
【図 1 3】



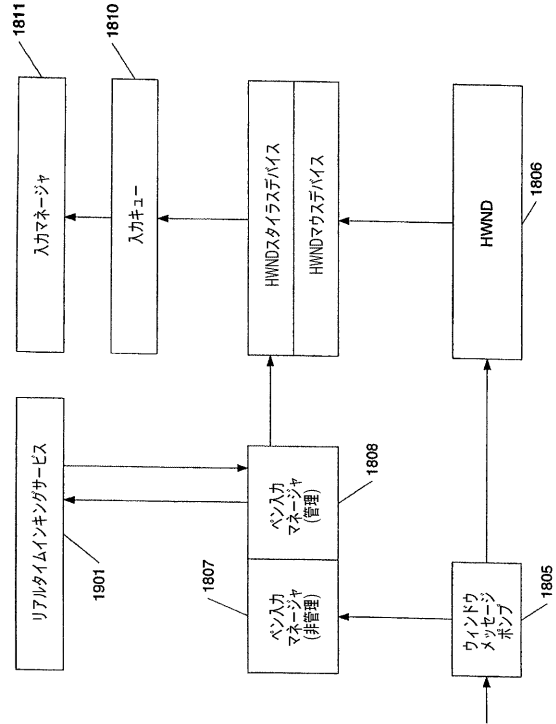
【図 1 4】



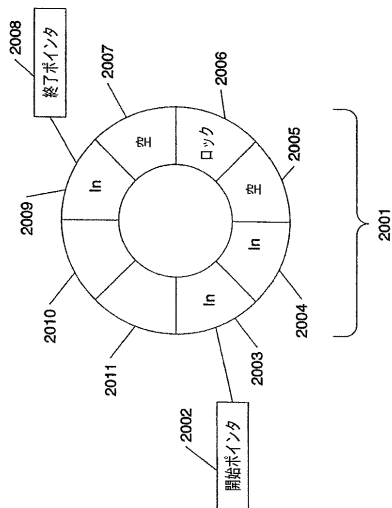
【図 18】



【図 19】



【図 20】



フロントページの続き

- (72)発明者 アレキサンダー ジェイ. コルミコフーゾトフ
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイ
クロソフト コーポレーション内
- (72)発明者 アリン ジェイ. ゴールドバーグ
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 ブリジット クランツ
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 キリル フェルドマン
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 マノ ケイ. ビスワス
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 ルドルフ バラツ
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 ピチャイア シンバガラクシュミ
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内

審査官 ▲吉▼田 耕一

- (56)参考文献 特開2003-141100 (JP, A)
米国特許出願公開第2003/0025675 (US, A1)
特開2003-216324 (JP, A)
特開2003-288607 (JP, A)
特開2003-228718 (JP, A)
特開平04-326488 (JP, A)

(58)調査した分野(Int.Cl., DB名)

G06F 3/041
G06T 11/80
G06T 13/00
G06T 13/80