ワークスペースの構築を可能とするマルチウィンドウシステムの提案

A proposal of a multi-window system that enables managing workspaces

柴田 博仁[*1]
Hirohito Shibata

大村 賢悟[*1]
Kengo Omura

[*1] 富士ゼロックス株式会社 研究技術開発本部
Research & Technology Group, Fuji Xerox Co., Ltd.

When we perform tasks in computers, we usually refer multiple documents and switch multiple software applications. This paper proposes an extended multi-window system (*Docking Window Framework*) that aims to reduce the overload of window operations during the tasks. The system enables docking windows to create workspaces for the tasks and supports operating multiple windows at the same time.

1. はじめに

コンピュータ上で作業を行う場合，1 つのドキュメントまたは 1 つのアプリケーションで作業が遂行されることは稀である．通常われわれは，複数のドキュメント，複数のアプリケーションを同時に起動し，並置し，切り替えながら作業を行う．論文を書く場合を例にとると，エディタやワードプロセッサの他に，辞書，自分がこれまでに書いた論文，他人の論文，図を作成するためのドローツールなどが必要となる．そして，これらに対応する複数のウィンドウを作業しやすい位置に配置し，各々に独自のワークスペースを構築して作業を行う [Henderson 86]．さらには，このような論文を書いたり，調べ物をしたり，ニュースやメールをチェックしたりというさまざまなタスクを並列的に作業し，これら複数のタスクを切り替えながら業務を行っているのが現状である [Mark 05]．

このような業務を現状のウィンドウシステムの枠組みで行なおうとすると，次のような問題が生じる．第 1 に，タスクごとにワークスペースを構築し，これらを切り替えながら業務を行うことができない (問題 1)．結果として，タスクの切り替えが面倒であり，どのようなタスクがどれくらいあるのかも視覚化されない．第 2 に，ウィンドウの数が増えるとウィンドウ間の重なりが生じ，ウィンドウの切り替え，移動，サイズ変更といったウィンドウ操作が増える (問題 2)．第 3 に，ワークスペースを構築するにあたり，必要なアプリケーションをいくつも起動し，必要なドキュメントを開いて各々のウィンドウを望ましい位置に配置しなくてはならない (問題 3)．

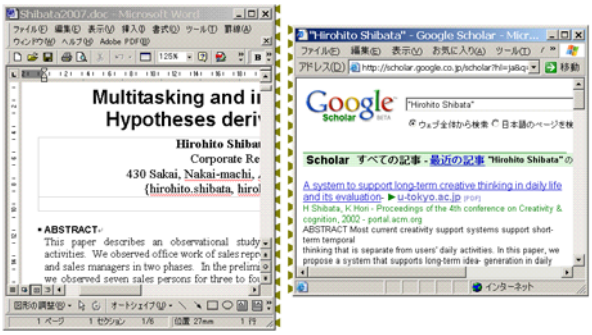
本稿では，これら問題点の解決を目指し，現状のウィンドウシステムの発展と位置づけられる枠組み *Docking Window Framework (DWF)* を提案し，その試作システムを紹介する．

2. 関連研究とアプローチ

ワークスペースの構築と切り替えに関する問題は古くから指摘されてきた．Rooms [Henderson 86] は，この問題に対する先駆的研究として有名であり，部屋メタファーを使って部屋を移動することでワークスペースの切り替えが可能である．これに対して Task Gallery [Robertson 00] では，画廊メタファーを利用して 3D 環境としてシステムを構築している．GroupBar [Smith 03] は Windows のタスクバーの拡張として実現されており，ウィンドウに対応するタスクバー上のアイコンをグループ化することが可能である．これらはどれも先の問題 1 を解決するものであり，ワークスペースの構築方法，待機中のタスク (ワークスペース) の見せ方に違いがある．

連絡先：柴田 博仁，富士ゼロックス株式会社，神奈川県足柄上郡中井町境 430，hirohito.shibata@fujixerox.co.jp

Before



After

図 1 ドッキング前後でのウィンドウの振る舞い

これに対して Elastic Windows [Kandogan 97] では，問題 1 の解決に加えて，問題 2 の解決策も提供する．このシステムは，ウィンドウを入れ子の階層で管理することを特徴とし，各階層において，ウィンドウ操作のコストを低減させるためウィンドウをタイル状に並べている．しかし，Elastic Windows 上で動作するアプリケーションはそれ専用に作ったものに限定され，既存のアプリケーションを Elastic Windows 上で動作させることができないという問題がある．

本研究では，既存アプリケーションの利用を前提に，前節で述べた 3 つの問題を全て解決するシステムを提案する．さらには，ワークスペースの構築と切り替えに対する簡単かつ直観的なユーザインタフェースを提供することを目指す．

3. システム

DWF は次の特徴をもつ.

- ウィンドウをドッキングするというユーザインタフェース
- 複数ウィンドウに対する一括操作
- タイルレイアウトの利用
- タスク状態の再現

(1) ウィンドウをドッキングするというユーザインタフェース

ワークスペースを構築するための簡単かつ直観的なユーザインタフェースとして, DWF ではウィンドウをパズルのピースをはめ込むようにドッキングさせてウィンドウ同士をつなぎ合わせる枠組みを提供する.

図 1 にドッキングの前後の振る舞いを示す. 図 1 上がドッキング前の状態である. ウィンドウを他のウィンドウの近くにドラッグするとウィンドウの両端にギザギザの「のりしろ」が表示される. これは, 両者のウィンドウがドッキング可能であることを示すものであり, ドッキングの際にどの位置とどの位置がつなぎ合わされるのかを示すものである. 図 1 上の状態でドラッグしているウィンドウをドロップすることで, ウィンドウのドッキングが行われる. 図 1 下がその結果を示すものであり, ドッキングされたウィンドウ同士はワークスペースを構成し, あたかも 1 つのウィンドウであるかのように振る舞う. ドッキングではウィンドウの位置とサイズが変化するため, その過程がわかりやすくなるようアニメーション表示を行う. また, ドッキングを行うと, これまでタスクバー上で 2 つに分かれていたアイコンが 1 つに表示される. これにより, タスクバー上でのアイコンの氾濫を防ぐことができる. なお, ドッキング操作で結合したウィンドウについては, タイトルバー上の対応するボタンをクリックすることで, 再び分離することが可能である.

2 節で紹介した従来のシステムはどれも, どのようなワークスペースがあり, そこにどのウィンドウが属すのかを何らかの手段により明示的に指定する必要があった. 本提案では, ウィンドウをドッキングさせることで 1 つのウィンドウのように振舞わせ, それをワークスペースと見なすという枠組みを提供している. これにより, ワークスペースの構築を宣言することなしに, ウィンドウを並べるという行為をそのままワークスペースの構築へとつなげることができる.

(2) 複数ウィンドウに対する一括操作

ワークスペース内のウィンドウに対しては一括で操作できることが望ましい. 操作回数が減るだけなく, 複数のウィンドウからなるワークスペースがあたかも 1 つのウィンドウであることをユーザに強く意識させることにつながる. DWF では, 結合された複数ウィンドウに対する次のような一括操作が可能である.

前面化 ワークスペース内のいずれかのウィンドウをクリック, あるいはタスクバーでの対応するアイコンをクリックすることで, ワークスペース内のすべてのウィンドウが同時に前面化される.

移動 ワークスペース内いずれかのタイトルバーをドラッグすると, ワークスペース内のすべてのウィンドウが同時に移動する.

サイズ変更 ワークスペース内のウィンドウをサイズ変更すると, それに応じて他のウィンドウのサイズも変化する. ワークスペースの長方形が維持されるよう, ワークスペース全体のサイズが変化することもある.

極小化/極大化 これは, ワークスペース内の全てのウィンドウ内容が見える状態にありながら, それでも指定したウィンドウをできるだけ小さく, あるいはできるだけ大きく表示する機能である. これは, ウィンドウ内容が全く見えないアイコンだけの存在にしたり, ウィンドウを全画面表示にする従来の最小化/最大化とは振る舞いが異なる. ワークスペース内のウィンドウに対して極小化/極大化の操作を施すことで, それに応じて他のウィンドウも位置調整を行う. この変化の過程はアニメーション表示される.

クローズ ワークスペースをクローズすることで, その中の全てのウィンドウを同時に閉じることが可能である.

(3) タイルレイアウトの利用

DWF ではスクリーン領域を有効に活用するため, ウィンドウの重なりがなく, 無駄なエリアが生じないタイルレイアウト方式を利用している. 各ウィンドウのサイズ変更に伴って他のウィンドウの位置やサイズを調整する仕方については, Elastic Windows [Kandogan 97] で採用されているものと基本的に同じである. これにより問題 2 の軽減が可能である.

(4) タスク状態の再現

DWF では, ワークスペースの状態 (ウィンドウの種類, 位置, サイズ, ウィンドウで開いているドキュメント) をファイルに保存し, あとで再現することが可能である. これにより問題 3 に対処する.

4. 実現方式

DWF の実現においては, DWF マネージャと呼ばれる常駐型アプリケーションが各ウィンドウの振る舞いを監視し, 各ウィンドウの位置やサイズなどの表示状態を制御する.

ウィンドウで開いているドキュメントも含めてワークスペースの再現を可能にするには, 各ウィンドウでどのドキュメントを開いているのかを把握する必要がある. しかし, 現状の Windows のアーキテクチャでは, 各ウィンドウでどのファイルを開いているのか外部のアプリケーションから知るための一般的な手段はない [Robertson 00]. そこで, DWF 上で動作するアプリケーションでは, 実装時に DWF のライブラリを利用するか, 既存アプリケーションにプラグインを構築して DWF マネージャにメッセージを送る方式を採用している.

5. おわりに

本稿では, ワークスペースの構築を可能にし, ウィンドウ操作負荷を軽減するための方式を提案した. 現在, 試作システムの効果を測定するための実験に着手している. さらに, 本稿の枠組みは, 基本的なウィンドウ部品をつなぎ合わせて新たなアプリケーションを作る枠組みへと発展させることが可能と考え, 現在, その方式を模索中である.

参考文献

[Henderson 86] Henderson, J.D.A., and Card, S.K.: Rooms: The use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface, *ACM Transactions on Graphics*, Vol. 5, No. 3, pp. 211-241, (1986).

[Kandogan 97] Kandogan, E. and Shneiderman, B.: Elastic Windows: Evaluation of multi-window operations, In *Proc. of CHI '97*, pp. 250-257, (1997).

[Mark 05] Mark, G., Gonzalez, V. and Harris, J.: No task left behind? Examining the nature of fragmented work. In *Proc. of CHI '05*, pp. 321-330, (2005).

[Robertson 00] Robertson, G., et al.: The Task Gallery: A 3D window manager, In *Proc. of CHI '00*, pp. 494-501, (2000).

[Smith 03] Smith, G., et al.: GroupBar: The TaskBar evolved, In *Proc. of OZCHI '03*, pp. 41-50, (2003).