クリップパイプ: ウインドウの見たままを流し続ける アプリケーション連携手法

大島 裕樹[†1]　宮下 芳明[†1†2]

本稿ではアプリケーションをウインドウごと取り込んで，別のアプリケーション上に連携する手法を提案する．ウインドウ内の連携したい範囲を指定するだけで，ユーザは手軽にアプリケーション同士の連携ができる．コンピュータ上でアプリケーション同士の連携を行う方法は，ファイル形式の変換やコピー&ペーストなどの手段があるが，どれも連携のための前準備が必要であったり，連携後の編集に支障が生じたりと不便であった．提案手法では，ウインドウの指定範囲の見たままがリアルタイムで連携される．連携のための前準備は軽減され，連携元の操作が即時反映されるため，余計な作業に時間を割く必要がなくなり，ユーザは本来の作業に集中することができる．

Clip Pipe: Application Linkage Method by Streaming the Area of the Window

YUKI OSHIMA [†1]　HOMEI MIYASHITA[†1†2]

We propose the method that capture the application with window and link it to another application. The user only has to specify the area that s/he wants to link in the window, so that s/he can make the linkage between applications easily. There are some ways to link application on your computer such as file format conversion and import, copy & paste etc. All of those means, however, require a preparation for linkage or cause some trouble in editing after the linkage, for that reason s/he felt inconvenience. In our proposed method, the specified area of the window is updated in real-time as it is. Preparation for the linkage will be reduced, and reflect the operation of linkage source immediately, it does not require time for an extra work so that the user can focus on the principal task.

1. はじめに

1.1 アプリケーション連携における問題点

コンピュータ上の作業で，複数のアプリケーションを並行して利用することがある．このとき，一方で作成された素材をもう一方のアプリケーションで取り込み，連携させたい場面は多い．ところが組み合わせによっては情報が読み込めず，連携のためにファイル形式を変換しなければならない場合がある．またクリップボードを使ったコピー&ペースト機能を活用しようにも，思うようにコピーができず，コピー可能な状態にわざわざファイル内容を編集することもある．これらの行為は，本来の作業とは関係のない操作であり，しかも内容を更新するたびに同様の作業を繰り返さなければならない．ユーザはアプリケーション同士の連携を行うたびに，不要な手間を負わされている．

本稿では，すべてのアプリケーションの情報を「ウインドウ」という形式に統一して，内容を見たままに連携する手法を提案する(図 1). 指定した範囲の情報はリアルタイムでやりとりでき，編集内容は即座に反映される．ユーザはウインドウ内の連携したい範囲を指定するだけでよく，内容を変更したい時には，元のアプリケーションを利用して

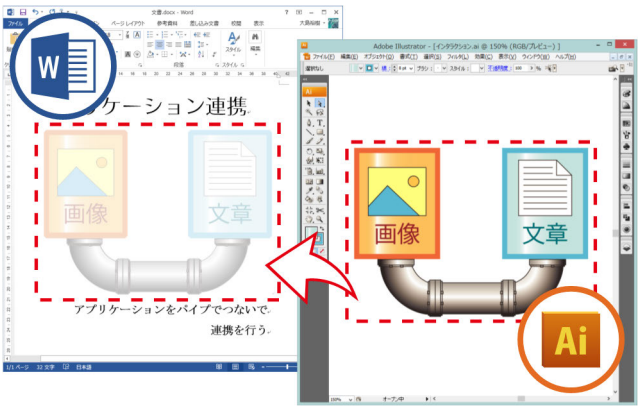

図 1　ウインドウ内容の見たままを連携
Fig. 1　Link the window contents

編集を行う．そのため煩わしいファイル変換や，コピーのための不要な編集作業を行う必要がなくなる．

アプリケーション同士の連携は，これまでファイル形式の変換やコピー&ペーストなどの方法によって実現されてきたが，これらの機能では連携を行った後に不便が生じることがある．ファイルの形式を変換するということは，元のアプリケーションで使用できた独自の機能を切り捨てる

†1 明治大学大学院理工学研究科新領域創造専攻ディジタルコンテンツ系
Program in Digital Contents Studies, Program in Frontier Science and Innovation, Graduate School of Science and Technology, Meiji University
†2 独立行政法人科学技術振興機構, CREST
JST, CREST

ことである. 変換したファイルに変更を加えたい場合には，機能の削がれた状態で編集を行うか，変換前のファイルを再編集し，再びファイルの変換を行う必要がある．万が一変換前のファイルを消してしまえば，最悪の場合最初から作り直しになることもある．コピー&ペーストでは文字と画像が混在した状態などでは，意図した通りにコピーができずに要求と異なったペーストが行われてしまう．このような問題に対して，文字情報を画像へと変換してからコピーすることで問題解決できる場合がある．しかし文字を画像に変換したことで，本来の文字としての編集は不可能になり，誤字を修正する時などに大きな手間が生じる．アプリケーションによっては拡張メタファイル形式で貼り付けを行うことで，画像と文字とが混在した情報を連携できる場合もあるが，貼り付け形式を毎回選択しなければならないため，手間が生じてしまうことに変わりはない．

提案手法では，ウインドウの内容をそのまま取り込み，リアルタイムに連携することで，連携後の編集操作も手軽に行うことができる．見たままを連携するため，文字情報をわざわざ画像に変換する必要もなく，編集は連携元のアプリケーションを利用できるため，連携後に編集内容に制限かかかることもない．連携の前後でコンテンツに対する操作性が変わらず，これまでの連携のための操作を排除することができるため，ユーザは本来の作業に集中することができる．

このリアルタイム連携は，異なるアプリケーション間だけでなく，同一アプリケーション上での連携にも適用することができる．既存の文書編集アプリケーションには，文中の対応する番号などを自動的に同期できる機能が存在するが，提案手法ではこれを文章や画像に対して行うことができる．繰り返し使用している表現を書き換える際に，どれか 1 つを編集するだけですべての内容が更新されるため，同一表現の修正忘れなどを防ぐことができる．

筆者らは提案手法を実現するためのシステム「クリップパイプ」の実装を行った．これまでの情報連携は，保存やコピーといった操作で一度情報を固定化してから行っていた．クリップパイプでは情報をリアルタイムで連携させる新しい手法を実現する．

1.2 既存のリアルタイム連携手法

既存のアプリケーションの中には，部分的にリアルタイムな連携を実現しているものもある．Microsoft Office シリーズ[1]では，アプリケーション上へ別のアプリケーションの一部を配置することができる. 例えば Word 上に Excel 上で作成したグラフを貼り付ける場合，このグラフは貼り付け先でも編集することができる．編集の際には別ウインドウで編集用の Excel 画面を呼び出すことができるため，機能や操作感を損なうことなく連携を行うことができる．

日本語ワープロソフト一太郎[2]では，図の挿入時に統合グラフィックソフト花子[3]を呼び出すことができ，花子の編集機能を用いて文書中の画像に様々な編集を行うことができる．Word や一太郎単体でも，グラフや画像の編集は可能であるが，その内容は非常に簡易的なものに限られる．

これらの機能が実現しているのは，一方向的なリアルタイム連携である. Word 上に連携したグラフに対する編集内容は，元の Excel のファイルには反映されない．連携の対象が 1 つだけなら問題ないが，1 つのグラフを複数の Word 上に連携した場合，それぞれのファイル上での編集内容が反映されず，やはり不便が生じる．一太郎上での花子を利用した編集は，あくまで一太郎上に配置された画像に対する編集であり，連携元となった画像ファイルに対して編集内容が反映されることはない．提案手法が実現するのは，連携元と連携先の両方の編集操作がそのまま反映される，双方向的なリアルタイム連携である．

2. 関連研究

アプリケーション間連携に関する問題については，特にWebアプリケーションに注目した連携手法が数多く提案されている．吉川らはWebサービス環境における，自由なアプリケーションの切り替えが妨げられていることに着目した．その原因として入出力インタフェースの相違を挙げ，同等の機能を持つアプリケーション間でのインタフェース変換を試みた[4]．大澤らは情報過多のWeb環境に対して，既存のWebアプリケーションを自動的に連携する手法を提案している[5]．これはユーザのタスクに応じてそれを自動処理する環境を構築するもので，ユーザの手間を軽減することを目的としている．下坂らはグリッド上アプリケーション連携システムを利用し，Webアプリケーションの連鎖的実行を実現した[6]．評価として15種類のアプリケーション連携を試み，実用的に利用可能という結論に至った．一方でアプリケーション間でのファイル交換には長い時間を要するため，改善が必要であるとも述べている．田辺らはWebアプリケーションとローカルアプリケーションの通信を実現するフレームワークを実装した[7]．Webアプリケーションはローカルアプリケーション以上にユーザの介入が困難な存在であり，ファイル操作などに関して特に厳しい制限を受けているが，田辺らのシステムはこの制限を緩和し，加えてWebブラウザ以外の領域に情報の提示を実現した．

クリップボードに関する研究として，Jeffreyらはコピー&ペーストの制約に着目し，テキスト構造を解析することでユーザに気を利かせたコピー&ペーストを実現している[8]．これにより，これまで複数の入力欄へと別々にペーストする必要があった情報を，一度の操作ですべてペースト可能にするなど，クリップボードの操作性向上を実現している．Florianらはクリップボードの抱える問題を，単一対象にしか作用できない点であると指摘した上で，実世界上にクリップボードに対応した入力装置を用意することで

マルチクリップボード環境の実現を試みた[9]．Florianらはコピー&ペーストの魅力を，操作の簡易さであると述べており，キーボードとは別の入力装置を用意することで，クリップボード本来の操作性を損ねない拡張を目指した．

クリップボードに関する研究で，特に過去状態の復元に着目した研究もある．猪股らはコピー&ペーストを行う際に，参照元の情報を貼り付け先に記録する手法を提案した[10]．コピーした内容を編集後，始めからやり直したくなることがあるが，参照元が分からないために状態復元できないことがある．猪股らの手法では，他のファイルやWebページからコピーされた部分をエディタ上で視覚化し，クリックすることで参照元のファイルへのアクセスができる．Guillaumeらはコピー&ペーストとドラッグ&ドロップの操作の問題点を指摘し，コピーや移動を簡単に行うための手法を提案した[11]．提案の1つとしてGuillaumeらはクリップボードにおける履歴機能の実装を行っている．コピーした内容を利用するクリップボードマネージャは以前から存在していたが，それらはコピーを行うために，マウスでの項目選択などを要求されるため，操作の簡易さを崩してしまっていた．提案手法ではCtrlキーとVキーを一定時間入力し続けることで履歴メニューを表示し，Vキーを押すことで項目を選択，Ctrlキーを離して貼り付ける内容を決定している．この操作手法を取り入れることで，コピー&ペーストの簡素な操作感を保ちつつ，クリップボード上で同時に複数の情報を扱うことに成功している．

ウインドウの一部分を抜き出すという思考を取り入れた研究も存在する．Desneyらは限られたサイズのディスプレイを有効活用するために，ウインドウ内の必要な部分のみを新しいウインドウに抜き出す手法を提案した[12]．抜き出した領域への操作は元の画面に反映され，コピーウインドウ内の状態はリアルタイムで更新される．

3. 提案手法

本稿での提案手法は，ウインドウ内の見たままを取り込み，別のアプリケーション上に連携するものである．編集は元のアプリケーションの操作を利用し，連携したい範囲を指定するだけで手軽にアプリケーションの連携ができる．ここでは，提案手法のリアルタイム連携による利点と問題およびその対策について述べる．

3.1 リアルタイム連携の長所

一般的に，アプリケーション間の連携は情報を固定して行われる．保存は現在編集中の内容をファイルという形に固定する行為であり，コピーはアプリケーション上のある瞬間の情報を，クリップボード上に固定する操作である．筆者が提案するのは，ウインドウ内の現在の状況をそのまま貼り付けることで情報を固定せず，リアルタイムでの連携を実現するものである．

リアルタイム連携の長所は編集の即時反映である．連携元のアプリケーションで操作を行うと同時に連携先の情報も書き換えられるため，一度連携してしまえば内容の修正も簡単に行うことができる(図 2)．編集がリアルタイムに反映されることで，連携先でのレイアウトの試行錯誤なども容易になる．プログラミング言語である Pure Data[13]やProcessing[14]の tweak モードでは，実行中のプログラムに対してソースを編集することが可能であり，即時内容を反映させることができる．描画した図形や出力する音をその場で調整できるため，修正と実行を繰り返すよりも効率良く作業を進めることができる．提案手法によって得られる恩恵も同様で，コンテンツの修正と連携を繰り返すよりも，連携した後にその場で編集を行う方が効率よく作業を行うことができる．

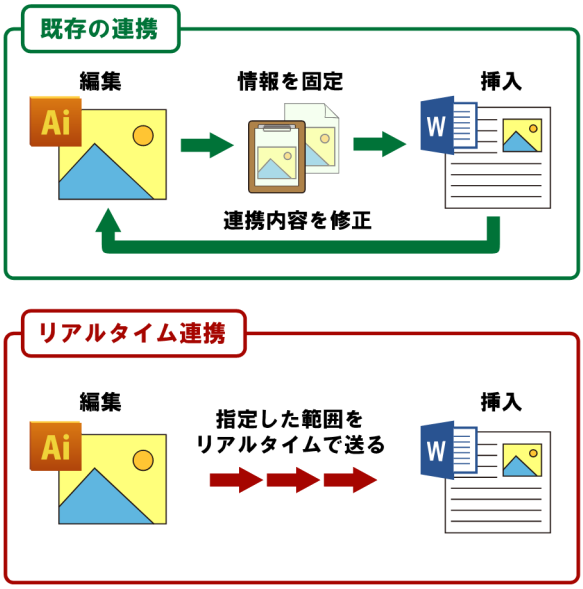

図 2　既存の連携とリアルタイム連携の比較
Fig. 2　Comparison of real-time linkage and existing

文書や画像などの制作において，そのコンテンツが本当に完成したかの判断は難しい．その瞬間には完成させたつもりでも，後になって見なおせば改善点が見つかり，編集を加えたくなることは少なくない．そうして再編集を行うたびに，元ファイルから保存形式の変換やコピーの操作を積み重ねていけば，ユーザの手間は膨大なものになる．リアルタイム連携はそれらの手間を解消し，制作途中においてユーザの手間を大きく削減することができる．

提案手法は Web アプリケーションの連携にも対応している．Web アプリケーションは複数のコンピュータから同時に編集できるものも多く，リアルタイム連携を組み合わせることで，他のユーザが編集した内容を自分のコンピュータ上に連携させることが可能になる．これまでは，他のユーザの作成したコンテンツをファイルとして受け取り，統合を繰り返してきたものが，Web アプリケーション上で編集してもらうだけで連携内容が更新されるようになり，

リアルタイムな協同作業が実現できる．

3.2 リアルタイム連携の問題と対策

情報の保存は，ユーザにとって作業の 1 つの区切りである．それまで常に変化し続けていた状況をファイルという形で固定することで，作業の中継点をつくることができる(図 3)．保存されたファイルの存在は，ユーザの作業が失われることに対する保険であり，万が一アプリケーションが不具合を起こしても，編集内容が残るという安心感をもたらす．さらに複数の段階に分けてバックアップを作成すれば，固定化による安心感はより強固になる．アプリケーション連携の場合も，安心感の提供という意味で情報の固定化は有益である．特に文書中への画像貼り付けなどでは，貼り付けられた情報は不変のものとして扱われ，誤操作などで不意に改変されてしまう危険性は極めて低くなる．加えて，万が一連携元のファイルを削除してしまっても，固定化して連携された情報が連携先に残り，リンク切れのようなリスクを心配する必要もなくなる．

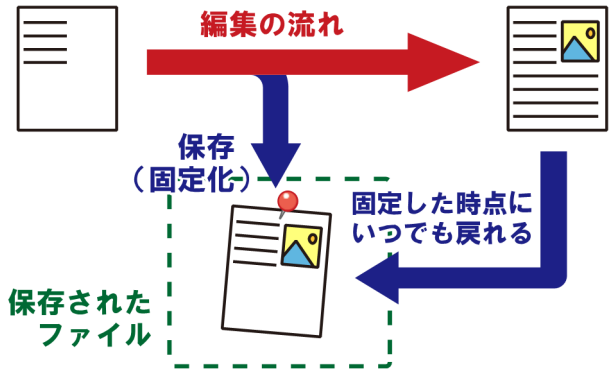

図 3　情報を固定するメリット

Fig. 3　Benefits of fixing the data

これらのメリットはすべて，リアルタイム連携では失われるものである．固定化されていないために意図しない操作をしてしまい，復元のために時間を費やせば，結局ユーザは本来の作業とは関係ない余計な作業に時間を割くことになる．ただ単純にリアルタイムで連携するだけでは，ユーザの手間は省けても，元々得られていたメリットが損なわれてしまう．また，連携内容を常にキャプチャし続けるため，多くの内容を同時に連携した時などは，通常の連携に比べて処理が遅くなってしまう可能性もある．

リアルタイムに編集することで処理が遅くなる問題は，既存のアプリケーション上でも生じている．例えば音声編集において，付加したエフェクトが膨大になり処理が遅くなる場合がある．これはフリーズ機能と呼ばれる，音声情報を一旦固定する機能を用いることで解決できる．固定された音声は一時的に編集が不可能になるが，処理の負荷が軽減され，また，いつでも元の状態に戻すことができる．リアルタイム連携の長所を生かしつつ，既存の連携の長所を損なわないために，内容同期の ON-OFF を簡単に切り替える仕組みを用意する必要がある．

4. システム

ここでは提案手法を実現するシステム「クリップパイプ」について述べる．クリップパイプは，複数のアプリケーションを「パイプ」でつなぎ，情報を流し続けるシステムである．情報を固定して状態を連携するクリップボードに対して，クリップパイプはその機能を拡張し，必要に応じて連携内容を動的に変更することができる．ここでは，クリップパイプを用いたリアルタイム連携を行うための機能と，リアルタイム連携の抱える問題点を補う機能について説明する．

4.1 アプリケーションの指定

ユーザは開いているウインドウの中から，連携するものを選択する．アプリケーションとして開かれているウインドウは勿論，フォルダや Web ブラウザに対しても連携が可能である．クリップパイプでは連携対象にしたアプリケーションのウインドウハンドルを取得し，クライアント領域を画像としてキャプチャする．ユーザは取り込みボタンを押した後に，マウスドラッグで範囲指定をすることで連携先に描画する内容を決定し，アプリケーション内容の連携を実現する．領域指定の方法は矩形選択方式と自由選択方式の 2 種類から選択することができる．ウインドウ上に映っているものであれば，なんでも取り込むことができ，ファイル形式などを意識せずに連携が可能になる．また．連携先での表示位置やサイズは自由に変更することができる．指定した領域は，貼り付けボタンを押した後にマウスドラッグをすることで，描画位置とサイズを調整して貼り付けることができる(図 4)．

図 4　連携までの流れ

Fig. 4　Flow of application linkage

クリップボードは，ショートカットキーのみで素早く利用できるという，操作の簡易さが大きな長所となっていた．クリップパイプでは，Ctrl キーを押しながらマウスドラッグをすることで，矩形での領域選択を行うことができる．また，Shift キーと Ctrl キーを同時に押しながらドラッグすることで，指定領域の貼り付けを行うことができ，ショートカットキーを用いた実用的な利用が可能である．

4.2 連携内容の編集

クリップパイプで連携した内容は，2 種類の方法で編集することができる．1 つは連携先から間接的に編集を行う方法であり，もう 1 つは連携元を直接編集する方法である．

連携先で描画されている内容をクリックすると，クリックした座標がそのまま連携元のアプリケーション内の座標へと変換される．この編集方法は，既に入力されている文章を少しだけ修正する場合などの，簡単な編集に適している．書式設定などの機能は使用できないが，連携先からウインドウを切り替えずに操作ができる．

一方で，連携元のアプリケーションで使用可能な機能を活用して編集したい場合もある．その場合，連携先で描画されている内容をダブルクリックすることで，連携元のアプリケーションを最前面へと呼びだすことができる．アプリケーションの使用には一切の制限はなく，編集内容はそのまま連携先へと反映される(図 5).

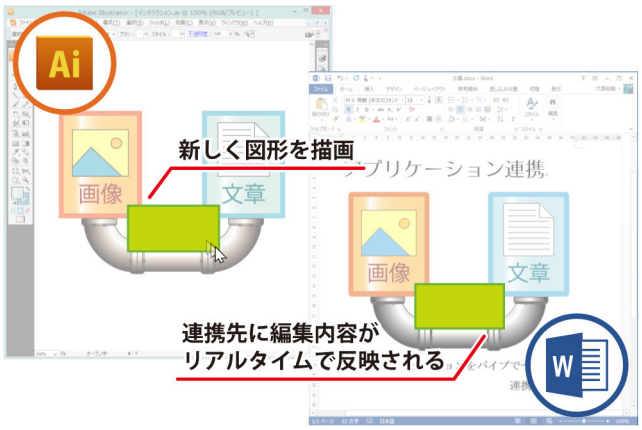

図 5　リアルタイム連携

Fig. 5　Real-time linkage

4.3 同期状態を切り替えるバルブボタン

リアルタイムでの連携では，連携元への意図しない操作などの危険が常に存在する．クリップパイプではこのリスクを軽減するために，連携した内容に対して同期の ON-OFF を即座に切り替える機能を取り入れている．同期状態を OFF にする方法は，連携元のファイルを閉じる方法と状態を切り替えるボタンをクリックする 2 通りの方法がある．今後編集をする予定がないのであればファイルを閉じれば良く，一時的に連携の同期を止めたい場合には切り替えボタンを利用すれば良い．

切り替えボタンは，パイプ間を流れる情報を管理するバルブのような存在である．リアルタイムに状態を同期している時にはバルブが開いた状態であるが，バルブボタンを押すことで，一時的に指定範囲の同期が止まり，連携元への操作も受け付けなくなる(図 6). 編集ができなくなることで，連携した素材への不意の操作を防ぐことができ，ユーザの安心感へとつながる．加えてリアルタイム連携のための画面キャプチャが停止するため，コンピュータへの負荷を軽減することにもつながる．バルブを閉めた後でも，再びバルブボタンを押すことで即座に編集可能な状態へと移行できる．また，ファイルを閉じて同期状態を OFF にした場合は，再びファイルを開くことで連携内容の同期が再開する．

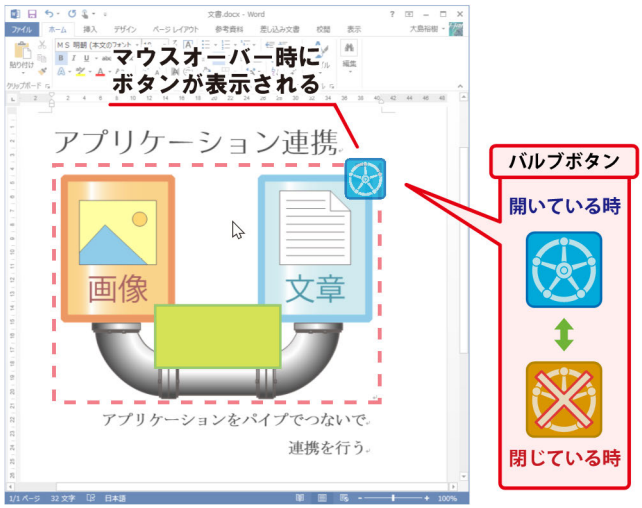

図 6　バルブボタンによる連携状態の変更

Fig. 6　State changes in the valve button

4.4 履歴機能による過去状態の復元

多くのアプリケーションには，ユーザの操作を取り消す Undo 機能が実装されている．そのため小規模なやり直し操作であれば，各アプリケーション上で Undo を実行すれば連携内容を復元することができる．アプリケーションの内容をそのまま連携しているため，連携した内容ごとに Undo 操作を行える．しかし，Undo 機能が利用できない状況，例えば一度ファイルを閉じてしまった場合や，各アプリケーションで実行可能な Undo 回数より前の状態を復元したい場合には，アプリケーション側の操作では要求を満たすことができない．

Undo 機能で求める状態が復元できない場合，バックアップしたファイルを利用して状態を復元する方法がある．クリップパイプでは，編集の節目に連携内容のバックアップを保持し，過去の状態への逆行を支援する．クリップパイプ使用時に作業の節目と判断できるのは，連携元のファイルが閉じられた時，バルブボタンが押された時，連携元で保存操作が行われた時の 3 つである．いずれかの操作が行

われた際には，その瞬間の連携内容がメモリ上に確保され，後から復元することができる(図 7)．ファイルが閉じられた場合の状態保持は，ファイルを保存せずに閉じた場合にも行われ，より多くの編集結果を連携履歴として残す．

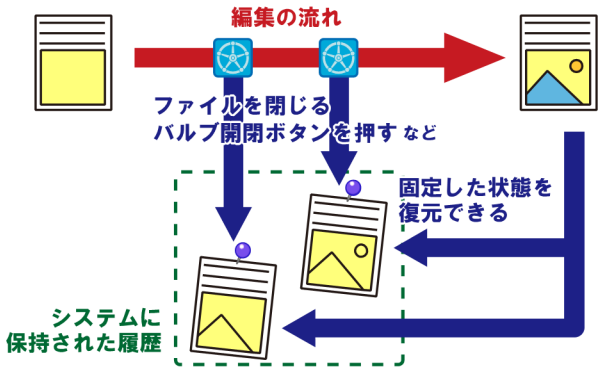

図 7　履歴機能による状態復元

Fig. 7　State restored by the history function

システム側で保持できる内容はその瞬間を記録した画像のみであるため，復元箇所から触接の編集を行うことは想定していないが，過去の状態と現在の編集状況を比較し，より適切なレイアウトを試行錯誤するなど，履歴機能が有効利用できる場面は多い．

5. おわりに

アプリケーション間でのデータのやり取りは，コピー&ペーストの機能により一見実現しているように見える．しかしこれは，クリップボード上で扱うことのできる情報の汎用性の高さによって実現した連携に過ぎない．データがクリップボードの扱う領域を出てしまえば，ユーザは情報連携のための試行錯誤を強いられる．情報をコピー可能な状態にするために，ファイルの内容を変更したり，ファイル形式を指定し直したりする作業は，情報受け渡し時の基本的手段となっているが，本来これらの作業はコンピュータ側で支援されるべきなのではないだろうか．

増え続ける高機能なアプリケーションには，専門的な処理に特化したものも多く，非常に有用である．だからこそ，そこで制作したコンテンツがそのアプリケーション専用となってしまうことは，非常に勿体のないことではないだろうか．作られたコンテンツの利用範囲が広がることは，ユーザにとって大きな利益を生み出し，アプリケーションの価値を大きく高めることができると考える．

現状のシステムでは，ウインドウ間で情報のキャプチャと描画を行っているだけである．提案手法は不要な手間の解消には貢献できるものの，連携においてコンピュータから支援できることはまだ多いと考える．例えば履歴機能に関しては，復元はキャプチャした画像に依存しており，復元地点から再編集ができないのはユーザに取っては物足りない．固定した瞬間の画像だけではなく，編集しているファイルそのものを保持して呼び出すなど，技術的に改善できないか今後も検討していく．

アプリケーション間の連携において，ユーザが求めるものは状況に応じて変化する．様々な状況に対する適切な連携支援への考察を深め，より多くの支援を実現していくことが今後の課題である．

参考文献

1) Microsoft Office. http://office.microsoft.com/ja-jp/
2) JUST SYSTEMS 日本語ワープロソフト 一太郎 http://www.justsystems.com/jp/products/ichitaro/
3) JUST SYSTEMS 統合グラフィックソフト 花子 http://www.justsystems.com/jp/products/hanako/
4) 吉川貴，中川智尋，太田賢，稲村浩．柔軟なインタフェース適応のための Web サービスグループ管理システム．電子情報通信学会技術研究報告, MoMuC，モバイルマルチメディア通信, Vol.104, No.38, pp.53-58, 2004.
5) 大澤哲也，福田直樹，飯島正，山口高平．オントロジーを用いた Web アプリケーション自動連携とその評価．電子情報通信学会技術研究報告, KBSE，知能ソフトウェア工学, Vol.104, No588, pp.49-54, 2005.
6) 下坂久司，廣安知之，三木光範．グリッド上のアプリケーション連携システムの評価．情報処理学会研究報告，ハイパフォーマンスコンピューティング, Vol.2006, No.87, pp.109-114. 2006.
7) 田辺正喜，大圃忠親，新谷虎松．ローカルアプリケーションと連携可能な Web アプリケーションフレームワーク Glimpse Connection の試作．情報処理学会研究報告, ICS，知能と複雑系, Vol.2006, No.110. pp.33-39, 2006.
8) Jeffrey Stylos, Brad A. Myers, Andrew Faulring. Citrine: Providing Intelligent Copy-and-Paste. UIST '04 Proceedings of the 17th annual ACM symposium on User interface software and technology, pp.185-188, 2004.
9) Florian Block, Nicolas Villar, Hans Gellersen. A Malleable Physical Interface for Copying, Pasting, and Organizing Digital Clips. TEI '08 Proceedings of the 2nd international conference on Tangible and embedded interaction, pp.117-120, 2008.
10) 猪股修一，倉本到，辻野嘉宏，水口充．編集後テキストからのコピー元文章参照手法．インタラクション 2010 論文集, 2010.
11) Guillaume Faure, Olivier Chapuis, Nicolas Roussel. Power Tools for Copying and Moving:Useful Stuff for your Desktop. CHI '09 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp.1675-1678, 2009.
12) Desney S. Tan, Brian Meyers, Mary Czerwinski. WinCuts: manipulating arbitrary window regions for more effective use of screen space. CHI '04 Extended Abstracts on Human Factors in Computing Systems, pp. 1525-1528, 2004.
13) Pure Data. http://puredata.info/
14) Processing. http://processing.org/