リモートエディタのSVGへの応用

Application of Remote Editor on SVG

宮里 忍 琉球大学理工学研究科情報工学専攻
河野 真治 琉球大学, 科学技術振興事業団さきがけ研究 21

Shinobu Miyazato Specialty of Information Engineering, University of the Ryukyus.
Shinji Kono Information Engineering University of the Ryukyus, Japan Science and Technology Corporation

リモートエディタは、異なるマシン上のファイルをリモートシェルや NFS を介さずに、エディタ間でリモートエディティング・プロトコル (REP) を用いて直接接続し、分散環境下でテキスト編集を行うエディタである。本稿ではリモートエディタを、XML で二次元のグラフィックスを記述するための言語である SVG(Scalable Vector Graphics) に応用し、Mac OS X の ベクトル系ドローツール Sketch を SVG ビューワ・エディタとして用い、SVG 文書をグラフィックとして編集する手法と、Mac OS X の標準テキストエディタ TextEdit を用い、SVG 文書をテキストとして編集する手法の二つを、サーバ・クライアント方式で同時に実現する方法を提案する。

1 リモートエディティング

我々はこれまで、pico,Emacs そして、Mac OS X の標準テキストエディタ TextEdit, 同じく Mac OS X のベクトル系ドローツール Sketch 上でリモートエディタの実装を行ない、リモートエディタ間のテキスト編集に特化したネットワーク・プロトコルとしてリモートエディティング・プロトコル (REP) [1][2][3][4][5] を提案してきた。サーバおよびクライアントに存在するテキストバッファを操作するコマンド列によって構成された REP は、ネットワークの遅延、切断に強く、巨大なテキストファイルの編集を可能にする。

SOAP や AppleScript は、アプリケーション間を結ぶメソッドを提供しているが、その機能は、特別な機能を呼び出すというものであり、「任意の二つのアプリケーションを接続する」ようなものとは異なる。REP は、「テキストを編集する」という汎用の機能を提供しており、そのテキストの中身に関して REP が知識を持つ必要はない。したがって、編集機能を提供するというインタフェースを提供するアプリケーションがあれば、そのアプリケーションに REP の機能を付加することは容易である。また、意図的に、「テキスト編集による機能の制御」を持つアプリケーションを作成することにより、REP によって相互に接続可能なアプリケーション群を作成することができる。

REP の編集コマンドは open,close,save,read,write, update からなり、一般的なテキストエディタのそれと同等のものである。すなわち、open,close はファイルの開閉、save はテキストバッファのファイルへの書き出し、read はファイルのテキストバッファへの読込み、write はユーザのキー入力といったテキストバッファへの書込に相当する。update コマンドは複数人で同一ファイルを編集する際に、クライアントエディタ間の同期をとるためのコマンドとしてサーバエディタによって使われる。

2 はじめに

SVG(Scalable Vector Graphics)[6] は W3C(World Wide Web Consortium) が提唱している、XML で二次元のグラフィックを記述するための言語である。ベクトルグラフィックは JPEG や PNG といったラスターグラフィックと比べると、拡大縮小してもきれいに描画できるといった特徴がある。また、SVG はその中にラスターグラフィックを埋め込むこともできる。

SVG には XML を用いたテキストデータとして編集される側面と、SVG エディタを用いてグラフィックデータとして編集される側面がある。

本稿では、我々が提案しているリモートエディタを用い、SVG 文書をテキストとグラフィックの両方

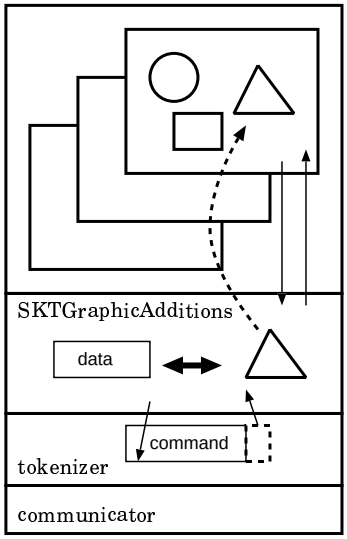

図 1: リモートエディタサーバ Sketch

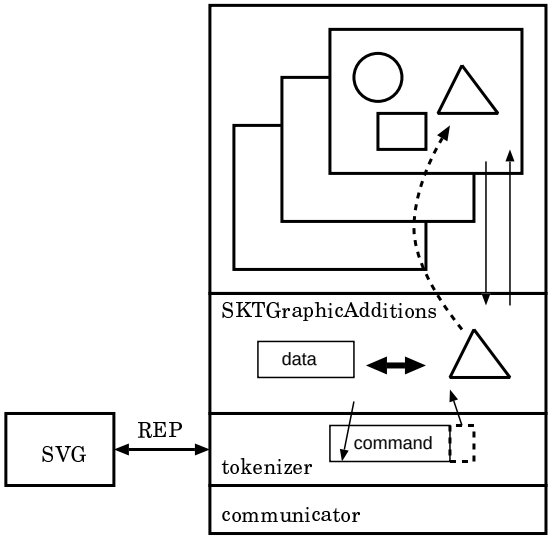

図 2: SVG-Sketch

の側面から双方向に編集することを目的とし、内部データが同じでデータ表現が異なる複数のアプリケーションを接続する場合のリモートエディタの役割について考察する。

3 SVG 文書の編集

SVG をグラフィックとして編集する際に用いるエディタは Sketch とし、SVG をテキストとして編集する際に用いるエディタは TextEdit とする。

3.1 リモートエディタサーバ Sketch

Sketch におけるリモートエディタの設計を図 1 に示す。Sketch はサーバエディタでありクライアントエディタは communicator に接続し、REP コマンドを用いて編集を行う。tokenizer は REP コマンドのエンコード・デコードを行い、SKTGraphicsAdditions は Sketch に依存する部分の実装であり、Sketch データを実際に描画し、Sketch 上でのユーザによる入力を tokenizer 経由でクライアントに渡す。

Sketch があつかう画像フォーマットはベクトル形式の画像ではあるが、SVG 形式ではなく Sketch 独自のフォーマットを用いている。つまり、SVG をサポートしたドローツールではない。ここでは Sketch における編集は SVG 文書として出力される必要がある。また、SVG 文書を Sketch が描画できる必要もある。そこで、Sketch 形式から SVG 文書の生成と、SVG 文書から Sketch 形式を生成する機能を Sketch 上の tokenizer に付加することについて考える。(図 2)

tokenizer に求められるのは以下の二つである。

- Sketch が持つグラフィックデータが更新された際に、SVG 文書側にその更新を伝える
- SVG 文書が更新された際に、Sketch にその更新を伝える

Sketch 形式もベクトルグラフィックであり、円や四角といったオブジェクトが持つ情報は SVG グラフィックのそれが持つ情報とほとんど一致する。よってデータ変換はそれほど難しいものではない。tokenizer から SVG 文書や Sketch データに更新を伝える方法としては、REP をそのまま使用することができる。そうすることで、SVG 文書を管理する部分はあたかもリモートエディタクライアントのように扱うことができる。

3.2 リモートエディタクライアント TextEdit

TextEdit はリモートエディタクライアントであり、Sketch に接続し SVG 文書をテキストとして編集する。リモートエディタクライアントは編集に必要な要素をサーバから読み込む。ここでは画面に表示する分のデータを読み込む。

Sketch に TextEdit を接続し編集を行う際の一般的な流れを以下に示す。

1. (Sketch) 起動

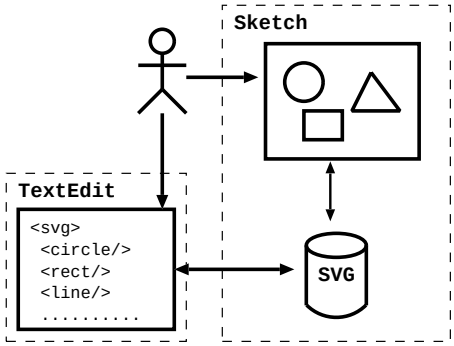

図 3: TextEdit を用いた編集

2. (TextEdit)Sketch に接続要求
3. (Sketch)TextEdit との接続を確立
4. (TextEdit)open コマンドを Sketch に送信
5. (Sketch)open コマンドが示すファイルに対応付けされたバッファを用意する
6. (TextEdit)read コマンドを Sketch に送信
7. (Sketch)read コマンドに対応する write コマンドを TextEdit に送信
8. (TextEdit) 編集が行われた際は、write コマンドを Sketch に送信
9. (Sketch)write コマンドに対応する処理をバッファに対して行い、場合によって結果を TextEdit に返す
10. (TextEdit)close コマンドを Sketch に送信
11. (Sketch) バッファを破棄し、TextEdit との接続を切断

また、Sketch 上で編集が行われた際、つまり SVG 文書が Sketch によって更新された場合には、Sketch から write コマンドが発行され TextEdit に伝えられる。

以上の設計でユーザは SVG 文書をグラフィックデータとして視覚的に捉え編集し、テキストデータとしてスクリプトなどを編集することができる。(図 3)

4 編集の単位と整合性

・編集の単位

REP はテキストをキャラクタ単位ではなく、行単位で編集するように設計されている。キャラクタ単位の編集では、ネットワークの速度が低下しているときは、キー入力の度に遅延が起き、ネットワークが完全に切断されたときには、編集作業は完全に中断されてしまう。行単位の編集では通信が起こる間隔がキャラクタ単位のときよりも長くなるので遅延が生じにくく、ネットワークが切断されても編集中の行に関しては編集が中断されることはない。

・編集の整合性

Sketch 上で例えば円オブジェクトが描かれたとすると、それに対応する SVG 文書が生成されテキストとして書き出される。生成される SVG 文書が SVG の DTD に対して valid であるように設計されていれば、Sketch で描かれた SVG 文書は全て必ず SVG の DTD に対して valid であることが保証される。Sketch は SVG の DTD に valid でない編集を行うことはできないという性質を持つ。しかし、TextEdit で SVG 文書をテキストとして直接編集する際には、ユーザは SVG の DTD に valid でない編集を行うことができる。これは Sketch の性質とは異なるものであり、SVG を編集するということにおいても妥当でない。

この問題は REP の編集履歴を用いたロールバック処理を行うことで回避することができる。リモートエディタにおける編集は REP を用いてサーバを介して行われる。例えばクライアント A が 1 行目の str1 を str2 に write コマンドで書き換えたとすると、サーバで (A,1,str1,str2) といった形で履歴を保持する。クライアントの編集により SVG 文書が well-formed でない間は無制限でクライアントからの編集を受け付け、SVG 文書が well-formed になったときと、クライアントが編集を終了しようとしたときに SVG 文書が valid かどうかを検証する。その結果が偽であった場合は履歴をもとに、編集を無効にする REP コマンドを生成しクライアントと自分自身に送る。これを SVG 文書が valid になるまで繰り返すことでロールバック処理を行うことができる。(図 4)

5 双方向編集

Sketch に Client A と Client B が TextEdit で接続し、同じファイルを、A は 1 から 10 行目まで、B は

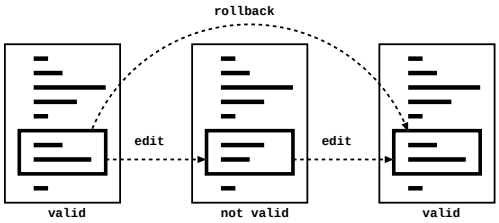

図 4: ロールバック処理

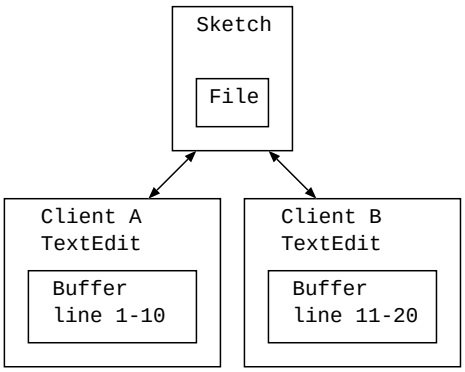

図 5: 複数人で同一ファイルの編集

11 から 20 行目までをバッファリングして編集していたとする。(図 5) ここで、A が 1 行目と 2 行目の間に新しい行を追加すると、B が編集している行はサーバにおいては 1 づつインクリメントされるが、B はそれを知ることができないために、B の編集がサーバで意図していない領域に対して行われてしまう可能性がある。そこで、リモートエディタサーバはクライアントごとの行変換テーブルを持つ。B の行変換テーブルは A による編集結果をうけて、B が指す行番号をサーバでの実際の行番号にマッピングする。例えば、A が 1 行目と 2 行目の間に新しい行を追加したという情報は、B の行変換テーブルに「2 行目以降は 1 インクリメントする」という形で蓄えられ、B からの 2 行目以降に対する read や write に適応される。B からサーバに 11 行目への write が送られて来ると、サーバでは 12 行目への write として処理される。

6 まとめと今後の課題

本稿では、Sketch と TextEdit 二つのリモートエディタを用い、SVG 文書をテキストとグラフィックの両方の側面から双方向に編集する手法を提案した。

SVG には rect,circle,ellipse,line,polyline,polygon といった基本形状があり、これらは Sketch の View を変えずに描画することができる。しかし、SVG において定義されているアニメーションやスクリプト言語を用いたイベント処理などはテキストとして編集することはできるが、Sketch の View では描画することはできない。今後の課題として、SVG をサポートするドローツールもしくはビューワに対してリモートエディタとしての実装を行うことが必要であると考える。

参考文献

[1] リモートエディタの kterm への応用, 情報処理学会第 90 回システムソフトウェアとオペレーティングシステム研究会, 宮里 忍, 河野 真治,2002.

[2] リモートエディティングプロトコルの Mac OS X のエディタへの応用 宮里忍, 河野真治 SWoPP 2001, July, 2001

[3] RemoteEditingProtocol を用いた複数ユーザ編集システム 新垣将史, 河野真治 日本ソフトウェア科学会第 17 回論文修,2000

[4] リモートエディタの実装とその XML への応用 新垣将史, 河野真治 日本ソフトウェア科学会第 16 回論文集,1999,pp293-296

[5] Emacs 上のリモートエディタ 新垣将史, 河野真治 電子情報通信学会技術研究報告,2000

[6] W3C "Scalable Vector Graphics (SVG) 1.1" http://www.w3.org/TR/SVG11/

[7] W3C "Simple Object Access Protocol (SOAP) 1.1" http://www.w3.org/TR/SOAP/

[8] Apple "AppleScript" http://www.apple.com/applescript