Web ページを自己再生産する JavaScript プログラム*

日立製作所 中央研究所
金田　泰*

Web ページ中の JavaScript プログラムは，そのプログラムじたいをふくむページ内容を消去して，あらたにページ内容を生成することができる．JavaScript プログラムはそのプログラムをふくむ正確に同一のページ内容を生成できる．したがって，Web ページはそこにふくまれる JavaScript プログラムによって自己再生産できる．正確な再生産はやくにはたたないが，内容の一部を変化させる “不正確な再生産” は実用目的でつかえる．この方法によって，たとえばページ中のボタンをおすことでアウトライン・モードから詳細表示モードに，またその逆に変化させることができる．この方法は，自己再生産するプログラムを文書がふくむことができるなら，SGML や XML など，HTML 以外の文書にも適用できる．またここではプログラムを再生産せずに Web ページを再生産する方法についてものべる．自己再生産する Web ページは部分的にだが Netscape Navigator において実際に動作する．

1. はじめに

プログラムの自己再生産ないし自己印刷はふるくからある話題である．自己再生産するプログラムはもとのプログラムとおなじ原始プログラムを出力する．たとえば，つぎの C プログラム [Ari 94] (上原稔による) は自己再生産するプログラムのひとつである．

```
char c,*x,*y,*z;main(){x="char c,*x,*y,*z;main(){x=";c='"';y="%s%c%s%c;c='%c';y=%c%s%c;¥
z=%c%s%c%s";z=";printf(y,x,c,x,c,c,c,y,c,c,z,c,z);}";printf(y,x,c,x,c,c,c,y,c,c,z,c,z);}
```

プログラムの自己再生産はたぶん実用的な意味では有用でない．なぜなら，再生産によってなにも変化がおこらないからである．

しかし，文書がプログラムをふくむことができて，かつそれがそのプログラムをふくむ文書を再生産することができるなら，この方法を応用して文書を自己再生産することができる．HTML ページは JavaScript プログラムをふくむことができて，このプログラムの実行によってそのページ内容を抹消し，あらたな内容を生成することができるから，Web ページは再生産することができる．Web ページの自己再生産の概要を図 1 にしめす．プログラムによって生成された JavaScript プログラムが動作するかどうかは JavaScript の仕様には記述されていない [Net 97a]．しかし，自己再生産するページとそれから生成されたプログラムは，すくなくとも Netscape Navigator 3.0 と 4.0 においては，部分的にだが実際に動作する．この正確な文書再生産のしかけについて 2 章で説明する．

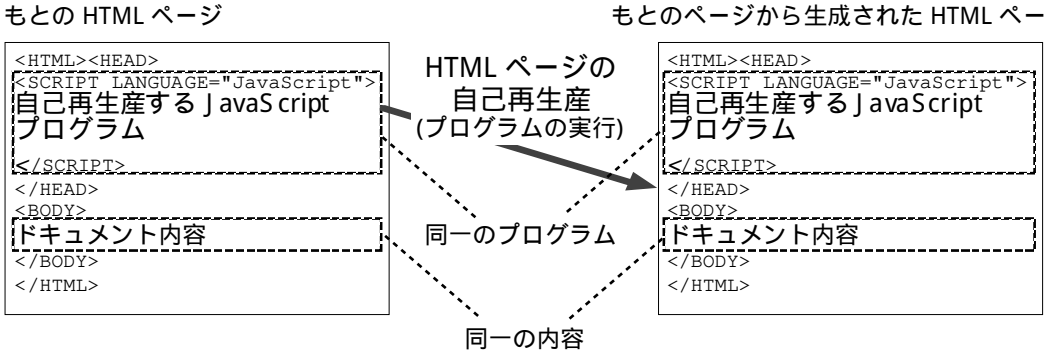

図 1.　Web ページ自己再生産の概要

* E-mail: kanada@crl.hitachi.co.jp.　この論文の英語版は ACM SIGPLAN Notices ’97 年 9 月号に掲載予定．

Web ページの正確な自己再生産は，プログラムの自己再生産と同様に実用的には有用でないとかんがえられる．しかし，“不正確な” 再生産は有用でありうる．なぜなら，それによって文書の変形や変換をシミュレートして実用目的につかうことができるからである．たとえば，不正確な再生産はユーザの意図にしたがって文書のビュー (みかけ) を変化させることを可能にする．自己再生産にもとづく文書の変化のしかけについて 3 章で説明する．実用的なものをふくむいくつかの例を 4 章でしめす．文書を部分的に変化させるために，実際にその部分をかきかえるのでなく Web ページ全体を再生成するのは，もとのページのテキストやすでに生成したテキストは JavaScript によってかきかえられないからである．かきかえられないのはアドホックな理由ではなくて，JavaScript の意味論からくる必然的な理由によっているとかんがえられる．文書のみかけをかえる方法は他にもあるが，自己再生産をつかう方法はいくつかの利点がある．それらについても 4 章でのべる．

JavaScript プログラムが Navigator 上で実際に再生産され動作することは，そのふるまいがおよそ予想どおりであることからわかる．しかし，生成されたプログラムを Navigator をつかって参照することはむずかしい．生成されたプログラムをみるための方法あるいは自己再生産するプログラムをデバグするための方法について 5 章で説明する．プログラムの自己再生産という技巧的な方法をつかわない文書再生産の方法についても 6 章でのべる．

2. 正確な文書自己再生産

正確に自己再生産する HTML ページのもっともかんたんな例を図 2 にしめす．この HTML ページは，表示される内容としてはただひとつのボタンをふくんでいる．このページの Navigator によるみかけを図 3 にしめす．図 2 にふくまれる JavaScript プログラムは，ユーザが図 3 のウィンドウに表示されているボタンをクリックすることによって起動される．それにより，正確におなじ HTML ページが生成され，ウィンドウ上に表示される．生成されたページはもとのページと正確におなじなので，ユーザは再生産をくりかえせる．図 2 にしめしたページはつぎの URL でアクセスすることができる: http://www.st.rim.or.jp/~kanada/reproduction/reproduction-b.html.

JavaScript program

```
<html><head><script LANGUAGE="JavaScript">
function reproduce() { var i; var q = "'"; document.clear();
var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");
var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";
var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);
for (i = 0; i <= 1; i++) {
document.writeln(A[i], '<html><head><script LANGUAGE="JavaScript">', B[i]);
document.writeln(A[i], 'function reproduce() { var i; var q = "'+Q[i]+'"; document.clear();', B[i]);
document.writeln(A[i], 'var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");', B[i]);
document.writeln(A[i], 'var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";', B[i]);
document.writeln(A[i], 'var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);', B[i]);
document.writeln(A[i], 'for (i = 0; i <= 1; i++) {', B[i]);
}; document.writeln(C); for (i = 1; i >= 0; i--) {
document.writeln(A[i], '}} <'+S[i]+'script><'+S[i]+'head><body>', B[i]);
document.writeln(A[i], '<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"><'+S[i]+'form>', B[i]);
document.writeln(A[i], '<'+S[i]+'body><'+S[i]+'html>', B[i]);
}) </script></head><body>
<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"></form>
</body></html>
```

Content (a button)

図 2. 正確に自己再生産する HTML ページ

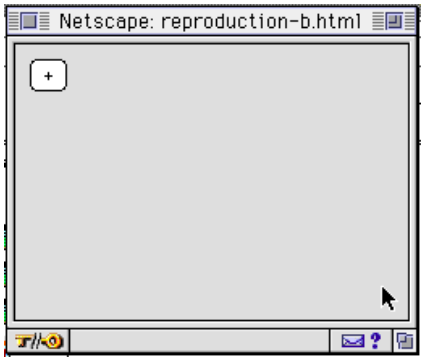

図 3. 正確に自己再生産するページのみかけ [1]

[1] この論文では，HTML ページのみかけをしめすのに Apple Macintosh 上の Netscape Navigator を使用する．

ページの自己再生産は Navigator の 3.0 版やそれ以降の版で実際に動作する．しかし，動作は完全ではない．完全でないのは，ユーザが再生産をくりかえしたあとに「戻る」のメニューをつかって再生成したページにもどってからページ上のボタンをおしても，ボタンが意図したようにははたらかないからである．はたらかないのは，Navigator のバグのために JavaScript プログラムがすでにうしなわれているからである．Microsoft Internet Explorer についていえば，図 2 のプログラムはまったく動作しない．JavaScript プログラムの再生産もしていないようにみえる．

以下，自己再生産の過程を説明する．図 4 においては，図 2 にしめしたのとおなじページを 5 個の部分にわけている．最初の部分にふくまれる "Part F1" とラベルづけられたプログラムは図 4 のプログラム全体を消去し (`document.clear();`)，配列の初期化をおこなう．走行中のプログラムのソースも消去されるが，走行しているプログラムそのもの (たぶん疑似コードに翻訳されている) は保存されている (はずである)．"Part F2" とラベルづけられた 2 番めの部分はプログラムの一部であり，for 文のなかにあるために 2 回くりかえして実行される．最初のくりかえしでは Part 1 とひとしいテキストが生成され，2 回めでは Part 2 とひとしいテキストが生成される．Part C すなわち 3 番めの部分は 1 回だけ実行され，Part C とひとしいテキストが生成される．Part L2 すなわち 4 番めの部分はふたたび 2 回くりかえして実行される．最初のくりかえしでは Part L2 とおなじテキストが生成され，2 回めでは Part L1 とおなじテキストが生成される．したがって，プログラム全体の実行によって，もとの HTML ページとちょうどひとしいテキストが生成される．

```
<html><head><script LANGUAGE="JavaScript">
function reproduce() { var i; var q = "'"; document.clear();
var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");
var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";
var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);
for (i = 0; i <= 1; i++) {
document.writeln(A[i], '<html><head><script LANGUAGE="JavaScript">', B[i]);
document.writeln(A[i], 'function reproduce() { var i; var q = "'+Q[i]+'"; document.clear();', B[i]);
document.writeln(A[i], 'var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");', B[i]);
document.writeln(A[i], 'var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";', B[i]);
document.writeln(A[i], 'var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);', B[i]);
document.writeln(A[i], 'for (i = 0; i <= 1; i++) {', B[i]);
]; document.writeln(C); for (i = 1; i >= 0; i--) {
document.writeln(A[i], '} <'+S[i]+'script><'+S[i]+'head><body>', B[i]);
document.writeln(A[i], '<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"><'+S[i]+'form>', B[i]);
document.writeln(A[i], '<'+S[i]+'body><'+S[i]+'html>', B[i]);
}} </script></head><body>
<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"></form>
</body></html>
```

Part F1
Part F2
Part C
Part L2
Part L1

図 4.　正確な自己再生産をするページの構造

配列 A と B とは Part F1 と F2 をかきわけるためにつかっている．配列 C は Part C をかくのにつかっている．配列 Q と S は一重引用符とスラッシュ文字とをかくためにつかっている．これらの文字は文字列中でエスケープ文字をつけなければならないので，特別の方法が必要になっている．

3. 不正確な自己再生産による文書の部分的な変化

部分的な文書の変形，すなわちもとの文書からみかけが多少ことなる文書を生成することは，図 2 にしめしたページをすこしかきかえることで実現できる．かんたんな例を図 5 にしめす．これは 2 モードのスイッチ (ボタン) をもつページである．このページの内容のおおまかな構造は正確な再生産をするページとひとしい．しかし，いくつかのこまかいちがいがある．このページは "+" というラベルがついたボタンをふくんでいる．ユーザがこのボタンをクリックすると，"–" というラベルがいたボタンをもつページが生成される．さらにユーザがこのあたらしいボタンをクリックすると，最初のページと正確にひとしい，"+" ボタンをもつページが生成される．ユーザはいくらでもボタンをおしつづけ，この過程をくりかえすことができる．これらの 2 状態のみかけは図 6 にしめしたとおりである．

このスイッチングはつぎのようにして実現されている．図 5 において "(1)" とラベルづけられた式の最初の "+" と，"(4)" とラベルづけられた部分の "+" がスイッチの値である．これらは，つぎに生成されるページにおいては "–" になる．このような値のスイッチングは式 (1) によっておこる．スイッチの値が "+" のときにはこの式の値は "–" になり，スイッチの値が "–" のときには式の値は "+" になる．この値が T への代入において T[0] (配列 T の最初の要素) に格納され，その値が "(2)" および "(3)" の部分でつかわれる．これらの部分をふくむ文は各 2 回実行される．再生成されたページに

おける (1)，(2) に対応する部分は，もとのページの部分 (2) から生成される．部分 (2) は正確にもとのページにおけるのとひとしくなるが，部分 (1) におけるスイッチの値は反転される．再生成されたページにおける (3), (4) に対応する部分は，もとのページの部分 (3) から生成される．部分 (3) は正確にもとのページにおけるのとひとしくなるが，部分 (4) におけるスイッチの値は反転される．図 5 と同一内容のページがつぎの URL にある: http://www.st.rim.or.jp/~kanada/reproduction/toggle-b.html．

```
<html><head><script LANGUAGE="JavaScript">
function reproduce() { var i; var q = "'"; document.clear();
var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");
var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";
var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);
var T = new Array("+"=="+"?"-":"+", q+"+T[i]+"+q);
for (i = 0; i <= 1; i++) {
document.writeln(A[i], '<html><head><script LANGUAGE="JavaScript">', B[i]);
document.writeln(A[i], 'function reproduce() { var i; var q = "'+Q[i]+'"; document.clear();', B[i]);
document.writeln(A[i], 'var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");', B[i]);
document.writeln(A[i], 'var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";', B[i]);
document.writeln(A[i], 'var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);', B[i]);
document.writeln(A[i], 'var T = new Array("'+T[i]+'"=="+"?"-":"+", q+"+T[i]+"+q);', B[i]);
document.writeln(A[i], 'for (i = 0; i <= 1; i++) {', B[i]);
}; document.writeln(C); for (i = 1; i >= 0; i--) {
document.writeln(A[i], '}} <'+S[i]+'script><'+S[i]+'head><body>', B[i]);
document.writeln(A[i], '<form><input TYPE="SUBMIT" VALUE="'+T[i]+'" onClick="reproduce()"><'+S[i]+'form>', B[i]);
document.writeln(A[i], '<'+S[i]+'body><'+S[i]+'html>', B[i]);
}} </script></head><body>
<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"></form>
</body></html>
```

(1) (2) (3) (4)

図 5.　不正確な自己再生産: スイッチつきのページ

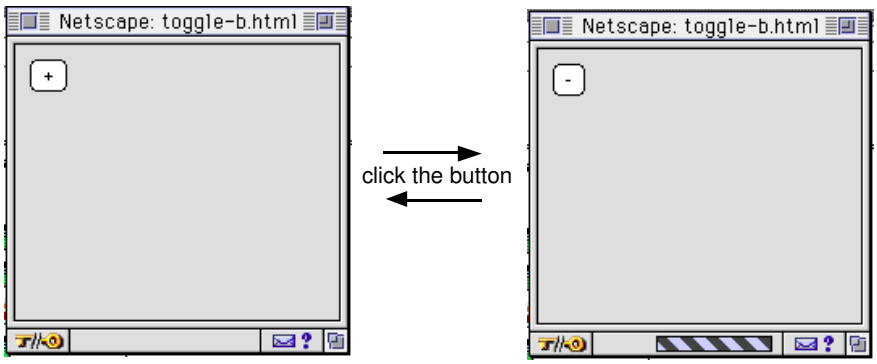

図 6.　スイッチつきのページの 2 状態のみかけ

4. 文書自己再生産の応用

この章ではいくつかの単純な例と，1 個のもうすこし実用性のある例とをしめす．すべての例はつぎの URL からアクセスできる: http://www.st.rim.or.jp/~kanada/reproduction/examples.html．

図 7 にしめす最初の例はカウンタつきのページである．もとのページにおいてはボタンは “0” とラベルづけられている．ユーザがボタンをクリックするたびに，ラベルは “1”, “2”, “3” と増加していく．したがって，このページの内容はつぎつぎに変化していき，祖先とおなじ内容になることはない．

図 8 にしめす 2 番めの例はラジオボタンをシミュレートする．このページには 3 個のボタンがある．そのなかのただ 1 個が “+” とラベルづけられ，他の 2 個は “–” とラベルづけられている．ユーザがこのなかのひとつをクリックすれば，そのボタンのラベルが “+” になり，他は “–” になる．

図 9 にしめす 3 番めの例は，より実用性がある例である．ページ上には 3 個のボタンがあり，それらにテキストが付随している．このページのみかけは図 10 のようになる．ユーザはボタンに付随するテキストを “ひらいたり” (open) “とじたり” (close) することができる．“とじた” 内容はアウトラインであり，“ひらいた” 内容はより詳細な内容である．この機能は Microsoft Windows 95 の Explorer や Apple Macintosh の Finder でフォルダを開閉する機能に似ている．図 9 の例は簡潔にするため実用的な例にはなっていないが，Web ブラウザがページ生成の機能をきちんとサポートすれば，この機能を実用目的でつかうことができる．

```
<html><head><script LANGUAGE="JavaScript">
function reproduce() { var i; var q = "'"; document.clear(); var c = 0;
var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");
var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";
var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);
var D = new Array(c+1, q+"+D[i]+"+q);
for (i = 0; i <= 1; i++) {
document.writeln(A[i], '<html><head><script LANGUAGE="JavaScript">', B[i]);
document.writeln(A[i], 'function reproduce() { var i; var q = "'+Q[i]+'"; document.clear(); var c = '+D[i]+';', B[i]);
document.writeln(A[i], 'var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");', B[i]);
document.writeln(A[i], 'var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";', B[i]);
document.writeln(A[i], 'var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);', B[i]);
document.writeln(A[i], 'var D = new Array(c+1, q+"+D[i]+"+q);', B[i]);
document.writeln(A[i], 'for (i = 0; i <= 1; i++) {', B[i]);
}; document.writeln(C); for (i = 1; i >= 0; i--) {
document.writeln(A[i], '}} <'+S[i]+'script><'+S[i]+'head><body>', B[i]);
document.writeln(A[i], '<form><input TYPE="SUBMIT" VALUE="'+D[i]+'" onClick="reproduce()"><'+S[i]+'form>', B[i]);
document.writeln(A[i], '<'+S[i]+'body><'+S[i]+'html>', B[i]);
}} </script></head><body>
<form><input TYPE="SUBMIT" VALUE="0" onClick="reproduce()"></form>
</body></html>
```

図 7.　不正確な自己再生産: カウンタつきのページ

```
<html><head><script LANGUAGE="JavaScript">
function reproduce(N) { var i; var q = "'"; document.clear();
var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");
var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";
var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);
var T0 = new Array(N==0?"+":"-", q+"+T0[i]+"+q);
var T1 = new Array(N==1?"+":"-", q+"+T1[i]+"+q);
var T2 = new Array(N==2?"+":"-", q+"+T2[i]+"+q);
for (i = 0; i <= 1; i++) {
document.writeln(A[i], '<html><head><script LANGUAGE="JavaScript">', B[i]);
document.writeln(A[i], 'function reproduce(N) { var i; var q = "'+Q[i]+'"; document.clear();', B[i]);
document.writeln(A[i], 'var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");', B[i]);
document.writeln(A[i], 'var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";', B[i]);
document.writeln(A[i], 'var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);', B[i]);
document.writeln(A[i], 'var T0 = new Array(N==0?"+":"-", q+"+T0[i]+"+q);', B[i]);
document.writeln(A[i], 'var T1 = new Array(N==1?"+":"-", q+"+T1[i]+"+q);', B[i]);
document.writeln(A[i], 'var T2 = new Array(N==2?"+":"-", q+"+T2[i]+"+q);', B[i]);
document.writeln(A[i], 'for (i = 0; i <= 1; i++) {', B[i]);
}; document.writeln(C); for (i = 1; i >= 0; i--) {
document.writeln(A[i], '}} <'+S[i]+'script><'+S[i]+'head><body><form>', B[i]);
document.writeln(A[i], '<input TYPE="SUBMIT" VALUE="'+T0[i]+'" onClick="reproduce(0)">', B[i]);
document.writeln(A[i], '<input TYPE="SUBMIT" VALUE="'+T1[i]+'" onClick="reproduce(1)">', B[i]);
document.writeln(A[i], '<input TYPE="SUBMIT" VALUE="'+T2[i]+'" onClick="reproduce(2)">', B[i]);
document.writeln(A[i], '<'+S[i]+'form><'+S[i]+'body><'+S[i]+'html>', B[i]);
}} </script></head><body><form>
<input TYPE="SUBMIT" VALUE="+" onClick="reproduce(0)">
<input TYPE="SUBMIT" VALUE="-" onClick="reproduce(1)">
<input TYPE="SUBMIT" VALUE="-" onClick="reproduce(2)">
</form></body></html>
```

図 8.　不正確な自己再生産: ラジオボタンつきのページ

```
<html><head><script LANGUAGE="JavaScript">var q = "'";
function mkt(N, V) { var j; this.length = V.length;
for (j = 0; j < V.length; j++) { var not = new Array(0);
not["+"] = "-"; not["-"] = "+"; not["closed"] = "open"; not["open"] = "closed";
this[j] = new Array(Math.floor(j/2)==N?not[V[j]]:V[j], q+"+T["+j+"][i]+"+q); }}
function reproduce(N) { var i; document.clear();
var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");
var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";
var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);
var T = new mkt(N, new Array("+", "closed",
"+", "closed", "+", "closed"));
for (i = 0; i <= 1; i++) {
document.writeln(A[i], '<html><head><script LANGUAGE="JavaScript">var q = "'+Q[i]+'";', B[i]);
document.writeln(A[i], 'function mkt(N, V) { var j; this.length = V.length;', B[i]);
document.writeln(A[i], 'for (j = 0; j < V.length; j++) { var not = new Array(0);', B[i]);
document.writeln(A[i], 'not["+"] = "-"; not["-"] = "+"; not["closed"] = "open"; not["open"] = "closed";', B[i]);
document.writeln(A[i], 'this[j] = new Array(Math.floor(j/2)==N?not[V[j]]:V[j], q+"+T["+j+"][i]+"+q); }}', B[i]);
document.writeln(A[i], 'function reproduce(N) { var i; document.clear();', B[i]);
document.writeln(A[i], 'var A = new Array("", "document.writeln(A[i], "+q); var B = new Array("", q+", B[i]);");', B[i]);
document.writeln(A[i], 'var C = "}; document.writeln(C); for (i = 1; i >= 0; i--) {";', B[i]);
document.writeln(A[i], 'var Q = new Array(q, q+"+Q[i]+"+q); var S = new Array("/", q+"+S[i]+"+q);', B[i]);
document.writeln(A[i], 'var T = new mkt(N, new Array("'+T[0][i]+'", "'+T[1][i]+'",', B[i]);
document.writeln(A[i], '"'+T[2][i]+'", "'+T[3][i]+'", "'+T[4][i]+'", "'+T[5][i]+'"));', B[i]);
document.writeln(A[i], 'for (i = 0; i <= 1; i++) {', B[i]);
}; document.writeln(C); for (i = 1; i >= 0; i--) {
document.writeln(A[i], '}} <'+S[i]+'script><'+S[i]+'head><body><pre><form>', B[i]);
document.writeln(A[i], '<input TYPE="SUBMIT" VALUE="'+T[0][i]+'" onClick="reproduce(0)">'+T[1][i]+'', B[i]);
document.writeln(A[i], '<input TYPE="SUBMIT" VALUE="'+T[2][i]+'" onClick="reproduce(1)">'+T[3][i]+'', B[i]);
document.writeln(A[i], '<input TYPE="SUBMIT" VALUE="'+T[4][i]+'" onClick="reproduce(2)">'+T[5][i]+'', B[i]);
document.writeln(A[i], '<'+S[i]+'form><'+S[i]+'body><'+S[i]+'html>', B[i]);
}} </script></head><body><pre><form>
<input TYPE="SUBMIT" VALUE="+" onClick="reproduce(0)">closed
<input TYPE="SUBMIT" VALUE="+" onClick="reproduce(1)">closed
<input TYPE="SUBMIT" VALUE="+" onClick="reproduce(2)">closed
</form></body></html>
```

図 9.　不正確な自己再生産: ページ・ビューの変化

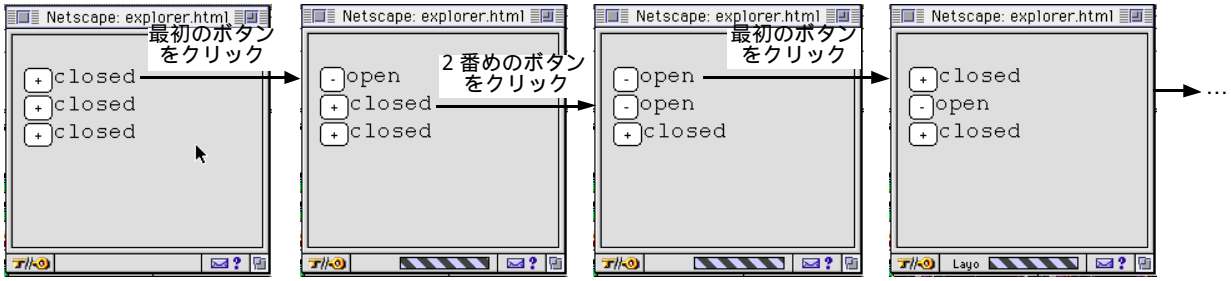

図 10.　ページ・ビューの変化例

文書のみかけを変化させるための他の方法として，下記のようなものがある．しかし，自己再生産をつかう方法はここで説明するようないくつかの利点をもっている．

- CGI スクリプトをつかう方法
  この方法では，ユーザがページ上のボタンをおしたとき common gateway interface (CGI) をつかって要求がサーバにおくられる．結果はサーバからクライアントにおくられて表示される．この方法をつかえば文書のビューを非常に自由に変化させられる．また CGI スクリプトは比較的記述容易である．しかし，要求と結果のページ内容とはユーザがビュー変更を要求するたびにネットワークをつうじておくられるので，この方法は自己再生産をつかう方法にくらべてはるかに応答がおそい．

- Java や ActiveX をつかう方法
  この方法ではボタンは Java アプレット [Sun 97] や ActiveX [Mic 97] コントロールのなかにおく．ユーザがこのボタンをおすと，アプレットやコントロールがビューをかえる．アプレットやコントロールはそのビューのなかのすべてを制御する．すなわち，それが文書ブラウザを実現しなければならない．したがって，おおきなプログラム・ステップが必要になるし，このあたらしいブラウザがくみこみの HTML ブラウザと互換になるようにするのは容易でない．これに対して，自己再生産にもとづく方法では再生産したテキストを表示するのはもとのブラウザである．

- Dynamic HTML を使用する方法
  Dynamic HTML をつかえば Java や ActiveX よりははるかに容易に文書のみかけをかえることができる．Microsoft と Netscape [Net 97b] はそれぞれ互換性のない Dynamic HTML の仕様を提案している．これらのうち，とくに Microsoft の仕様にしたがえばアウトライニングは容易に実現できる．しかし，従来の HTML とはおおきくことなるオブジェクト・モデルをもちこんでいるため妥当性に疑問があるうえ，Netscape などと仕様に関するコンセンサスがえられるのかどうかも現在のところはわからない．かきかえの自由度も自己再生産にもとづく方法よりおとっているとかんがえられる．

5. 生成された文書内容をみる方法

JavaScript プログラムが Navigator 上で実際に再生産され動作することは，そのふるまいがおよそ予想どおりであることからわかる．しかし，生成されたプログラムを Navigator をつかって参照することはむずかしい．生成された HTML ページが表示されているときにページ・ソースのウィンドウをひらくと，そこに表示されるのはもとのページである．プログラムによって生成されたページ p から他のページをひらき，p にもどってからページ・ソースのウィンドウをひらけば，生成されたテキストをみることができる．しかし，Navigator のバグのため，そのテキストは JavaScript プログラムをふくんでいない．したがって，生成されたプログラムをみるためのほかの方法が必要である．とくに，自己再生産するプログラムをデバグするときには必要である．

生成されたプログラムをみるための方法を説明する．`document.writeln` の実引数がふくんでいる `head` タグと `script` タグとを下記のように `body` タグと `pre` タグとで置換すれば，置換後のページはもとのページが再生産するのとほぼおなじプログラムをテキストとしてふくむようになる．

```
<head><script LANGUAGE="JavaScript"> ==> <body><pre>
```

しかし，このページはもはや再生産をしない．以下，プログラムをみるためのより詳細な方法を図 11 をつかって説明する．まず，タグをかきかえた HTML テキストを Navigator 3.0 で表示し，そのページ上にあるボタンをおす．この操作によって生成されたページ上にあるボタンをおし，そこでひらいた JavaScript のエラー・メッセージ・ウィンドウをすべてとじる．ひとつまえのページにもどって，ページ・ソース・ウィンドウをひらく．これでプログラムをテキストとして参照できる．

上記のタグ置換により HTML 構文は不正になる．構文エラーをきらうなら，ほかのタグもかきかえて構文エラーをなくせばよい．しかし，デバグのような一時的な目的には上記の方法で十分である．

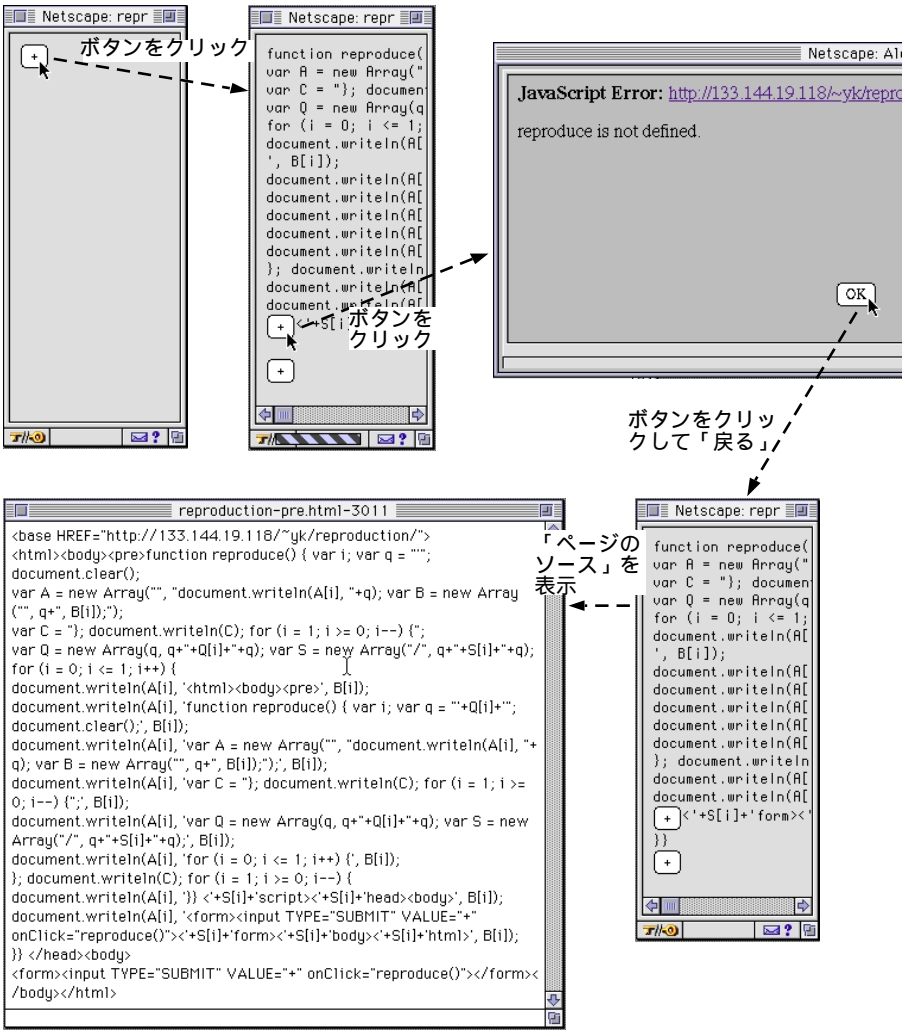

図 11. 生成されたページを Netscape Navigator 3.0 をつかってみる方法

6. 文書再生産のための他の方法

プログラムの自己再生産は技巧的である．プログラム再生産なしに文書の再生産をおこなう方法もあり，この章ではそれについて説明する．Navigator においては，JavaScript プログラムは HTML テキストから分離して，ことなるファイル (URL) におくこともできる．これを利用して正確に自己再生産するページをプログラム再生産なしに実現した HTML ページを図 12 (a) にしめす．また，そこから参照される JavaScript プログラムを図 12 (b) にしめす．後者の URL は reproduction-e.js である．図 12 (a) がふくむ `SRC` 属性つきの `script` タグが外部の JavaScript プログラムを参照している．このプログラムは HTML ページを再生産する．再生産されたページはおなじプログラムを参照する．不正確な再生産も同様にして実現できる．カウンタのあるページは図 13 のように実装できる．不正確な再生産は JavaScript の関数へのパラメタを利用して実現できる．このばあい関数名は `reproduction` であり，パラメタは `c` である．

この方法によって，より技巧的でなく，わかりやすく，したがってより実用的とかんがえられる自己再生産するページが実現される．しかし，この方法はプログラム自己再生産をともなう方法ほどおもしろくはなく，またつぎのような 2 つの不利な点がある．第 1 に，ページの再生産をおこなうたびにおなじプログラムをつかうため，プログラム再生産をともなう方法ほど強力ではない．第 2 に，図 12, 13 の方法は Navigator 3.0 では動作しない．これらのプログラムが動作する唯一のブラウザは Navigator 4.0 とそのプレビュー版である．

```
<html><head><script LANGUAGE="JavaScript" SRC="reproduction-e.js">
</script></head><body>
<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"></form>
</body></html>
```

(a) HTML によるページ内容

```
function reproduce() {
document.clear();
document.writeln('<html><head><script LANGUAGE="JavaScript" SRC="reproduction-e.js">');
document.writeln('</script></head><body>');
document.writeln('<form><input TYPE="SUBMIT" VALUE="+" onClick="reproduce()"></form>');
document.writeln('</body></html>');
document.close();
}
```

(b) JavaScript プログラム (reproduction-e.js)

図 12.　プログラム再生産をともなわない「正確な自己再生産」

```
<html><head><script LANGUAGE="JavaScript" SRC="count-up-e.js"></script>
</head><body>
<form><input TYPE="SUBMIT" VALUE="0" onClick="reproduce(0)"></form>
</body></html>
```

(a) HTML によるページ内容

```
function reproduce(c) {
var c1 = c + 1;
document.clear();
document.writeln('<html><head><script LANGUAGE="JavaScript" SRC="count-up-e.js"></script>');
document.writeln('</head><body>');
document.writeln('<form><input TYPE="SUBMIT" VALUE="'+c1+'" onClick="reproduce('+c1+')"></form>');
document.writeln('</body></html>');
document.close();
}
```

(b) JavaScript プログラム (count-up-e.js)

図 13.　プログラム再生産をともなわない「不正確な再生産」: カウンタつきのページ

7. 結論

もし文書がその文書を再生産するプログラムをふくむことができるなら，この機能を文書のビューをかえるために利用することができる．文書のビューや内容をかえることで，アウトラインから詳細表示への変更のような実用目的を実現できる．この機能は現在のところ HTML だけ，しかも Netscape Navigator だけで動作する．しかし，もし JavaScript をサポートするすべての Web ブラウザが再生産機能をサポートし，ISO などできめられる JavaScript の標準がサポートするなら，文書のビューをかえるなどの目的でこの機能をインターネットじゅうでつかえるようになる．また，この機能は HTML ドキュメントだけでなく，SGML や XML をはじめとするマークアップ言語にも適用できる．

参考文献

[Ari 94] 有沢 誠: プログラミング思考法，実践ソフトウェア開発工学シリーズ，日科技連，1994．

[Mic 97] *active platform*, http://www.microsoft.com/activeplatform/default.asp, Microsoft Corporation.

[Net 97a] *JavaScript 1.1 Language Specification*, http://home.netscape.com/eng/javascript/index.html, Netscape Communications Corp.

[Net 97b] *Netscape Communicator: What's Hot*, http://home.netscape.com/comprod/products/communicator/-beta_features.html, Netscape Communications Corp.

[Sun 97] *What is Java?*, http://java.sun.com/nav/whatis/index.html, Sun Microsystems, Inc.