

Design and Implementation of Web-Based Software Demonstrations

Francis J. Wright
School of Mathematical Sciences
Queen Mary and Westfield College
University of London

<http://www.maths.qmw.ac.uk/~fjw/>

Abstract and transparencies on the web in PDF

29 April 1999

Four demonstrations via <http://cathode.maths.qmw.ac.uk/>
or <http://centaur.maths.qmw.ac.uk/>

Outline of Talk

- ◆ Resources required
- ◆ The user model
- ◆ The transaction model
- ◆ Technical issues
- ◆ A web-based form interaction
- ◆ The `diffgrob2` demonstration:-
 - the web page and its HTML source
 - the perl CGI script
- ◆ Input processing in the `ODESolve` demonstration

Resources required:

Web documentation:

- *WebMaster in a Nutshell*
- *Programming Perl*
- *(CGI Programming on the World Wide Web)*
- *(JavaScript: The Definitive Guide)*

all published by O'Reilly.

Also online documentation on the web.

Web server:

- O'Reilly WebSite
- Apache
- Microsoft Personal Web Server

(all available as free versions).

HTML editor:

- Microsoft FrontPage Express
- Netscape Composer
- Emacs

(all free).

CGI processor: perl (free). (A debugger is also useful!)

Software to demonstrate:

- REDUCE / ODESolve
- Maple / diffgrob2.

Package information:

- user manual
- online help page

as basis for the demonstration.

Some understanding of the package!

The user model

- ◆ Naive, not a professional mathematician, little knowledge of differential equations, no experience of computer algebra. Need pre-defined examples.
- ◆ Also flexible facilities to input problems.
- ◆ Brief self-contained documentation, links to further information.
- ◆ Flexible input format; mathematical output format (optionally).
- ◆ Not a network-based research facility!

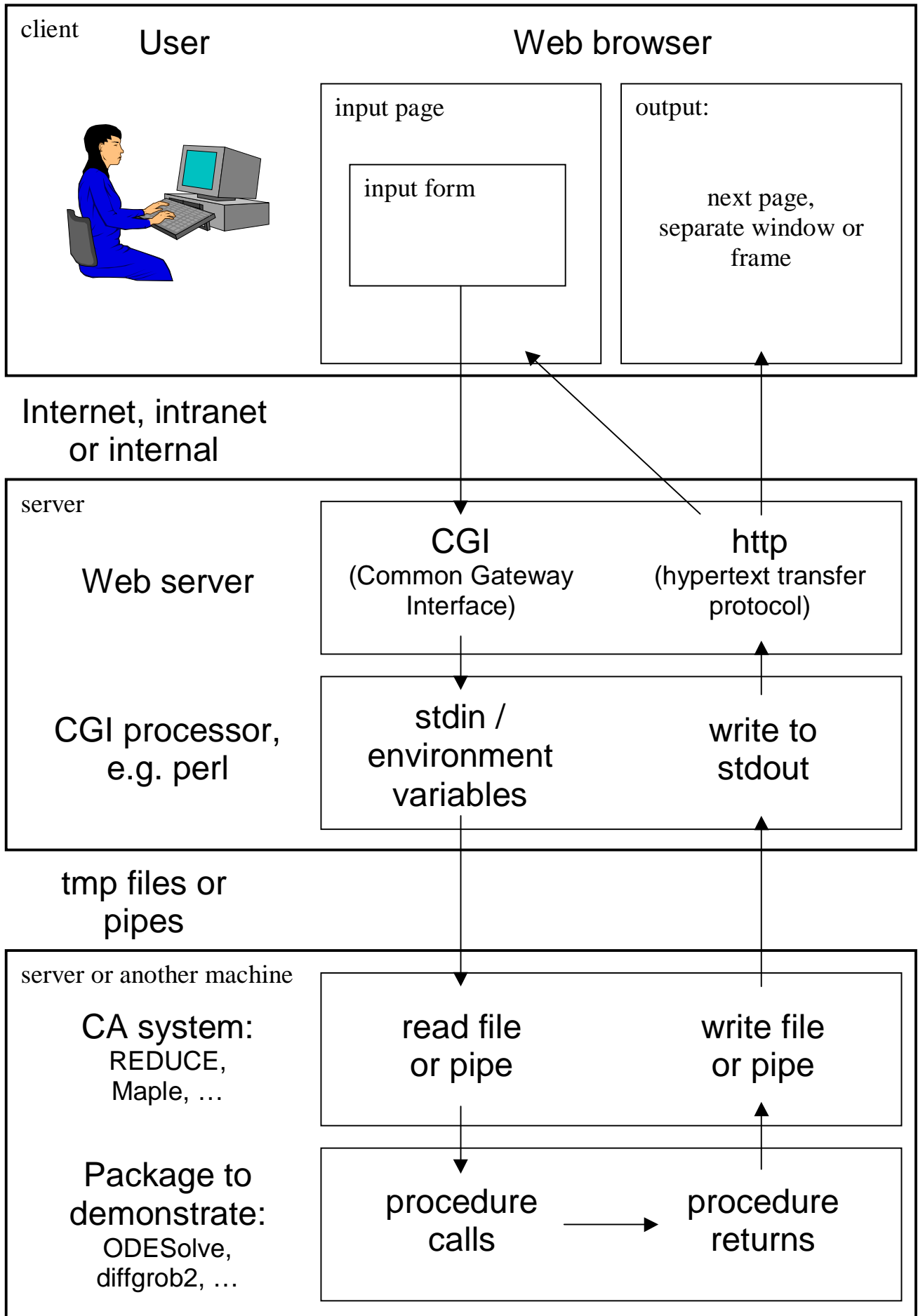
The transaction model

- ◆ Non-interactive, single batch run, no stored state.
- ◆ HTML form.
- ◆ **Input options:** procedure arguments → form fields; options → menus, check boxes or “radio buttons”. Normally one-dimensional input of mathematics.
- ◆ **Output options:** tty prettyprinted, T_EX (via techexplorer), MathML.
- ◆ CGI can pre-process input and post-process output.

Technical issues

- ◆ Live and development web sites – forms *require* http: not file:
- ◆ I/O via pipes or temporary files
- ◆ Security against hacking
- ◆ Resource control
- ◆ Choice of CGI processor language
- ◆ Simultaneous accesses
- ◆ Portability, e.g. Windows – UNIX, Netscape – Microsoft Internet Explorer
- ◆ Input validation – browser (CGI) and/or client (e.g. JavaScript)
- ◆ What if user terminates query?
- ◆ Multiple related demos – code re-use, hidden fields

A web-based form interaction



The diffgrob2 demonstration

Part of the form on the web page ...

The screenshot shows a Netscape browser window with the title "CATHODE DEMONSTRATION of diffgrob2 - Netscape". The address bar contains the URL "http://centaur.maths.qmw.ac.uk/CATHODE/DiffGrob2_demo.html". The main content area of the browser displays a web form with the following sections:

Optionally enter a problem

If no problem is entered here then the example in the [table of operations](#) above will be run instead.

The input differential expressions are always displayed before the output, in the form

```
Input = ...
Output = ...
Factorized Output = ...
```

Enter the variables

Dependent: *DepVars* (e.g. u, v)

Independent: *IndVars* (e.g. x, y, z)

Automatically set up assignments of the form $U := u(x, y, z)$, regardless of the case used for the dependent variables. You can then use whichever side of these assignments you prefer in setting up the rest of the problem. (This facility is illustrated by the examples in the [table of operations](#) above.)

Enter the equations

Enter a comma-separated sequence of Partial Differential Equations (*PDEs*). They must be differential expressions (not equations) in standard Maple syntax, e.g.

```
diff(U, z$2) - y*diff(U, x$2), diff(U, y$2)
```

... and some of the corresponding HTML source

```
<form action="/cgi-bin/DiffGrob2_demo.pl" method="POST"
      target="output">

<hr>

<h2><a name="Problem"><font face="Arial">
Optionally enter a problem</font></a></h2>

<h3><font face="Arial">Enter the variables</font></h3>
<table border="0">
  <tr>
    <td>Dependent: <em><strong>DepVars</strong></em> (e.g.
    <em>u, v</em>) </td>
    <td><input type="text" size="50" name="DepVars"></td>
  </tr>
  <tr>
    <td>Independent: <em><strong>IndVars</strong></em> (e.g.
    <em>x, y, z</em>) </td>
    <td><input type="text" size="50" name="IndVars"></td>
  </tr>
</table>
<p><input type="checkbox" name="Assign"
      value="1">Automatically set up assignments ...

<h3><font face="Arial">Enter the equations</font></h3>
<p>Enter a comma-separated sequence of Partial
Differential Equations ...
<font face="Courier New">diff(U,z$2)-y*diff(U,x$2),
diff(U,y$2)</font></p>
<p><textarea name="PDEs" rows="4" cols="80"
      wrap="VIRTUAL"></textarea>

<hr>

</form>
```

The perl CGI script

```
#!/usr/local/bin/perl

# CGI code for CATHODE DiffGrob2 demo
# Author: Francis J. Wright, Time-stamp: <27 April 1999>

use CGI;
$query = new CGI;

# Die to stdout!
sub Die { print @_; exit; }

$Transcript = $query->param('Transcript');
$TeX = ($query->param('TeX') and not $Transcript);
$Pretty = ($query->param('Pretty') and not $TeX);

if ($TeX) {
    print $query->header('application/x-tex'); # for techexplorer
} else {
    print $query->header('text/plain'); # default is text/html
}

if ($PDEs = $query->param('PDEs')) {
    ($DepVars = $query->param('DepVars')) or
        Die "Please go back to the form and enter the dependent
variables!\n";
    ($IndVars = $query->param('IndVars')) or
        Die "Please go back to the form and enter the independent
variables!\n";
    $PDEs =~ s/, \s*$//;          # kill any trailing comma
}
$Ordering = $query->param('Ordering');
$Assign = $query->param('Assign');

$Operation = $query->param('Operation');
$Operation =~ s/\r//g;          # discard CRs (for Windows)
$Operation =~ /.*/;            # match first line
$Operation = $&;               # first line
unless ($PDEs) {
    $Assign = 1;                # needed for examples
    $DepVars = $';              # subsequent lines
    $DepVars =~ s/^\n//;        # minus leading newline
    $DepVars =~ /.*/;           # match first line
    $DepVars = $&;             # first line
    $IndVars = $';              # subsequent lines
    $IndVars =~ s/^\n//;        # minus leading newline
    $IndVars =~ /.*/;           # match first line
    $IndVars = $&;             # first line
    $PDEs = $';                 # subsequent lines
    $PDEs =~ s/^\n//;          # minus leading newline
}
}
```

```

# Construct the Maple input script:

# Directory for temporary Maple input and output files.
# (Avoid trying to use pipes for now!)
chdir("/tmp") or Die "Can't change to /tmp directory: $!\n";

# Must support multiple simultaneous executions, which needs unique
# filenames. This is a quick hack that should normally work. $^T
# returns time when the program started (seconds since Jan 1, 1970)
$map_in = 'map_in_' . $^T;
$map_out = 'map_out_' . $^T;

open(MAP_IN, "> $map_in") or Die "Can't open map_in: $!\n";
print MAP_IN
    "unprotect(system,ssystem):unassign(system,ssystem):\n";
print MAP_IN "diff2 := `e:/DiffGrob2/diff2`:
libname := diff2,libname:
with(diffgrob2):\n";
print MAP_IN "pretty := 1:\n" if $Pretty;
if ($Assign) {
    $DepVars = lc $DepVars;
    for $DV (split /\s*,\s*/, $DepVars) {
        print MAP_IN uc $DV, " := $DV($IndVars):\n";
    }
}
print MAP_IN "allvars := [[$IndVars], [$DepVars]]:\n";
print MAP_IN "sys := map2dg([$PDEs], [$DepVars], [$IndVars]):\n";
print MAP_IN "'Input' = pdetex(sys);\n";
print MAP_IN "'Output' = $Operation, allvars, $Ordering,
    'Result');\n";
print MAP_IN "if type(Result, {list, set}) then
Result := map(factor, Result) else Result := factor(Result) fi:\n";
print MAP_IN "'`Factorized Output`' = pdetex(Result);\n";
close MAP_IN;

$run_maple = 'E:\MAPLEV4\BIN.WIN\CMAPLE.EXE -f';
# -f : use as filter
# -q : quiet
$run_maple = $run_maple . ' -q' unless $Transcript;
system("$run_maple < $map_in > $map_out");

if ($? != 0) {
    $status = $? >> 8;
    print "Maple exited with non-zero status: $status\n";
    $status = $? & 256;
    print "It may have been killed by signo: $status\n";
    print "Last system call error was $!\n";
}

open(MAP_OUT, "$map_out") or Die "Can't open map_out: $!\n";

print "\\begin{verbatim}\n" if $TeX;

```

```

print "Transcript of " if $Transcript;
$Operation =~ s/\(.*//;
print "Maple demonstration of diffgrob2 procedure
\"$Operation\" with term ordering \"$Ordering\" ...\n\n";

if ($TeX) {
    while (<MAP_OUT>) {
        if (/^\s*Input\s*=/) {
            print "\\end{verbatim}\n\\[\n";
        } elsif (/^\s*Output\s*=/) {
            print "\\]\n\\[\n";
        }
        print;
    }
    print "\\]\n";
}
else {
    print while <MAP_OUT>;
}

close MAP_OUT;

unlink $map_in, $map_out;

```

Input processing in the ODEsolve demonstration

```

# Accept restricted Maple syntax:
$ODE =~ s/\bdiff\b/df/g; # Maple diff -> REDUCE df
$ODE =~ s/(\w+)\$(\d+)/$1,$2/g; # Maple x$n -> REDUCE x, n (within
df)

# Accept ' to denote derivative:
if ($ODE =~ /['`]/) {
    $IndVar or Die "Must specify IndVar to use ' for
derivative!\n";
    $ODE =~ s/'//g; $ODE =~ s/`//g;
    unless ($DepVar) { $ODE =~ /(\w+)'/; $DepVar = $1; }
    $ODE =~ s/(\w+)'+/df($&)/g;
    $ODE =~ s/'/, $IndVar/g;
}

```