BRAUN

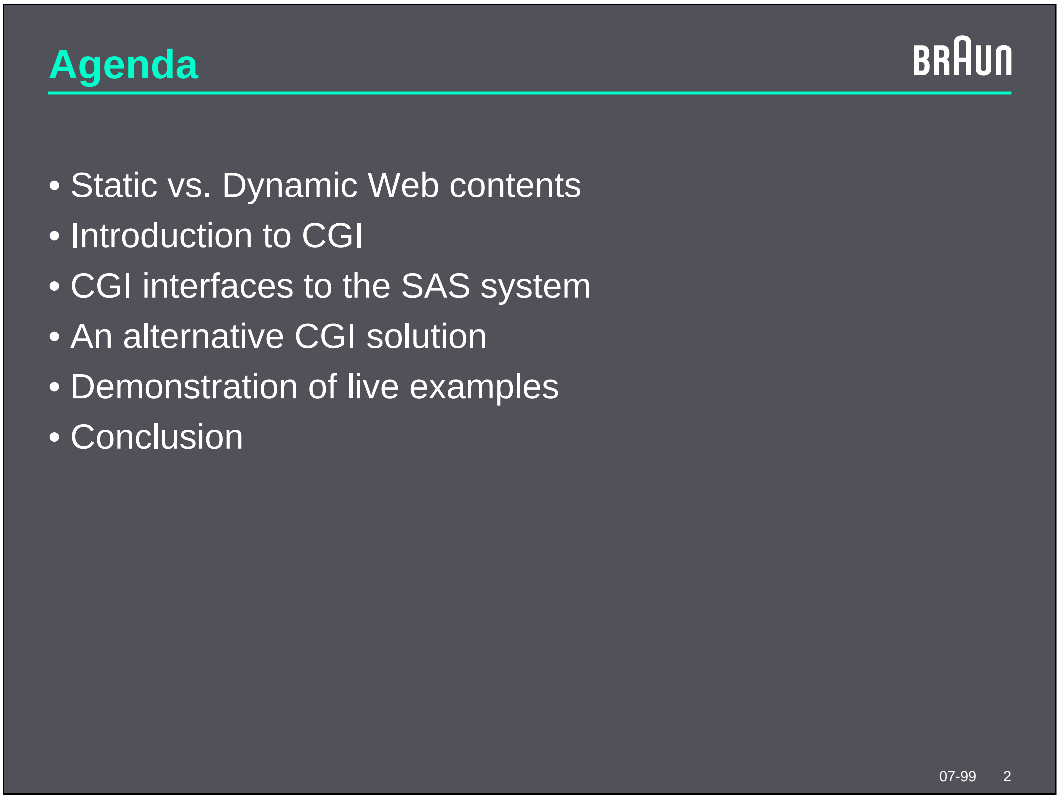
Delivering active WEB contents
with SAS/Base and a single macro

H.Hasselmann Braun Kronberg / Germany

07-99 1

**"Delivering active WEB contents
with SAS/Base and a single macro"**

In December ’95 Braun started to build a Performance Warehouse, covering most performance aspects affecting a heterogeneous system landscape. In addition to fourteen hundred graphical and text-based but ‘static’ reports, each performance database is also accessible as a kind of ‘real-time’ application through a WEB-based query interface. This kind of dynamic reporting was realized by accessing SAS data through a SAS based common gateway interface (CGI) solution running on the WEB server.

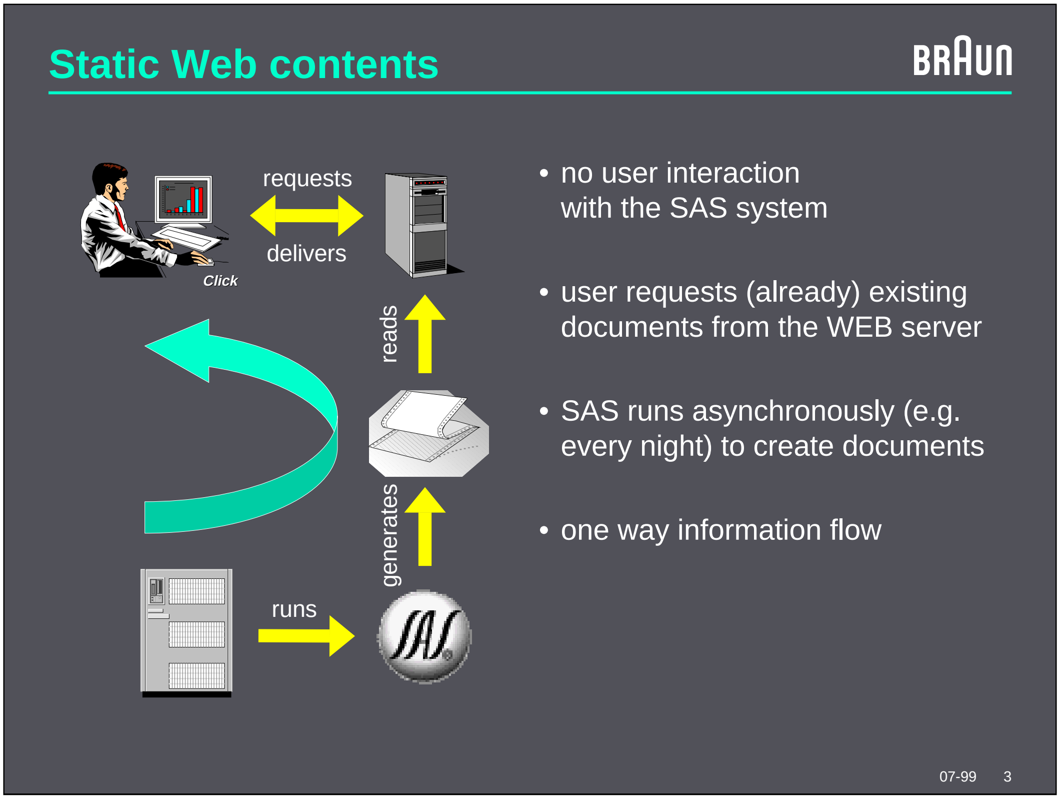


Agenda

- Static vs. Dynamic Web contents
- Introduction to CGI
- CGI interfaces to the SAS system
- An alternative CGI solution
- Demonstration of live examples
- Conclusion

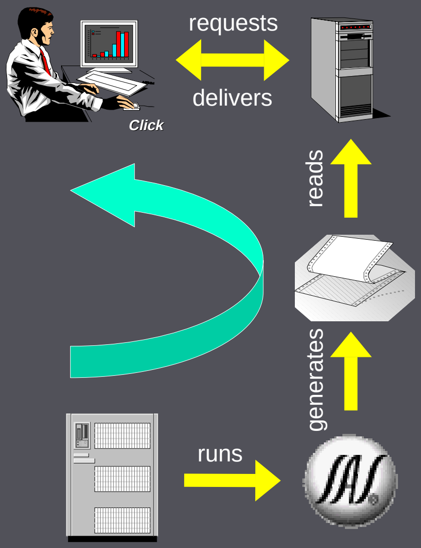
Agenda

- Static vs. Dynamic Web contents
- Introduction to CGI
- CGI interfaces to the SAS system
- An alternative CGI solution
- Demonstration of live examples
- Conclusion



Static Web contents

BRAUN

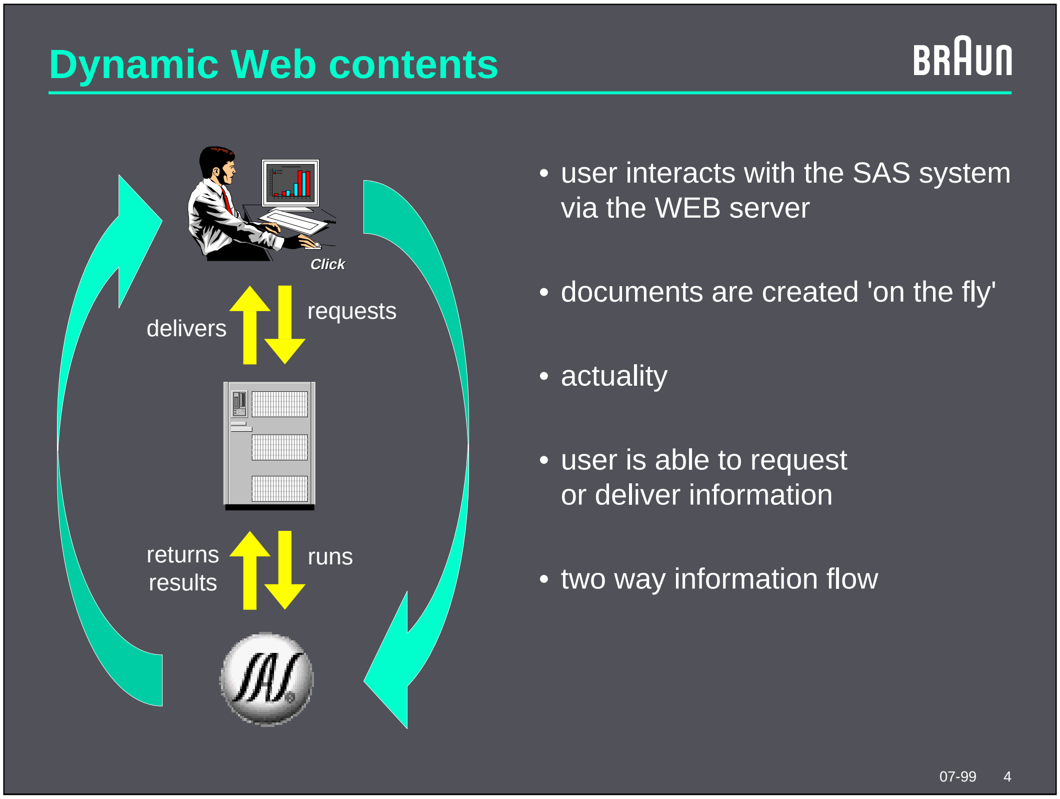


- no user interaction with the SAS system
- user requests (already) existing documents from the WEB server
- SAS runs asynchronously (e.g. every night) to create documents
- one way information flow

Static WEB contents

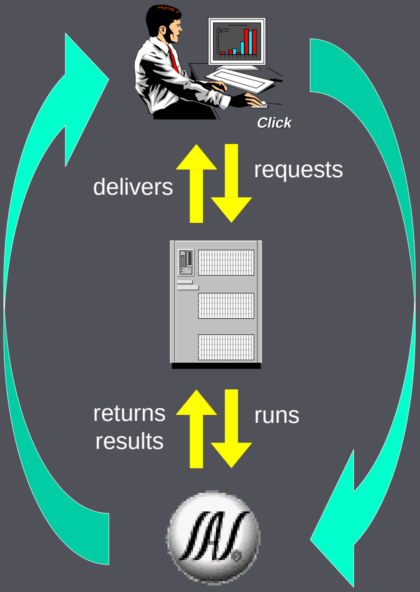
For non experienced CGI users, this and the following chart, explain the difference between static and dynamic WEB contents.

- synchronous vs. asynchronous SAS run
- existing vs. ’on the fly’ created documents
- one way vs.. two way information flow



Dynamic Web contents

BRAUN



- user interacts with the SAS system via the WEB server
- documents are created 'on the fly'
- actuality
- user is able to request or deliver information
- two way information flow

Dynamic WEB contents

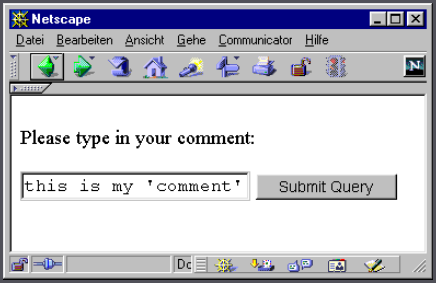
For non experienced CGI users, the previous and this chart, explain the difference between static and dynamic WEB contents.

- Synchronous vs. asynchronous SAS run
- existing vs. ’on the fly’ created documents
- one way vs.. two way information flow

CGI (Common Gate Interface)

BRAUN

```
<html>
 <body>
    <form action="/cgi-bin/test-cgi">
    Please type in your comment:
    <input name=comment>
    <input type=submit>
   </form>
 </body>
</html>
```



Please type in your comment:

this is my 'comment'

HTML form

- prerequisite:
  CGI enabled WEB server
  (see: http://web.golux.com/coar/cgi/)
- creation of the form with HTML
  (see: http://www.w3.org/MarkUp/)
  - the contents of the form are enclosed in <form> tags
- form is displayed with a WEB Browser (e.g.. Netscape)

CGI (Common Gateway Interface) / HTML form

here you see a simple form and how it is discussed how to create it, using HTML. This form will be used in the following three CGI examples.

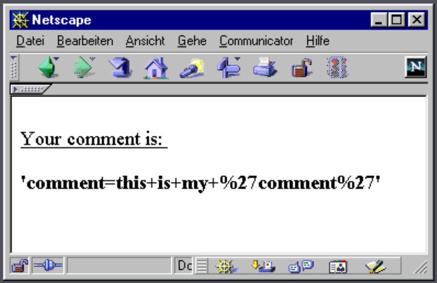
CGI (Common Gate Interface)

BRAUN

```
#!/bin/sh

echo "Content-type: text/html"
echo ""

echo "<u>Your comment is:</u>"
echo "<b> '$QUERY_STRING' "
```



Your comment is:

'comment=this+is+my+%27comment%27'

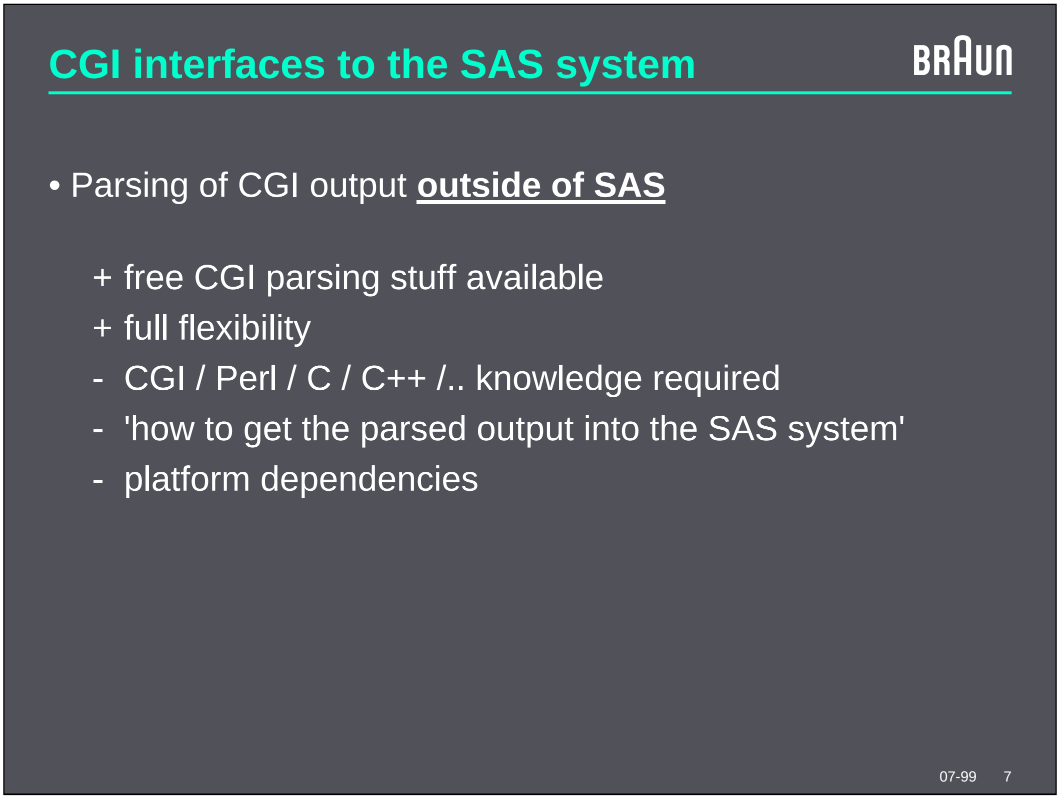
Example 1

- simple, non SAS CGI example
- 5 line shell script which is started by the WEB server
  - mime type specification
  - HTML embedded results
- form results are URL-encoded and have to be parsed (see: http://www.w3.org/Addressing/)

CGI (Common Gateway Interface) / Example 1

Here you see a sample shell script, which demonstrates a simple CGI application. Together with the previously shown HTML form and an CGI enabled WEB server you will have the first very simple CGI application running.

URL encoded results are shown in the WEB browser window.



CGI interfaces to the SAS system

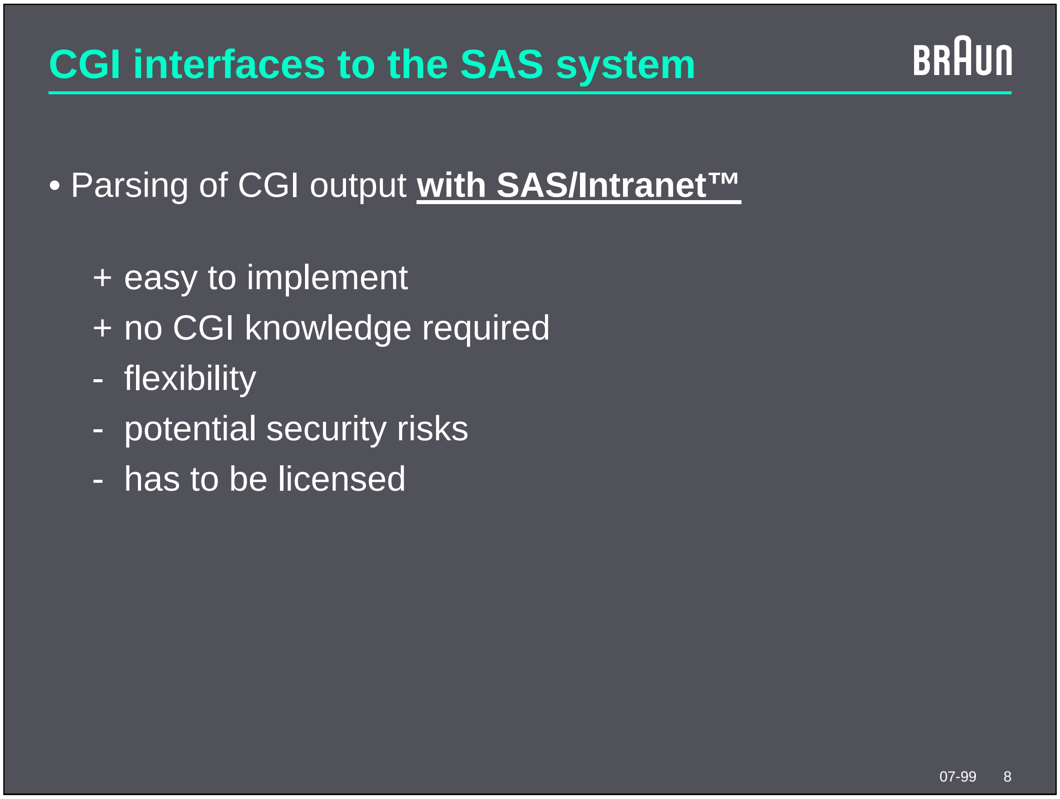
- Parsing of CGI output **<u>outside of SAS</u>**

  + free CGI parsing stuff available
  + full flexibility
  - CGI / Perl / C / C++ /.. knowledge required
  - 'how to get the parsed output into the SAS system'
  - platform dependencies

CGI interfaces to the SAS system

Parsing of CGI output **<u>outside of SAS</u>**

+ free CGI parsing stuff available
+ full flexibility
- CGI / Perl / C / C++ /.. knowledge required
- ’how to get the parsed output into the SAS system’
- platform dependencies



CGI interfaces to the SAS system

- Parsing of CGI output **<u>with SAS/Intranet™</u>**

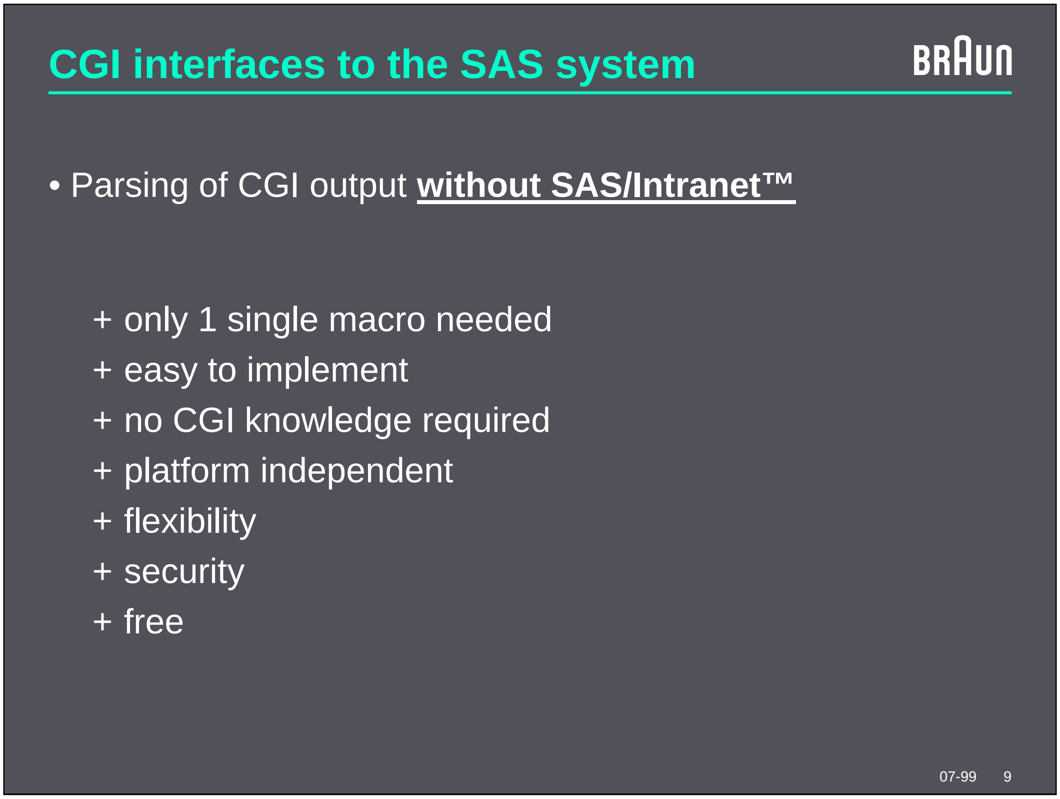
  + easy to implement
  + no CGI knowledge required
  - flexibility
  - potential security risks
  - has to be licensed

07-99 8

CGI interfaces to the SAS system

Parsing of CGI output **<u>with SAS/Intranet™</u>**

+ easy to implement
+ no CGI knowledge required
- flexibility
- potential security risks
- has to be licensed



CGI interfaces to the SAS system

BRAUN

• Parsing of CGI output **without SAS/Intranet™**

+ only 1 single macro needed
+ easy to implement
+ no CGI knowledge required
+ platform independent
+ flexibility
+ security
+ free

07-99 9

CGI interfaces to the SAS system

Parsing of CGI output **without SAS/Intranet™**

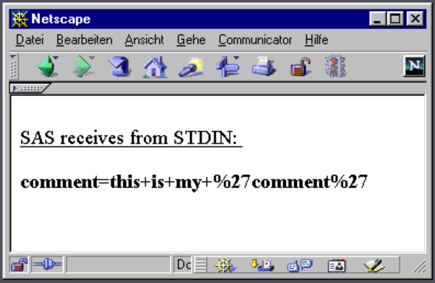
+ only 1 single macro needed
+ easy to implement
+ no CGI knowledge required
+ platform independent
+ flexibility
+ security
+ free

An alternative CGI solution

BRAUN

```
#!/bin/sh
echo "$QUERY_STRING‘cat -‘" \
  | sas demo1.sas -log /tmp/demo1.log

data _null_; infile  STDIN ; input;
  file  STDOUT  noprint notitle;
  put ’Content-type: text/html’ /;
  put ’<u>SAS receives from STDIN:</u>’;
  put ’<b>’ _infile_; stop;
```



SAS receives from STDIN:

comment=this+is+my+%27comment%27

Example 2

- SAS is called via a 2 line shell script which delivers URL encoded form data to 'standard input'
- URL encoded form data are read from predefined fileref STDIN
- HTML embedded results are written to predefined fileref STDOUT

An alternative CGI solution / Example 2

This example consists of 2 parts. A 2 line shell script which starts up SAS, and a simple SAS program which reads in the URL encoded form data from STDIN and prints the data to STDOUT.

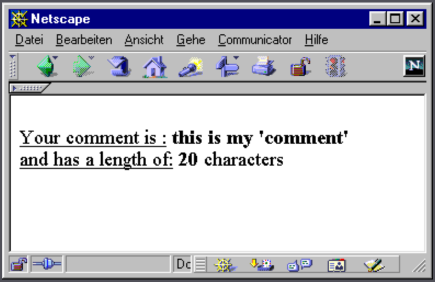
Note that the result is still URL encoded, so that accessing the original form data is not possible.

An alternative CGI solution

BRAUN

```
%get_cgi;    /* CGI parsing is done here */

data _null_; set STDIN;
  where TAG eq 'comment';
  file STDOUT noprint notitle;
  put 'Content-type: text/html' /;
  put '<u>Your comment is :'
        '</u><b> ' VAL '</b><br>';
  put '<u>and has a length of:</u><b> ';
  len = length(VAL); put len '</b>
characters';
```



```
Your comment is : this is my 'comment'
and has a length of: 20 characters
```

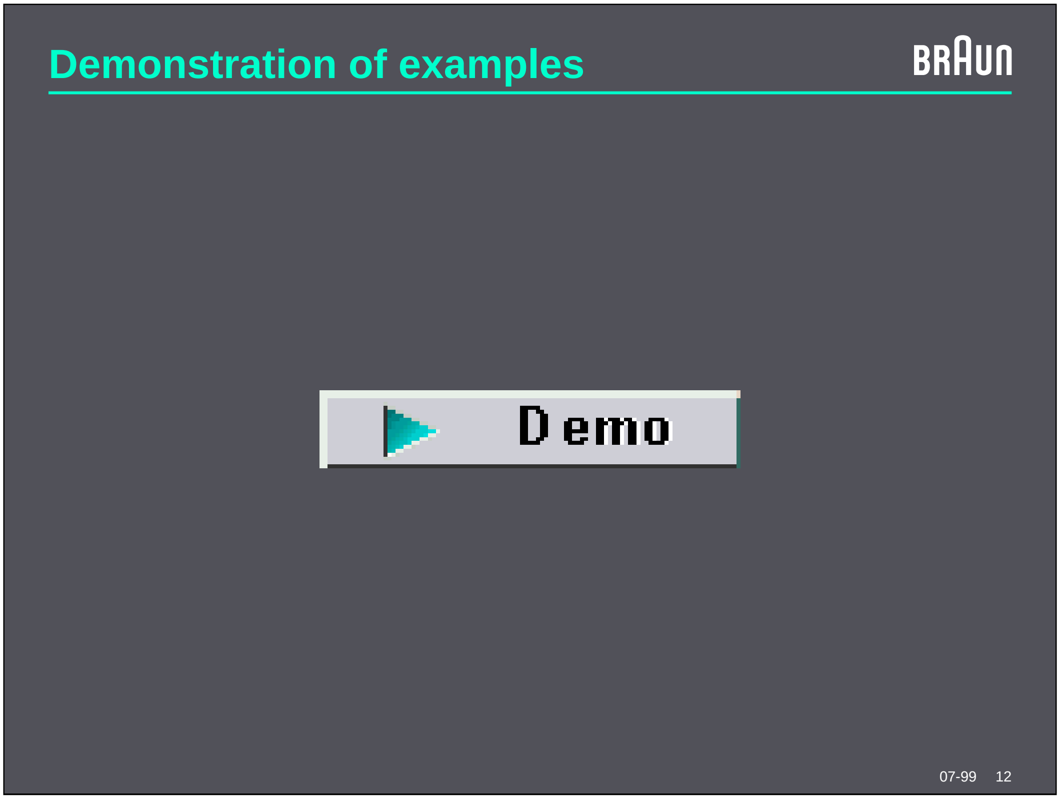
Example 3

- URL encoded form data is parsed with a single free macro
- parsing results are written to a table STDIN with columns TAG and VAL
- different approach unlike SAS/Intranet which is using macro variables
- response time impact is less than 1 second

An alternative CGI solution / Example 3

At this example uses the macro (%get_cgi) to parse the form data from STDIN. parsing results are written to a table STDIN with columns TAG which contains the HTML tag and VAL which contains the entered value. Now the original form data could be accessed and easily processed.

The response time impact of this CGI interface (comparing to a continuously running SAS/Intranet™ session is (depending on the platform you are using) less then 1 second.



Demonstration of examples

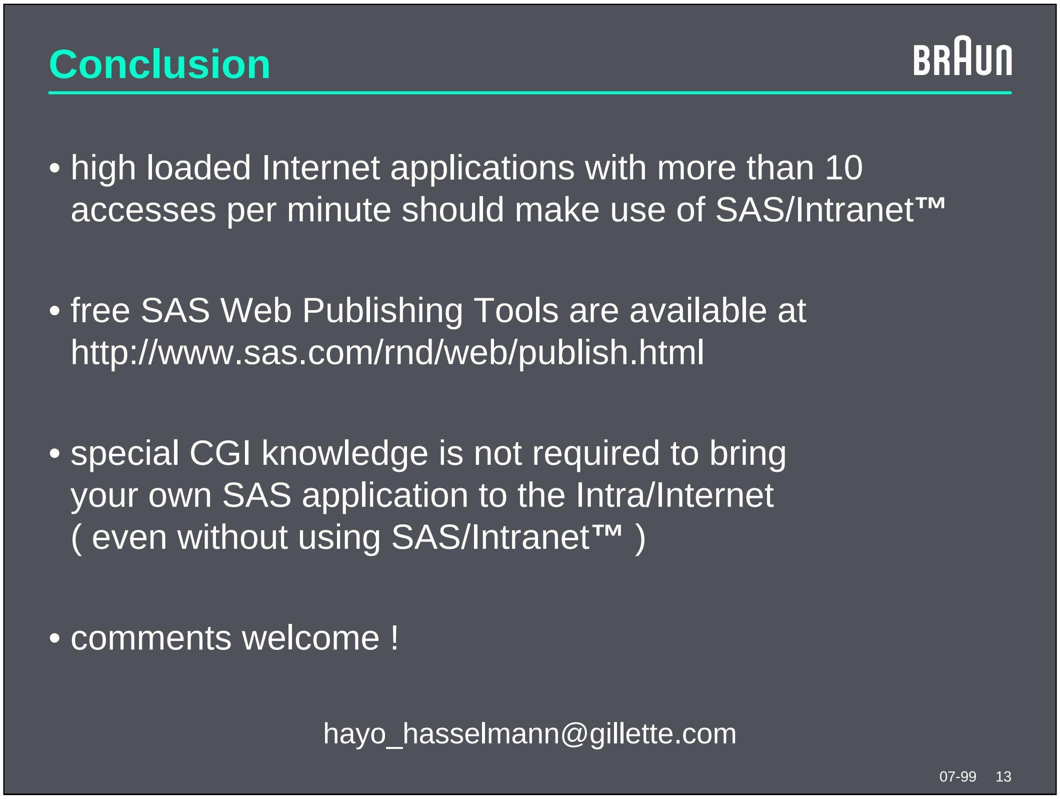
BRAUN

Demo

07-99 12

Demonstration of examples

At this point live examples will be shown



Conclusion

BRAUN

- high loaded Internet applications with more than 10 accesses per minute should make use of SAS/Intranet™
- free SAS Web Publishing Tools are available at http://www.sas.com/rnd/web/publish.html
- special CGI knowledge is not required to bring your own SAS application to the Intra/Internet (even without using SAS/Intranet™)
- comments welcome !

hayo_hasselmann@gillette.com

Conclusion

• high loaded Internet applications with more than 10 accesses per minute should make use of SAS/Intranet™

• free SAS Web Publishing Tools are available at http://www.sas.com/rnd/web/publish.html

• special CGI knowledge is not required to bring your own SAS application to the Intra/Internet (even without using SAS/Intranet™)

BRAUN

Questions...?

Questions ?