How to proceed when 1 000 call agents tell you: 'My Computer is slow'

Tobias Oetiker <tobi@oetiker.ch>

OETIKER+PARTNER AG

22nd Large Installation System Administration Conference

boot up

- users blame IT performance
- stop watch and heisenbugs
- sysinternals tools
- autoit and winspy
- sorry, no quick fix
- but we can monitor it

boot up

- users blame IT performance
- stop watch and heisenbugs
- sysinternals tools
- autoit and winspy
- sorry, no quick fix
- but we can monitor it

boot up

- users blame IT performance
- stop watch and heisenbugs
- sysinternals tools
- autoit and winspy
- sorry, no quick fix
- but we can monitor it

boot up

- users blame IT performance
- stop watch and heisenbugs
- sysinternals tools
- autoit and winspy
- sorry, no quick fix
- but we can monitor it

boot up

- users blame IT performance
- stop watch and heisenbugs
- sysinternals tools
- autoit and winspy
- sorry, no quick fix
- but we can monitor it

boot up

- users blame IT performance
- stop watch and heisenbugs
- sysinternals tools
- autoit and winspy
- sorry, no quick fix
- but we can monitor it

design goals

- passive monitoring from users perspective
- let users give their input
- minimal impact
- simple setup and update
- central data store

design goals

- passive monitoring from users perspective
- let users give their input
- minimal impact
- simple setup and update
- central data store

design goals

- passive monitoring from users perspective
- let users give their input
- minimal impact
- simple setup and update
- central data store

design goals

- passive monitoring from users perspective
- let users give their input
- minimal impact
- simple setup and update
- central data store

design goals

- passive monitoring from users perspective
- let users give their input
- minimal impact
- simple setup and update
- central data store

three tools

- CPV monitor: observe the system
- CPV reporter: easy problem reporting
- CPV explorer: view the results

three tools

- CPV monitor: observe the system
- CPV reporter: easy problem reporting
- CPV explorer: view the results

three tools

- CPV monitor: observe the system
- CPV reporter: easy problem reporting
- CPV explorer: view the results

cpv monitor and perl/CPAN

Look it's perl honey!

- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

cpv monitor and perl/CPAN

Look it's perl honey!

- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

cpv monitor and perl/CPAN

Look it's perl honey!

- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

cpv monitor and perl/CPAN

Look it's perl honey!

- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

cpv monitor and perl/CPAN

Look it's perl honey!

- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

cpv monitor and perl/CPAN

Look it's perl honey!

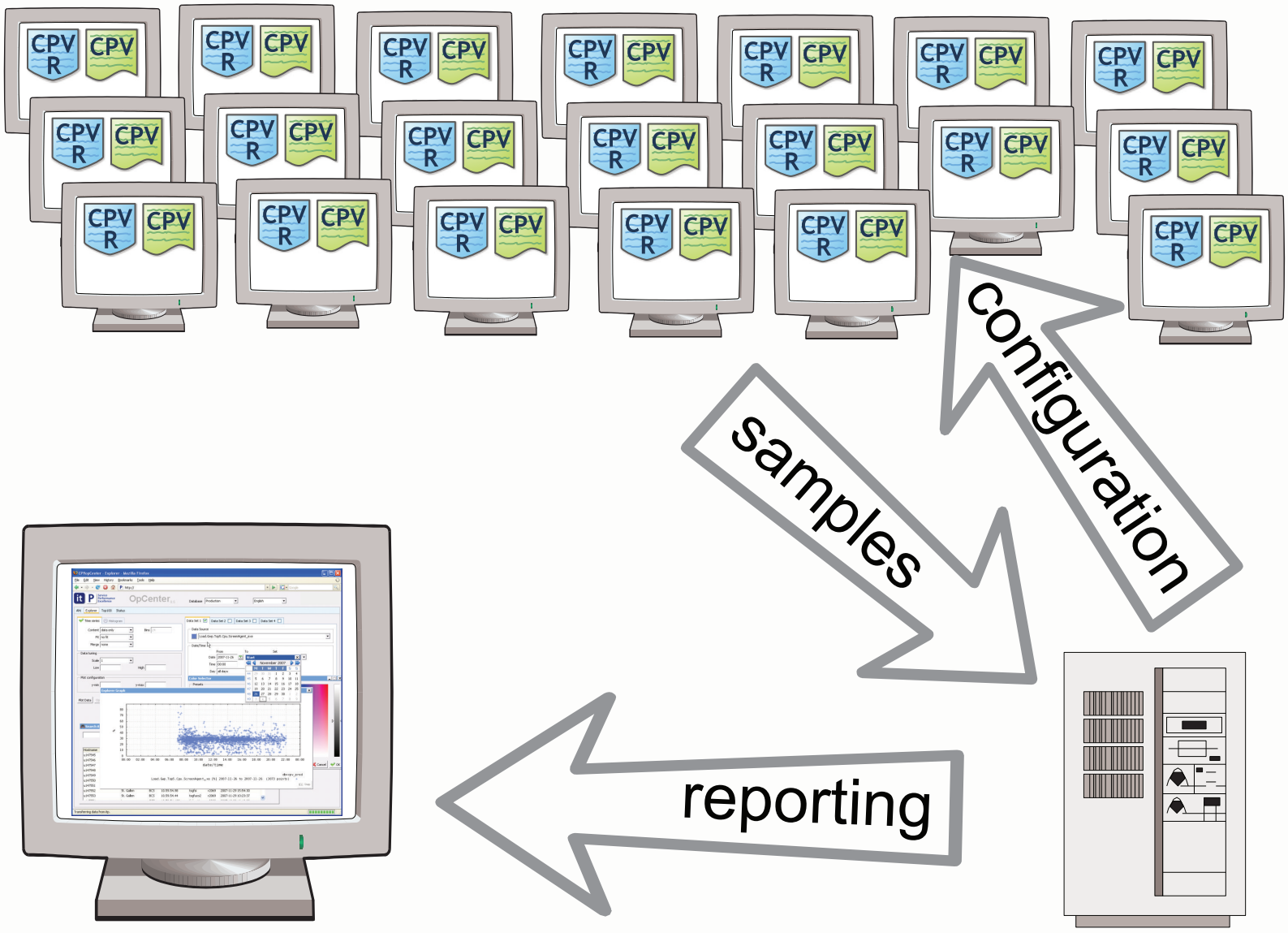
- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

cpv monitor and perl/CPAN

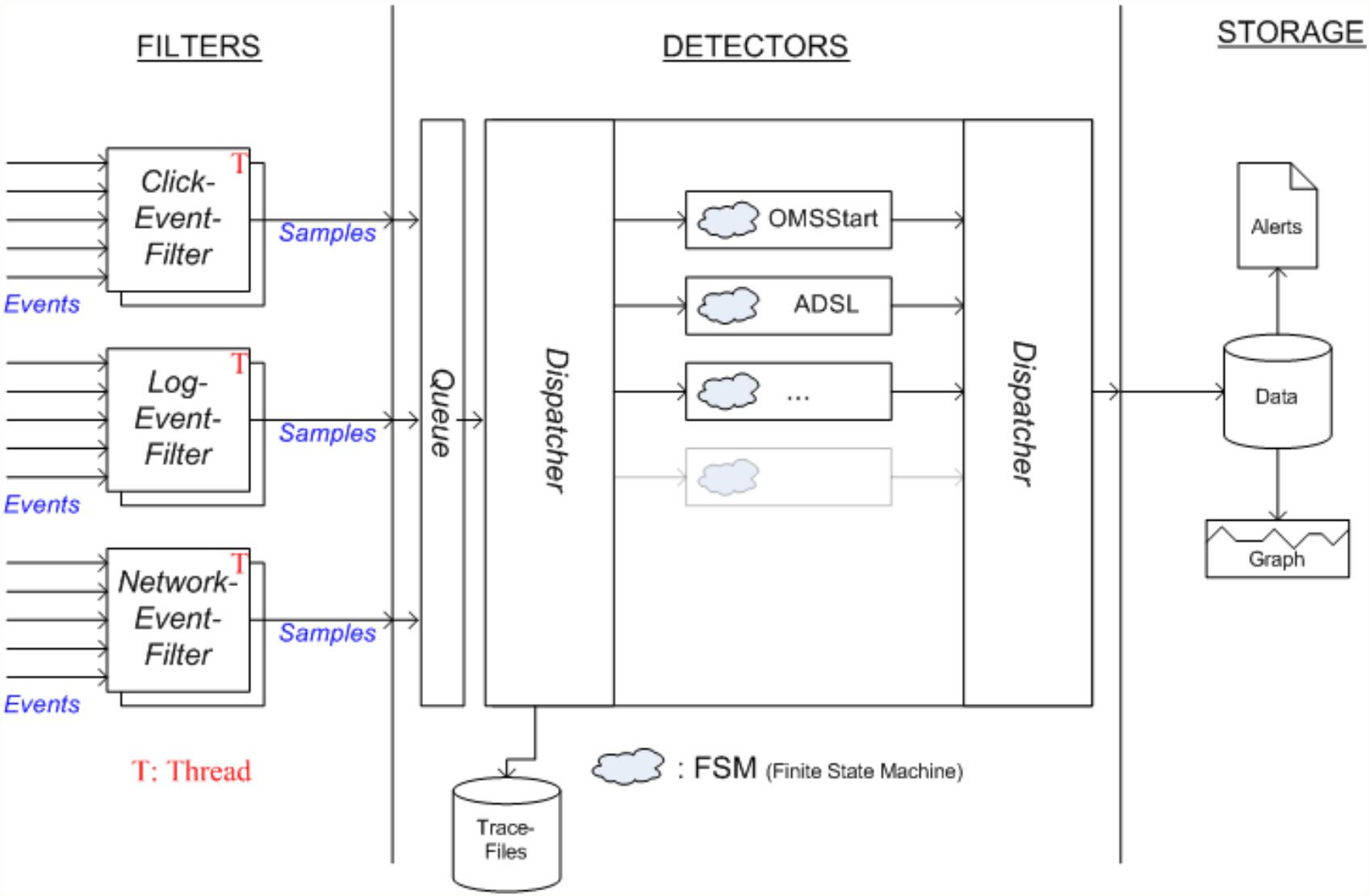
Look it's perl honey!

- AutoIt
- `use Win32::GuiTest;`
- `use Win32::API;`
- `use Win32::OLE;`
- `use Win32::GUI;`
- `use FSA::Rules;`
- `use threads;`

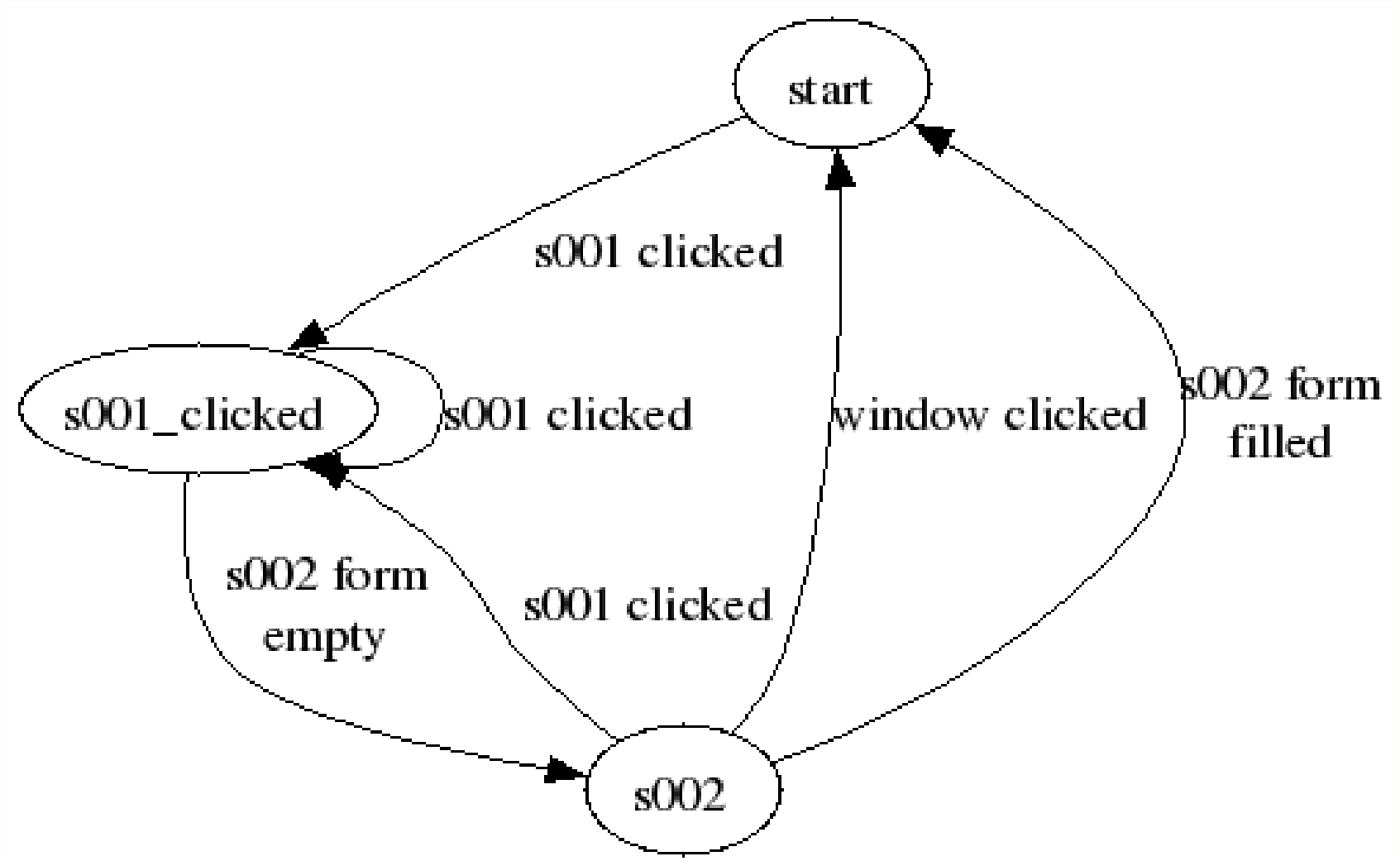
cpv system overview



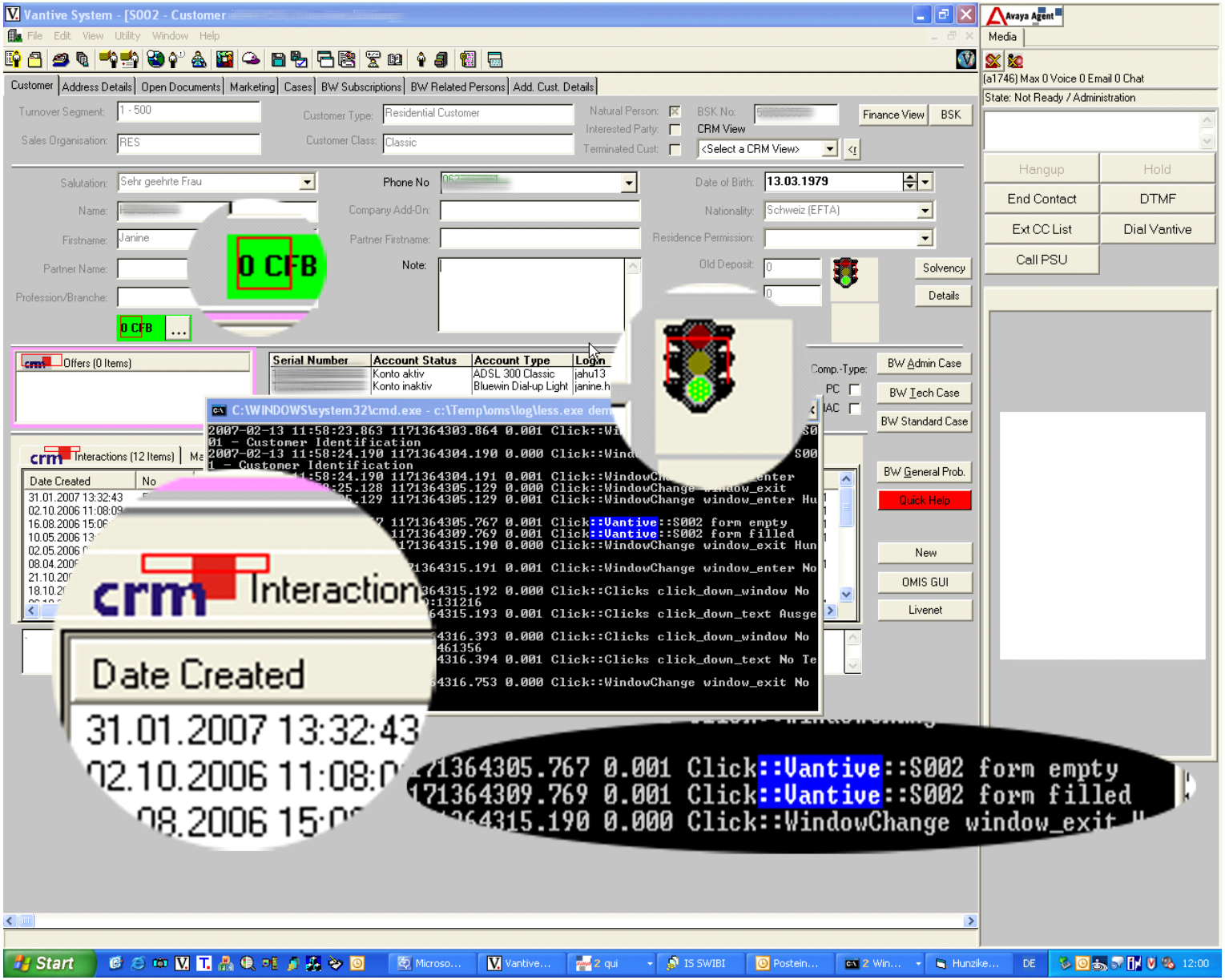
cpv monitor structure



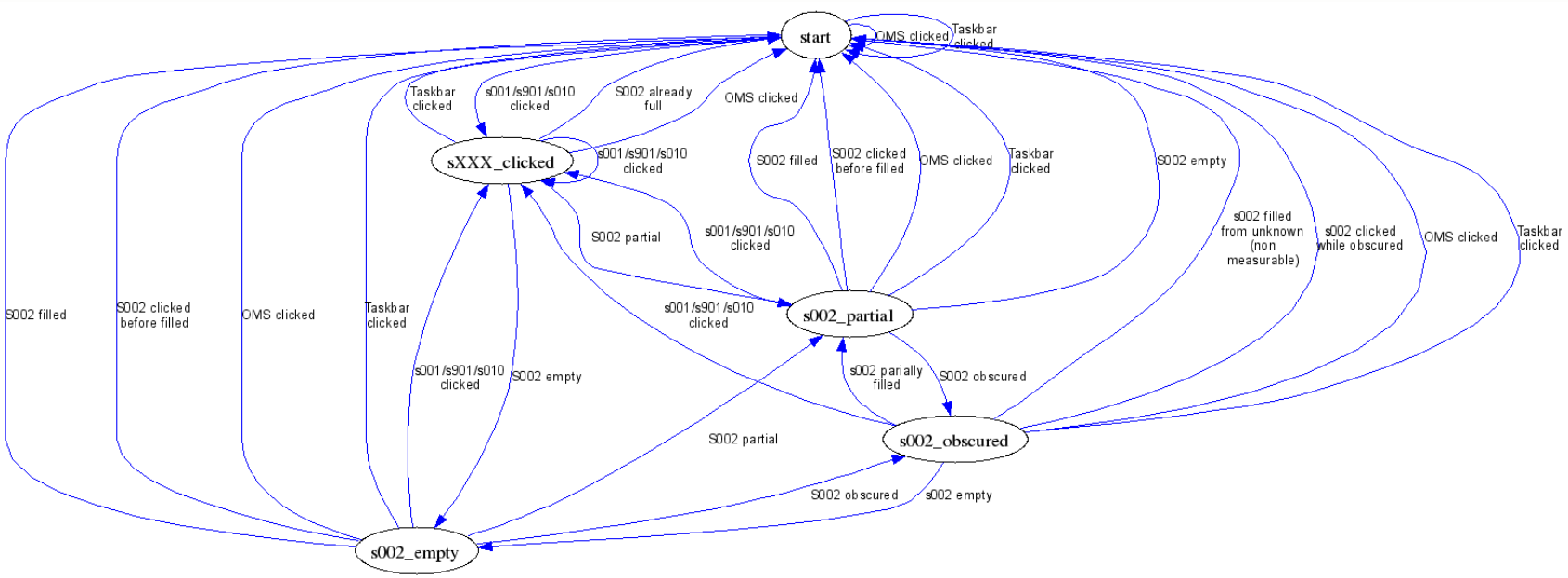
lesson #1: fsm are cool



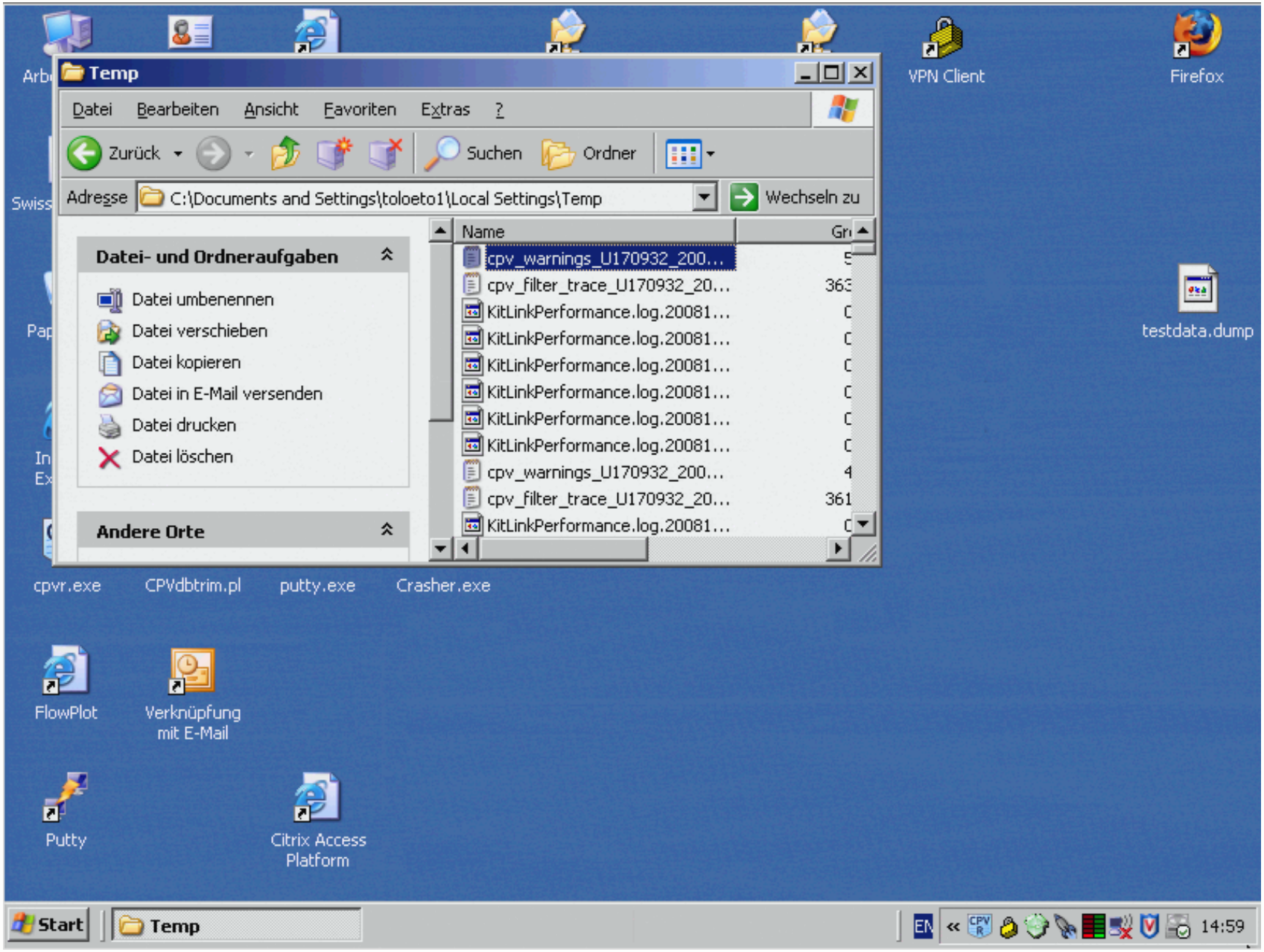
lesson #1: seemingly simple



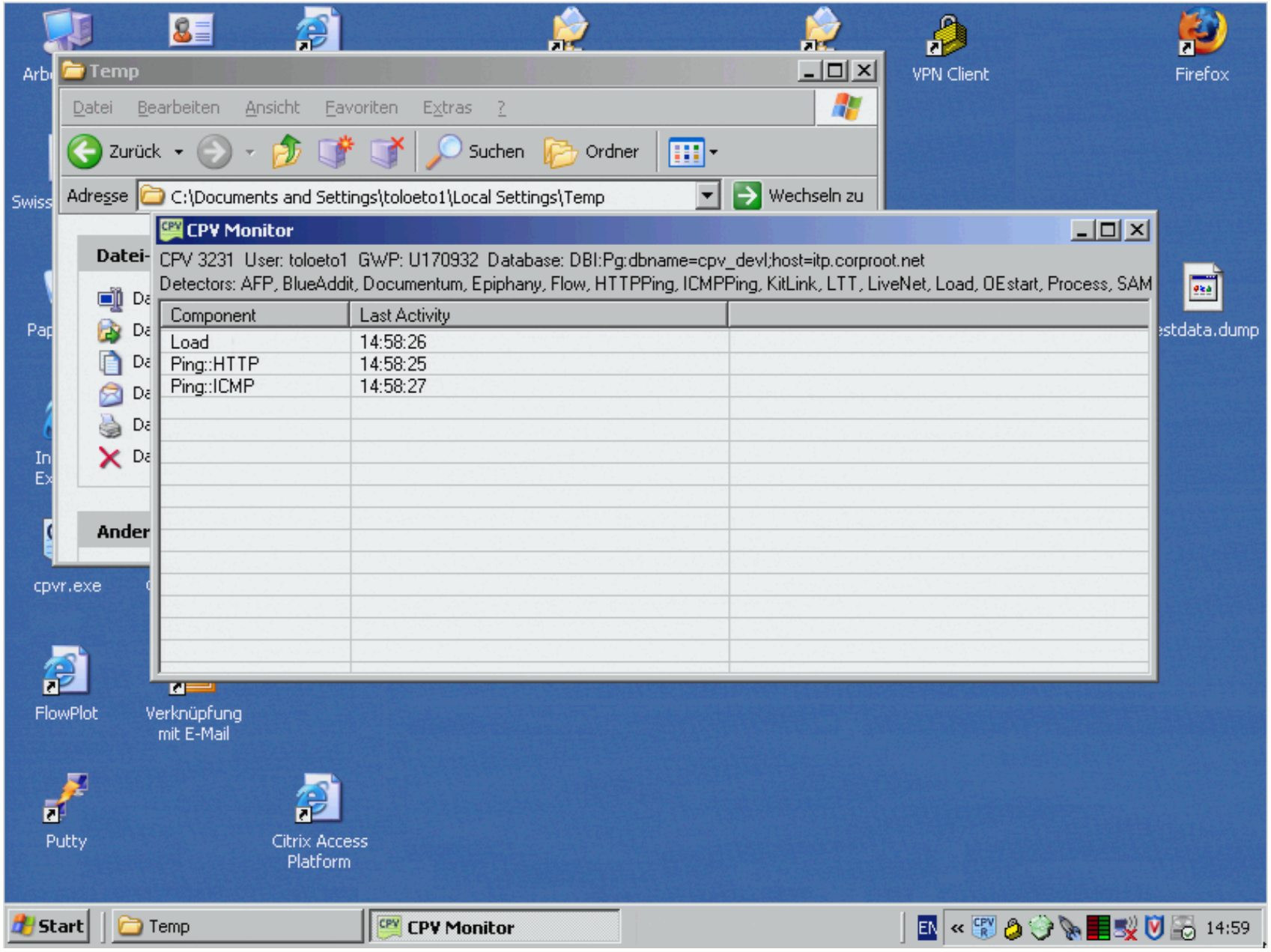
lesson #1: complexity trap



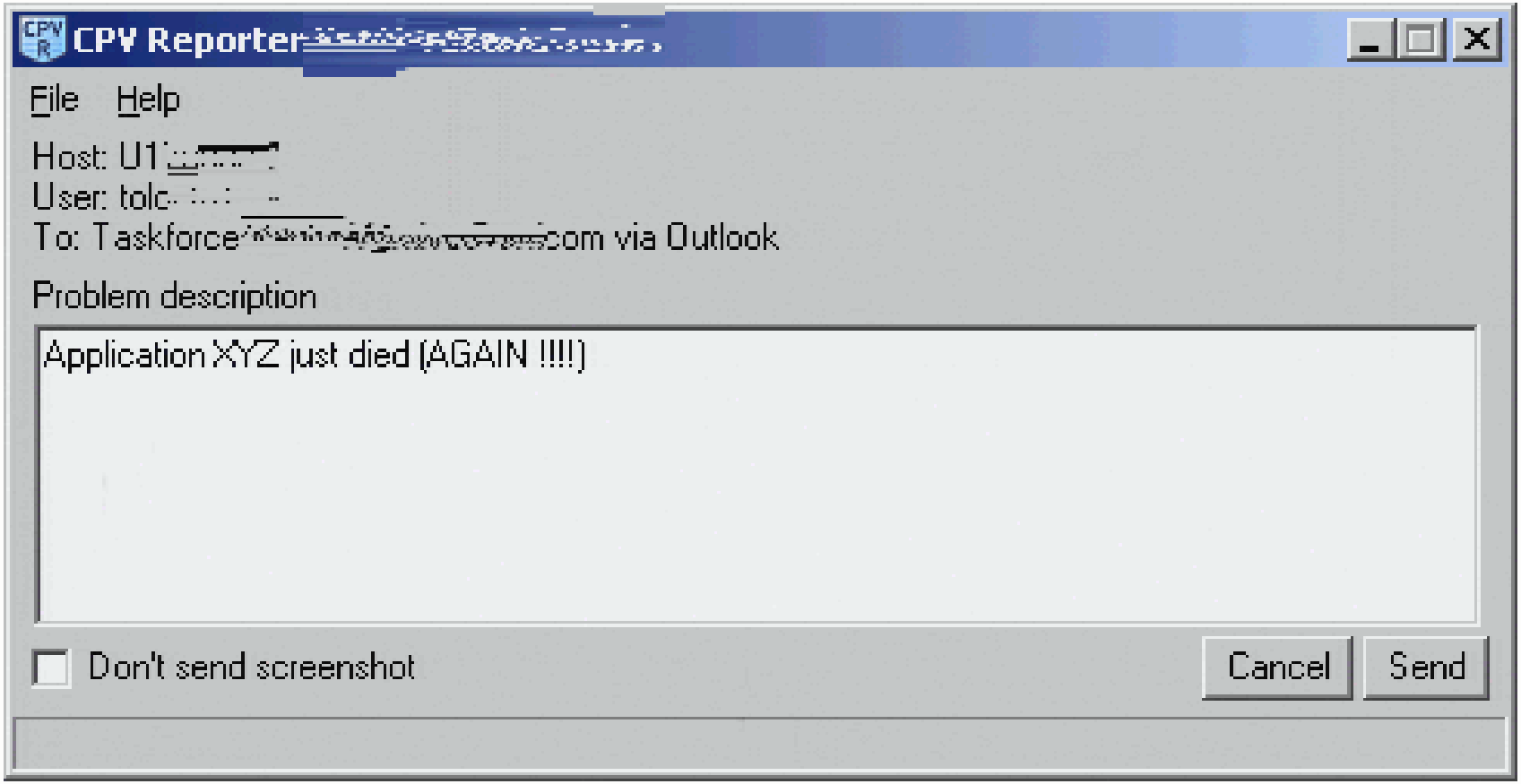
cpv monitor

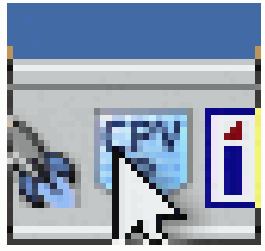

Temp
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück Suchen Ordner
Adresse C:\Documents and Settings\toloeto1\Local Settings\Temp
Wechseln zu
Datei- und Ordneraufgaben
Datei umbenennen
Datei verschieben
Datei kopieren
Datei in E-Mail versenden
Datei drucken
Datei löschen
Andere Orte
Name
cpv_warnings_U170932_200...
cpv_filter_trace_U170932_20...
KitLinkPerformance.log.20081...
cpv_warnings_U170932_200...
cpv_filter_trace_U170932_20...
KitLinkPerformance.log.20081...
VPN Client
Firefox
testdata.dump
cpvr.exe
CPVdbtrim.pl
putty.exe
Crasher.exe
FlowPlot
Verknüpfung mit E-Mail
Putty
Citrix Access Platform
Start
Temp
14:59

cpv monitor monitor

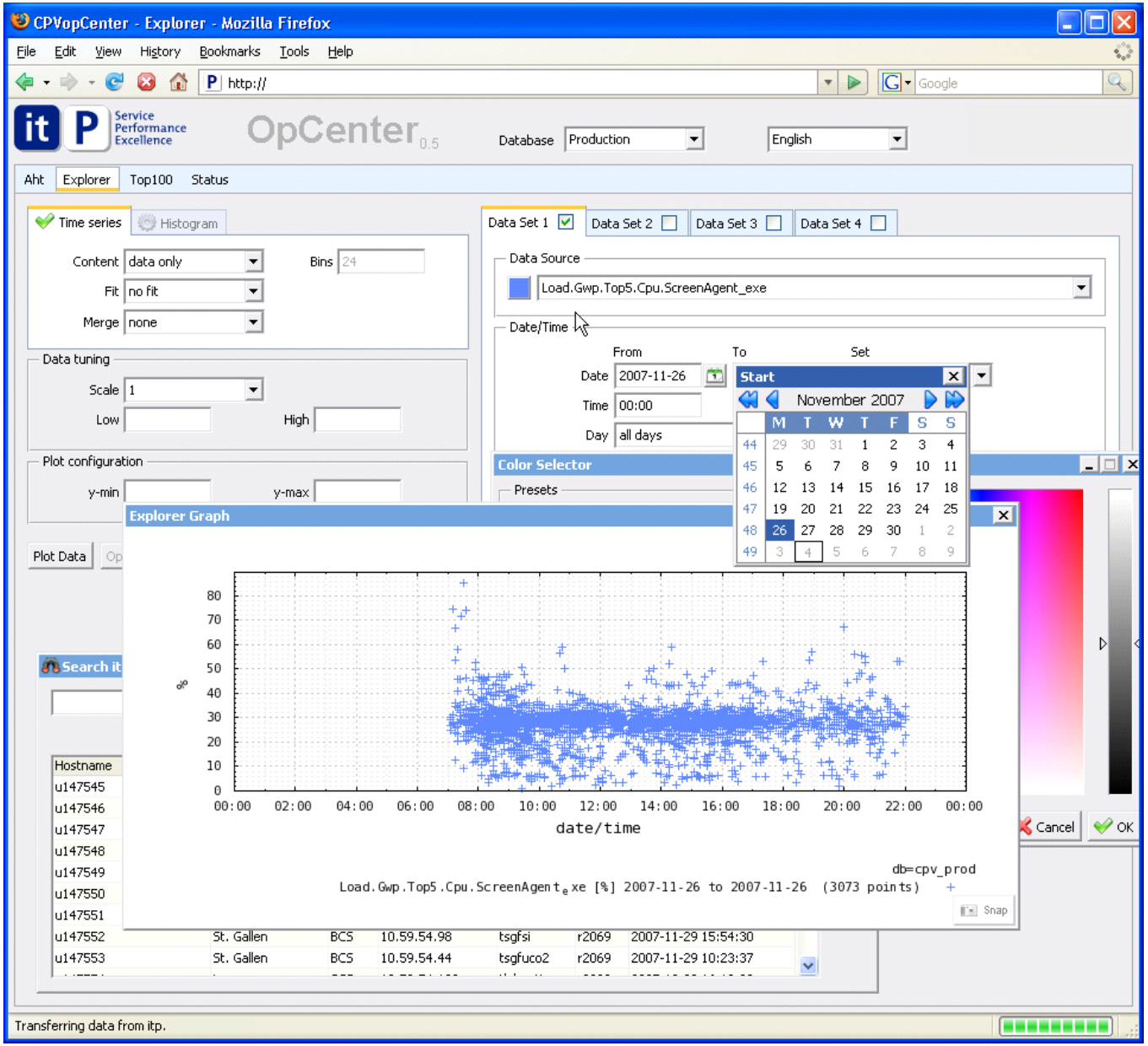


cpv reporter





cpv explorer



thinking BIG

wants

- $\sim$ 1500 clients in the call-center
- dynamic configuration
- individual profiles

infrastructure

data store : PostgreSQL

configuration : Apache, CPVservice.cgi

analysis : Apache, Qooxdoo, CPVjson.cgi, Gnuplot

thinking BIG

wants

- $\sim$ 1500 clients in the call-center
- dynamic configuration
- individual profiles

infrastructure

data store : PostgreSQL

configuration : Apache, CPVservice.cgi

analysis : Apache, Qooxdoo, CPVjson.cgi, Gnuplot

thinking BIG

wants

- $\sim$ 1500 clients in the call-center
- dynamic configuration
- individual profiles

infrastructure

data store : PostgreSQL

configuration : Apache, CPVservice.cgi

analysis : Apache, Qooxdoo, CPVjson.cgi, Gnuplot

thinking BIG

wants

- $\sim$ 1500 clients in the call-center
- dynamic configuration
- individual profiles

infrastructure

data store : PostgreSQL

configuration : Apache, CPVservice.cgi

analysis : Apache, Qooxdoo, CPVjson.cgi, Gnuplot

thinking BIG

wants

- $\sim$ 1500 clients in the call-center
- dynamic configuration
- individual profiles

infrastructure

data store : PostgreSQL

configuration : Apache, CPVservice.cgi

analysis : Apache, Qooxdoo, CPVjson.cgi, Gnuplot

thinking BIG

wants

- $\sim$ 1500 clients in the call-center
- dynamic configuration
- individual profiles

infrastructure

data store : PostgreSQL
configuration : Apache, CPVservice.cgi
analysis : Apache, Qooxdoo, CPVjson.cgi, Gnuplot

observation tools

- GetWindowText and friends
- Reading log files
- Windows WMI (Load, Processes)
- Active Probing (Ping, HTTP)
- HTTPAnalyzer ($$$) for http(s)
- Full Custom Probes

observation tools

- GetWindowText and friends
- Reading log files
- Windows WMI (Load, Processes)
- Active Probing (Ping, HTTP)
- HTTPAnalyzer ($$$) for http(s)
- Full Custom Probes

observation tools

- GetWindowText and friends
- Reading log files
- Windows WMI (Load, Processes)
- Active Probing (Ping, HTTP)
- HTTPAnalyzer ($$$) for http(s)
- Full Custom Probes

observation tools

- GetWindowText and friends
- Reading log files
- Windows WMI (Load, Processes)
- Active Probing (Ping, HTTP)
- HTTPAnalyzer ($$$) for http(s)
- Full Custom Probes

observation tools

- GetWindowText and friends
- Reading log files
- Windows WMI (Load, Processes)
- Active Probing (Ping, HTTP)
- HTTPAnalyzer ($$$) for http(s)
- Full Custom Probes

observation tools

- GetWindowText and friends
- Reading log files
- Windows WMI (Load, Processes)
- Active Probing (Ping, HTTP)
- HTTPAnalyzer ($$$) for http(s)
- Full Custom Probes

lesson #2: finding outlook errors

- outlook modal popup send button does not work
- GetAsyncKeyState: Although the least significant bit of the return value indicates whether the key has been pressed since the last query, due to the pre-emptive multitasking nature of Windows, another application can call GetAsyncKeyState and receive the "recently pressed" bit instead of your application. **The behavior of the least significant bit** of the return value is retained strictly for compatibility with 16-bit Windows applications (which are non-preemptive) and **should not be relied upon**.
- `GetClassName(WindowFromPoint(GetCursorPos()))eq 'MsoCommandBar';`

lesson #2: finding outlook errors

- outlook modal popup send button does not work
- GetAsyncKeyState: Although the least significant bit of the return value indicates whether the key has been pressed since the last query, due to the pre-emptive multitasking nature of Windows, another application can call GetAsyncKeyState and receive the "recently pressed" bit instead of your application. **The behavior of the least significant bit** of the return value is retained strictly for compatibility with 16-bit Windows applications (which are non-preemptive) and **should not be relied upon**.
- `GetClassName(WindowFromPoint(GetCursorPos()))`
  `eq 'MsoCommandBar';`

lesson #2: finding outlook errors

- outlook modal popup send button does not work
- GetAsyncKeyState: Although the least significant bit of the return value indicates whether the key has been pressed since the last query, due to the pre-emptive multitasking nature of Windows, another application can call GetAsyncKeyState and receive the "recently pressed" bit instead of your application. **The behavior of the least significant bit** of the return value is retained strictly for compatibility with 16-bit Windows applications (which are non-preemptive) and **should not be relied upon**.
- `GetClassName(WindowFromPoint(GetCursorPos()))`
  `eq 'MsoCommandBar';`

lesson #3: WMGetText

- GetWindowText or WMGetText
- Application becomes real busy with WMGetText
- stay with GetWindowText

lesson #3: WMGetText

- GetWindowText or WMGetText
- Application becomes real busy with WMGetText
- stay with GetWindowText

lesson #3: WMGetText

- GetWindowText or WMGetText
- Application becomes real busy with WMGetText
- stay with GetWindowText

lesson #4: server issues

- 2008-10-27: 1,459 devices sent 2,417,807 samples
- 4 Core / 32-bit / 4 GB ram
- 40 days of data 100,000,000 samples
- index does not fit in ram
- too much data for processing

lesson #4: server issues

- 2008-10-27: 1,459 devices sent 2,417,807 samples
- 4 Core / 32-bit / 4 GB ram
- 40 days of data 100,000,000 samples
- index does not fit in ram
- too much data for processing

lesson #4: server issues

- 2008-10-27: 1,459 devices sent 2,417,807 samples
- 4 Core / 32-bit / 4 GB ram
- 40 days of data 100,000,000 samples
- index does not fit in ram
- too much data for processing

lesson #4: server issues

- 2008-10-27: 1,459 devices sent 2,417,807 samples
- 4 Core / 32-bit / 4 GB ram
- 40 days of data 100,000,000 samples
- index does not fit in ram
- too much data for processing

lesson #4: server issues

- 2008-10-27: 1,459 devices sent 2,417,807 samples
- 4 Core / 32-bit / 4 GB ram
- 40 days of data 100,000,000 samples
- index does not fit in ram
- too much data for processing

lesson #5: index compaction

- function based index
- hours since 2007 is good for 7 years with 2 byte
- 2 byte for metric id
- 2 byte for workstation id
- two WHERE conditions

lesson #5: index compaction

- function based index
- hours since 2007 is good for 7 years with 2 byte
- 2 byte for metric id
- 2 byte for workstation id
- two WHERE conditions

lesson #5: index compaction

- function based index
- hours since 2007 is good for 7 years with 2 byte
- 2 byte for metric id
- 2 byte for workstation id
- two WHERE conditions

lesson #5: index compaction

- function based index
- hours since 2007 is good for 7 years with 2 byte
- 2 byte for metric id
- 2 byte for workstation id
- two WHERE conditions

lesson #5: index compaction

- function based index
- hours since 2007 is good for 7 years with 2 byte
- 2 byte for metric id
- 2 byte for workstation id
- two WHERE conditions

lesson #6: random data reduction

- too much data for statistics
- how to get 12 % of the samples?
- add 2 byte random value to each sample
- select all sample with rand $<$ maxrand $\frac{12}{100}$

lesson #6: random data reduction

- too much data for statistics
- how to get 12 % of the samples?
- add 2 byte random value to each sample
- select all sample with rand $<$ maxrand $\frac{12}{100}$

lesson #6: random data reduction

- too much data for statistics
- how to get 12 % of the samples?
- add 2 byte random value to each sample
- select all sample with rand $<$ maxrand $\frac{12}{100}$

lesson #6: random data reduction

- too much data for statistics
- how to get 12 % of the samples?
- add 2 byte random value to each sample
- select all sample with rand $<$ maxrand $\frac{12}{100}$

lesson #7: threaded perl

- works very well on win32
- full copy — lots of memory
- save require modules after creating the thread
- only thread where really necessary

lesson #7: threaded perl

- works very well on win32
- full copy — lots of memory
- save require modules after creating the thread
- only thread where really necessary

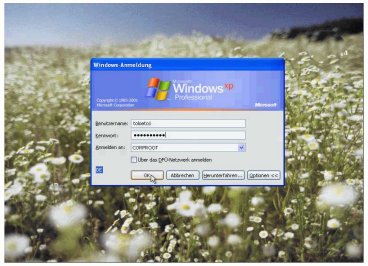
lesson #7: threaded perl

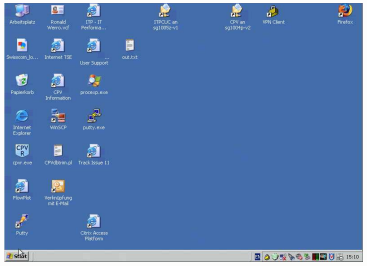
- works very well on win32
- full copy — lots of memory
- save require modules after creating the thread
- only thread where really necessary

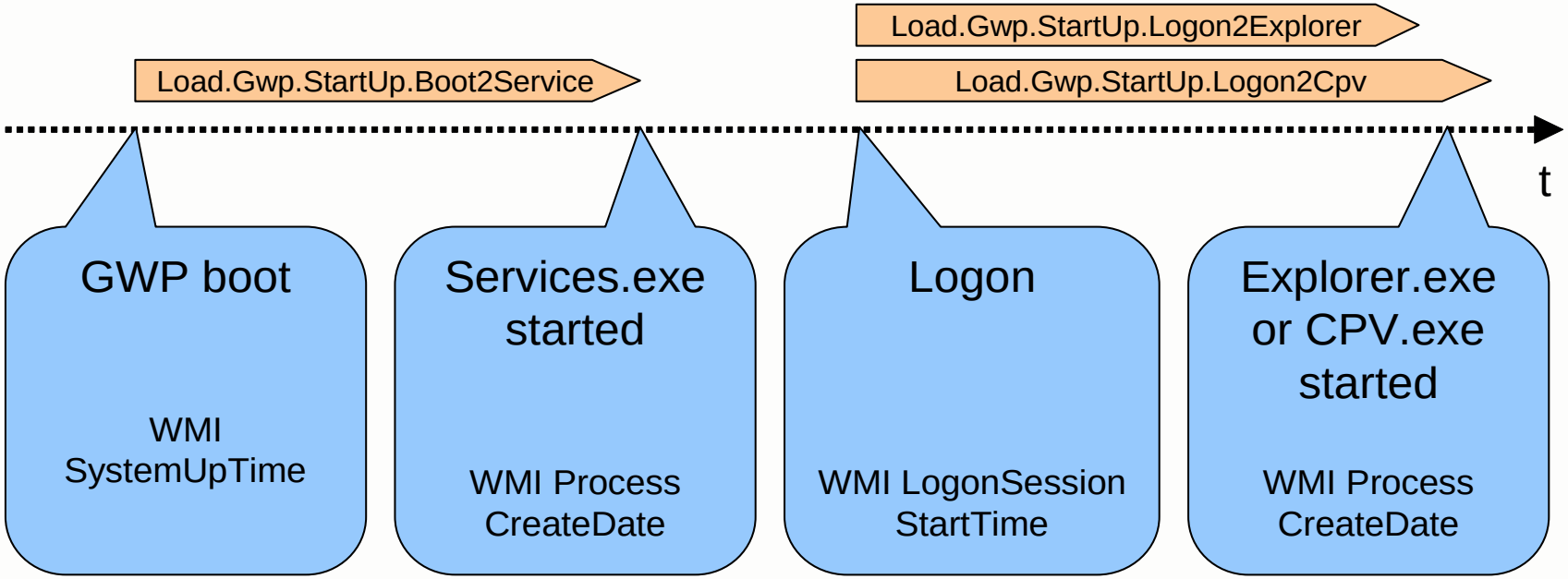
lesson #7: threaded perl

- works very well on win32
- full copy — lots of memory
- save require modules after creating the thread
- only thread where really necessary

lesson #8: measuring boot and logon time







lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

Implementation

- find active window
- attach process handle
- poll for exit code

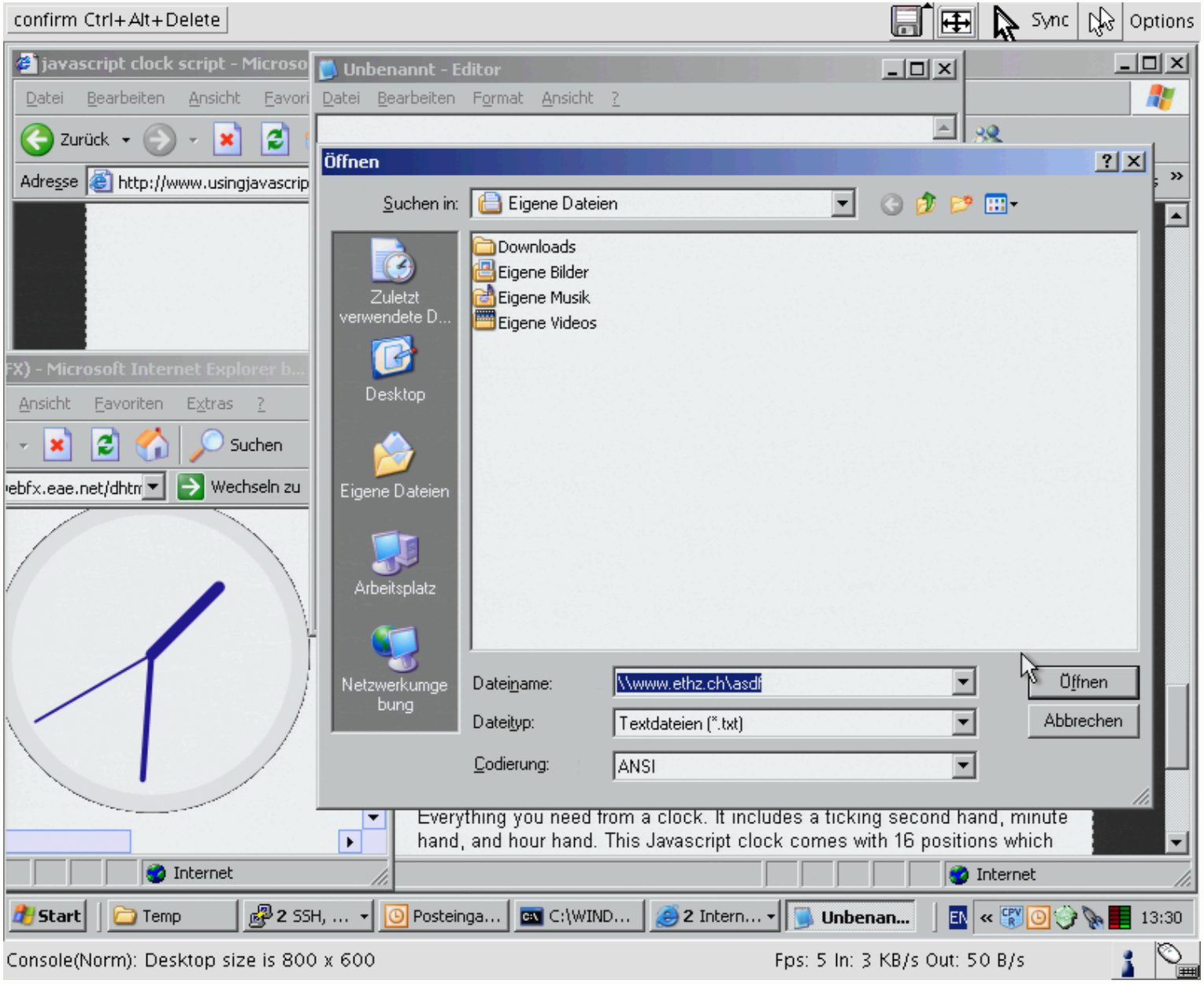
lesson #9: detecting crashes

- no wait but process handle
- no signals only exit codes
- 0xC0000005 - segfault
- 0x00000103 - still running
- `TerminateProcess` can define exit code

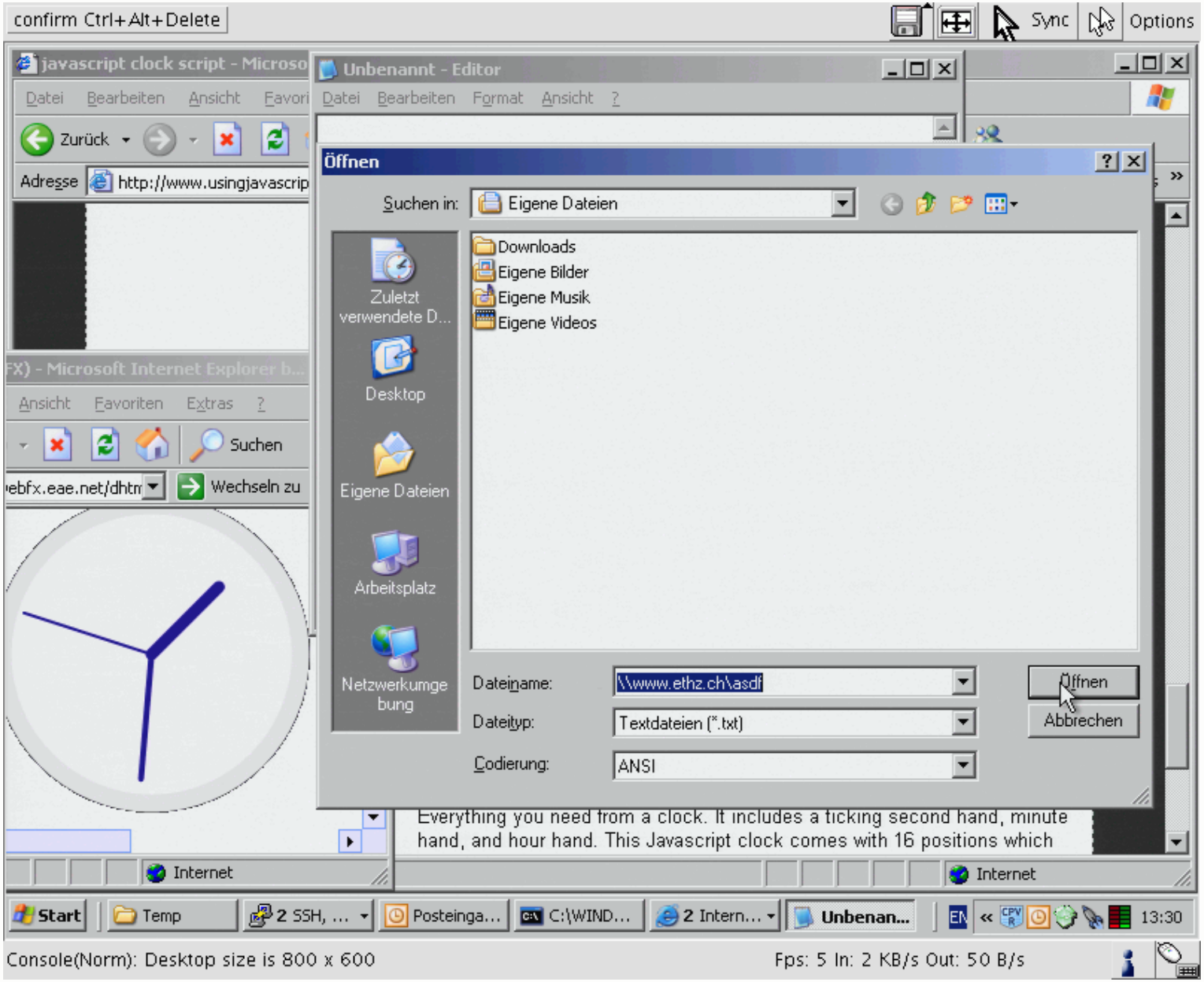
Implementation

- find active window
- attach process handle
- poll for exit code

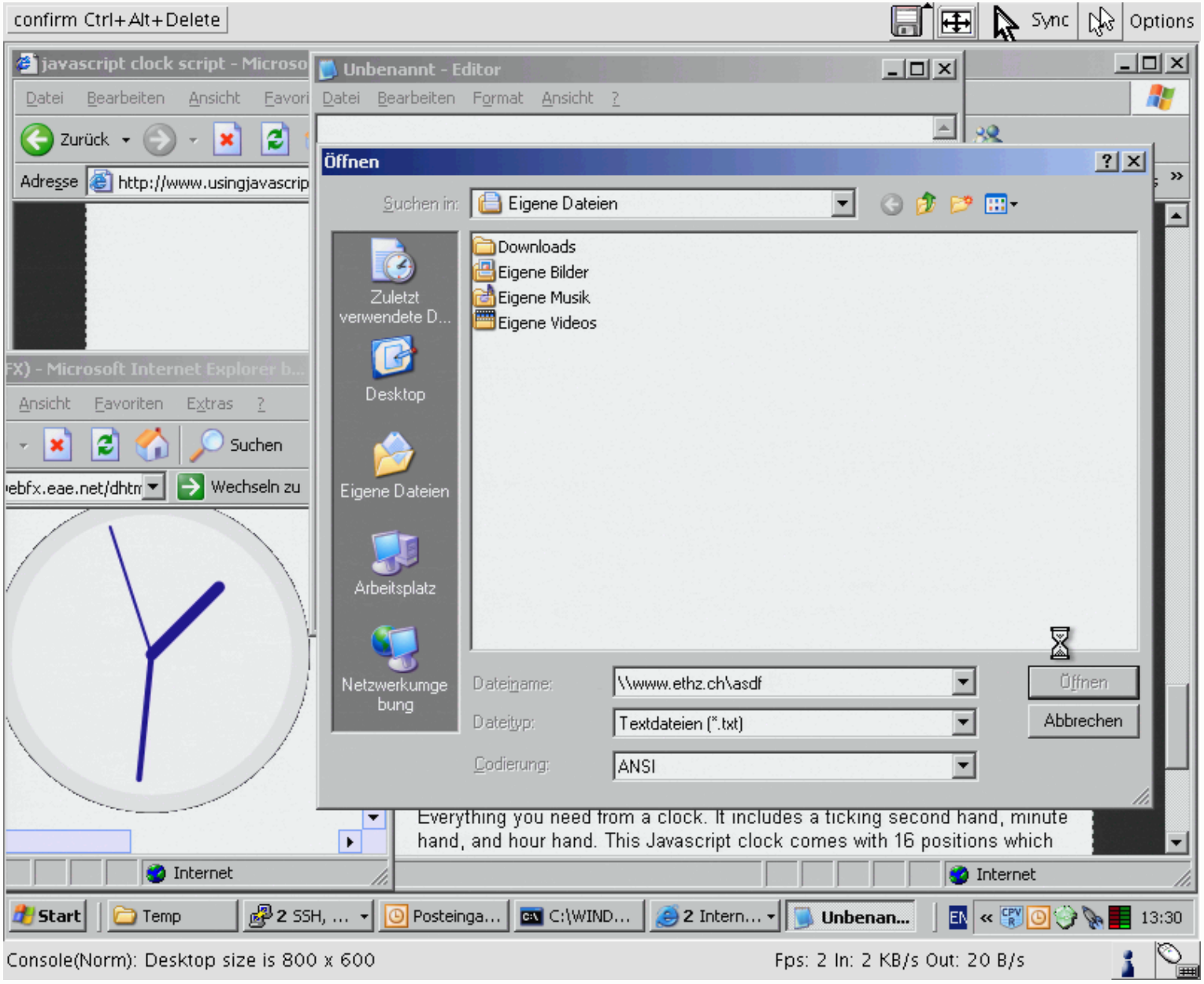
lesson #10: application hangs - symptoms



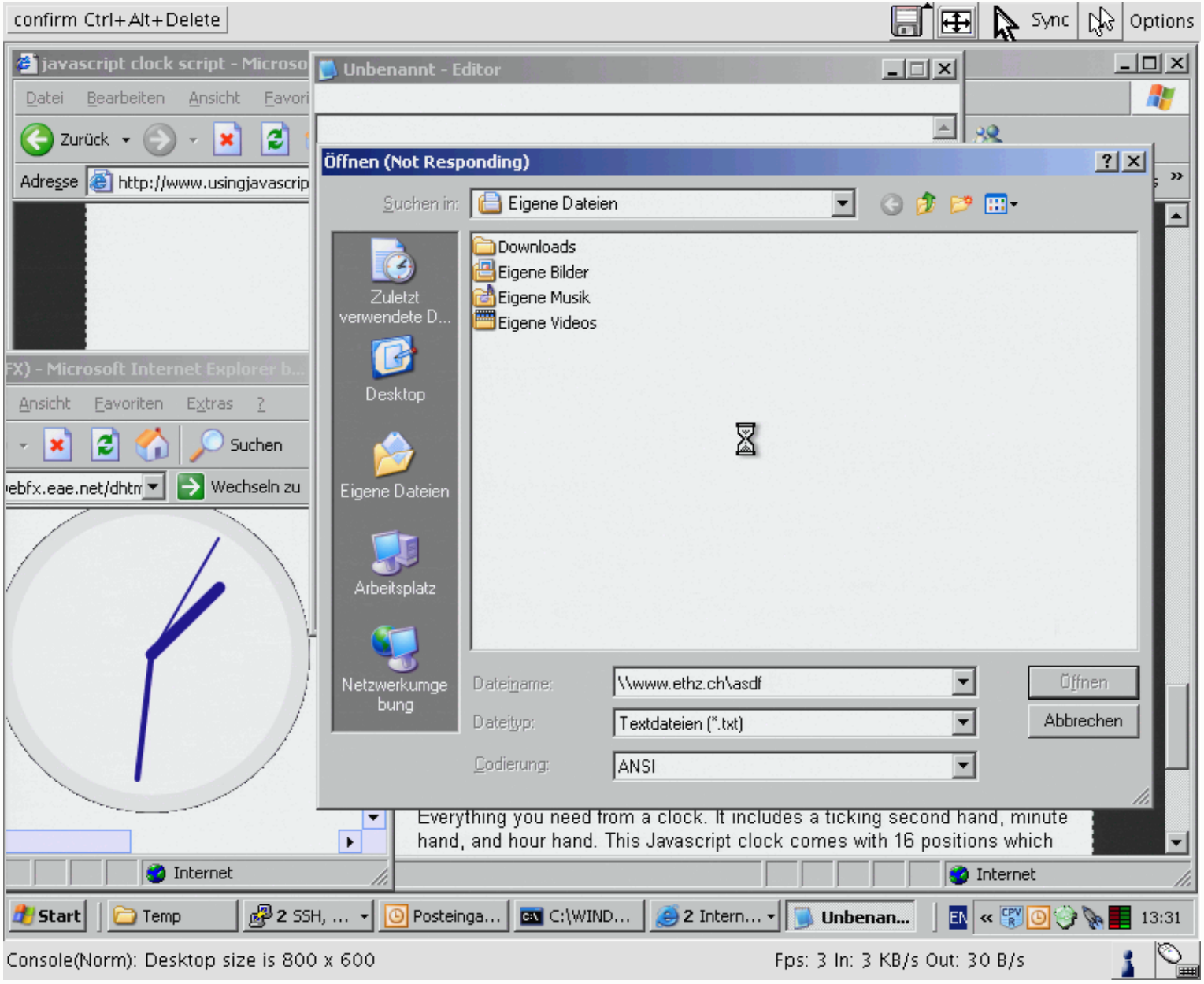
lesson #10: application hangs - symptoms



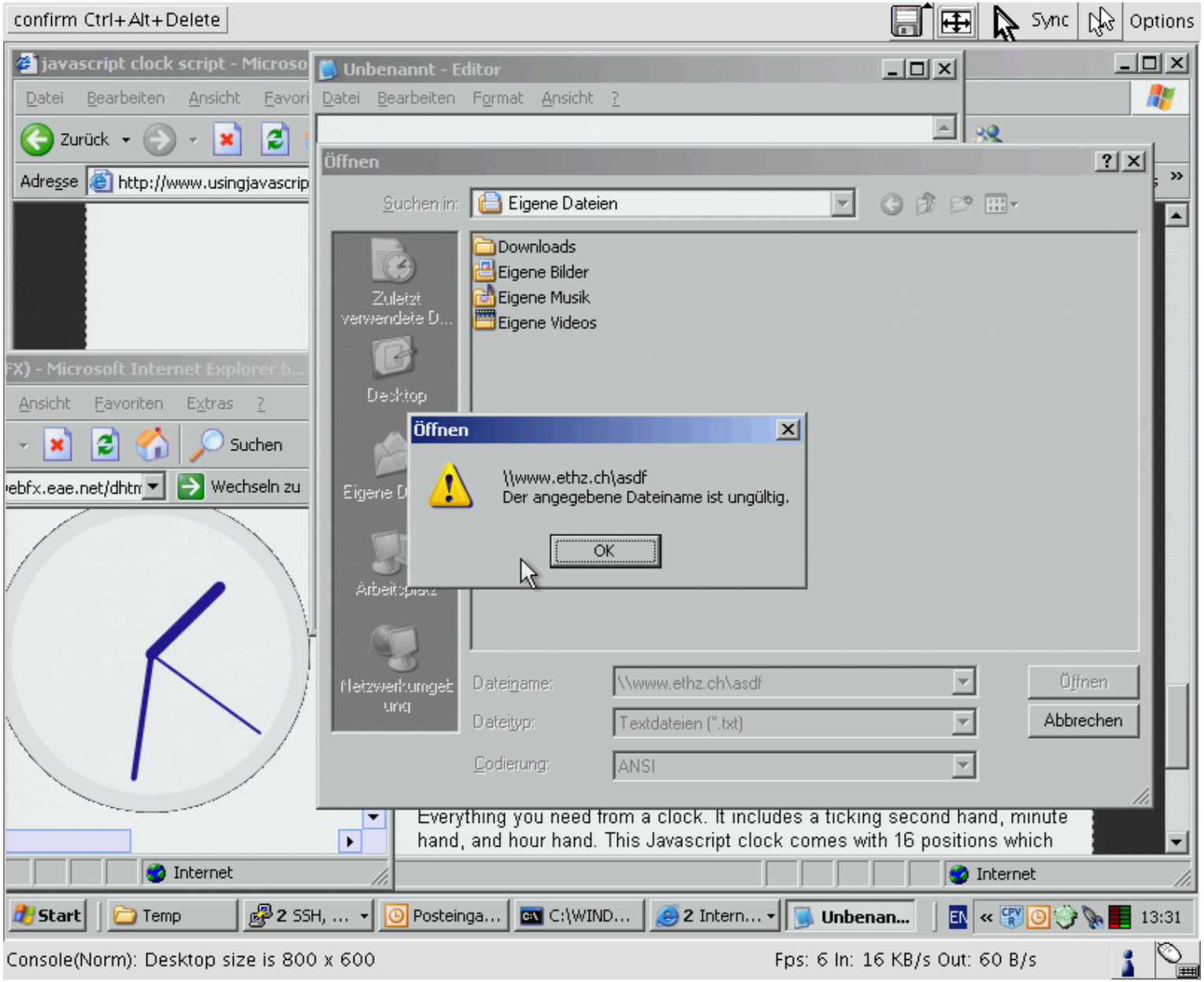
lesson #10: application hangs - symptoms



lesson #10: application hangs - symptoms



lesson #10: application hangs - symptoms



lesson #10: application hangs - detection

- dead apps don't process messages
- explorer fakes responsiveness

Implementation

- find active window
- window ping: SendMessage `WM_NULL`
- wait until the window is back

lesson #10: application hangs - detection

- dead apps don't process messages
- explorer fakes responsiveness

Implementation

- find active window
- window ping: SendMessage `WM_NULL`
- wait until the window is back

lesson #10: application hangs - detection

- dead apps don't process messages
- explorer fakes responsiveness

Implementation

- find active window
- window ping: SendMessage `WM_NULL`
- wait until the window is back

lesson #10: application hangs - detection

- dead apps don't process messages
- explorer fakes responsiveness

Implementation

- find active window
- window ping: SendMessage `WM_NULL`
- wait until the window is back

lesson #10: application hangs - detection

- dead apps don't process messages
- explorer fakes responsiveness

Implementation

- find active window
- window ping: SendMessage `WM_NULL`
- wait until the window is back

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

positive

- CPV reporter - being part of the solution
- CPV explorer - data accessibility
- case: CRM crash detection
- ongoing: webapp monitoring
- structured problem solving
- closed feedback loop
- SLA benchmarks

challenge

- CPV drama triangle - victim / rescuer
- who is begin observed
- mapping the human ways
- side effects
- high observability assumptions

challenge

- CPV drama triangle - victim / rescuer
- who is begin observed
- mapping the human ways
- side effects
- high observability assumptions

challenge

- CPV drama triangle - victim / rescuer
- who is begin observed
- mapping the human ways
- side effects
- high observability assumptions

challenge

- CPV drama triangle - victim / rescuer
- who is begin observed
- mapping the human ways
- side effects
- high observability assumptions

challenge

- CPV drama triangle - victim / rescuer
- who is begin observed
- mapping the human ways
- side effects
- high observability assumptions

future work

- DLL injection
- webapps, webapps, webapps
- dealing with the data

future work

- DLL injection
- webapps, webapps, webapps
- dealing with the data

future work

- DLL injection
- webapps, webapps, webapps
- dealing with the data

Questions

Tobi Oetiker <tobi@oetiker.ch>
OETIKER+PARTNER AG

Commercial Contact:
Claus Henning Simon <ClausHenning.Simon@swisscom.com>
Swisscom IT Services AG