WADS 2009

On the Design of Adaptive-and-dependable Systems

Lessons learned and experiences at the University of Antwerp

Vincenzo De Florio

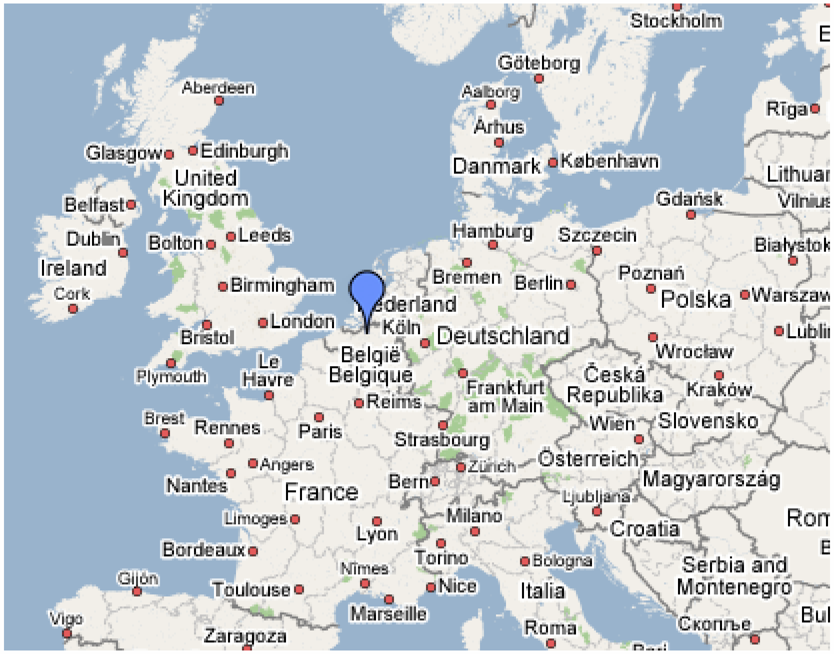
http://www.pats.ua.ac.be/vincenzo.deflorio

Agenda

- Adaptive-and-Dependable Software Systems
  - Where
  - What, Why, How
- How: @ UA
  - Memory-based metaphor
- Conclusions

Introduction – ADSS: Where

- UA, University of Antwerp, Belgium
  - Approximately 10.000 students, third largest in Flanders
- Quite young university
  - 2003, merge of three smaller universities
  - roots go back to 1852



- Seven Faculties, including Sciences
  - Dept. of Computer Science and Mathematics

UA ⇒ PATS

Performance Analysis of Telecommunication Systems

Research Group

Welcome

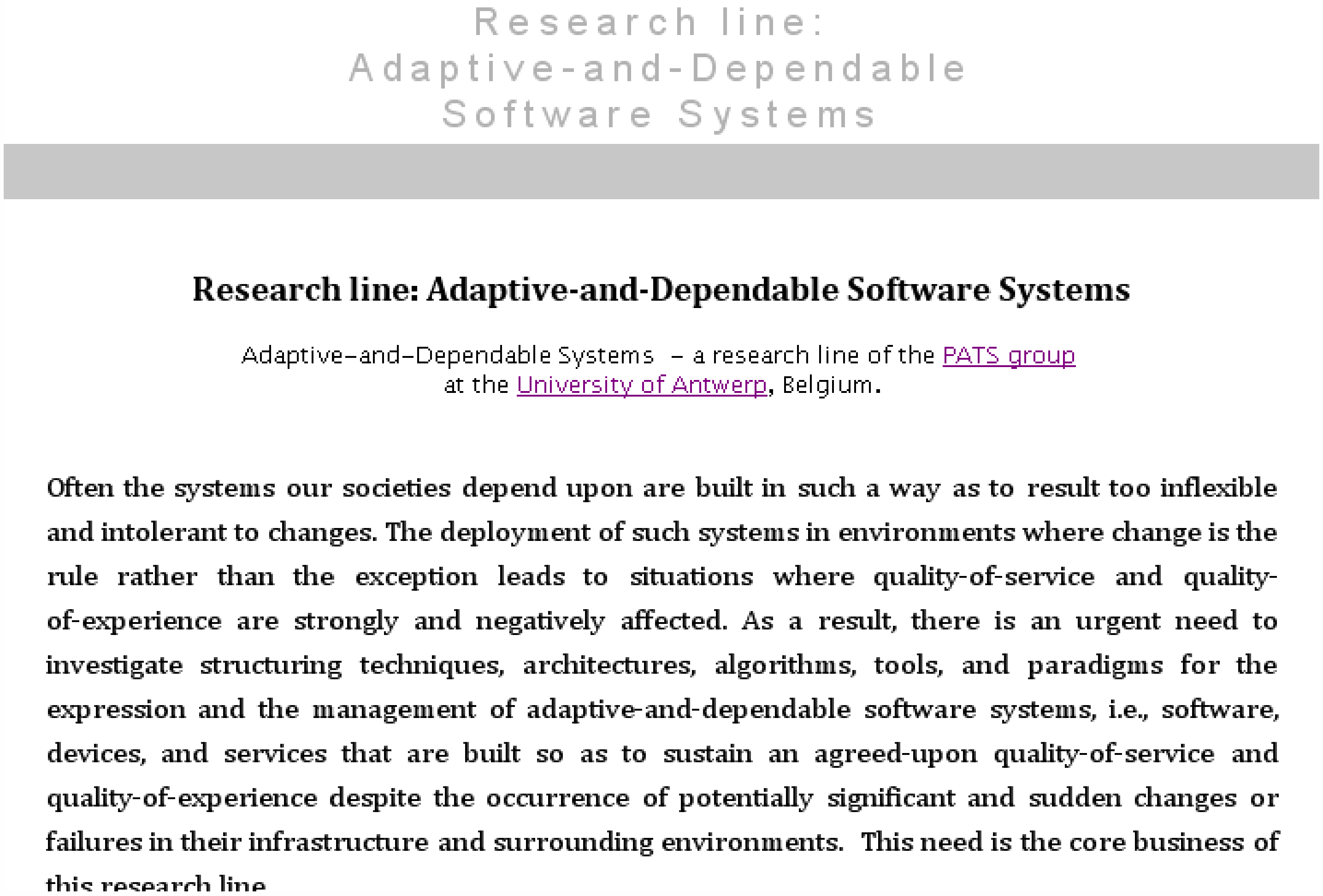
The **Performance Analysis of Telecommunication Systems (PATS) research group** is part of the Department of Mathematics and Computer Science of the University of Antwerp. The group was founded in 1995. PATS performs basic, applied and contract research related to the performance analysis of telecommunication systems and the impact of performance on the architecture and the design of these systems.

The PATS research group is one of the groups that are involved in the Interdisciplinary Institute for Broadband Technology (IBBT) which was founded by the Flemish government on March 19th of 2004.

(Read more ...)

www.pats.ua.ac.be

UA ⇒ PATS ⇒ ADSS



Research line: Adaptive-and-Dependable Software Systems

Research line: Adaptive-and-Dependable Software Systems

Adaptive–and–Dependable Systems – a research line of the PATS group
at the University of Antwerp, Belgium.

Often the systems our societies depend upon are built in such a way as to result too inflexible and intolerant to changes. The deployment of such systems in environments where change is the rule rather than the exception leads to situations where quality-of-service and quality-of-experience are strongly and negatively affected. As a result, there is an urgent need to investigate structuring techniques, architectures, algorithms, tools, and paradigms for the expression and the management of adaptive-and-dependable software systems, i.e., software, devices, and services that are built so as to sustain an agreed-upon quality-of-service and quality-of-experience despite the occurrence of potentially significant and sudden changes or failures in their infrastructure and surrounding environments. This need is the core business of this research line

ADSS: What?

- OK, but what are « Adaptive-and-dependable sw systems »?
- Let me answer by recalling first what Real-Time Software (RTS) is:
  - "Real-time software is software that interacts with the world on the world's schedule, not the software's.
  - It *senses the world* and *responds to changes* in the world when those changes occur."

ADSS: What?

- RTS = an entity that executes in a «virtual world,» but monitors and synchronizes with the physical world – what time is concerned
- RTS = organized and built so as to keep track of *the timing* of physical world's events and do as much as possible to avoid *timing failures*
- An ADSS is something similar

ADSS: What?

- ADSS may be considered as a *generalisation* of RTS:
- It is organized and built so as to keep track of ~~(the timing of)~~ physical world's events and do as much as possible to avoid ~~(timing)~~ failures
  - QoS failures, QoE failures
- Both RTS and ADSS: Open world assumption

ADSS: What

- Thus ADSS is “software that is built so as to sustain an agreed-upon quality-of-service and quality-of-experience despite the occurrence of potentially significant and sudden changes or failures in their infrastructure and surrounding environments.”

- ADSS: Why

ADSS: Why?

- Worst-case analyses do not pay off anymore!
  - Truly effective approaches forbid upper bounds; instead, they require a precise characterization of the allocation of resources over time
  - Unwanted emergent behaviors can only be avoided if the systems are built with “a finer-grain control of the redundancy degree” (Esposito and Cotroneo, 2009)
    and of the other available resources

ADSS: Why?

- Worst-case analyses do not pay off anymore (cont.'ed)
  - WCA = no optimal way to choose the amount of redundancy
    - « What is the *minimal* redundancy matching the *current environmental conditions* (threats / disturbances…)? »

→ Close world solutions are inefficient

ADSS: Why?

- Hidden intelligence syndrome!
  - A dependable system is built atop several assumptions or hypotheses
    - Explicit or implicit ones
  - Those are «contracts» that must not be ignored, lest dependencies turn into failures

ADSS: Why?

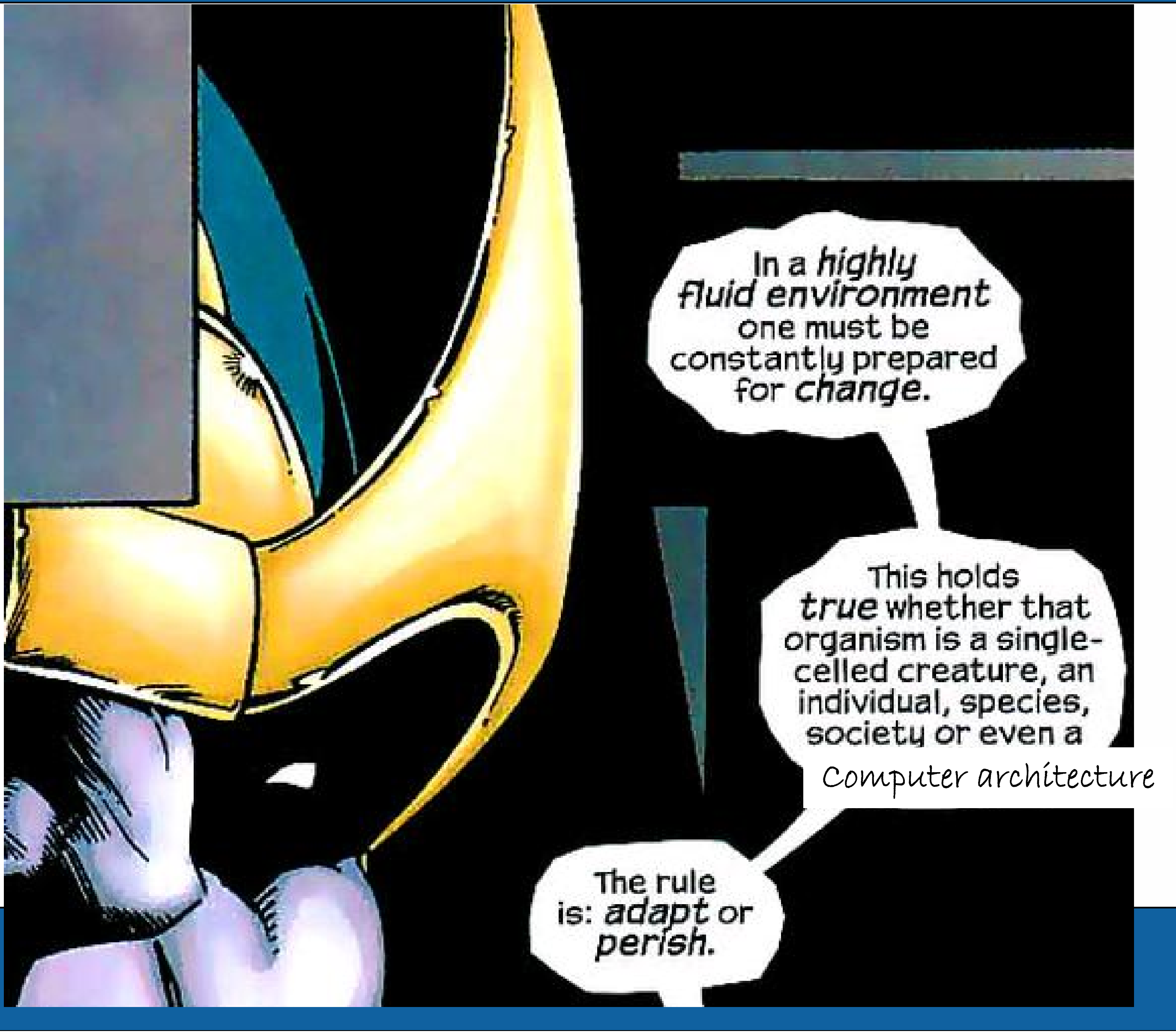
- Hidden intelligence syndrome (cont.'ed)
  - A few examples
    - «HW includes a MMU» ⇒ memory errors may be detected
    - «Memory technology is SDRAM» ⇒ memory fails through single-event effects (instead of bitflips)
    - «The platform includes hardware interlocks» ⇒ any malfunction shuts down the system
    - «Reasonable amount of redundancy is 3 replicas» ⇒ single failure assumption

ADSS: Why?

- Hidden intelligence syndrome (cont.'ed)
  - HIS calls for ways to express & evaluate assumptions such as those
  - The fault model, the system model, the platform dependencies should be expressable and verifiable
  - Software reuse, porting, re-deployment, call for re-evaluation and re-organization

→ *Necessary services of any truly dependable architecture: ADSS!*

ADSS: Why?


In a *highly fluid environment* one must be constantly prepared for *change.*
This holds *true* whether that organism is a single-celled creature, an individual, species, society or even a
Computer architecture
The rule is: *adapt* or *perish.*

ADSS: Why?

- Indeed we're living in «highly fluid environments»!
  - "Large, networked and evolving systems either fixed or mobile, with demanding requirements driven by their application domain"
  - "Complex, ever changing, ubiquitous and pervasive systems" (Simoncini, 2009)
- Those are the systems that suffer most from the Horning syndrome
  - "What is the most often overlooked risk in software engineering?
    That the environment will do something the designer never anticipated" [J. Horning]

ADSS: Why?

- Ultra large-scale systems!
  - A shift from “small, monolithic and vertical architectures [..] toward large highly modular, autonomous, heterogeneous and integrated systems of systems” (Esposito & Cotroneo, 2009)
    - Large scale Complex Critical Infrastructures : based on best-effort WANs, though both reliable and timely!

→ Require adaptive-and-dependable sw architectures

ADSS: Why

- The only possible assumption is the open-world one
- “The assumption that the system software architecture is known and fixed at an early stage of system development does not apply anymore. On the contrary the ubiquitous scenario promotes the view that systems can be dynamically composed out of available components”
- “In this setting the software architecture can only be dynamically induced” (Inverardi, today!)

- ADSS: How

ADSS: How?

- Not a single research direction
- ADSS@UA/PATS :
  - ACCADA, A Continuous Context-Aware Deployment and Adaptation framework on top of OSGi (Ning Gui)
  - SoA+AOP framework (OSGi/Equinox) (Hong Sun)
  - Apache Muse/Axis2 framework (Jonas Buys)
  - Reflective C
    - Adaptive data structures…

Reflective C

- Reflective & refractive variables (RR vars)
- Redundant variables
- Meta variables

RR vars

- Main idea: memory accesses as a metaphor for detecting changes and reacting from changes
- An abstraction to realize open-world software
- RR vars = volatile variables whose identifier links them with an external device, e.g. a sensor, or an RFID, or an actuator

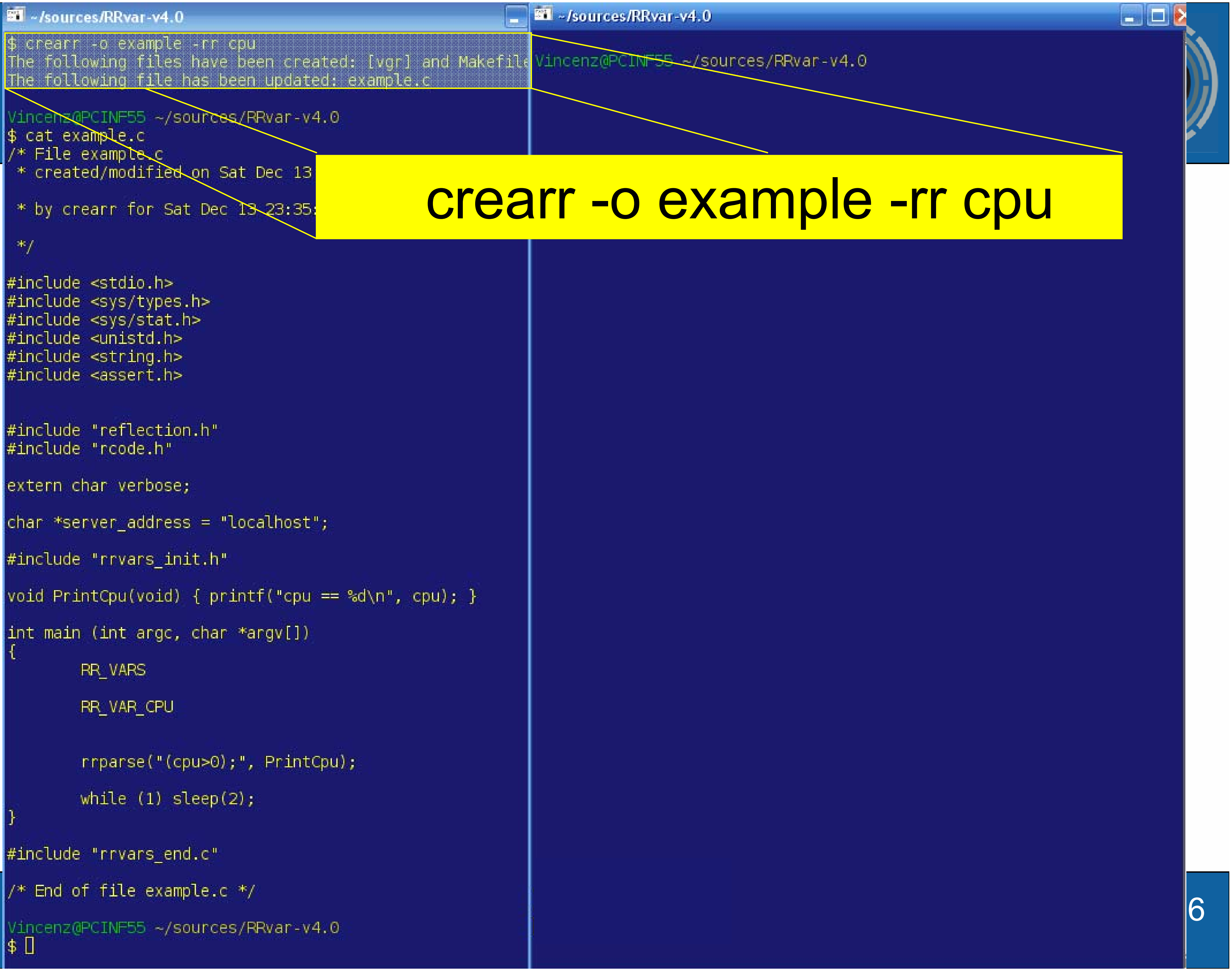
RR vars

- Reflective variables: memory cells get asynchronously updated by probes
  - Probes: service threads interfacing external devices
- Refractive variables: Write accesses trigger a request to perform some action
  - E.g. set frame dropping policy of a media player or amount of redundancy to be employed
  - Write accesses *refract* (that is, get redirected) onto corresponding external devices

RR vars

- An hello world application can be built via program crearr
- This creates a “hello world” code that uses reflective variable cpu:

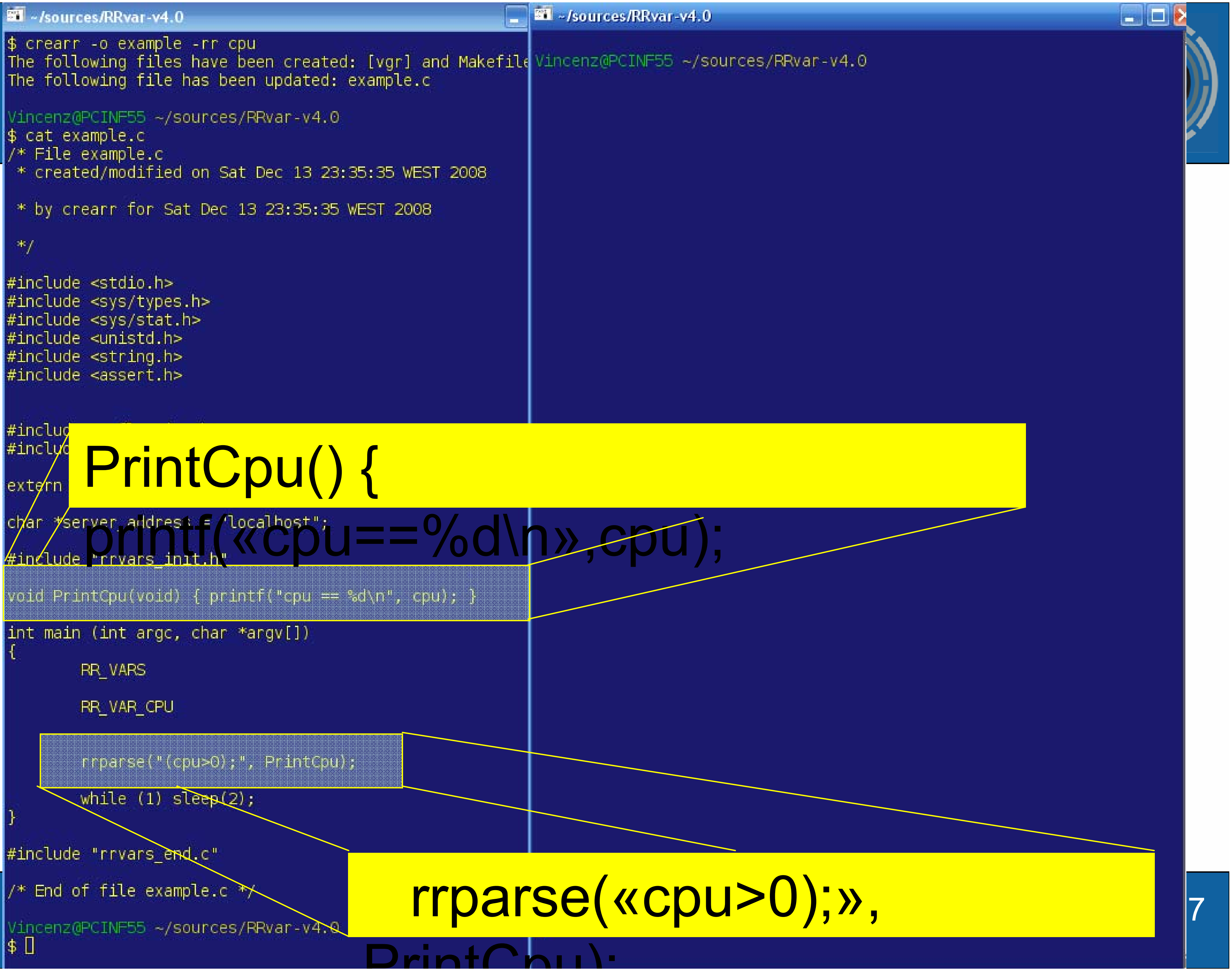
```
crearr -o example -rr cpu
```



```
$ crearr -o example -rr cpu
The following files have been created: [vgr] and Makefile
The following file has been updated: example.c

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ cat example.c
/* File example.c
 * created/modified on Sat Dec 13

 * by crearr for Sat Dec 13 23:35:

 */

#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>
#include <assert.h>


#include "reflection.h"
#include "rcode.h"

extern char verbose;

char *server_address = "localhost";

#include "rrvars_init.h"

void PrintCpu(void) { printf("cpu == %d\n", cpu); }

int main (int argc, char *argv[])
{
        RR_VARS

        RR_VAR_CPU


        rrparse("(cpu>0);", PrintCpu);

        while (1) sleep(2);
}

#include "rrvars_end.c"

/* End of file example.c */

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ 
```

```
Vincenz@PCINF55 ~/sources/RRvar-v4.0
```

crearr -o example -rr cpu

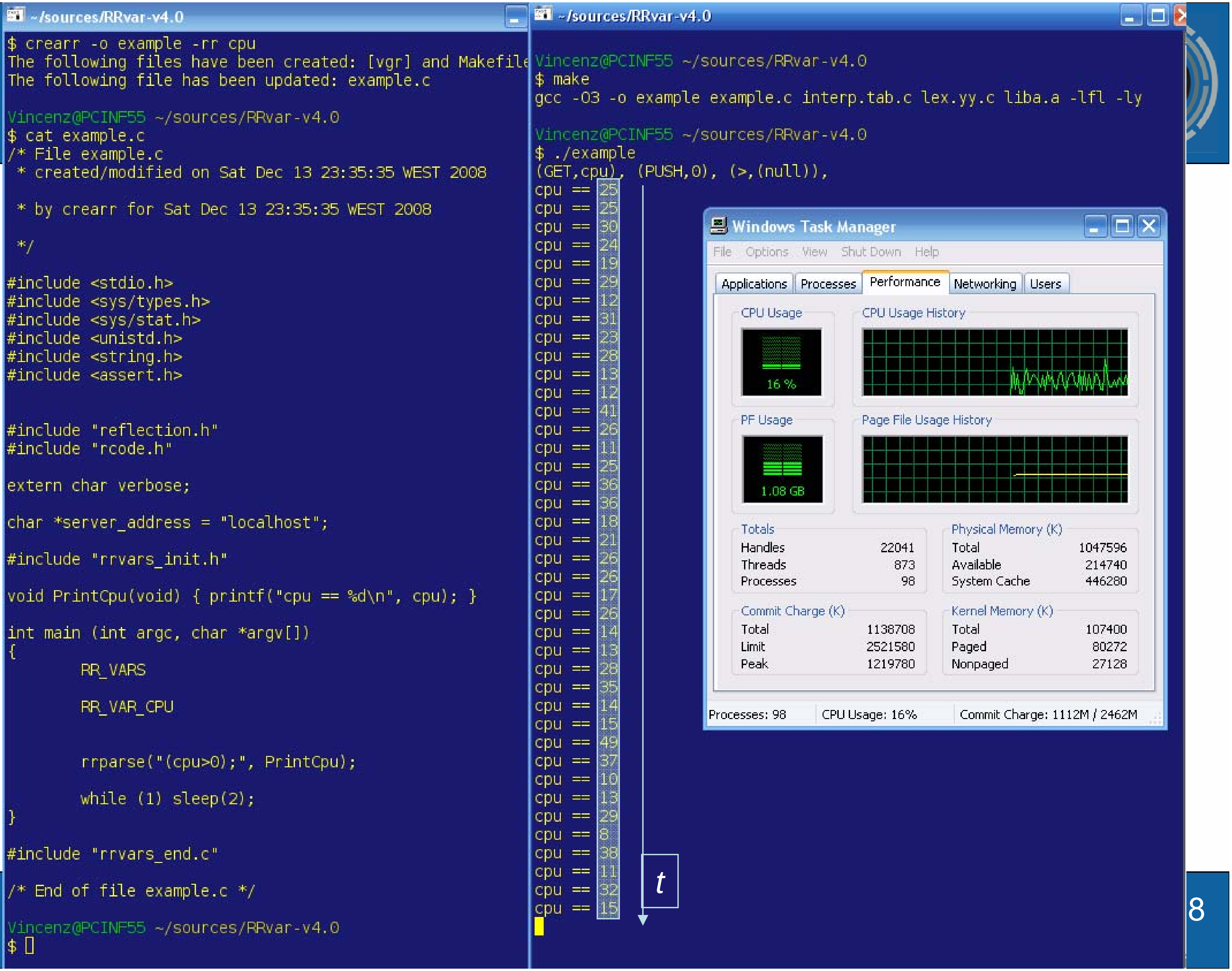


```
$ crearr -o example -rr cpu
The following files have been created: [vgr] and Makefile
The following file has been updated: example.c

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ cat example.c
/* File example.c
 * created/modified on Sat Dec 13 23:35:35 WEST 2008

 * by crearr for Sat Dec 13 23:35:35 WEST 2008

 */

#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>
#include <assert.h>


#inclu
#inclu

extern

char *server_address = "localhost";

#include "rrvars_init.h"

void PrintCpu(void) { printf("cpu == %d\n", cpu); }

int main (int argc, char *argv[])
{
        RR_VARS

        RR_VAR_CPU


        rrparse("(cpu>0);", PrintCpu);

        while (1) sleep(2);
}

#include "rrvars_end.c"

/* End of file example.c */

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ 
```

```
Vincenz@PCINF55 ~/sources/RRvar-v4.0
```

PrintCpu() {
printf(«cpu==%d\n»,cpu);

rrparse(«cpu>0);»,
PrintCpu);



~/sources/RRvar-v4.0

```
$ crearr -o example -rr cpu
The following files have been created: [vgr] and Makefile
The following file has been updated: example.c

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ cat example.c
/* File example.c
 * created/modified on Sat Dec 13 23:35:35 WEST 2008

 * by crearr for Sat Dec 13 23:35:35 WEST 2008

 */

#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>
#include <assert.h>


#include "reflection.h"
#include "rcode.h"

extern char verbose;

char *server_address = "localhost";

#include "rrvars_init.h"

void PrintCpu(void) { printf("cpu == %d\n", cpu); }

int main (int argc, char *argv[])
{
        RR_VARS

        RR_VAR_CPU


        rrparse("(cpu>0);", PrintCpu);

        while (1) sleep(2);
}

#include "rrvars_end.c"

/* End of file example.c */

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ 
```

~/sources/RRvar-v4.0

```
Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ make
gcc -O3 -o example example.c interp.tab.c lex.yy.c liba.a -lfl -ly

Vincenz@PCINF55 ~/sources/RRvar-v4.0
$ ./example
(GET,cpu), (PUSH,0), (>,(null)),
cpu == 25
cpu == 25
cpu == 30
cpu == 24
cpu == 19
cpu == 29
cpu == 12
cpu == 31
cpu == 23
cpu == 28
cpu == 13
cpu == 12
cpu == 41
cpu == 26
cpu == 11
cpu == 25
cpu == 36
cpu == 36
cpu == 18
cpu == 21
cpu == 26
cpu == 26
cpu == 17
cpu == 26
cpu == 14
cpu == 13
cpu == 28
cpu == 35
cpu == 14
cpu == 15
cpu == 49
cpu == 37
cpu == 10
cpu == 13
cpu == 29
cpu == 8
cpu == 38
cpu == 11
cpu == 32
cpu == 15
```

t

Totals
Handles 22041
Threads 873
Processes 98

Physical Memory (K)
Total 1047596
Available 214740
System Cache 446280

Commit Charge (K)
Total 1138708
Limit 2521580
Peak 1219780

Kernel Memory (K)
Total 107400
Paged 80272
Nonpaged 27128

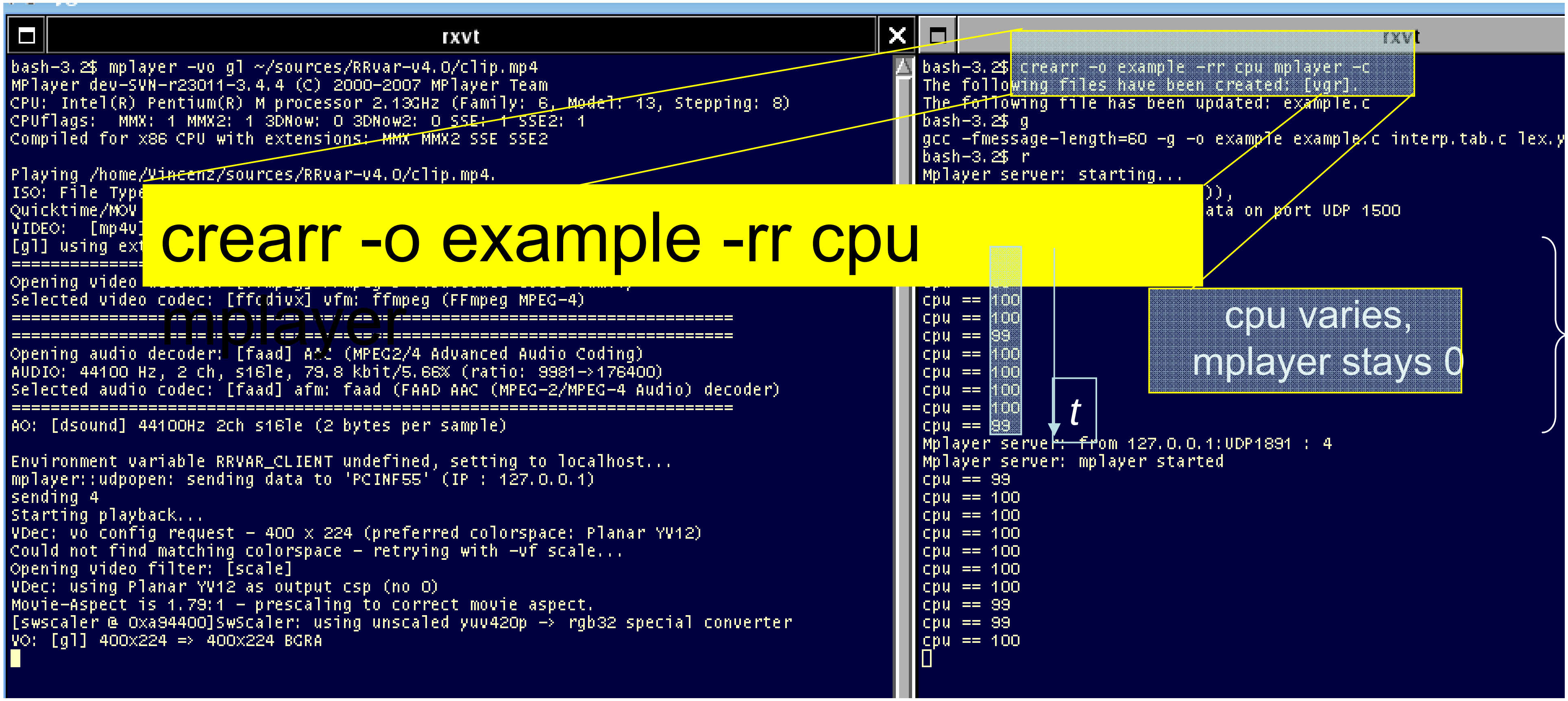
Processes: 98 | CPU Usage: 16% | Commit Charge: 1112M / 2462M

RR vars

- Callbacks through function rrparse.
- When a guard is evaluated as true, the callback is executed
- Default guard is trivial: amount of CPU > 0
- Default callback: print current amount of CPU
    - “Similar” behavior:

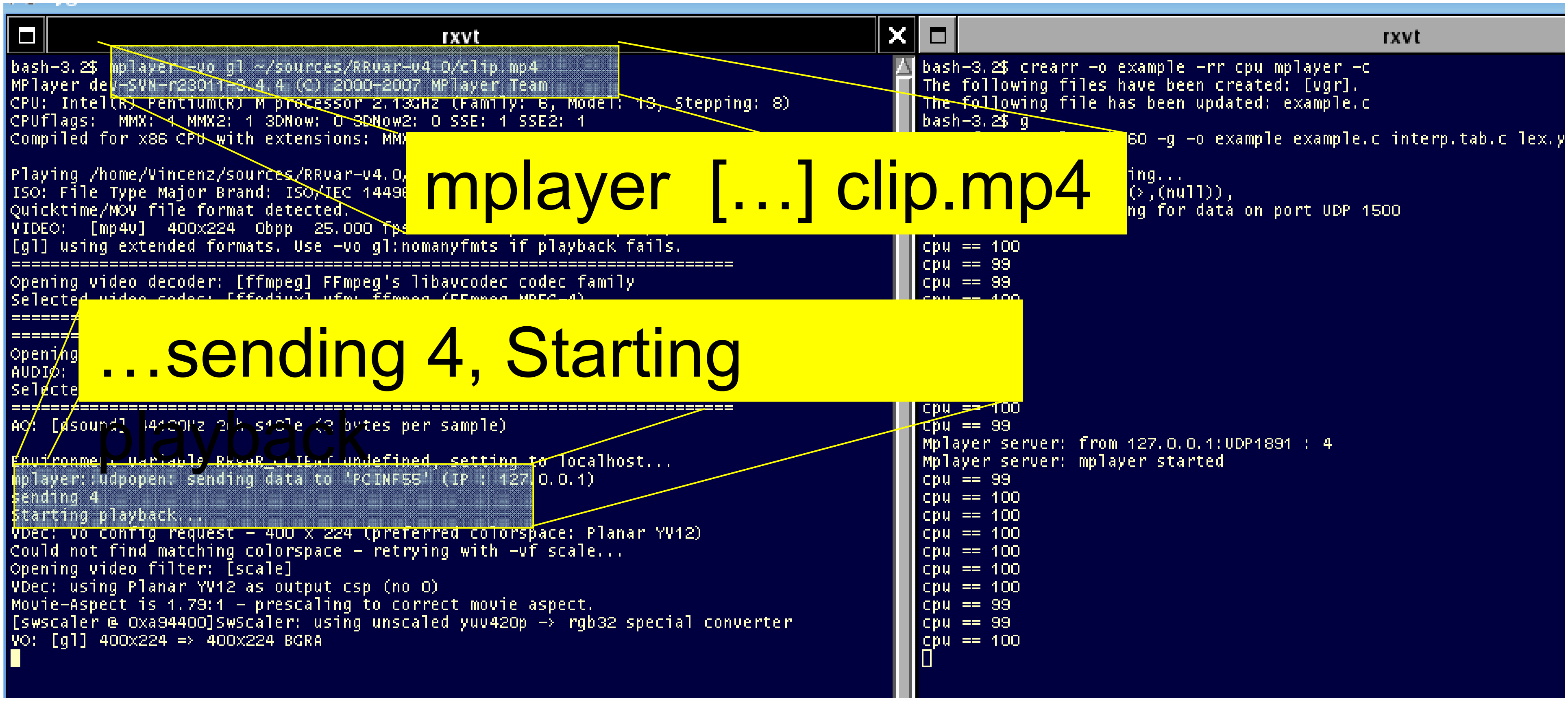
```
while (1) { if (cpu > 0) Callback(); }.
```

- Another example:



```
bash-3.2$ mplayer -vo gl ~/sources/RRvar-v4.0/clip.mp4
MPlayer dev-SVN-r23011-3.4.4 (C) 2000-2007 MPlayer Team
CPU: Intel(R) Pentium(R) M processor 2.13GHz (Family: 6, Model: 13, Stepping: 8)
CPUflags:  MMX: 1 MMX2: 1 3DNow: 0 3DNow2: 0 SSE: 1 SSE2: 1
Compiled for x86 CPU with extensions: MMX MMX2 SSE SSE2

Playing /home/Vincenz/sources/RRvar-v4.0/clip.mp4.
ISO: File Type
Quicktime/MOV
VIDEO:  [mp4v]
[gl] using ext
=============
Opening video
Selected video codec: [ffodivx] vfm: ffmpeg (FFmpeg MPEG-4)
=====================================================================
=====================================================================
Opening audio decoder: [faad] AAC (MPEG2/4 Advanced Audio Coding)
AUDIO: 44100 Hz, 2 ch, s16le, 79.8 kbit/5.66% (ratio: 9981->176400)
Selected audio codec: [faad] afm: faad (FAAD AAC (MPEG-2/MPEG-4 Audio) decoder)
=====================================================================
AO: [dsound] 44100Hz 2ch s16le (2 bytes per sample)

Environment variable RRVAR_CLIENT undefined, setting to localhost...
mplayer::udpopen: sending data to 'PCINF55' (IP : 127.0.0.1)
sending 4
Starting playback...
VDec: vo config request - 400 x 224 (preferred colorspace: Planar YV12)
Could not find matching colorspace - retrying with -vf scale...
Opening video filter: [scale]
VDec: using Planar YV12 as output csp (no 0)
Movie-Aspect is 1.79:1 - prescaling to correct movie aspect.
[swscaler @ 0xa94400]SwScaler: using unscaled yuv420p -> rgb32 special converter
VO: [gl] 400x224 => 400x224 BGRA
```

```
bash-3.2$ crearr -o example -rr cpu mplayer -c
The following files have been created: [vgr].
The following file has been updated: example.c
bash-3.2$ g
gcc -fmessage-length=60 -g -o example example.c interp.tab.c lex.y
bash-3.2$ r
Mplayer server: starting...
                          )),
                          ata on port UDP 1500



cpu == 100
cpu == 100
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 4
Mplayer server: mplayer started
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 99
cpu == 100
```

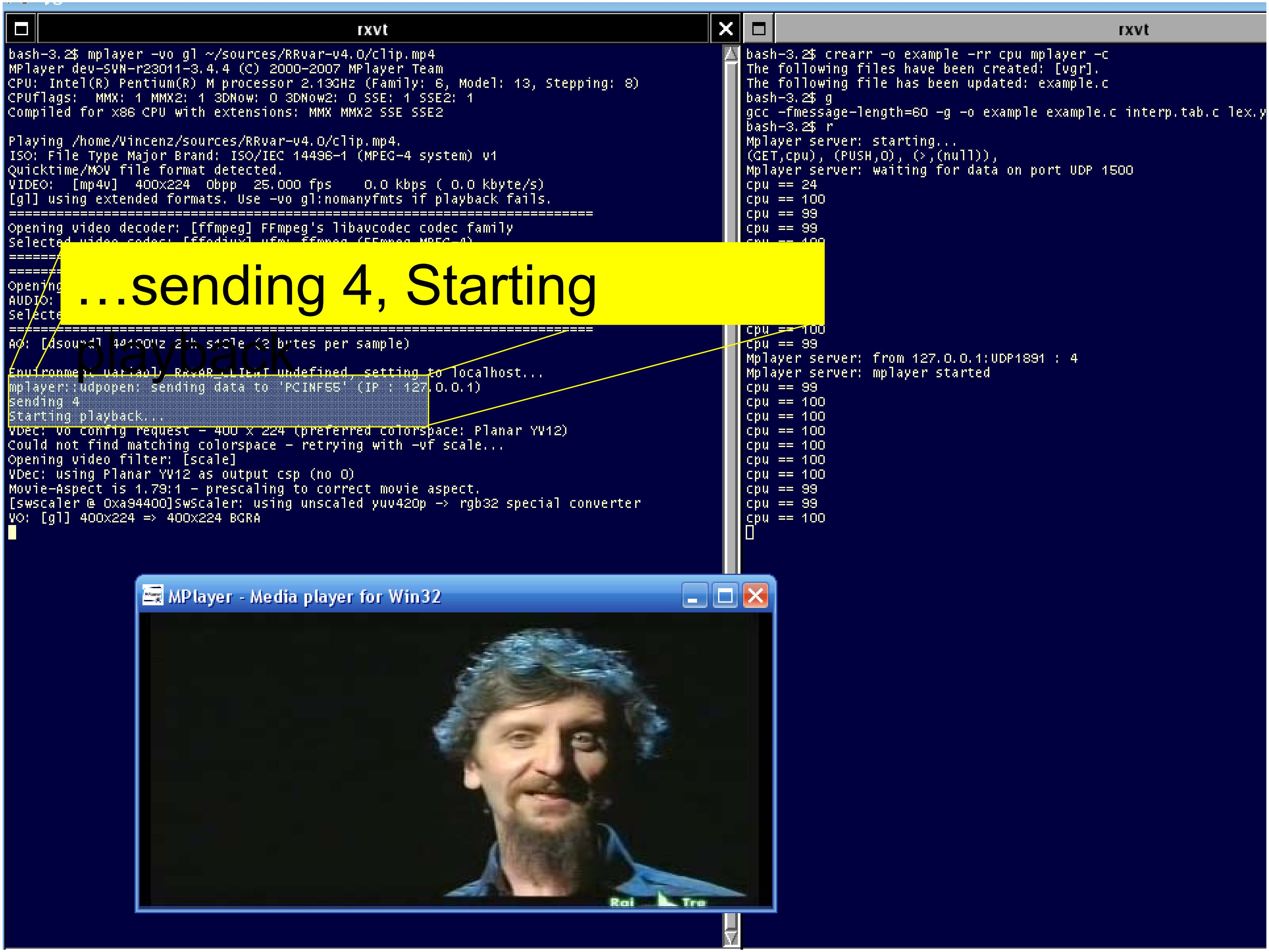


```
bash-3.2$ mplayer -vo gl ~/sources/RRvar-v4.0/clip.mp4
MPlayer dev-SVN-r23011-3.4.4 (C) 2000-2007 MPlayer Team
CPU: Intel(R) Pentium(R) M processor 2.13GHz (Family: 6, Model: 13, Stepping: 8)
CPUflags:  MMX: 1 MMX2: 1 3DNow: 0 3DNow2: 0 SSE: 1 SSE2: 1
Compiled for x86 CPU with extensions: MMX

Playing /home/Vincenz/sources/RRvar-v4.0/
ISO: File Type Major Brand: ISO/IEC 1449
Quicktime/MOV file format detected.
VIDEO:  [mp4v]  400x224  0bpp  25.000 fps
[gl] using extended formats. Use -vo gl:nomanyfmts if playback fails.
==========================================================================
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
Selected video codec: [ffodivx] vfm: ffmpeg (FFmpeg MPEG-4)
======
======
Opening
AUDIO:
Selecte
==========================================================================
AO: [dsound] 44100Hz 2ch s16le (2 bytes per sample)

Environment variable RRVAR_CLIENT undefined, setting to localhost...
mplayer::udpopen: sending data to 'PCINF55' (IP : 127.0.0.1)
sending 4
Starting playback...
VDec: vo config request - 400 x 224 (preferred colorspace: Planar YV12)
Could not find matching colorspace - retrying with -vf scale...
Opening video filter: [scale]
VDec: using Planar YV12 as output csp (no 0)
Movie-Aspect is 1.79:1 - prescaling to correct movie aspect.
[swscaler @ 0xa94400]SwScaler: using unscaled yuv420p -> rgb32 special converter
VO: [gl] 400x224 => 400x224 BGRA
```

```
bash-3.2$ crearr -o example -rr cpu mplayer -c
The following files have been created: [vgr].
The following file has been updated: example.c
bash-3.2$ g
                 60 -g -o example example.c interp.tab.c lex.y

                 ing...
                 (>,(null)),
                 ng for data on port UDP 1500

cpu == 100
cpu == 99
cpu == 99
cpu == 100







cpu == 100
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 4
Mplayer server: mplayer started
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 99
cpu == 100
```

mplayer [...] clip.mp4

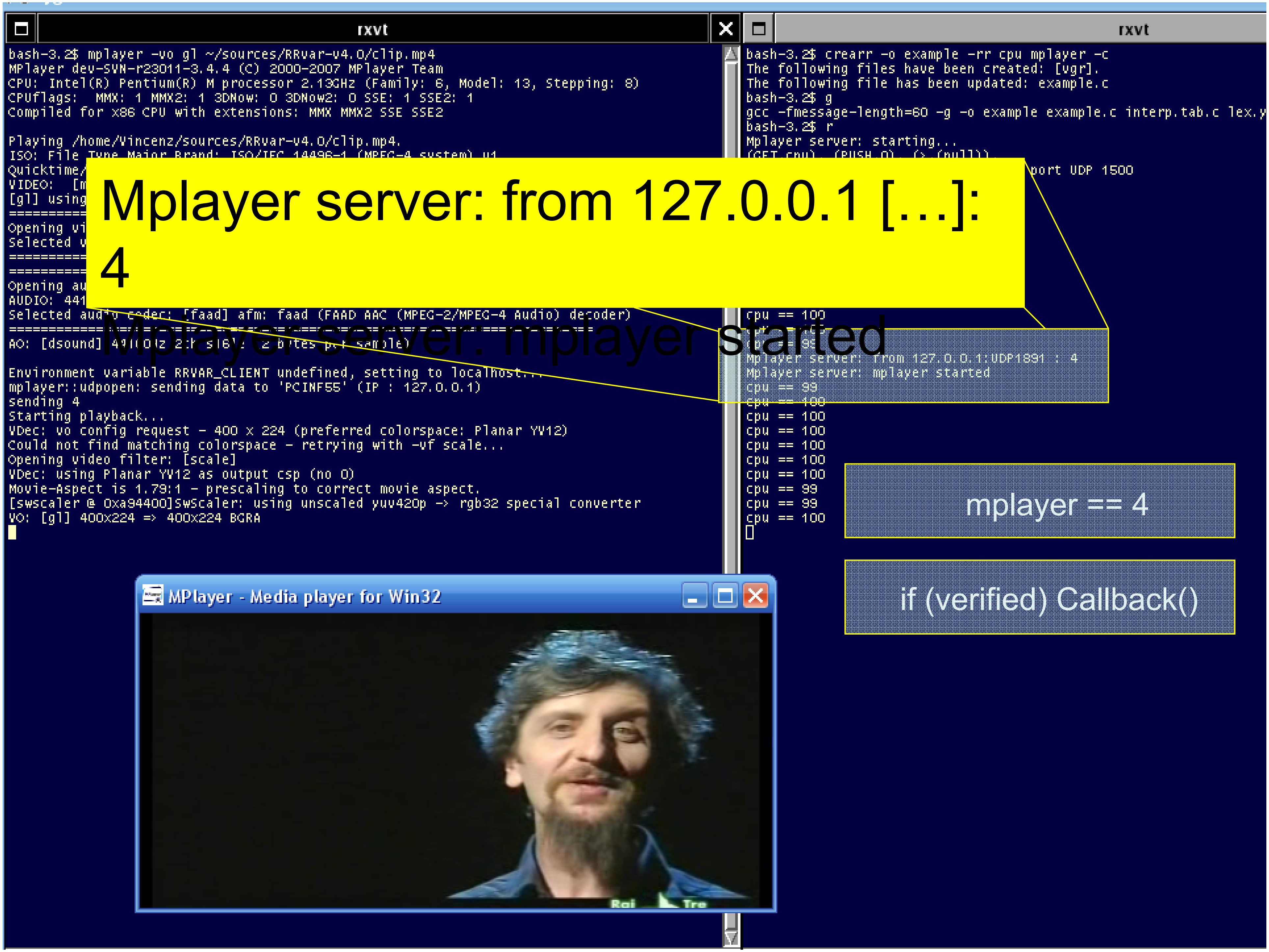
...sending 4, Starting playback



…sending 4, Starting playback

```
bash-3.2$ mplayer -vo gl ~/sources/RRvar-v4.0/clip.mp4
MPlayer dev-SVN-r23011-3.4.4 (C) 2000-2007 MPlayer Team
CPU: Intel(R) Pentium(R) M processor 2.13GHz (Family: 6, Model: 13, Stepping: 8)
CPUflags:  MMX: 1 MMX2: 1 3DNow: 0 3DNow2: 0 SSE: 1 SSE2: 1
Compiled for x86 CPU with extensions: MMX MMX2 SSE SSE2

Playing /home/Vincenz/sources/RRvar-v4.0/clip.mp4.
ISO: File Type Major Brand: ISO/IEC 14496-1 (MPEG-4 system) v1
Quicktime/MOV file format detected.
VIDEO:  [mp4v]  400x224  0bpp  25.000 fps    0.0 kbps ( 0.0 kbyte/s)
[gl] using extended formats. Use -vo gl:nomanyfmts if playback fails.
==========================================================================
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
Selected video codec: [ffodivx] vfm: ffmpeg (FFmpeg MPEG-4)
======
======
Opening
AUDIO:
Selecte
==========================================================================
AO: [dsound] 44100Hz 2ch s16le (2 bytes per sample)

Environment variable RRVAR_CLIENT undefined, setting to localhost...
mplayer::udpopen: sending data to 'PCINF55' (IP : 127.0.0.1)
sending 4
Starting playback...
VDec: vo config request - 400 x 224 (preferred colorspace: Planar YV12)
Could not find matching colorspace - retrying with -vf scale...
Opening video filter: [scale]
VDec: using Planar YV12 as output csp (no 0)
Movie-Aspect is 1.79:1 - prescaling to correct movie aspect.
[swscaler @ 0xa94400]SwScaler: using unscaled yuv420p -> rgb32 special converter
VO: [gl] 400x224 => 400x224 BGRA
```

```
bash-3.2$ crearr -o example -rr cpu mplayer -c
The following files have been created: [vgr].
The following file has been updated: example.c
bash-3.2$ g
gcc -fmessage-length=60 -g -o example example.c interp.tab.c lex.y
bash-3.2$ r
Mplayer server: starting...
(GET,cpu), (PUSH,0), (>,(null)),
Mplayer server: waiting for data on port UDP 1500
cpu == 24
cpu == 100
cpu == 99
cpu == 99
cpu == 100
cpu == 100
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 4
Mplayer server: mplayer started
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 99
cpu == 100
```



```
bash-3.2$ mplayer -vo gl ~/sources/RRvar-v4.0/clip.mp4
MPlayer dev-SVN-r23011-3.4.4 (C) 2000-2007 MPlayer Team
CPU: Intel(R) Pentium(R) M processor 2.13GHz (Family: 6, Model: 13, Stepping: 8)
CPUflags:  MMX: 1 MMX2: 1 3DNow: 0 3DNow2: 0 SSE: 1 SSE2: 1
Compiled for x86 CPU with extensions: MMX MMX2 SSE SSE2

Playing /home/Vincenz/sources/RRvar-v4.0/clip.mp4.
ISO: File Type Major Brand: ISO/IEC 14496-1 (MPEG-4 system) v1
Quicktime/
VIDEO:  [m
[gl] using
==========
Opening vi
Selected v
==========
==========
Opening au
AUDIO: 441
Selected audio codec: [faad] afm: faad (FAAD AAC (MPEG-2/MPEG-4 Audio) decoder)
==========================================================================
AO: [dsound] 44100Hz 2ch s16le (2 bytes per sample)

Environment variable RRVAR_CLIENT undefined, setting to localhost...
mplayer::udpopen: sending data to 'PCINF55' (IP : 127.0.0.1)
sending 4
Starting playback...
VDec: vo config request - 400 x 224 (preferred colorspace: Planar YV12)
Could not find matching colorspace - retrying with -vf scale...
Opening video filter: [scale]
VDec: using Planar YV12 as output csp (no 0)
Movie-Aspect is 1.79:1 - prescaling to correct movie aspect.
[swscaler @ 0xa94400]SwScaler: using unscaled yuv420p -> rgb32 special converter
VO: [gl] 400x224 => 400x224 BGRA
```

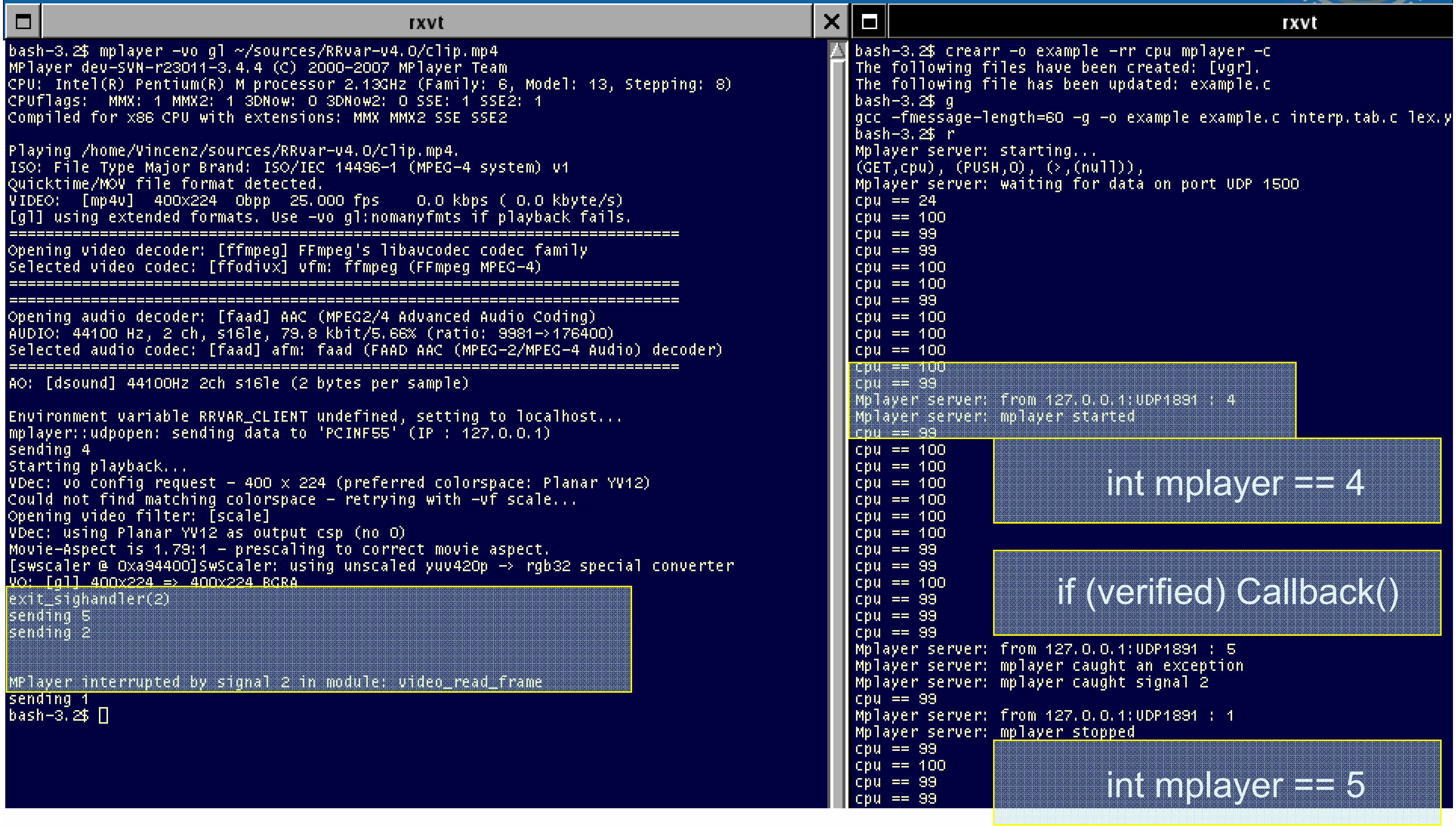
```
bash-3.2$ crearr -o example -rr cpu mplayer -c
The following files have been created: [vgr].
The following file has been updated: example.c
bash-3.2$ g
gcc -fmessage-length=60 -g -o example example.c interp.tab.c lex.y
bash-3.2$ r
Mplayer server: starting...
(GET,cpu), (PUSH,0), (>,(null)),
                                               port UDP 1500
cpu == 100
cpu == 100
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 4
Mplayer server: mplayer started
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 99
cpu == 100
```

Mplayer server: from 127.0.0.1 […]: 4

Mplayer server: mplayer started

mplayer == 4

if (verified) Callback()



```
bash-3.2$ mplayer -vo gl ~/sources/RRvar-v4.0/clip.mp4
MPlayer dev-SVN-r23011-3.4.4 (C) 2000-2007 MPlayer Team
CPU: Intel(R) Pentium(R) M processor 2.13GHz (Family: 6, Model: 13, Stepping: 8)
CPUflags:  MMX: 1 MMX2: 1 3DNow: 0 3DNow2: 0 SSE: 1 SSE2: 1
Compiled for x86 CPU with extensions: MMX MMX2 SSE SSE2

Playing /home/Vincenz/sources/RRvar-v4.0/clip.mp4.
ISO: File Type Major Brand: ISO/IEC 14496-1 (MPEG-4 system) v1
Quicktime/MOV file format detected.
VIDEO:  [mp4v]  400x224  0bpp  25.000 fps    0.0 kbps ( 0.0 kbyte/s)
[gl] using extended formats. Use -vo gl:nomanyfmts if playback fails.
==========================================================================
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
Selected video codec: [ffodivx] vfm: ffmpeg (FFmpeg MPEG-4)
==========================================================================
==========================================================================
Opening audio decoder: [faad] AAC (MPEG2/4 Advanced Audio Coding)
AUDIO: 44100 Hz, 2 ch, s16le, 79.8 kbit/5.66% (ratio: 9981->176400)
Selected audio codec: [faad] afm: faad (FAAD AAC (MPEG-2/MPEG-4 Audio) decoder)
==========================================================================
AO: [dsound] 44100Hz 2ch s16le (2 bytes per sample)

Environment variable RRVAR_CLIENT undefined, setting to localhost...
mplayer::udpopen: sending data to 'PCINF55' (IP : 127.0.0.1)
sending 4
Starting playback...
VDec: vo config request - 400 x 224 (preferred colorspace: Planar YV12)
Could not find matching colorspace - retrying with -vf scale...
Opening video filter: [scale]
VDec: using Planar YV12 as output csp (no 0)
Movie-Aspect is 1.79:1 - prescaling to correct movie aspect.
[swscaler @ 0xa94400]SwScaler: using unscaled yuv420p -> rgb32 special converter
VO: [gl] 400x224 => 400x224 BGRA
exit_sighandler(2)
sending 5
sending 2


MPlayer interrupted by signal 2 in module: video_read_frame
sending 1
bash-3.2$ 
```

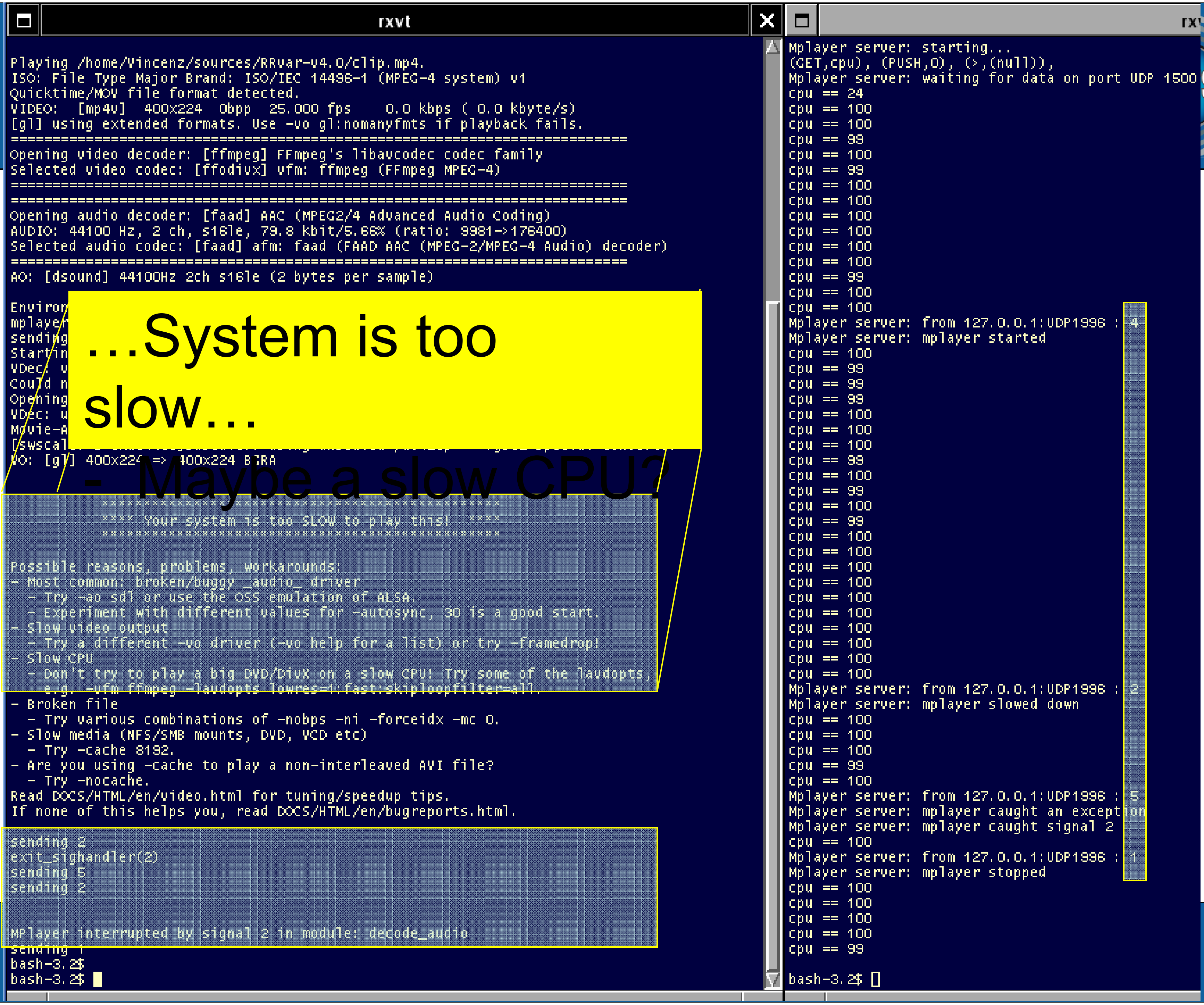
```
bash-3.2$ crearr -o example -rr cpu mplayer -c
The following files have been created: [vgr].
The following file has been updated: example.c
bash-3.2$ g
gcc -fmessage-length=60 -g -o example example.c interp.tab.c lex.y
bash-3.2$ r
Mplayer server: starting...
(GET,cpu), (PUSH,0), (>,(null)),
Mplayer server: waiting for data on port UDP 1500
cpu == 24
cpu == 100
cpu == 99
cpu == 99
cpu == 100
cpu == 100
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 4
Mplayer server: mplayer started
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 99
cpu == 100
cpu == 99
cpu == 99
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 5
Mplayer server: mplayer caught an exception
Mplayer server: mplayer caught signal 2
cpu == 99
Mplayer server: from 127.0.0.1:UDP1891 : 1
Mplayer server: mplayer stopped
cpu == 99
cpu == 100
cpu == 99
cpu == 99
```

int mplayer == 4

if (verified) Callback()

int mplayer == 5

if (verified) Callback()



```
Playing /home/Vincenz/sources/RRvar-v4.0/clip.mp4.
ISO: File Type Major Brand: ISO/IEC 14496-1 (MPEG-4 system) v1
Quicktime/MOV file format detected.
VIDEO:  [mp4v]  400x224  0bpp  25.000 fps    0.0 kbps ( 0.0 kbyte/s)
[gl] using extended formats. Use -vo gl:nomanyfmts if playback fails.
==========================================================================
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
Selected video codec: [ffodivx] vfm: ffmpeg (FFmpeg MPEG-4)
==========================================================================
==========================================================================
Opening audio decoder: [faad] AAC (MPEG2/4 Advanced Audio Coding)
AUDIO: 44100 Hz, 2 ch, s16le, 79.8 kbit/5.66% (ratio: 9981->176400)
Selected audio codec: [faad] afm: faad (FAAD AAC (MPEG-2/MPEG-4 Audio) decoder)
==========================================================================
AO: [dsound] 44100Hz 2ch s16le (2 bytes per sample)

Environ
mplayer
sending
Startin
VDec: v
Could n
Opening
VDec: u
Movie-A
[swscal
VO: [gl] 400x224 => 400x224 BGRA


          ************************************************
          **** Your system is too SLOW to play this!  ****
          ************************************************

Possible reasons, problems, workarounds:
- Most common: broken/buggy _audio_ driver
  - Try -ao sdl or use the OSS emulation of ALSA.
  - Experiment with different values for -autosync, 30 is a good start.
- Slow video output
  - Try a different -vo driver (-vo help for a list) or try -framedrop!
- Slow CPU
  - Don't try to play a big DVD/DivX on a slow CPU! Try some of the lavdopts,
    e.g. -vfm ffmpeg -lavdopts lowres=1:fast:skiploopfilter=all.
- Broken file
  - Try various combinations of -nobps -ni -forceidx -mc 0.
- Slow media (NFS/SMB mounts, DVD, VCD etc)
  - Try -cache 8192.
- Are you using -cache to play a non-interleaved AVI file?
  - Try -nocache.
Read DOCS/HTML/en/video.html for tuning/speedup tips.
If none of this helps you, read DOCS/HTML/en/bugreports.html.

sending 2
exit_sighandler(2)
sending 5
sending 2


MPlayer interrupted by signal 2 in module: decode_audio
sending 1
bash-3.2$
bash-3.2$ 
```

```
Mplayer server: starting...
(GET,cpu), (PUSH,0), (>,(null)),
Mplayer server: waiting for data on port UDP 1500
cpu == 24
cpu == 100
cpu == 100
cpu == 99
cpu == 100
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 100
cpu == 100
Mplayer server: from 127.0.0.1:UDP1996 : 4
Mplayer server: mplayer started
cpu == 100
cpu == 99
cpu == 99
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 100
cpu == 99
cpu == 100
cpu == 99
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 100
Mplayer server: from 127.0.0.1:UDP1996 : 2
Mplayer server: mplayer slowed down
cpu == 100
cpu == 100
cpu == 100
cpu == 99
cpu == 100
Mplayer server: from 127.0.0.1:UDP1996 : 5
Mplayer server: mplayer caught an exception
Mplayer server: mplayer caught signal 2
cpu == 100
Mplayer server: from 127.0.0.1:UDP1996 : 1
Mplayer server: mplayer stopped
cpu == 100
cpu == 100
cpu == 100
cpu == 100
cpu == 99

bash-3.2$ 
```

…System is too slow…

- Maybe a slow CPU?

t

Performance failure avoidance

```
void SystemIsSlow(void) {
   printf("Mplayer reports 'System too slow to
   play       clip’ and CPU is above threshold:\n");
   // drop frames more easily
   mplayer = HARDFRAMEDROP; }
...
   rrparse("(cpu>98)&&(mplayer==2);",
        SystemIsSlow);
```

Other RR vars

- int watchdog
  - Watchdog states if negative, and the amount of received heartbeats otherwise
- int bandwidth
  - Estimated bandwidth available b/w two TCP endpoints
- int linkbeacons[«MAC address»]
  - Number of beacons received during the current observation period in an ad hoc network
- int linkrates[«MAC address»]
  - Estimated bandwidth available between two nodes in an ad hoc network

Redundant variables

- « Worst case analysis do not pay off anymore… »
  - Common approach to choosing how much redundancy to employ: close-world assumption: “Fixed, reasonable choice, dependent on the context” ⇒
    1. overshooting: over-dimensioning the design with respect to the actual threat being experienced
    2. undershooting: underestimating the threat in view of an economy of resources

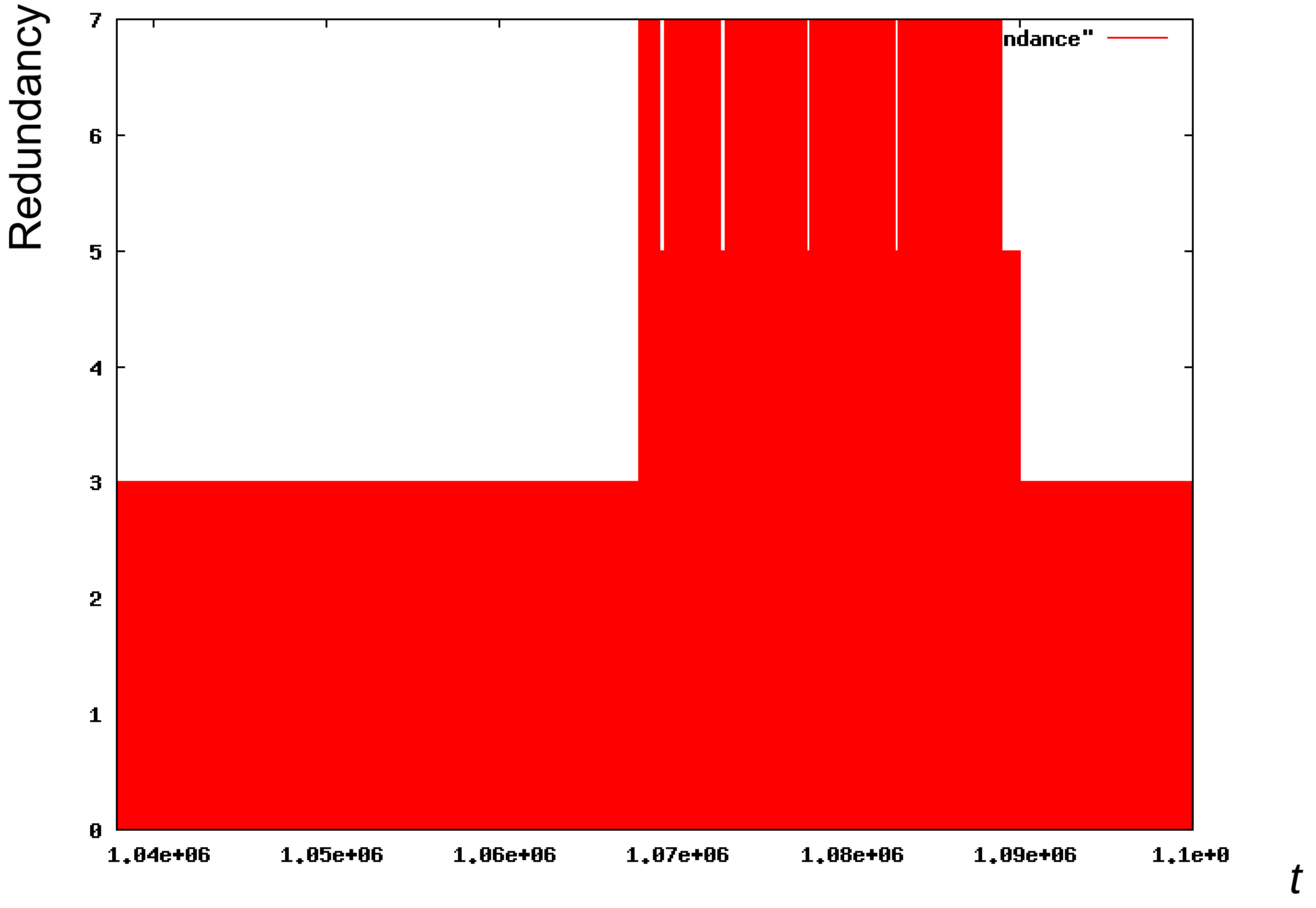
Redundant variables

- Adaptively redundant data structures
  - Variables whose contents get replicated several times so as to protect them from memory faults
    - Writing to a redundant variable = writing to *n* replicas, located somewhere and according to some strategy
    - Reading from a redundant variable = reading the *n* cells, performing majority voting
  - The result of this process is monitored by a RR var probe, which measures the amount of votes that differ from the majority
    - A measure of the disturbance in the surrounding environment

Redundant variables

- *n* is *n(t)*
- Under normal situation, *n*=3
  - The system triplicates the memory cells of redundant variables
  - This corresponds to tolerating up to one memory fault
- Under more critical situations, the amount of redundancy is adjusted
- The adjustment logic should tune in the ideal degree of redundancy with respect to the current disturbances

Redundant variables



Meta RR vars

- As already explained, RR vars have:
  - public side, where the adaptation and error recovery logics are specified by the user in a familiar form
  - private side, *separated but not hidden*, where the probing and actuation logics are defined.
- The logic in the private side can be indeed monitored and controlled by means of *meta RR vars*, i.e., variables reflecting / refracting on the state of the RR var system

Meta RR vars

- Information produced by error detectors is not discarded but fed into a fault identification mechanism (α-count)
- The current value of this mechanism is available to the user in the form of meta RR var alphacount[i]
  - i identifies the error detector

Meta RR vars

- This allows to set up assertions on the validity of the fault model, e.g.

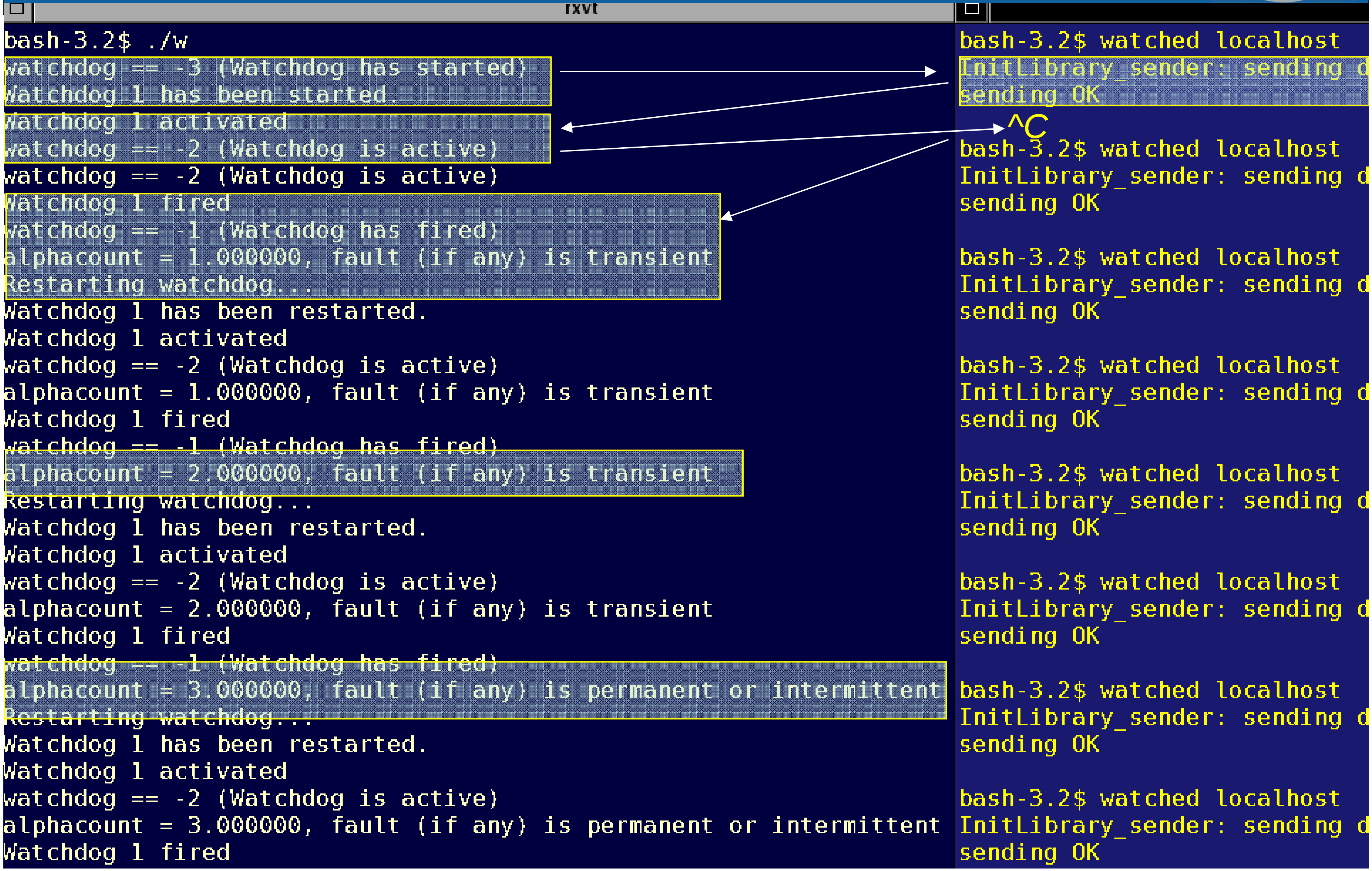
```
void AssumptionMismatch(void) {
        printf("Wrong fault model assumption
                  caught\n");
}
...
rrparse("(alphacount[1]>3.0);",
        AssumptionMismatch);
                // 3.0 = Alpha-count threshold
```

Meta RR vars

- A scenario involving a watchdog (left-hand window) and a watched task (right-hand).
- The watched task is repeatedly interrupted and restarted, so as to emulate the effect of some permanent fault.
- As a consequence, the watchdog “fires” and updates an α-count variable.
- The value of the α-count variable increases until it reaches a threshold (3.0)

→ Fault is labeled as permanent-or-intermittent.

Meta RR vars



```
bash-3.2$ ./w
watchdog == -3 (Watchdog has started)
Watchdog 1 has been started.
Watchdog 1 activated
watchdog == -2 (Watchdog is active)
watchdog == -2 (Watchdog is active)
Watchdog 1 fired
watchdog == -1 (Watchdog has fired)
alphacount = 1.000000, fault (if any) is transient
Restarting watchdog...
Watchdog 1 has been restarted.
Watchdog 1 activated
watchdog == -2 (Watchdog is active)
alphacount = 1.000000, fault (if any) is transient
Watchdog 1 fired
watchdog == -1 (Watchdog has fired)
alphacount = 2.000000, fault (if any) is transient
Restarting watchdog...
Watchdog 1 has been restarted.
Watchdog 1 activated
watchdog == -2 (Watchdog is active)
alphacount = 2.000000, fault (if any) is transient
Watchdog 1 fired
watchdog == -1 (Watchdog has fired)
alphacount = 3.000000, fault (if any) is permanent or intermittent
Restarting watchdog...
Watchdog 1 has been restarted.
Watchdog 1 activated
watchdog == -2 (Watchdog is active)
alphacount = 3.000000, fault (if any) is permanent or intermittent
Watchdog 1 fired
```

```
bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK
 ^C
bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK

bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK

bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK

bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK

bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK

bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK

bash-3.2$ watched localhost
InitLibrary_sender: sending d
sending OK
```

In conclusion…

- Worst-case analyses do not pay off anymore
  - → Redundant vars as optimal way to choose the amount of redundancy
- Horning syndrome
  - → RR vars to express and realize open-world systems
- Hidden intelligence syndrome
  - → Meta RR vars to set up assertions on the validity of the fault / system models and platform

In conclusion…

- An excerpt of our current research directions in Antwerp
- Future steps: other mechanisms to allow more systematically the design time hypotheses about system and environment to be expressed and asserted
- Ultimate challenge: intelligent management of the dependability strategies

Thank you
for your attention!

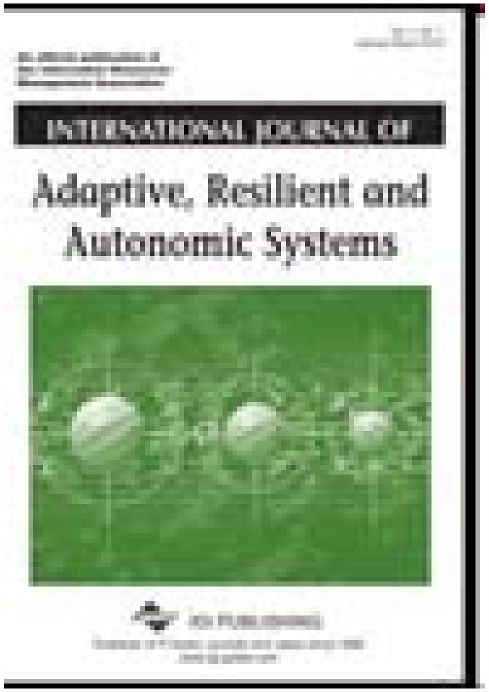
Questions?

vincenzo.deflorio@ua.ac.be

References

- C. Esposito and D. Cotroneo, “Resilient and Timely Event Dissemination in Publish/Subscribe Middleware”, to appear in IJARAS #1, Oct. 2009
- L. Simoncini, “Technological and Educational Challenges of Resilient Computing”, to appear in IJARAS #1, Oct. 2009
- J. Horning, “ACM Fellow Profile --- James Jay (Jim) Horning”, ACM Software Engineering Notes vol.23 no.4, 1998.

IJARAS



INTERNATIONAL JOURNAL OF

Adaptive, Resilient and Autonomic Systems

IGI PUBLISHING

http://www.igi-global.com/journals/details.asp?id=34265