# Co-evolution of Source Code and the Build System

Supervisors:

Herman Tromp
*GH-SEL, Ghent University*

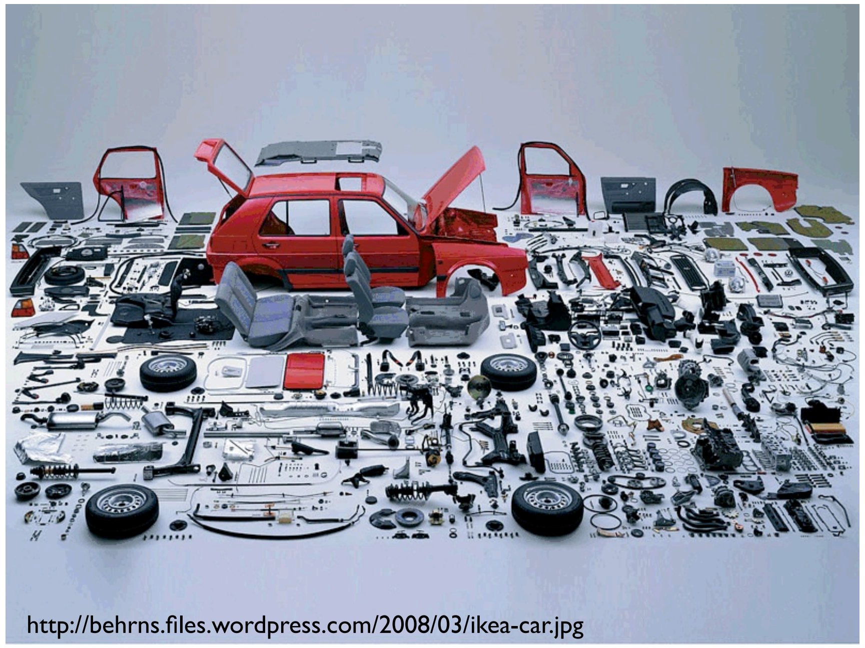
Wolfgang De Meuter
*SOFT, Vrije Universiteit Brussel*

**Bram Adams**
SAIL, Queen's University
http://sailhome.cs.queensu.ca/~bram/

Source code and build system **co-evolve**.

We need tools and techniques to **understand** this co-evolution.

# Building a Car

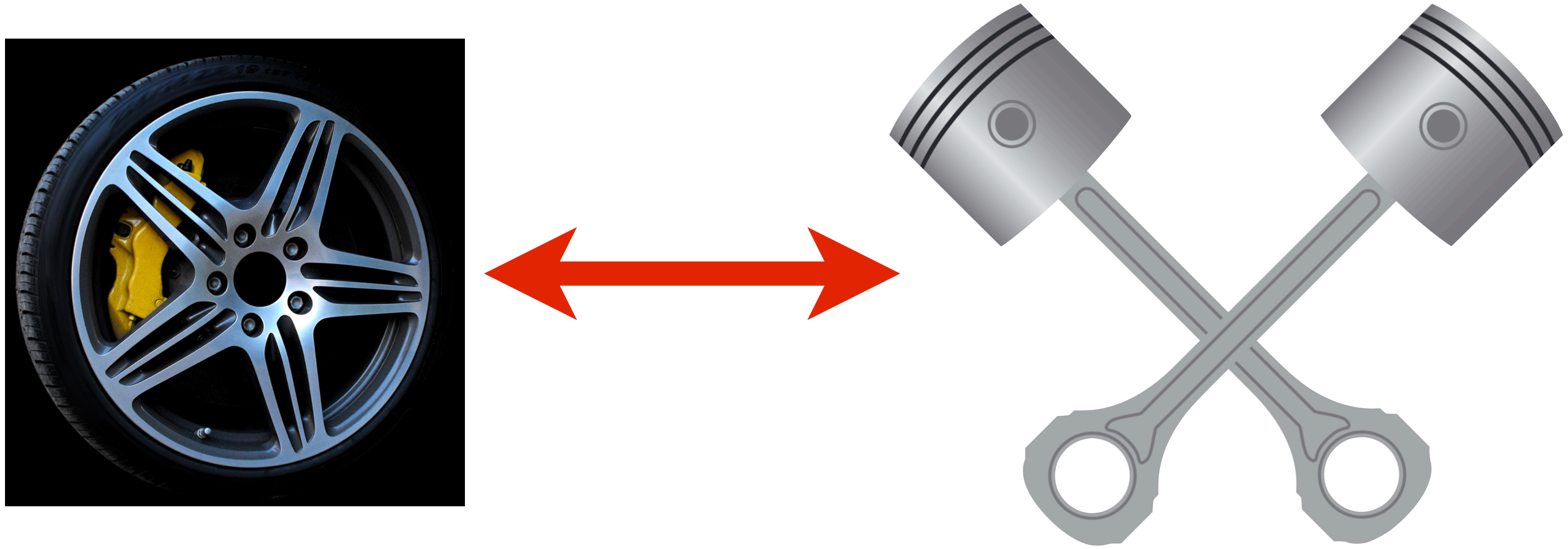


http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

?

# Building a Car: Configuration

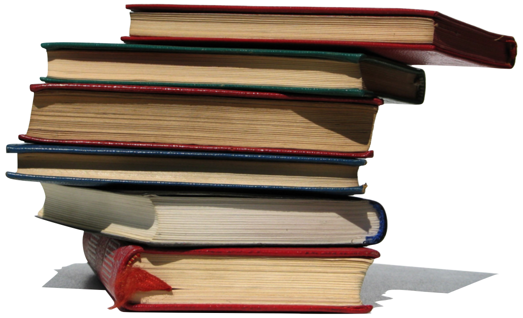
1. features



2. tools

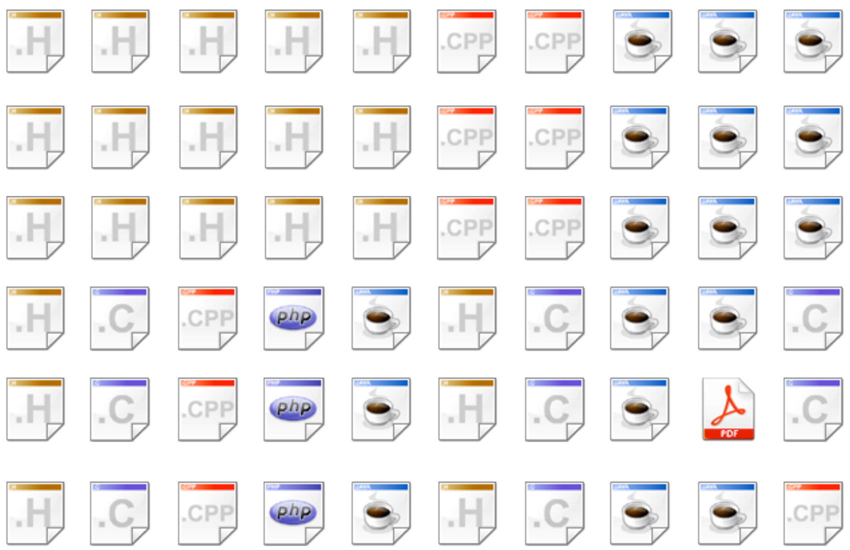
# Building a Car: Actual Building

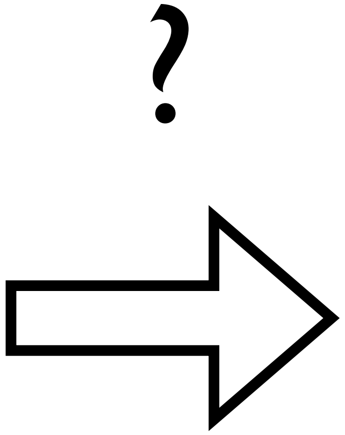
1. prescriptions

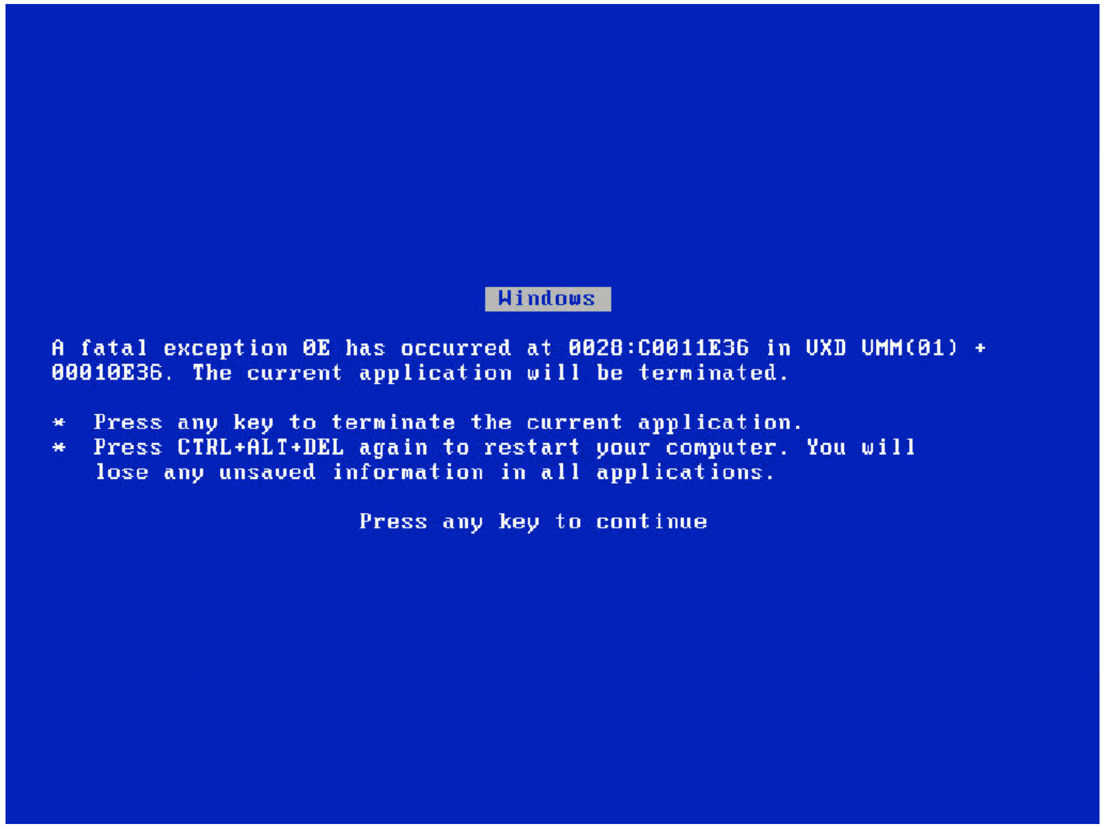


2. dependencies

# Building Software

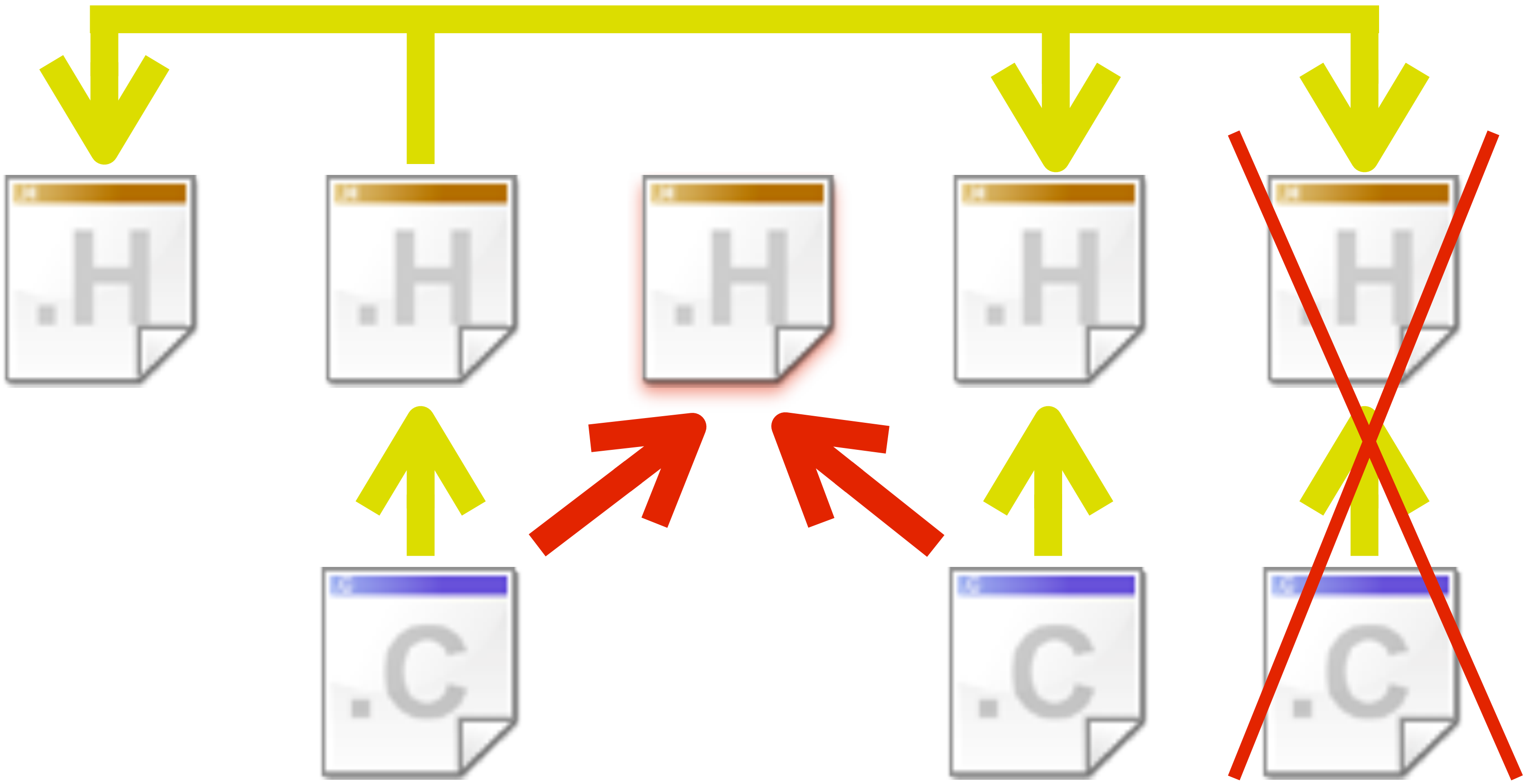






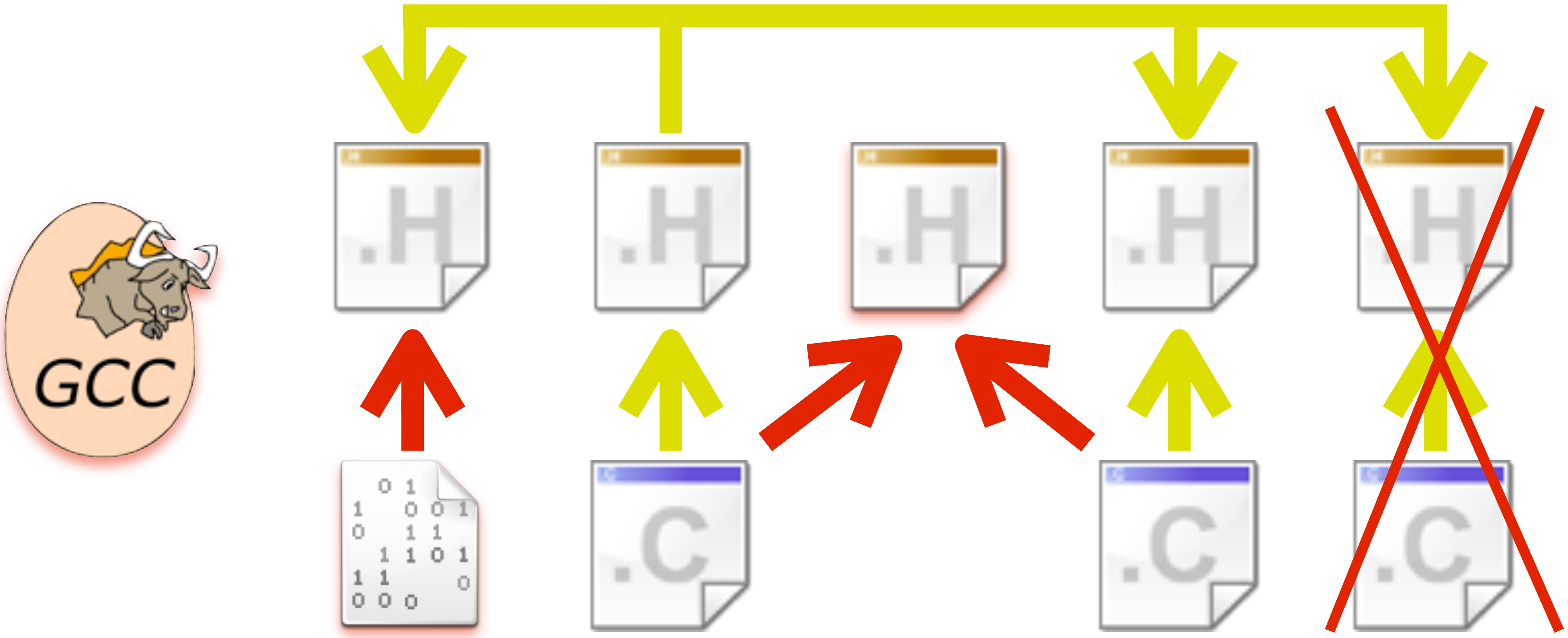
```
A fatal exception 0E has occurred at 0028:C0011E36 in VXD VMM(01) +
00010E36. The current application will be terminated.

*  Press any key to terminate the current application.
*  Press CTRL+ALT+DEL again to restart your computer. You will
   lose any unsaved information in all applications.

                        Press any key to continue
```

# Configuration Layer



1. features

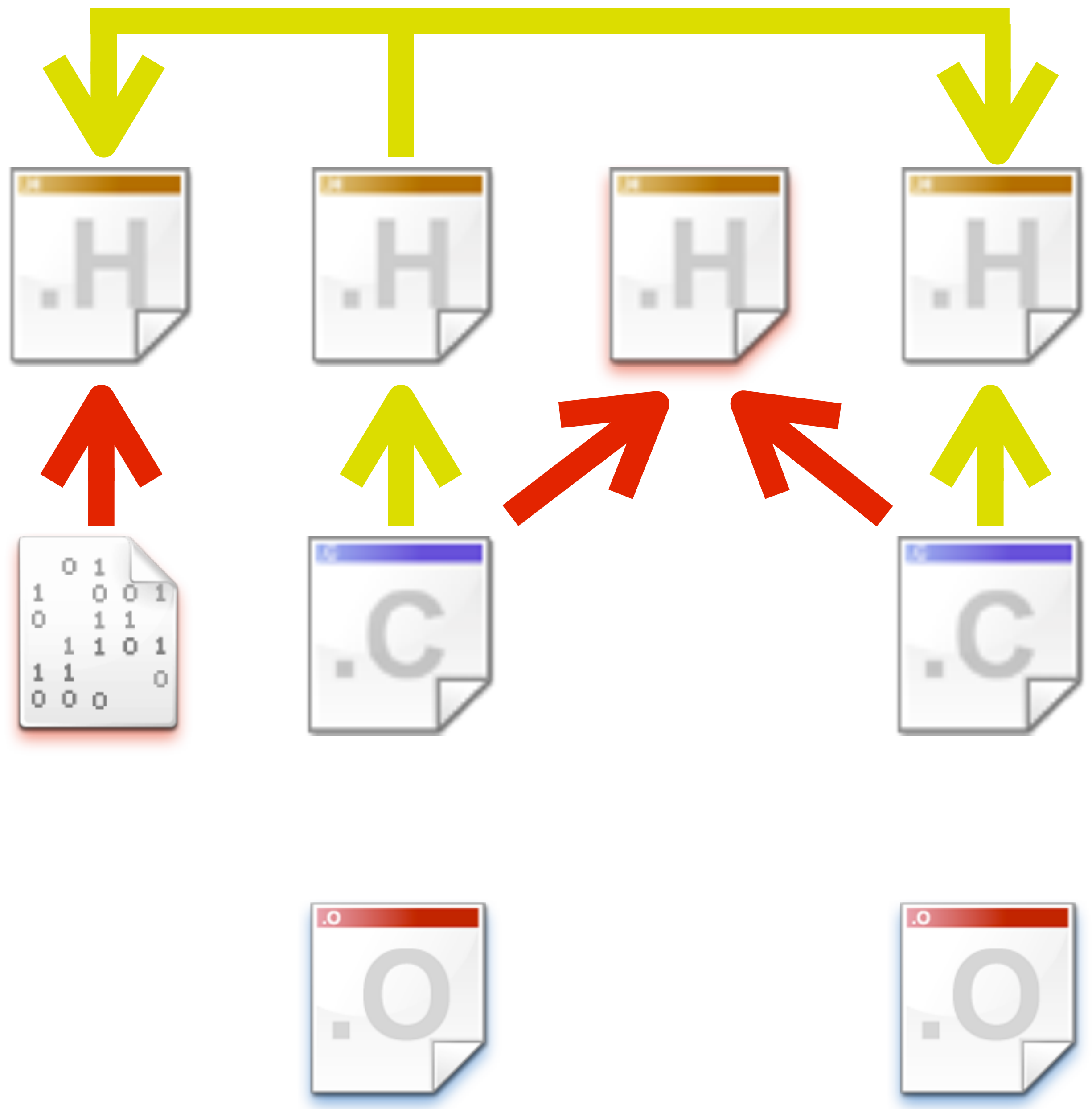
# Configuration Layer



1. features

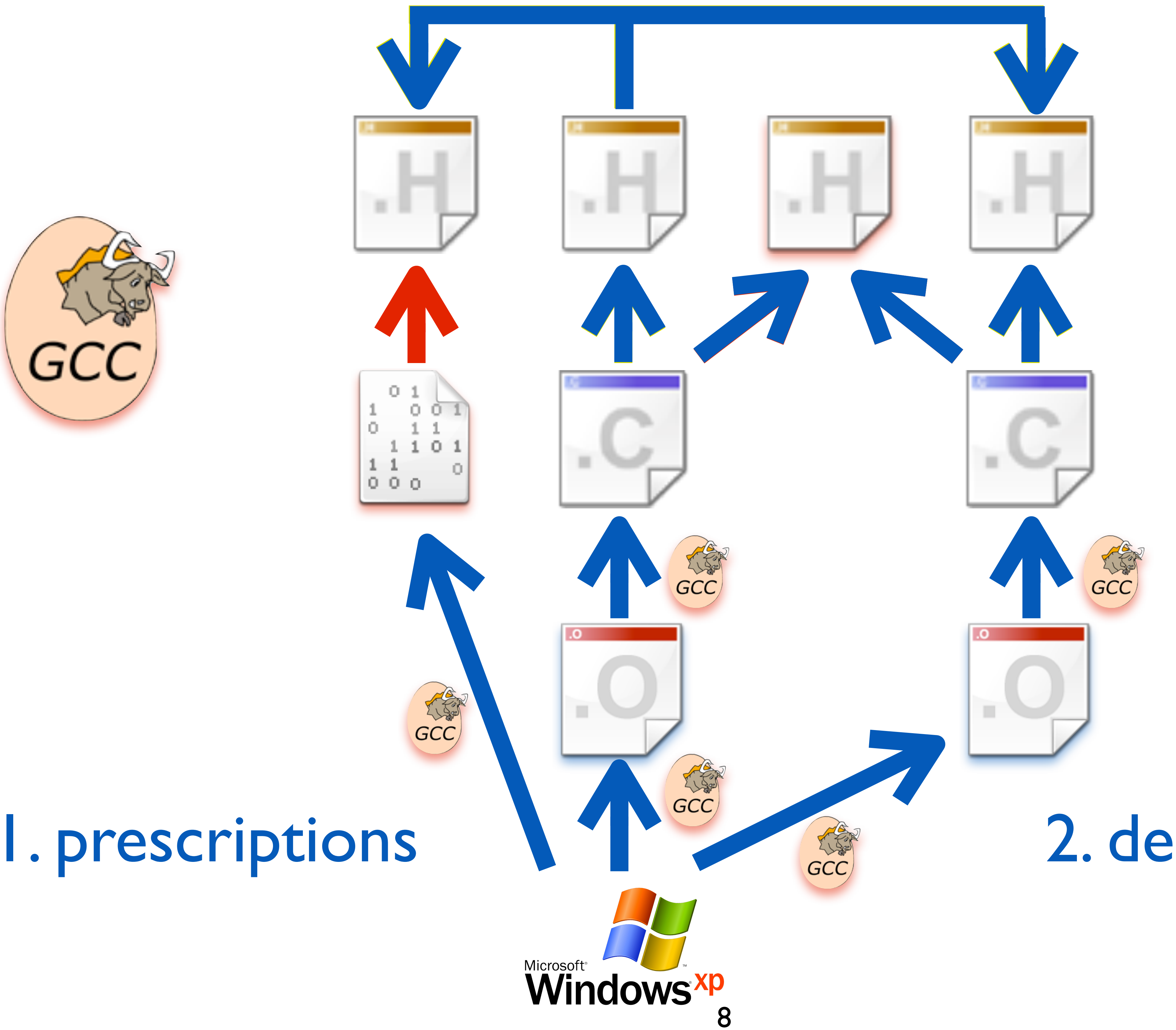
2. tools

# Build Layer



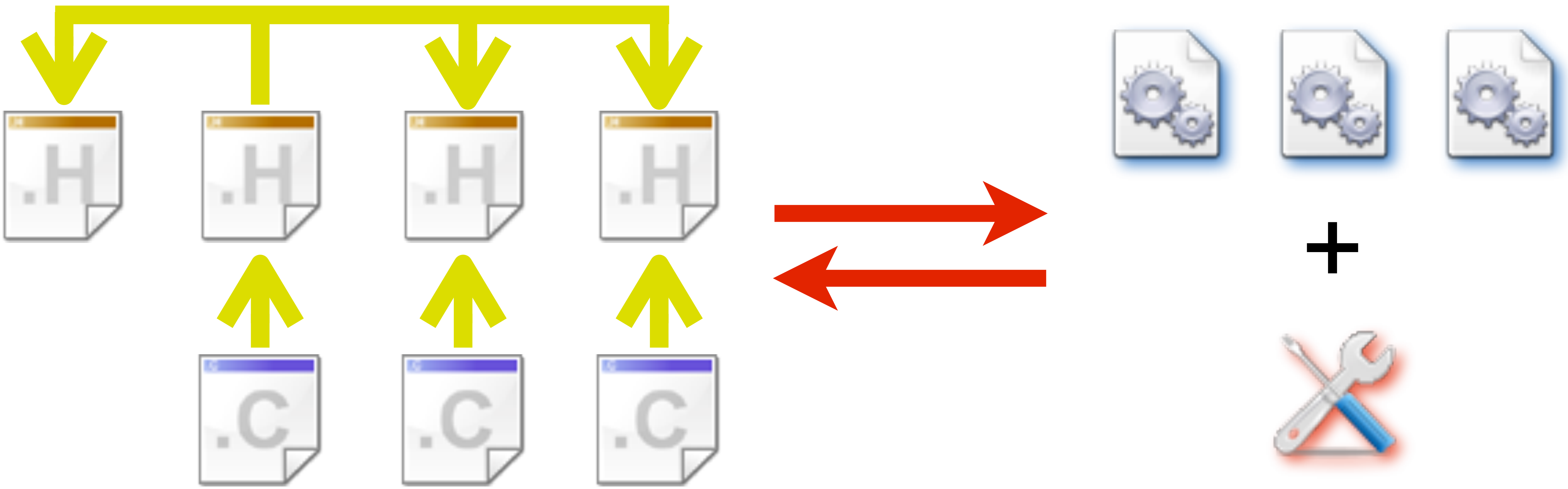
1. prescriptions

# Build Layer



1. prescriptions

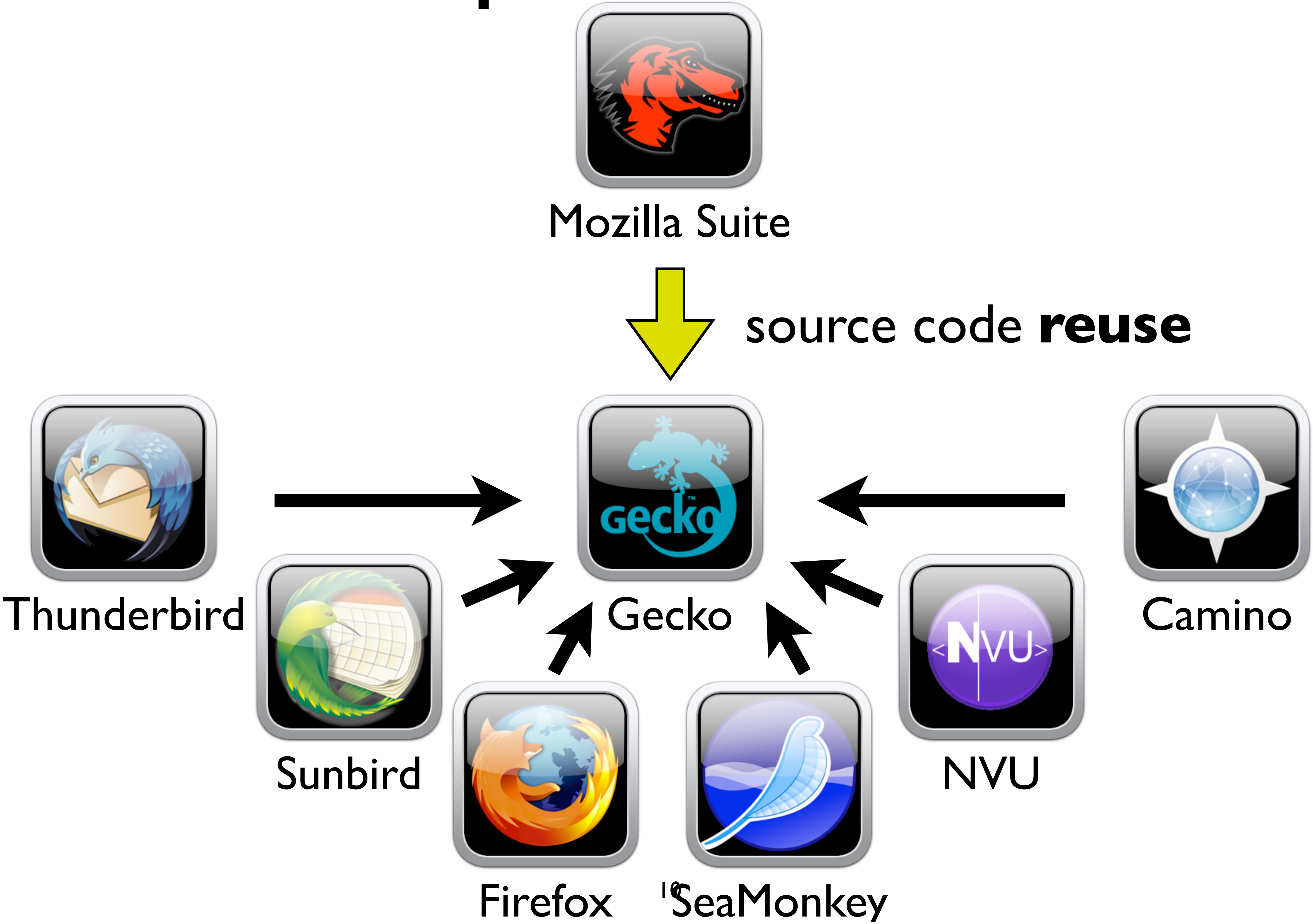
2. dependencies



**Source Code**

**Build System**

# Component Reuse



Mozilla Suite

source code **reuse**

Thunderbird

Sunbird

Firefox

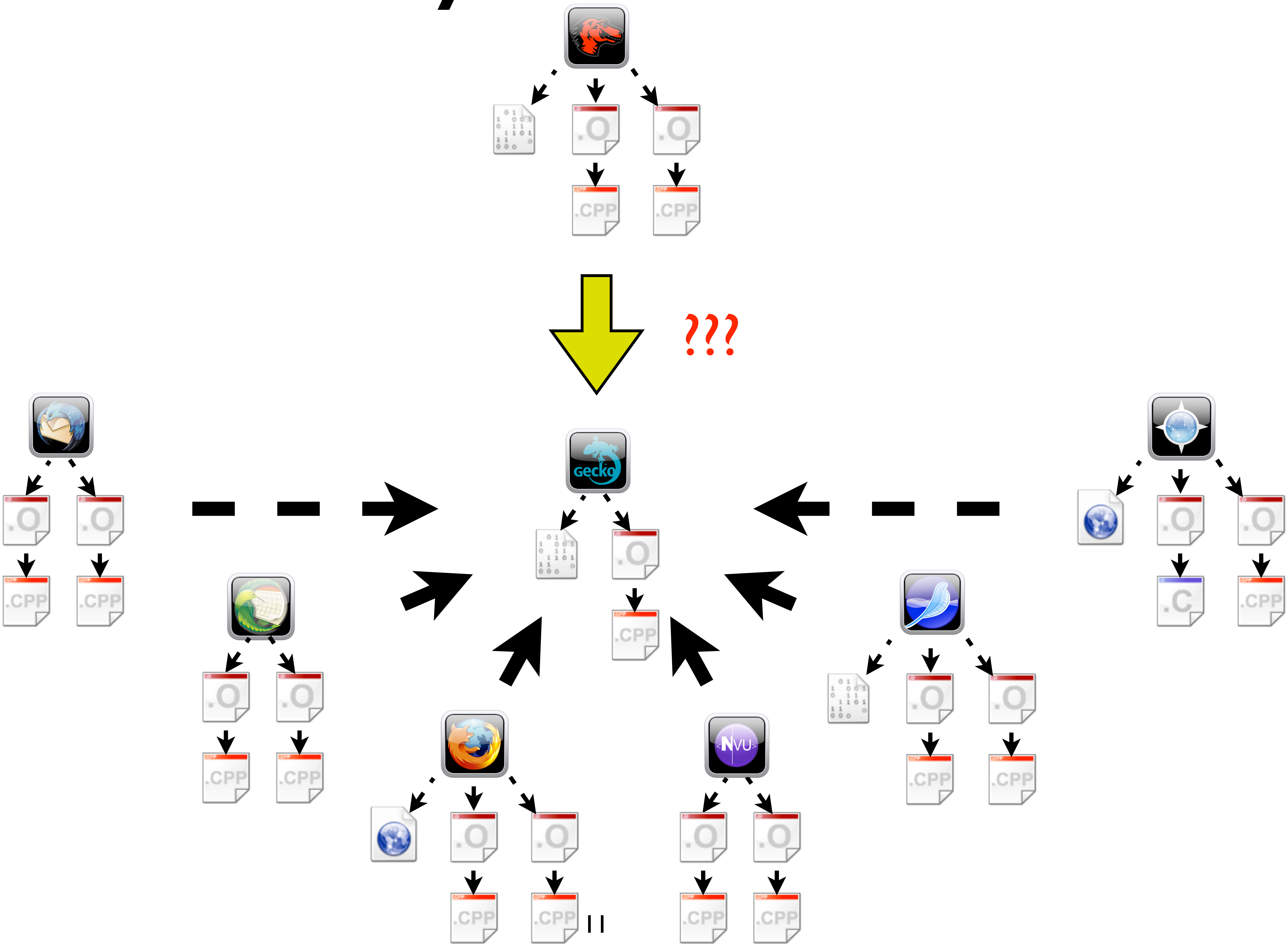
Gecko

SeaMonkey

NVU

Camino

# Build System Reuse?



Source code and build system **co-evolve**.

We need tools and techniques to **understand** this co-evolution.

1. Research Hypothesis

2. Tool Support to Understand Build Systems

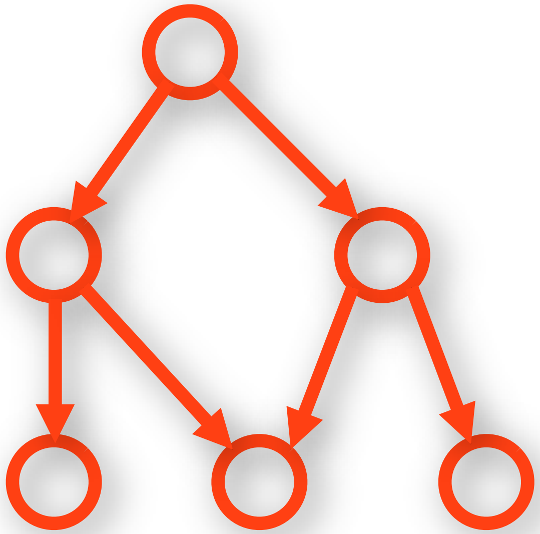
3. Evolution of Linux Kernel Build System

4. Conceptual Reasons of Co-evolution

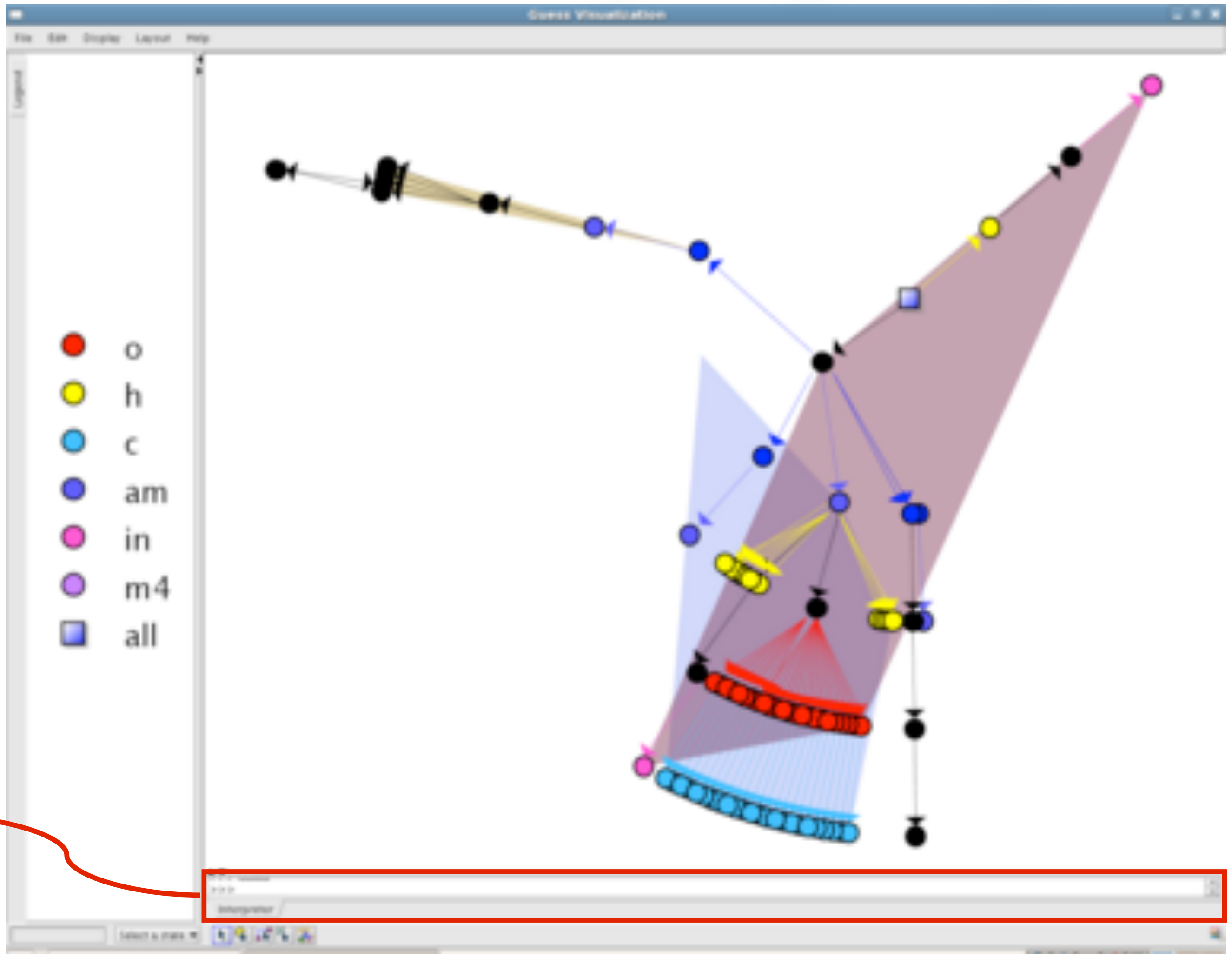
5. The Pitfalls of PhD Research

6. Conclusion

# Understanding the Build System



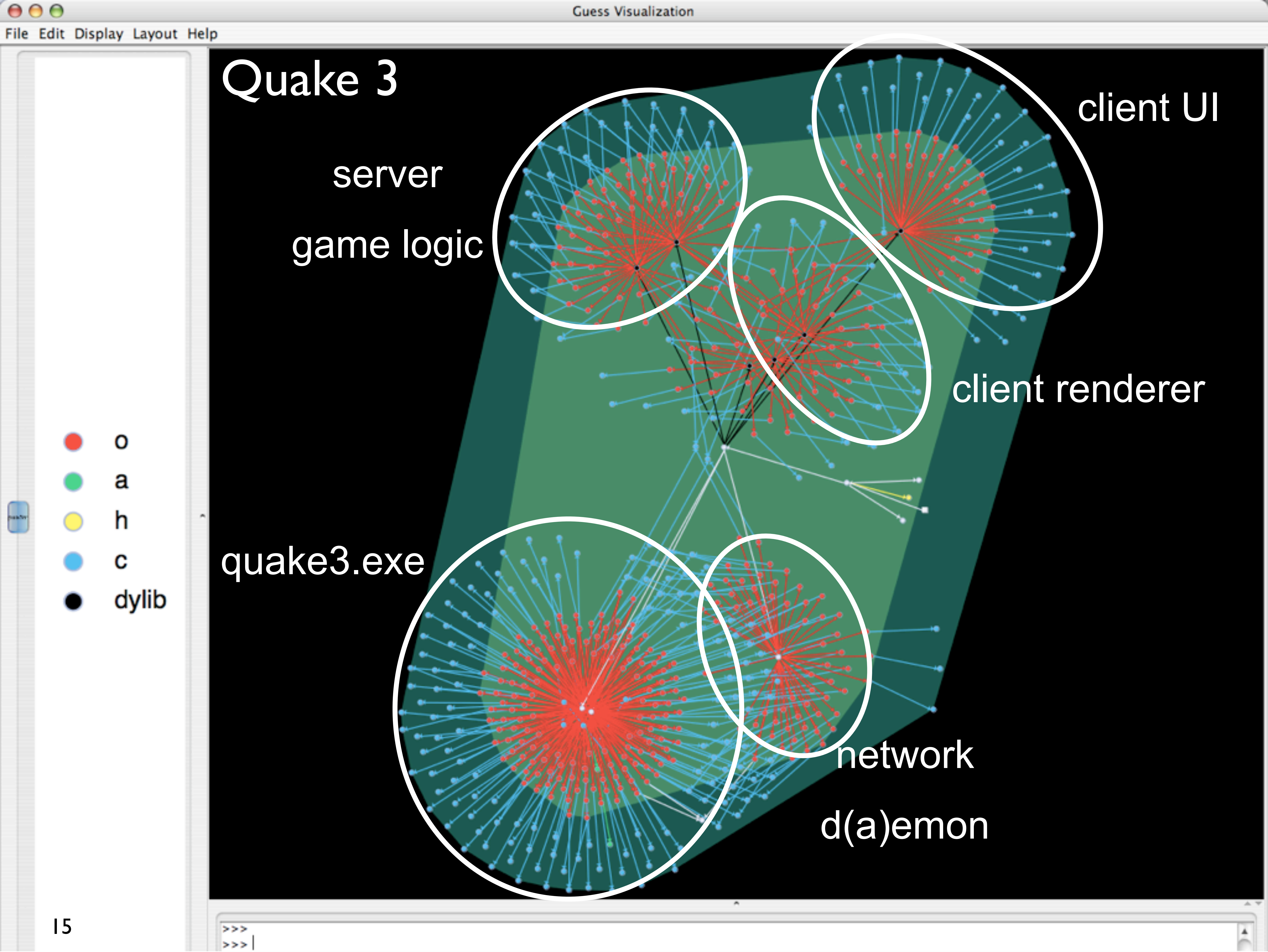
MAKAO



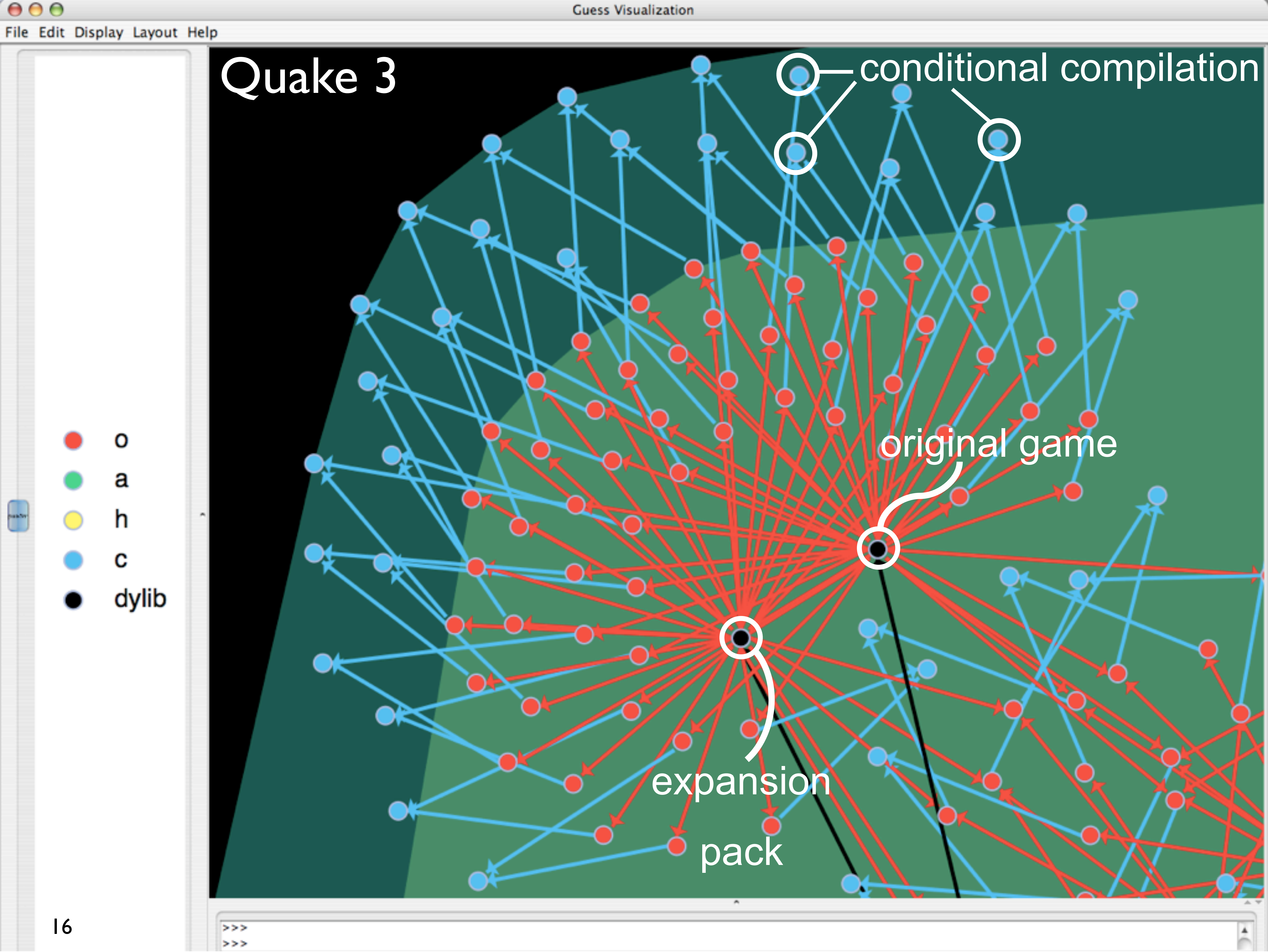
embedded Gython

Prolog

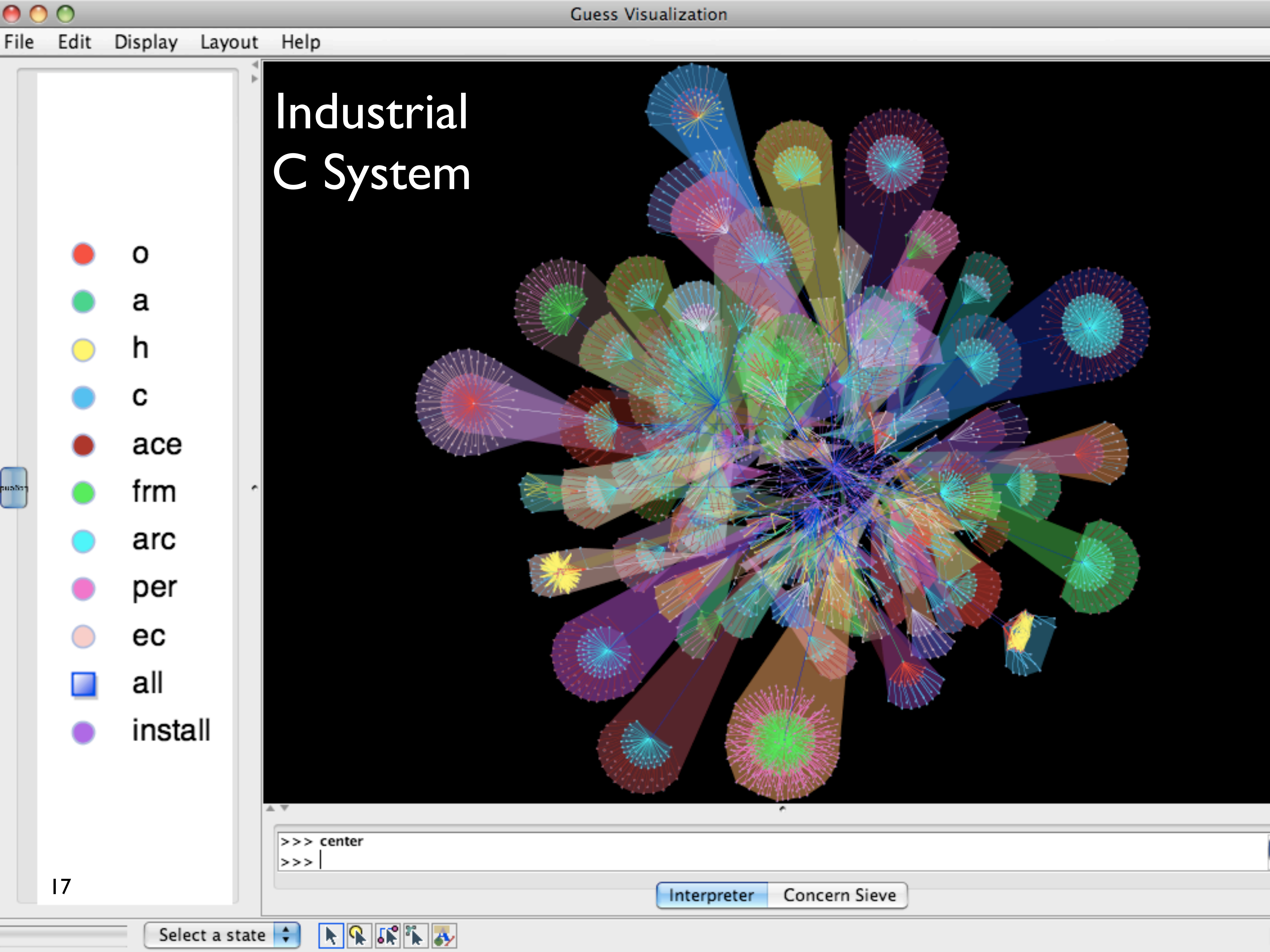
[ICSM '07]


Guess Visualization
File
Edit
Display
Layout
Help
Quake 3
server
game logic
client UI
client renderer
quake3.exe
network
d(a)emon
● o
● a
● h
● c
● dylib

```
>>>
>>> |
```


Guess Visualization
File Edit Display Layout Help
Quake 3
conditional compilation
original game
expansion
pack
o
a
h
c
dylib

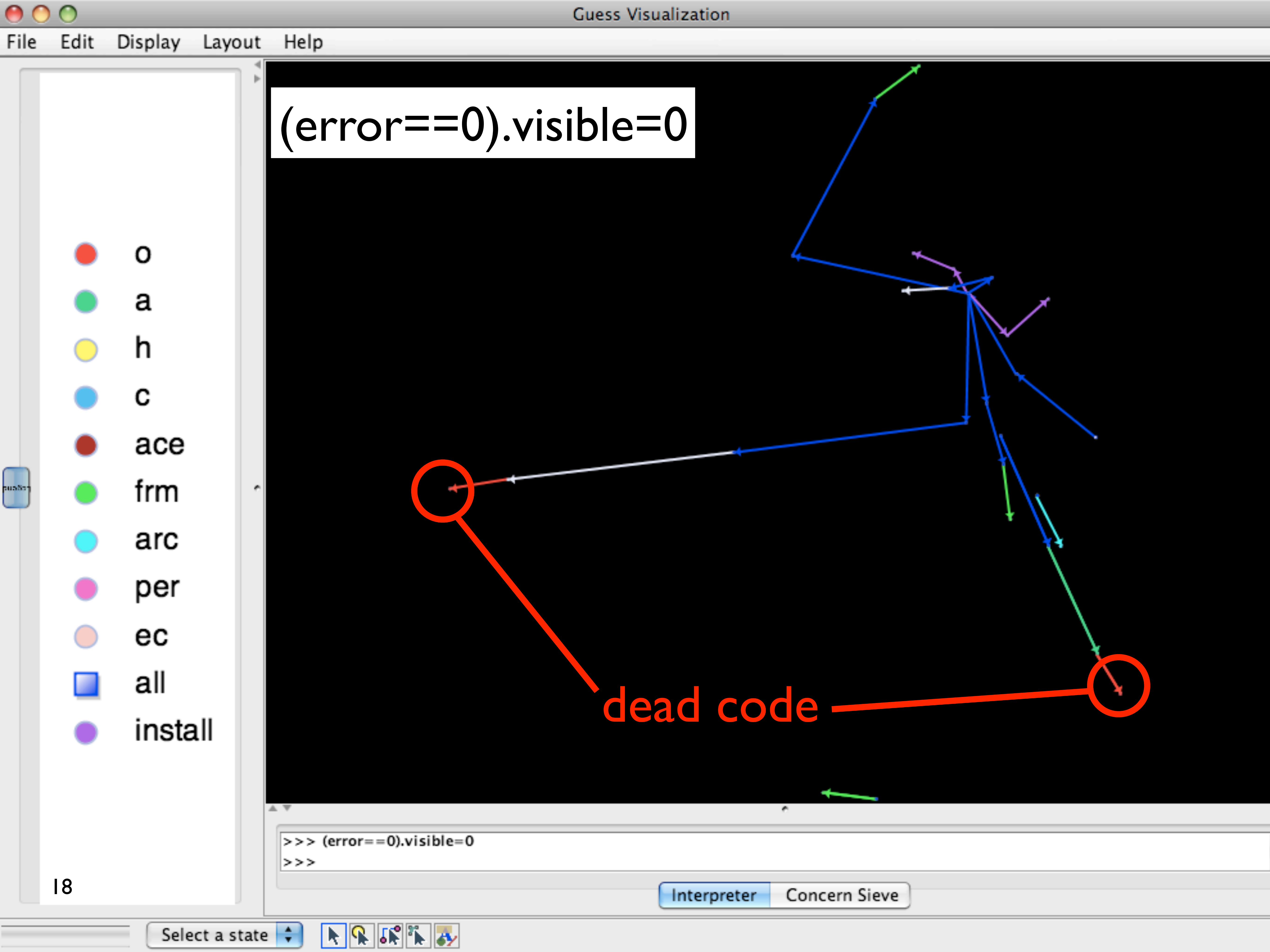
```
>>>
>>>
```



# Industrial C System

- 🟥 o
- 🟩 a
- 🟨 h
- 🟦 c
- 🟥 ace
- 🟩 frm
- 🟦 arc
- 🟪 per
- ● ec
- □ all
- 🟪 install

```
>>> center
>>> |
```



(error==0).visible=0

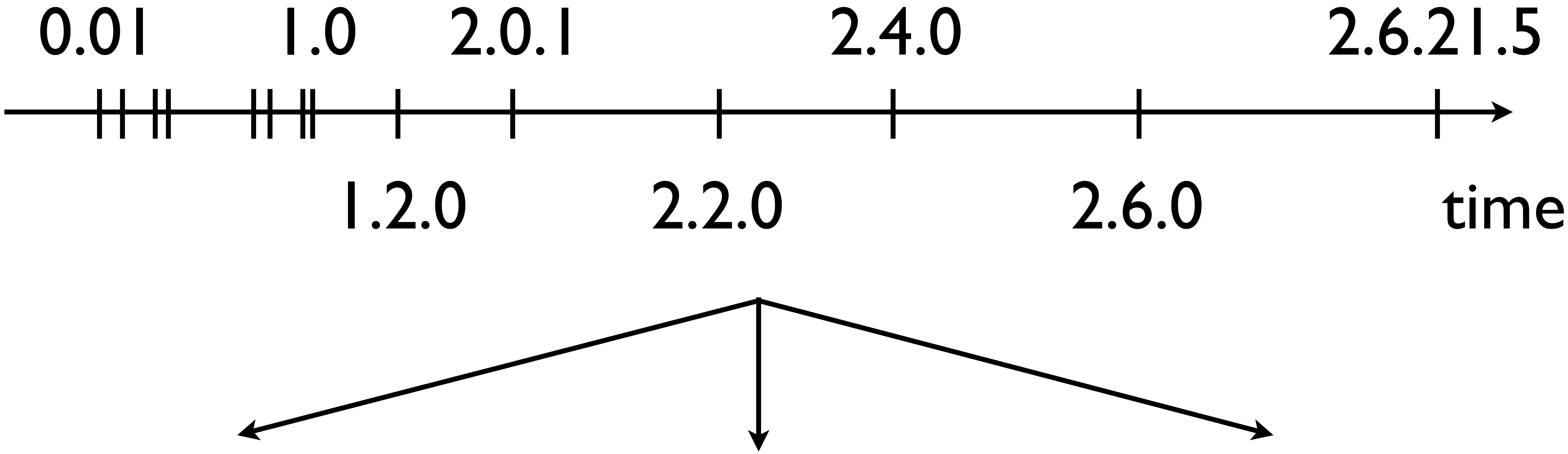
- ● o
- ● a
- ● h
- ● c
- ● ace
- ● frm
- ● arc
- ● per
- ● ec
- ■ all
- ● install

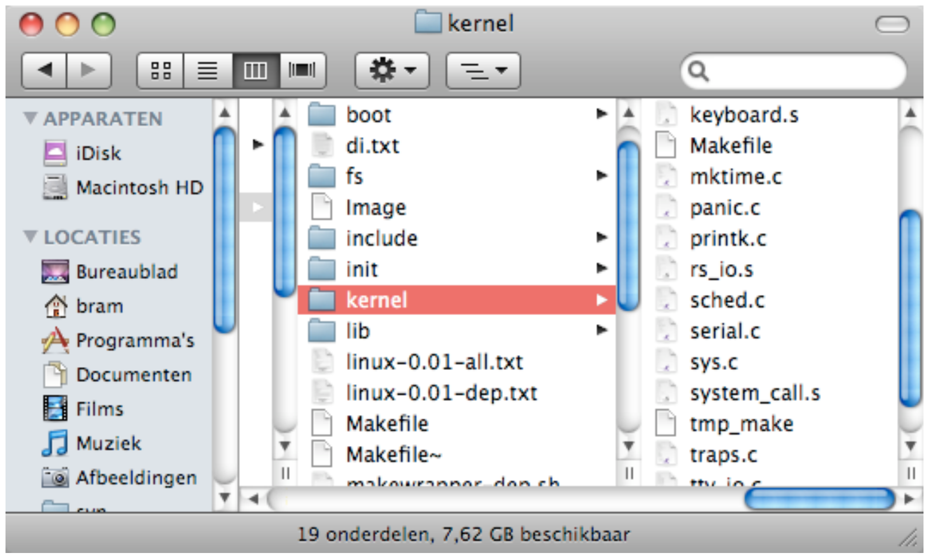
```
>>> (error==0).visible=0
>>>
```

dead code

1. Research Hypothesis
2. Tool Support to Understand Build Systems
3. Evolution of Linux Kernel Build System
4. Conceptual Reasons of Co-evolution
5. The Pitfalls of PhD Research
6. Conclusion

# Linux Case Study

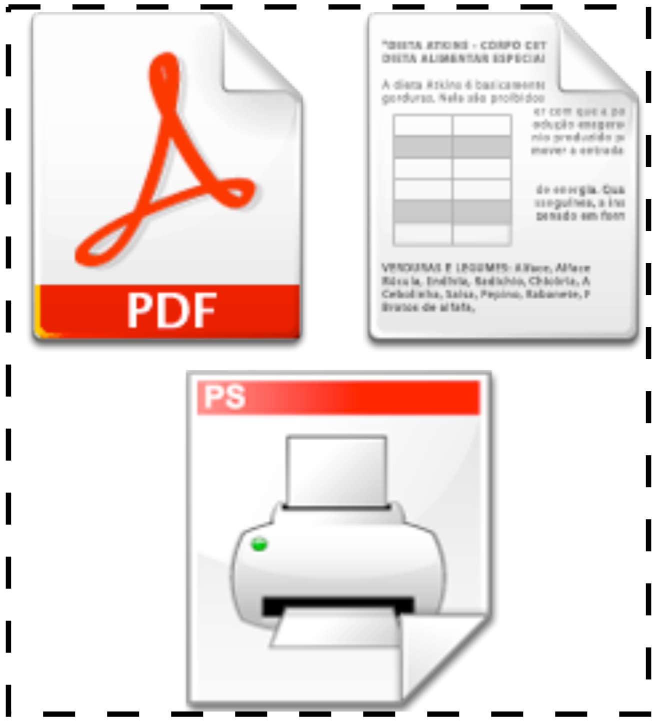


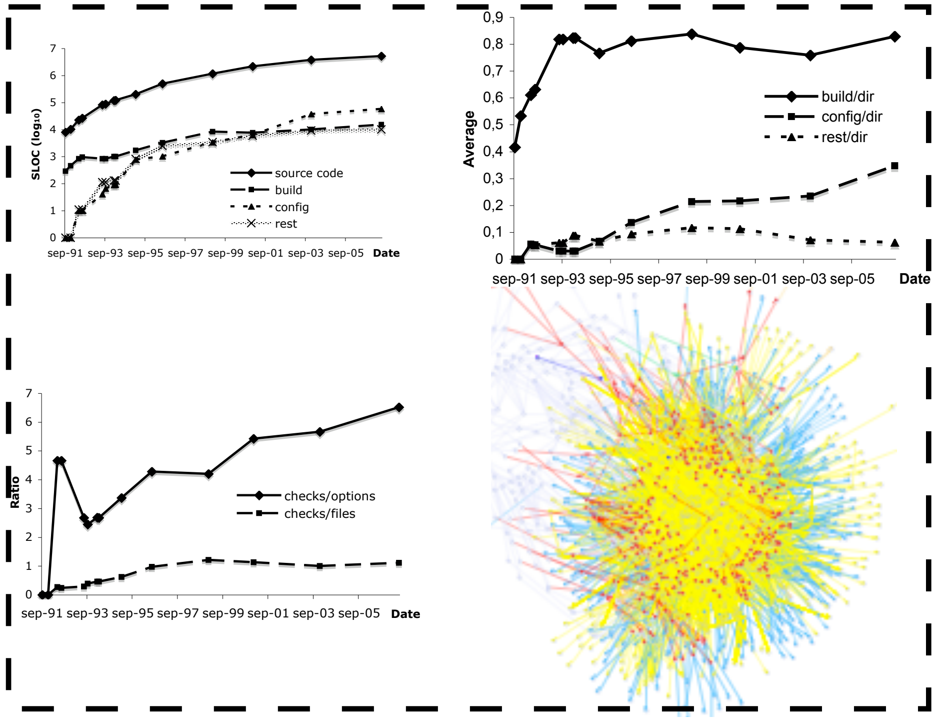


boot
di.txt
fs
Image
include
init
kernel
lib
linux-0.01-all.txt
linux-0.01-dep.txt
Makefile
Makefile~
makewrapper-dep.sh

keyboard.s
Makefile
mktime.c
panic.c
printk.c
rs_io.s
sched.c
serial.c
sys.c
system_call.s
tmp_make
traps.c
tty_io.c

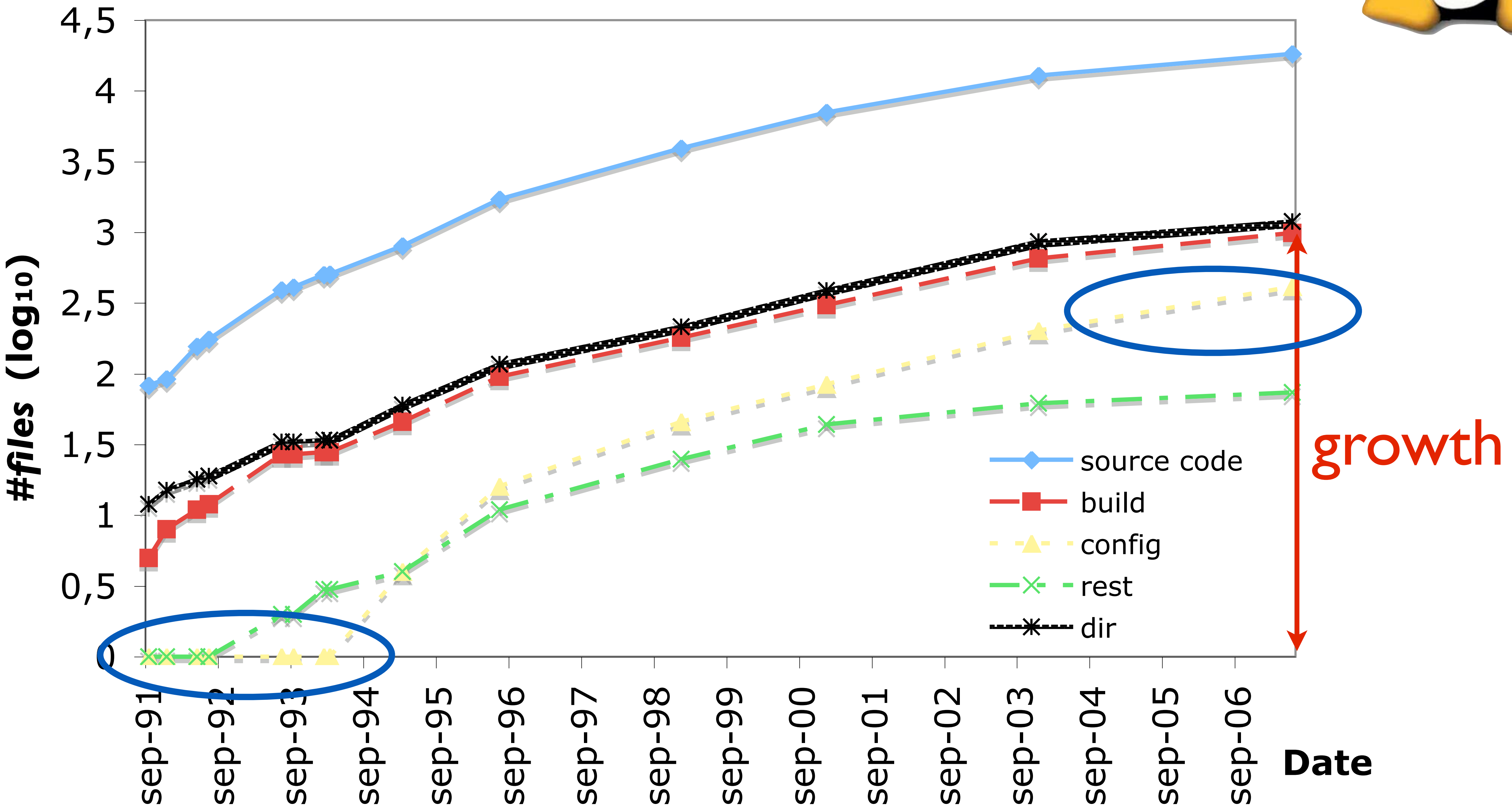
19 onderdelen, 7,62 GB beschikbaar



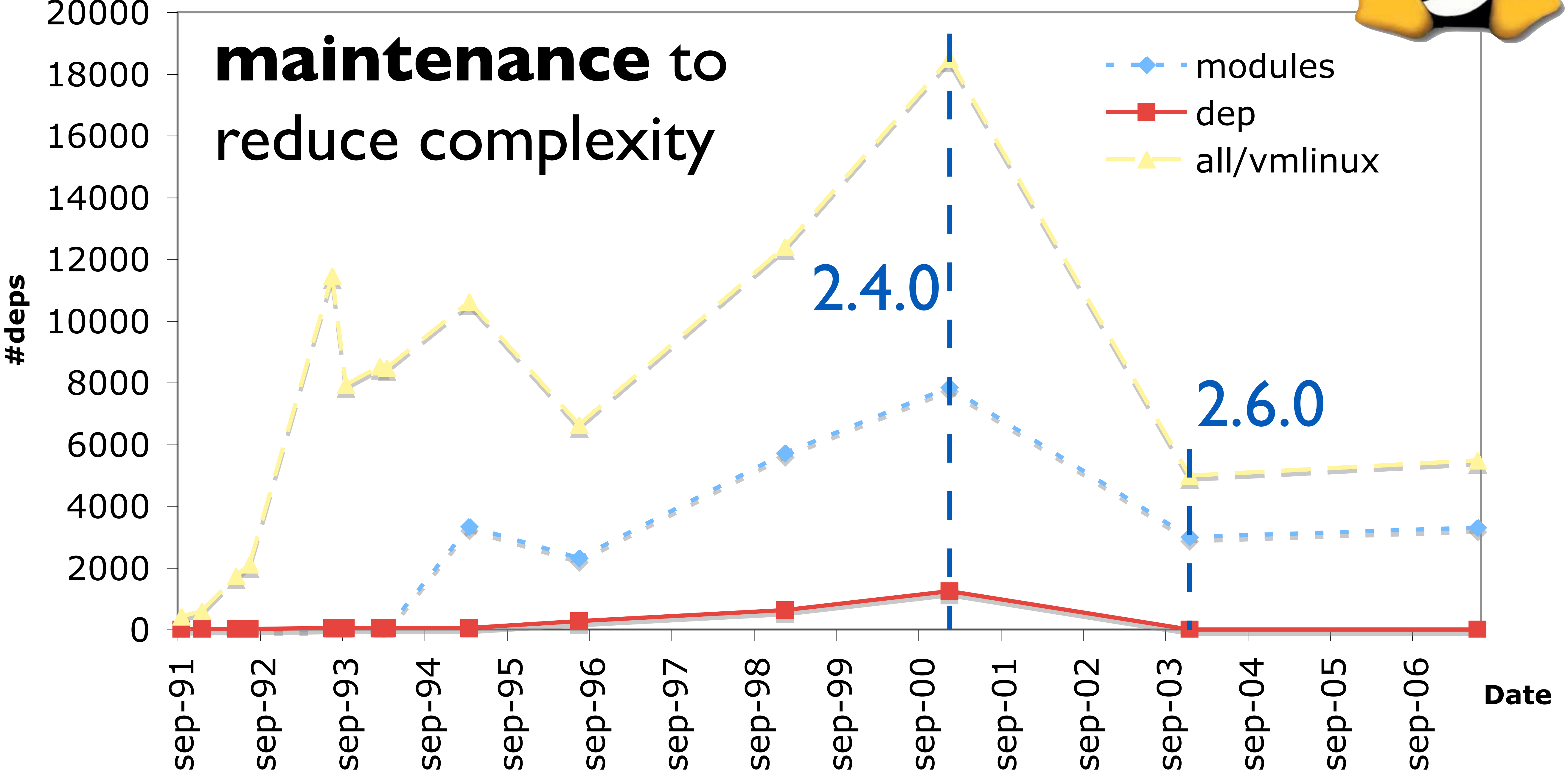


[ERCIM '07]

# Evolution of SLOC



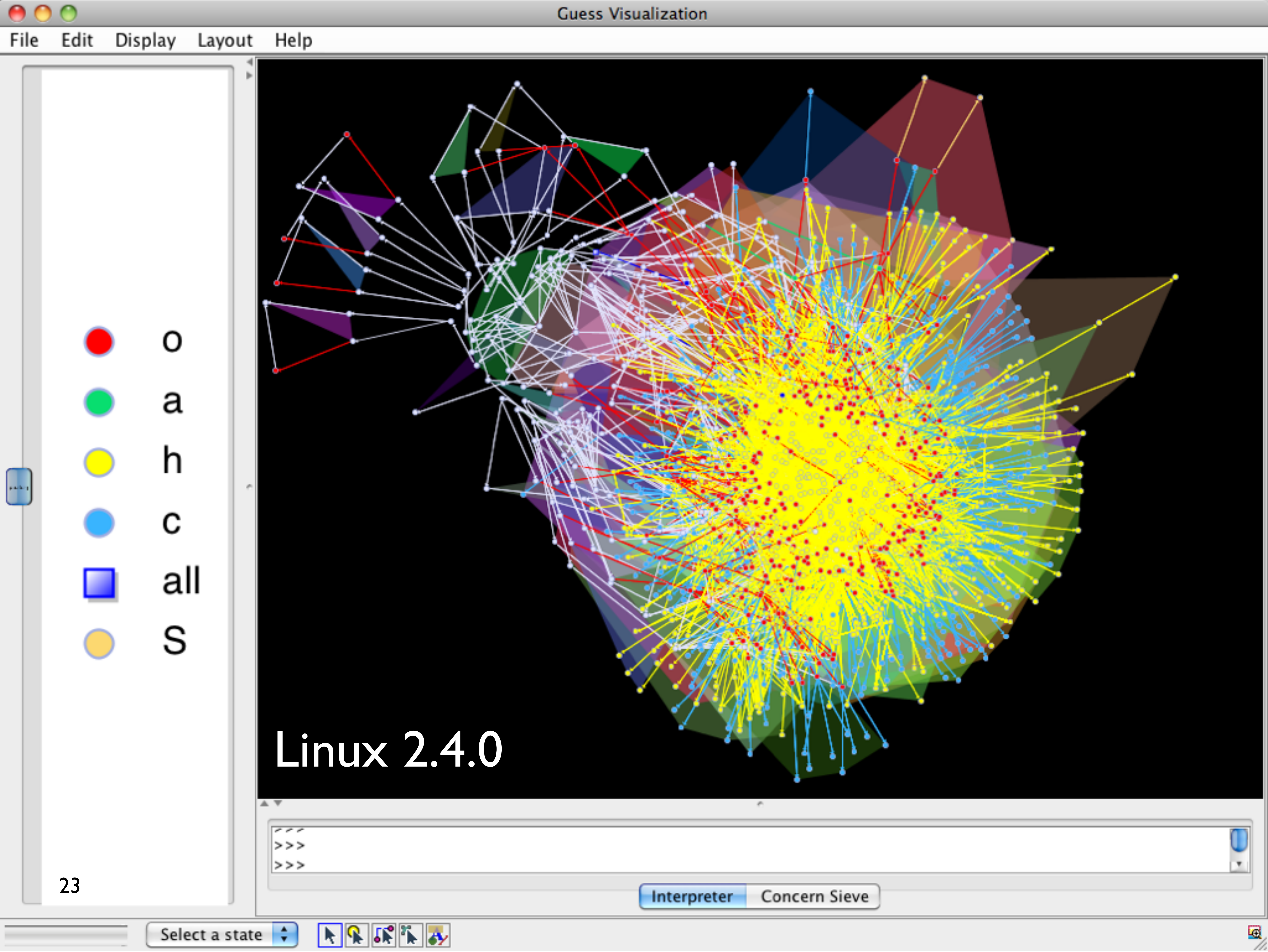
# Evolution of #dependencies



**maintenance** to reduce complexity

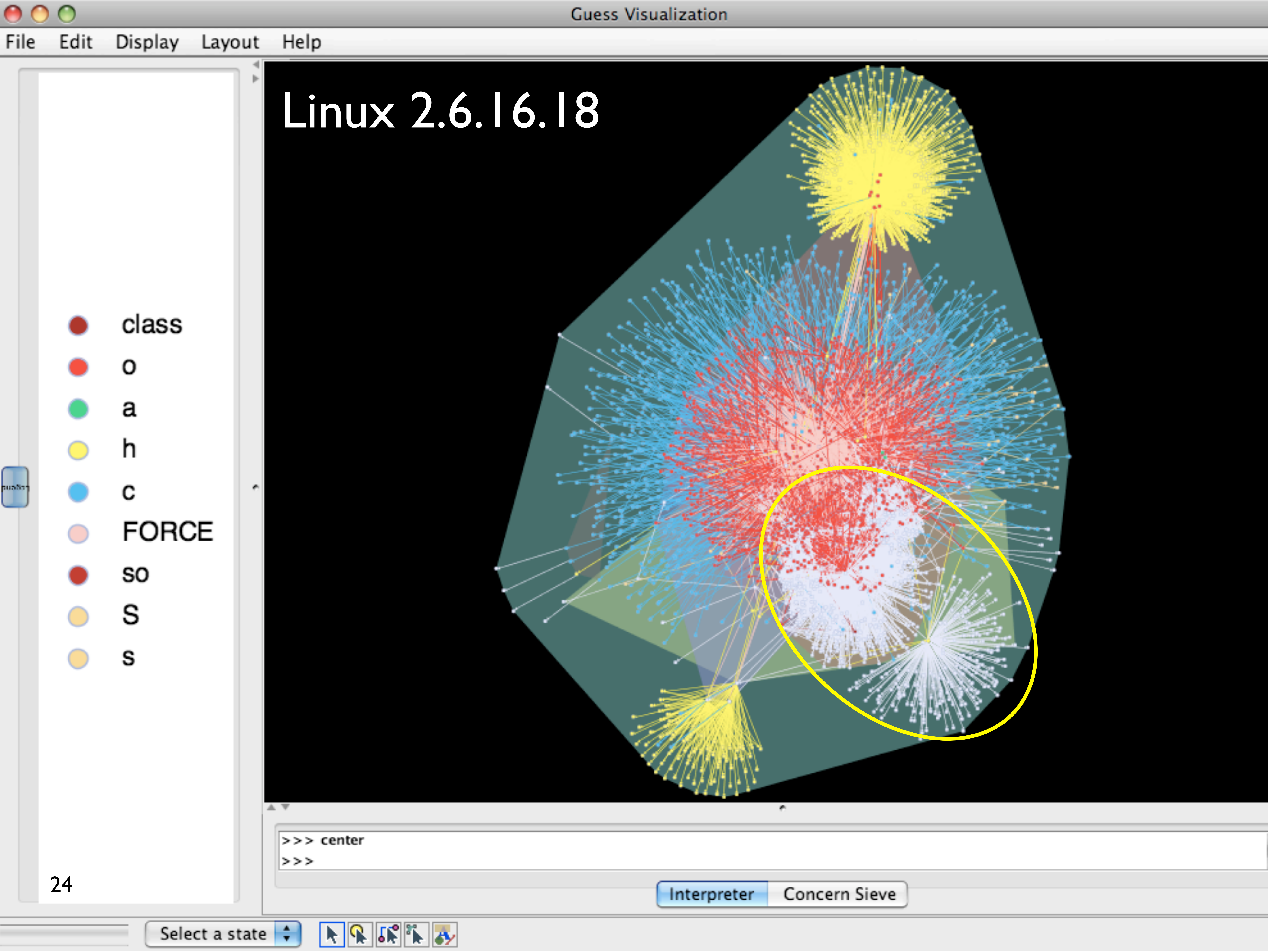
2.4.0

2.6.0


Guess Visualization
File
Edit
Display
Layout
Help
● o
● a
● h
● c
■ all
● S
Interpreter
Concern Sieve
Select a state

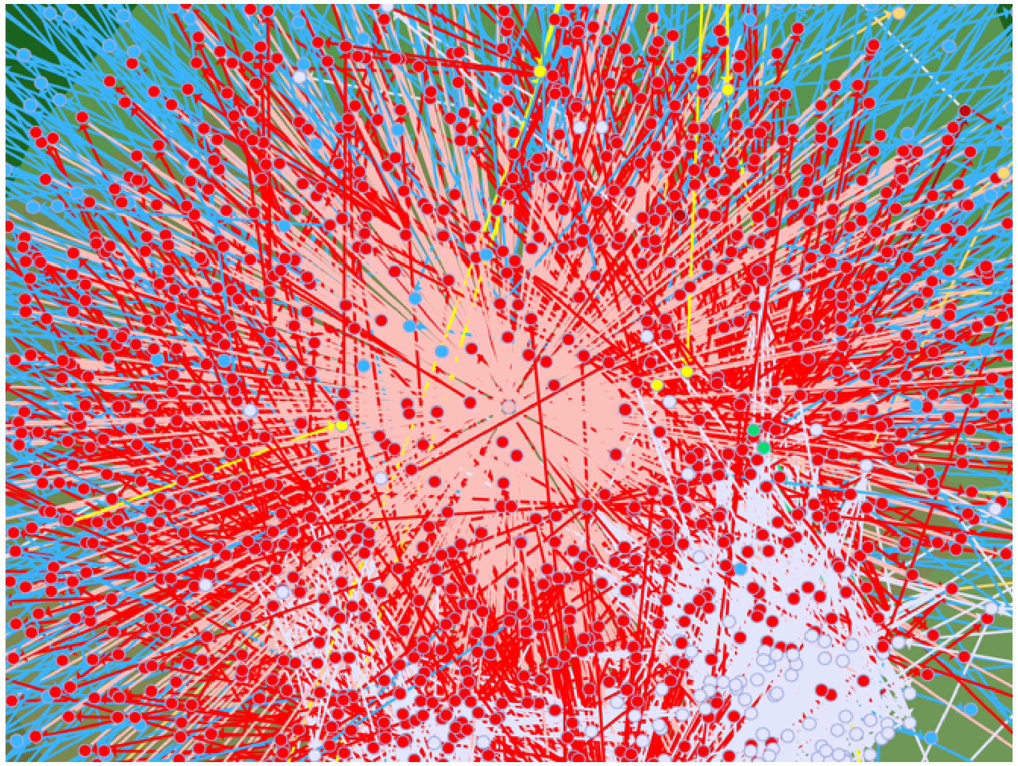
Linux 2.4.0

```
>>>
>>>
```

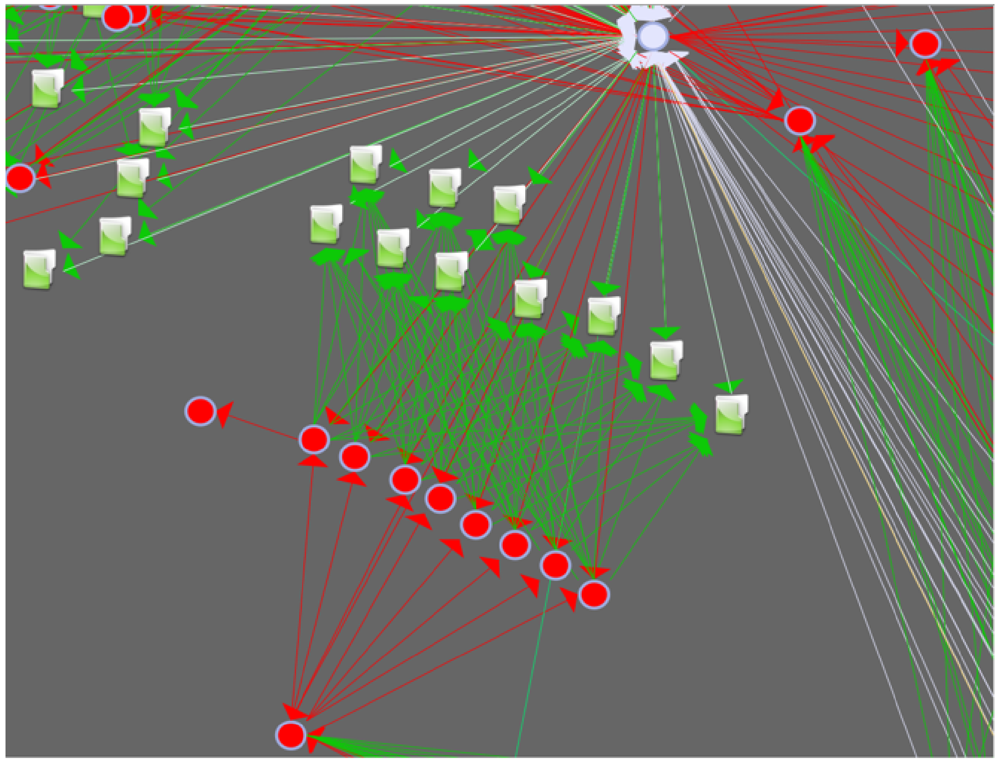

Guess Visualization
File
Edit
Display
Layout
Help
Linux 2.6.16.18
● class
● o
● a
● h
● c
● FORCE
● so
● S
● s
Interpreter
Concern Sieve
Select a state

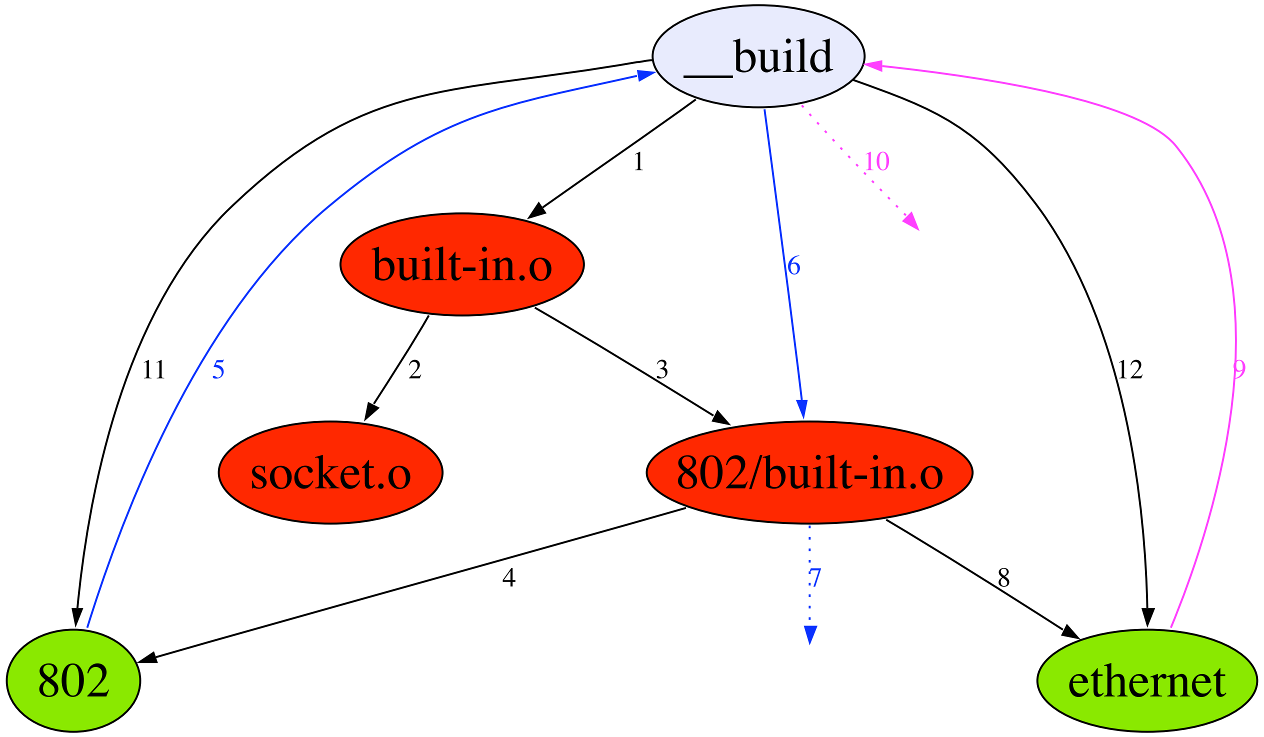
```
>>> center
>>>
```

FORCE



composite object



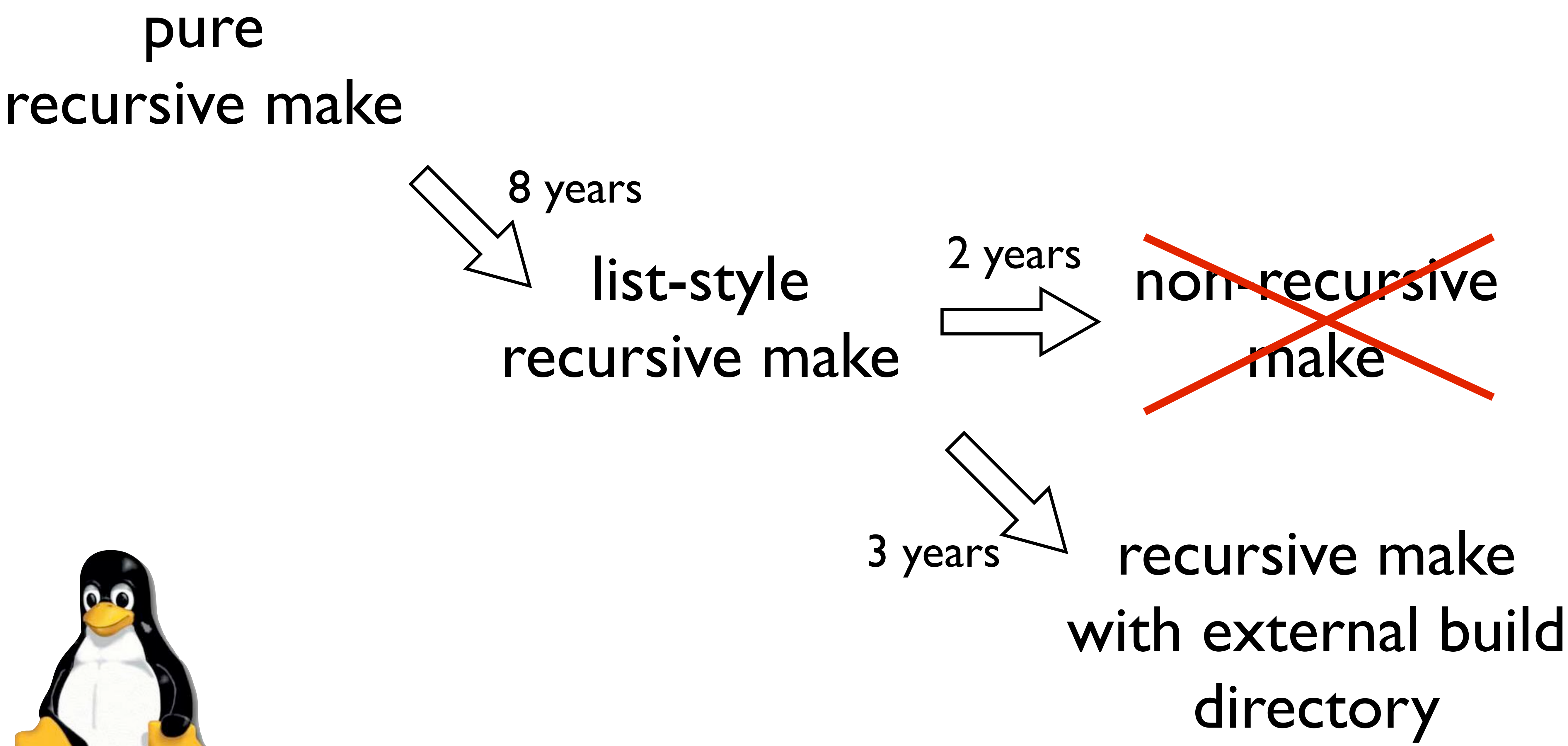


circular dependency chain

# Build Idioms

1. Research Hypothesis
2. Tool Support to Understand Build Systems
3. Evolution of Linux Kernel Build System
4. Conceptual Reasons of Co-evolution
5. The Pitfalls of PhD Research
6. Conclusion

# 1. **Modular** source code needs a **modular** build system



pure recursive make

8 years

list-style recursive make

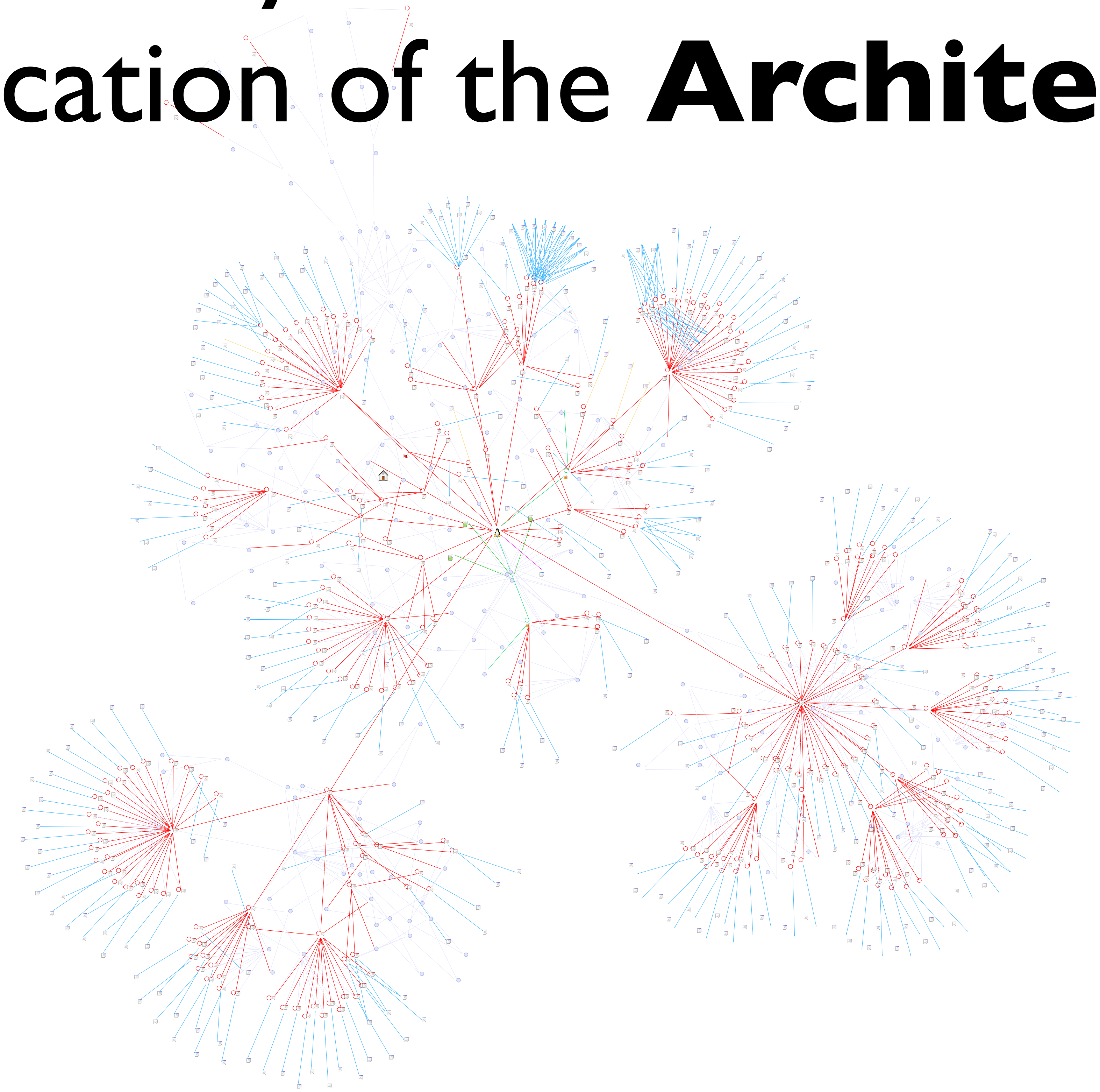
2 years

~~non-recursive make~~

3 years

recursive make with external build directory

# 2. The Build System is an **Executable** Specification of the **Architecture**



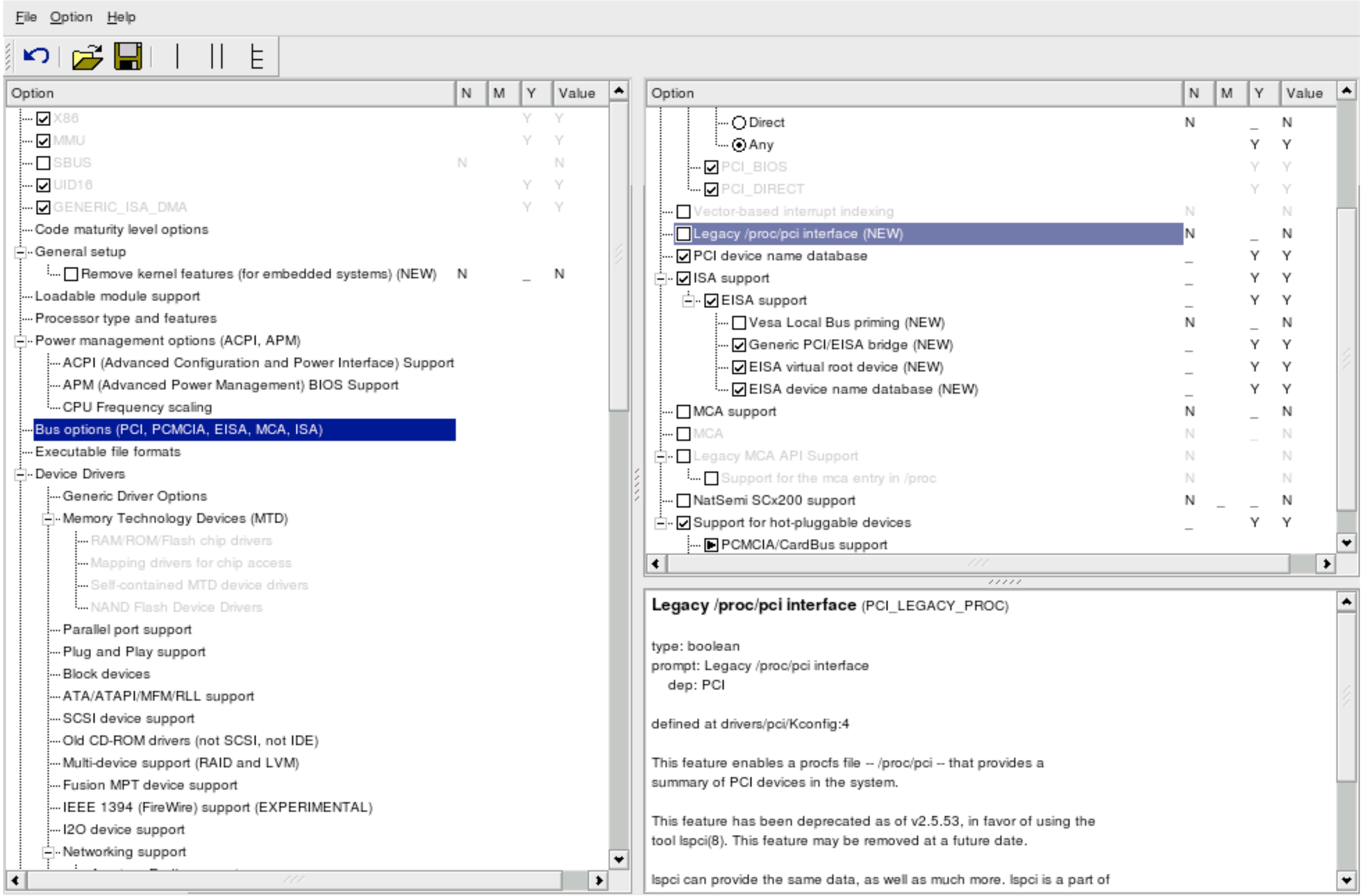
# 3. **Correctness** Trumps **Efficiency**

speculatively removing source code dependencies to **speed up** the build

⟺

**inconsistent** build products

# 4. Configuration Layer **Controls** the Static **Variability** of Source Code



| Option | N | M | Y | Value |
|---|---|---|---|---|
| ☑ X86 | | | Y | Y |
| ☑ MMU | | | Y | Y |
| ☐ SBUS | N | | | N |
| ☑ UID16 | | | Y | Y |
| ☑ GENERIC_ISA_DMA | | | Y | Y |
| Code maturity level options | | | | |
| General setup | | | | |
| ☐ Remove kernel features (for embedded systems) (NEW) | N | | _ | N |
| Loadable module support | | | | |
| Processor type and features | | | | |
| Power management options (ACPI, APM) | | | | |
| ACPI (Advanced Configuration and Power Interface) Support | | | | |
| APM (Advanced Power Management) BIOS Support | | | | |
| CPU Frequency scaling | | | | |
| Bus options (PCI, PCMCIA, EISA, MCA, ISA) | | | | |
| Executable file formats | | | | |
| Device Drivers | | | | |
| Generic Driver Options | | | | |
| Memory Technology Devices (MTD) | | | | |
| RAM/ROM/Flash chip drivers | | | | |
| Mapping drivers for chip access | | | | |
| Self-contained MTD device drivers | | | | |
| NAND Flash Device Drivers | | | | |
| Parallel port support | | | | |
| Plug and Play support | | | | |
| Block devices | | | | |
| ATA/ATAPI/MFM/RLL support | | | | |
| SCSI device support | | | | |
| Old CD-ROM drivers (not SCSI, not IDE) | | | | |
| Multi-device support (RAID and LVM) | | | | |
| Fusion MPT device support | | | | |
| IEEE 1394 (FireWire) support (EXPERIMENTAL) | | | | |
| I2O device support | | | | |
| Networking support | | | | |

| Option | N | M | Y | Value |
|---|---|---|---|---|
| ○ Direct | N | | _ | N |
| ● Any | | | Y | Y |
| ☑ PCI_BIOS | | | Y | Y |
| ☑ PCI_DIRECT | | | Y | Y |
| ☐ Vector-based interrupt indexing | N | | | N |
| ☐ Legacy /proc/pci interface (NEW) | N | | _ | N |
| ☑ PCI device name database | _ | | Y | Y |
| ☑ ISA support | _ | | Y | Y |
| ☑ EISA support | _ | | Y | Y |
| ☐ Vesa Local Bus priming (NEW) | N | | _ | N |
| ☑ Generic PCI/EISA bridge (NEW) | _ | | Y | Y |
| ☑ EISA virtual root device (NEW) | _ | | Y | Y |
| ☑ EISA device name database (NEW) | _ | | Y | Y |
| ☐ MCA support | N | | _ | N |
| ☐ MCA | N | | _ | N |
| ☐ Legacy MCA API Support | N | | | N |
| ☐ Support for the mca entry in /proc | N | | | N |
| ☐ NatSemi SCx200 support | N | _ | _ | N |
| ☑ Support for hot-pluggable devices | _ | | Y | Y |
| ▶ PCMCIA/CardBus support | | | | |

**Legacy /proc/pci interface** (PCI_LEGACY_PROC)

type: boolean
prompt: Legacy /proc/pci interface
  dep: PCI

defined at drivers/pci/Kconfig:4

This feature enables a procfs file -- /proc/pci -- that provides a summary of PCI devices in the system.

This feature has been deprecated as of v2.5.53, in favor of using the tool lspci(8). This feature may be removed at a future date.

lspci can provide the same data, as well as much more. lspci is a part of

1. Research Hypothesis
2. Tool Support to Understand Build Systems
3. Evolution of Linux Kernel Build System
4. Conceptual Reasons of Co-evolution
5. The Pitfalls of PhD Research
6. Conclusion

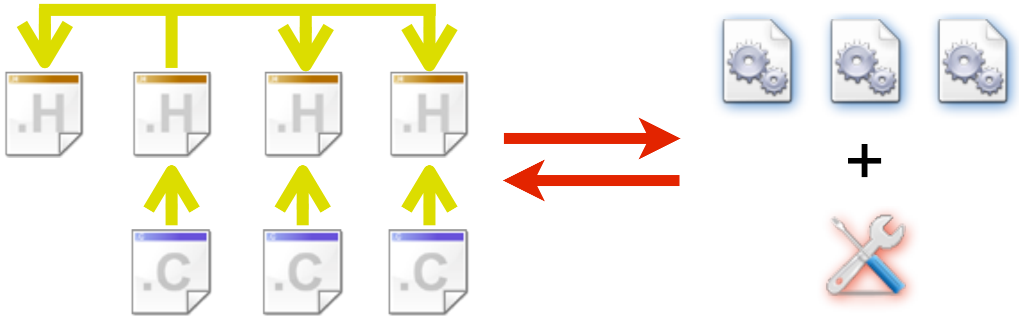
# I'm Grateful my Supervisors ...

- gave me the freedom to develop my "hobby project" into a PhD dissertation
- stimulated me to attend conferences and workshops
- taught me to learn from rejected papers

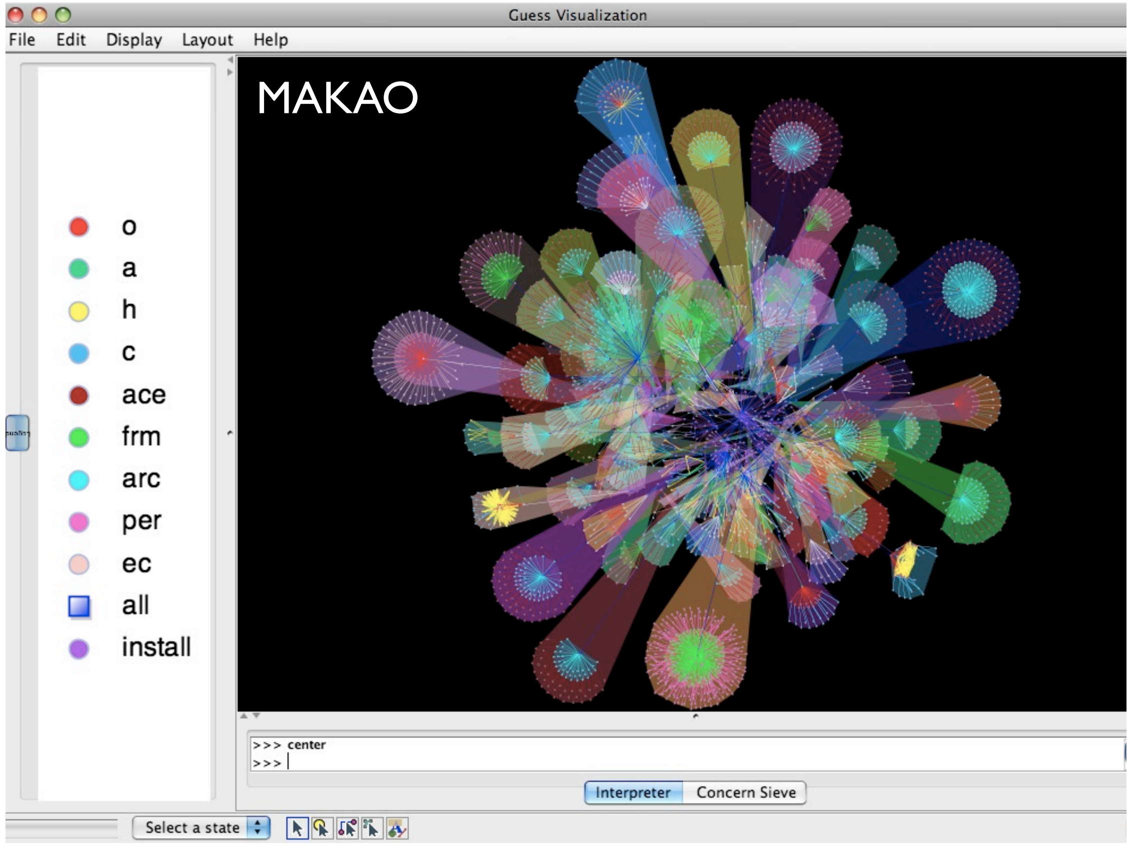
# I Should Have Known that ...

- a concise dissertation is more impressive than a wordy one ;-)
- even vegetarians like salami slicing
- statistics is your friend

1. Research Hypothesis
2. Tool Support to Understand Build Systems
3. Evolution of Linux Kernel Build System
4. Conceptual Reasons of Co-evolution
5. The Pitfalls of PhD Research
6. Conclusion



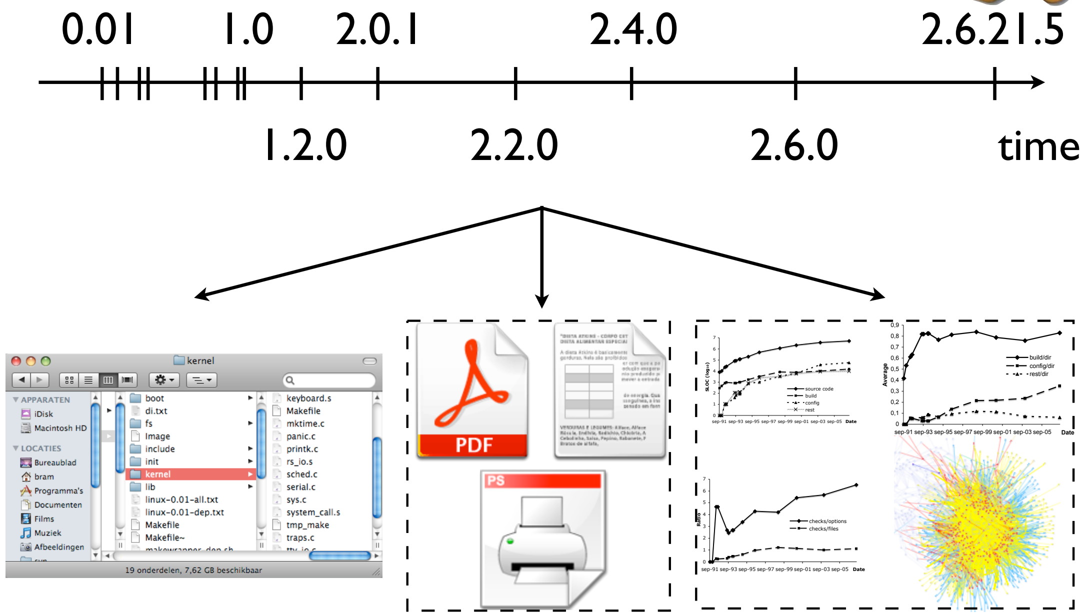
**Source Code** **Build System**



```
>>> center
>>> |
```

# Questions?

## Linux Case Study



## Conceptual Reasons of Co-evolution

1. **Modular** source code needs a **modular** build system
2. The Build System is an **Executable** Specification of the **Architecture**
3. **Correctness** Trumps **Efficiency**
4. Configuration Layer **Controls** the Static **Variability** of Source Code