Coevolution of Variability Models and Related Artifacts

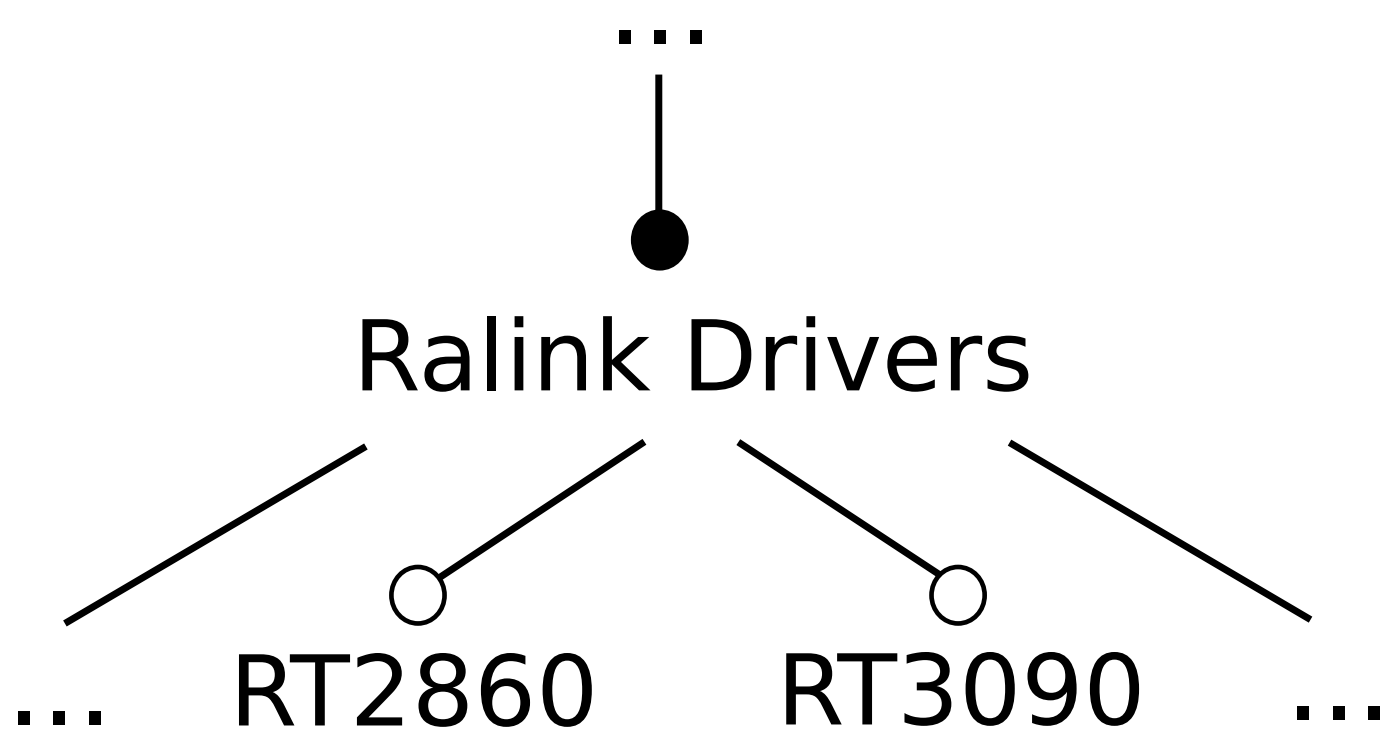
A Case Study from the Linux Kernel

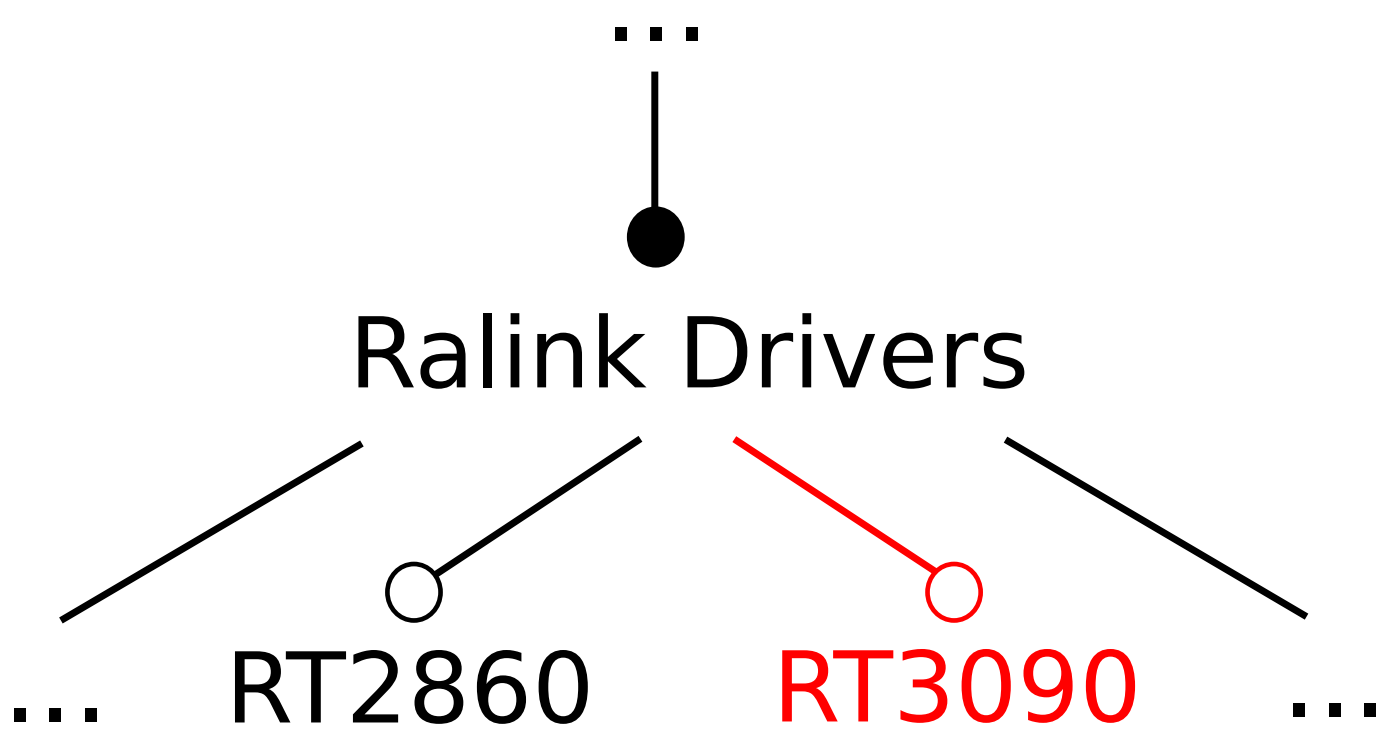
Leonardo Passos[1] Jianmei Guo[1] Leopoldo Teixeira[2]
Krzysztof Czarnecki[1] Andrzej Wąsowski[3] Paulo Borba[2]

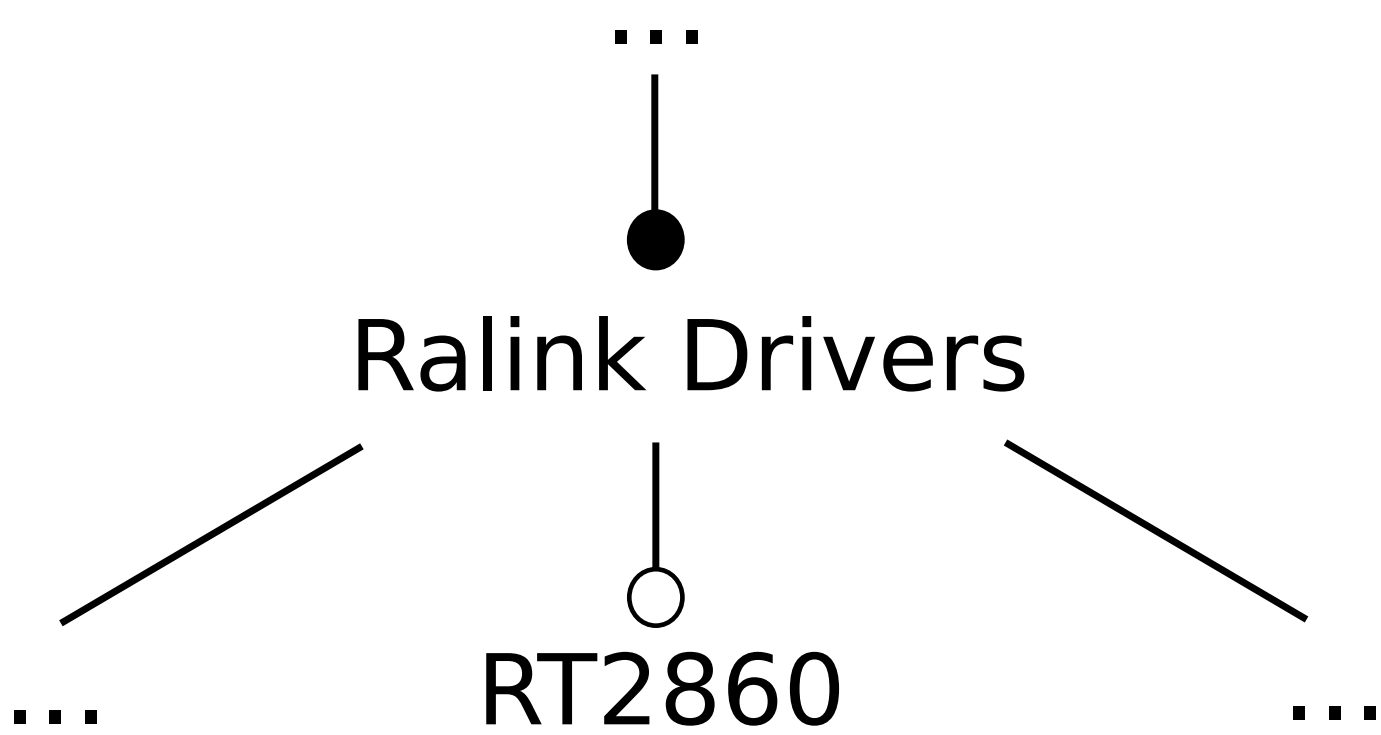
[1]University of Waterloo [2]Federal University of Pernambuco [3]IT University

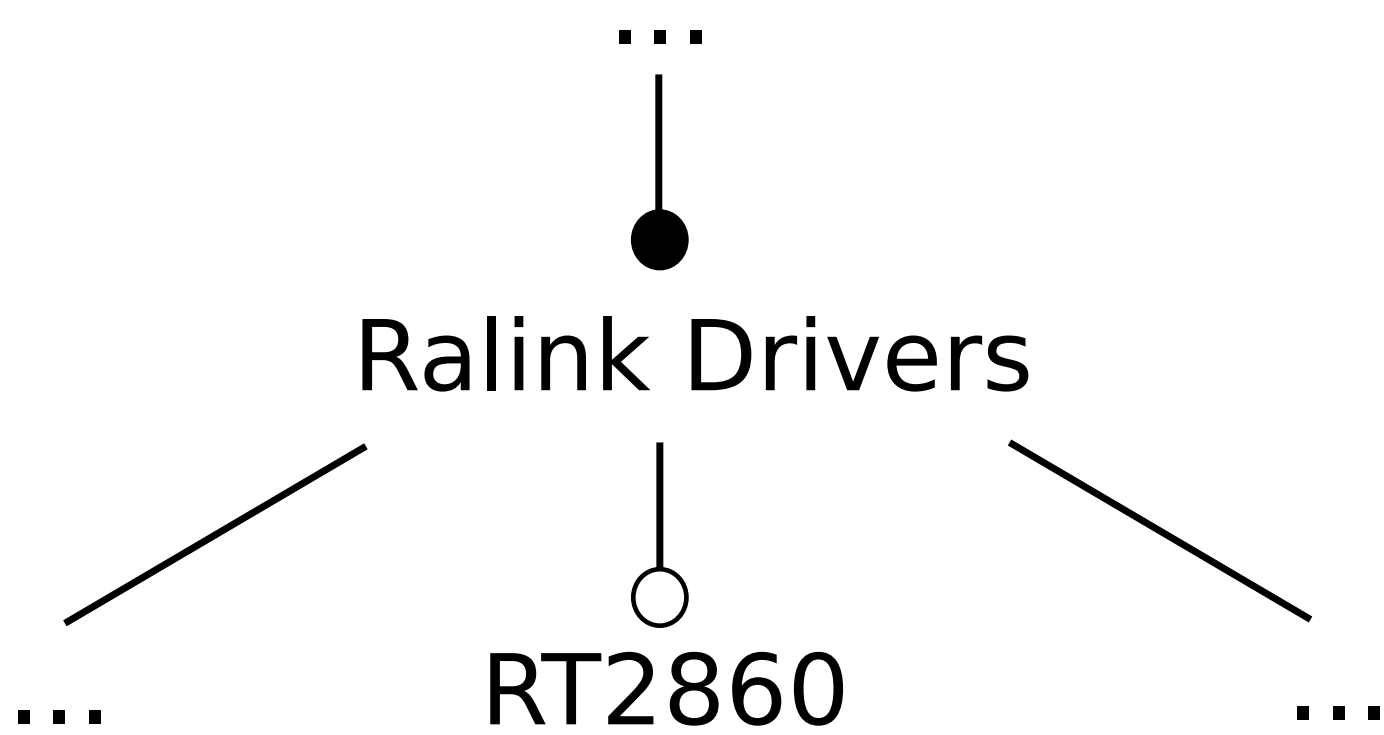
17th International Software Product Line Conference

A concrete change from the Linux kernel


...
Ralink Drivers
...
RT2860
RT3090
...


...
Ralink Drivers
...
RT2860
RT3090
...

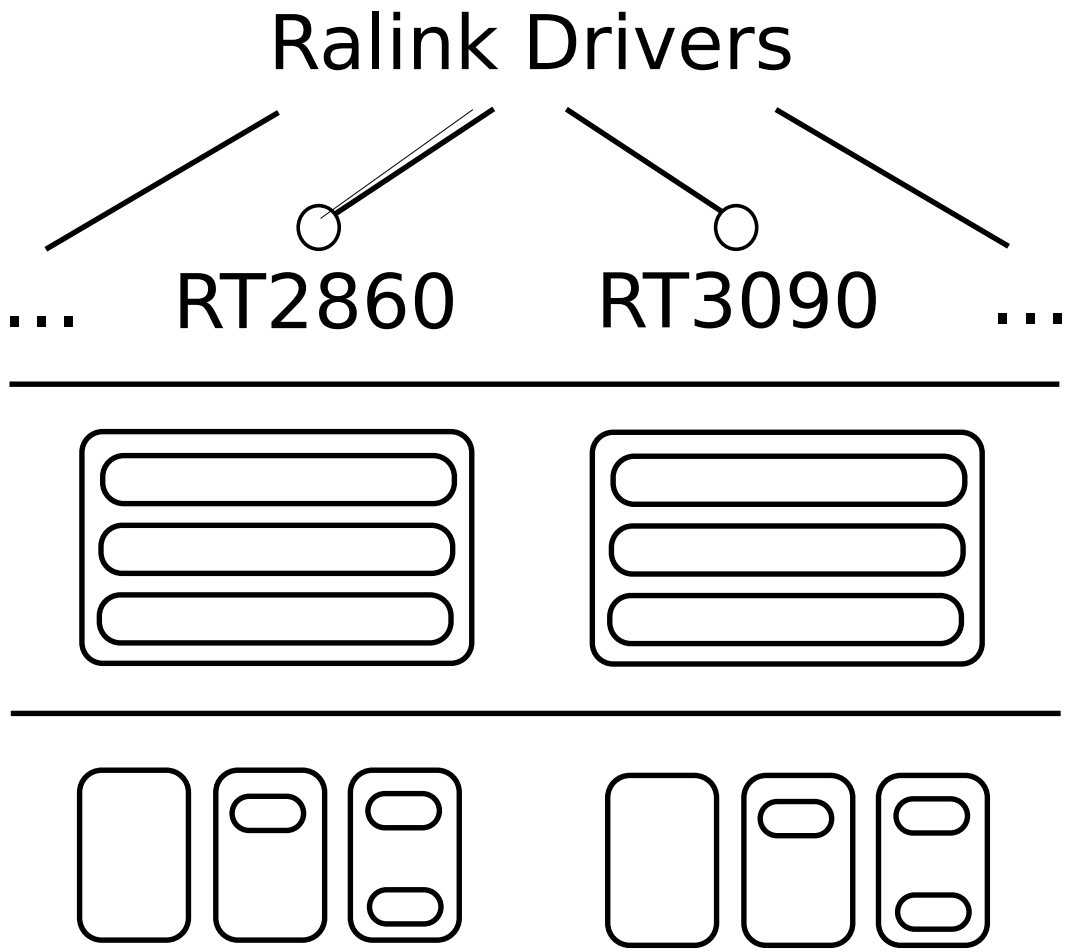

...
Ralink Drivers
...
RT2860
...

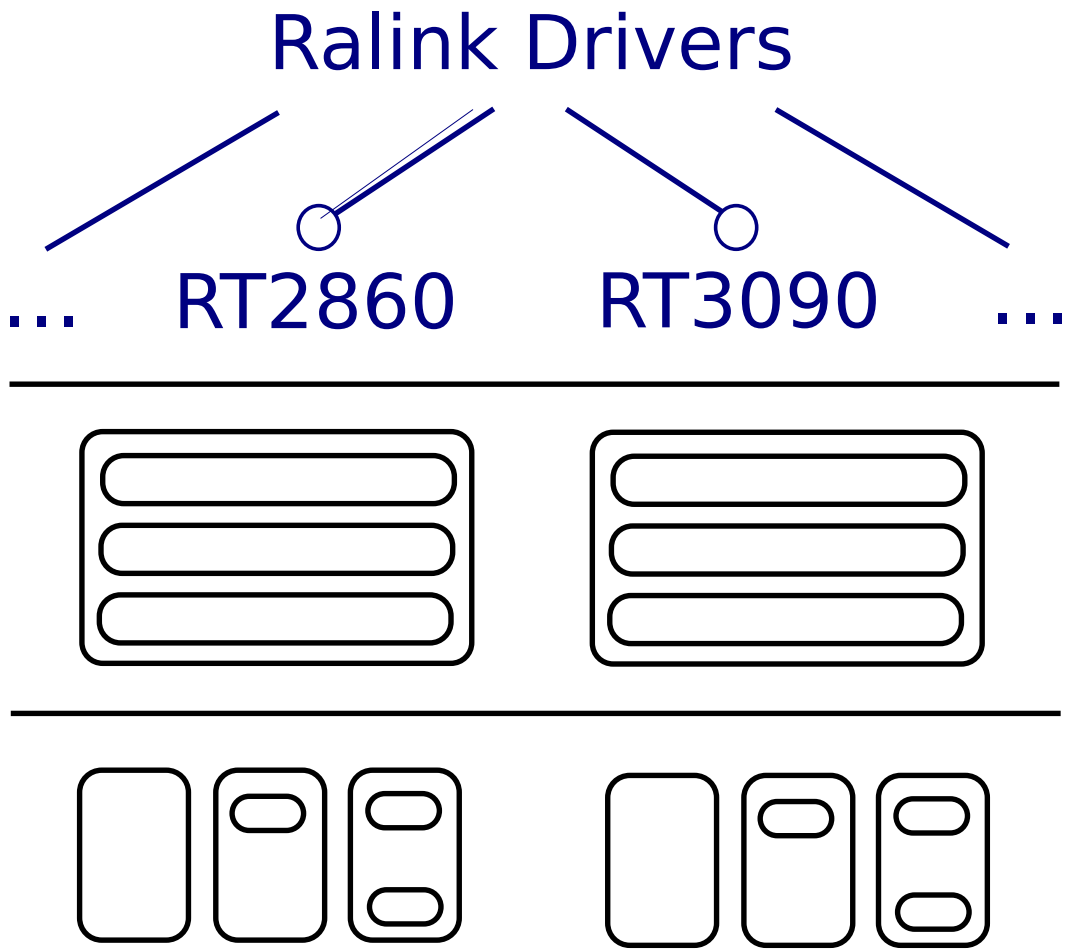


Does it mean that RT3090 is no longer supported?

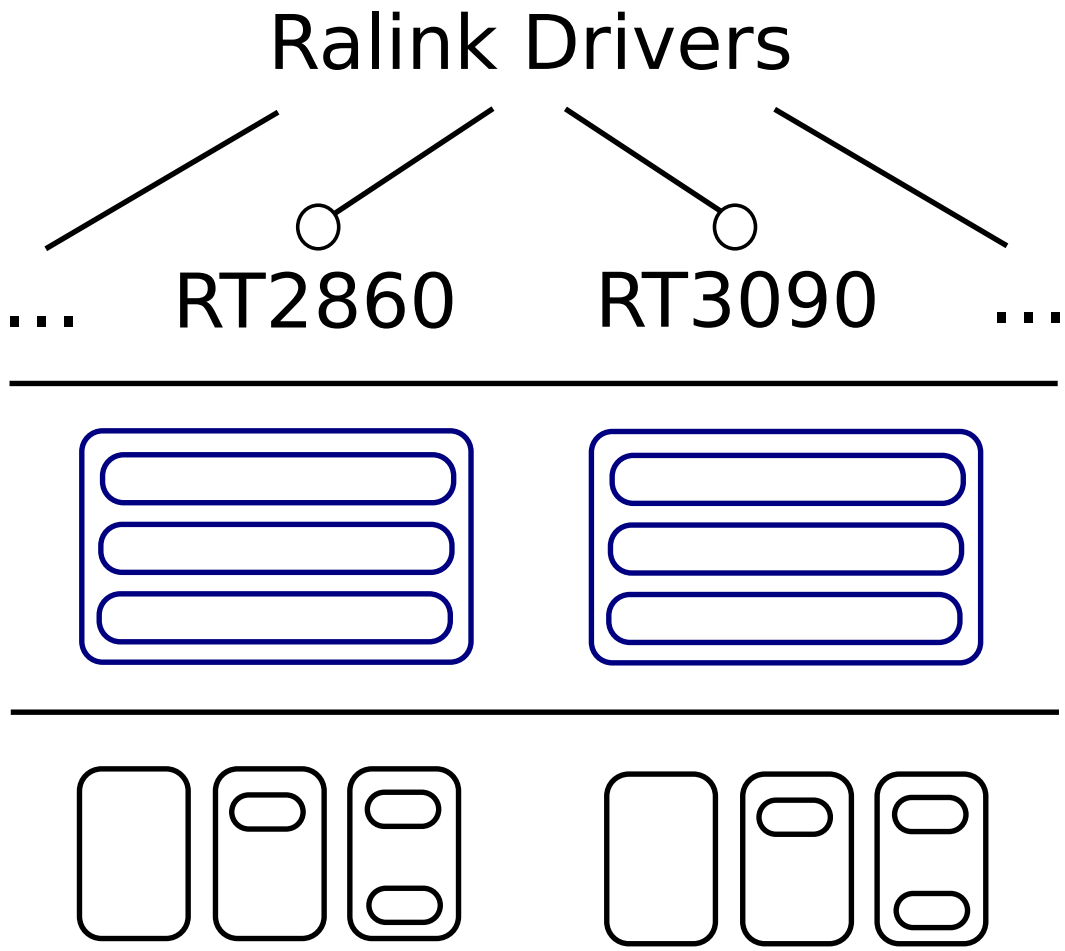
Existing evolution studies tend to focus on the variability model alone

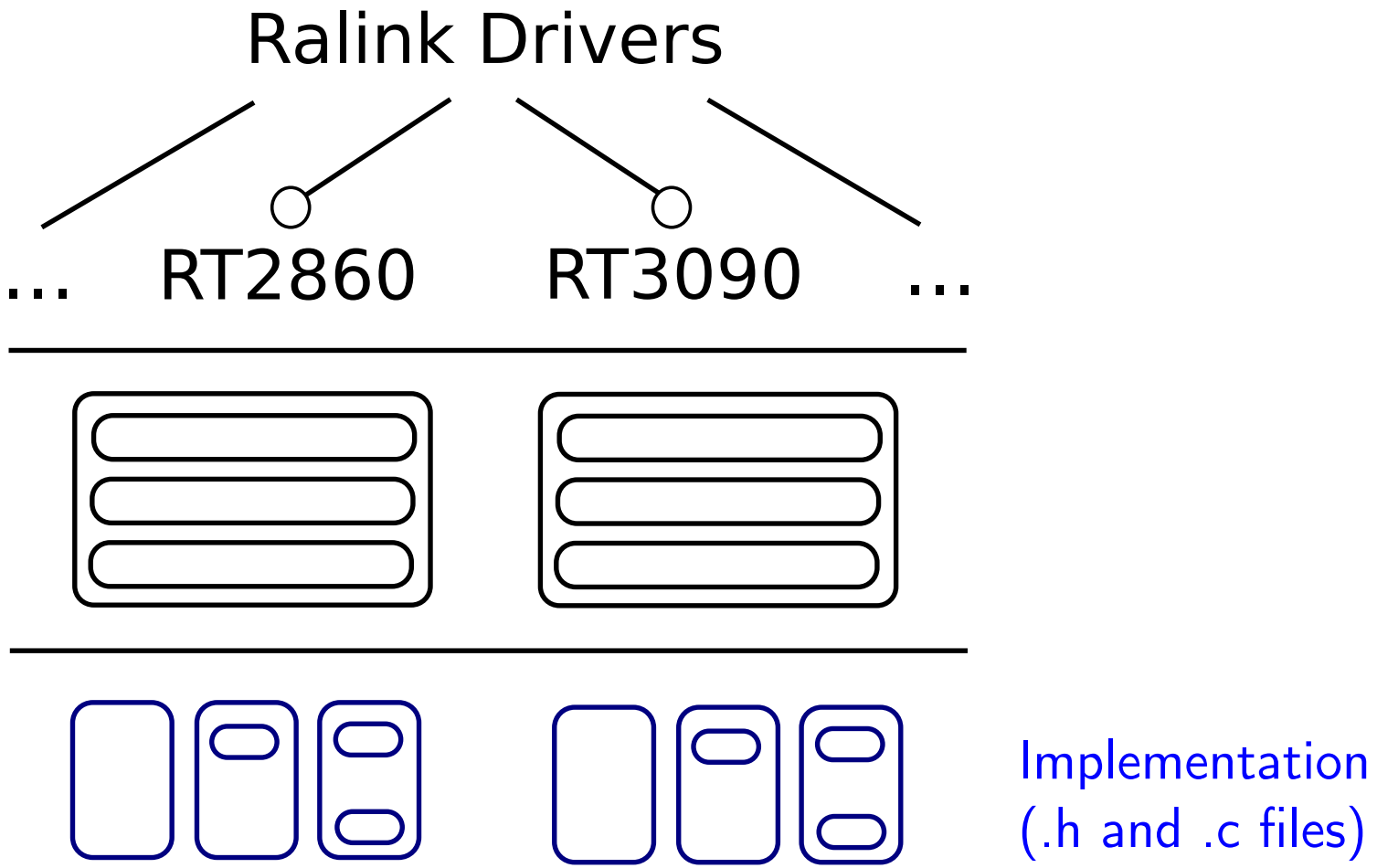
That doesn't tell the whole story...


Ralink Drivers
...
RT2860
RT3090
...

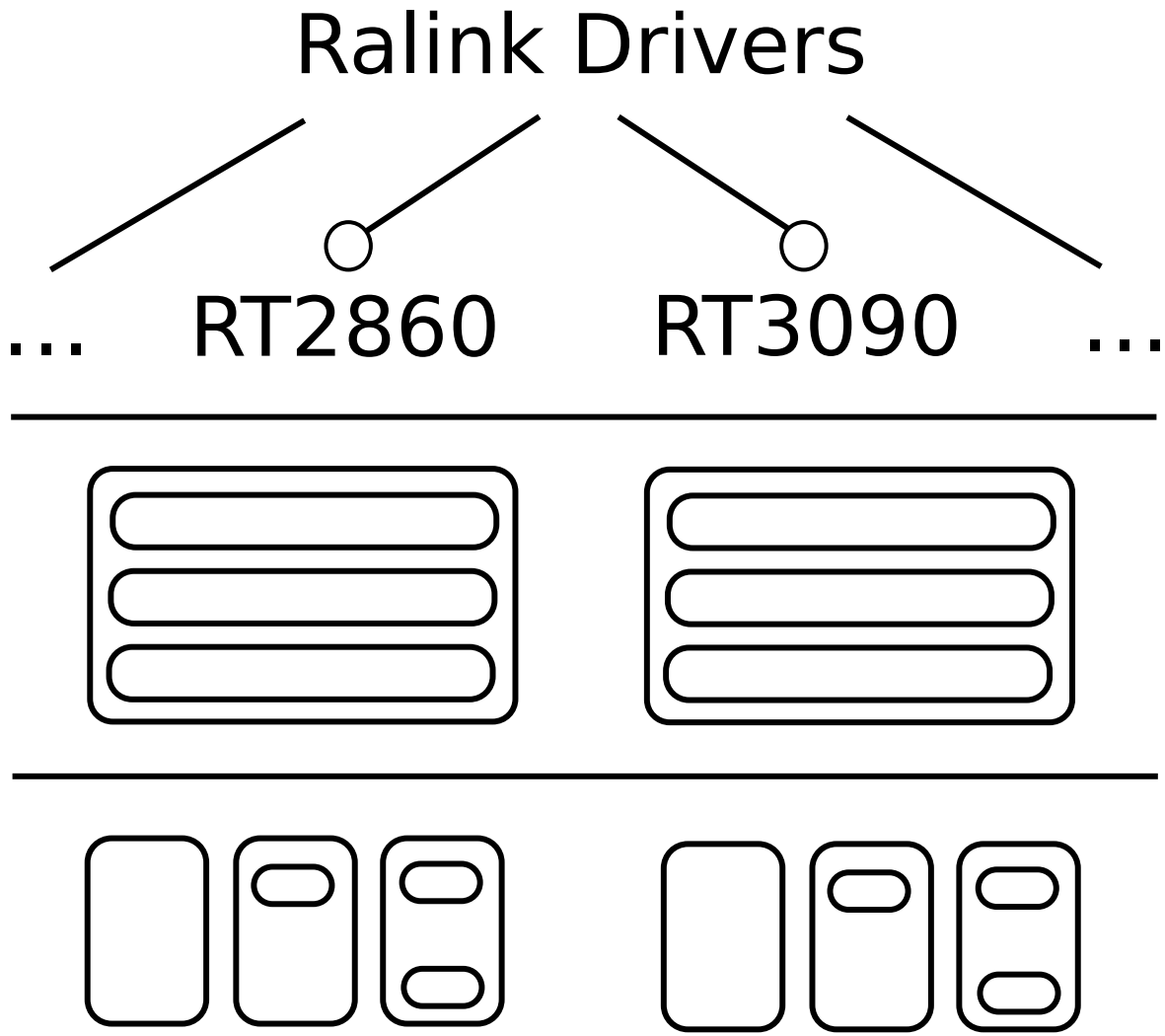


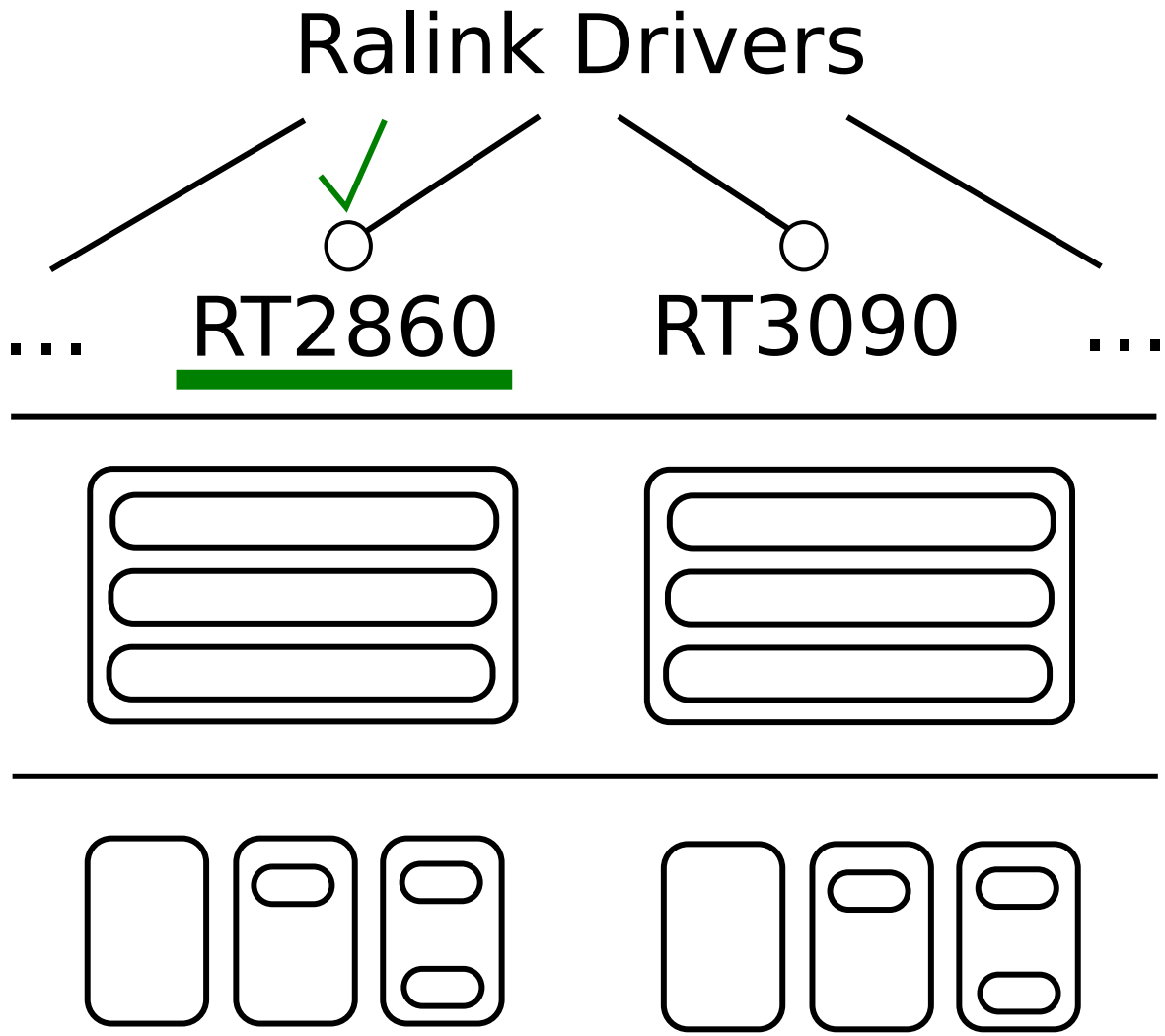
Variability model
(Kconfig)

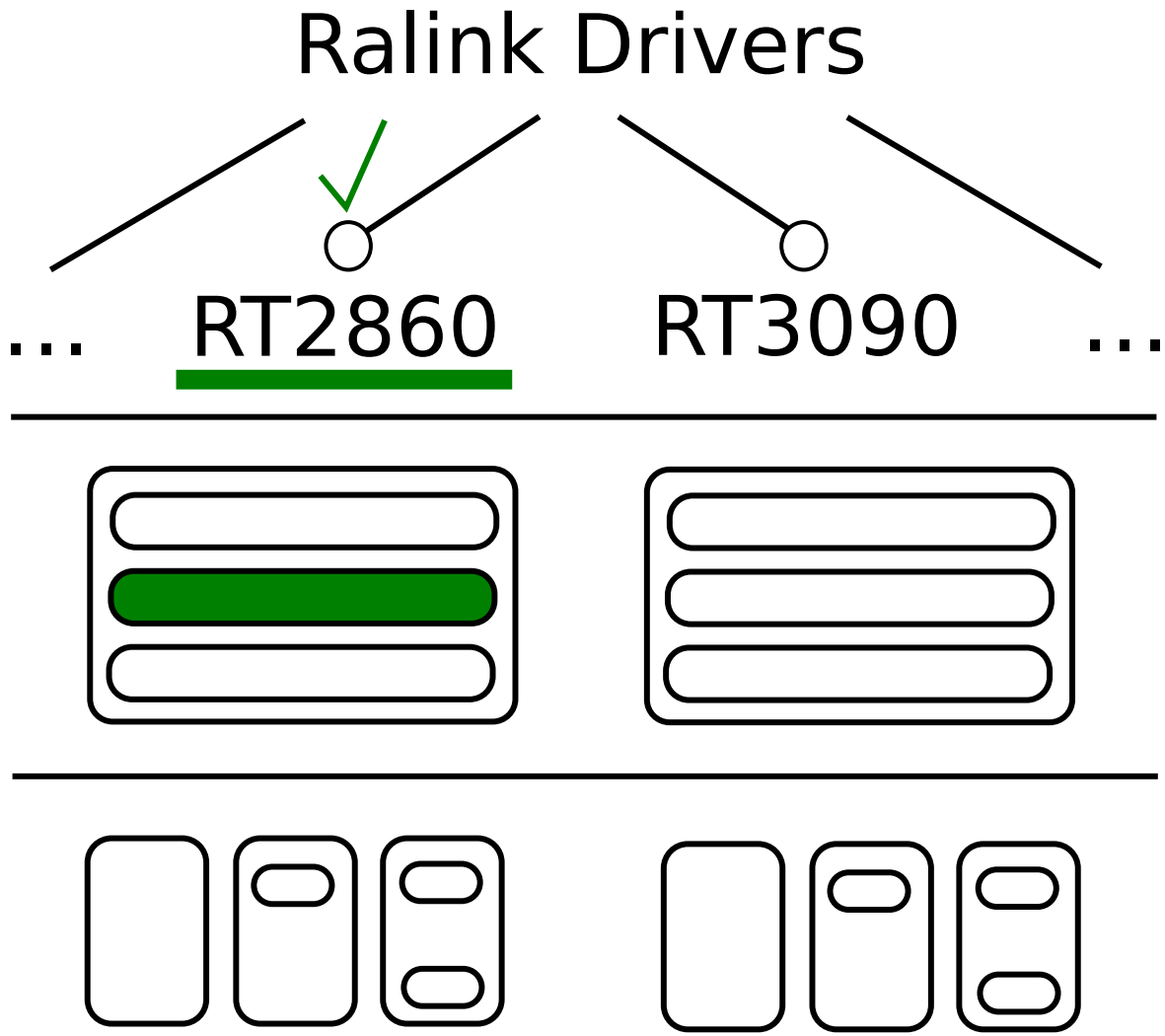

Ralink Drivers
...
RT2860
RT3090
...
Mapping
(Makefiles)

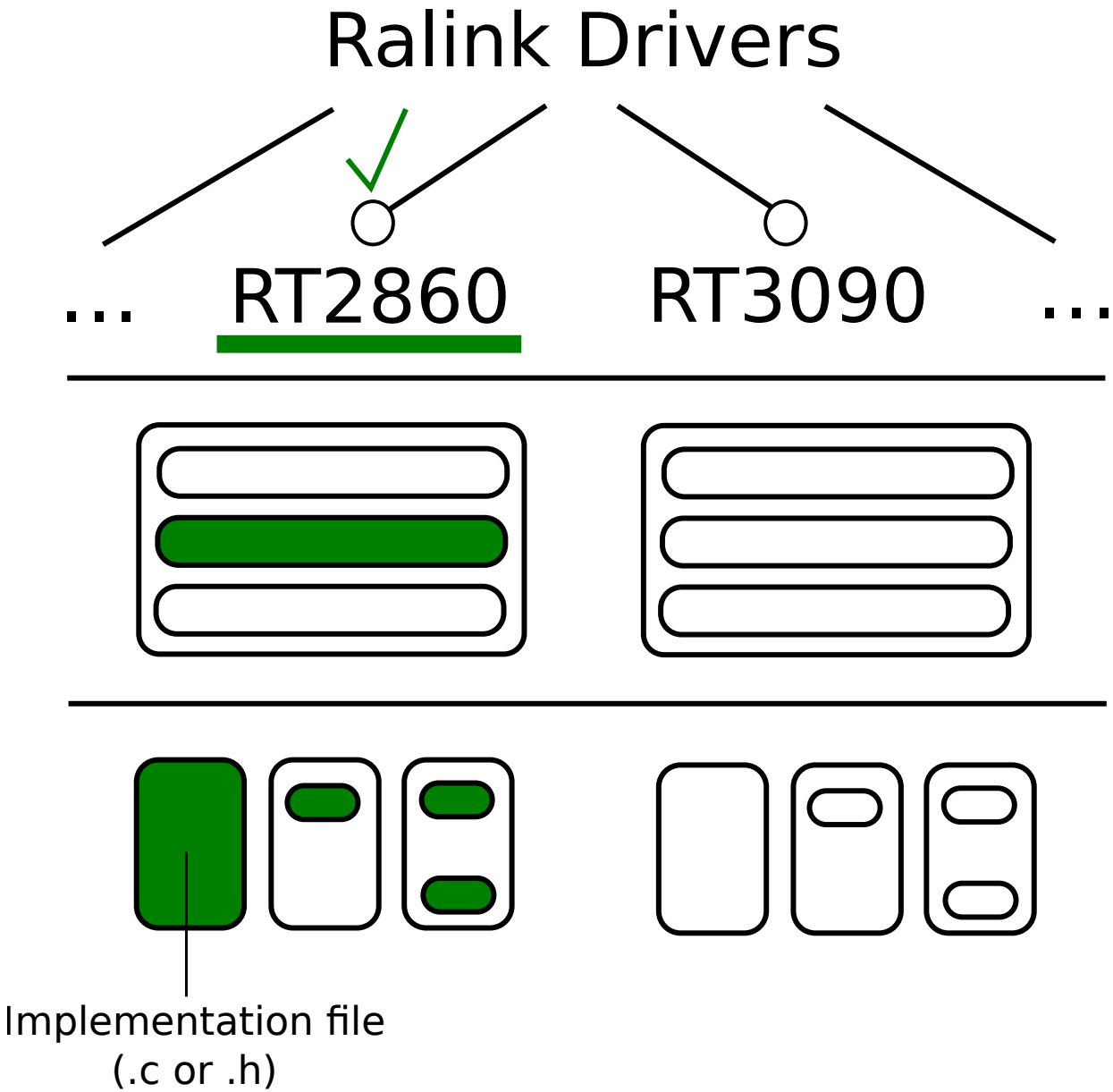

Ralink Drivers
...
RT2860
RT3090
...
Implementation
(.h and .c files)

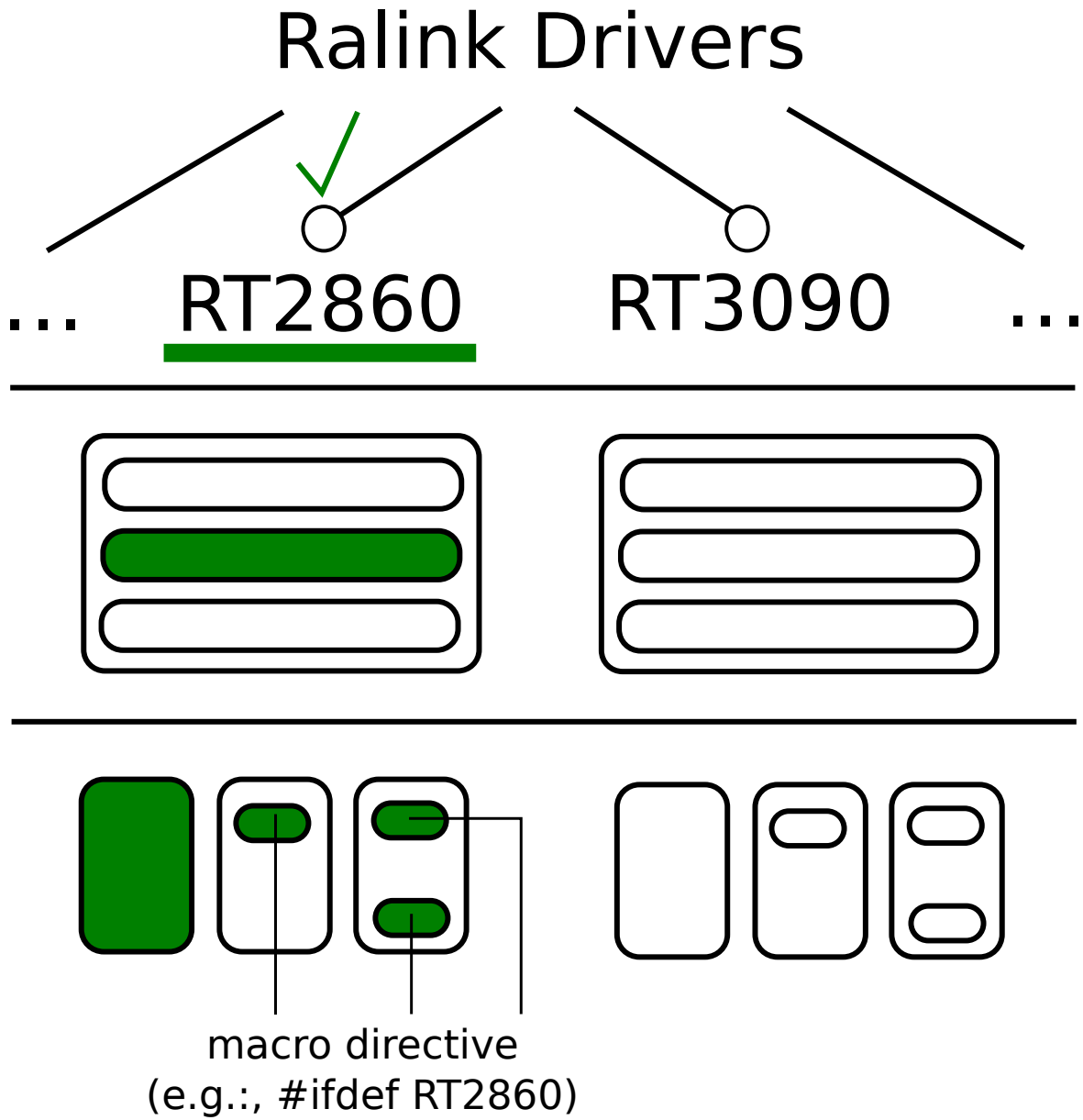
Different artifacts, in different spaces, are connected. . .


Ralink Drivers
...
RT2860
RT3090
...

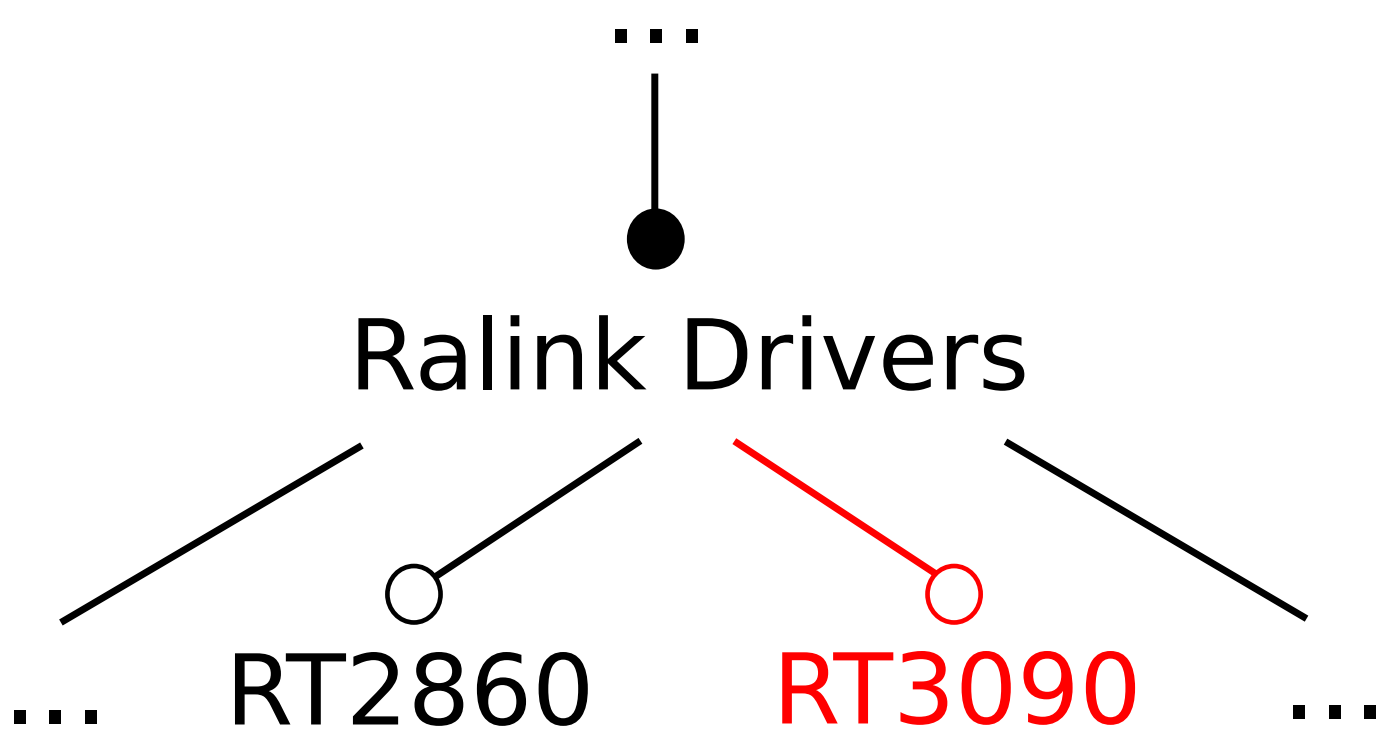

Ralink Drivers
...
RT2860
RT3090
...


Ralink Drivers
...
RT2860
RT3090
...

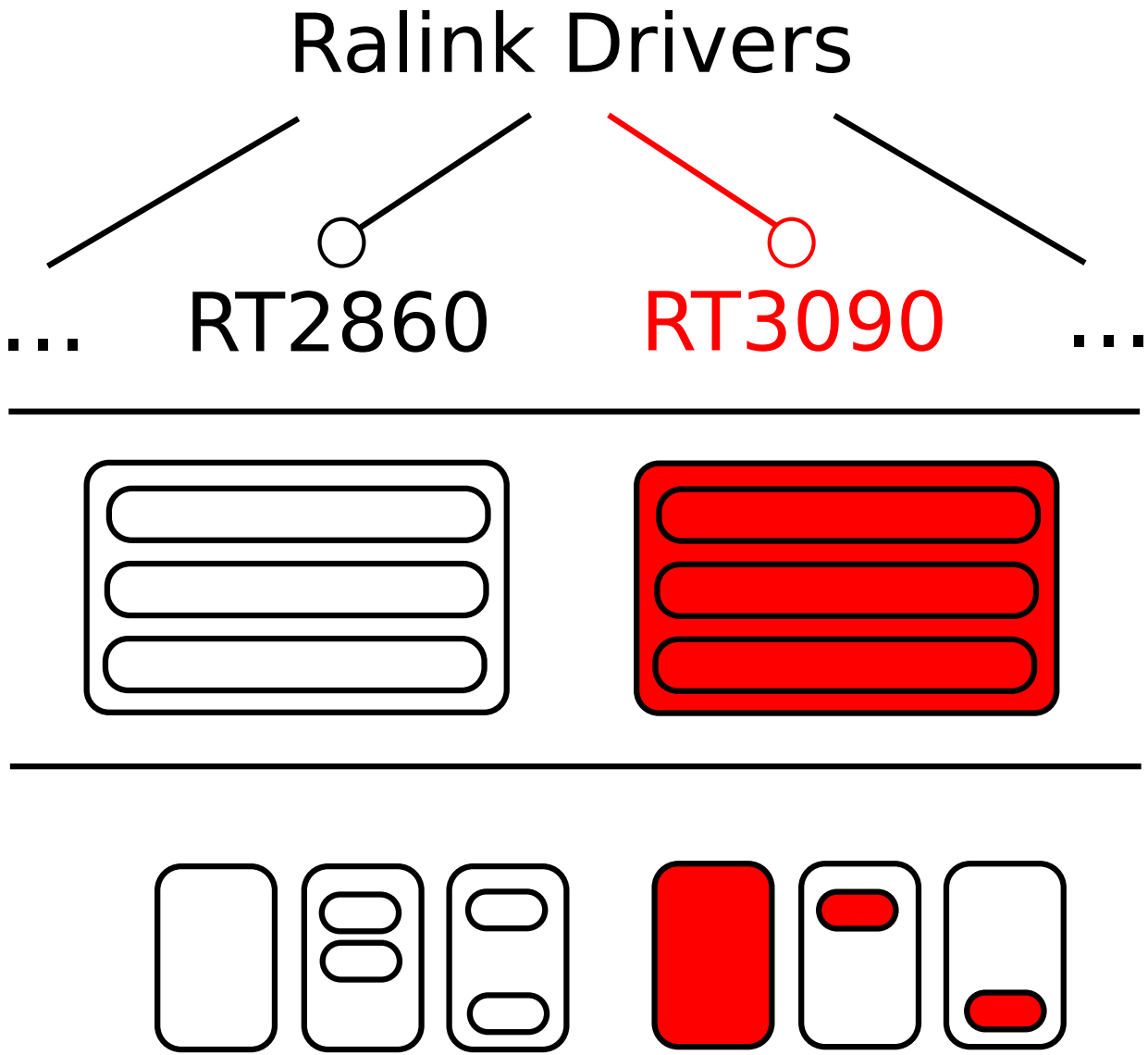

Ralink Drivers
...
RT2860
RT3090
...
Implementation file
(.c or .h)

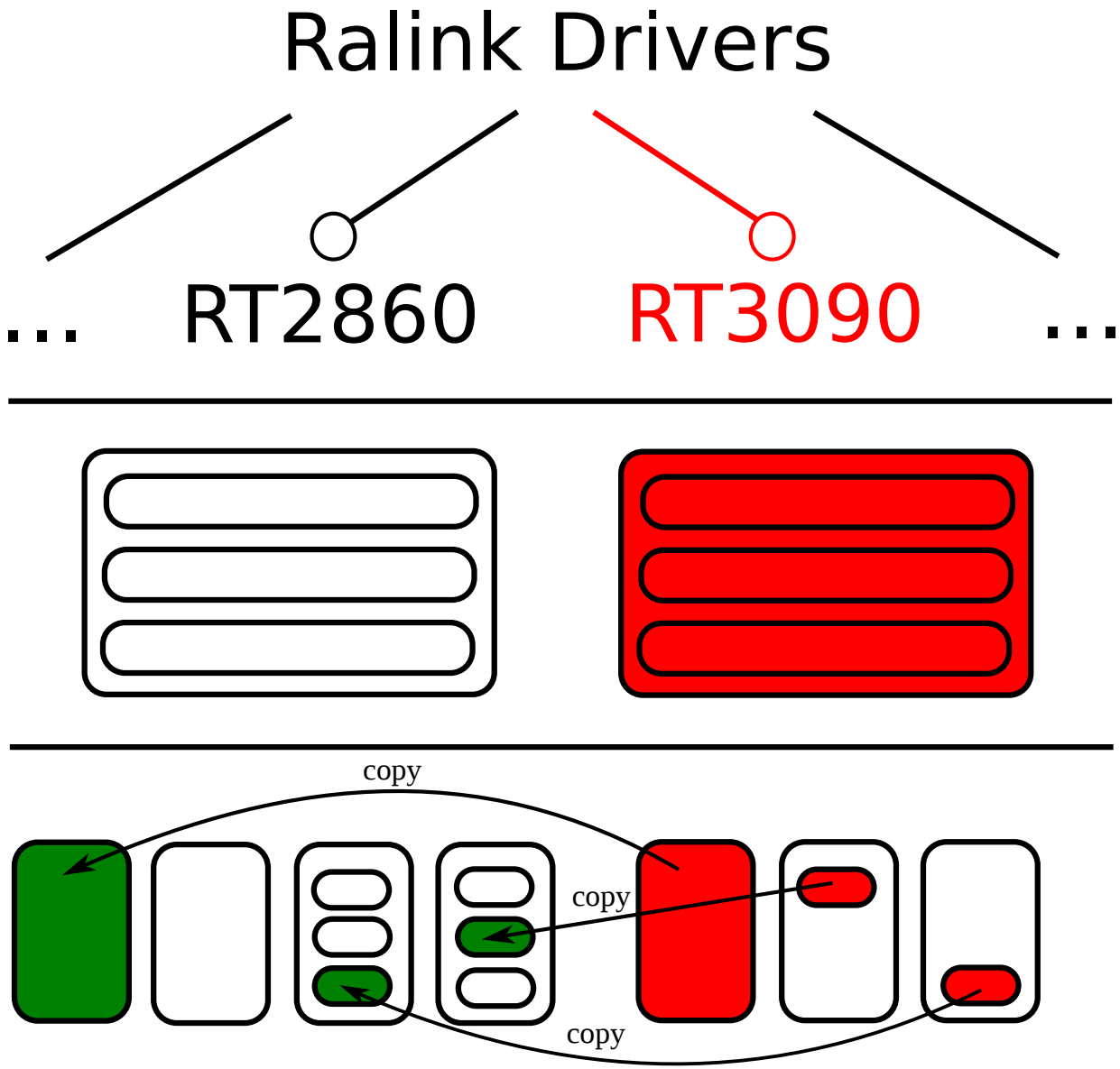

Ralink Drivers
...
RT2860
RT3090
...
macro directive
(e.g.:, #ifdef RT2860)

With the three spaces in mind, the real picture of . . .

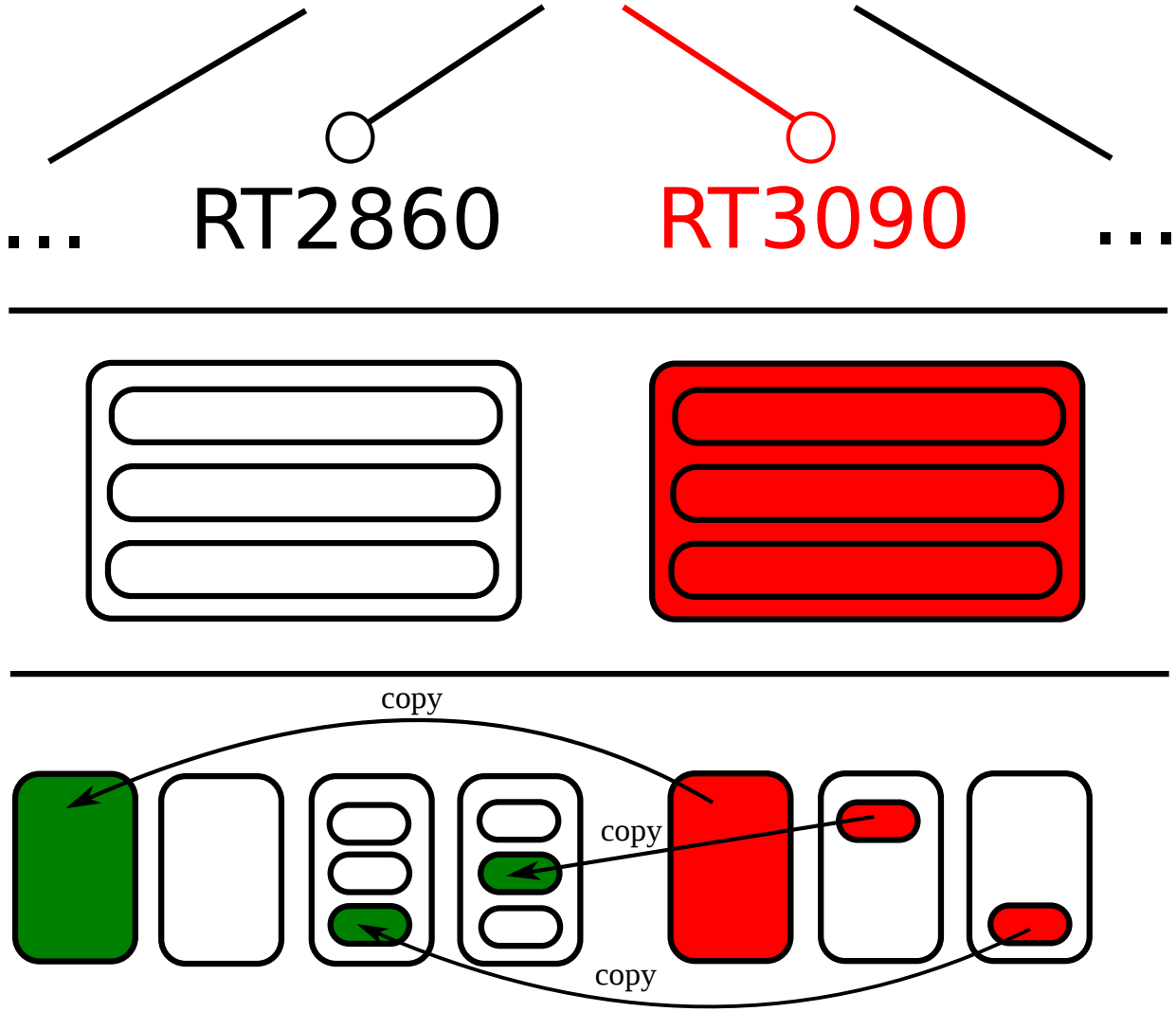


is


Ralink Drivers
...
RT2860
RT3090
...


Ralink Drivers
...
RT2860
RT3090
...
copy
copy
copy

Ralink Drivers



Therefore, RT3090 is merged into RT2860

But, what can be said about the few studies that take coevolution into account?

Studies that consider coevolution

- Borba et al., GPCE 2011

Studies that consider coevolution

- Borba et al., GPCE 2011
  - Small case study: Mobile Media and TaRGeT $\approx$ 40 features

Studies that consider coevolution

- Borba et al., GPCE 2011
  - Small case study: Mobile Media and TaRGeT $\approx$ 40 features
  - Unlikely to capture the complexity typically found in large systems

Studies that consider coevolution

- Borba et al., GPCE 2011
  - Small case study: Mobile Media and TaRGeT $\approx$ 40 features
  - Unlikely to capture the complexity typically found in large systems
  - Case study restricted to refinement changes (guarantee product compatibility)

Studies that consider coevolution

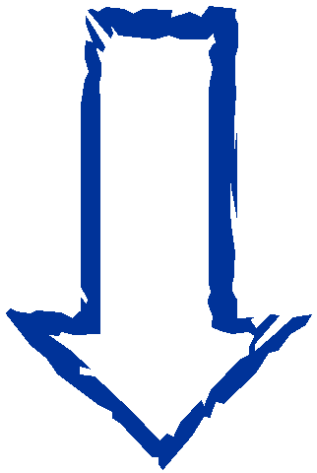
- Borba et al., GPCE 2011
  - Small case study: Mobile Media and TaRGeT $\approx$ 40 features
  - Unlikely to capture the complexity typically found in large systems
  - Case study restricted to refinement changes (guarantee product compatibility)
- Seidl et al., SPLC 2012

Studies that consider coevolution

- Borba et al., GPCE 2011
  - Small case study: Mobile Media and TaRGeT $\approx$ 40 features
  - Unlikely to capture the complexity typically found in large systems
  - Case study restricted to refinement changes (guarantee product compatibility)
- Seidl et al., SPLC 2012
  - Validated over a small and fictitious example

We need practical case studies of variability coevolution in large complex variability rich software

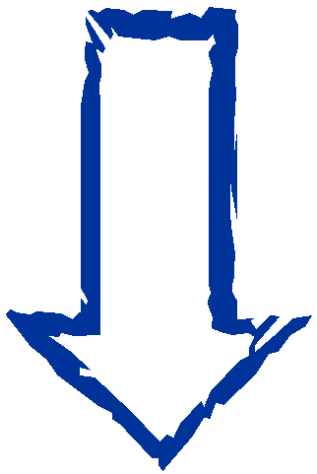
Better understanding



tools mirroring coevolution as it happens in practice

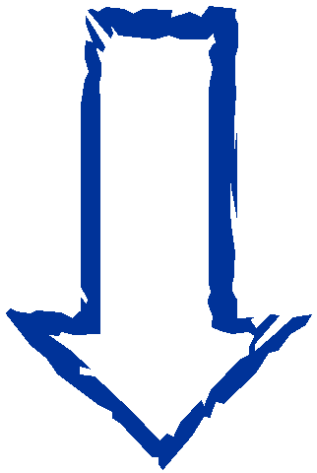
Our work

A catalog of 13 coevolution patterns from a large and complex system (Linux kernel)



Provides a concrete set of coevolution operations performed in practice

A set of findings from the analyses of the catalog and its instances



Presented as *take home lessons*

Linux as a subject of study

- Mature: over 20 years of development

Linux as a subject of study

- Mature: over 20 years of development
- Complex (in v3.3), with over

Linux as a subject of study

- Mature: over 20 years of development
- Complex (in v3.3), with over
  - $12,000$ features

Linux as a subject of study

- Mature: over 20 years of development
- Complex (in v3.3), with over
  - $12,000$ features
  - $80,000$ compile-time variation points

Linux as a subject of study

- Mature: over 20 years of development
- Complex (in v3.3), with over
  - $12,000$ features
  - $80,000$ compile-time variation points
  - $16,000$ Makefiles

Linux as a subject of study

- Mature: over 20 years of development
- Complex (in v3.3), with over
  - $12,000$ features
  - $80,000$ compile-time variation points
  - $16,000$ Makefiles
  - $30,000$ header and C files

Linux as a subject of study

- Changes are kept in a publicly available SCM Repository (git)

Linux as a subject of study

- Changes are kept in a publicly available SCM Repository (git)
- Continuous development

Linux as a subject of study

- Changes are kept in a publicly available SCM Repository (git)
- Continuous development
- Variability spreads different artifacts (spaces):

Linux as a subject of study

- Changes are kept in a publicly available SCM Repository (git)
- Continuous development
- Variability spreads different artifacts (spaces):
  - Variability model: Kconfig

Linux as a subject of study

- Changes are kept in a publicly available SCM Repository (git)
- Continuous development
- Variability spreads different artifacts (spaces):
  - Variability model: Kconfig
  - Mapping: Makefile

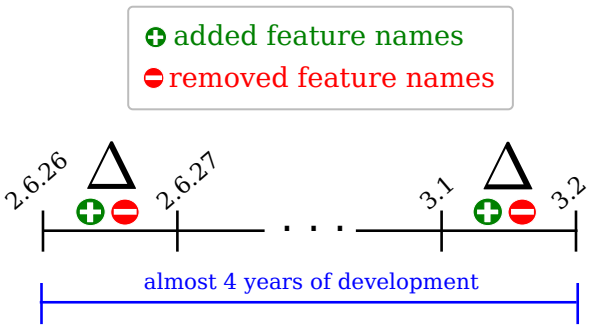
Linux as a subject of study

- Changes are kept in a publicly available SCM Repository (git)
- Continuous development
- Variability spreads different artifacts (spaces):
  - Variability model: Kconfig
  - Mapping: Makefile
  - Implementation: annotated C code (CPP directives: #ifdefs)

Catalog of evolution patterns

Methodology for extracting patterns

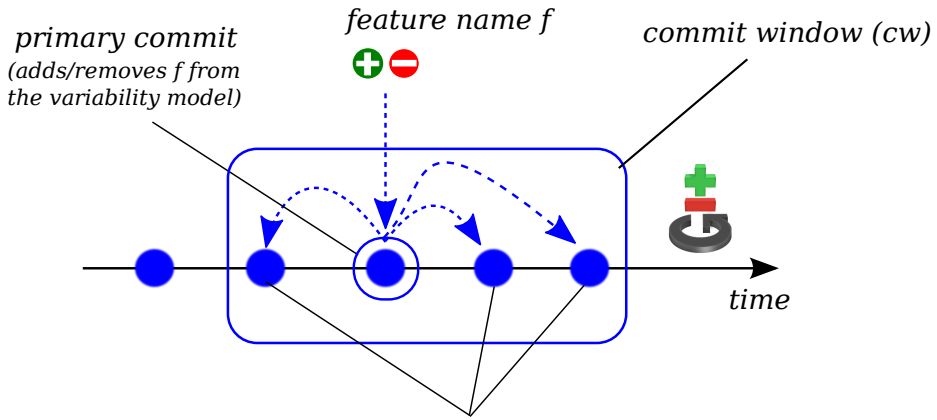
1. Sample Collection



Added feature names: 206 (5%)
(Additions sample)

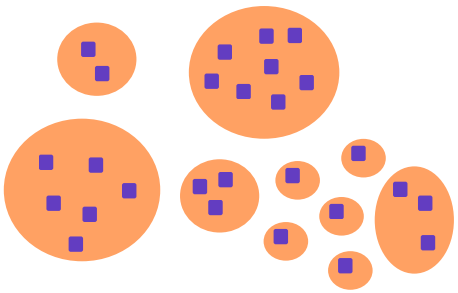
Sample of removed feature names: 101 (10%)
(Removals sample)

2. Commit retrieval

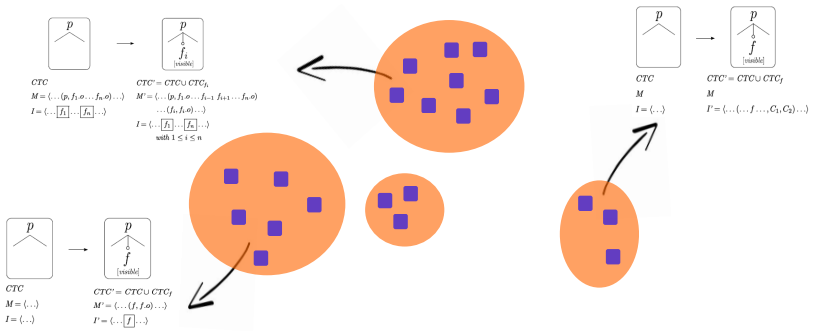


$$\sum |cw_i| \geq 500$$

3. Analysis & Clustering



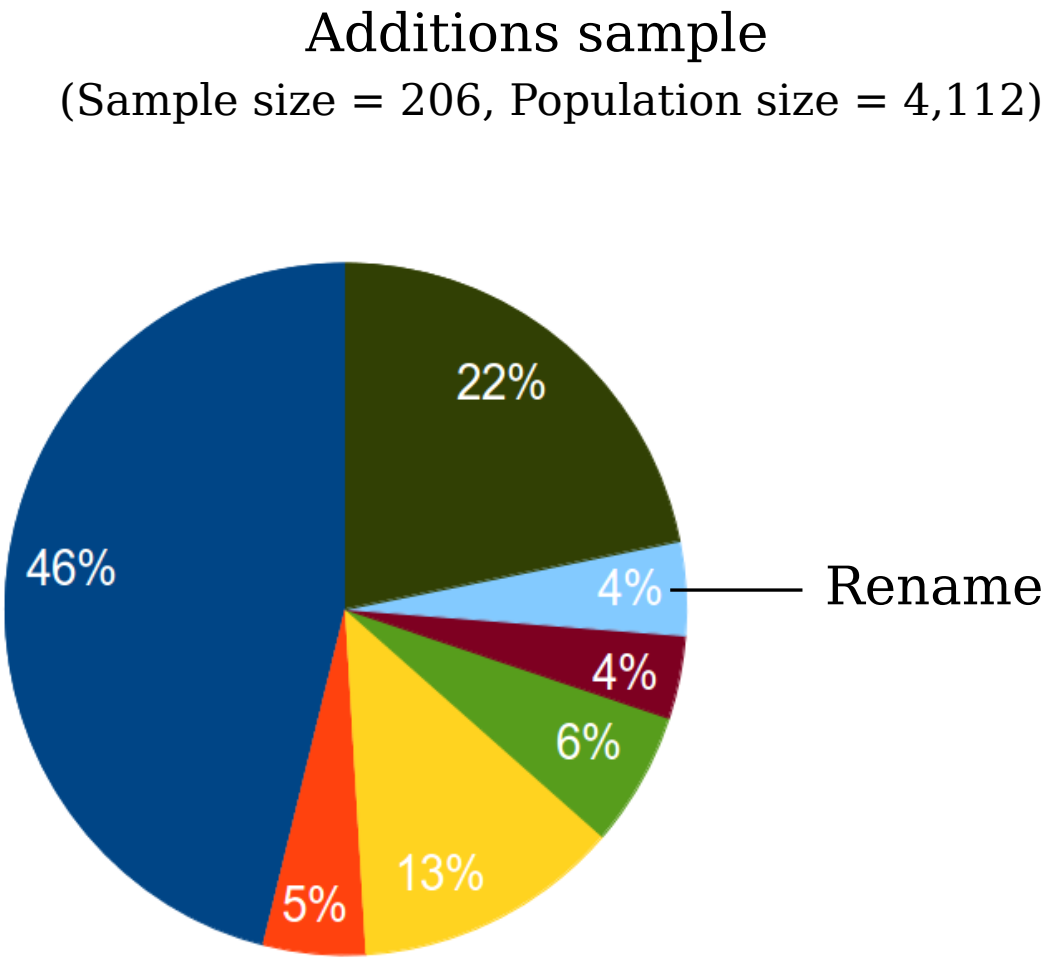
4. Pattern extraction



Catalog of evolution patterns

(additions sample)

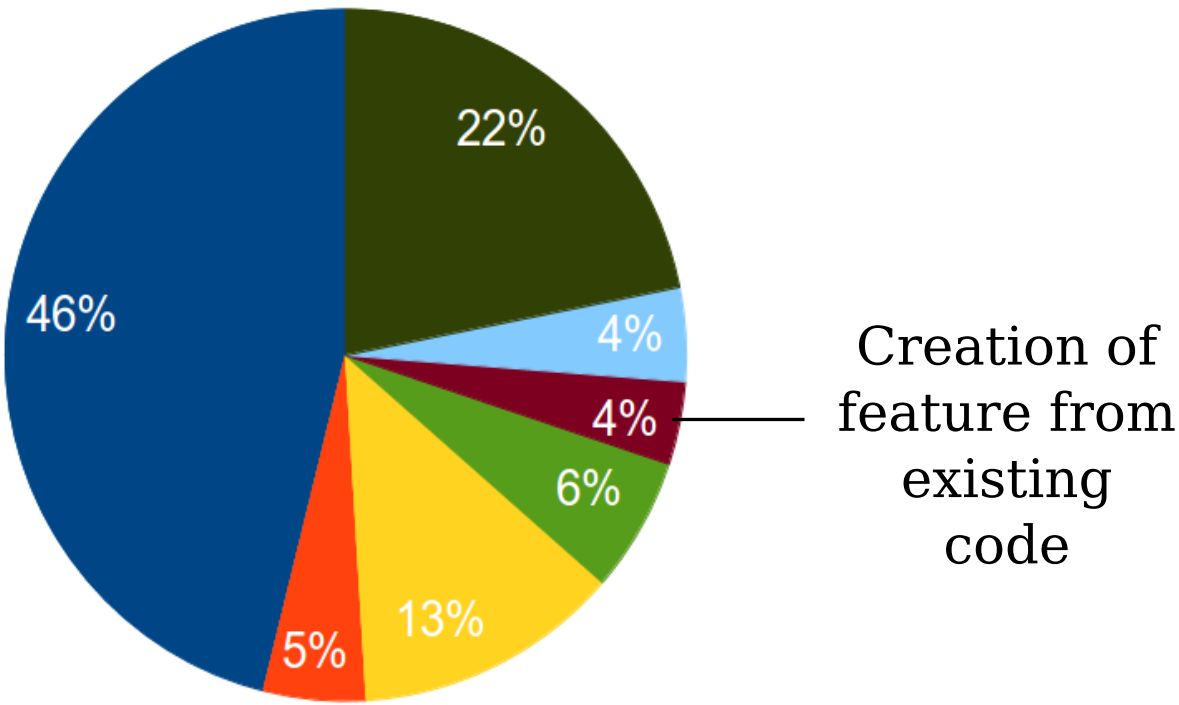
Pattern catalog (additions sample)



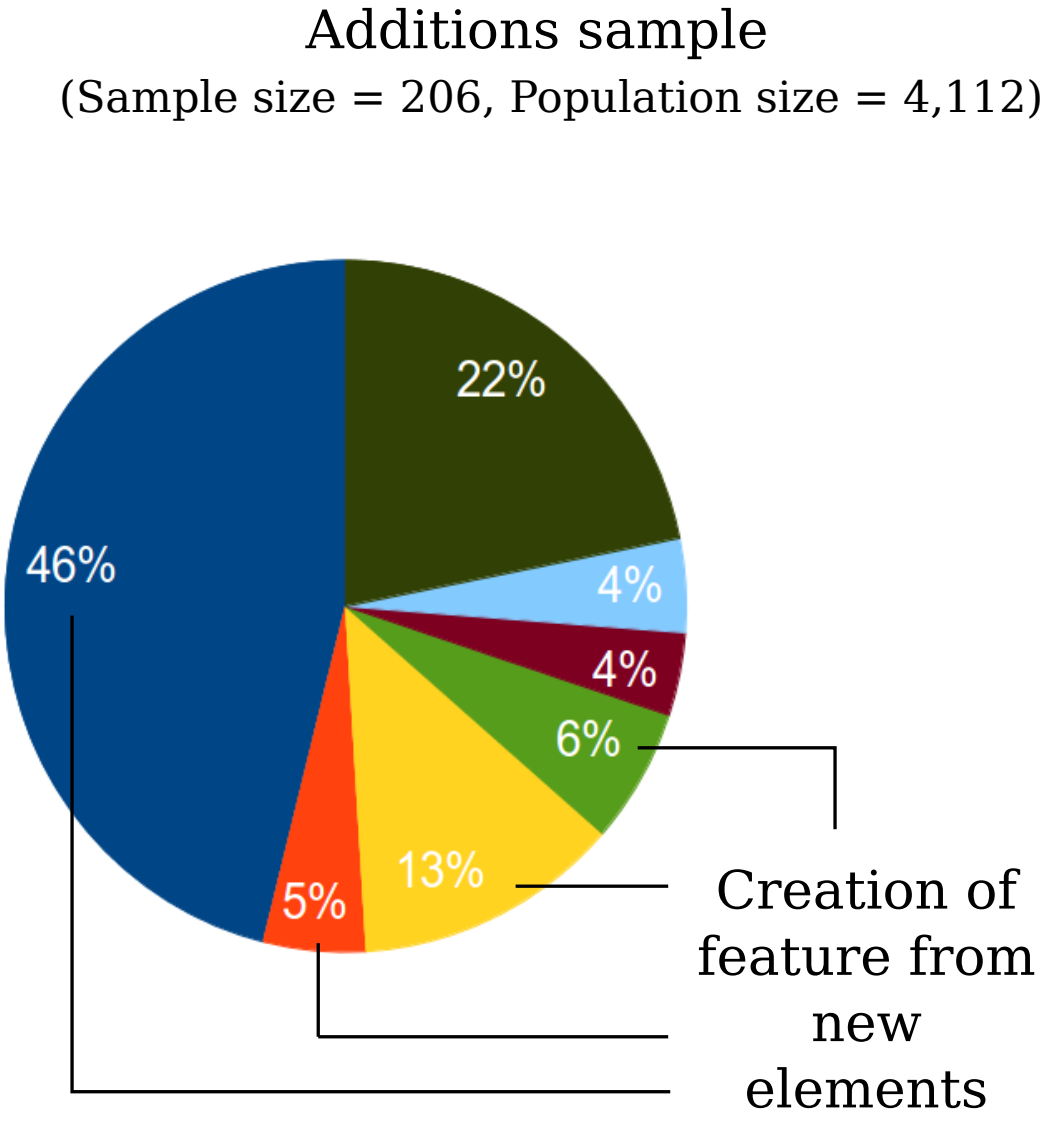
Pattern catalog (additions sample)

Additions sample

(Sample size = 206, Population size = 4,112)



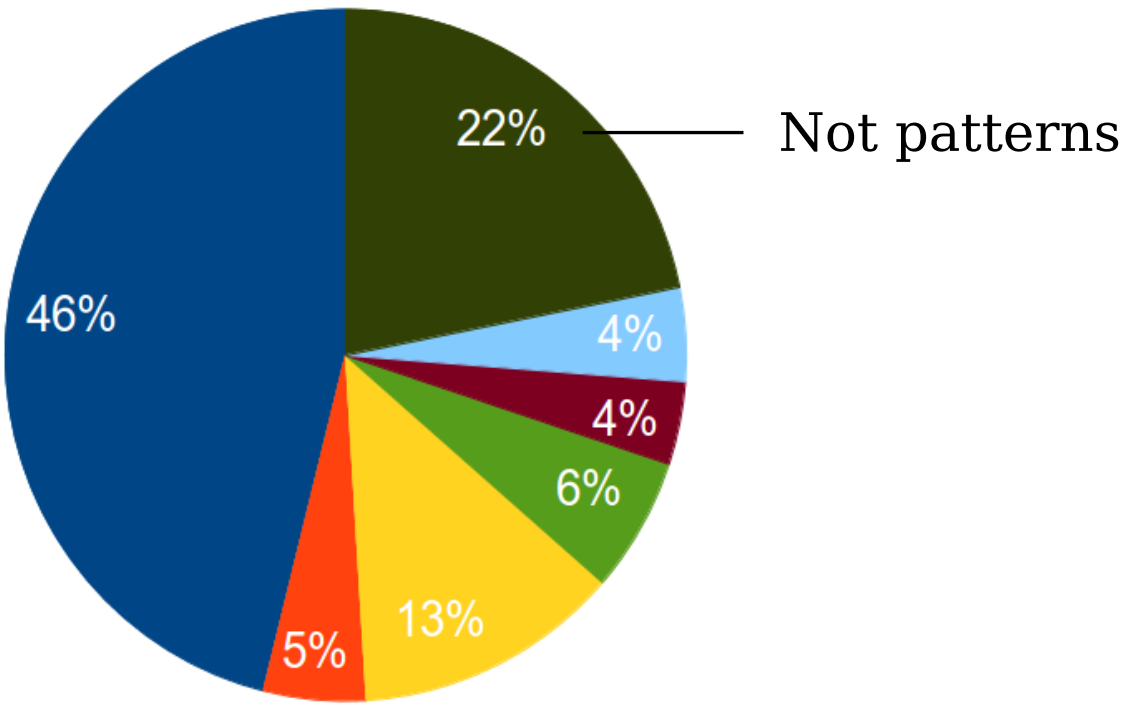
Pattern catalog (additions sample)



Pattern catalog (additions sample)

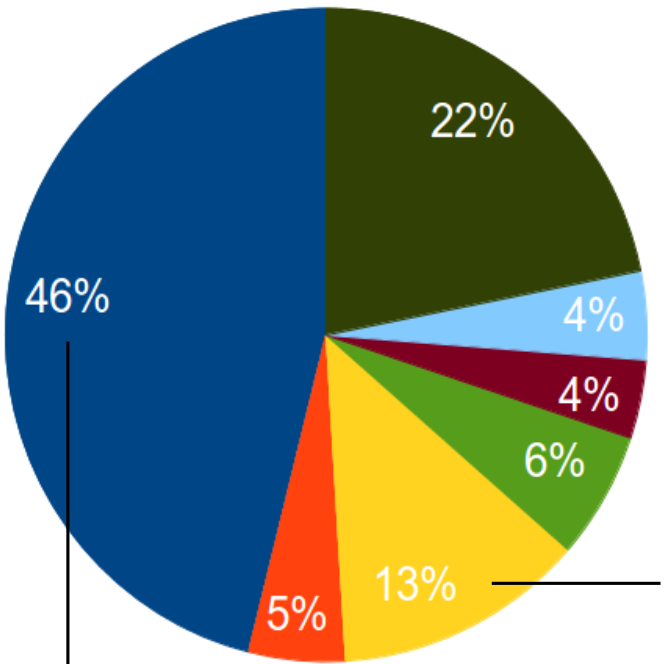
Additions sample

(Sample size = 206, Population size = 4,112)



Pattern catalog (additions sample)

Two most frequent patterns:



Add **V**isible **O**ptional **N**on-**M**odular **F**eature (AVONMF)

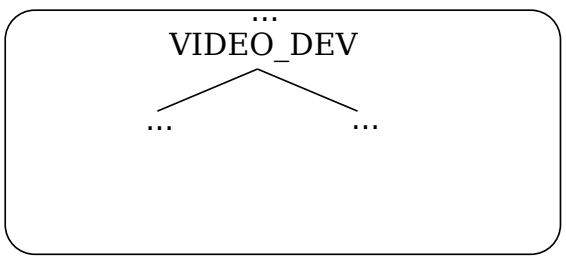
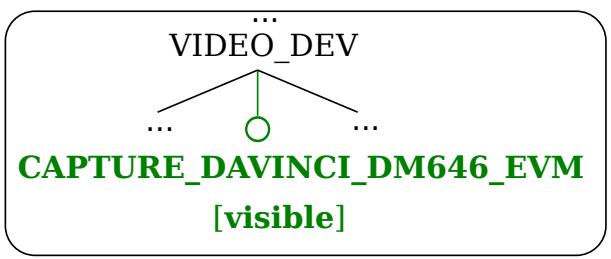
Add **V**isible **O**ptional **M**odular **F**eature (AVOMF)

Add Visible Optional Modular Feature

(Example)

Add Visible Optional Modular Feature

Example: CAPTURE_DAVINCI_DM646_EVM

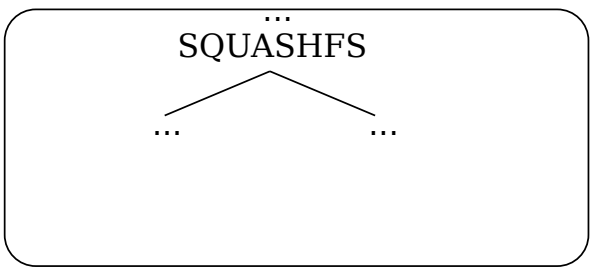
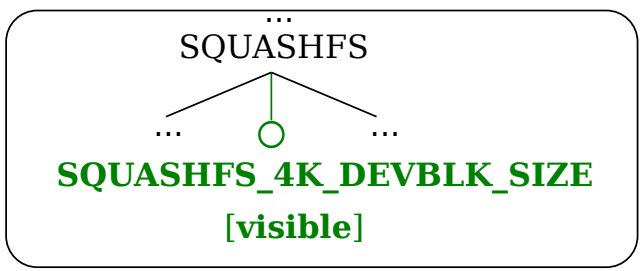
(Before state)	**(After state)**
	
CTC: set of cross tree constraints	CTC' = CTC + **(CAPTURE_DAVINCI_DM646_EVM requires MACH_EVM)**
M: Mapping	M' = M + **new compilation rule:** **if CAPTURE_DAVINCI_DM646_EVM is set** **compile vpfi_capture.c**
I: Implementation	I'= I + **vpfi_capture.c**

Add Visible Optional Non-Modular Feature

(Example)

Add Visible Optional Non-Modular Feature

Example: SQUASHFS_4K_DEVBLK_SIZE

(Before state)	**(After state)**
	
CTC: set of cross tree constraints	CTC' = CTC + constraints of SQUASHFS_4K_DEVBLK_SIZE
M: Mapping	M: Mapping
I: Implementation	I'= I + **<** **#ifdef SQUASHFS_4K_DEVBLK_SIZE** **#define SQUASHFS_DEVBLK_SIZE 4096** **#else** **#define SQUASHFS_DEVBLK_SIZE 1024** **#endif** **>**

Findings

(additions sample)

Findings (additions sample)

- AVOMF:
  - Most features in Linux are modular
  - Modular features in AVOMF cause little scattering outside their module
- AVONMF:
  - ifdefs of non-modular features are coarse-grained, appearing mostly in the global (e.g., a conditional data structure) or function level (e.g., conditional statements)

Take home lesson #1

(practical)

Disciplined use of annotation-based techniques such as #ifdefs do not hinder evolution (hypothesis)

Catalog of evolution patterns

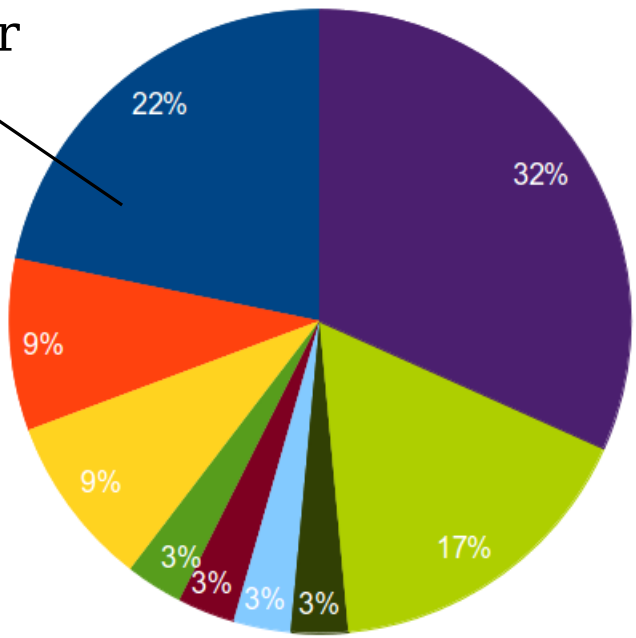
(removals sample)

Pattern catalog (removals sample)

Removals sample

(Sample size = 101, Population size = 1,002)

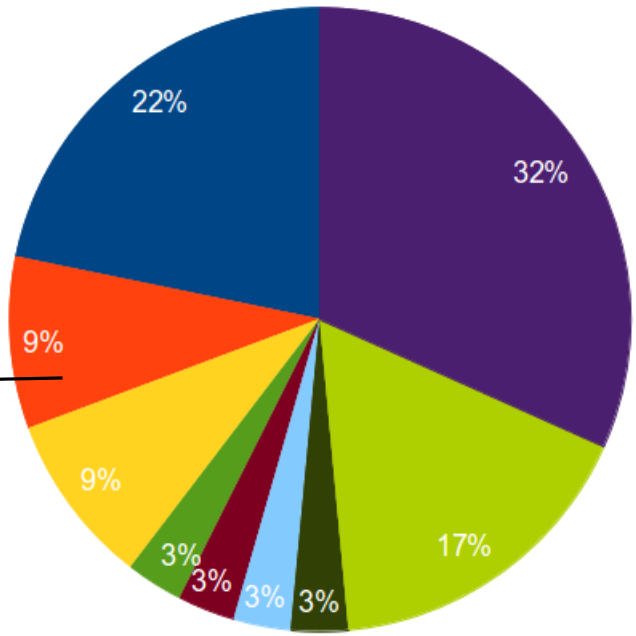
Remove **V**isible
Optional **M**odular
Feature
(RVOMF)



Pattern catalog (removals sample)

Removals sample

(Sample size = 101, Population size = 1,002)

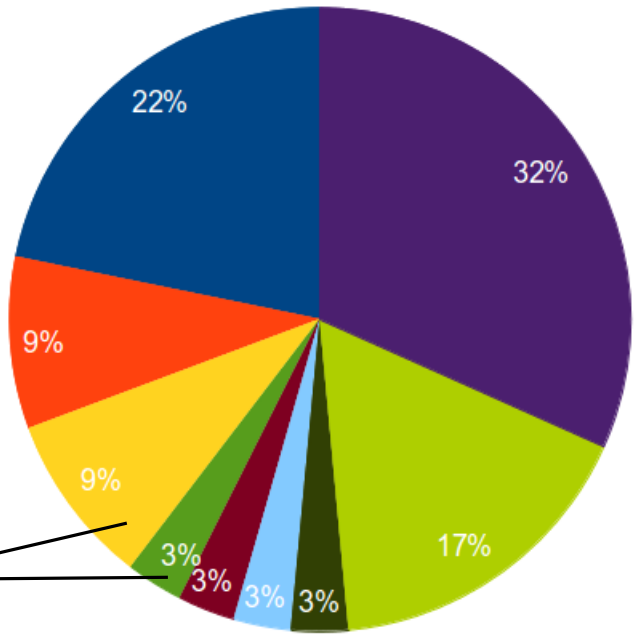


Remove **V**isible
Optional
Non-**M**odular
Feature
(RVONMF)

Pattern catalog (removals sample)

Removals sample

(Sample size = 101, Population size = 1,002)



Inverse removals of patterns in the additions sample

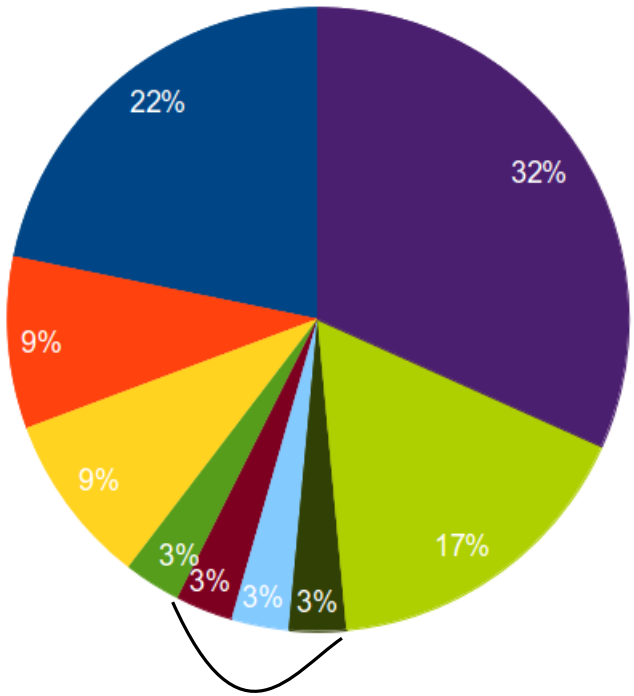
Pattern catalog (removals sample)

Removals sample

(Sample size = 101, Population size = 1,002)

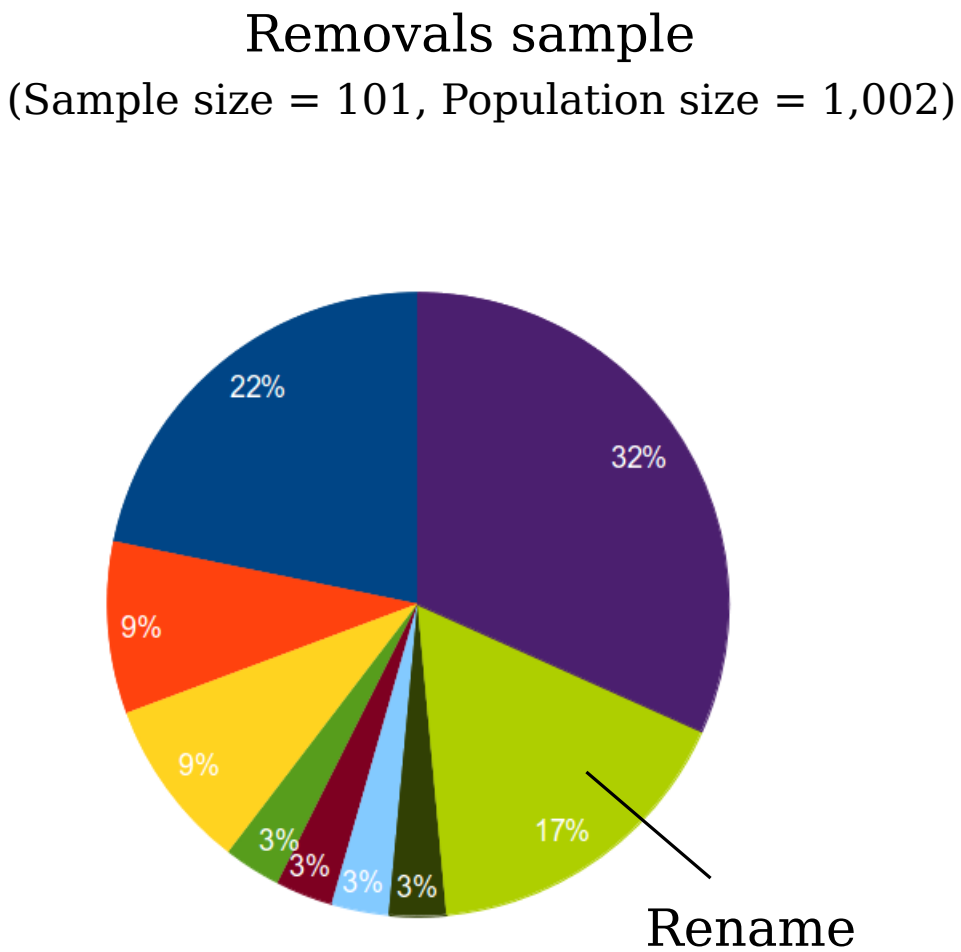
Merge of RT3090 into RT2860

Merge **V**isible **O**ptional **F**eature into **S**ibling (MVOFS)

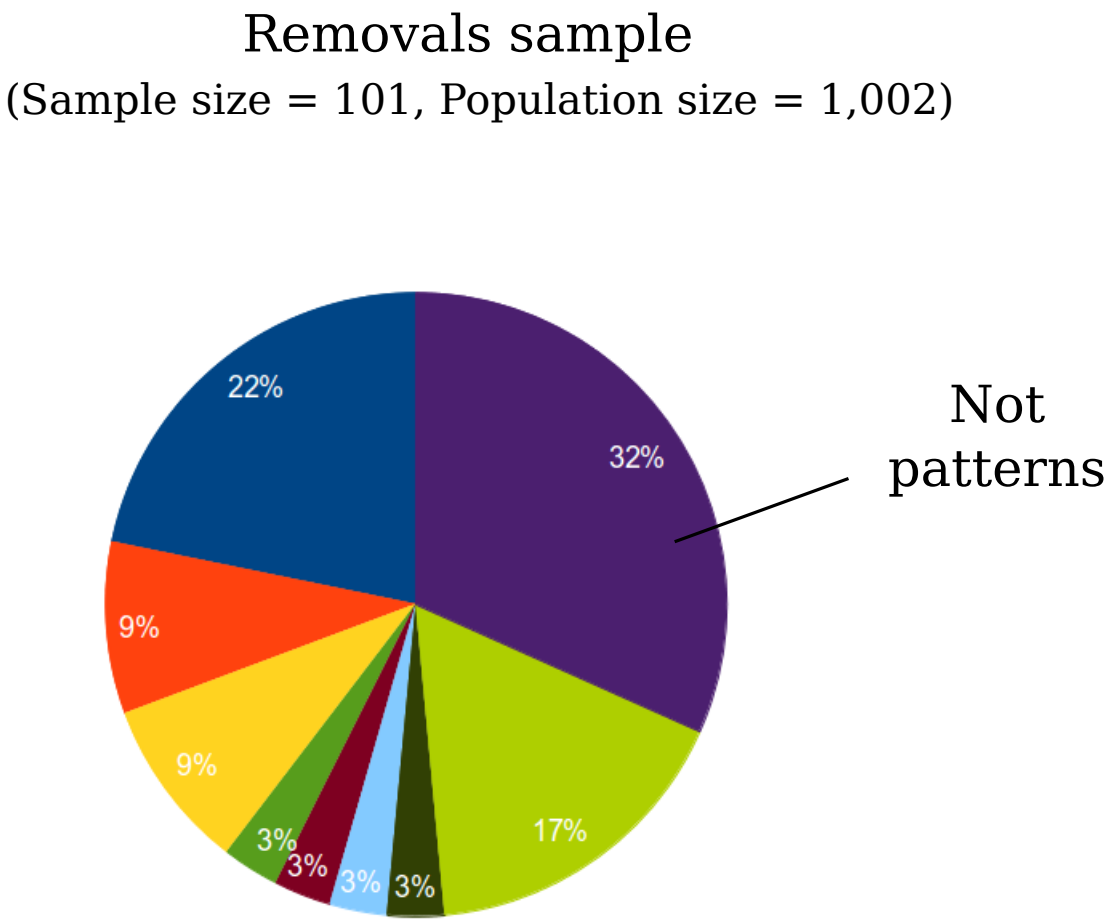


Merges: feature name is removed from the variability model, but feature continues to be supported elsewhere

Pattern catalog (removals sample)



Pattern catalog (removals sample)



Findings (removals sample)

- Merges:
  - Evolution requires a holistic view
  - Merges can only be identified by retrieving coevolving artifacts
- Complete feature removal (e.g., RVOMF = AVOMF^{-1})
  - Affect the set of supported products; thus not a refinement
  - Refinement changes are too restrictive in practice, as complete feature removals are often frequent (43% of all removals)

Take home lesson #2

(methodological)

Effective understanding of variability evolution requires looking at the coevolution of different artifacts

Take home lesson #3

(requirements for tool implementors)

The set of patterns provide concrete operations on how artifacts coevolve

Catalog provides evidence on which operations are important (e.g., we did not find split operations)

Conclusion

- We sumarized a catalog of 13 patterns that include coevolution of:
  - Variability model
  - Mapping
  - Implementation
- Subject of analysis: a large and complex software system – the Linux kernel
- Presented three take home lessons (full list of findings in the paper)

Thanks for listening!