UNIVERSITÄT OSNABRÜCK

Eingebettete Softwaresysteme

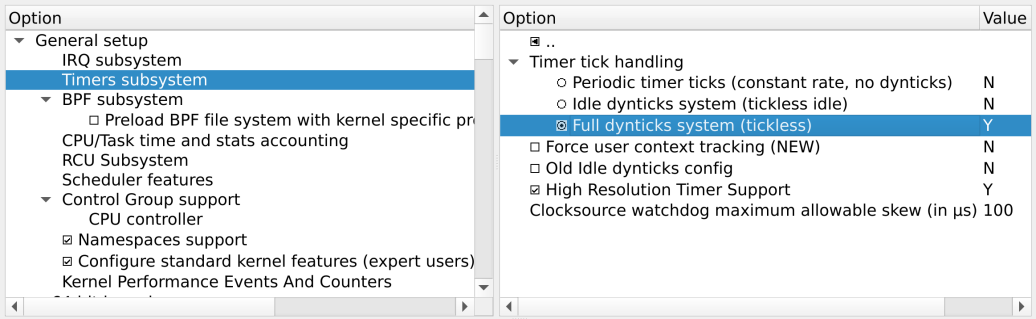
kconfig-webconf: Retrofitting Performance Models onto Kconfig-Based Software Product Lines

Birte Friesel, Kathrin Elmenhorst, Lennart Kaiser, Michael Müller, Olaf Spinczyk
birte.friesel@uos.de

Sep 15, 2022

Osnabrück University / Embedded Software Systems

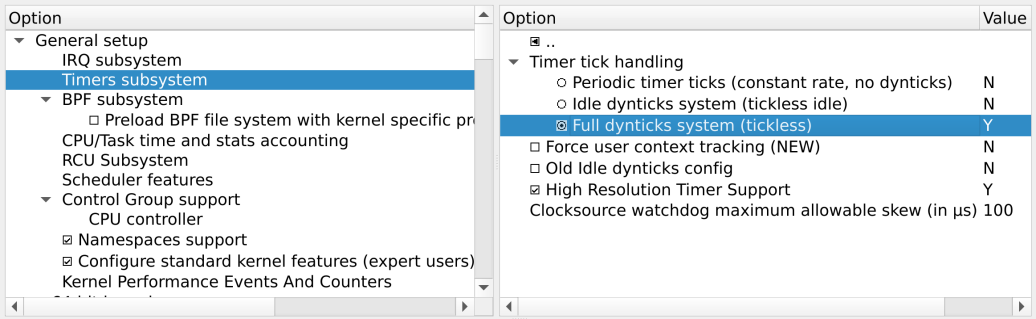
Kconfig-Based Software Product Lines



Linux: kconfig-qconf

- Kconfig: variability modeling language [Sin+07; EKS15]

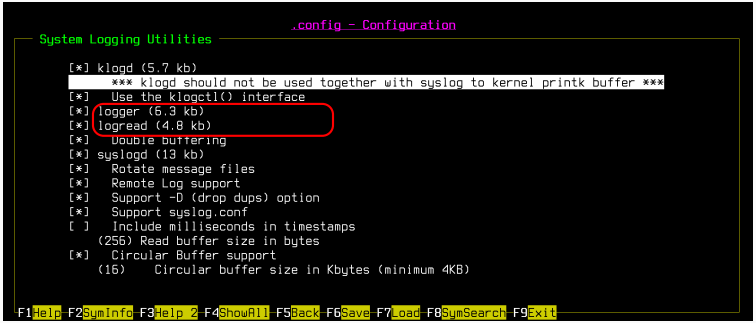
Kconfig-Based Software Product Lines



Linux: kconfig-qconf

- Kconfig: variability modeling language [Sin+07; EKS15]
- Does not support performance models

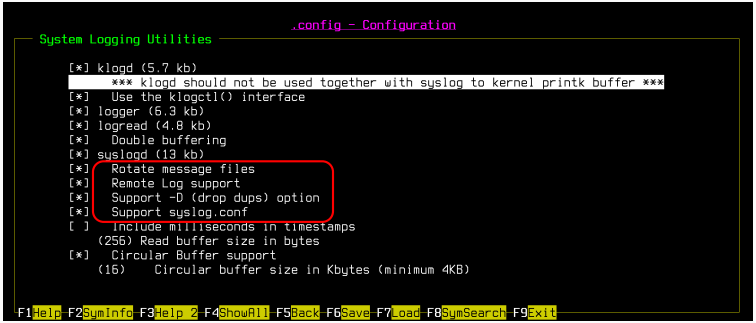
Kconfig-Based Software Product Lines



Busybox: kconfig-nconf

- Kconfig: variability modeling language [Sin+07; EKS15]
- Does not support performance models

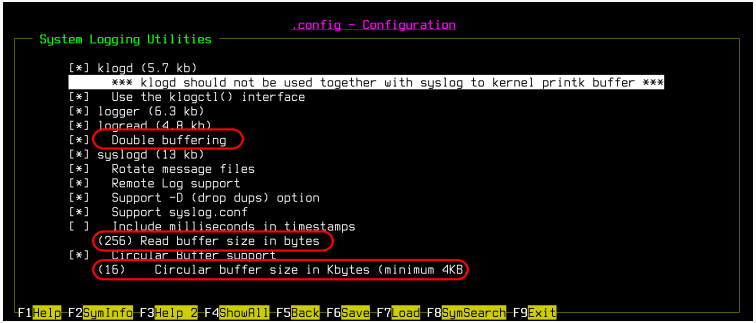
Kconfig-Based Software Product Lines



Busybox: kconfig-nconf

- Kconfig: variability modeling language [Sin+07; EKS15]
- Does not support performance models

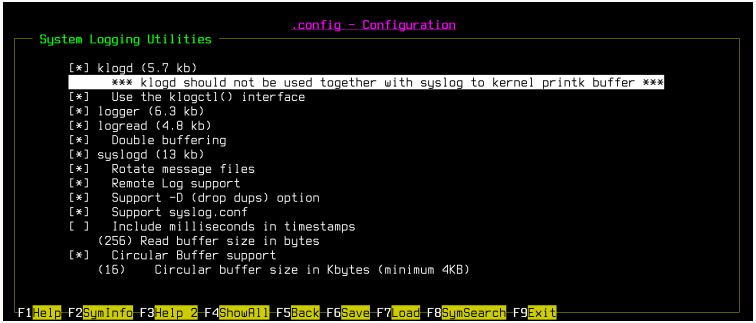
Kconfig-Based Software Product Lines



Busybox: kconfig-nconf

- Kconfig: variability modeling language [Sin+07; EKS15]
- Does not support performance models

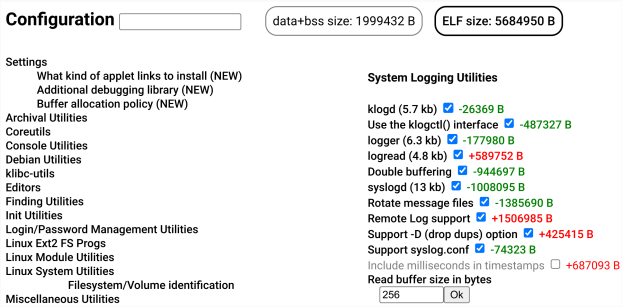
Kconfig-Based Software Product Lines



Busybox: kconfig-nconf

- Kconfig: variability modeling language [Sin+07; EKS15]
- Does not support performance models
- Integrated into build process → switching is costly

Kconfig + Performance Models

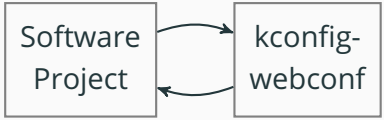


⇒ kconfig-webconf: Performance-aware drop-in replacement for Kconfig

Implementation

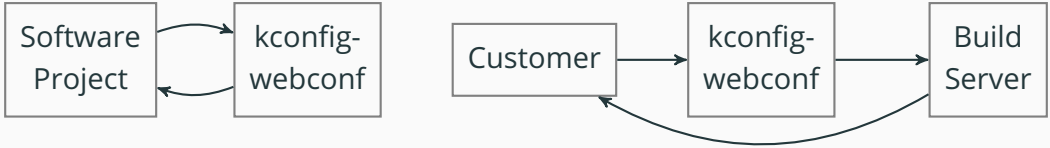
- kconfig-webconf: single-page React application
  Rationale: Browsers are ubiquitous cross-platform GUI

Implementation



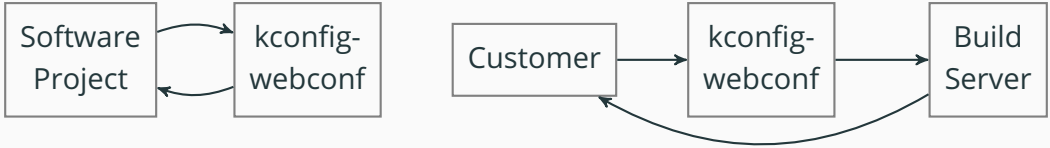
- kconfig-webconf: single-page React application
  Rationale: Browsers are ubiquitous cross-platform GUI
- local usage (no internet connection required)

Implementation



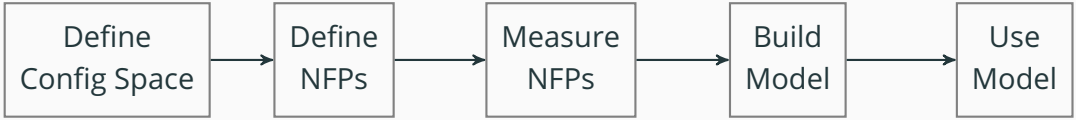
- kconfig-webconf: single-page React application
  Rationale: Browsers are ubiquitous cross-platform GUI
- local usage (no internet connection required)
- web usage (e.g. combined with build server)

Implementation



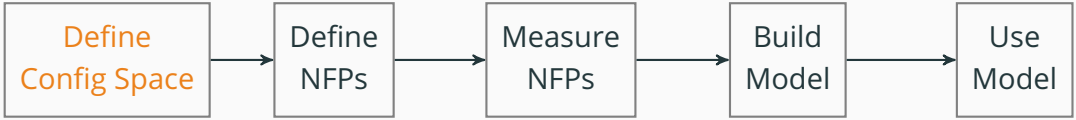
- kconfig-webconf: single-page React application
  Rationale: Browsers are ubiquitous cross-platform GUI
- local usage (no internet connection required)
- web usage (e.g. combined with build server)
- dfatool: Python3 companion tool for NFP model generation

Workflow



Example: **busybox** multi-call binary

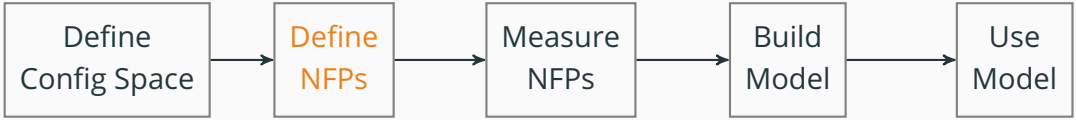
Workflow



Kconfig: machine-readable variability model → known configuration space

```
config MD5_SMALL
    int "MD5: Trade bytes for speed (0:fast, 3:slow)"
    default 1  # all "fast or small" options default to small
    range 0 3
    help
    Trade binary size versus speed for the md5sum algorithm.
    Approximate values running uClibc and hashing
    linux-2.4.4.tar.bz2 were:
    value               user times (sec)  text size (386)
    0 (fastest)         1.1                6144
    1                   1.4                5392
    2                   3.0                5088
    3 (smallest)        5.1                4912
```

Workflow



Define and measure performance attributes (non-functional properties)

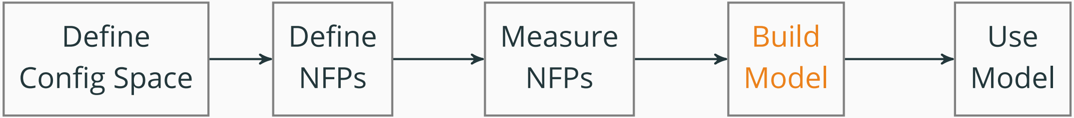
```
df@ios /t/t/busybox-1.35.0 > make nfpkeys
{"Image": {"ROM": {"unit": "B", "description": "ELF size", "minimize": true},
 "RAM": {"unit": "B", "description": "data+bss size", "minimize": true}}}
df@ios /t/t/busybox-1.35.0 > make nfpvalues
{"Image": {"ROM": 566328, "RAM": 2344}}
```

Workflow

Automatic measurements in configuration space

```
df@ios /t/t/busybox_data > explore-kconfig.py --log-level debug --random 2000 --with-neighbourhood ../busybox-1.35.0
INFO:root:Running randconfig 1 of 2000
INFO:dfatool.kconfig:build error
INFO:root:Running randconfig 2 of 2000
INFO:root:Exploring neighbourhood of /tmp/tmpfs/busybox_data/RandomConfig-b43a6b027123600c80b85d064e20053b/.config
DEBUG:dfatool.kconfig:Set DESKTOP to y
warning: FEATURE_MOUNT_FAKE (defined at /tmp/tmpfs/busybox-1.35.0/Kconfig:4746) has direct dependencies MOUNT with valu
g symbols:
 - FEATURE_MTAB_SUPPORT (defined at /tmp/tmpfs/busybox-1.35.0/Kconfig:5119), with value y, direct dependencies MOUNT |
NT (value: y)
DEBUG:dfatool.kconfig:Set EXTRA_COMPAT to y
DEBUG:dfatool.kconfig:Set FEDORA_COMPAT to n
DEBUG:dfatool.kconfig:Set INCLUDE_SUSv2 to n
```

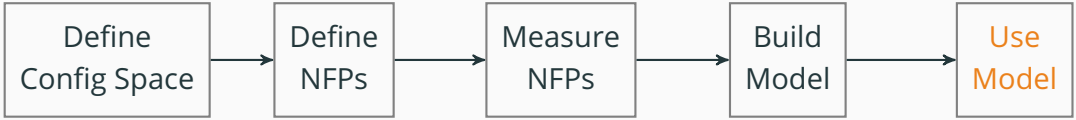
Workflow



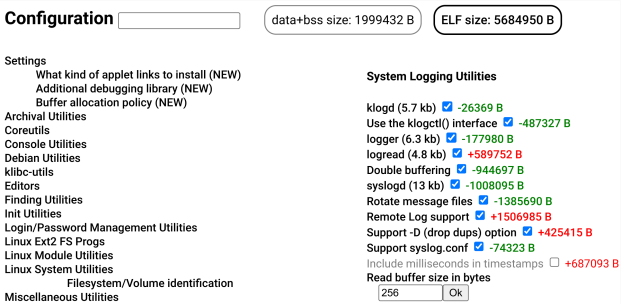
Automatic performance model generation

```
df@ios /t/t/busybox-data > analyze-kconfig.py --force-tree --export-webconf busybox.json ../busybox-1.35.0/Kconfig .
Model error after cross validation (kfold, 10):
key                 attribute                    static                   parameterized                LUT
Image               RAM               |||  157.08% /    2033715  |||    9.40% /     196390  |||  157.08% /    2033715
Image               ROM               |||  106.35% /    4837516  |||    9.27% /     538199  |||  106.35% /    4837516
```

Workflow



Performance-aware system configuration



Further Resources

- Live Demo
  - `ess.cs.uos.de/git-build/kconfig-webconf/splc22`
  - `ess.cs.uos.de/git-build/kconfig-webconf/splc22/embeddedml`
- Source Code
  - `ess.cs.uos.de/git/software/kconfig-webconf` (Frontend)
  - `ess.cs.uos.de/git/software/dfatool` (Model Generation)

Details ⇒ Demo session during coffee break

References i

[EKS15] Sascha El-Sharkawy, Adam Krafczyk, and Klaus Schmid. **"Analysing the Kconfig Semantics and Its Analysis Tools".** In: Proceedings of the 2015 ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences. GPCE 2015. Pittsburgh, PA, USA: Association for Computing Machinery, 2015, pp. 45–54. ISBN: 9781450336871. DOI: 10.1145/2814204.2814222. URL: https://doi.org/10.1145/2814204.2814222.

[Sin+07] Julio Sincero et al. **"Is The Linux Kernel a Software Product Line?"** In: Open Source Software and Product Lines (SPLC-OSSPL 2007). Ed. by Frank van der Linden and Björn Lundell. Kyoto, Japan, Sept. 2007.