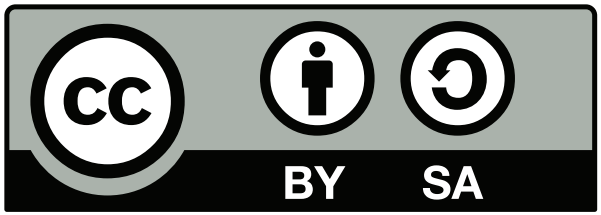
Configuration Management

Markus Raab

Institute of Information Systems Engineering, TU Wien

6.3.2019



Language

- Slides are in English
- Papers are in English
- Book is in English

Language of the Talk?

Task

Hands up if you prefer German.

Motivation

Misconfiguration

- *misconfigurations* [1, 8, 10, 11] are a major cause of system failures [4, 5, 9]
- much time needed to fix misconfigurations [3, 4, 7, 11]
- configuration is a user interface for both developers and system administrators

No-Futz

- Holland et al. [2] defined *futzing* to denote *"tinkering or fiddling experimentally with something."*
- With ***no-futz computing*** Holland et al. [2] mean *"that futzing should be allowed, but should never be required."*
- currently configuration is error-prone and under-specified, *futzing* is often required

Examples

Not every misconfiguration involves big companies, cloud, and huge amounts of money:

- No internet access because resolv.conf symlink broken.
- KDE crash because of ulimit setting.
- Out-of-service of computers during exam.

First Assignment

- Have you already experienced misconfiguration?
- Did you read about misconfiguration in the news?

Task

Discuss with your neighbor and tell us the best stories.

Content Overview

learning outcome:

- being familiar with the topics

Terminology

Definition

A ***configuration setting***, or ***setting*** in short, fulfills these properties:

1. It is provided by the execution environment.
2. It is *consumed* by an application.
3. It consists of a key, a configuration value, and potentially *metadata*. The ***configuration value***, or ***value*** in short, influences the application's behavior.
4. It can be *produced* by the maintainer, user, or system administrator of the software.

Requirement

A configuration library must be able to integrate (legacy) systems and must fully support (legacy) configuration files.

Requirement

Validation of configuration settings must happen systematically before the application is even started.

Requirements

Task

Discuss about requirements a configuration framework should fulfil.

Topic: *sources of configuration*

- configuration file formats
- command-line arguments
- environment variables

Topic: *design of configuration*

- complexity reduction
- configuration specification
- documentation of configuration

Topic: *architecture of configuration access*

- architectural decisions
- modularity
- avoidance of dependences

Topic: *introspection vs. code generation*

- introspection
- code generation

Topic: *misconfiguration detection and testing*

- early misconfiguration detection
- testability (how much variability is tested)

Topic: *context-aware configuration*

- cascading configuration
- context-awareness

Task

Break.

Topic: *validation vs. auto-detection*

- points in time for configuration access and validation
- validation techniques
- configuration-less systems (auto-detection)

Topic: *configuration as a user interface*

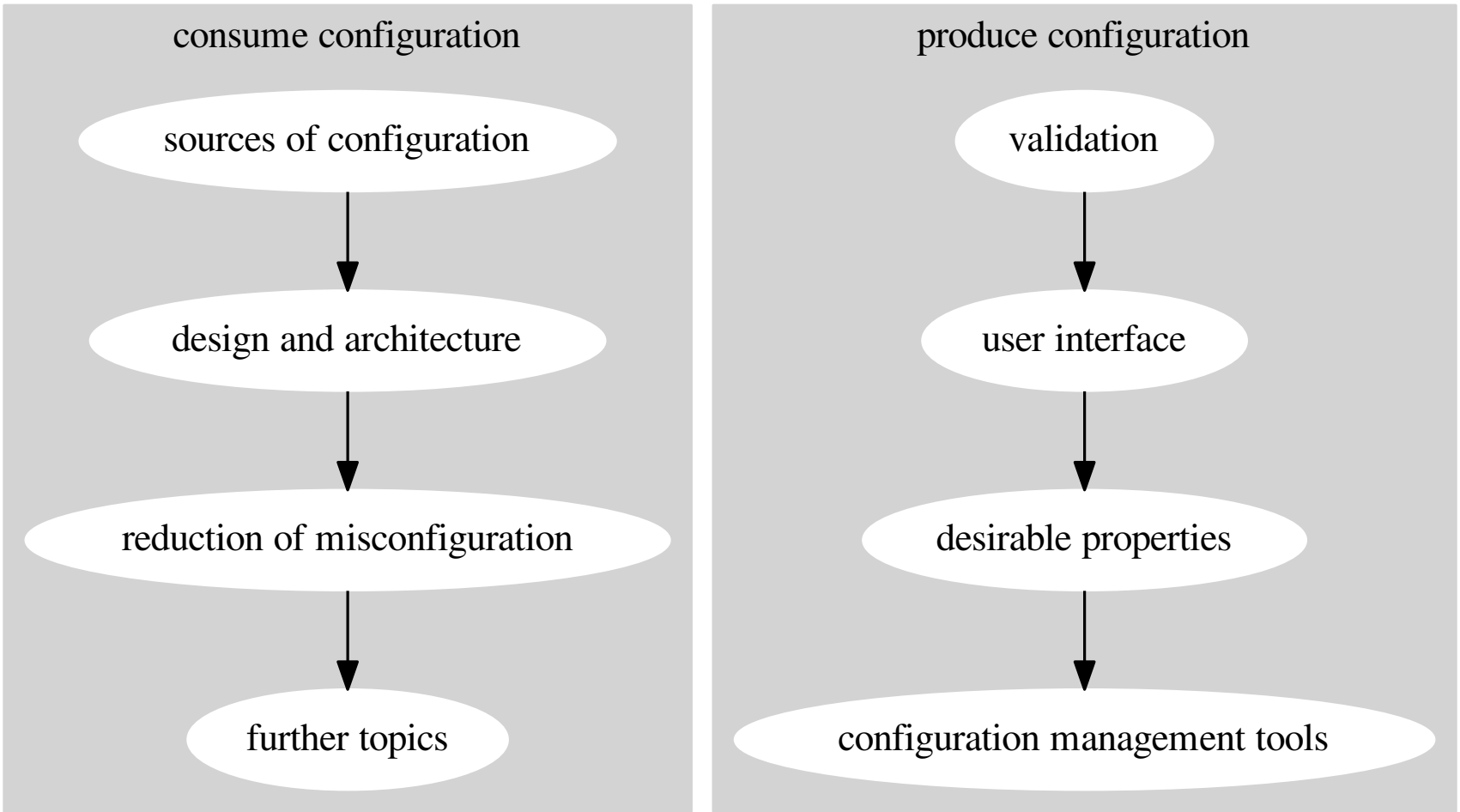
- How system administrators work.
- Which user interfaces exist.
- How to specify configuration.
- How to design error messages.

Topic: *desirable properties of configuration*

- self-description
- changeability
- idempotence
- round-tripping

Topic: *configuration management tools*

- history
- Infrastructure as Code
- configuration drift
- push vs. pull
- full vs. partial changes
- examples: Puppet, Chef, Ansible, CfEngine, Nix, ...


consume configuration
sources of configuration
design and architecture
reduction of misconfiguration
further topics
produce configuration
validation
user interface
desirable properties
configuration management tools

Elektra

- Elektra improves configuration management.
- Configuration management tools can use Elektra.
- Elektra implements what we discuss in this lecture.
- Elektra allows applications to fulfil the requirements we will discuss.
- Obviously you will also be able to apply the knowledge from this LVA without Elektra.
- Developed at TU Wien (`https://libelektra.org`).

In which topics are you interested?

Task (1)

Choose a partner for this task.

Task (2)

Go to stations and discuss topic with your partner.

Task (3)

Write down the most interesting topics. (Can be topics of stations or new topics.)

Organisation

Communication:

- TISS (forum and news)
  `https://tiss.tuwien.ac.at/course/courseDetails.xhtml?courseNr=194030&semester=2019S`
- GitHub (private and public repository)
  `https://git.libelektra.org`
- EMail `markus.raab@complang.tuwien.ac.at`
- before/after/during lectures

Task

Create a pull request to the Elektra Repository. The pull request description should contain "CM". With the account you used, you will get access to a private repository.

Deadline: 13.03.2019 23:59

Previous Knowledge

- Obviously *no* prior knowledge about Configuration or Configuration Management necessary.
- If you already have experience, you can use it in your talk and assignments.
- Knowledge about large-scale software construction is beneficially.
- You should have some understanding of software engineering and software requirements.
- Programming skills is a must.

Programming Languages

Elektra supports following programming languages:

- C
- C++
- Java
- Haskell
- Ruby
- Python
- Lua

You will get a grade once you submitted a PR with “CM” in the description.

To get a positive grade:

- All parts must be done.
- All parts must be positive.

Grade is calculated from:

- 30 %: homework
- 30 %: team exercise
- 10 %: talk
- 30 %: test
- +: extrapoints

Talk

about anything related to configuration management.

- The duration must be 20 minutes.
- It must be about your experience.
  - E.g., about the homework you did.
  - I.e., not only about study of literature.
  - If you extensively use some tool before, please share your experience.

Homework

The homework consists of three parts:

1. Play around with Elektra, create a PR with anything you find. For example:
   - Improve a tutorial or write a new one (if none exists).
   - Write new examples.
   - Write new tests.
   - Fix documentation.
2. Fix five bugs (from the pool of issues, or assigned by me based on your skills)
3. Implement a non-trivial feature, for example:
   - Make a configuration management tool use Elektra.
   - Write a validation plugin.
   - Write support for a new configuration file format (with an existing parser).

Task

Think about a talk and homework till next week. Write it down in the private repository (first come, first served).

Team Exercise

You can select your team (2 people) and your task related to *improve* configuration management. Every team member has a different task:

1. Elektrify some application.
2. Configure the application using Configuration Management.

And then swap roles and start again with different task.

If possible, you should use the parts developed in the homework.

Task

Talk with someone who is not necessarily your neighbor about a potential collaboration in the team exercise.

Lecture is every week Wednesday 09:00 - 11:00.

06.03.2019: topic, teams
13.03.2019: registration, initial PR
20.03.2019: Guest Lecture
27.03.2019: (HS?)
03.04.2019:
08.05.2019: (HS?)
10.04.2019: mid-term submission of exercises
15.05.2019:
22.05.2019:
29.05.2019:
05.06.2019: final submission of exercises
12.06.2019:
19.06.2019: last corrections of exercises
26.06.2019: exam

Guest Lecture?

Title: Current Trends in Learning-based Program Analysis for Configurations

Name: Jürgen Cito, MIT

Date/Time: 20.03.2019, 09:00 (c.t.)

Questions?

Task

Please register for the course and start playing with Elektra.

Task

Any questions?

[1] Mona Attariyan and Jason Flinn. Automating configuration troubleshooting with dynamic information flow analysis. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI'10, pages 1–11, Berkeley, CA, USA, 2010. USENIX Association.

[2] David A. Holland, William Josephson, Kostas Magoutis, Margo I. Seltzer, Christopher A. Stein, and Ada Lim. Research issues in no-futz computing. In *Hot Topics in Operating Systems, 2001. Proceedings of the Eighth Workshop on*, pages 106–110. IEEE, May 2001. doi: 10.1109/HOTOS.2001.990069.

[3] Ratul Mahajan, David Wetherall, and Tom Anderson. Understanding BGP misconfiguration. In *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '02, pages 3–16, New York, NY, USA, 2002. ACM. ISBN 1-58113-570-X.

[4] David Oppenheimer, Archana Ganapathi, and David A. Patterson. Why do Internet services fail, and what can be done about it? In *USENIX Symposium on Internet Technologies and Systems*, volume 67. Seattle, WA, 2003.

[5] Soila Pertet and Priya Narasimhan. Causes of failure in web applications (cmu-pdl-05-109). *Parallel Data Laboratory*, page 48, 2005.

[6] Markus Raab and Gergö Barany. Introducing context awareness in unmodified, context-unaware software. In *Proceedings of the 12th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1: ENASE,*, pages 218–225. INSTICC, ScitePress, 2017. ISBN 978-989-758-250-9. doi: 10.5220/0006326602180225.

[7] Ariel Rabkin and Randy Katz. Static extraction of program configuration options. In *Software Engineering (ICSE), 2011 33rd International Conference on*, pages 131–140. IEEE, 2011.

[8] Ya-Yunn Su, Mona Attariyan, and Jason Flinn. Autobash: Improving configuration management with operating system causality analysis. pages 237–250, 2007. doi: 10.1145/1294261.1294284. URL `http://dx.doi.org/10.1145/1294261.1294284`.

[9] Avishai Wool. A quantitative study of firewall configuration errors. *Computer*, 37(6):62–67, 2004.

[10] Tianyin Xu and Yuanyuan Zhou. Systems approaches to tackling configuration errors: A survey. *ACM Comput. Surv.*, 47(4):70:1–70:41, July 2015. ISSN 0360-0300. doi: 10.1145/2791577. URL `http://dx.doi.org/10.1145/2791577`.

[11] Zuoning Yin, Xiao Ma, Jing Zheng, Yuanyuan Zhou, Lakshmi N. Bairavasundaram, and Shankar Pasupathy. An empirical study on configuration errors in commercial and open source systems. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 159–172, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0977-6. doi: 10.1145/2043556.2043572.