

Sipwise solutions

Hochverfügbare VoIP-Systeme im Carrier-Umfeld

Andreas Granig + Michael Prokop



Der häufigste Grund für Ausfälle ist ...

Der häufigste Grund für Ausfälle ist ...

Human Error

Bei ...

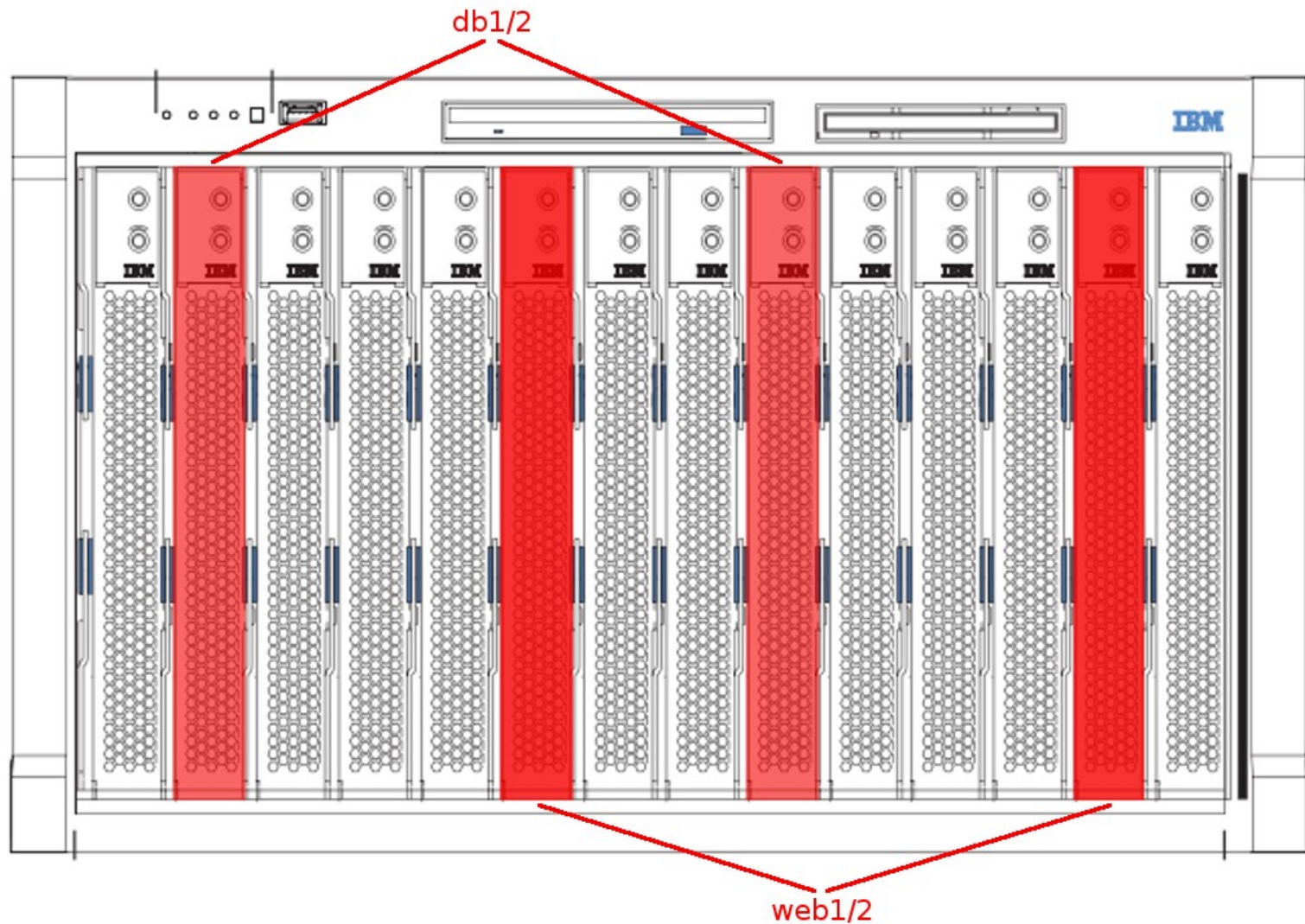
- Software Upgrades
- Konfigurations-Änderungen
- Hardware Maintenance

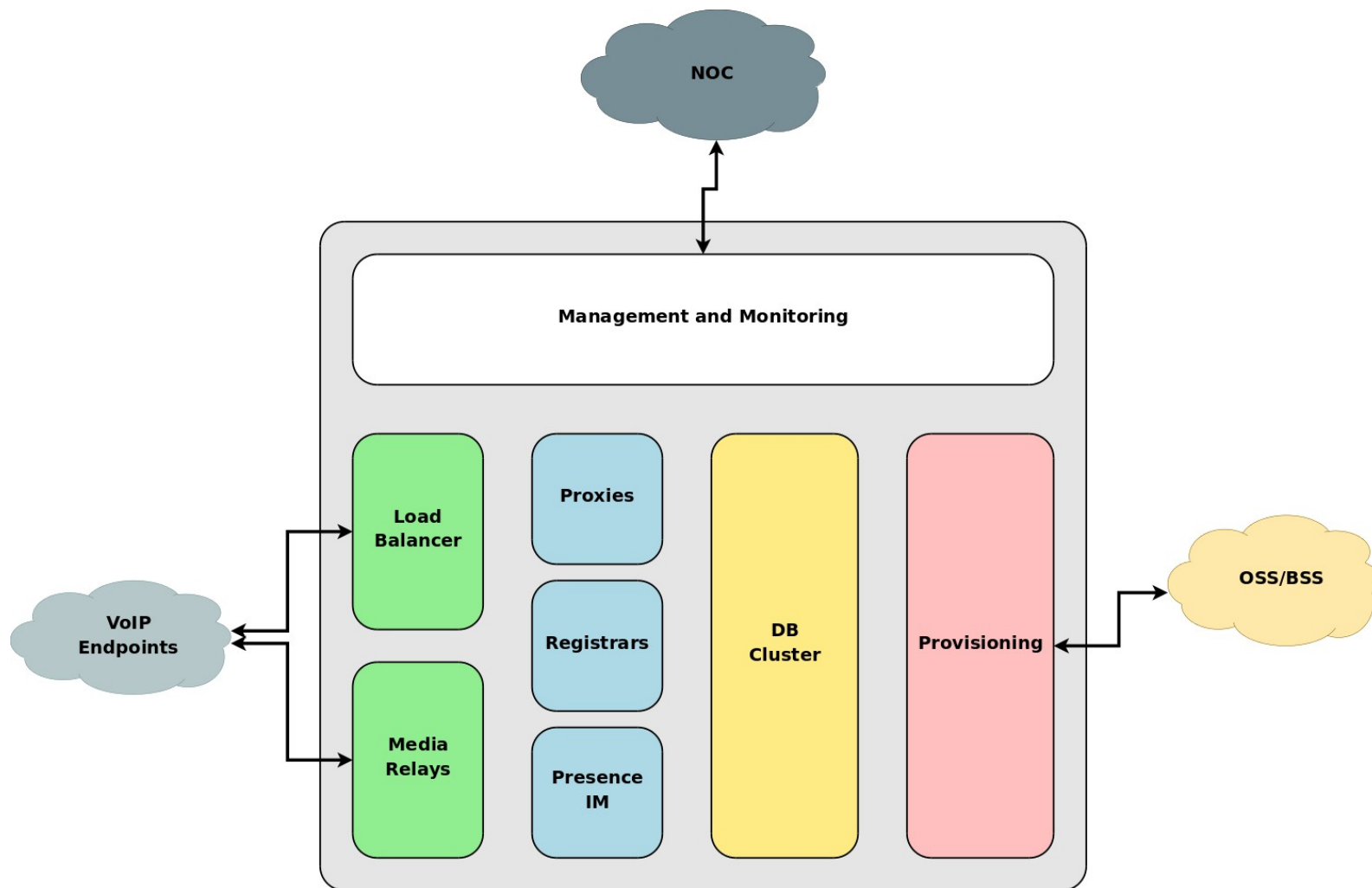
- Carrier Grade HW-Design
- Modulare SW-Komponenten
- Deployment und Upgrades

... aus Sicht eines Herstellers



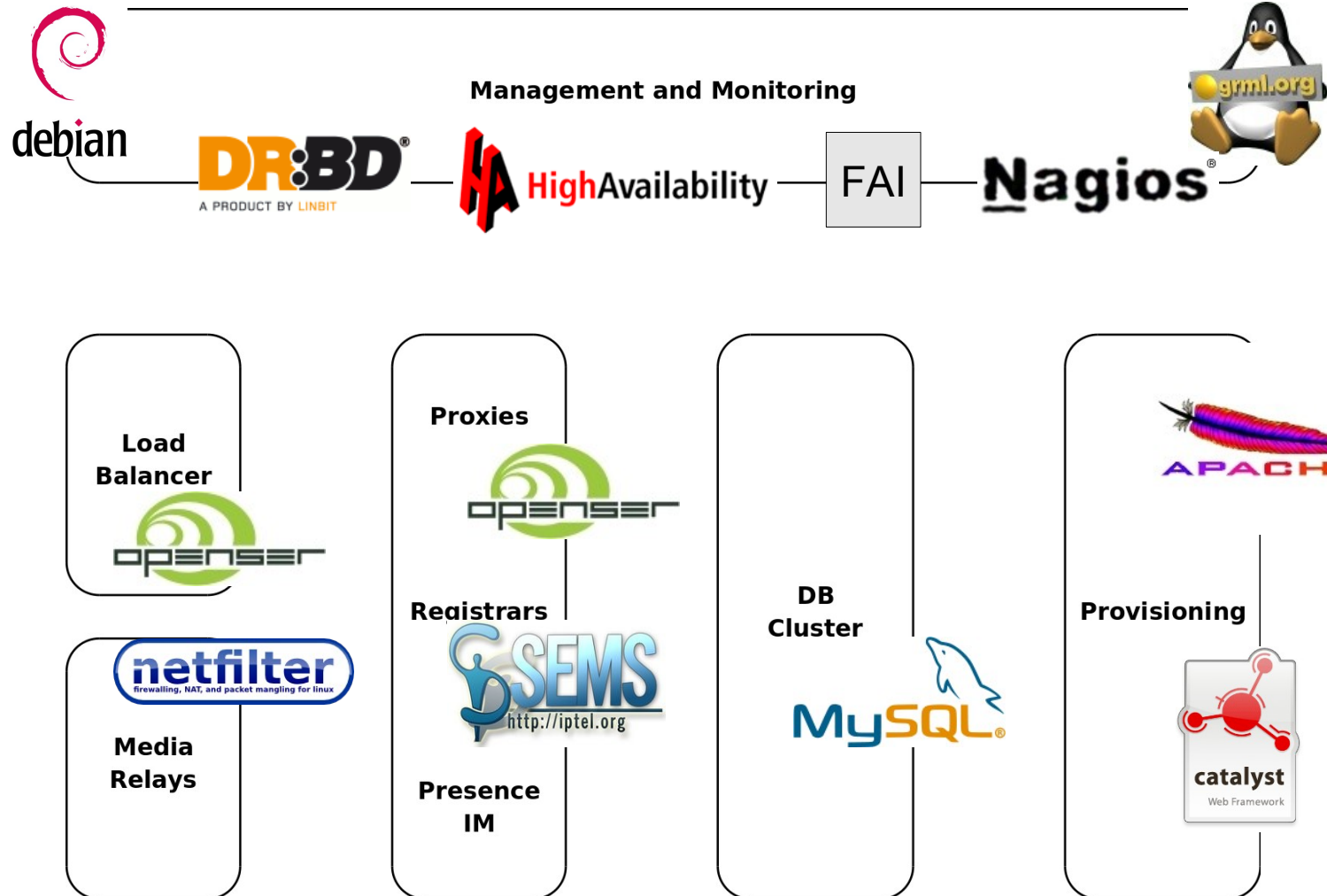
Copyright: <http://www.flickr.com/photos/12967790@N00/74452762> - cc-by-nc-sa-2.0





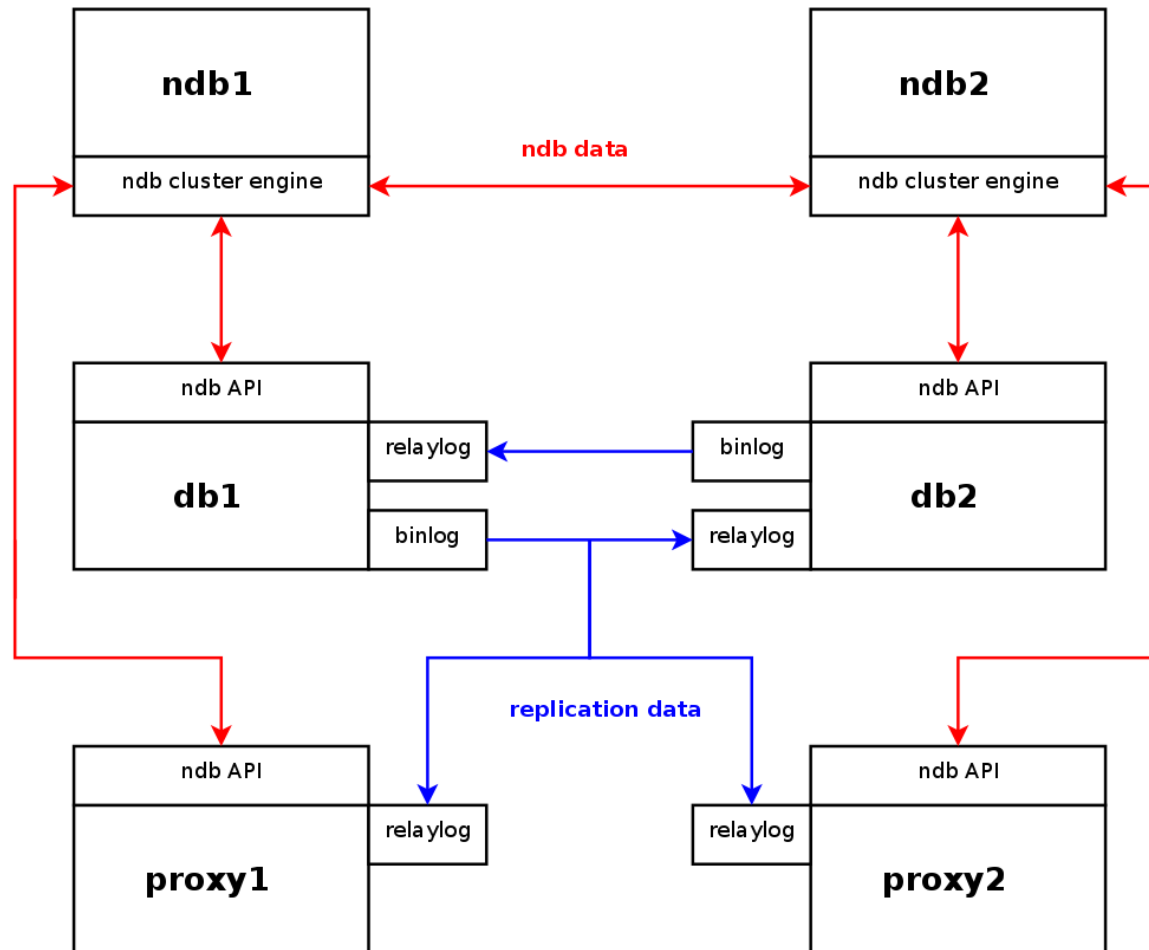
Modulare SW-Komponenten

Einsatz von FOSS



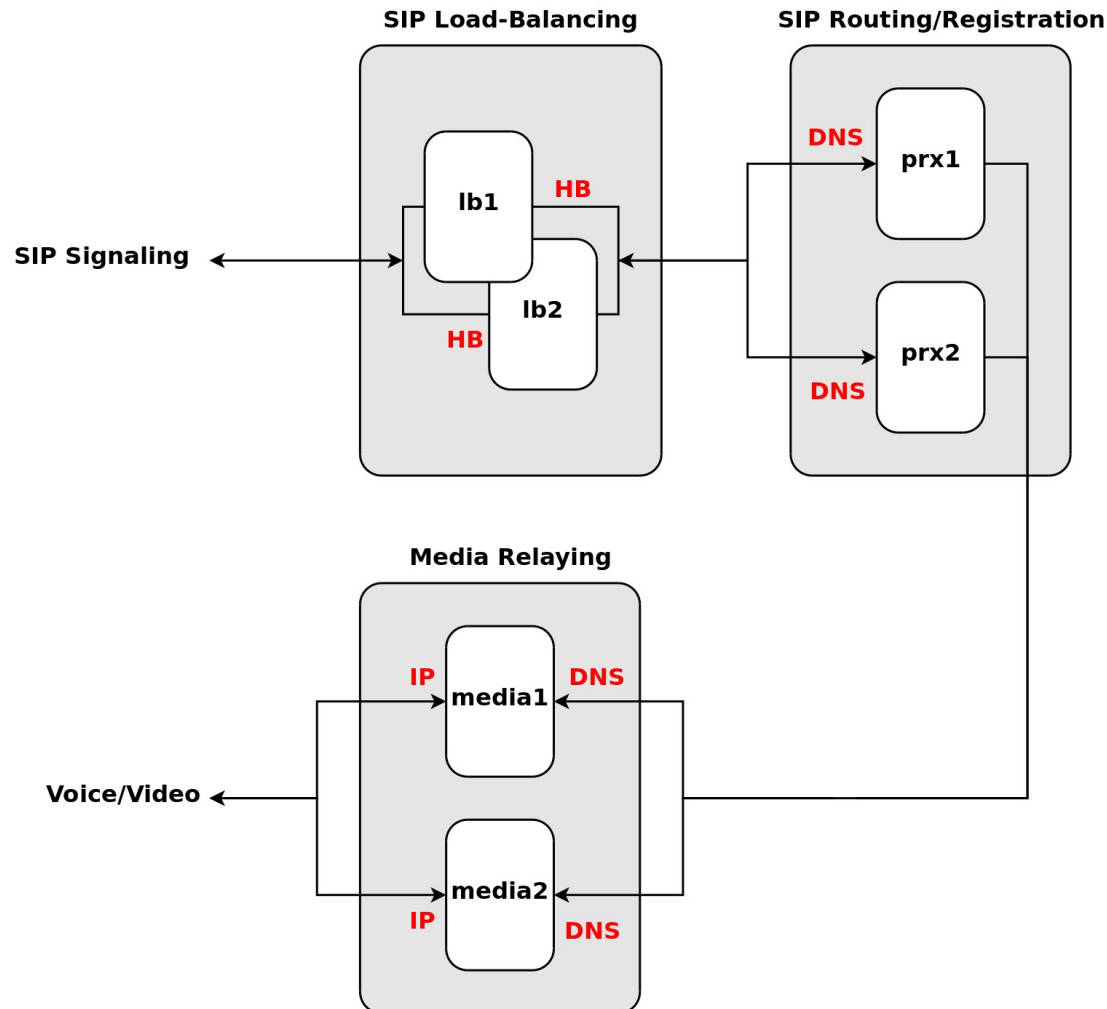
Modulare SW-Komponenten

Hochverfügbares DB-Design



Modulare SW-Komponenten

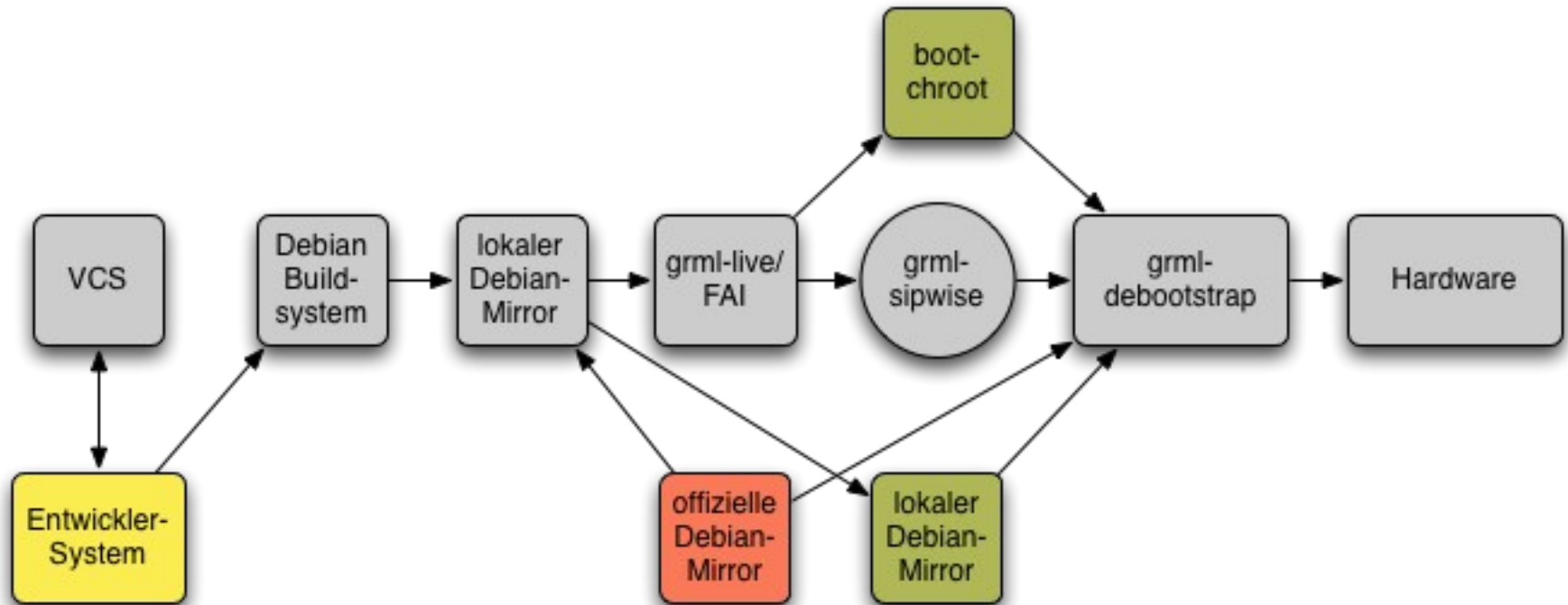
Hochverfügbares SIP-Design



Herausforderung aufgrund ...

- vieler Deployments ...
- mit unterschiedlichen Features ...
- in verschiedenen Versionen.

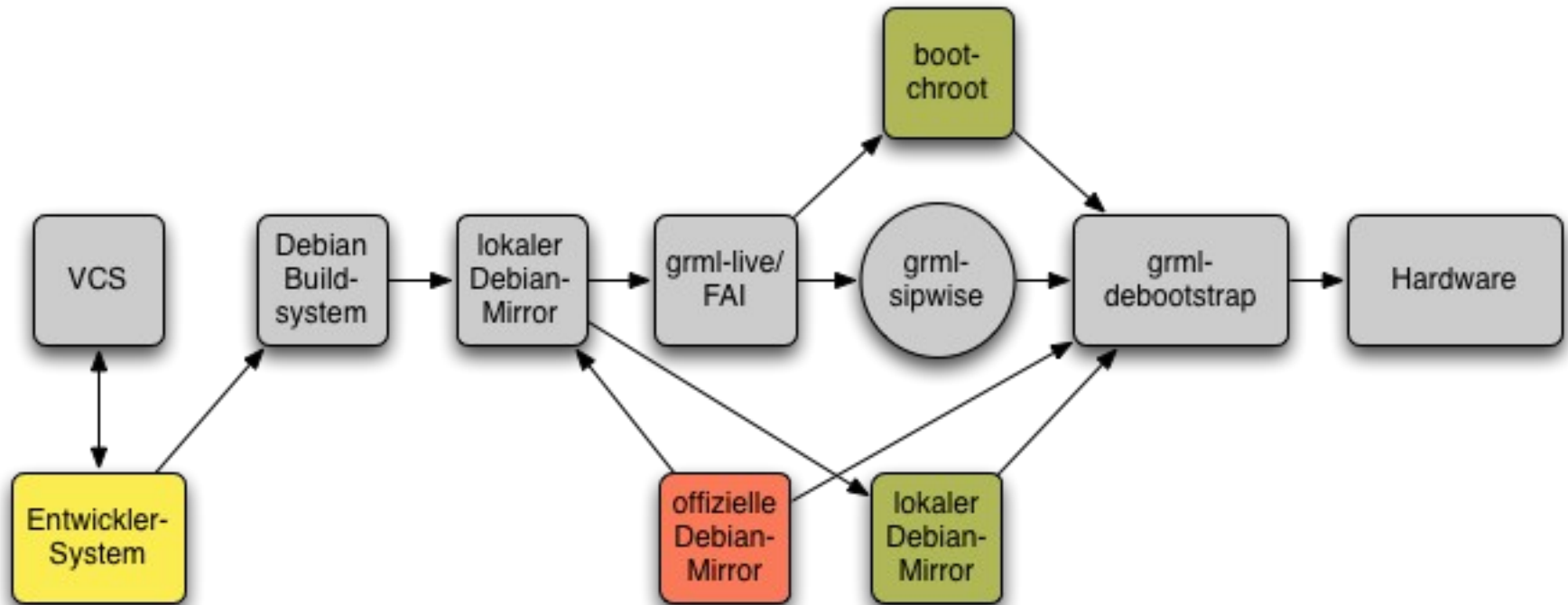
Wie behält man den Überblick?



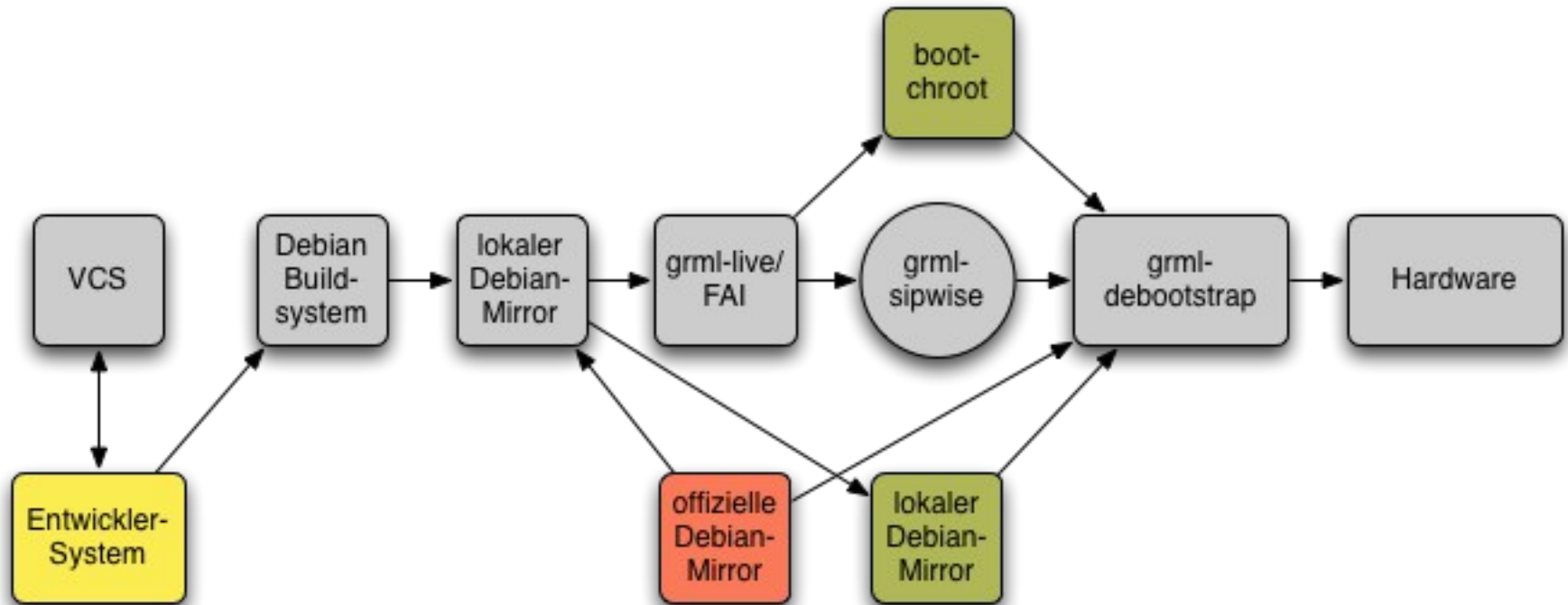
- aktuell Subversion
- Git? → in Evaluierungsphase
 - `git-svn`
- dpatch: VCS-unabhängiger Workflow
 - `dpatch-edit-patch 01_demo.dpatch`
 - `dpatch apply 01_demo.dpatch`
 - `dpatch apply-all`
 - `dpatch-edit-patch patch 02_new_feature 01_demo`
 - `dpatch deapply-all`
 - ...

Grober Ablauf:

- `$VCS-buildpackage` bzw.
`dpkg-buildpackage -sa -S -I.$VCS`
- `dput $builddsys foo_0.42-1_*.changes`
- `cowbuilder [...] --build \`
`~/dput/foo_0.42-1.dsc`
- `debsign foo_0.42-1_*.changes`
- `dput $mirror foo_0.42-1_*.changes`



- offizielle Debian-Pakete von öffentlichen Debian-Servern
- lokaler Mirror für FAI + grml-live:
 - Integration von angepassten Debian-Paketen
 - Integration von 3rd-Party-Paketen
 - ermöglicht Reproduzierbarkeit (Helper-Skript dpkg-to-fai)!
- apt-cacher[-ng] transparent zwischengeschaltet (Synchronisation von Cache von apt-cacher und lokalem Debian-Mirror)



FAI (Fully Automatic Installation):

- `fai dirinstall`
- erledigt Paketauswahl (auch reproduzierbare!)
- angepasste Hooks/Skripte/...
- Klassenkonzept!

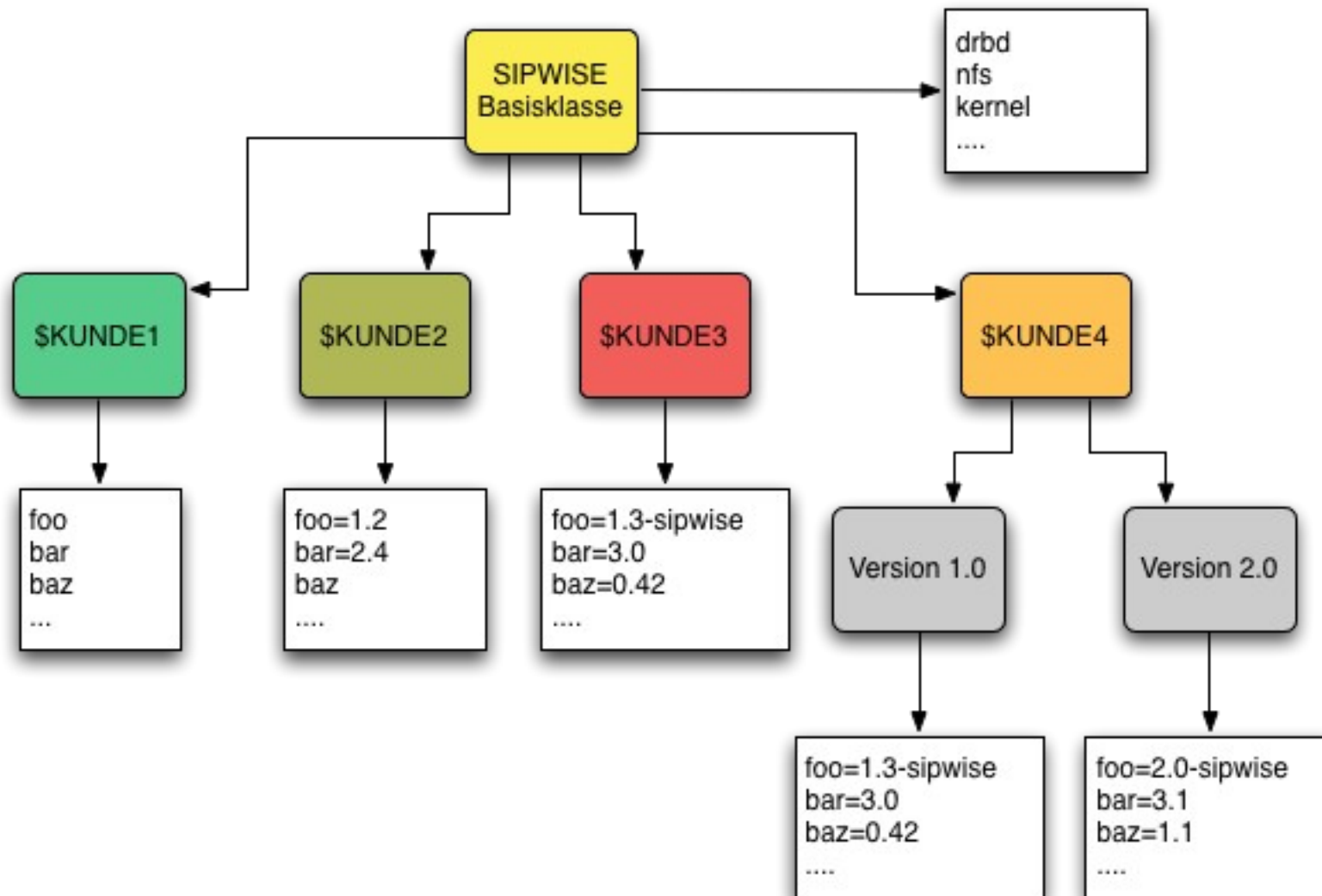
BTW: Vortrag um 16:15 Uhr im HS3 von
FAIs Hauptentwickler Thomas Lange!

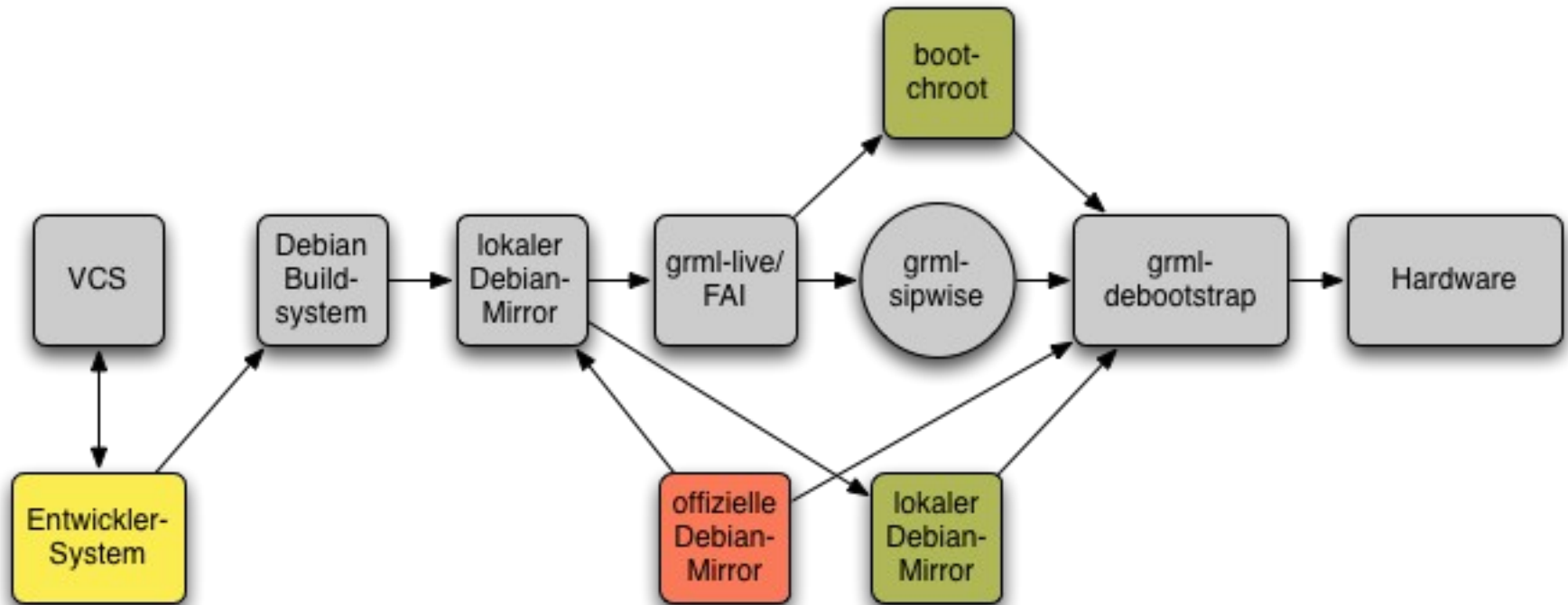
\$CONFIGSPACE

config/

basefiles/	=> Basis-Chroot
class/	=> Variable
debconf/	=> Paketkonfiguration
files/	=> fcopy & CO
hooks/	=> instsoft.[SIPWISE \$SPECIFIC]
package_config/	=> [SIPWISE \$SPECIFIC]
scripts/	=> SIPWISE + \$SPECIFIC->SIPWISE

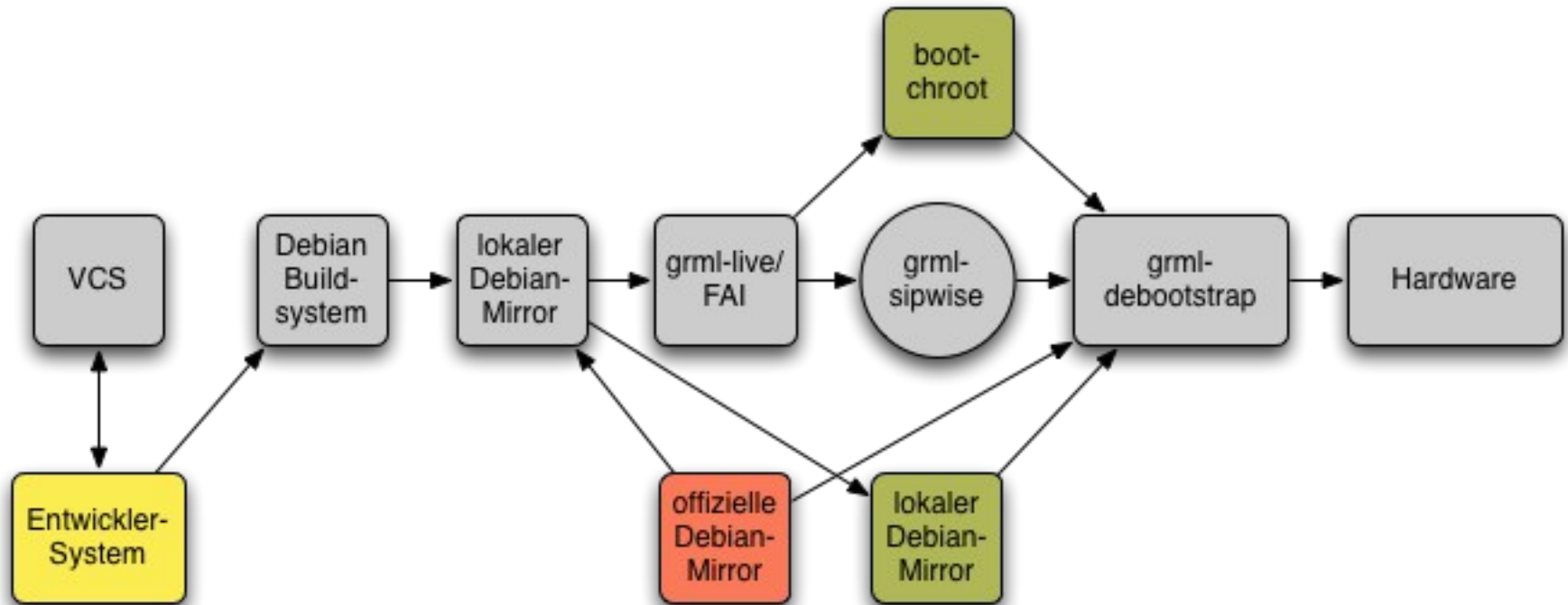
Warum der ganze Aufwand?!





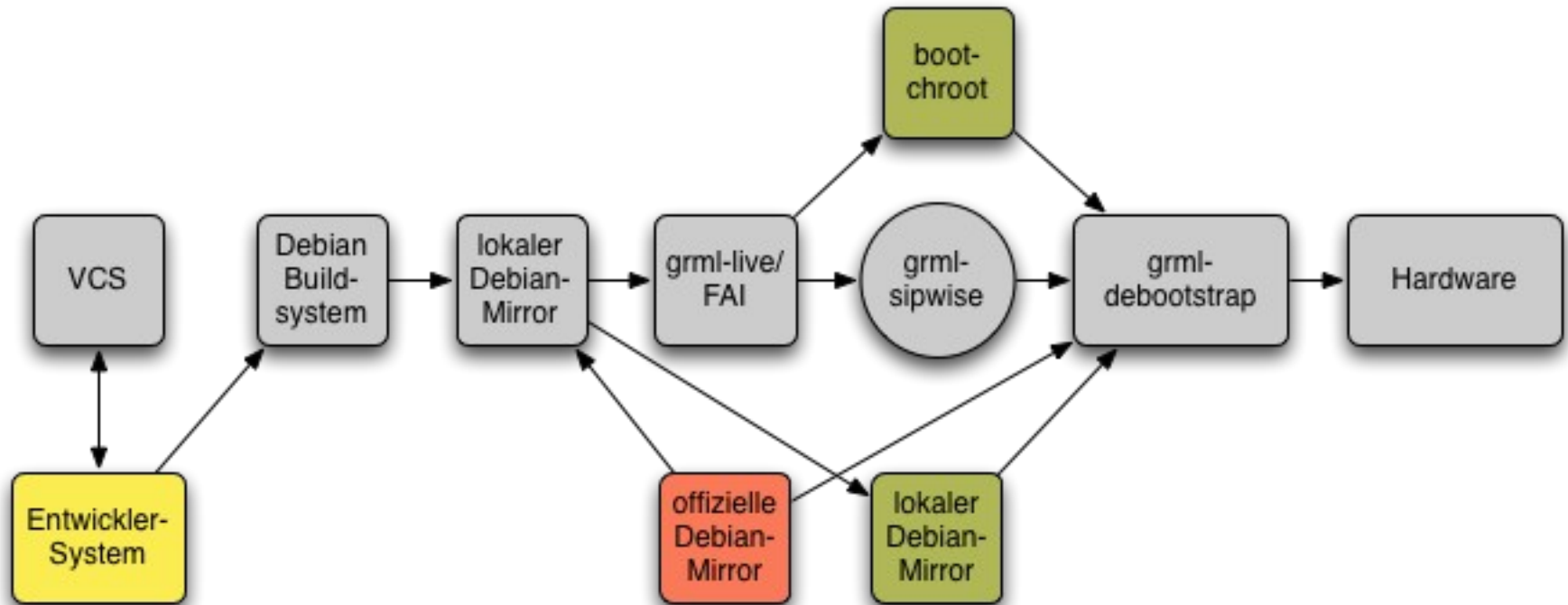
grml-sipwise => ISO

- angepasstes Linux Live-System
- basiert auf grml.org
- via grml-live gebaut



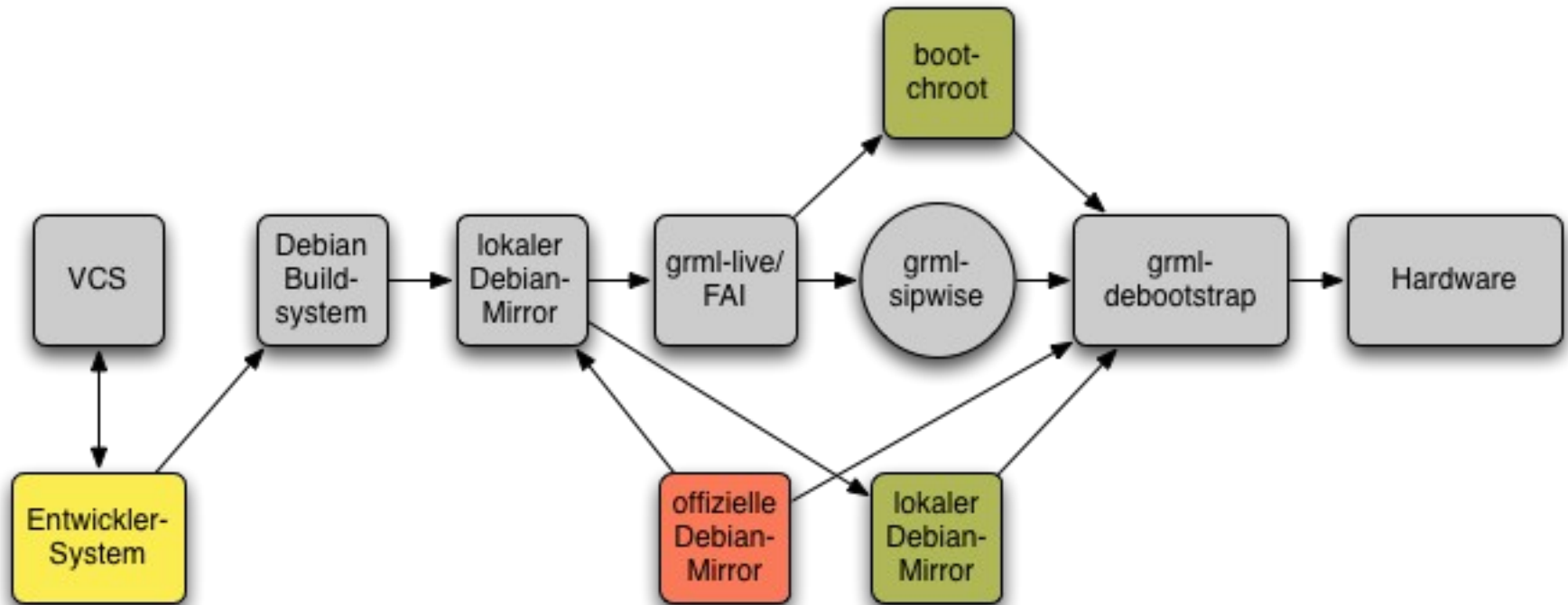
grml-debootstrap:

- Setup von SW-RAID (sofern kein HW-RAID)
- lokaler Mirror auf Install-Medium oder alternativ Netzinstallation
- DRBD-Deployment
- => Deployment von boot1/boot2 (inkl. boot-chroot)



boot-chroot wird via PXE gebootet:

- je nach Blade-Slot unterschiedliches System
- Initialisierung von lokalen Festplatten beim 1. Bootvorgang
- “Volatile”-Dateien via NFS



In Arbeit:

- automatisches Firmware-Upgrade
- 100% automatische Installation
 - auf USB-Stick mitgelieferte Konfigurationsdatei definiert Installation
 - automatische Dokumentation der Hardware
 - selbstkonfigurierendes Blade-Hotplugging

Fazit? Wichtig:

- Automatisiert
- Dokumentiert
- Reproduzierbar
- Skalierbar

Fragen?

Andreas Granig <agranig@sipwise.com>

Michael Prokop <prokop@grml-solutions.com>

<http://sipwise.com/>