

# Reproduzierbare Veröffentlichungen

Arne Babenhauserheide

20. Dezember 2013

## Die Versuchung

- „Haben Sie einmal ein Paper mit per Skript erstellten Grafiken veröffentlicht?“
- „Haben Sie Skripte und Daten veröffentlicht?“
- „Warum erfinden Sie die Daten nicht? Das wäre weniger Arbeit. . .“

*„Niemals! Das verbietet die wissenschaftliche Integrität!“*

## Die Versuchung

- „Haben Sie einmal ein Paper mit per Skript erstellten Grafiken veröffentlicht?“
- „Haben Sie Skripte und Daten veröffentlicht?“
- „Warum erfinden Sie die Daten nicht? Das wäre weniger Arbeit. . .“

*„Niemals! Das verbietet die wissenschaftliche Integrität!“*

## Doch es passiert - leider

- Einstieg: „passte ich den Untersuchungsentwurf an“.
- Dietrich Stapel: „Es war grau und es war üblich“.
- Sturz: „erfindet die Daten“.
- „Forscher gratulieren“.
- „Drei [seiner] Doktoranden sind Ungereimtheiten [. . .] aufgefallen“.

*[Quarks & Co., 2013-06-04]*

## Wissenschaft

- referenzierbar?
- vertrauenswürdig?
- prüfbar?
- wiederholbar?

*Vertrauen in andere durch saubere Veröffentlichungen.*

## Forschende

- „Welche Daten nutze ich?“
- „Liefere ich alle Daten?“
- „Wie habe ich das Paper damals erstellt?“

*Vertrauen in die eigene Veröffentlichung.*

# Anforderungen an ein System

- Diagramme und Tabellen aus aktuellen Daten
- Benötigte Daten und Skripte beilegen
- Automatisiert mit Selbsttest

# Eine Lösung: org-mode + autotools

## Emacs Org-Mode

```
#+BEGIN_SRC python
import pylab
data = pylab.genfromtxt(
    "data.txt")
pylab.plot(data)
pylab.savefig(
    "image.png")
print "#+caption: desc"
print "[[./image.png]]"
#+END_src
```

## GNU autotools distcheck

```
autoreconf -i && \
./configure && \
make distcheck
→ repro-pub-0.1.tar.gz
```



Versuchsaufbau **exakt** beschreiben.

# Danke für Ihre Aufmerksamkeit!

Bitte stellen Sie Ihre Fragen

# Quellen

- Emacs: [gnu.org/s/emacs](http://gnu.org/s/emacs)
- Org-Mode: [orgmode.org](http://orgmode.org)
- GNU autoconf: [gnu.org/s/autoconf](http://gnu.org/s/autoconf)
- GNU automake: [gnu.org/s/automake](http://gnu.org/s/automake)
- Einfacher Einstieg:  
[draketo.de/light/english/free-software/makefile-to-autotools](http://draketo.de/light/english/free-software/makefile-to-autotools)

## Kontakt

[arne.babenhauserheide@kit.edu](mailto:arne.babenhauserheide@kit.edu) oder [arne\\_bab@web.de](mailto:arne_bab@web.de)

# Hier genutzte Werkzeuge

- Autoconf (configure.ac)
- Automake (Makefile.am)
- Emacs + org-mode (vortrag.org)

# configure.ac

dn1 run 'autoreconf -i' to generate a configure script.

dn1 Then run ./configure to generate a Makefile.

dn1 Finally run make to generate the project.

```
AC_INIT([Repro Pub], [0.5.0],
        [arne.babenhauserheide@kit.edu])
# Check for programs I need for my build
AC_CANONICAL_TARGET
AC_ARG_VAR([emacs], [How to call Emacs.])
AC_CHECK_TARGET_TOOL([emacs], [emacs], [no])
AS_IF([test "x$emacs" = "xno"],
      [AC_MSG_ERROR([cannot find Emacs.])])
# Run automake
AM_INIT_AUTOMAKE([foreign])
AM_MAINTAINER_MODE([enable])
AC_CONFIG_FILES([Makefile])
AC_OUTPUT
```

# Makefile.am (1/2)

```
# basic definitions
vortrag = vortrag.pdf
vortrag_DATA = vortrag.org data.txt dist-tarball.png
vortragdir = .
# dist_pkgdata_DATA = rohdaten
EXTRA_DIST = ${vortrag_DATA} ${vortrag}
# kill editor backups and latex stuff
MOSTLYCLEANFILES = \\#* *~ *.bak *.vrb *.bbl \
                  *.blg *_flymake.*
CLEANFILES = ${vortrag}
DISTCLEANFILES = ${CLEANFILES} \\#* *~ *.bak *.vrb *.bbl \
                  *.blg *_flymake.* auto/*el

all : ${vortrag}
```

## Makefile.am (2/2)

```
# emacs org-mode beamer build instructions
${vortrag} : ${vortrag_DATA}
    if test "$<" != "$(notdir $<)"; then \
        cp -u "$<" "$(notdir $<)"; fi
    echo yes | @emacs@ --batch --load "~/.emacs" \
        --visit "$(notdir $<)" \
        --funcall org-beamer-export-to-pdf
    if test "$<" != "$(notdir $<)"; \
    then rm -f "$(notdir $<)"; \
        rm -f $(basename $(notdir $<)).tex \
            $(basename $(notdir $<)).tex~ \
            auto/$(basename $(notdir $<)).el; \
    else rm -f $(basename $<).tex \
        $(basename $<).tex~ \
        auto/$(basename $<).el; \
    fi
```