

Softwareentwicklung  
in der Wissenschaft

# Planet Simulator

Enno Köster

# Gliederung

- Überblick
- Software
- Entwicklung
- Abgleich mit ISO-Norm
- Verbesserungsvorschläge
- Fazit

# Überblick

## Institut

- Meteorologisches Institut
  - Theoretische Meteorologie
    - Langzeituntersuchungen zum Klima
    - Hurrikan-Weg-Vorhersage
- CliSAP
  - Exzellenzinitiative des Bunds
  - Stellt Mittel unter anderem für die Entwicklung des Planet Simulator bereit

# Überblick

## Software

- Planet Simulator
  - Numerisches Modell
  - Intermediate precision
  - Kontinuierliche Weiterentwicklung
  - Modularer Aufbau
  - GUI einzigartig
  - Besonders einfach zu benutzen

# Überblick

## Interviewpartner

- Edilbert Kirk
  - Kurzer Lebenslauf:
    - 1977:Diplom in Kiel (Nebenfach: Ozeanographie)
    - 1984: Doktor in Köln (Nebenfach: Informatik)
    - 1985–heute: Wissenschaftler an der Uni Hamburg
  - Rolle im Projekt
    - Projektleiter
    - Entwickler
    - Admin

# Überblick

## Entwickler

- Zwei aktive Entwickler am CliSAP
- Doktoranden entwickeln Module und Erweiterungen im Rahmen ihrer Forschung
- Inzwischen auch Kooperation mit einigen ausländischen Unis
  - Staubmodell
  - Vegetationsmodell aus Maryland

# Überblick

## Nutzer

- Weltweit eingesetzt
- Universitäten
  - In der Lehre
  - Aber auch in der Forschung

# Überblick

## Kollaboration und Releases

- Treffen der Hauptentwickler
  - Termin flexibel nach Notwendigkeit
  - Alle paar Monate und kurz vor dem Release
- Informelle Usergroup über Mailingliste
  - Für kleine Änderungen zuständig
- Außerdem Support über Forum



# Software

## Versionshistorie

- Erste Basis: Simple-GCM
  - Gute Ideen, grauenvoller Code
- 90er: PUMA
  - Refaktorisierung des Codes
- 2000: Planet Simulator
  - Neue erweiterte Version von PUMA
  - Parallele Weiterentwicklung von PUMA (Dynamischer Kern) und Planet Simulator (komplettes Klimamodell)

# Software

## Abhängigkeiten

- Xlib für die GUI
- NetCDF-Bibliothek für Postprocessor
- MPI für Kommunikation
  - Früher mpich2, jetzt openMPI
- Tcsh für MoSt (ModelStarter)
- Rest interner Code
  - Speziell angepasste FFT, etc

# Software

## Ein-/Ausgabedaten

- Eingabe:
  - Benötigt nur Randdaten
  - Weitere Daten möglich
  - Üblicherweise nur ein paar Mbyte
- Ausgabe
  - Eigenes kompaktes Datenformat
  - z.B. keine Winddaten
  - Postprocessor wandelt z.B. in NetCDF um

# Software

## MoSt und Planet Simulator GUI

MoSt - Model Starter (16.00) - University of Hamburg

UH PUMA Planet Simulator SAM SOM

Pre-process Save & Exit Save & Run Abort

**Model**

PUMA  
 SAM  
 SOM  
 Planet Simulator  
 Earth  Mars

**Modules**

ML Ocean  
 LSG Ocean  
 Sea Ice  
 Vegetation

**Parallelism**

# of CPUs

**Resolution**

Latitudes  
 Levels

**Options**

Debug mode  
 Write Output  
 Run with GUI  
 Orography  
 Annual cycle  
 Diurnal cycle

**Simulation**

Start year  
 Years to run

**Namelist**

KICK	1
NAFTER	24
NDDEL	8
NDIAG	240
NFLS	1
NHZ	0
NLEVT	9
NRUIDO	0
NSPONGE	1
NSTRATO	0
NSYMM	0
NTSPD	24
NVG	0
NYODEN	0
ALRPV	0.002
DISP	0,0
DTEP	53,0
DTNS	-70,0
DTROP	12000,0
DTTRP	2,0
DVDIFF	0,0
EDGEPV	10,0
FLSAMP	15,0
FLSDP	9000,0
FLSOFF	-5,0
FLSP0	10000,0
PMAXPV	10000,0
RADPV	50,0
ROTSPD	1,0
SIGMAX	6,0000e-07
TOK	250,0
TDISS	0,25
TGR	288,0

**Orography**

Lon: (105;164) Lat: (-45;-11)  
 Change [gpm]

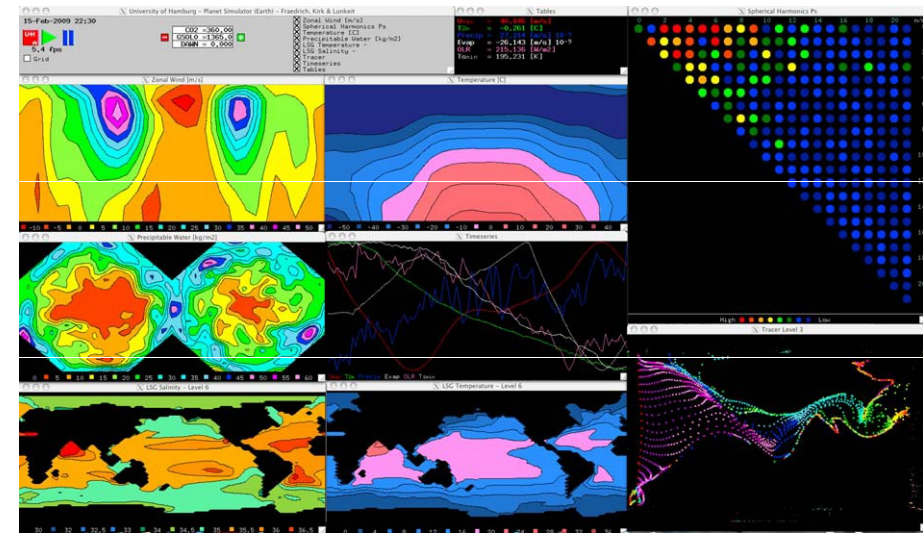
**Spherical Harmonics mode selector**

0	2	4	6	8	10	12	14	16	18	20	0	
●	●	●	●	●	●	●	●	●	●	●	●	2
●	●	●	●	●	●	●	●	●	●	●	●	4
●	●	●	●	●	●	●	●	●	●	●	●	6
●	●	●	●	●	●	●	●	●	●	●	●	8
●	●	●	●	●	●	●	●	●	●	●	●	10
●	●	●	●	●	●	●	●	●	●	●	●	12
●	●	●	●	●	●	●	●	●	●	●	●	14
●	●	●	●	●	●	●	●	●	●	●	●	16
●	●	●	●	●	●	●	●	●	●	●	●	18
●	●	●	●	●	●	●	●	●	●	●	●	20

**PUMA T21 only!**

MB 1: Toggle mode  
 MB 2: Toggle column  
 MB 3: Toggle line

● Mode is on  
 ● Mode is off  
 n : Total Wavenumber  
 m : Zonal Wavenumber  
 ● Switch all modes on  
 ● Switch all modes off



# Entwicklung

## Genutzte Tools

- Versionierung: Subversion
- IDE: keine
  - Fortran-Unterstützung in IDEs meistens schlecht
- Debugging: GDB und Valgrind

# Entwicklung

## Codequalität

- Keine Einhaltung von ISO-Normen
- Nur klare Konventionen für Code
  - Namensregeln für Variablen
  - Klar definierte Schnittstellen für Module
    - Init
    - Zeitschritt
    - Epilog
  - Gute Kommentierung
    - Welche z.T. Von Hauptentwicklern nachbearbeitet wird

# Entwicklung

## Tests / Validation

- Scripte
  - Restart-Check
    - Besonders wichtig
    - Überprüft Checkpointing-Mechanismus
- Vergleich mit älteren Versionen
- Vergleich mit von Hand ausgerechneten Daten
  - Nur für einzelne Module
- Und natürlich Vergleich mit Realdaten

# Abgleich mit ISO-Norm

## Funktionalität

- Angemessenheit
  - Wahrscheinlich gegeben
- Richtigkeit
  - Durch Validation sichergestellt
- Interoperabilität
  - Arbeitet mit gängigen Klimadatenformaten
- Sicherheit
  - Nicht relevant
- Ordnungsmäßigkeit
  - Nicht relevant



# Abgleich mit ISO-Norm

## Zuverlässigkeit

- Reife
  - Kein Hinweis auf Probleme → Gegeben
- Fehlertoleranz
  - Unbekannt
- Wiederherstellbarkeit
  - Gegeben durch Wiederaufnahme der Tätigkeit am letzten Checkpoint
- Konformität
  - Nicht angestrebt

# Abgleich mit ISO-Norm

## Benutzbarkeit

- Verständlichkeit
  - Unbekannt, Nutzer meist Meteorologen und damit entsprechend vorgebildet
- Erlernbarkeit
  - Gut, durch GUI und Dokumentation einfach zu verstehen
- Bedienbarkeit
  - Gut, GUI unterstützt Erstbedienung, ist aber nicht zwingend einzusetzen
- Attraktivität
  - Weltweit eingesetzt, offensichtlich großes Interesse

# Abgleich mit ISO-Norm

## Effizienz

- Zeitverhalten
  - Unbekannt
- Verbrauchsverhalten
  - Im Vergleich zu anderen Modellen sparsam
  - Skaliert gut auf mehreren Kernen
  - Skaliert schlecht auf mehreren Knoten
- Konformität
  - Keine Einhaltung von Normen angestrebt

# Abgleich mit ISO-Norm

## Änderbarkeit

- Analysierbarkeit
  - Wahrscheinlich gut, Code gut kommentiert
- Modifizierbarkeit
  - Wahrscheinlich gut, Code modular aufgebaut
- Stabilität
  - Wahrscheinlich gut, keine Probleme bekannt
- Testbarkeit
  - Geringer Aufwand

# Abgleich mit ISO-Norm

## Übertragbarkeit

- Anpassbarkeit
  - Hervorragend durch modularen Aufbau
- Installierbarkeit
  - Sehr gut, durch Unterstützung durch Configure-Skript
  - Anpassung an unbekannte Konfigurationen einfach
- Koexistenz
  - Arbeitet mit gängigen Klimadatenformaten
- Austauschbarkeit
  - Unbekannt
- Konformität
  - Keine Konformität angestrebt

# Verbesserungsvorschläge

- Keine dringenden Probleme im Projekt
- Direkter Zugang zum Repository von außen
  - Im Moment Code nur per Tarball zugänglich
- Wechsel des Versionierungssystems
  - Entwicklerzahl steigt langsam aber stetig
  - Dadurch evtl. Git oder Mercurial in Zukunft besser
- Wechsel von Xlib zu Qt/gtk
  - Portierung nach C++ sowieso schon im Gange

# Fazit

- Interessantes Interview
- Großartiges Projekt
- Gesunder Menschenverstand statt Verfolgung von Normen
- Nur Kleinigkeiten verbesserungswürdig

# Quellen

- The Planet Simulator: A Coupled System of Climate Modules with Real Time Visualization
  - <http://www.mi.uni-hamburg.de/fileadmin/files/forschung/theomet/docs/pdf/2009-Kirk-Frae-Ulmen-ecp094507.pdf>
- Wikipedia-Artikel zur ISO/IEC 9126
  - [http://de.wikipedia.org/wiki/ISO/IEC\\_9126](http://de.wikipedia.org/wiki/ISO/IEC_9126)
- Webseite der Theoretischen Meteorologie
  - <http://www.mi.uni-hamburg.de/Theoretische.6.0.html>