

WaterGAP –  
A global hydrology and water use  
model

Seminar Softwareentwicklung in  
der Wissenschaft

Maral Chimed-Ochir



# Gliederung

1. Übersicht
2. WaterGAP
  - 2.1 Beispiel: Räumliche Auslösung
  - 2.2 Entwicklung
  - 2.3 Einsatz
3. Modellschema
  - 3.1 Modellinput & Pre-Processing
  - 3.2 Modellentwicklung
  - 3.3 Modelloutput & Post Processing
4. Vorgehensweise
5. Testen
6. Entwicklungsschritte
7. Qualitätskontrolle
8. Besonderheiten
9. Zusammenfassung
10. Fazit



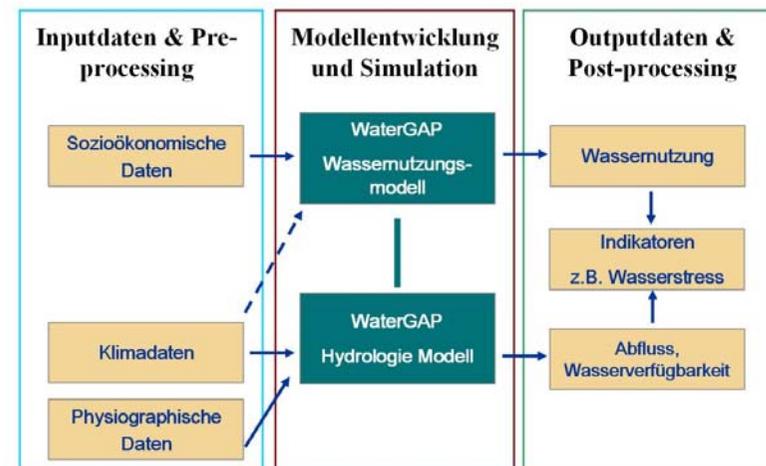
# 1. Übersicht

- **Institution:** Center for Environmental Systems Research (CESR), Universität Kassel
- **Wissenschaftliche Software:** WaterGAP (Water Global Assessment and Prognosis)
- **Interviewpartner:** Tim aus der Beek (Dipl.-Hydrologe), wissenschaftlicher Mitarbeiter und Doktorand
- **Die Rolle des Interviewpartners:** Weiterentwicklung von WaterGAP3, zurzeit bei der Entwicklung eines Teilmoduls (Bewässerung)

# 1. Übersicht (Fortsetzung)

- **Forschungsschwerpunkt:** räumlich explizite Modellierung der globalen Wasserverfügbarkeit, Wasserqualität und Wassernutzung und Simulation zukünftiger Entwicklung
- **Problemstellung der Arbeit:** Wann, wo und aus welchem Grund sind welche Wasserressourcen vorhanden bzw. genutzt?
- **Ziel:** natürliche und sozioökonomische Prozesse realitätsnah in einem Computermodell wiederzugeben und damit weitergehende Fragestellungen bearbeiten zu können

WaterGAP Modellschema



Bilderquelle: Tim aus der Beek „WaterGAP Introduction“

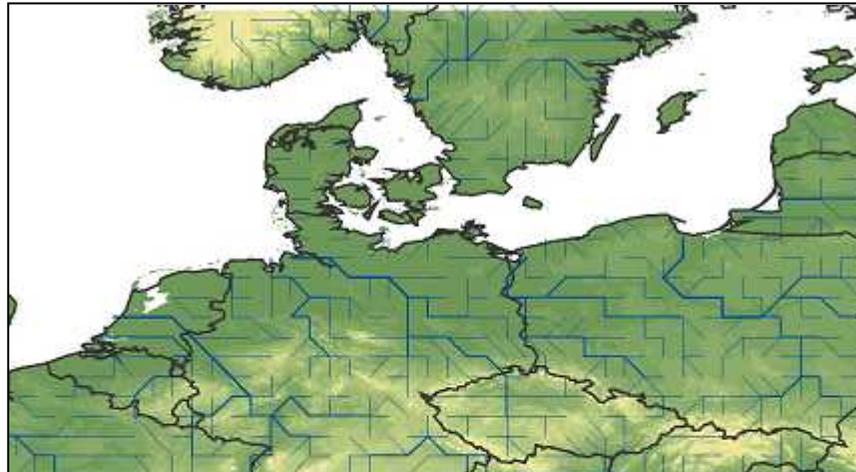


## 2. WaterGAP: Water - Global Assessment and Prognosis

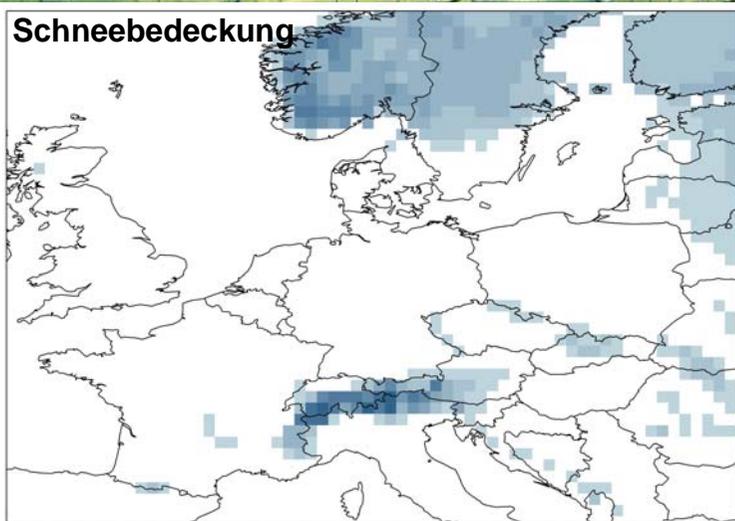
- simuliert den Einfluss von
  - naturräumlichen und klimatischen Bedingungen auf die Wasserverfügbarkeit
  - demographischen, ökonomischen und technologischen Veränderungen auf die Wassernutzung
  
- deckt gesamte Landmasse der Erde mit Ausnahme der Antarktis ab
  
- räumliche Auflösung von 0,5°x0,5° und 5x5 Bogenminuten
  - Rasterzellgröße ca. 2000 – 3000 km<sup>2</sup> und 40-70 km<sup>2</sup>
  - 69896 Zellen und 2,3 Mil. Zellen
  
- zeitliche Auflösung:
  - Berechnungsschrittweite: 1 Tag
  - Ausgabe Ergebniszeitreihe: 1 Monat, 1 Tag

## 2.1. Beispiel: Räumliche Auflösung

WaterGAP2.x: 0.5°



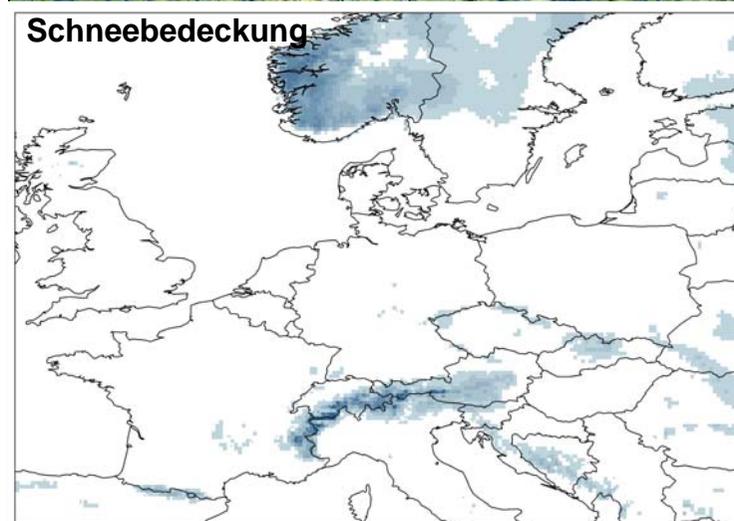
Schneebedeckung



WaterGAP3.x: 5'



Schneebedeckung



Bilderquelle: Tim aus der Beek „WaterGAP Introduction“



## 2.2. WaterGAP-Entwicklung

### ■ **Historie der Software:**

- WaterGAP1: 1996 – 1999
- WaterGAP2: 2000 – bis heute
- WaterGAP3: 2007 – bis heute

### ■ **Entwickler:**

- WaterGAP3 mit 10 Entwicklern am CESR
- WaterGAP2 mit 5 Entwicklern an der Universität Frankfurt bzw. mit 2 Entwicklern am Geoforschungszentrum Potsdam

->Die Entwickler sind gleichzeitig Nutzer!



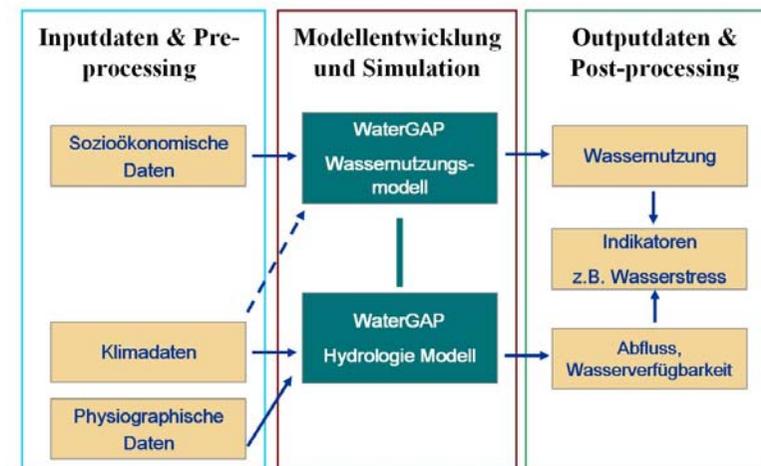
## 2.3. WaterGAP - Einsatz

- **Benutzer:**
  - Entwickler = Hauptbenutzer
  - Evtl. Master- und Diplomstudierende
  
- **Ergebnis-Usergroup:**
  - Modellergebnisse können über WebGIS analysiert und heruntergeladen werden
  - Veröffentlichungen erscheinen in den wissenschaftlichen Journalen und den Reports von UNEP, EU, EEA usw.
  
- WaterGAP ist nicht frei verfügbar!  
->Zu komplex, kein Handbuch und keine GUI

### 3. Modellschema

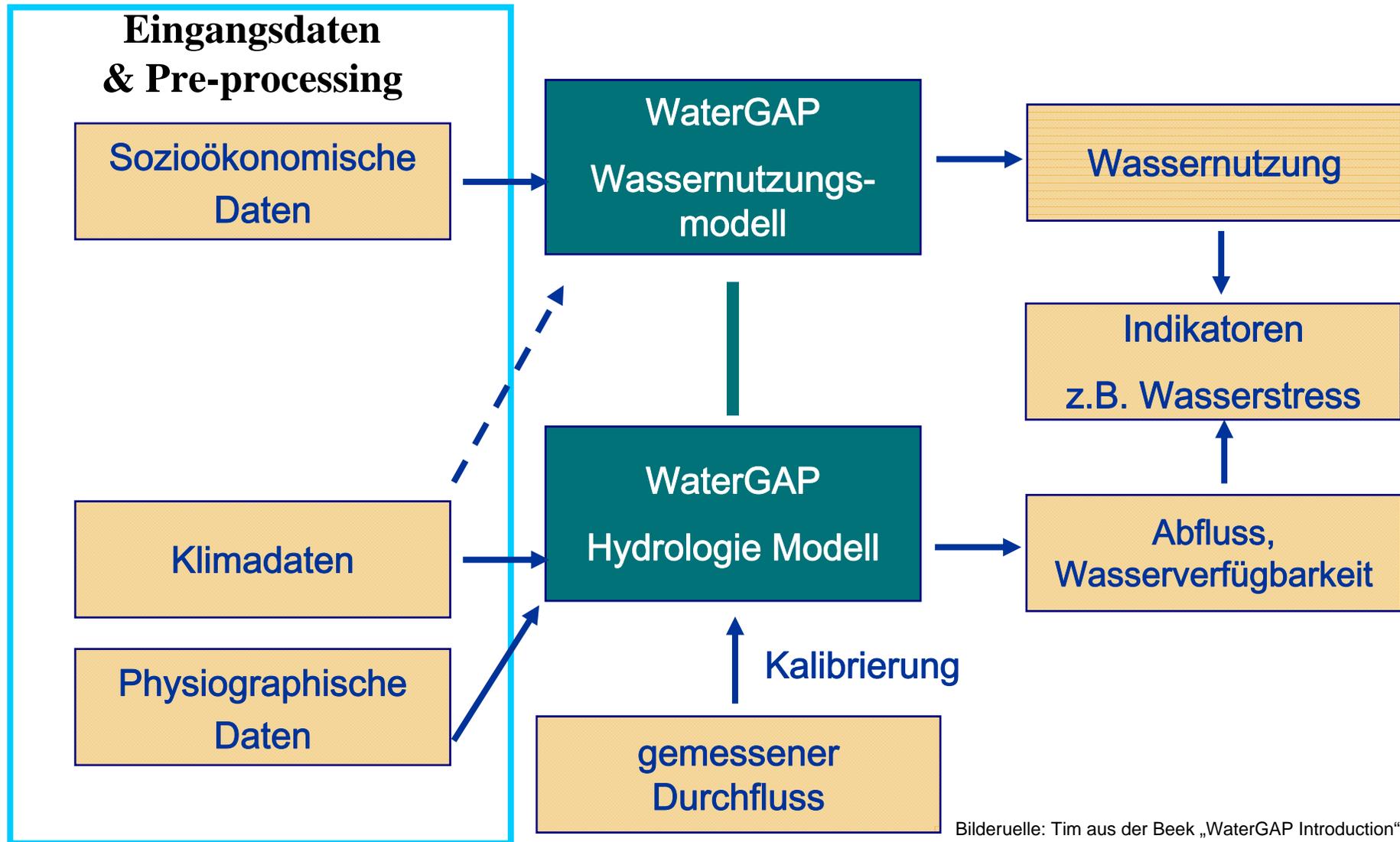
- WaterGAP besteht aus folgenden Teilmodellen:
  - Hydrologie-Modell: ca. 20 000 Zeilen
  - 5 Wassernutzungsmodelle (Bewässerung, Viehhaltung, Elektrizitätsproduktion, verarbeitendes Gewerbe, Haushalte): ca. 31 000 Zeilen
  - Wasserqualitätsmodell: ca. 11 000 Zeilen
- Die Teilmodelle werden durch Soft-Coupling miteinander verknüpft
- Alle Teilmodelle innerhalb des Frameworks sind konzeptionell mathematisch-physikalisch

WaterGAP Modellschema



Bilderquelle: Tim aus der Beek „WaterGAP Introduction“

### 3.1. Eingangsdaten & Pre-Processing



Bilderuelle: Tim aus der Beek „WaterGAP Introduction“



## 3.1. Inputdaten & Pre-Processing (Fortsetzung)

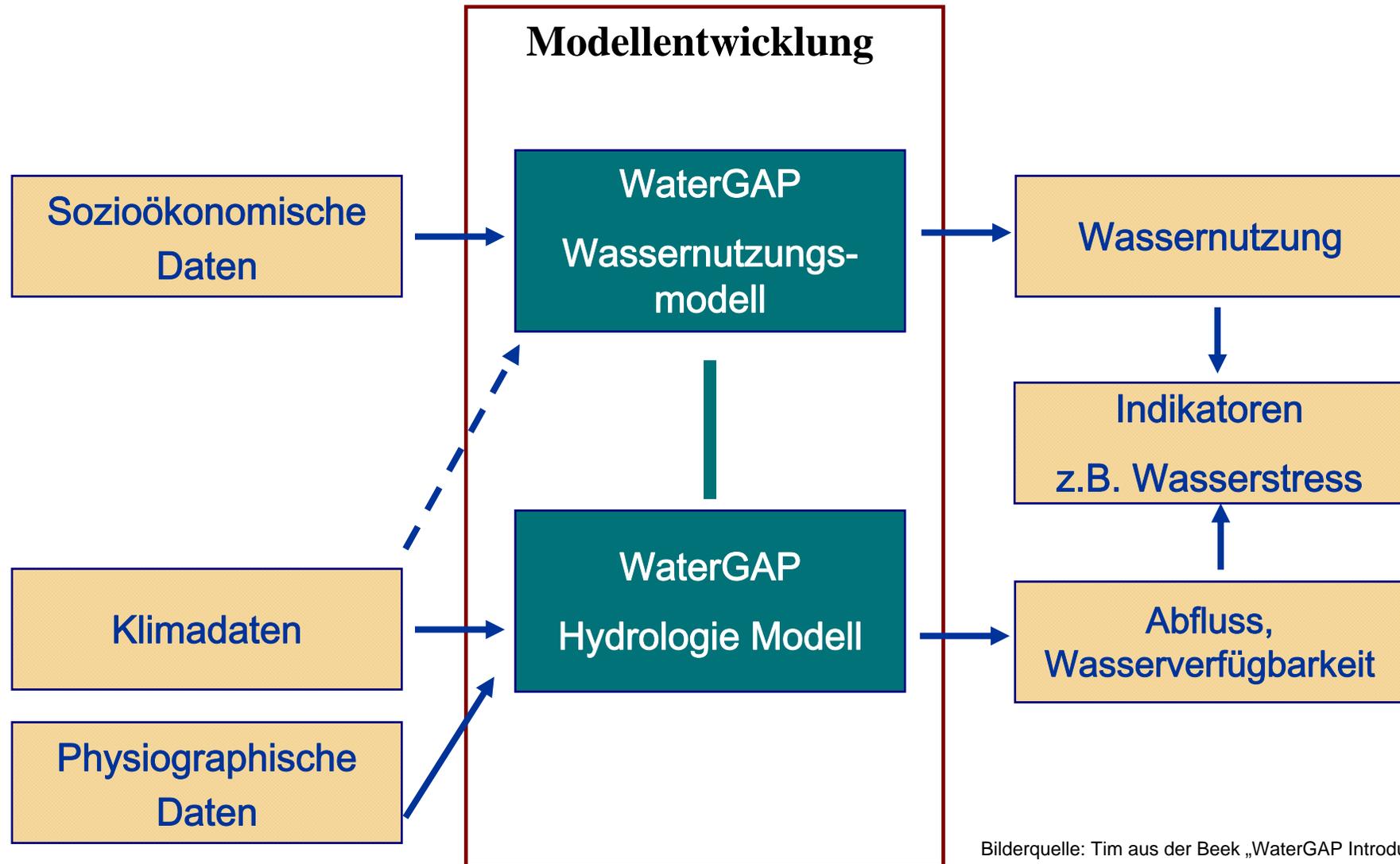
### ■ **Eingangsdaten:**

- Klimadaten (Temperatur, Niederschlag etc.): tägliche und monatliche Grids von 1901 bis 2002 mit ca. 350 GB Größe als binäre Datei
- Basisdaten (physiographische Daten: Boden, Landnutzung, Bevölkerung etc.): statische Grids mit 2GB
- Sozioökonomische Daten (gemessene Abflüsse, Kraftwerke etc.): ca. 300 GB

### ■ **Pre-Processing:**

- Skalierung der Daten auf erforderliche räumliche und zeitliche Aggregation (Formatierung)
- Homogenisierung der Dateiformate (Umwandlung)
- Konzepte zur Organisation der Inputdaten (z.B. DBMS)

## 3.2. Modellentwicklung



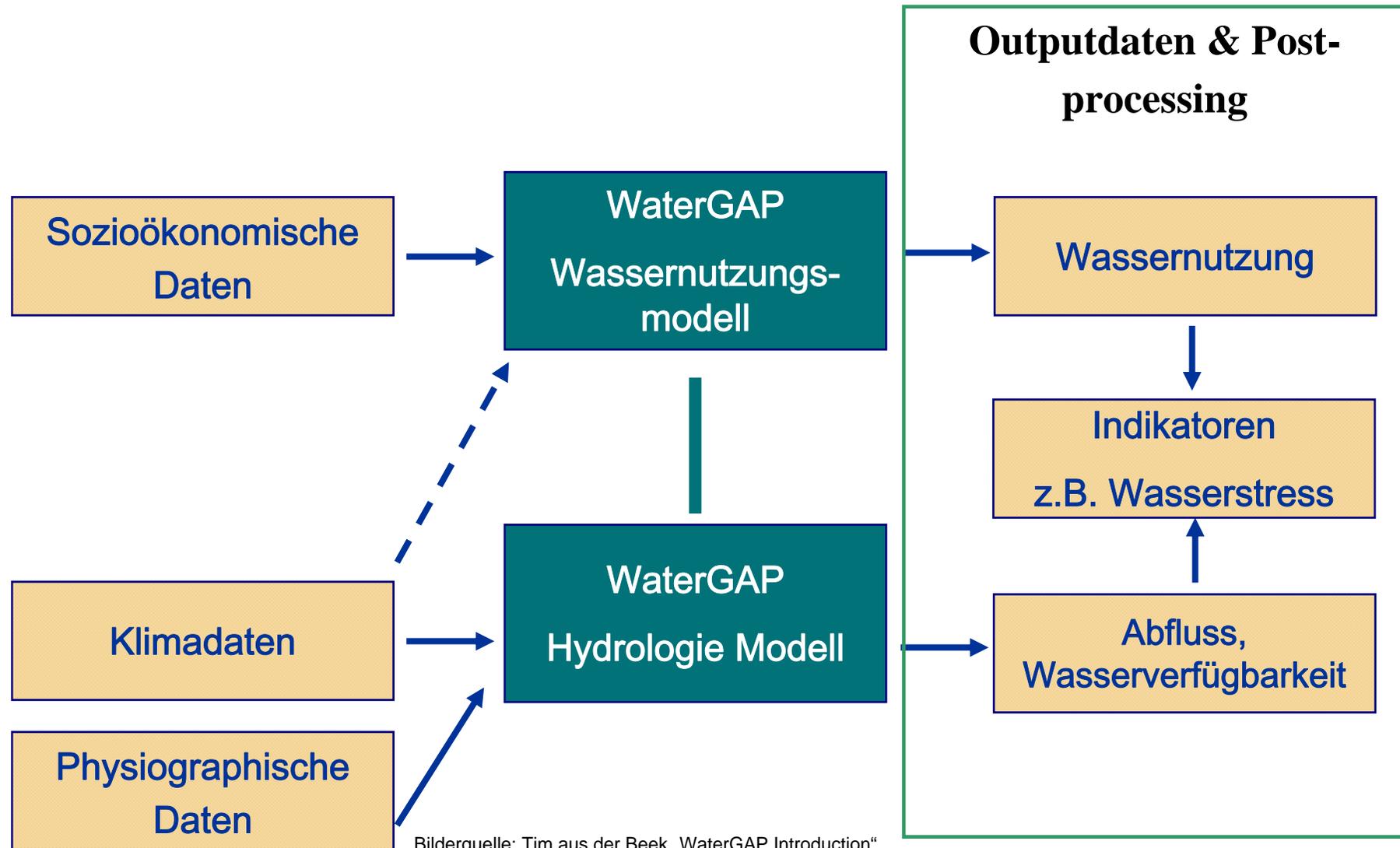
Bilderquelle: Tim aus der Beek „WaterGAP Introduction“



## 3.2. Modellentwicklung (Fortsetzung)

- **Programmiersprache:** Mischung aus C und C++ (je nach Teilmodell)
- **Entwicklungswerkzeuge:** Eclipse, Texteditoren
- **Software-Bibliotheken:**
  - MPI-Bibliotheken zum Parallel-Processing
  - NetCDF-Bibliothek zum Post-Processing von binären Dateien
  - MySQL zum einlesen aus der DB
- **Genutzte Hardwaresysteme:**
  - am Cluster der Universität Kassel: nur Hydrologie-Modell
  - Workstation AMD Opteron 2218 mit 4 Kernen, 6,2 TB HD, 32 GB RAM, Linux-OS (SUSE): restliche Modelle
- Teilmodelle werden von unterschiedlichen Entwicklern entwickelt

### 3.3. Modelloutputs & Post Processing



Bilderquelle: Tim aus der Beek „WaterGAP Introduction“



### 3.3. Modelloutputs & Post processing (Fortsetzung)

- **Modelloutputs:** tägliche und monatliche Grids mit ca. 40 dynamischen Ausgabeparametern (Wasserverfügbarkeit, Grundwasserneubildung etc.) von 1901 bis 2100 als binäre Dateien (UNF-Format) und Textdateien für einzelne Gridzellen und Länder
  
- **Post Processing:**
  - Visuelle Überprüfung mittels GIS (z.B. ArcGIS 9)
  
  - Einsatz von ca. 15 Programmen (z.B. VBA) zur Umwandlung (z.B. UNF<->NetCDF), Auswertung (z.B. Evaluierung der Modellgüte) und Formatierung (z.B. zeitlich/räumliche Aggregation) des Modelloutputs



## 4. Vorgehensweise der Modellentwicklung

1. **Literaturrecherche**
2. **Methodenauswahl**
3. **Datenakquise** (zu assimilierende Inputdaten für Bearbeitung der neuen wissenschaftlichen Fragestellung überhaupt vorhanden?)
4. **Plan zur Implementierung**
5. **Implementierung**
6. **Test/Validierung des neuen Modelloutputs**

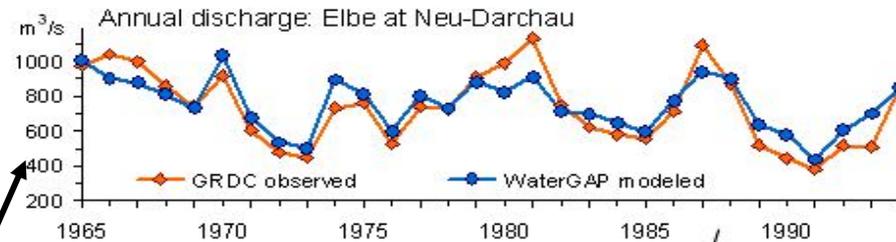
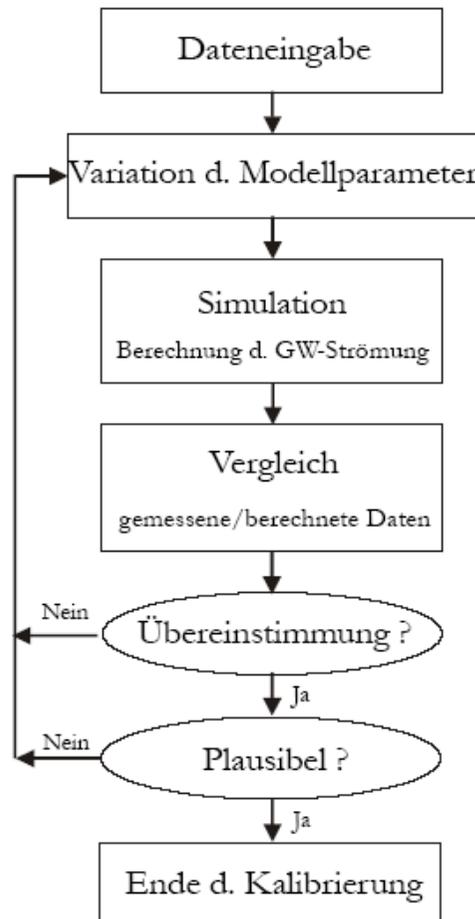
-> Eventuell zurück zur Literaturrecherche falls ausgewählte Methode nicht passend



## 5.1. Debuggen

- **Debuggen:** kein klassisches Debugging, da zu aufwendig wegen der Parallelisierung  
-> stattdessen bei Fehlern Ausgabe von Zwischenwerten über printf und Überprüfung des Codes
- Es gibt keine Softwaretests nach bestimmten Verfahren.
- Wenn keine Fehler vorliegen, wird das Modelloutput mit den gemessenen Werten verglichen.  
-> weiter zum Schritt Kalibrierung

## 5.2. Kalibrierung/Validierung



**Kalibrierung**

**Validierung**

**Sensitivitätsanalyse**

**Unsicherheitsanalyse**

Quelle: Tim aus der Beek „WaterGAP Introduction“

## 5.2. Kalibrierung/Validierung (Fortsetzung)

### ■ Validation: (zur Erinnerung)

Testen wissenschaftlicher Software  
Ist-Zustand

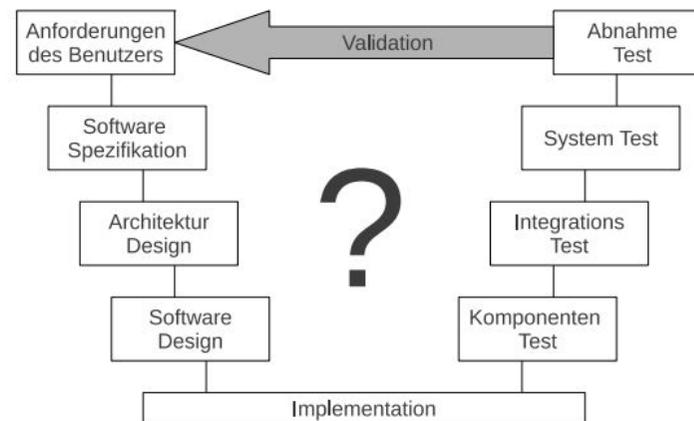


Abbildung: V-Modell Ist-Zustand in der Wissenschaft  
(Quelle: Abbildung von Florian Ehmke)

### ■ Validation der Ergebnisse:

- Vergleich der Modellergebnisse mit gemessenen Daten, sowie mit Statistiken
- Vergleich zusätzlich mit anderen hydrologischen Modellen
- falls notwendig, Modellergebnisse durch Experten analysieren lassen.



## 6. Entwicklungsschritte

- **Releasemanagement/Versionskontrolle:**
  - Verwendung von Subversion
- **Mittel der Interaktion:**
  - Interne schriftliche technische Berichte und Kurzanleitungen
- **Kollaboration:**
  - Hauptbenutzergroup an der Uni Kassel, Frankfurt und GFZ-Potsdam ca. 15-20 Entwickler
- **Softwareeinführung:** durch involvierte Entwickler
- **Softwarelebenszyklus:** nicht festgelegt
- **Pflege:** Programmierung mit Kommentierung im Code, anschließendes Testen, Einpflegen und Release in Subversion
- **Datenarchivierung:**
  - Nach Abschluss des Projekts werden die Modellergebnisse gezippt und im Back-Up System archiviert



## 7. Qualitätskontrolle: Funktionalität

- Es wird angenommen, dass WaterGAP richtig funktioniert.
  
- **Funktionalität**
  - Angemessenheit: keine Angabe
  
  - Richtigkeit: wird angenommen
    - Modellergebnisse schließt aber „right for wrong reasons“ nicht aus
    - Die empirische Formeln stellen eine Fehlerquelle dar.
  
  - Interoperabilität: keine Angabe
  
  - Sicherheit: als sicher angesehen
    - Daten und Code liegen auf dem internen Server
    - Das Programm ist nicht online erreichbar
  
  - Ordnungmäßigkeit: keine Angabe



## 7. Qualitätskontrolle: Zuverlässigkeit

### ■ Zuverlässigkeit

- Reife: ausgereift genug um stabil zu laufen
- Fehlertoleranz: Programmabbruch nur bei schwerwiegenden Fehlern (z.B. fehlender Input)
- Wiederherstellbarkeit: durch Neustart der Software
- Konformität: keine Angabe



## 7. Qualitätskontrolle: Benutzbarkeit

- Das Modell wird hauptsächlich von Entwicklern benutzt.  
-> in diesem Kontext stellt die Benutzbarkeit kein Problem dar.
  
- **Benutzbarkeit für den Einsteiger**
  - Verständlichkeit: sehr schwierig
    - Erst muss man komplexe hydrologische Prozesse verstehen.
  
  - Erlernbarkeit: sehr aufwendig
    - Die Einarbeitung dauert mehrere Monate.
  
  - Bedienbarkeit: benutzerunfreundlich
    - Es gibt keine GUI.
  
  - Attraktivität: kaum
  
  - Konformität: keine Angabe  
-> keine Bewertung notwendig, da sie nicht vorgesehen ist.



## 7. Qualitätskontrolle: Effizienz

- Höher Bedarf an Rechenzeit und Speicher  
-> fordert hohe Effizienz!
  
- **Effizienz**
  - Zeitverhalten: optimiert durch Parallelisierung für den Cluster
    - Ein Modelldurchlauf dauert ca. 1,5 Tage.
  
  - Verbrauchsverhalten: erhöhter Bedarf
    - Arbeitsspeicher mit 32GB RAM ist zufriedenstellend.
    - Zielspeicher (Festplatte) ist zu klein (6TB).
  
  - Konformität: keine Angabe  
-> aber hohe Effizienz ist notwendig
  
- Allgemein ist die Software als effizient angesehen.



## 7. Qualitätskontrolle: Änderbarkeit

### ■ Änderbarkeit

- Analysierbarkeit: sehr aufwendig
- Modifizierbarkeit: sehr aufwendig
- Stabilität: nicht voraussehbar
- Testbarkeit: sehr aufwendig



## 7. Qualitätskontrolle: Übertragbarkeit

- Alle 5-6 Jahre werden die Workstation gewechselt.
- **Portabilität**
  - Anpassbarkeit: keine
    - Läuft nur unter Linux (SUSE). Keine Angabe zur Kompatibilität mit anderen Linux-Distributionen
  - Installierbarkeit: keine Installation nötig
  - Koexistenz: nicht bekannt
  - Konformität: keine Angabe
- Die Migration des Systems zu 64bit-Architektur stellte kein Problem dar.



## 8. Besonderheiten

- nur Naturwissenschaftler, keine Informatiker bei der Entwicklung involviert  
->Parallelisierung durch Informatiker
- keine dauerhaften Stelle zur Programmierung/Pflege vorhanden wegen der Querfinanzierung durch Drittmittelprojekte (z.B. BMBF, EU, usw.)
- hauptsächlich Verwendung von Freeware (z.B. Eclipse, MySQL) aus finanziellen Mitteln



## 9. Zusammenfassung

- Es war notwendig die Modellierung zu verstehen, damit die Entwicklungsschritte richtig dargestellt werden können.
- Beim Interview erhielt ich ab und zu Mal keine oder eine falsche Antwort zu softwaretechnischen Fragen.
  - > Der Interviewpartner kennt die formalen Aspekte der Softwareentwicklung nicht
- Die Wissenschaftler sind mit ihren Programmierproblemen auf sich allein gestellt.
- Hauptproblemstellung: Auswahl der richtigen wissenschaftlichen Methoden und mathematisch-physikalische Modellierung
  - > bei Fehlersituationen ist schwierig, die Ursache festzustellen



## 10. Fazit

- SWE ist notwendig um neue wissenschaftliche Fragestellungen zu beantworten.  
->Weiterentwicklung des Modell-Frameworks um klimatische, physiographische und sozioökonomische Veränderungen realitätsnah wiedergeben zu können
- Unterstützend im Pre- und Post-Processing von Input-/Outputdaten
  - Zur Automatisierung der Modelldurchläufe
- Einstieg in die Modellentwicklung ist schwierig für Informatiker.  
->Das Modell ist sehr komplex. Das Fachwissen ist unbedingt notwendig um es zu verstehen.
  - Eine Hilfstätigkeit ist sinnvoller (z.B. zur Parallelisierung, Automatisierung oder Implementierung bestimmter Funktionen)
- Traditionelle Softwareentwicklungsprinzipien müssen nicht an erster Stelle stehen, es schadet aber nicht einige Aspekte einzusetzen, z.B:
  - bessere Mittel zur Interaktion (z.B. Team Foundation Server)
  - besseres Datenmanagement (Back-Up, Versionsmanagement)
  - bessere Dokumentation (z.B. eigenes Wiki)



## Quelle

- Tim aus der Beek:  
„WaterGAP3 Introduction“ 2010
- Alcamo et al:  
„Development and testing of the WaterGAP2 global model of water use and availability“ 2003
- Doell et al:  
„A global hydrological model for deriving water availability indications: model tuning and validation“ 2003



## Bilderquelle

- Aus Dokumentationen und Veröffentlichungen von WaterGAP
- Abbildung aus Florian Ehmkes Folien (F. 19)