Principles of
Integrative Environmental Simulations

Rolf Hennicker

Sebastian Bauer, Stephan Janisch, Matthias Ludwig

Ludwig-Maximilans-Universität München

GLOWA
Danube

gefördert vom Bundesministerium für Bildung und Forschung

Integrative Techniques, Scenarios and Strategies for the Future of Water in the Upper Danube Basin (2000 – 2010)

Research Groups and Investigation Area

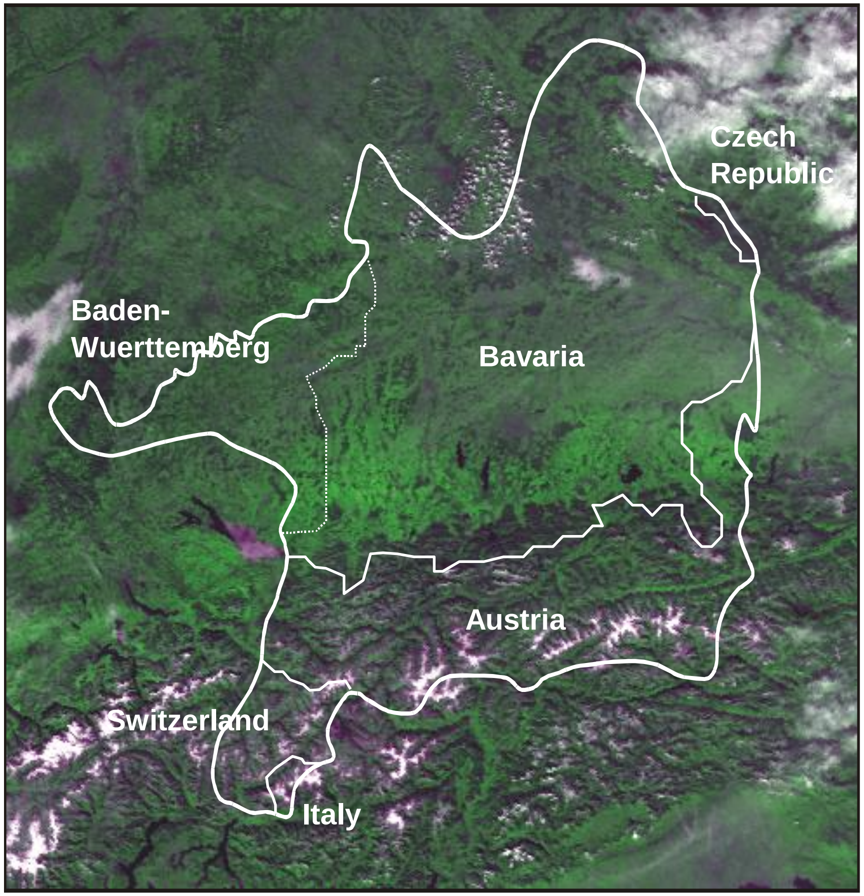
Natural Sciences

- Hydrology
- Plant Ecology
- Glaciology
- Meteorology
- Groundwater
- Surface Water

Social Sciences

- Environmental Psychology
- Tourism Research
- Water Supply
- Agricultural Economics
- Environmental Economics

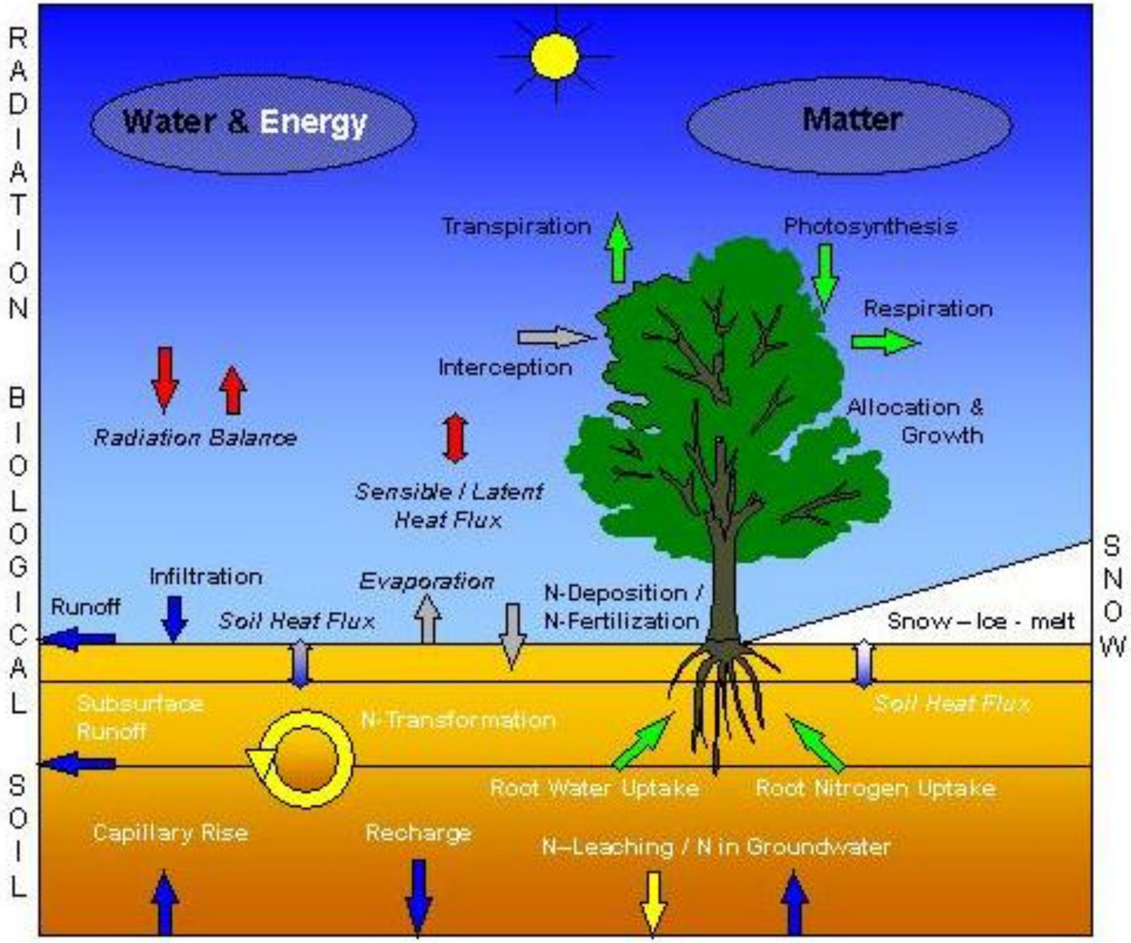
+ Informatics



Upper Danube Basin:

- Area: 77.000 km²
- Population: 8.2 Mio.
- Elevation Gradient: 3300 m

Mutually Dependent Processes in Nature and Society



- „Stand-alone“ modelling of the single processes is not sufficient !
- An integrative view is needed !

General Goal

Development of an integrative platform for

- coupled simulations of various models of natural-science and socio-economic disciplines
- support of decision making on the basis of scenarios for changing climate and society

Tasks of the Informatics group

- support for the conceptual integration of the various disciplines
- development of a framework for coupled simulations

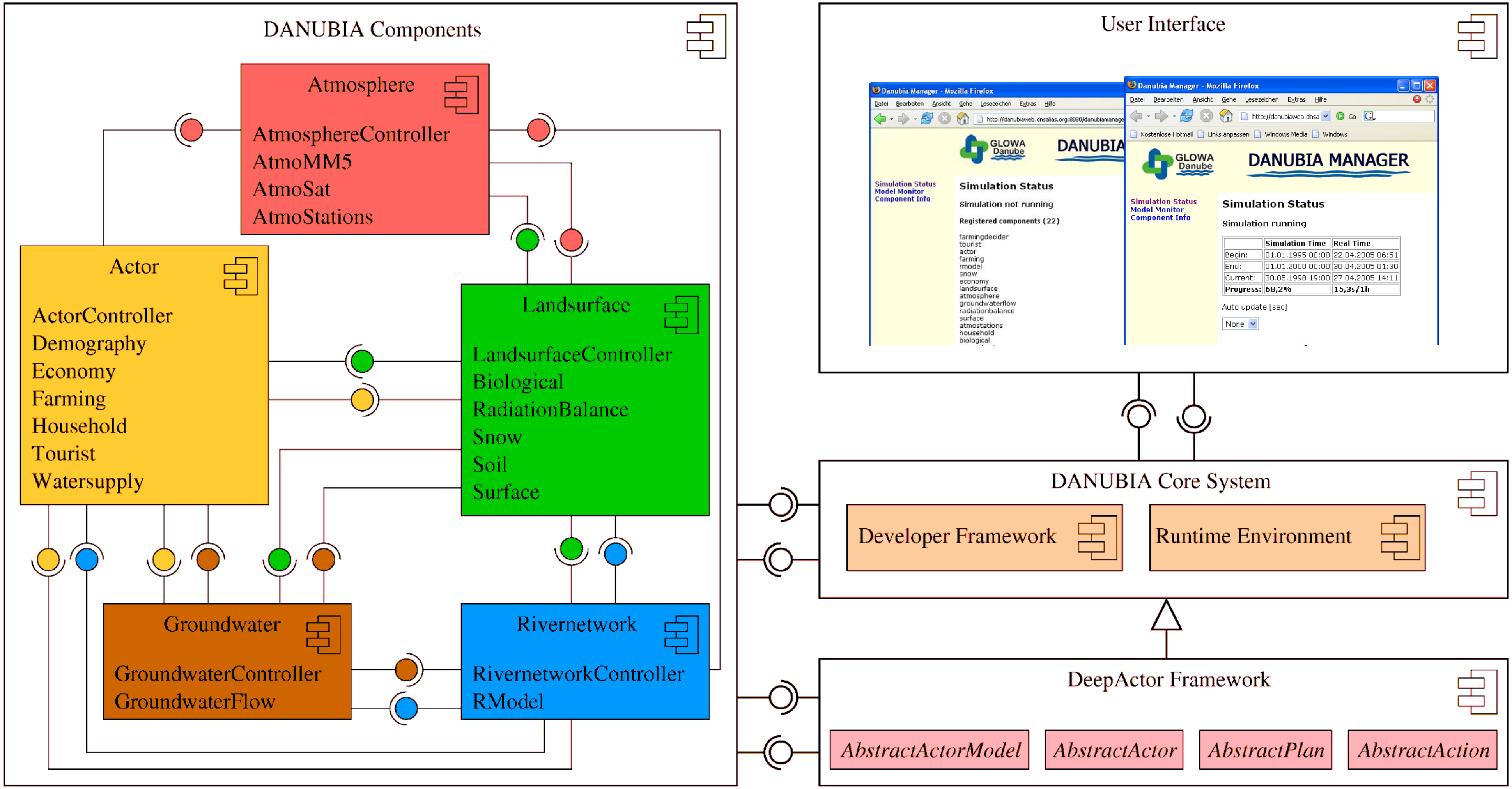
Crucial Aspects of Integrative Simulations

- Identification of interfaces for ***data exchange*** (between the different models)
- Consistent modeling of the ***simulation space***
- Treatment of ***time*** (life cycle and coordination of simulation models)
- Modeling of decision making ***actors*** (housholds, water suppliers, farmers, …)

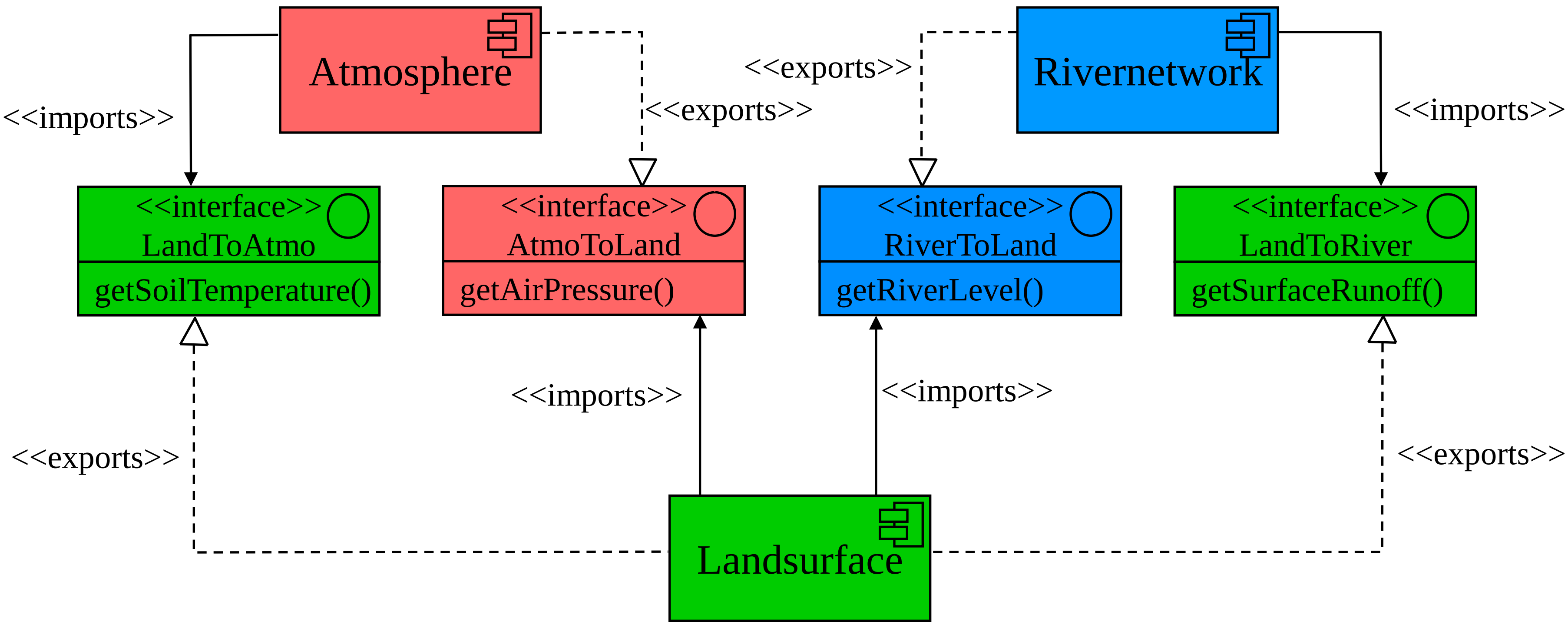
Development Principles

- ***Common graphical modeling language (UML)*** used by all project groups for the description of interfaces, concepts and designs
- ***Framework technology***
  - to facilitate the integration of simulation models,
  - to implement generally valid rules
- ***Formal methods*** to verify critical parts of the integrative system

System Architecture



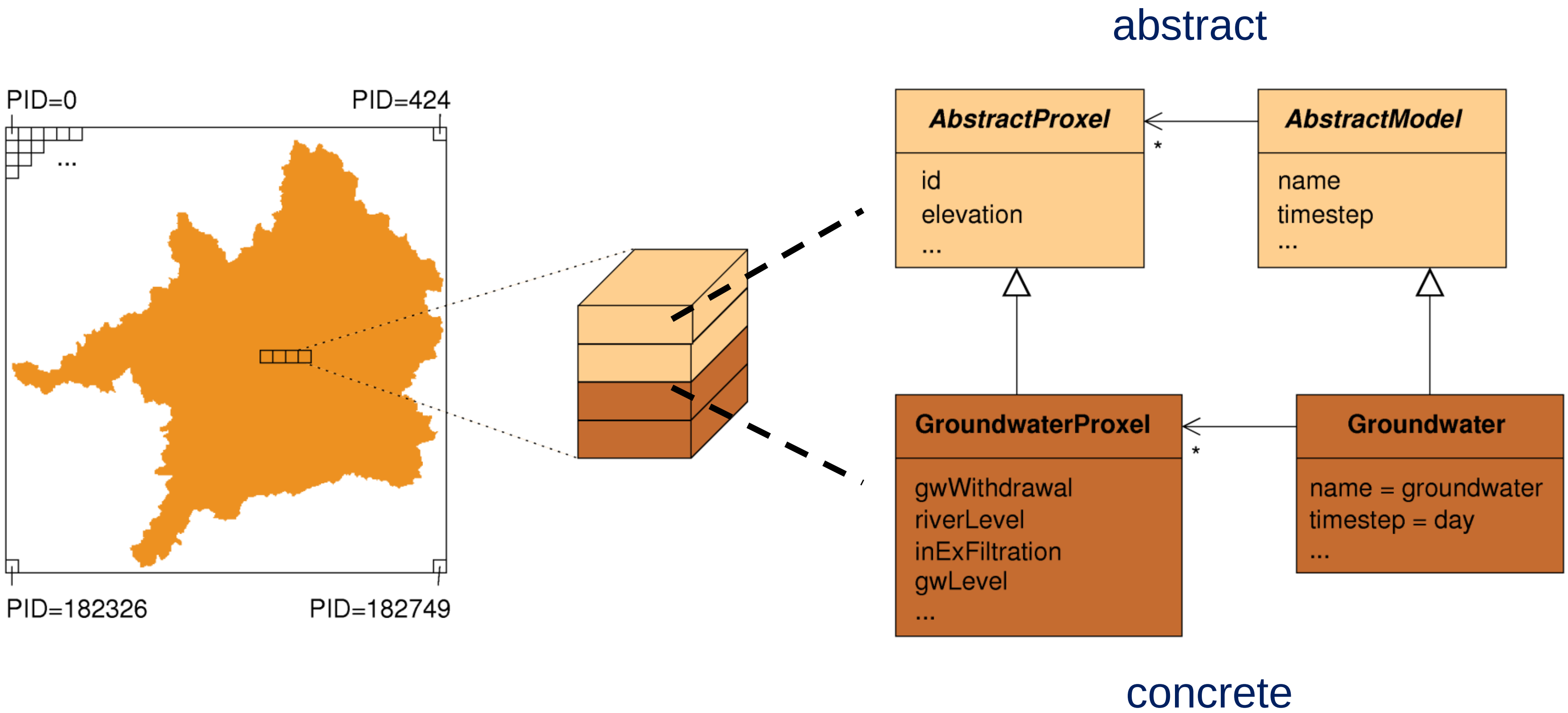
Aspect “Data Exchange”: Modeling of Interfaces



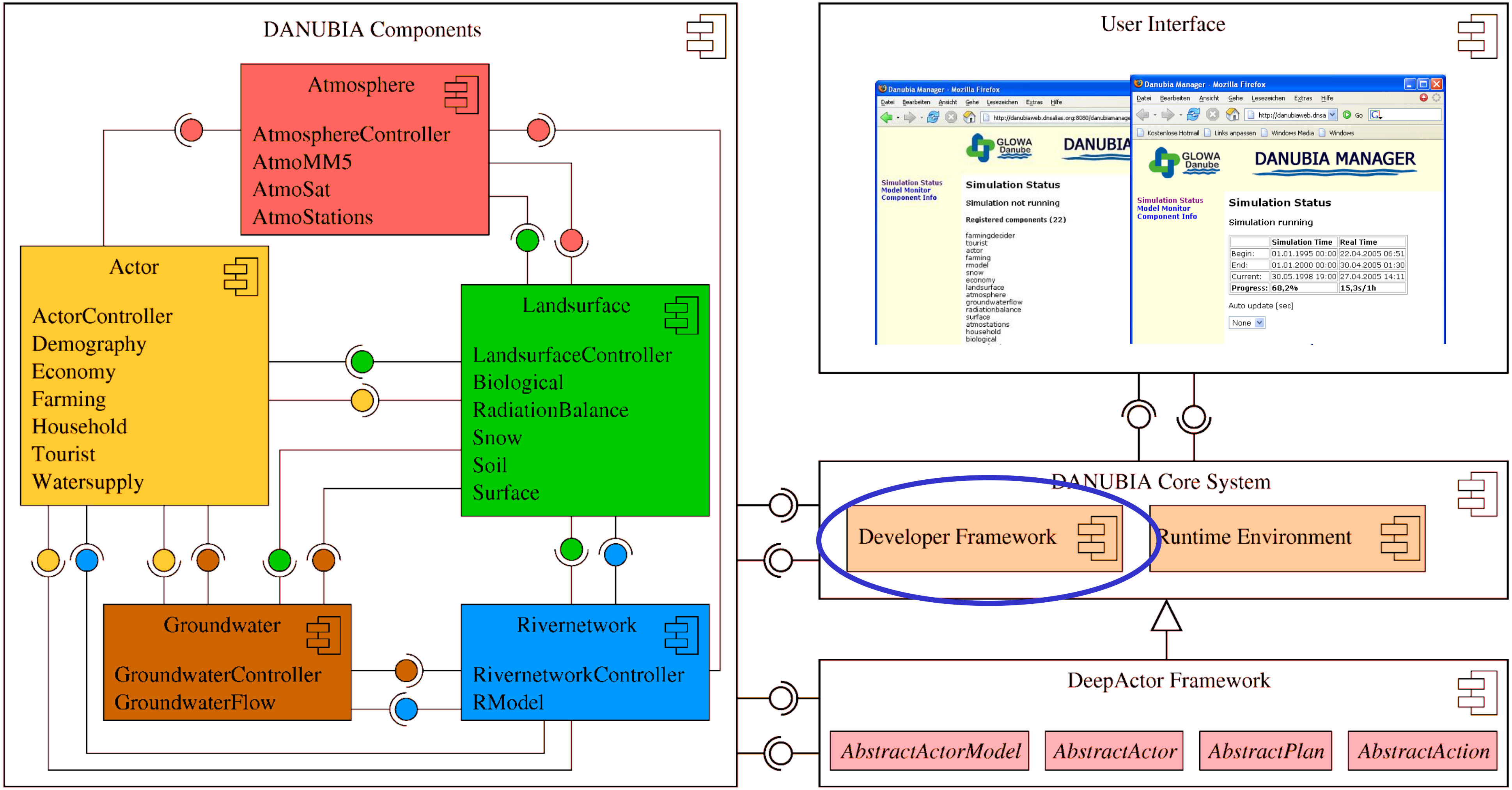
Aspect "Simulation Space"

Approach

- a simulation area consists of a set of "**proxels**" (process pixels, 1km x 1km)
- each proxel can be identified by a **unique proxel id** (pid) and is modeled as an **object** which has a "state"
- **computations** are performed "proxelwise"



System Architecture (revisited)

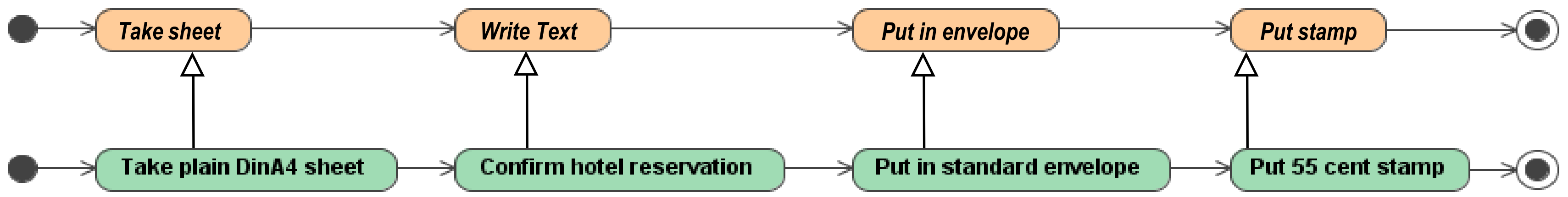


The Framework Idea

- Extract common properties and rules which hold ***for all*** simulation models and implement them in a general, abstract ***template***.
- The model developer must only implement the ***open pieces*** of the template (according to his/her domain).

Example (writing a letter)

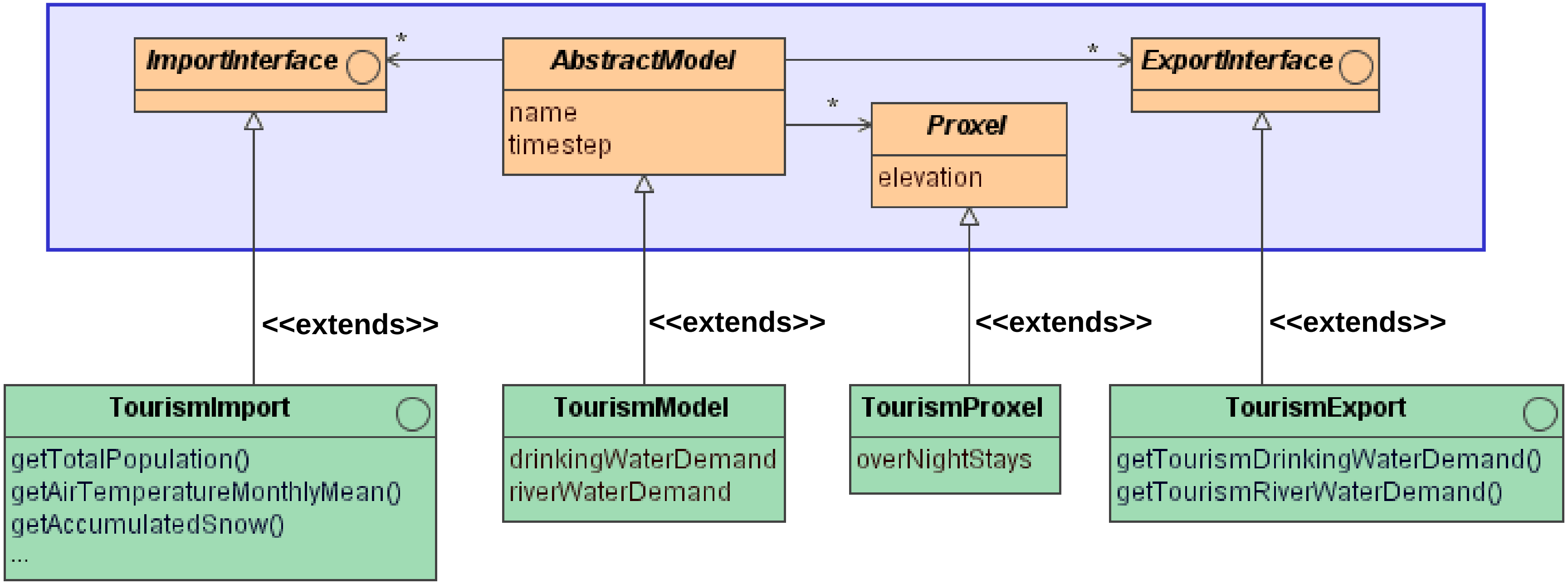
abstract template



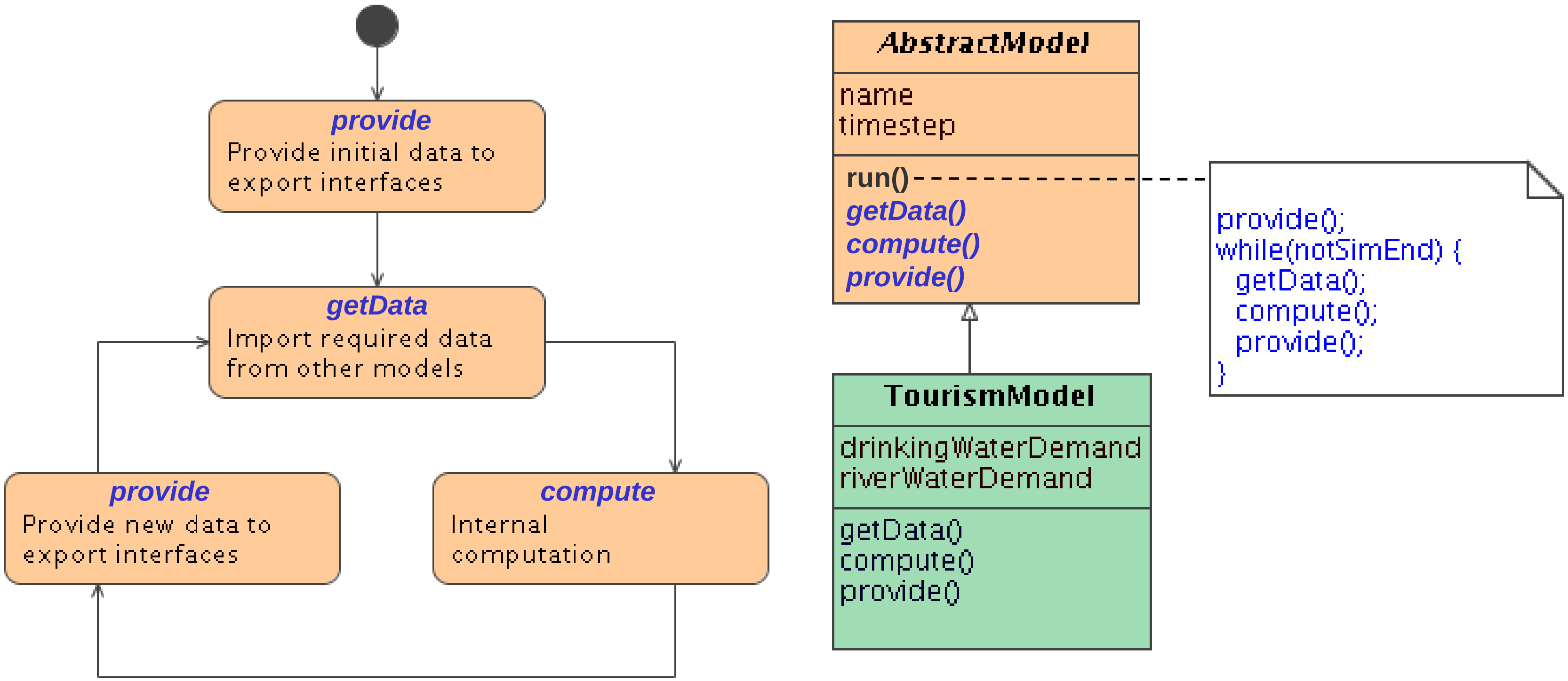
concrete instantiation

The Developer Framework

Common (static) properties of all simulation models



Aspect “Time”: Common Life Cycle of Simulation Models



The Coordination Problem

- All simulation models run in parallel and exchange date at run time.
- Each model participating in an integrative simulation has an ***individual local time step*** (e.g. 1 h, 1 day, 1 month).
- Every simulation model must be supplied with ***valid data***, i. e. with data that fits to the local model time of the importing simulation model.

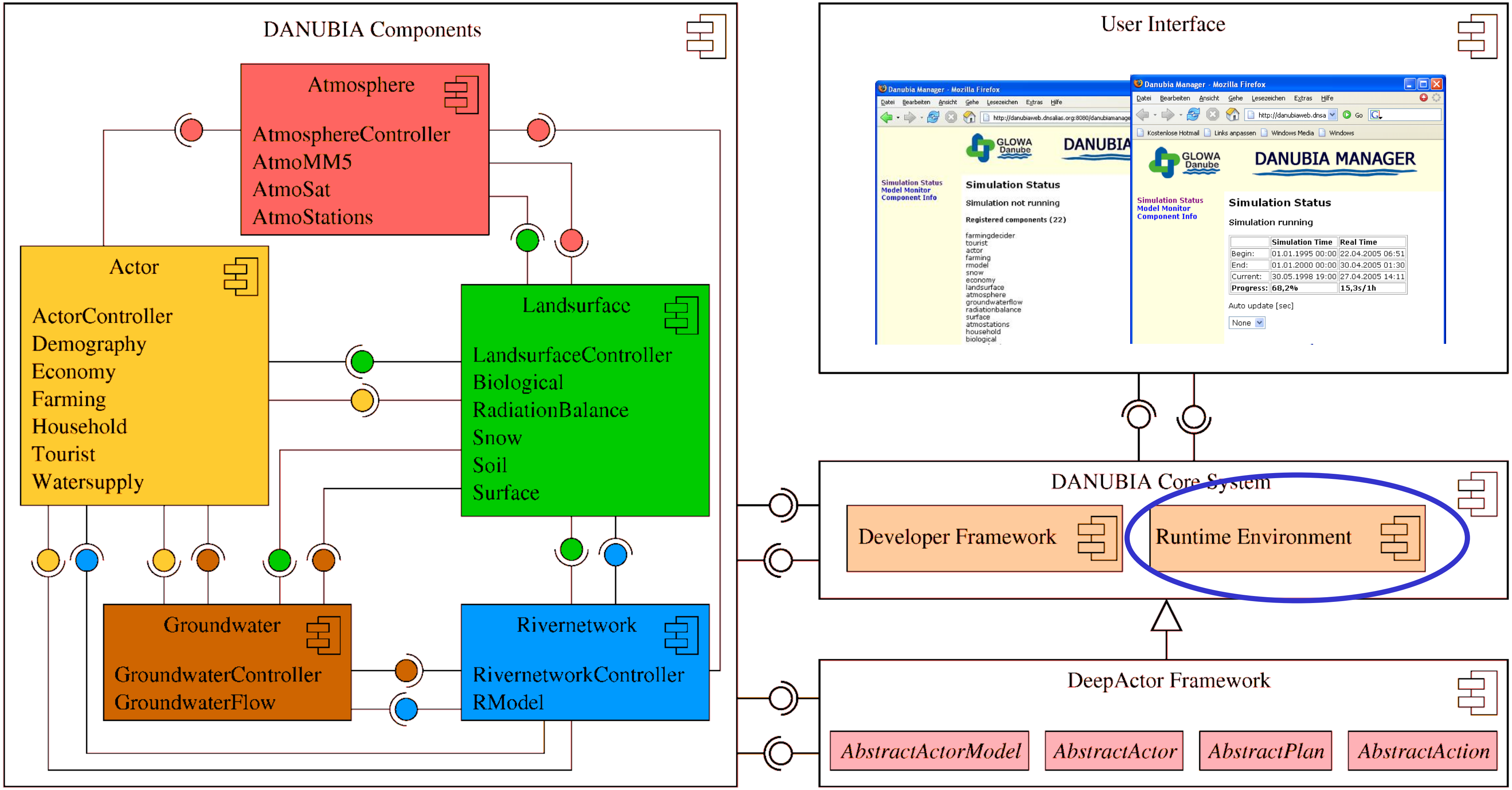
Process algebrac specification with FSP [Magee, Kramer]:

```
const simStart = 0
const simEnd = 6
range Time = simStart..simEnd

property VALIDDATA(User, StepUser, Prov, StepProv) = VD[simStart][simStart],

VD[nextGet:Time][nextProv:Time] =
  // no obsolete data
  (when (nextGet<nextProv)
    [User].get[nextGet] -> VD[nextGet+StepUser][nextProv]
  // no overwritten data
  |when (nextGet>=nextProv)
    [Prov].prov[nextProv] -> VD[nextGet][nextProv+StepProv]).
```

System Architecture (revisited)



Application

Scenario for climate change and/or society development

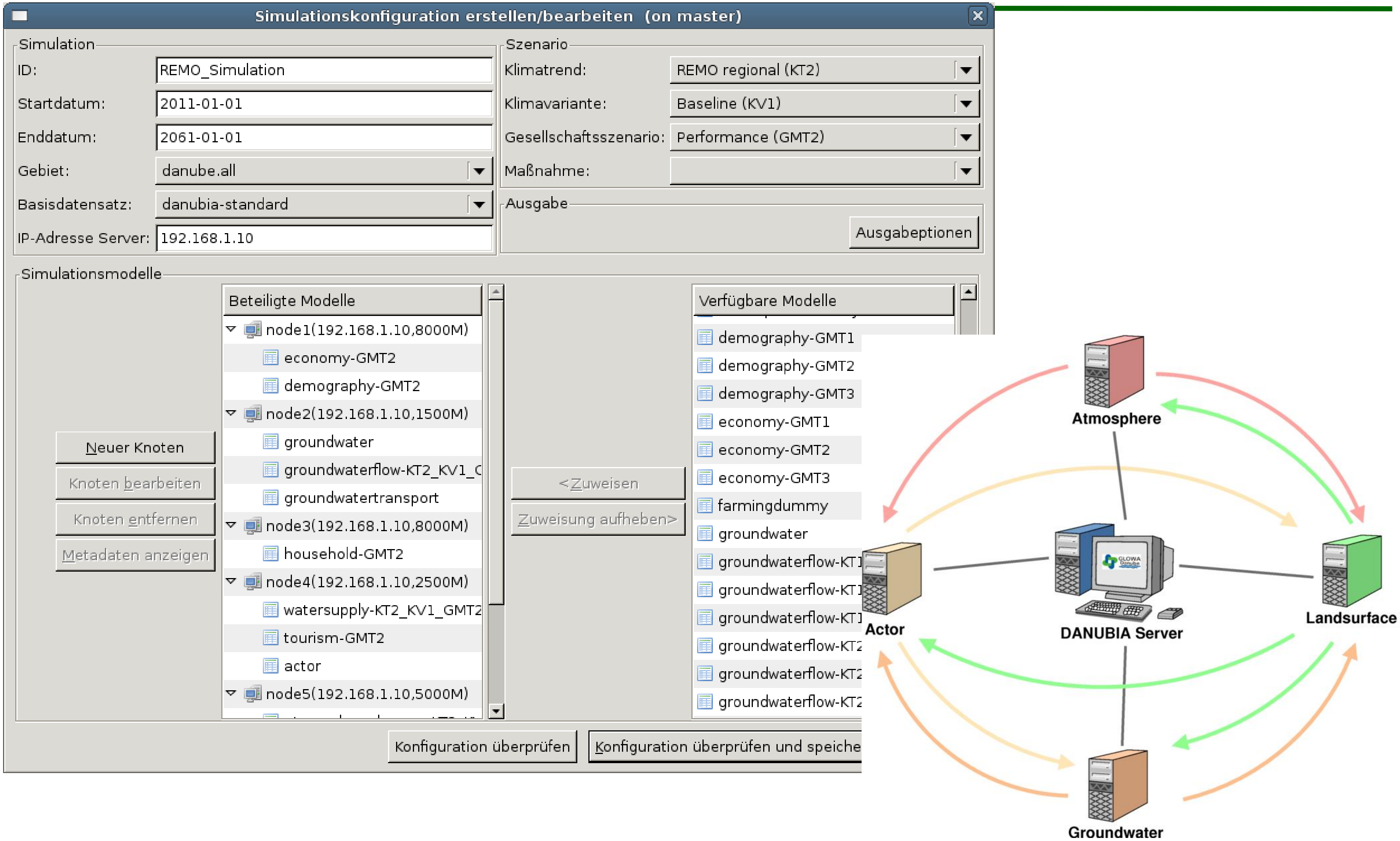
Integrative simulation

Result data, processing and analysis

Climate and Society Scenarios

Auswahl 1: Klimatrend	Auswahl 2: Klimavariante	Auswahl 3: Gesellschafts-szenario	Auswahl 4: Maßnahme
IPCC regional	*Baseline*[1]	*Baseline*	*Maßnahme 1*
REMO regional	*5 warme Winter*[1]	*Performance*	*Maßnahme 2*
MM5 regional	*5 heiße Sommer*[1]	*Allgemeinwohl*	*Maßnahme ...*
Fortschreibung	*5 trockene Jahre*[1]		
	REMO skaliert & biaskorrigiert[2]		
	MM5 skaliert & biaskorrigiert[2]		

Configuration of Integrative Simulations



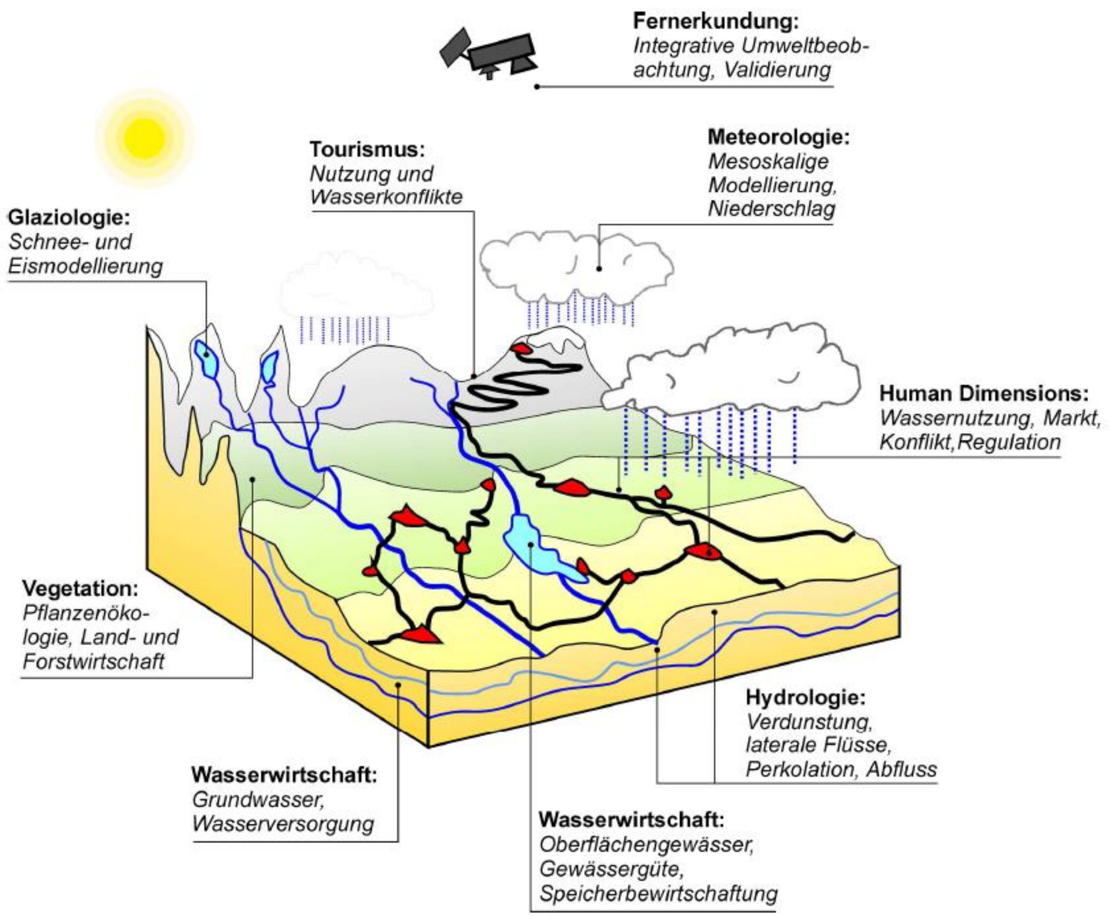
Results for the Upper Danube Basin (2011 – 2060)

- Used Climate Scenario (IPCC):
  temperature increase 3.3 C – 5.2 C between 1990 and 2090.
- Trends for precipitation:
  More rainfall in winter, less in summer, per year -3.5% to -16.4%
- Consequences:
  - Reduction of water power production
  - Possible restrictions for ship traffic in summer due to low water levels
  - 30 – 60 days less snow cover per year in lower alpine regions due to temperature increase but possible improvements in high-level alpine regions
  - Less winter tourism but moderate increase of summer tourism
- Further results
  - Less private water use expected (around 20%) due to changing behaviours and new technologies (for saving water)
  - No expected shortage of drinking water, but the need for temporary adaptation strategies of water suppliers is likely (e.g. more cooperation and networks)
  - (Almost) all glaciers in the Upper Danube catchment will vanish until 2045

Conclusion: Experiences on the Role of Informatics

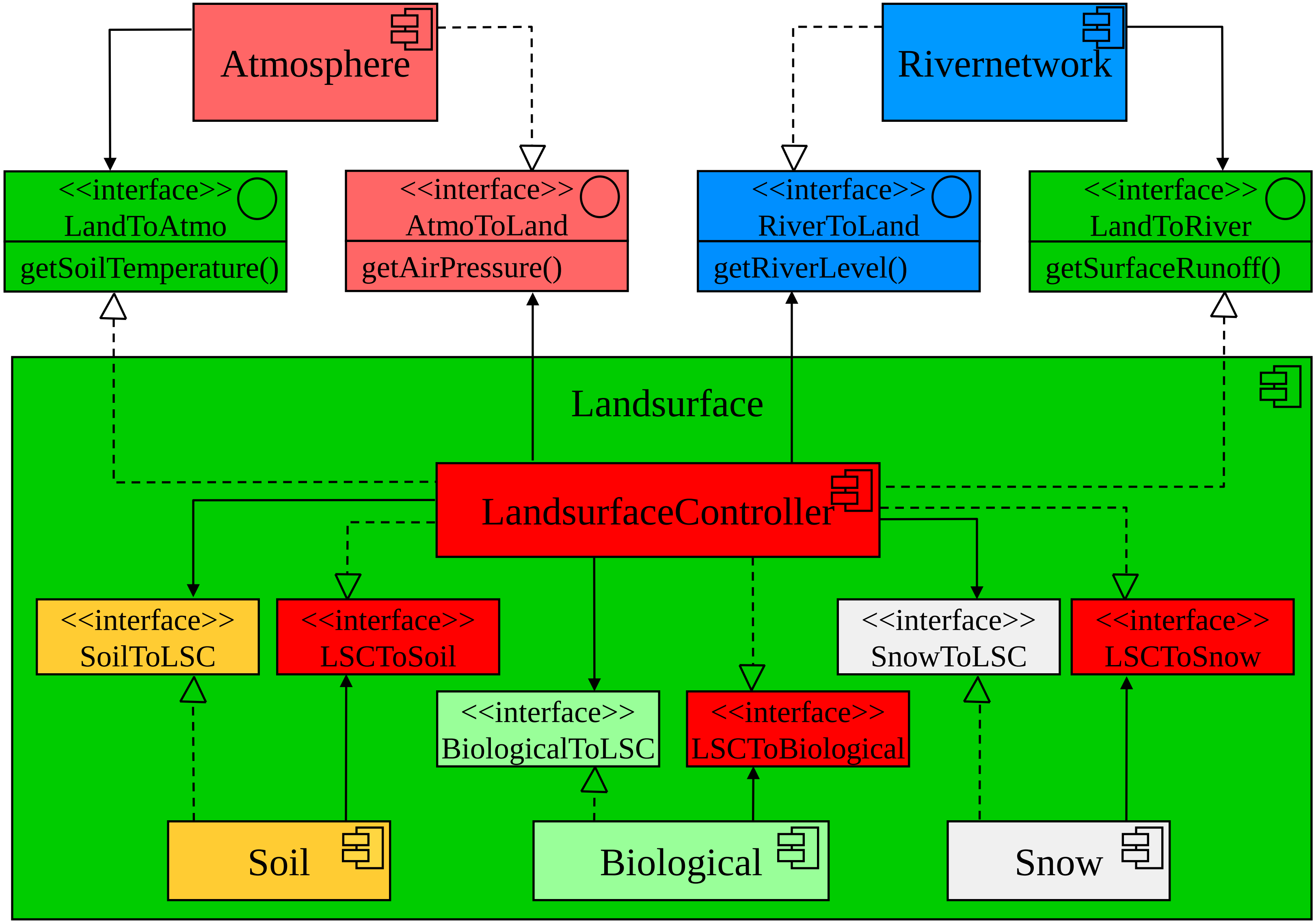
- Well-known methods of Informatics like **abstraction** and **separation of concerns** can be very useful for the **conceptual integration** in multy-disciplinary projects.
- As a tool for communication the use of a **common graphical modelling language** (UML) has been proven to be very valuable**:**
  - more precision in discussions between scientists of different disciplines,
  - common understanding of the integrative aspects
- Framework technology
  - **supports model developers** to integrate their simulation models into the overall system structure
  - implements **general rules** (templates) which support the reliability of the system
- With the help of **formal methods** the **correctness** of the temporal coordination (being the heart of the whole system) could be verified.

Mutually Dependent Processes in Nature and Society



- „Stand-alone“ modelling of the single processes is not sufficient
- Integrative view is needed

Hierarchical Structure



The Coordination Problem

- Each simulation model participating in an integrative simulation has an ***individual local time step*** (e.g. 1 h, 1 day, 1 month).
- Every simulation model must be supplied with ***valid data***, i. e. with data that fits to the local model time of the importing simulation model.

Example: M1 time step = 2, **M2** time step = 3

gets overwritten data!

M1 prov[t=0] get[t=0] comp prov[t=2] get[t=2]

M2 prov[t=0] get[t=0] comp prov[t=3] get[t=3]

gets obsolete data!

M1 prov[t=0] get[t=0] comp prov[t=2] get[t=2] comp prov[t=4] get[t=4]

M2 prov[t=0] get[t=0] comp

Formalisation of the Coordination Problem

Idea:

- Consider simulation models ***pairwise*** and only under ***one role*** at a time: either as a ***user*** or as a ***provider*** of data.
- A user must not get data “too early“, a provider must not deliver data “too early“.

Process algebrac specification with FSP [Magee, Kramer]:

```
const simStart = 0
const simEnd = 6
range Time = simStart..simEnd

property VALIDDATA(User, StepUser, Prov, StepProv) = VD[simStart][simStart],

VD[nextGet:Time][nextProv:Time] =
  // no obsolete data
  (when (nextGet<nextProv)
    [User].get[nextGet] -> VD[nextGet+StepUser][nextProv]
  // no overwritten data
  |when (nextGet>=nextProv)
    [Prov].prov[nextProv] -> VD[nextGet][nextProv+StepProv]).
```

Process MODEL

```
MODEL(step) = (start -> INIT),

INIT = (enterProv[simStart] -> prov[simStart] ->
        exitProv[simStart] -> M[simStart]),

M[t:Time] =
  if (t+step <= simEnd)
  then (enterGet[t] -> get[t] -> exitGet[t] ->
  compute[t] ->
  enterProv[t+step] -> prov[t+step] -> exitProv}[t+step] -> M[t+step])
  else  STOP.
```

Process TIMECONTROLLER

```
const nrModels = 2
range Models = 1..nrModels

TIMECONTROLLER(step1,step2) =
   (start -> TC[simStart][simStart][simStart][simStart]),

TC[nextGet1:Time][nextProv1:Time][nextGet2:Time][nextProv2:Time] =
    //enterGet
    (when (t<nextProv1 & t<nextProv2)
          [Models].enterGet[t:Time] ->
          TC[nextGet1][nextProv1][nextGet2][nextProv2]
    //exitGet
    |[1].exitGet[t:Time] -> TC[t+step1][nextProv1][nextGet2][nextProv2]
    |[2].exitGet[t:Time] -> TC[nextGet1][nextProv1][t+step2][nextProv2]
    //enterProv
    |when (nextGet1>=t & nextGet2>=t)
          [Models].enterProv[t:Time] ->
          TC[nextGet1][nextProv1][nextGet2][nextProv2]
    //exitProv
    |[1].exitProv[t:Time] -> TC[nextGet1][t+step1][nextGet2][nextProv2]
    |[2].exitProv[t:Time] -> TC[nextGet1][nextProv1][nextGet2][t+step2]).
```

Verification of the Design Model

Design model of an integrative simulation

||SIM-SYS = ([1]:MODEL(2) || [2]:MODEL(3) || TIMECONTROLLER(2,3)).

Check of the requirements

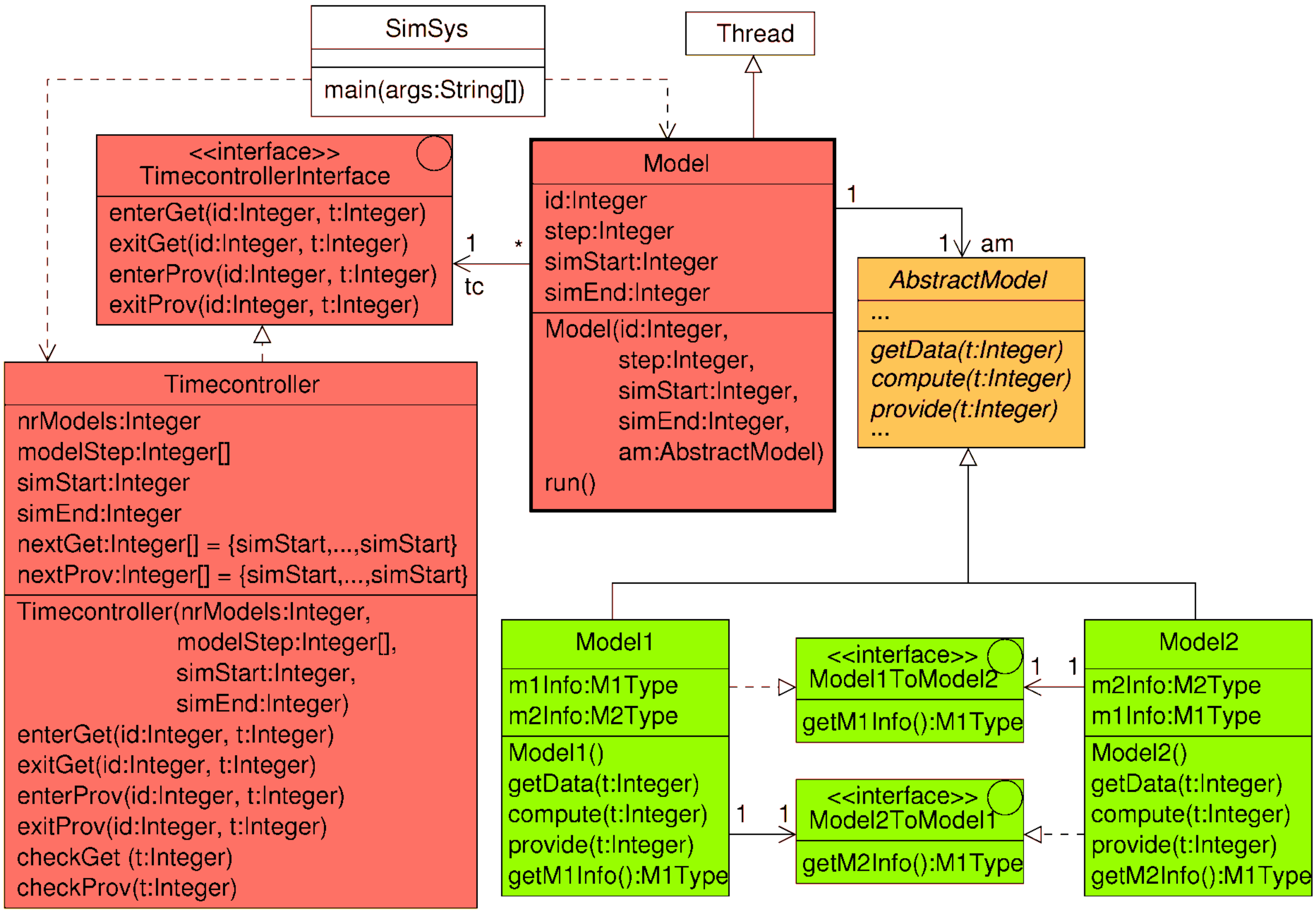
SIM-SYS |= VALIDDATA(User=**1**,StepUser=**2**,Prov=**2**,StepProv=**3**)

i.e. (SIM-SYS || VALIDDATA(User=**1**,StepUser=**2**,Prov=**2**,StepProv=**3**))

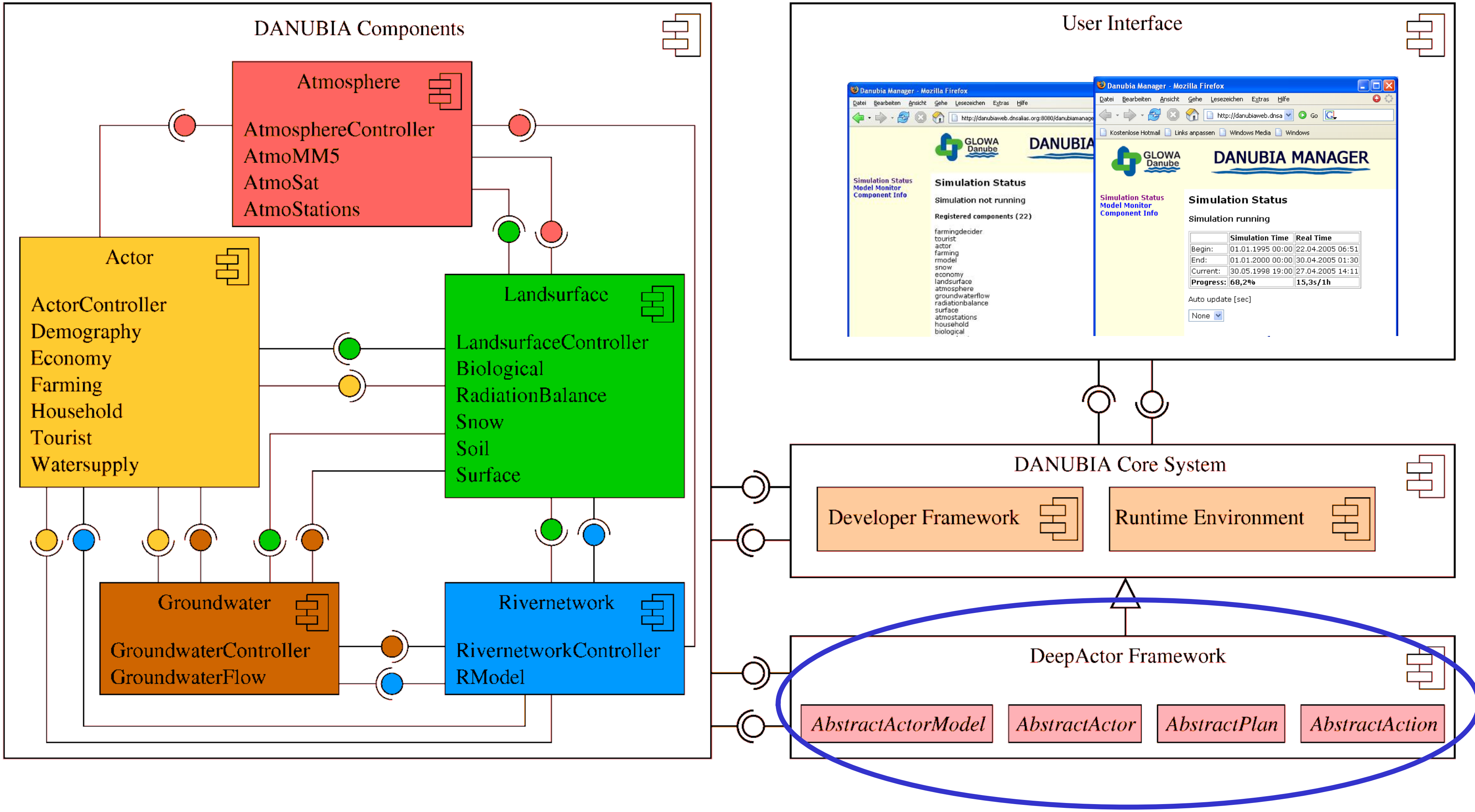
Error state not reachable!

SIM-SYS |= VALIDDATA(User=**2**,StepUser=**3**,Prov=**1**,StepProv=**2**)

The Coordination Framework



System Architecture (continued)

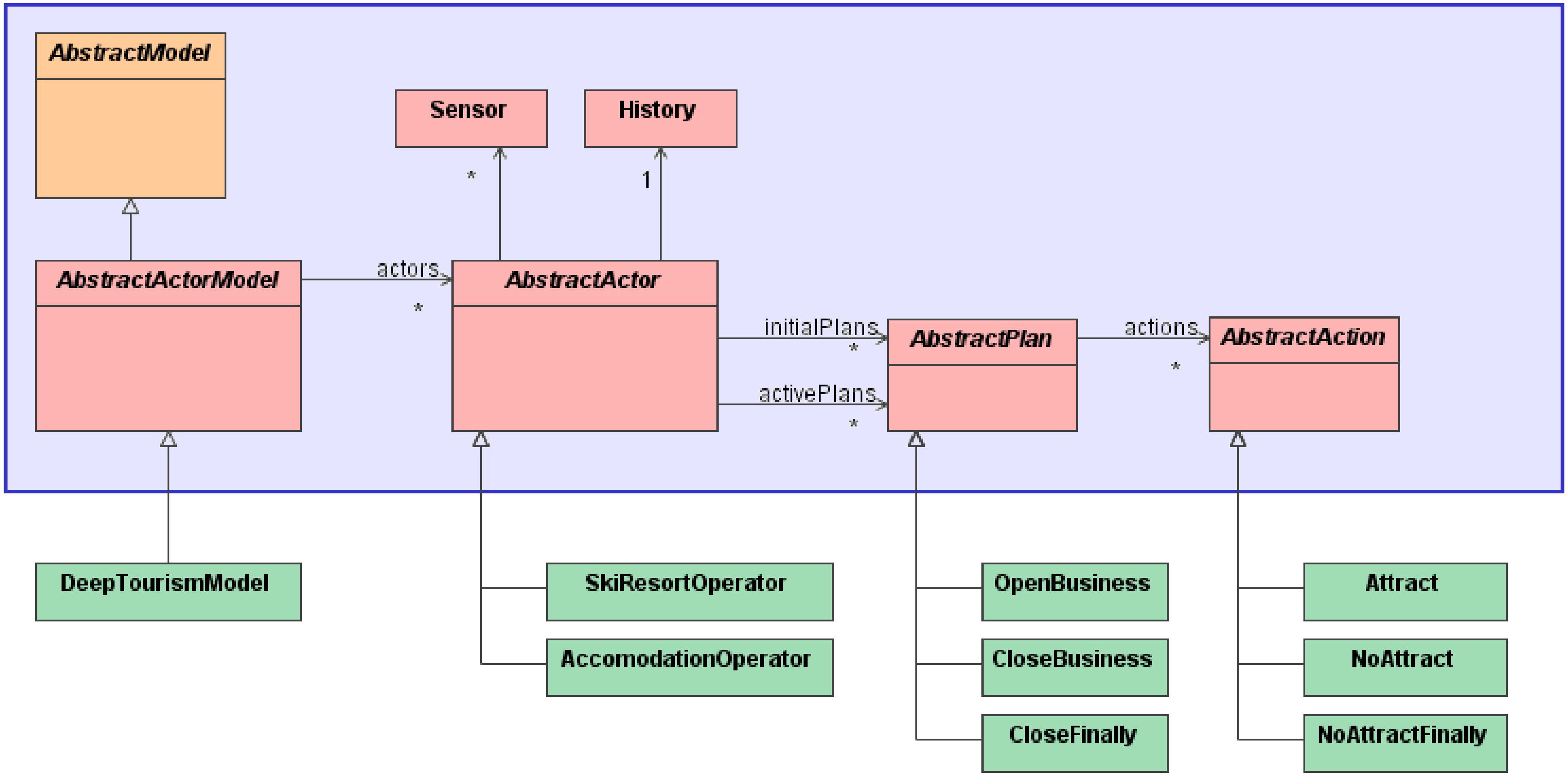


DeepActor Models: Common Properties

- Deep actor models integrate into their simulations decision-making entities, called ***actors*** (e.g. households, water-suppliers, farmers, tourists).
- Any actor has a repository of potential plans that the actor can execute (***initial plans***) .
- In each computation step an actor decides which of the initial plans should actually be executed (***active plans***).
- To support decisions each actor has
  - ***sensors*** through which he can observe the "environment" and
  - a ***history*** to remember previous decisions

The DeepActor Framework

Common structural properties of all DeepActor models



Common Behaviour of all DeepActor Models