CSDMS Users

- Who are they?
- How might they use the CSDMS?
- What design elements would be important to them?
- How could they contribute to CSDMS?

Researcher

- Find modules to use in model development.
- Test hypotheses to support data interpretation.
- Pre- and post-processing visualization tools.
- Contribute code to CSDMS.

Planner, Consultant

- Run scenarios on a pre-packaged CSDMS model.
- Use GIS interface tools to generate model input and relate model output to environmental factors, land use.
- Quantify uncertainties in model output.

Educator

- Illustrate surface processes using results from pre-packaged models.
- Build intuition with “what-if”-type model runs.
- Develop case studies that integrate field data and model simulations.
- Prepare exploratory exercises for students

Key Design Elements

- Community built, freely available
- Suite of integrated modules
- Range of time and space scales
- Algorithms for sediment/solute transport and deposition in a complete suite of earth environments
- Tools for input, output, visualization, data management
- Stable, responsive infrastructure

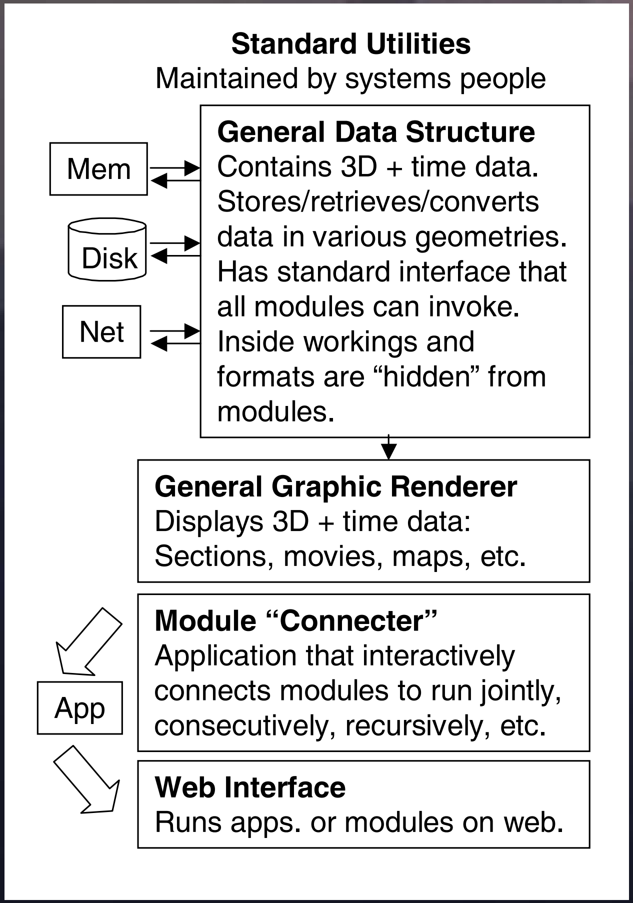
CSDMS System Components

- User-friendly graphical interface
- Interchangeable community-contributed process modules
- I/O and visualization tools
- Linkers and interfaces to transfer data among different modules
- Protocols for linking modules
- Grid generators
- Equation solvers

System Components, cont'd.

- Tools for developing dynamically adapting grids
- Tools for unconventional mathematics (e.g. cellular automata)
- Tools for model nesting and interactions across scales
- Protocols and techniques for linking modules or domains with different solution techniques

Tentative Software Architecture

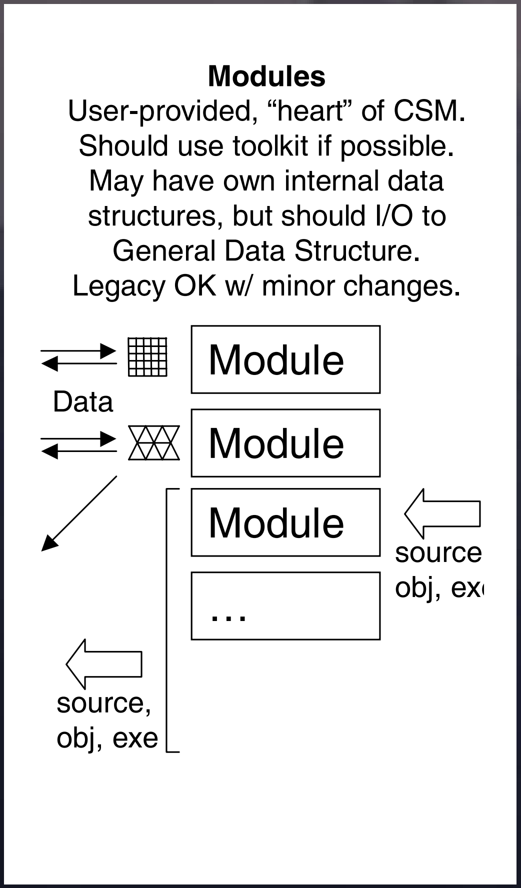


Component 1

Standard Utilities:

- maintains and stores all data and variable arrays
- GIS and data handling tools
- module connector

Tentative Software Architecture

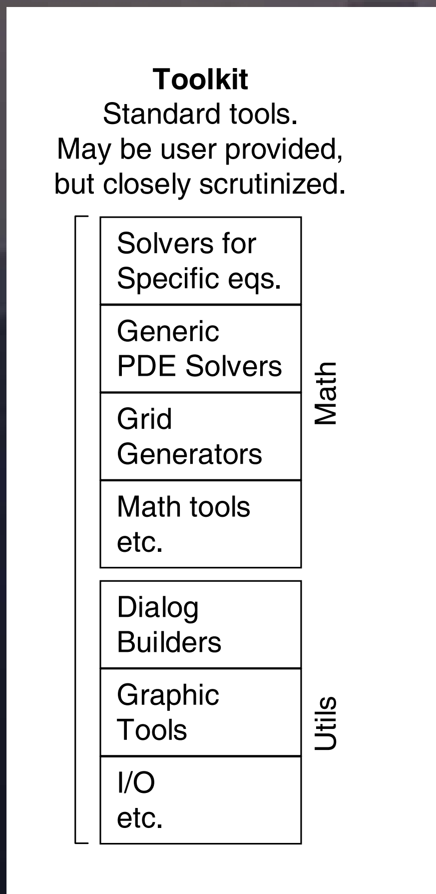


Component 2

Module Component:

- programs for specific processes built around conservation equations, including mass conservation
- incorporates biological, physical and chemical effects

Tentative Software Architecture

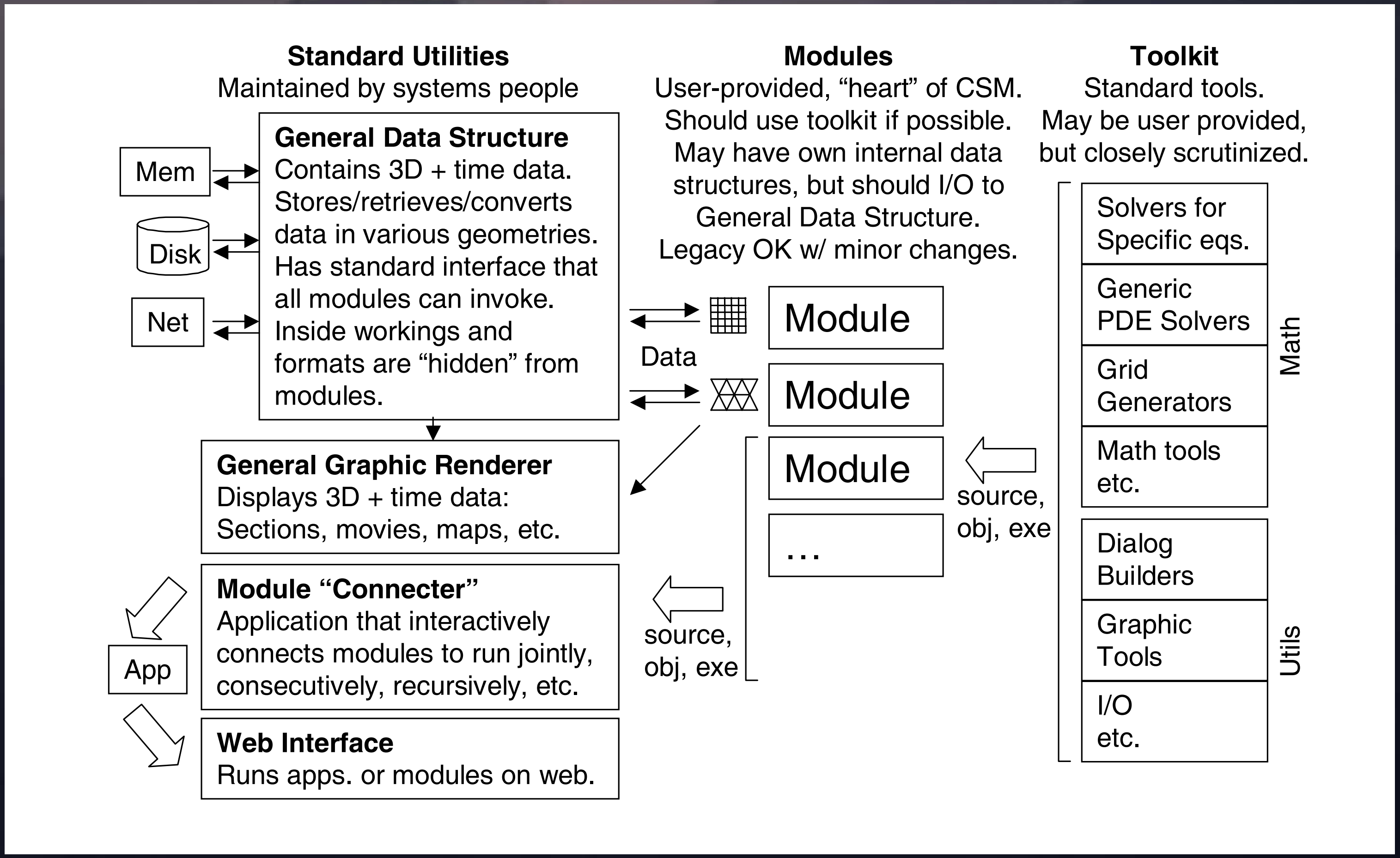


Component 3

Toolkit:

- grid generators
- equation solvers
- distributed processing

Linked components



System requirements

- Modeling system should be able to incorporate novel computational strategies (cellular automata, particles, agents).

System requirements

- Modeling system should be able to incorporate novel computational strategies (cellular automata, particles, agents).
- System must incorporate moving boundaries and allow the modeled geomorphology to evolve.

Moving boundaries

(after MARGINS S2S Science Plan)

System requirements

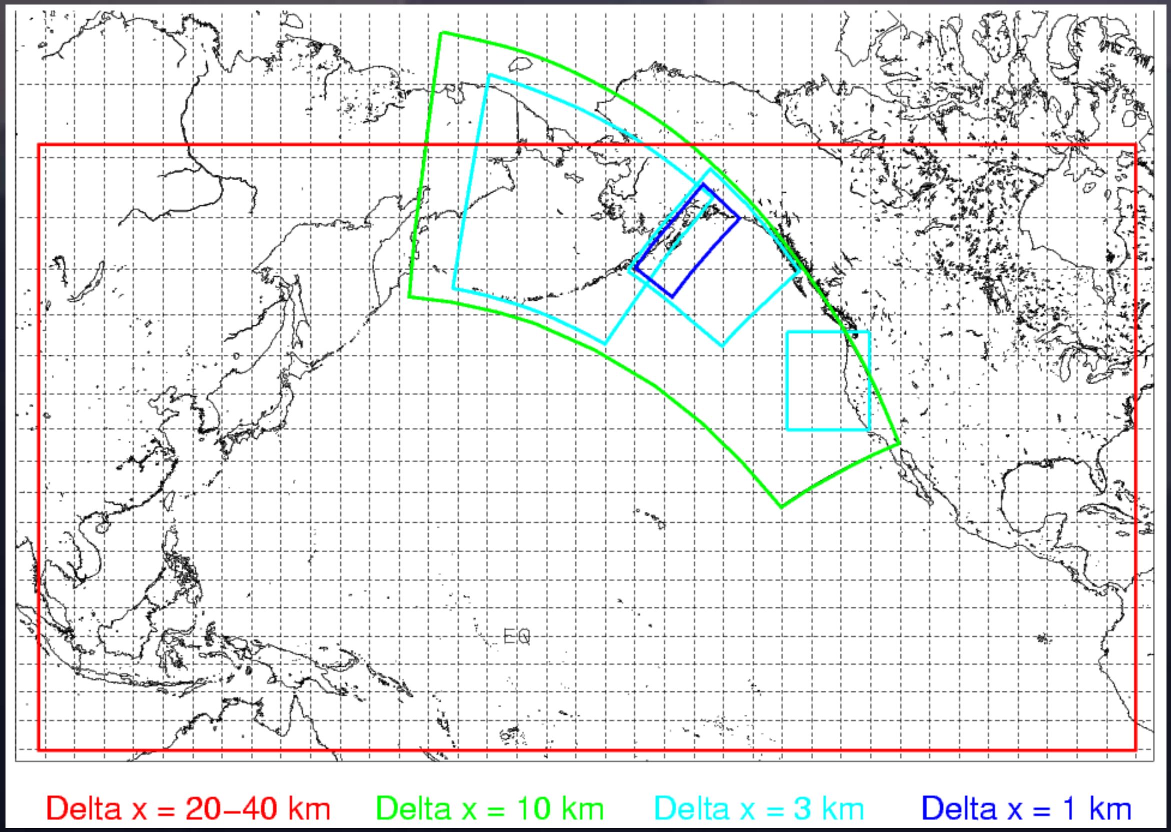
- Modeling system should be able to incorporate novel computational strategies (cellular automata, particles, agents).
- System must incorporate moving boundaries and allow the modeled geomorphology to evolve.
- System must be able to accommodate distributed “source terms.”

System requirements

- Modeling system should be able to incorporate novel computational strategies (cellular automata, particles, agents).
- System must incorporate moving boundaries and allow the modeled geomorphology to evolve.
- System must be able to accommodate distributed “source terms.”
- Modules can be nested in time or space scales.

Example of a nested oceanographic model

The Coastal Gulf of Alaska Circulation Model (3-km grid), part of the University of Alaska's *S*ea-*A*ir-*L*and *M*odeling and *O*bserving *N*etwork



The large scale north Pacfic model (red) supplies boundary conditions to the smaller scale northeast Pacific domain (green), which then is run to supply boundary conditions to each sub-domain.

Managing community input

Problem:

How to allow input from a diverse community while maintaining standards for compatibility and prediction quality?

Possible solution:

Hierarchy of module categories ranging from “proposed but untested” to “fully tested and recommended for routine application.”

Data structures

- must be designed so that model components can communicate with each other
- must have the flexibility to evolve
- could vary during model run
- links between the data structure and modules will require (complex?) interpolation methods