

Scoping Workshop on
Community Hydrologic Modeling Platform

**Why We Need To Rethink How
We Do
Mathematical Modeling**

Peter K. Kitanidis
Stanford University

Outline

- Mathematical Modeling;
- Why A Community;
- What Do We Expect From Models;
- Role Models Of Communities.

Ingredients of Modeling

A mathematical model involves

- Science (e.g., *processes* such as Navier-Stokes or Advection-Dispersion, or *inference* such as Bayesian);
- Mathematical Methods and Algorithms (e.g., conjugate gradient or geometric multigrid solver, data filtering);
- Code.

Scale of Modeling

- Small-scale (one person can program everything from scratch);
- Intermediate-scales (research group);
- Large-scale (community).

Why a community

- We need multiple types of expertise and technologies;
- We must have new tools added, reflecting new methods and the evolution of old methods;
- We must share in the debugging, evaluation, maintenance, distribution, and support of software.

Challenges In Using Programs Developed By Others ...

Or

What Do We Expect From Programs?

- | | |
|--|--------|
| <ul style="list-style-type: none">• Integrity;• Generality;• Efficiency;• Readability and Simplicity; | Tier 1 |
| <ul style="list-style-type: none">• Modularity;• Encapsulation;• Reusability. | Tier 2 |

What is expected from computer programs

- **Integrity** The program does what it is supposed to do in a sufficiently accurate way. This is affirmed through careful debugging and testing.

- **Generality** What is the actual scope of applications of this program? What are the limitations? Ideally, a program should be applicable to as many cases as reasonably possible; few really are.

- **Efficiency** The program must achieve its goals with minimum use of resources: computer memory and CPU time. However, many programs were developed for small jobs, where efficiency does not matter.

- **Readability** The program must be written in a clear, precise, and logical fashion. It must also contain comments explaining what is done.
- **Simplicity** The simpler the program, the easier it is to read, modify, and debug.

- **Modularity** Any major enterprise can be broken into sub-tasks. It is good style to implement each task in a separate unit (subroutine or function). Modularity makes for programs that are shorter, easier to read, and easier to debug. Modularity is an essential element of reusability.

- **Encapsulation** A unit consists of the interface (input, output, and help) and procedures. A user should be able to use the file without need to open the file.

- **Reusability** A unit performing specific tasks may be used in different programs, with many objectives.

What Do We Need?

- At the local scale enforce the tier 1 specifications (Integrity, Generality, Efficiency, Readability, and Simplicity).
- At the community scale enforce the tier 2 specifications (Modularity, Encapsulation, and Reusability) by setting up standards and classes.

What Type of Community?

MODFLOW Community?

- Successful in meeting modest objectives;
- Limited in scope and expandability.

International Groundwater Modeling Center

- Repository of models;
- Technology transfer;
- But no leadership in developing new standards or platforms.

The National Center for Atmospheric Research

- Plans, organizes, and conducts atmospheric and related research programs in collaboration with universities.
- Facilitates the distribution of models and does technology transfer;
- Leadership in developing new standards or platforms?

MATLAB (and similar products)

- Collection of tools;
- Combination of tools can be organized into toolboxes;
- Tremendous flexibility and expandability;
- Issues of support, control, and cost...

COMSOL Multiphysics (and similar products)

- Center point is finite-element solution of PDEs (extremely flexible);
- Tremendous productivity tool;
- Valuable teaching tool;
- Empowers the modeler;
- But is not efficient in solving one specific problem.

Concluding ...

- We should pick the best features from many prototypes – avoid mistakes.
- Ideally, develop a *flexible* environment where modelers can add their tools, without losing sight of specific objectives.
- Maximum modularity, encapsulation, and reusability are possible only in a object-oriented programming organization.