Common Modeling Environment (CME): A Framework for Integrated Decision Support Systems[1]

Jerry Stuth[2], Dan Schmitt[3], Kristen Zander[4],Clint Heath[5], Jay Angerer[6], Jeff Vitale[7]

ABSTRACT

Assessing impact of technologies and policies requires the use of a suite of decision support models that address the complexity of economic systems at the sector level, farm level economics and human welfare, crop production, grazing land production, livestock performance and resulting environmental processes. Typically "integration" of these processes involved manual transfer of data files between applications or limited digital integration in a subset of modules. Further, there was limited ability to modify models in a manner that allowed tighter "digital" integration. There is growing need within SANREM and with other partners, including FAO, to package a number of different research simulations together to develop a more detailed and holistic view of non-homogeneous activities and environments. This need to package integrated suites of models so that they can be run on a single computer or internet/intranet, led to the pursuit of the Common Modeling Environment (CME) concept. CME is an evolving set of information technology that is modular and designed to grow in sophistication as needs are identified within organizations. This system brings cross-platform delivery, a scalable distributed computing model, and shared common input data to many research models with minimal model modification. A model server process can be run on any platform that has a *JAVA* virtual machine installed and quickly allows incorporation of stand alone models without undue stress on research model developers.

INTRODUCTION

A primary problem facing policy analysts in today's decision environment is the need to use a wide variety of economical, sociological, biophysical and environmental models in a coordinated manner that allows a rational sequence of analyses to represent the complexity facing the decision maker. Models represent a long-term investment by their developers to properly represent complex relationships in an evolutionary process of verification, upgrade knowledge and validation cycling as new technologies allow enhancements throughout the development profile. Model developers are reluctant to change their finely tuned models to accommodate a broader set of analysis; thereby, creating problems of data sharing and transfer between model input/output. This has given rise to loose coupling of models to avoid inter-dependency between models.

Author Contact: Jerry Stuth, Email: j-stuth@tamu.edu

[1]Paper presented to the SANREM Research Scientific Synthesis Conference, November 28-30, 2001, Athens, GA

[2]Texas A&M University, [3]Center for Natural Research Information Technology, [4]Texas A&M University, [5]Center for Natural Research Information Technology, [6]Texas A&M University, [7]Texas A&M University

Over the past 10 years a large number of decision support tools have been designed to address a wide array of economic, biophysical, spatial, demographic, social and natural resource issues affecting sustainable development of societies throughout the world. These computer-based tools are designed with a specific set of data needs, targeted toward a specific user group using a unique set of computer development tools and database environments.

Many of the analytical environments associated with such issues as food security, sustainable development, climate change, desertification and biodiversity require a suite of tools to be used to address emerging issues given the complexity of problems associated with these issues (Meehl et al. 2000). Typically, a minimum set of analyses involves some form of economic analysis supported with one or more biophysical models. These suites of models are generally referred to as "loosely coupled" (Yourdon and Constantine 1979) in that data from one model is moved in digital form after the output had been modified to fit data structures of the receiving model, typically moving such things as crop yields to economic farm models with the acreage determined with GIS tools such as ARCVIEW. As you increase "coupling" of models in a digital sense the complexity of parameterization increases but strength of system feedback also increases.

Clearly, coupling or integration of models to address a broad range of development issues must keep a clear focus on such issues as availability of data to support the use of the models, skills levels of the targeted users, training needed to effectively use the models, institutional commitment to use of the analytical systems and the sustainability of personnel to use those models. Often the call is made for much more simple models with low data requirements, resulting in low level coupling and requiring special care to capture the right data output to feed input of other models needed to address a more complex development issue. However, most of the more difficult issues in development are complex, requiring complex analytical tools to explore possible interventions, technologies or policies (Britton 2001).

Impact assessment (IA) of new technologies and policies that address such issues as food security and natural resource management have a strong spatial component, a representation of multiple resources of land use (cropland, grazing land, livestock) in biophysical models and multi-scale economic analyses addressing farm, community, national and regional issues. The impact assessment group (IAG) has been developing methodology under the umbrella of a global decision support system for impact assessment in the SANREM CRSP over the past three years. The IAG suite of tools currently uses three spatial tools (ACT 3.0, ARCVIEW, ERDAS), a statistical package (SPSS), several biophysical models (SWAN, PHYGROW, NUTBAL PRO, SWAT), a series of small utilities (WXGEN, LANDDEMAND, CURVE EXPERT, etc) and several economic models (ASM, FLAM, FLIPSIM).

Clearly, this suite of tools is not designed for an inexperienced person to use, even with the appropriate discipline training such as agricultural economics, agronomy, animal science or rangeland management. The question arises: "how do we capture the analysis and package that analysis in a manner that is usable by those technical staff that support analysis by decision makers that effectuate policy or choices of technology funding"?

In order to address this issue, a concept referred to as the "Common Modeling Environment" or CME was pursued by IAG. The goal of CME is to provide an information technology development environment that can accommodate a wide variety of models in a manner where the developer of the model defines the level of interaction with their modeling environment without any changes in the design of the application. CME also is designed to allow the model developer to define those variables a user can change and rerun the model. The goal of the CME is to develop a middleware language (Britton 2001) that allows the packaging of a suite of pre-parameterized models that reflect the legacy analysis of a given impact assessment project and defines those selected variables that an inexperienced person could "tweak" to explore additional analyses without additional training or capacity building within a targeted organization. To address this concern, a web-based common modeling environment is being developed to allow a variety of models to be presented for use in a cohesive manner.

This ambitious objective has been pursued for the past few years of SANREM to a point where a new generation of tools can be made available to modelers interested in delivery of their analysis in a more controlled analytical environment. The CME concept is evolving with new features constantly added to the middleware language to expand features of the system. The evolving middleware language in CME is targeted for use by the model developer while the interface, once defined by the modeling team, is targeted to the primary user group of the suite of tools.

This paper focuses on the technical issues underlying CME and provides a framework for system that can be used to set up models for use by inexperienced users. Initial funding for CME was provided to the Center for Natural Resource Information Technology via the Texas Agricultural Experiment Station. However, recently the FAO World Agricultural Information Center (WAICENT) has also provided funds along with the SANREM CRSP to design inputs to the CME concept to help create a mechanism to deliver a wide variety of analytical tools to their partners throughout the developing world.

METHODOLOGIES

The CME concept originally grew from the recurring situation of needing to package a number of different research simulations together to develop a more detailed and realistic view of non-homogeneous activities and environments. The CME structure is comprised of a browser enabled interface, CME client, CME server, model factory, resident databases, and remote databases/models/GIS tools (figure 1).

Browser Enabled Interface

One initial concern of CME is to develop an interface that is intuitive to use by non-modelers. To address this concern, a web-based common modeling environment is being developed to allow a variety of models to be presented for use in a cohesive manner. A common user interface for model configuration and results analysis decreases the model configuration learning time and makes cross model comparisons and aggregations quick and easy. This system brings cross platform delivery, a scalable distributed computing model, and shared common input data to many research models with minimal model modification.

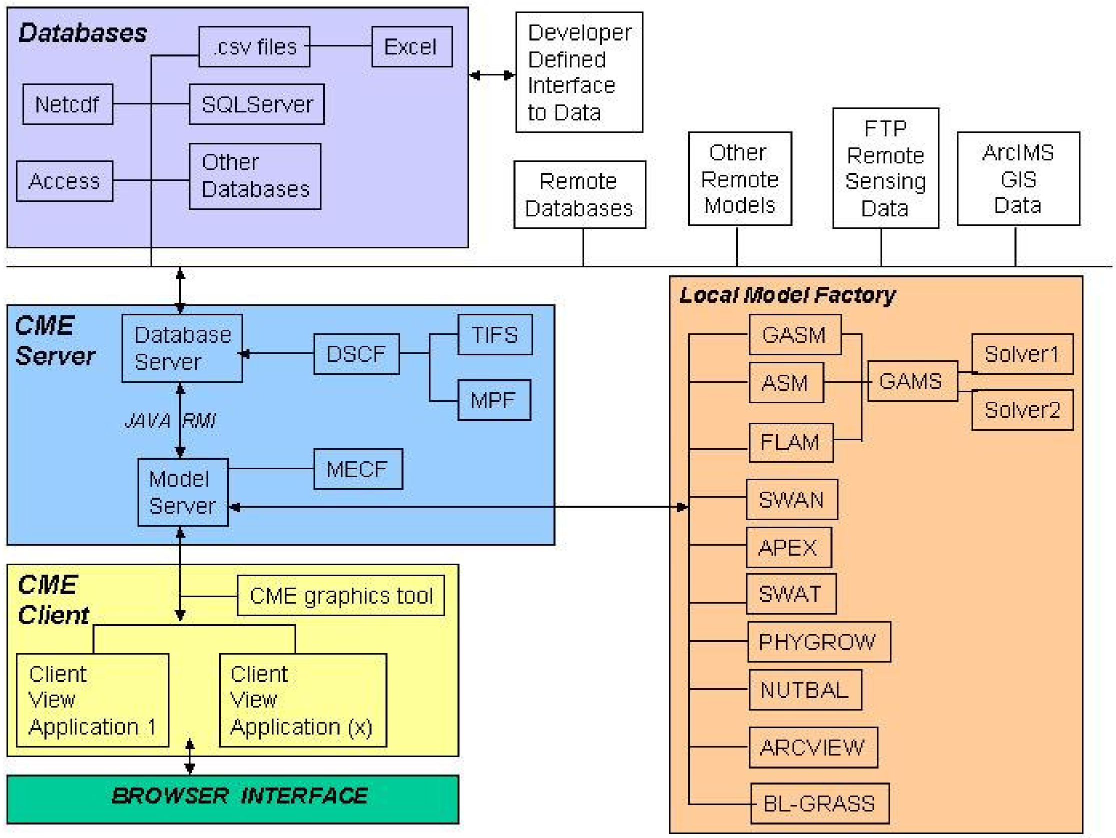

Figure 1. Integrated structure of the Common Modeling Environment (CME).

CME Client

The model client program is a java application, which can also be run on many existing machines and provide rapid local response for interface interactions, as well as access to common data, and access to the remote server for the most up to date version of the simulation model on the most up to date hardware the model maintainers can make available

Graphics Module: This module is comprised of the CME graphics applications and middleware language that defines how the data is presented. We have added the ability to save graphs (including underlying data) from a model run into XML so that they can be emailed to other users of the CME for their viewing and continued analysis. Previously, the only way CME users could share graphs was by converting the graph into a JPG image with screen capture software or by saving it as a postscript file. Previously users could also export the raw data for import into a tool like Microsoft Excel, but none of these options allow a CME user to import that graph back into CME and manipulate it. As part of this feature, the menus were altered to provide a more user-friendly layout and to allow for Graph Save/Import features. Also, the software was altered so that if a user has graphs up when they close down the application, those graphs are automatically restored when the application is restarted.

The CME was designed to allow users to run a suite of models located on a server or multiple servers. This is accomplished by using the Java programming language. Java Database Connectivity (JDBC) is used to connect the client to the database, and Remote Method Invocation (RMI) is used to connect the client to the servers. The CME is platform independent and creates a client-server environment for model runs; thereby freeing up CPU time and memory on the user's machine that was previously used by the model, and moving data exploration tasks to the client computer.

Client View Module: The Client Viewer defines how the model is viewed by the user. Model developers define those variables that are displayed on the screen and the range of values that can be entered in to the selected model. These inputs can vary from a few variables to a complete data entry system. The most common interface is likely to be models that have already been parameterized, tested and run with prior analyses. Users would then just change a few variables of interest, run the model and observe the output.

CME Server

Model Server **:** The server application reads its configuration from a user defined "Model Execution Configuration File" or MECF. This file is created by the modeler and tells the CME server application exactly how to access model-specific input data, what variables will be configurable on-the-fly, and what output to generate. A model server parser and model input/output SQL parser have been developed to allow model developers to keep their model designs intact and linked to import/export specification tables. This allows data entry, data output graphing and data exchange over the web. The CME software is all written in Java, which allows the client and server applications to be run on separate machines and communicate via Java Remote Method Invocation (RMI). After the model server program has been downloaded and configured properly to run on a machine, the modeler must create a Model Execution Configuration File (MECF) for the models running on the server. Only one MECF is required per server. This file tells the server process how and where to run a model, where to find the Template Input File Syntax (TIFS) and which output files will be created for users to view. An explanation of the MECF format and TIFS format can be found in the next two sections.

Database Server: The CME server application has the ability to connect to ODBC compliant database via the Java Database Connectivity (JDBC) interface. If the modeler has the input data in a SQL, Access, Oracle, or other database, they can create a CME Database Server Configuration File (DSCF) that tells the CME application exactly where that database resides and how to connect to it. The modeler also must create a Template Input File Syntax (TIFS) file that tells the database server exactly what kind of tables and fields are in the database so that it can form proper queries into the database.

Model Factory

Existing models are encapsulated in a model server process. This model server is configured to know about the required input files, export the results of interest and how to run the model. This allows the incorporation of existing stand alone models, as well as quickly assimilating new models into the suite of tools without undue stress on research model developers. The model server process can be run on any platform that has a java virtual machine. This means existing models on Sun, NT, Windows 95, and Dec Alpha machines can be incorporated into the modeling environment. Model server processes can be distributed across more than one machine to scale up model server performance with demand.

If a modeler would like to add a model to the CME, they would first need to set up a server that is accessible from the web. The server should have the Java Development Kit (JDK) or the Java Runtime Environment (JRE) installed in order to run the model server processes required. The model server process is distributed as a JAR (Java Archive) file, named modelserve.jar. It is available for download from http://cnrit.tamu.edu/CME/downloads. After installing the JDK or Java Runtime on the server machine, the CLASSPATH environment variable should be set to point to the full path where the modelserve.jar file resides, ie: \home\model\modelserve.jar

The current release of the Common Modeling Environment has been improved in terms of consistent server status and restarts with no loss of completed model results. An example of this server and client have been bundled together to produce a standalone PC version of the Phygrow simulation model used in the impact assessment process on rangelands.

To use the CME to run a model, a user must first download a Java Development Kit (JDK) or Java Runtime Environment (JRE) (there is one available from the java.sun.com website.) The CME is a java application and requires a java runtime environment. After a user has installed the JDK or JRE on their machine, the next step is to download the CME client12.jar file from the downloads page. The JAR file is the only file needed for the CME to run on a user's machine. The downloads page contains step by step instructions on how to set up your machine to run the CME. After this completing to procedure outlined, the CME window should appear on the client machine and be ready for model execution. When the CME is run on a client machine, it will appear as a window with a series of action tabs.

RESULTS AND DISCUSSIONS

The documentation and software for the CME can be acquired at http://cnrit.tamu.edu/CME. The site allows viewing of a test that was conducted by interfacing a rangeland model (PHYGROW) and a crop model (EPIC). Currently, the PHYGROW models runs exclusively under the CME environment allowing full parameterization.

The ability to save all the results of a model run into a single compressed file on the CME user's local hard drive was added after observing user activities associated with the beta test of PHYGROW and EPIC. This provides the ability of users to share entire model runs with each other or for a user to be able to take a model run on a laptop while on travel where there may not be an Internet connection. It also allows users of the CME to share model results with each other where one user can make changes and send back to the other. Currently, the CME allows users to save and exchange a link to an existing model running on a particular model server, but this new feature allows for the results of the model run to be backed up, mirrored on other model servers, or accessed when a user does not have an Internet connection to a model server.

To improve speed of the system after observing the test on PHYGROW, the compression sockets were changed for the client and server parts of CME so that they communicate with each other and share data in compressed form. This tripled the speed at which the CME client can interact with models on a server. The performance improvement will be most noticeable for CME users connecting to model servers over slower Internet connections.

After observing use of CME for the past year among an array of graduate students and scientists in our research group, the following enhancements were identified as future development needs:

1. User interface enhancements: Alter the user interface of the CME software so that it is more intuitive and has an improved, user-friendly look-and-feel. An attractive splash-screen describing the software while the software loads while loading with enhanced design of buttons, tabs, menus, etc. so that the program is easier for new users to understand and more intuitively laid-out.

2. Enhanced database interaction: The database interfaces needs to be enhanced to allow CME to more efficiently handle queries and speed up database access time, allowing easier use of other ODBC compliant databases that we currently are not using (like Microsoft Access). These enhancements would allow the results of a model run and place those into the input files for another model run, thereby letting one model's results feed another model's input.

CONCLUSION

The common modeling environment offers multiple model development teams to place their models on the web without altering the code or changing their data structures. The framework provided by the CME system could provide a framework for multiple projects to place the analysis and models used in a wide array of research programs in a computing environment that would provide a cohesive suite of tools for future users. The legacy of the analysis is provided in a form that could be further explored by future users with minimal overhead on the model developers.

The ongoing efforts in collaboration with FAO's WAICENT group offers the most logical entry point for institutionalization of CME. Given the global network of FAO and its strong commitment to distribution of information, CME could play a crucial role in transitioning data into analysis.

REFERENCES

Britton, C. 2001. IT Architectures and Middleware: Strategies for Building Large Integration Strategies. Unisys Series. Addison-Wesley. NY

Houtekamer, P.L., L. Lefaivre, J. Derome, H. Ritchie, and H.L. Mitchell: 1996. A system simulation approach to ensemble prediction. Mon. Wea. Rev. 124:1225-1242

Meehl, G. A., G. J. Boer, C. Covey, M. Latif, and R. J. Stouffer. 2000. The Coupled Model Intercomparison Project (CMIP). *Bull. Amer. Meteor. Soc.* 81:313-318.

Yourdon, E. and Constantine, L.L. 1979 *Structured Design*, Prentice-Hall, Englewood Cliffs, New Jersey.

Acronyms Used:

.csv	Comma separated value format extension
ACT	Almanac Characterization Tool
APEX	Agricultural Policy Environment Extender
ArcIMS	Arc Internet Mapping Systems
ARCVIEW	Arc Geographical Information Systems
ASM	Agricultural Sector Model
BL-GRASS	Blackland Geographic Resources Analysis Support System
CME	Common Modeling Environment
CRSP	Collaborative Research Support Program
CURVEEXPERT	Curve Fitting Software
DSCF	Database Server Configuration File
EPIC	Environmental Policy Integrated Climate
ERDAS	Earth Resource Data Analysis System
FAO	Food and Agriculture Organization of the United Nations
FLAM	Farm Level Analysis Model
FLIPSIM	Farm Level Policy Analysis System
FTP	File Transfer Protocol
GAMS	General Algebraic Modeling System
GASM	Global Agricultural Sector Model
GIS	Geographical Information Systems
IAG	Impact Assessment Group
JAVA	Sun Microsystems Computer Language
JDBC	Java Database Connectivity
JDK	Java Development Kit
JPG	Joint Photographic Experts Group Format
JRE	Java Runtime Environment
LANDDEMAND	Livestock Demand for Land Calculator
MECF	Model Execution Configuration File
MPF	Model Parameters File
Netcdf	Network Common Data Format
NUTBAL	Nutritional Balance Analyzer
ODBC	Object Database Connectivity
PAIA	Priority Areas for Interdisciplinary Action
PHYGROW	Phytomass Growth Simulator
RMI	Remote Method Invocation
SANREM	Sustainable Agricultural and Natural Resource Management
SPSS	Statistical Programs for Social Sciences
SOLVER	Analytical add ons for the GAMS environment
SQL	Structured Query Language
SWAN	Soil Water Analysis
SWAT	Soil and Water Assessment Toll
TAMU	Texas A&M University
TIFS	Template Input Files
WAICENT	World Agricultural Information Center
WXGEN	Weather Generator for EPIC
XML	Extended Markup Language