

# Parallel Distributed Application Framework for Earth Science Data Processing

Petr Votava<sup>1,2</sup>, Rama Nemani<sup>1</sup>, Keith Golden<sup>2</sup>, Dan Cooke<sup>3</sup>, Hector Hernandez<sup>3</sup>,  
Changming Ma<sup>3</sup>

<sup>1</sup> Numerical Terradynamic Simulation Group, University of Montana  
Missoula, MT 59801, USA

{votava,nemani}@ntsg.umt.edu

<sup>2</sup> NASA Ames Research Center  
Moffet Field, CA 94035, USA

kgolden@ptolemy.arc.nasa.gov

<sup>3</sup> Department of Computer Science, Texas Tech University  
Lubbock, TX 79413, USA

{dcooke,hector,ma}@cs.ttu.edu

**Abstract** The management and the processing of Earth science data has been gaining importance over the last decade due to higher data volumes generated by a larger number of instruments and ground stations, and due to the increase in complexity of Earth science models. In order to take full advantage of the vast amount of available data, we need the ability to seamlessly merge data from different sources within the models, and we must be able to process this data in an efficient and timely manner. In order to support these goals, we are developing the Java Distributed Application Framework (JDAF) - a flexible and extensible framework that simplifies the design and implementation of Earth science processing systems. In order to achieve better performance and resource utilization, we parallelize some of the Earth science algorithms within the framework.

## 1 Introduction

The management and the processing of Earth science data has been gaining an importance over the last decade due to higher data volumes generated by a larger number of instruments and ground stations, and due to the increase in complexity of Earth science models that use these data. This growth in volumes and complexity is just the beginning, because in order for us to be able to understand and predict the processes within the Earth, we need a global data set spanning many years and possibly even decades. Additionally, in order to take full advantage of the vast amount of data being produced, we need the ability to seamlessly merge data from different sources within the models. With the launch of NASA's Terra and Aqua missions during the last 3 years, the need for more efficient and scalable data systems is even more apparent. This is true not only at the level of large data processing centers, but also at smaller research labs and universities. Our project addresses many of the issues facing the Earth science community by developing an efficient and scalable technology that incorporates automation in access, transport, translation, distributed processing and analysis of the Earth science

data. Moreover, the technology provides a capability for fusion of data from multiple satellites and ground stations. This enables us to process more data more efficiently and in an intelligent way, and thus give the Earth scientists better foundations for their research and, by doing so, improve the ability to understand and predict Earth system processes.

We are currently testing a prototype of our technology on the Terrestrial Observation and Prediction System (TOPS) [1], whose goal is to provide nowcasts and long-term forecasts of several biospheric variables for the continental US. The main inputs required by TOPS are landcover, daily temperatures, humidity, radiation, fraction of photosynthetically active radiation (FPAR) [2], leaf area index (LAI) [2], and wind vectors. These datasets currently come from a variety of sources: MODIS Terra (FPAR/LAI and landcover), NCEP's Rapid Update Cycle (temperatures, wind vectors, humidity, radiation), and CPC (weather stations providing precipitation and additional temperature information). We are planning additions of other sources of data, for example FPAR/LAI products from Aqua MODIS and from MISR, so that we can improve the reliability of our inputs and thus provide better forecasts generated by TOPS. We will also extend the coverage of TOPS forecasts to other regions around the world, and so we will require even more datasets. Because of the characteristics of the TOPS system, it is a very suitable test-bed for our technology, and provides an important connection with the Earth science community.

The remainder of the paper is structured as follows: in section 2 we introduce in more detail some of the challenges faced by the Earth science community, in section 3 we describe the foundations of our system and in section 4 we show an example of a system where we are currently using our framework. Finally, in section 5 we describe our plans for the future.

## **2 Background**

### **2.1 Earth Science**

The latest generation of NASA Earth Observing System [3] satellites has brought a new dimension to continuous monitoring of the living part of the Earth System, the Biosphere. EOS data can now provide weekly global measures of vegetation productivity and ocean chlorophyll, and many related biophysical factors such as land cover changes or snowmelt rates. However, information with the highest economic value would be forecasting impending conditions of the biosphere that would allow advanced decision-making to mitigate dangers, or exploit positive trends. NASA's strategic plan for the Earth Science Enterprise identifies ecological forecasting as a focus for future research. Ecological forecasting predicts the effects of changes in the physical, chemical and biological, environment on ecosystem activity. Imagine if we could accurately predict shortfalls or bumper crops of agricultural production, or West Nile virus epidemics or wildfire danger 3-6 months in advance, allowing improved preparation and logistical efficiency. The climate forecasting skills of many coupled Ocean-Atmosphere general circulation models (GCM) [4] have steadily improved over the past decade. Given observed anomalies in sea-surface temperatures from satellite data, GCMs are able to forecast general climatic conditions 6-12 months into the future, trends of hotter/colder tem-

peratures and wetter/drier precipitation than normal, with reasonable accuracy. While such climatic forecasts are useful alone, the advances in ecosystem modeling allow us to explore specifically the impacts of these future climate trends on the ecosystem directly.

One of the key problems in adapting climate forecasts to natural ecosystems is the 'memory' that these systems carry from one season to the next (e.g. soil moisture, plant seed banks, fire fuel accumulation etc.). Simulation models are often the best tools to carry forward the spatiotemporal 'memory' information. The power of models that can describe and predict ecosystem behavior has advanced dramatically over the last two decades, driven by major improvements in process-level understanding, computing technology, and the availability of a wide-range of satellite- and ground-based sensors.

## **2.2 Data Processing Issues**

Many of the Earth science processing systems, both in small research labs and universities and in large data centers, are driven by a large number of scripts performing most of the scheduling and processing setups. Despite some advantages of this approach, mainly its rapid development, there are many drawbacks, including difficulties in maintainability, scalability to a larger number of datasets and processes, and flexibility in accommodating new processes and data streams in the existing system. Some of the issues stem from the nature of the scripts and their lack of language-level support to accommodate the translation of the design into the implementation. Other issues relate to the nature of the systems that are developed in this way - they are fast prototypes that often stay around as the only implementation of the system design.

Apart from the lack of flexible and extensible processing framework, one of the main problems in current Earth science systems is a lack of common metadata standards. This makes dealing with large volumes of data very difficult in terms of data fusion and overall system flexibility. Additionally, the data come in many different formats (HDF, HDF-EOS, ASCII, GRIB, binary, and many others), projections, and quality, which can introduce another complexity into the system's ability to handle multiple data streams.

Due to the inflexibility of many current Earth science processing systems, there is often a long time lag between determining a need for a new capability and actually implementing this need in the production. Our Java Distributed Application Framework (JDAF) [5] adds this flexibility that will significantly cut the time required to support new capabilities, whether we need to add a new data stream, produce a new data product, or add a new model.

## **3 System Design**

### **3.1 Overview**

There were two main goals in the design of our system - flexibility and performance. We wanted to be able to add new components into the system with minimum integration efforts and make them produce results in a reasonable amount of time. One of the problems with many Earth science algorithms, is that they are contained in tens of thousands

of lines of legacy code written in C, Fortran and C++, and it would take substantial efforts to re-write all of these algorithms in a way more suitable for integration. Instead, we have decided to write a set of simple wrappers in C++ that would provide interface between our system and the legacy code. These wrappers are subsequently used for interaction between the Java framework and the processing algorithms using the Java Native Interface (JNI) [6]. But adding new processing algorithm to the system is only part of the solution, we would also like to integrate new data streams without the need of changing the legacy code. This part is much harder, because the I/O components of the existing algorithms are often almost inseparable from the core science processing. We instead deploy a set of “filters” that on-the-fly preprocess the new data into a format expected by the processing components. This task is greatly simplified due to the Earth Science Markup Language (ESML) [7] - an XML-based language that provides mechanisms for reading scientific data sets in many formats only by changing their external descriptions stored in XML files, and without the need to modify existing I/O code.

While flexibility of our system was the main goal, we also recognize the importance of performance, because many of the applications of Earth science data can be part of time-critical systems, for example fire or flood monitoring. We exploit the inherent parallelism of Earth science algorithms in two ways - first, we use a cluster of workstations to do the processing of independent components (for example to process the data obtained from the MODIS instrument on Terra or Aqua satellites, we need about 20 files that would cover the United States - these can all be processed independent of each other and we distribute this processing on the cluster). Secondly, we create MPI [10] wrappers around some of the more computationally intensive algorithms, like the FPAR/LAI MODIS algorithm and perform internal parallelization that improves the system performance over smaller geographic regions.

### 3.2 Data Descriptions

An important feature of the application framework is the decoupling of the data from the algorithms. The objective is that rather than rewriting the data processing algorithms to fit the appropriate data set, we implement filters that preprocess the data to the format required by the algorithms. This facilitates faster development and encourages code reuse.

In order for an application to be able to handle multiple data formats in a flexible way, it needs to obtain detailed information about the data - this information can range from data type to distribution information. Because the data vary so greatly in their formats, from ASCII and simple binary, to Hierarchical Data Format (HDF) and HDF-EOS, we had to find a metadata scheme that would be capable of including all the different datasets that are of interest to the Earth science community. We have decided to use metadata standard developed by the Federal Geographic Data Committee (FGDC) [11] <sup>1</sup>. The standard specifies a generic framework that can be used to describe any

---

<sup>1</sup> Although the Earth Science Markup Language (ESML) is very suitable for interpreting the structure of the data, it lacks the metadata descriptions, including projection, quality, and many others. While ESML will very likely be our only choice in the future, we are currently working on both ESML and our FGDC descriptions, because we are dependent not only on the structure of the data, but also on its metadata.

geospatial data with regard to the following aspects: Identification, Data Quality, Spatial Data Organization, Spatial Reference, Entity and Attribute Information, Distribution, and Metadata Reference. The standard also specifies which of the above sections are mandatory and which are optional. This simplifies greatly the data descriptions in case of simple data sets when not all the information has to be included, but remains very expressive in description of complex data sets. FGDC also provides a set of tools for checking that metadata conforms to the specifications, and for conversion to XML and HTML formats.

Since main parts of our application framework are written in Java, we have decided to use XML for the metadata implementation, because Java provides extensive support for handling of XML documents. It is the Java and XML combination that brings the flexibility and extensibility into the design of the application framework. There are two types of data that the framework must be able to handle - external that are coming from outside of the system, and internal that are produced by the system. While we have little control over the external data and their formats besides creating the ESML and FGDC-compliant metadata descriptions, we have decided to use HDF5 [8] for internal data representation. HDF5 is the latest version of the Hierarchical Data Format that provides better support for Java and new internal organization that is very suitable for our applications.

### **3.3 Processing, Distribution and Parallelization**

Many Earth science algorithms are very complex, but they also often have only a small degree of spatial dependency and thus are ideal for parallel processing. We utilize the distributed features of the Java programming language to accommodate parallel processing. With our framework we can build flexible and scalable processing 'pipelines' that include preprocessing, processing and automated result analysis as independent modules. This gives us the flexibility to add and remove modules on the fly, as well as re-use existing code, and thus enables us to concentrate more on the science itself rather than on system integration. Finally, we implement a batch mode so that the system can run without user interventions for long periods of time, providing the scientists with an automated way to obtain the results they need quickly and efficiently.

One of the many aspects of the Earth science data processing is the volume of the data. This fact together with time complexity of some of the algorithms led us to provide mechanisms for parallel execution of the data processing whenever possible. For example, a task of creating images of continental US from MODIS data involves reprojecting, mosaicing, subsampling and image conversion of the data. The MODIS data comes in the form of tiles (continental US consists of about 20 tiles) that, at least during part of the processing, are independent of each other. The part of the system that handles the parallelism is the scheduler. The user/developer submits a request to the scheduler describing what he/she wants to accomplish and how, and the scheduler will load appropriate algorithm modules, I/O modules, and setup the execution sequence that corresponds to the user's requirements, executing modules in parallel whenever possible. In the example above, the scheduler would execute all the reprojection processes in parallel with synchronization point before entering the mosaicing process. The implementation of the execution environment is done by facilities of Java RMI [9]. Each

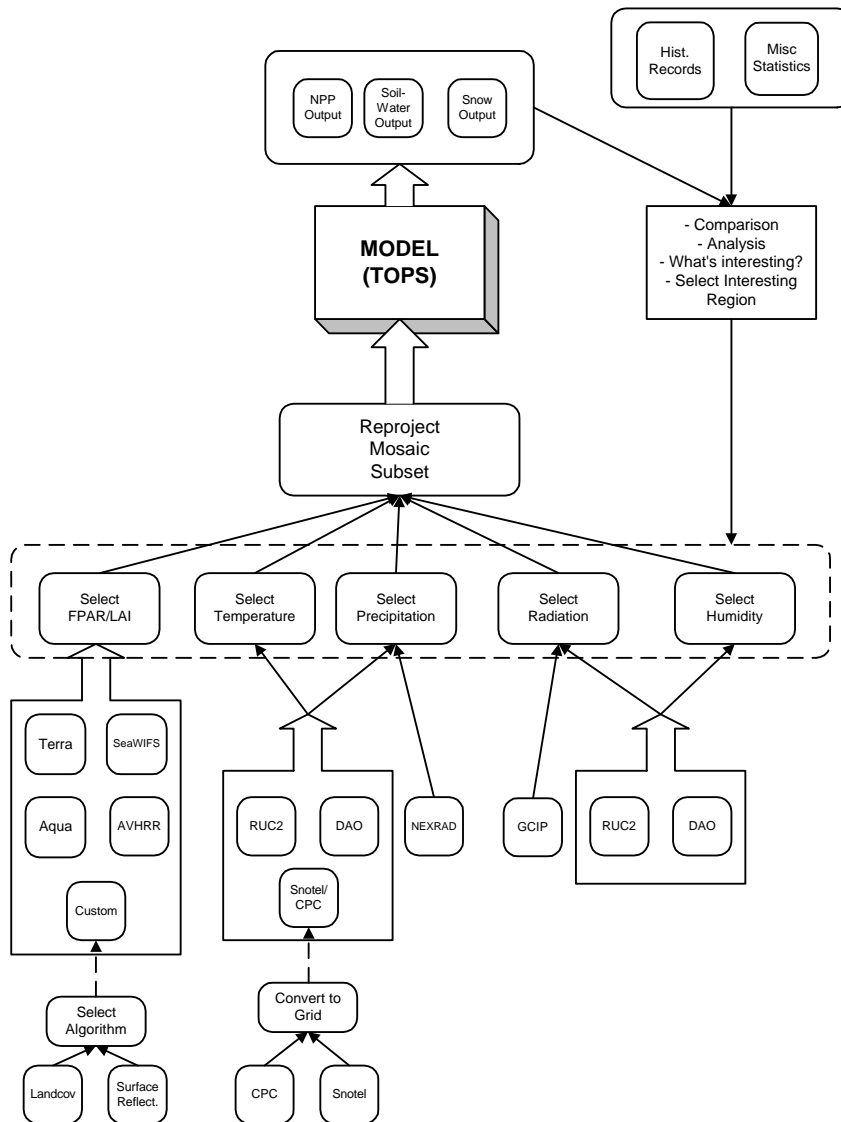
algorithm object has to implement a simple interface, and provide the code for `execute()` method of the class. This object is then passed to the scheduler, which in turn forwards it to one of the execution servers. The execution server calls the `execute()` method provided by the object and returns the results of the execution to the caller. The algorithms themselves are often implemented in C or C++ and we use the Java Native Interface (JNI) to make calls to the shared libraries that contain the appropriate modules.

In addition to exploiting the external (per-file) parallelism with RMI, we also perform internal parallelization of some of the key and compute intensive algorithms using MPI. This is important when we want to perform large number of regional simulations and analysis in interactive mode, or if the algorithm is a part of a near-real-time system, as it may be in case of determining fire danger over a particular region. This internal parallelization is especially efficient on CPU-bound algorithm executing in an environment with more resources (CPUs) that is needed for external parallelization. We have started by parallelizing one of the key EOS products - Leaf Area Index (LAI) and Fraction of Photosynthetically Active Radiation (FPAR), which are an essential input into our TOPS system. We have obtained mixed results from this undertaking. While we were able to speed-up the standalone version of the algorithm significantly, we have also realized that with the advances in CPU capabilities over last 5 year many previously CPU-bound algorithms are now I/O-bound. This will shift some of our future efforts to pay more close attention to parallelization of the I/O components of the Earth science algorithms.

## 4 TOPS Case Study

In order to estimate possible future states of the biosphere, we are building a flexible system that integrates ecosystem models with frequent satellite observations, that can be forced by weather or climate forecasts, and downscaled to resolutions appropriate to resolve surface processes. The Terrestrial Observation and Prediction System is a modeling software system that automatically integrates and pre-processes EOS data fields so that land surface models can be run in near-realtime with minimal intervention, helping accurate and timely interpretation of Earth Observing system data. Such a system will allow us to determine the vulnerabilities of different socio-economic and resource systems to fluctuations within our biosphere, and would help in mitigating potential negative impacts. The goal of the TOPS system is the monitoring and prediction of changes in key environmental variables. Early warnings of potential changes in these variables such as soil moisture, snow pack, primary production and stream flow, could enhance our ability to make better socio-economic decisions relating to natural resource management and food production. The accuracy of such warnings depends on how well the past, present, and future conditions of the ecosystem are characterized. The overall data flow through the system is depicted in Figure 1. The inputs needed by TOPS include:

- Fractional Photosynthetically Active Radiation (FPAR) and Leaf Area Index (LAI)
- Temperatures (minimum, maximum, and daylight average)
- Precipitation



**Figure 1.** Terrestrial Observation and Prediction System (TOPS) data fusion.

- Solar Radiation
- Humidity

We have several potential candidates inputs at the beginning of each model run. The basic properties of the inputs are listed in Table 1. Even with this fairly small model, there is a good variety of inputs we need to select from, depending on our goal.

Source	Variables	Frequency	Resolution	Coverage
Terra-MODIS	FPAR/LAI	1 day	1km, 500m, 250m	global
Aqua-MODIS	FPAR/LAI	1 day	1km, 500m, 250m	global
AVHRR	FPAR/LAI	10 day	1km	global
SeaWIFS	FPAR/LAI	1 day	1km x 4km	global
DAO	temp, precip, rad, humidity	1 day	1.25 deg x 1.0 deg	global
RUC2	temp, precip, rad, humidity	1 hour	40 km	USA
CPC	temp, precip	1 day	point data	USA
Snotel	temp, precip	1 day	point data	USA
GCIP	radiation	1 day	1/2 deg	Continental
NEXRAD	precipitation	1 day	4 km	USA

The first step in the TOPS processing is the selection of the inputs. One thing to note is that a criterion in selection may also be availability, because some inputs are not always available. For example, both the Terra and Aqua satellites experienced technical difficulties and data dropouts over periods ranging from few hours to several weeks. Before the selection process can begin, we have to get the data into some common format, so that the dataset comparison is possible. In the case of TOPS, this comparison is done with gridded data, so we have to make sure that we convert the point data (CPC and Snotel) to grid data, which by itself is fairly complex and time-consuming process. Next, the data are selected based on the goal, which at the start of the process is just the current status of the variables we are interested in (primary productivity, soil water content, ...) over the continental US. After the data are selected, we must put them into common format, which may involve reprojecting them into a common projection, subset the dataset from its original spatial extent, and populate the input grid used by the model. The data are then ran through the TOPS model, which generates desired outputs. What follows is a new step in many Earth science systems: the data are compared against long-term records and statistics, and the system determines whether there is something important happening in the covered area. An example of such events may include new fires being ignited, or rapid ice-melt and thus flooding potential. Whatever the “interesting” event is, the system tries to investigate it further, and one way of accomplishing this is by getting a higher resolution information and going through the input selection process again. The goal has now changed, both in terms of detail, but also in geographic extent, because we no longer need to run the model over the entire continent, but only over several selected areas. Furthermore, we would like more detailed information, so we may actually choose to run a more complex model that runs longer, but provides us with higher quality information on the ongoing events, together with the prognosis for near future. As we can see, when this feedback loop is added to

TOPS, the complexity of the system goes up even further. TOPS provides only a simple illustration of the potential problems, and is by far not as complex of many other models and systems in the Earth science, some of which take dozens of different inputs, with sizes reaching into terabytes for each model run.

## 5 Conclusions and Future Work

The comparison of the system developed using our application framework to our baseline script-driven system shows improvements in both performance and flexibility. While it can take days and even weeks to integrate a new module into our script-oriented system, it took less than one hour to integrate a complete MODIS production module using the application framework. Additionally, the application framework uses a scheduler to distribute the execution over multiple machines in more intelligent and flexible way than the remote-shell style of execution of the script-based system. While much of the functionality of the application framework is also possible to implement in a scripted environment, the development time, complexity, and maintainability of such system is much more costly.

We have a prototype of the TOPS system that is producing data on continuous basis, but we want to add a better handling of the input data fusion. This is accomplished through the deployment of a planner that uses constraint-based logic to generate optimal processing plans aimed at higher data quality. Additionally, we are also planning to add a natural language interface (NLI) to the system, so that users can ask questions like “What is the fire danger for Western Montana for tomorrow?”, the system will generate the appropriate plan, execute it, and return the results to the user. This makes the system even more interactive, but it also points out the importance of the fast execution of some of the processing components through parallelization and distributed processing. Finally, we will be integrating XML-RPC and SOAP into our servers for better flexibility and easier deployment.

## References

1. Nemani, R., Votava, P., Roads, J., White, M., Thornton, P. and Coughlan, J. *Terrestrial observation and prediction system: Integration of satellite and surface weather observations with ecosystem models*. Proceedings of the 2002 International Geoscience and Remote Sensing Symposium (IGARSS), Toronto, Canada. 2002.
2. Knyazikhin, Y., et al. *MODIS Leaf Area Index (LAI) and Fraction of Photosynthetically Active Radiation Absorbed by Vegetation (FPAR) Product (MOD15) Algorithm Theoretical Basis Document*. 1999.
3. King, M., and R. Greenstone. *1999 EOS Reference Handbook, A Guide to NASA's Earth Science Enterprise and the Earth Observing System*. 1999.
4. McGuffie, K., and A. Henderson-Sellers. *A Climate Modeling Primer*. John Wiley & Sons, 1997.
5. Votava, P., et al. *Distributed Application Framework for Earth Science Data Processing*. Proceedings of the 2002 International Geoscience and Remote Sensing Symposium (IGARSS), Toronto, Canada. 2002.
6. Gordon, R. *Essential JNI: Java Native Interface*. Prentice Hall, 1998.

7. Ramachandran, R., et al. *Earth Science Markup Language: A Solution for Generic Access to Heterogeneous Data Sets*. In Proceedings of Earth Science Technology Conference, 2001.
8. The National Center for Supercomputing Applications. *HDF5 abstract data model*. Presented to NASA HDF-EOS Workshop, 1999.
9. Pitt, E., and K. McNiff. *java.rmi: The Remote Method Invocation Guide*. Addison Wesley, 2001.
10. Gropp, W., Lusk, E., Skjellum, A. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. 2nd edition, MIT Press - Scientific and Engineering Computation Series, 1999.
11. Federal Geographic Data Committee.  
*Content Standard for Digital Geospatial Metadata Workbook version 2.0*.  
2000.