

Interface Design for Interoperability for the Land
Information System
Submitted under Task Agreement GSFC-CT-2
Cooperative Agreement Notice (CAN)
CAN-00OES-01
Increasing Interoperability and Performance of
Grand Challenge
Applications in the Earth, Space, Life, and
Microgravity Sciences

History:

Revision	Summary of Changes	Date
1.0	Draft	01/06/03
2.0	Revised with CT team comments	03/20/03
3.0	Modified for LIS 4.0	07/31/04



NASA Goddard Space Flight Center,
Greenbelt, MD 20771

Contents

1	Introduction	4
2	Land Surface Modeling in LIS	4
3	LIS Architecture	6
4	Interoperability in LIS	7
4.1	Internal Interoperable Design Features in LIS	7
4.2	External Interoperable Features in LIS	9
4.3	Use of ESMF in LIS	10
4.4	ALMA Interfaces in LIS	12
4.4.1	Input Interface	14
4.4.2	ALMA Wrappers	15
4.4.3	Output Interface	15
5	Final Remarks	17

List of Figures

1	A schematic representation of land surface modeling in LIS	5
2	LIS software architecture and components	6
3	LIS software architecture and components	10
4	Object design for inclusion of a new LSM (<code>LSS_ex</code>)	11
5	Use of ESMF in LIS	13
6	ALMA interfaces in LIS	14

List of Tables

1	Mapping of Forcing variables to NOAH input variables	16
2	Mapping of ALMA and VIC output variables : General Energy Balance	17
3	Mapping of ALMA and VIC output variables : General Water Balance	18
4	Mapping of ALMA and VIC output variables : Surface State Variables	18
5	Mapping of ALMA and VIC output variables : SubSurface State Variables	19
6	Mapping of ALMA and VIC output variables : Evaporation Variables	19
7	Mapping of ALMA and VIC output variables : Other hydrologic Variables	19

8	Mapping of ALMA and VIC output variables : Cold Season Processes	20
9	Mapping of ALMA and VIC output variables : Variables to be compared with remote sensed data	20

1 Introduction

This document describes the design policy for interoperability for the Land Information System (LIS) [5] implemented under funding from NASA's ESTO Computational Technologies Project. This design is submitted to satisfy the Task Agreement GSFC-CT-2 under Cooperative Agreement Notice CAN-00-OES-01 increasing interoperability and performance of grand challenge applications in the earth, space, life, and microgravity sciences.

Code interoperability is important not only between components of a research application, but also between different applications, to decrease the cost of development. Research applications with reusable components facilitate faster development of future applications and enables a broader user base.

This document outlines two different types of interoperability that LIS intends to define and adopt:

- **Internal Interoperability:** This is interoperability that is provided by LIS to the land surface modeling community. LIS will provide an interoperable framework for the land surface modeling community by defining adaptive, extensible interfaces for incorporating new land surface models into LIS.
- **External Interoperability:** Participate with Earth, space, life, and microgravity scientific communities by adopting the utilities and compliance guidelines provided by the earth system modeling framework (ESMF) [4]. LIS will also comply with established land surface modeling standards such as assistance for land modeling activities (ALMA) [1].

2 Land Surface Modeling in LIS

Land surface modeling seeks to predict the terrestrial water, energy, and biogeochemical processes by solving the governing equations of the soil-vegetation-snowpack medium. The land surface and atmosphere are coupled to each other over a variety of time scales through the exchanges of water, energy, and carbon. An accurate representation of land surface processes is critical for improving models of the boundary layer and land-atmosphere coupling at all spatial and temporal scales and over heterogeneous domains. Long term descriptions of land use and fluxes also enable in the accurate assessments of climate characteristics. In addition to the impact on the atmosphere, predicting land surface processes is also critical many real-world applications such as ecosystem modeling, agricultural forecasting, mobility assessment, and water resources prediction and management.

A schematic representation of land surface modeling in LIS is shown in Figure 1. Land surface models typically require three types of inputs: 1) Initial conditions, which describe the initial state of the land surface; 2) Boundary conditions, which describe both the upper (atmospheric) fluxes or states also known as “forcings” and the lower (soil) fluxes or states; and 3) Parameters, which are a function of soil, vegetation, topography, etc. Using these inputs, land surface models solve the governing equations of the soil-vegetation-snowpack medium and predicts surface fluxes (sensible, latent, ground heat, runoff, evaporation) and soil states (moisture, temperature, snow), providing a realistic representation of the transfer of mass, energy, and momentum between a vegetated surface and the atmosphere [8]. The model results are aggregated to various temporal and spatial scales to assess water and energy balances. The results can also be compared with in-situ observations if available.

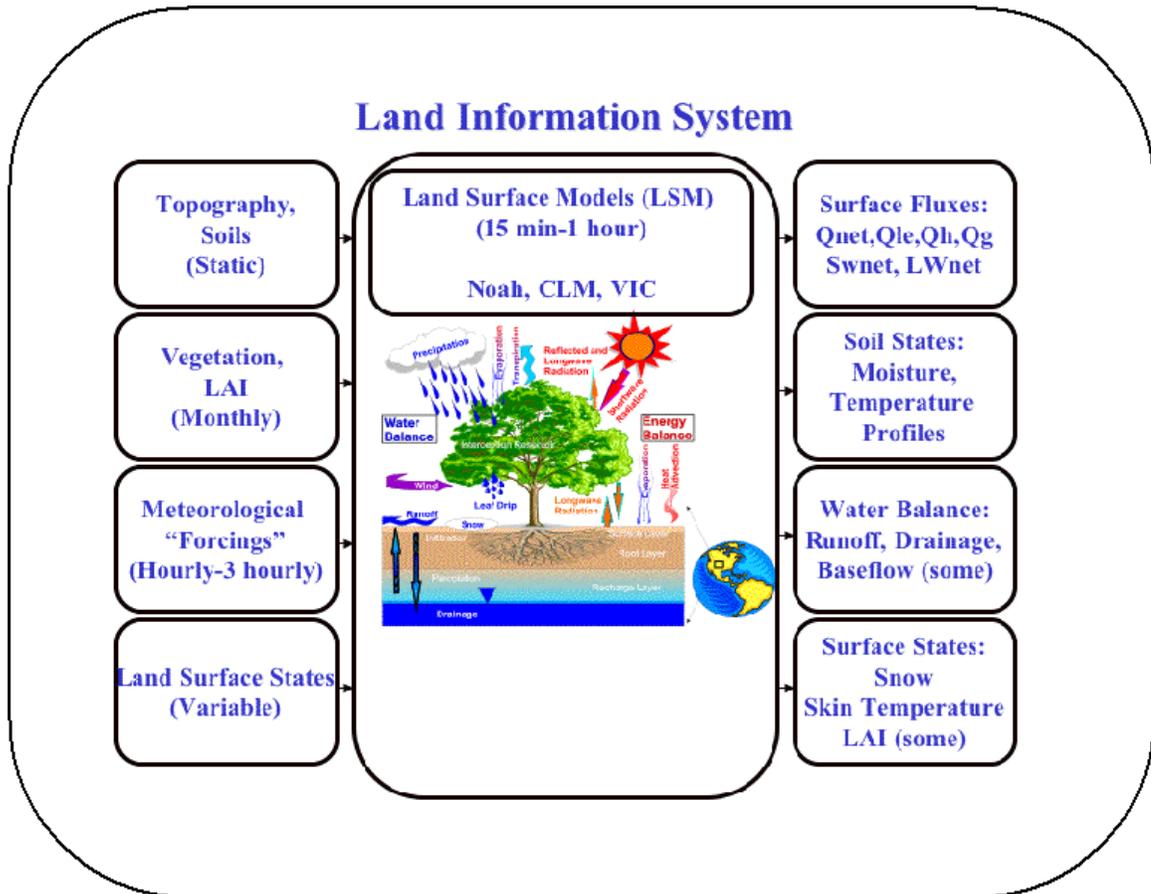


Figure 1: A schematic representation of land surface modeling in LIS

3 LIS Architecture

The LIS software system consists of a number of components: (1) LIS driver: the core software that integrates the use of land surface models, high performance computing, use of various sources of data, and the domains of execution. (2) A number of community land surface models and (3) tools to manage the data throughput at high resolutions, and (4) visualization tools. Various software components of LIS are shown in Figure 2.

LIS driver uses a model control and input/output system that drives multiple offline one-dimensional LSMs. The inputs to these LSMs consist of data from various satellite and ground-based observational systems. The input data are processed and supplied to the land surface models, which in turn produce optimal output fields of land surface states and fluxes. In addition to using real time output from numerical prediction models and satellite radar measurements, LIS also uses model parameters from topography, vegetation, and soil coverages. The LIS driver employs the use of high performance computing tools to meet the increased computational requirements and the Grid Analysis and Display System (GrADS [3])-DODS (Distributed Oceanographic Data System) (GDS [9]) for data management. The visualization capabilities in LIS are based on a multi-tier client-server system architecture. Different data servers is employed to handle various types client requests, such as web-based and DODS-based.

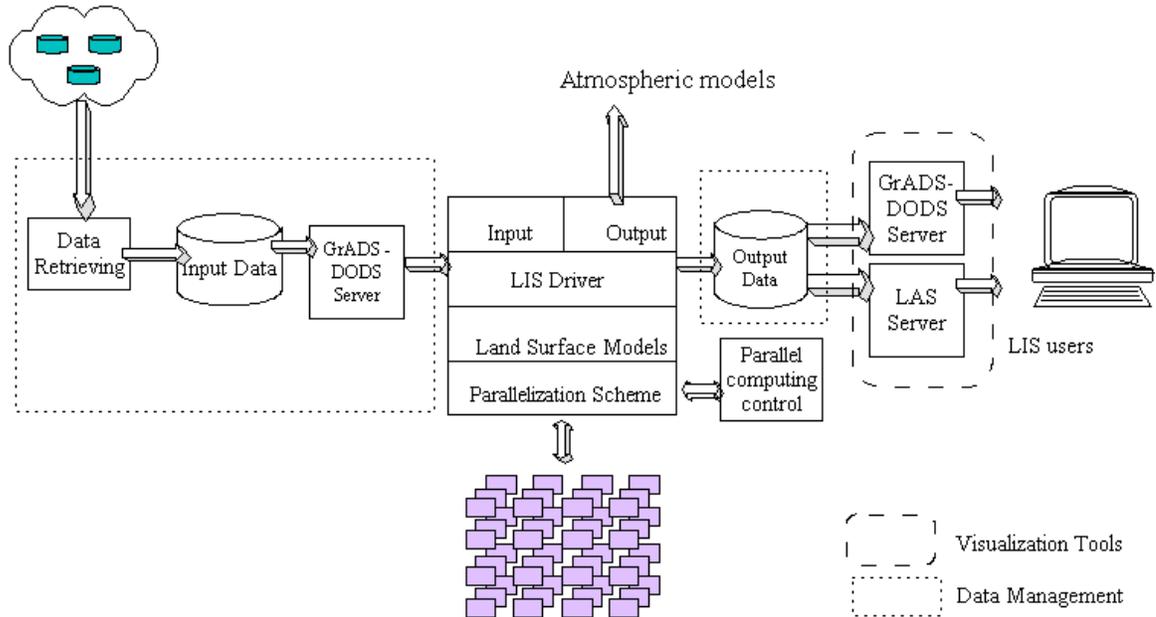


Figure 2: LIS software architecture and components

4 Interoperability in LIS

LIS consists of an integrated set of components and routines organized into several functional abstractions to facilitate development of land surface modeling applications. The components in LIS are implemented using object oriented design principles. LIS integrates these components into a framework facilitating their effective use in solving a variety of different problems. The design of LIS aims at providing extensible features that enable the reuse of LIS structure as well as the use of standards and tools proposed by other earth system modeling groups. The following sections describe these aspects of LIS design.

4.1 Internal Interoperable Design Features in LIS

As defined in the software engineering literature [6], an object-oriented framework represents a software system designed for a family of problems and provides reusable design for applications within that domain. The reusable, “semi-complete” nature of object oriented frameworks makes it easier to build correct, portable, efficient, and inexpensive applications. An object oriented framework normally provides a number of points of flexibility in the design called “hot spots” [7]. Hot spots are abstract methods that must be implemented in order to use the framework for a specific application. The parts of a framework that cannot be altered are called the kernel or frozen spots. The use of hot spots provides implicit reuse of high quality proven software. By incorporating these principles into design, LIS aims to provide an “off-the-shelf” framework for land surface modeling applications.

The LIS software is primarily written in Fortran 90 programming language. Fortran 90 provides a number of features that are useful for object oriented style of programming such as derived types, modules, and generic interfaces, but lacks the support for object oriented properties such as inheritance and run-time polymorphism. However, it is possible to emulate these properties in software [2], enabling an object-oriented programming style in Fortran 90. The compile-time polymorphism in LIS is simulated by the use of virtual function tables. C language allows the capabilities to store functions. A Fortran 90 program can interface with C to store Fortran 90 functions to be invoked at runtime. By combining the features of both these languages, LIS uses a complete set of operations with function pointers.

The overall LIS design incorporates many object oriented principles, such as encapsulation of data and control, inheritance, and compile-time polymorphism. Similar to the “semi-complete” nature of the object oriented frameworks, LIS design provides common functionalities for land surface modeling, leaving the variable functionalities to be filled in by the user. The number of variable functionalities in LIS include: (1) interfaces to facilitate the incorporation of LSMs, (2) meteorological input schemes,

(3) various domains, (4) observation-based inputs and (5) land surface parameters. Components representing each of these functionalities are organized into different packages as shown in Figure 3. The dotted lines represent the dependencies between different packages in which an arc from A to B indicating that package A uses package B.

The **driver** package refers to the LIS driver routines, which handles operations related to the overall control, including the implementation of time, data management tools using the GDS server, I/O methods, error logging, load balancing, domain decomposition and parallel processing. The abstractions of land surface model operations, meteorological and observational forcing schemes, and domains are encapsulated in different modules included in the **driver** package.

The “plugin” packages, **lsm-plugin**, **forcing-plugin**, and **domain-plugin**, contain extensible interfaces for incorporating new land surface models, meteorological forcing schemes, and domains, respectively. The implementations of the land surface model schemes such as CLM (from National Center for Atmospheric Research (NCAR)), Noah (from National Center for Environmental Prediction (NCEP)), and VIC (from University of Washington and Princeton University) are organized in the **lsms** package. Similarly, **baseforcing** package contains the implementations of meteorological forcing schemes (such as GEOS (Goddard Earth Observing System from NASA’s Goddard Space Flight Center), GDAS (Global Data Assimilation System from NCEP), etc.) and the **domain** package contains the implementation of domains (such as a lat/lon -based, gaussian, etc.). The implementations of observational-based schemes for precipitation and radiation are included in packages **obspcps** and **obsrads**, respectively. LIS uses the interfaces in the plugin packages to invoke these actual implementations, as shown in Figure 3 by the dotted lines from the plugin packages to **lsms**, **domain**, **baseforcing**, **obsrads**, and **obspcps** packages.

The implementation of user-defined components and their invocation are based on the object oriented principles of inheritance and compile-time polymorphism. The implementation of a land surface scheme, for example, is assumed to be a specific instance of the abstraction defined by the **lsm_module** in the **driver** package. This module defines abstract methods that capture the basic operation of a land surface model, such as initialization, use of parameters, use of boundary conditions, model execution, routines for output and restart. The specific instance of a land surface model scheme provides implementations of each of these methods, which then are then included and invoked through the **lsm-plugin** interfaces. The **lsm_module** contains a global table of pointers for each LSM in the inheritance hierarchy that is user-defined in the **lsm-plugin** package. The **lsm_module** acts as the polymorphic class, delegating the program flow based on the global pointer that is instantiated. An example of the implementation of a new LSM (**LSS_ex**) is shown in Figure 4. The **LSS_ex** object derives from the **lsm_module**. The **lsm-plugin** stores the spe-

cific methods implemented for the `LSS_ex` in the inheritance hierarchy, and invokes them when required. The flexible design of the `lsm-plugin` also allows for ensemble implementations of land surface models. Such an implementation helps in sharing of functional modules across models without having to reimplement routines for each model. Plugin components for domain and meteorological forcing schemes are implemented in a similar fashion, with a corresponding polymorphic class and a table of global pointers to control delegations and control flow. The procedure to implement new components is similar to that described for a land surface scheme, by defining the new component to derive from the corresponding polymorphic class.

In addition to providing implementations for the required interfaces of the polymorphic classes, the user defined components can also reuse existing modules in LIS such as the implementation of time management (`timemgr`), implementation of high performance computing environment (`spmdMod`), GDS-based I/O (`opendap_module`), etc.

The adaptable interfaces in LIS described above enable the reuse of the broad set of data, high performance computing, data management, visualization tools, and the land modeling infrastructure in LIS. Further, the LIS framework allows researchers to perform intercomparisons of output and sensitivity experiments of different LSMs, meteorological scheme, and input on various domains.

4.2 External Interoperable Features in LIS

To interoperate with other scientific modeling communities, LIS has adopted a number of modeling standards and other frameworks in its design. ALMA is a flexible data exchange convention that LIS adopts to facilitate the exchange of forcing data for LSMs and the results produced by them. The output data variables and formats, and the variables passed between LIS and the land models follow the ALMA specification. By implementing the ALMA convention, LIS can exchange data with other land modeling systems that are also ALMA compliant. Further, ALMA compliance enables LIS to be used for intercomparisons of land surface models.

Another interoperable component that LIS uses is the ESMF. The purpose of ESMF is to develop a framework that provides a structured collection of building blocks that can be customized to develop model components. ESMF can be broadly viewed as consisting of an infrastructure of utilities and data structures for building model components and a superstructure for coupling and running them. ESMF provides a utility layer that presents a uniform interface for common system functions such as time manager, basic communications, error handler, diagnostics, etc. LIS uses a number of ESMF utility tools that has enabled software reuse and as a result, ease of development.

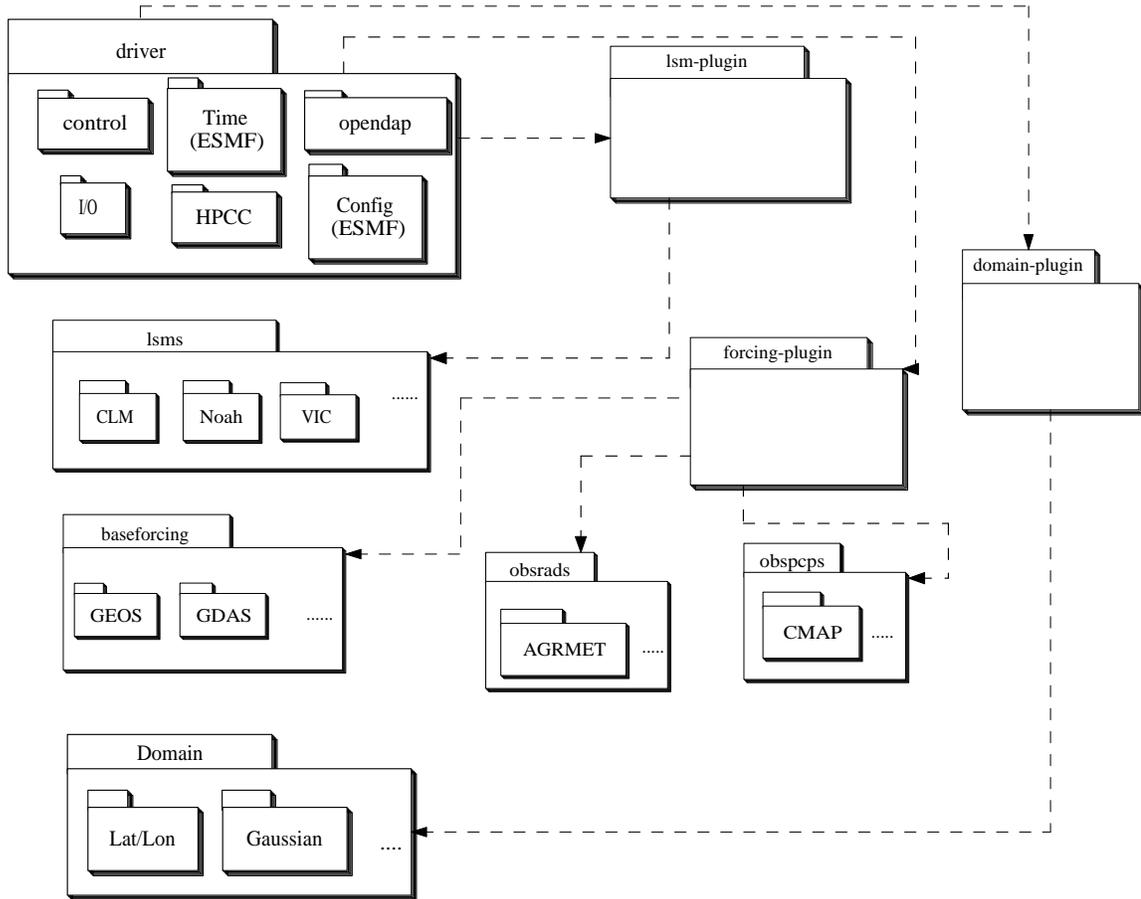


Figure 3: LIS software architecture and components

4.3 Use of ESMF in LIS

ESMF defines a number of guidelines for applications that are intended to be coupled with other earth system models. The basic ESMF architecture is a layered structure, where the user applications implemented in ESMF are sandwiched between the ESMF Superstructure and ESMF Infrastructure.

To enable LIS to couple to other Earth system models using ESMF, the LIS user code is wrapped using the superstructure layer to provide a context for interconnecting input and output data streams. The motivation behind overlaying the ESMF superstructure over LIS is to enable LIS to act as the land modeling component in a coupled system. The land models implemented in LIS follow a common ALMA data exchange convention. The combination of a common structure for user component interaction (ESMF) and data exchange convention (ALMA) facilitates the use of LIS

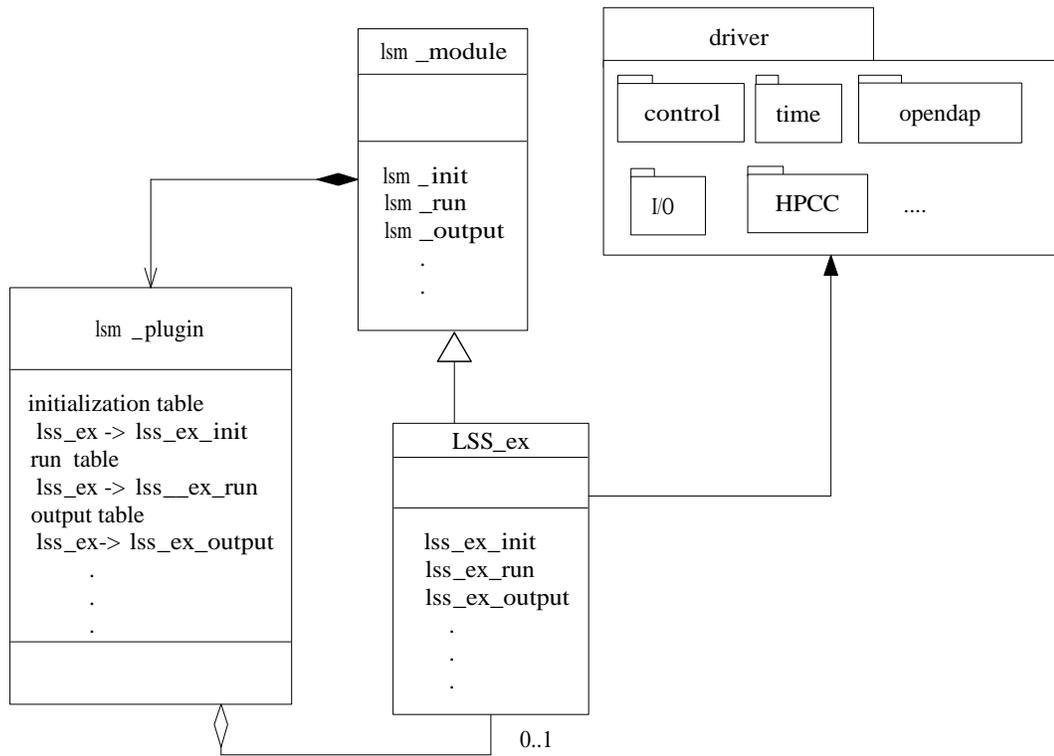


Figure 4: Object design for inclusion of a new LSM (LSS_ex)

as a common land component without having to adapt each land surface model to be ESMF compliant.

As mentioned earlier, LIS uses output from operational forecast models as meteorological inputs to the land surface models in LIS. To simulate a coupled system where the land surface models are forced by outputs from atmospheric models, the forcing components in LIS are implemented as a separate ESMF gridded component. Figure 5 shows the LIS-ESMF prototype with the land surface models implemented as a gridded component (`lsmGridComponent`) and the forcing schemes as a separate component (`forcingGridComponent`). The `AppDriver` is the main program that controls the overall program execution. The control flow of the program is done through the `init`, `run`, and `finalize` methods implemented in each gridded component. The `forcingGridComponent` encapsulates different types of forcings, implemented through the plugin interfaces in LIS, and similarly, the `lsmGridComponent` encapsulates the implementations of different lsms. The exchange of data from the forcing component to lsm component is achieved through a coupler component (`forcing2lsmCoupler`). In this prototype implementation, there is no feedback from the land surface models to the atmospheric components. Further, the gridded components use the modular structures implemented in the `driver` package, but the `driver` does not control the program execution.

In addition to the superstructure layering, LIS also adopts elements of infrastructure constructs that are required to support consistent, guaranteed behavior of components. These currently include the ESMF infrastructure utilities to support consistent time, alarms, and calendar management, configuration tools, use of ESMF fields, arrays, and constructs to support parallel processing such as virtual machine, `DELayouts`.

LIS provides the flexibility to add land surface models, new domains, and the ESMF structure provides the capabilities to interact with other esmf-compliant systems. In addition to the advantage of not having to tailor each land model to be ESMF-compliant, a coupled system using LIS as the land component can take advantage of the high performance computing, data management, remote data accessing capabilities using the GDS server, and the visualization tools in LIS.

4.4 ALMA Interfaces in LIS

ALMA is a data exchange convention to facilitate the exchange of forcing data for LSM and the results produced by these schemes. The ALMA scheme enables inter-comparisons of land surface schemes and ensures that the implementation of procedures to exchange data needs to be done only once. ALMA provides a list of variables needed to force LSMs and a summary of output variable definitions for LSM inter-comparisons.

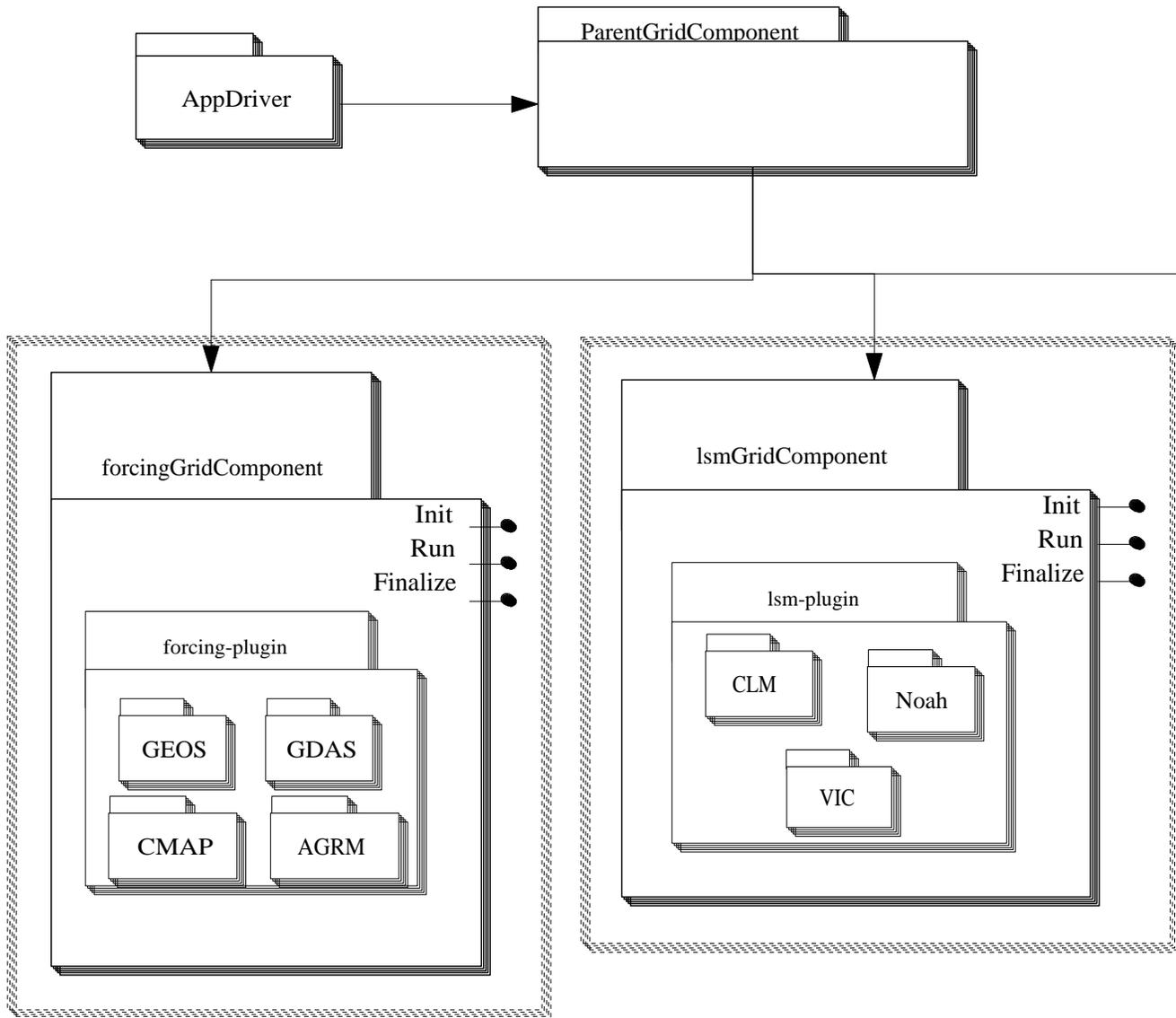


Figure 5: Use of ESMF in LIS

By implementing the ALMA convention in the LIS driver, LIS exchanges data with other land surface modeling systems that are also ALMA compliant. Further, it also enables the use of LIS for intercomparison of land surface models for high resolution global modeling.

In order for LIS to be ALMA compliant, a number of interfaces are defined as shown in Figure 6. The forcing data is fetched from various locations on the internet, and after preprocessing is fed to the LIS driver, which in turns controls the execution of different LSMs. The input interface converts the forcing data into an ALMA compliant form. The ALMA wrappers for each LSM perform the translation of LIS driver variables to the LSM variables. The output interfaces convert the outputs from various LSMs into an ALMA format. Various design issues for these interfaces are discussed below.

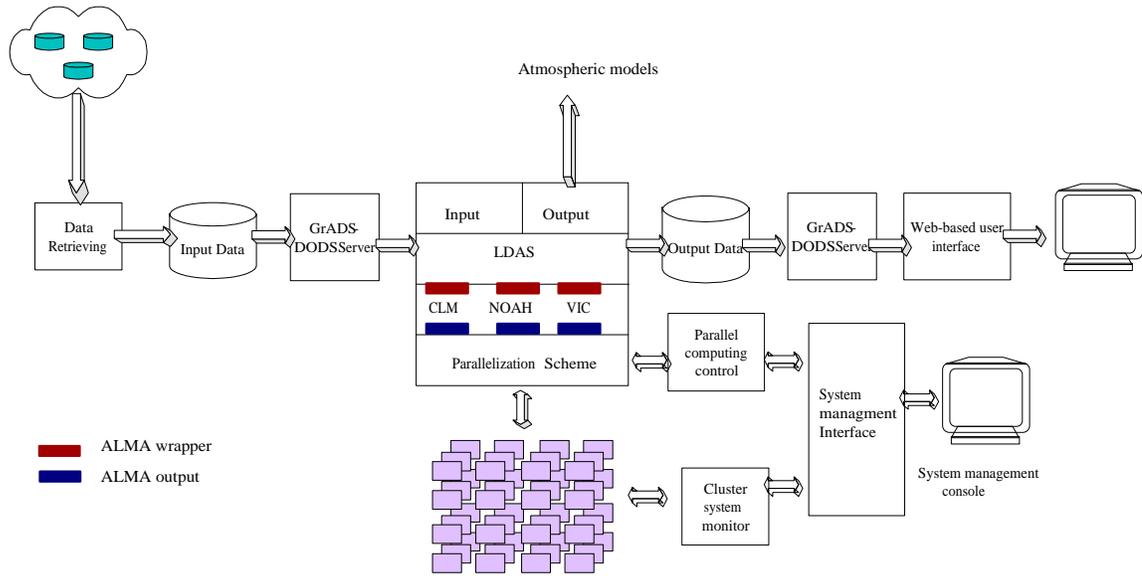


Figure 6: ALMA interfaces in LIS

4.4.1 Input Interface

Global atmospheric model predictions provide baseline forcing for LIS, but whenever possible, the modeled fields are replaced or corrected by observation-based fields. The global data are currently in various different data formats. The preprocessing routines for input data convert the fetched data from internet into the format that the LIS driver uses. The Input interfaces then uses the input forcing ALMA definitions to generate an ALMA compliant format. Special attention need to be paid to the following issues.

- Units : All the required information to convert a forcing variable to the ALMA definition form need to be supplied. An exhaustive list of possibilities for the each forcing field need to be defined. Some special cases might also include dimensionless variables. For e.g, all the required information to convert a relative humidity value to a specific humidity value need to be specified.
- Direction : The sign conventions for each variable definition need to be converted to the ALMA format. Some forcing schemes might define a positive sign to be for an exchange from land to atmosphere, whereas another might consider positive sign to be for a downward direction from the sun to the earth. An exhaustive list of possible fields that need to convert a given directional definition to the ALMA format need to be specified.

4.4.2 ALMA Wrappers

Each LSM scheme included in LIS is expected to be capable of receiving variables in the ALMA form. The ALMA wrappers for each LSM will perform the required conversion from LIS driver variables to LSM variables in accordance with the ALMA format. The design includes a list of conversions required. Table 1 includes a sample list a mapping of forcing data to NOAH variables. Similar tables are implemented for other LSMs included in LIS.

4.4.3 Output Interface

Defining a generic output interface that converts output variables from different LSMs to an ALMA format is difficult, since explicit information is required to do the mapping from an LSM variable to a corresponding ALMA output variable. One of the intents of the ALMA standard is to put the onus of complying to the ALMA output standard on the LSMs so that intercomparisons between them can be done seamlessly. LIS adopts this philosophy, assuming that the LSMs are ALMA compliant.

ALMA output standard lists a number of mandatory variables that are required to do water and energy balance. The output interface uses these variables to compute water and energy balance calculations for different LSMs. The output of recommended and optional variables depends on the LSM employed.

A mapping between lists of ALMA output variables and VIC variables are presented in Tables 2 to 9. Similar to the input interface, a mapping between the output variables and ALMA output variables are implemented for other land surface models in LIS.

The ALMA standard categorizes each ALMA variables into a priority category, which appears in Tables 2 to 9 under the heading Priority. The priority indicates

Table 1: Mapping of Forcing variables to NOAH input variables

ALMA variable	Units	Sign (direction of positive values)	NOAH variable	Units	Sign	Required Conversion
<i>Wind_N</i>	$\frac{m}{s}$	Northward	<i>VWIND</i>	$\frac{m}{s}$	Northward	-
<i>Wind_E</i>	$\frac{m}{s}$	Eastward	<i>UWIND</i>	$\frac{m}{s}$	Eastward	-
<i>Rainf</i>	$\frac{kg}{m^2s}$	Downward	<i>PRCP</i>	$\frac{mm}{s}$	Downward	
<i>Snowf</i>	$\frac{kg}{m^2s}$	Downward				
<i>Tair</i>	<i>K</i>	-	<i>SFCTEMP</i>	<i>K</i>	-	-
<i>Qair</i>	$\frac{kg}{kg}$	-	<i>Q2</i>	-	-	-
<i>PSurf</i>	<i>Pa</i>	-	<i>SFCPRS</i>	<i>Pa</i>	-	
<i>SWdown</i>	$\frac{W}{m^2}$	Downward	<i>SOLDN</i>	$\frac{W}{m^2}$	Downward	-
<i>LWdown</i>	$\frac{W}{m^2}$	Downward	<i>LWDN</i>	$\frac{W}{m^2}$	Downward	-
<i>LSRainf</i>	$\frac{kg}{m^2s}$	-				
<i>CRainf</i>	$\frac{kg}{m^2s}$	-	<i>CPCP</i>	$\frac{mm}{s}$	-	
<i>CSnowf</i>	$\frac{kg}{m^2s}$	-				
<i>LSSnowf</i>	$\frac{kg}{m^2s}$	-				
<i>SVRainf</i>	$(\frac{kg}{m^2s})^2$	-				
<i>SVSnowf</i>	$(\frac{kg}{m^2s})^2$	-				
<i>Wind</i>	$\frac{m}{s}$	-	<i>SFCSPD</i>	$\frac{m}{s}$	-	-

Table 2: Mapping of ALMA and VIC output variables : General Energy Balance

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>SWnet</i>	$\frac{W}{m^2}$	Downward	M	<i>net_short</i>	$\frac{W}{m^2}$	Downward	Y
<i>LWnet</i>	$\frac{W}{m^2}$	Downward	M				
<i>Qle</i>	$\frac{W}{m^2}$	Upward	M	<i>latent</i>	$\frac{W}{m^2}$	Upward	Y
<i>Qh</i>	$\frac{W}{m^2}$	Upward	M	<i>sensible</i>	$\frac{W}{m^2}$	Upward	Y
<i>Qg</i>	$\frac{W}{m^2}$	Downward	M	<i>grnd_flux</i>	$\frac{W}{m^2}$	Downward	Y
<i>Qf</i>	$\frac{W}{m^2}$	Solid to Liquid	R				U
<i>Qv</i>	$\frac{W}{m^2}$	Solid to Vapor	O				
<i>Qtau</i>	$\frac{N}{m^2}$	Downward	R				U
<i>Qa</i>	$\frac{W}{m^2}$	Downward	O	<i>advection</i>	$\frac{W}{m^2}$	Downward	Y
<i>DelSurfHeat</i>	$\frac{J}{m^2}$	Increase	R				U
<i>DelColdCont</i>	$\frac{J}{m^2}$	Increase	R	<i>deltaCC</i>	$\frac{J}{m^2}$	Increase	Y

whether the variable is mandatory(M), recommended (R), or optional(O), to comply with the standard.

The status category in Tables 2 to 9 indicates the current status of the ALMA variable in the land surface model. A yes(Y) indicates that the ALMA mandatory variable is currently output from the model. A no(N) indicates that the ALMA mandatory variable does not exist in the current output from the model and would require changes in the model code to calculate the variable. An unavailable(U) variable indicates that the variable is not part of the model output currently.

5 Final Remarks

The goal of LIS is to develop a leading edge land surface modeling and data assimilation system to support broad land surface research and application activities, to help define earth system modeling interoperability standards, and to lead the effective application of high performance computing to high-resolution, real-time earth system studies. The framework oriented design of LIS presented in this document and the use and adoption of standards such as ESMF and ALMA helps in providing a platform for land surface modelers and researchers. The extensible interfaces in LIS helps to ease the cost of development of new applications. Utilities such as tools for high performance computing and data assimilation helps researchers in rapid prototyping and development. Further, participation in the standards laid out by ESMF also helps in coupling with other earth system models. The flexible design features in LIS

Table 3: Mapping of ALMA and VIC output variables : General Water Balance

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>Snowf</i>	$\frac{kg}{m^2s}$	Downward	M				U
<i>Rainf</i>	$\frac{kg}{m^2s}$	Downward	M	<i>prec</i>	$\frac{mm}{hr}$	Downward	Y
<i>Evap</i>	$\frac{kg}{m^2s}$	Upward	M	<i>evap</i>	$\frac{mm}{hr}$	Upward	Y
<i>Qs</i>	$\frac{kg}{m^2s}$	Out of grid cell	M	<i>runoff</i>	$\frac{mm}{hr}$	Out of grid cell	Y
<i>Qrec</i>	$\frac{kg}{m^2s}$	Into grid cell	O				N
<i>Qsb</i>	$\frac{kg}{m^2s}$	Out of grid cell	M	<i>baseflow</i>	$\frac{mm}{hr}$	Out of grid cell	Y
<i>Qsm</i>	$\frac{kg}{m^2s}$	Solid to liquid	M				U
<i>Qfz</i>	$\frac{kg}{m^2s}$	Liquid to solid	M				U
<i>Qst</i>	$\frac{kg}{m^2s}$	-	R				U
<i>DelSoilMoist</i>	$\frac{kg}{m^2}$	Increase	M				
<i>DelSWE</i>	$\frac{kg}{m^2}$	Increase	M				
<i>DelSurfStor</i>	$\frac{kg}{m^2}$	Increase	M				U
<i>DelIntercept</i>	$\frac{kg}{m^2}$	Increase	R				U

Table 4: Mapping of ALMA and VIC output variables : Surface State Variables

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>SnowT</i>	<i>K</i>	-	M				U
<i>VegT</i>	<i>K</i>	-	M				U
<i>BareSoilT</i>	<i>K</i>	-	M				U
<i>AvgSurfT</i>	<i>K</i>	-	M	<i>surf_temp</i>	<i>C</i>	-	Y
<i>RadT</i>	<i>K</i>	-	M	<i>rad_temp</i>	<i>K</i>	-	Y
<i>Albedo</i>	-	-	M	<i>albedo</i>	-	-	Y
<i>SWE</i>	$\frac{kg}{m^2}$	-	M	<i>swq</i>	<i>mm</i>	-	Y
<i>SWEVeg</i>	$\frac{kg}{m^2}$	-	O	<i>snow_canopy</i>	<i>mm</i>	-	Y
<i>SurfStor</i>	$\frac{kg}{m^2}$	-	M				

Table 5: Mapping of ALMA and VIC output variables : SubSurface State Variables

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>SoilMoist</i>	$\frac{kg}{m^2}$	-	M	<i>moist</i>	<i>mm</i>	-	Y
<i>SoilTemp</i>	<i>K</i>	-	R				U
<i>SMLiqFrac</i>	-	-	O				U
<i>SMFrozFrac</i>	-	-	O				U
<i>SoilWet</i>	-	-	M				U

Table 6: Mapping of ALMA and VIC output variables : Evaporation Variables

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>PotEvap</i>	$\frac{kg}{m^2s}$	Upward	R				U
<i>ECanop</i>	$\frac{kg}{m^2s}$	Upward	R	<i>evap_canop</i>	$\frac{mm}{hr}$	Upward	Y
<i>TVeg</i>	$\frac{kg}{m^2s}$	Upward	M	<i>evap_veg</i>	$\frac{mm}{hr}$	Upward	Y
<i>Esoil</i>	$\frac{kg}{m^2s}$	Upward	M	<i>evap_bare</i>	$\frac{mm}{hr}$	Upward	Y
<i>EWater</i>	$\frac{kg}{m^2s}$	Upward	R				U
<i>RootMoist</i>	$\frac{kg}{m^2}$	-	M				N
<i>CanopInt</i>	$\frac{kg}{m^2}$	-	R	<i>Wdew</i>	$\frac{mm}{hr}$	-	Y
<i>EvapSnow</i>	$\frac{kg}{m^2}$	-	R	<i>sub_snow</i>	$\frac{mm}{hr}$	-	Y
<i>SubSnow</i>	$\frac{kg}{m^2s}$	-	R	<i>sub_canop</i>	$\frac{mm}{hr}$	-	Y
<i>SubSurf</i>	$\frac{kg}{m^2s}$	-	R				U
<i>ACond</i>	$\frac{m}{s}$	-	M				U

Table 7: Mapping of ALMA and VIC output variables : Other hydrologic Variables

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>Dis</i>	$\frac{m}{s}$	-	O				U
<i>WaterTableD</i>	<i>m</i>	-	O				N

Table 8: Mapping of ALMA and VIC output variables : Cold Season Processes

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>SnowFrac</i>	-	-	O				U
<i>RainSnowFrac</i>	-	-	O				U
<i>SnowfSnowFrac</i>	-	-	O				U
<i>IceFrac</i>	-	-	O				U
<i>IceT</i>	m	-	O				U
<i>Fdepth</i>	m	-	O				U
<i>Tdepth</i>	m	-	O				U
<i>SAlbedo</i>	-	-	R				U
<i>SnowTProf</i>	K	-	R				U
<i>SnowDepth</i>	m	-	R	<i>snow_depth</i>	<i>cm</i>	-	Y
<i>SliqFrac</i>	-	-	R				U

Table 9: Mapping of ALMA and VIC output variables : Variables to be compared with remote sensed data

ALMA variable	Units	Sign (direction of positive values)	Priority	VIC variable	Units	Sign	Status
<i>LWup</i>	$\frac{W}{m^2}$	Upward	R				

makes it unique in acting as the land component in a coupled ESMF-system.

References

- [1] ALMA. <http://www.lmd.jussieu.fr/ALMA>.
- [2] V. K. Decyk, C. D. Norton, and B. K. Szymanski. How to express C++ concepts in Fortran 90. *Scientific Programming*, 6(4):363–390, 1997.
- [3] B. Doty and J. L. Kinter. The grid analysis and display system (grads):a desktop tool for earth science visualization. In *American Geophysical Union 1993 Fall Meeting*, pages 6–10, San Francisco, CA, December 1993.
- [4] ESMF. <http://www.esmf.ucar.edu>.
- [5] LIS. <http://lis.gsfc.nasa.gov>.
- [6] P. Nowack. Architectural abstractions for frameworks. In Jan Bosch and Stuart Mitchell, editors, *Lecture Notes in Computer Science, Object-Oriented Technology, ECOOP'97 Workshops*, volume 1357, pages 116–118, Jyvaskyla, Finland, June 1997. Springer Verlag.
- [7] W. Pree. *Design Patterns for Object-Oriented Software Development*. Addison-Wesley, 1995.
- [8] P. J. Sellers, Y. Mintz, and A. Dalcher. A simple biosphere model (SiB) for use within general circulation models. *Journal of Atmospheric Science*, 43:505–531, 1986.
- [9] J. Wielgosz, B. Doty, J. Gallagher, and D. Holloway. Grads and dods. In *Seventeenth International Conference on Interactive Information and Processing*, Albuquerque, NM, January 2001.