

Milestone J Implementation Report 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 for Milestone J	09/09/03



NASA Goddard Space Flight Center,  
Greenbelt, MD 20771

## Contents

<b>1 Objective</b>	<b>3</b>
<b>2 Implementation Status</b>	<b>3</b>
<b>3 Performance Evaluation</b>	<b>4</b>

## List of Figures

1	Performance comparison of pre-ESMF 2.0 (LIS 3.0) and ESMF 2.0 (LIS-ESMF2) code for 0.25 degree runs. Each case has 3 independent runs, with the execution time shown on the y-axis. Both cases were evaluated with LIS running CLM2, GEOS base forcing, 0.25 degree resolution, global coverage, without GDS servers. . . . .	4
2	Performance comparison of pre-ESMF 2.0 (LIS 3.0) and ESMF 2.0 (LIS-ESMF2) code for 1km runs. Both versions were tested with 64 and 128 compute nodes, with the execution time shown on the y-axis. Both cases were evaluated with LIS running CLM2, GEOS base forcing, 1km resolution, global coverage, with 5 GDS servers. . . . .	6

## 1 Objective

The objective of Milestone J is to implement LIS as a partially ESMF compliant gridded component. The development of LIS as an ESMF gridded component will enable all the land models in LIS to interact with other ESMF compliant systems without having to adapt each land surface model to be ESMF compliant. The performance impact from the adoption of ESMF code will also be evaluated.

## 2 Implementation Status

ESMF is an evolving framework and having to follow a dynamic developing code posed a number of problems for adopting the code into LIS. The ESMF version 2.0 was released in July, 2004. LIS team adopted this version as the basis for Milestone J code development.

Prior to Milestone J, LIS used an older version of ESMF, primarily the time manager utility. The time management interfaces underwent significant changes and revisions for ESMF2.0 and the changes were propagated back into the LIS code. LIS team found a number of code bugs in the time management routines that still remains unresolved. For facilitating submission of the Milestone J code and testing, LIS team implemented their own bug fixes for the ESMF version 2.0.

ESMF version 2.0 included a virtual machine (VM) based abstraction of the parallel processing environment. The original release of this code contained a few problems that were subsequently fixed by the ESMF team. Due to the limited time available to the LIS team, the parallel mode of operation using MPI with the use of VM was not implemented for this prototype. Performance benchmarks at coarse resolutions were conducted on a single processor and the global 1km benchmarks did not require the use of MPI.

The design details of the LIS-ESMF prototype is explained in the updated Interoperability design document for Milestone J. This version of the code uses the ESMF time management utilities including the use of alarms. The configuration tool is used to eliminate the use of fortran namelists for specifying runtime parameters. The “forcing” and the land surface models are implemented as two separate ESMF gridded components which interact through the use of a coupler. The information exchange between these components occurs through the use of ESMF\_States and the use of ESMF data types.

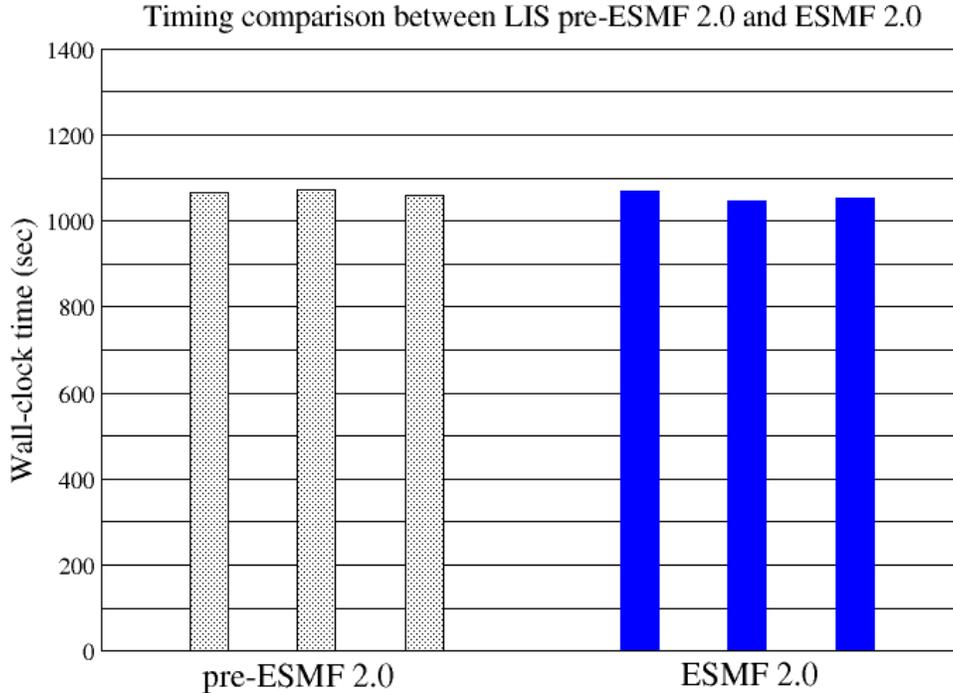


Figure 1: Performance comparison of pre-ESMF 2.0 (LIS 3.0) and ESMF 2.0 (LIS-ESMF2) code for 0.25 degree runs. Each case has 3 independent runs, with the execution time shown on the y-axis. Both cases were evaluated with LIS running CLM2, GEOS base forcing, 0.25 degree resolution, global coverage, without GDS servers.

### 3 Performance Evaluation

The performance impact of the ESMF 2.0 implementation on the LIS code is evaluated. Two representative runs were undertaken. One is the global 0.25 degree sequential run on a single processor without using any GrADS-DODS (GDS) servers. The other is the global 1km parallel run with multiple processors, with GDS servers. For both runs, we used CLM2 land surface model and GEOS base forcing. This configuration has been proved to be the worst-case scenario from past tests.

Figure 1 shows the performance comparison between pre-ESMF 2.0 (LIS 3.0) code and the LIS code implemented with ESMF 2.0 for this Milestone. This test suite was

configured as sequential runs with a single CPU, 0.25-degree global simulation with CLM2 and GEOS forcing. Each code version was tested three times independently. For both code versions and all the runs, the execution time fluctuated between 1045 and 1074 seconds, and there are no systematic performance differences discernable between the two versions.

Figure 2 shows the performance comparison between the two versions for the global 1km LIS runs. This test suite was configured as parallel runs with LIS' "pool-of-task" job management system, on 64 and 128 processors, respectively. CLM2 model and GEOS base forcing were used, with 5 GDS servers running in parallel to serve the forcing and parameter data, as we did for our Milestone G runs.

For the test results shown in Figure 2, the LIS code with ESMF2.0 implementation was slightly faster than the pre-ESMF2.0 version, for both 64 and 128-processor runs. However, by the nature of the "pool-of-task" job management system, large fluctuations in the timing are common, and based on the tests for both 0.25 degree and 1km resolutions, we would conclude that the performance impact of ESMF2.0 on LIS performance is negligible.

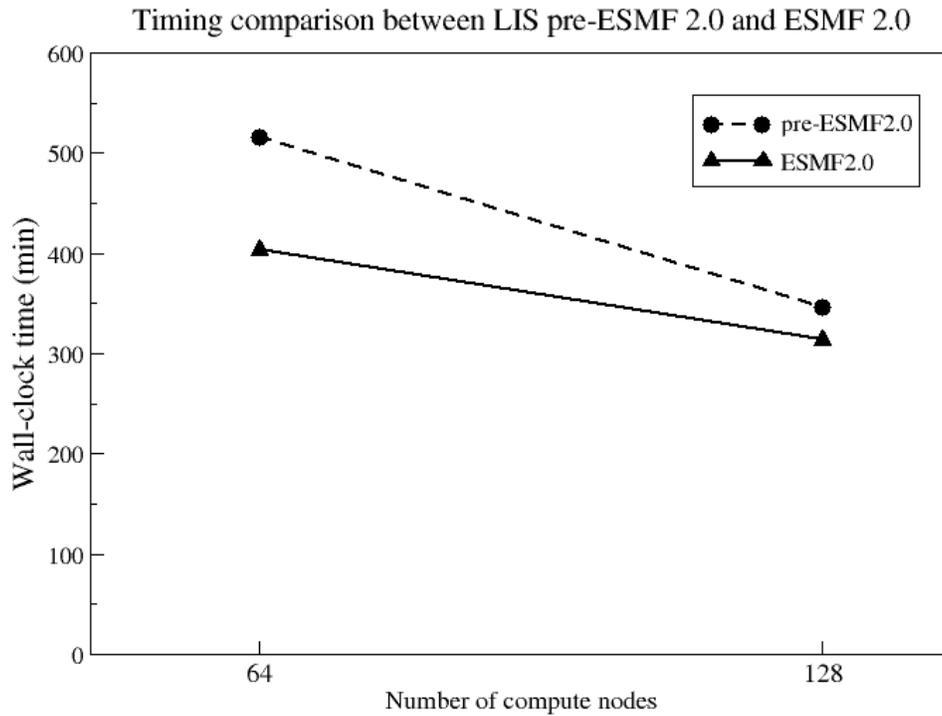


Figure 2: Performance comparison of pre-ESMF 2.0 (LIS 3.0) and ESMF 2.0 (LIS-ESMF2) code for 1km runs. Both versions were tested with 64 and 128 compute nodes, with the execution time shown on the y-axis. Both cases were evaluated with LIS running CLM2, GEOS base forcing, 1km resolution, global coverage, with 5 GDS servers.