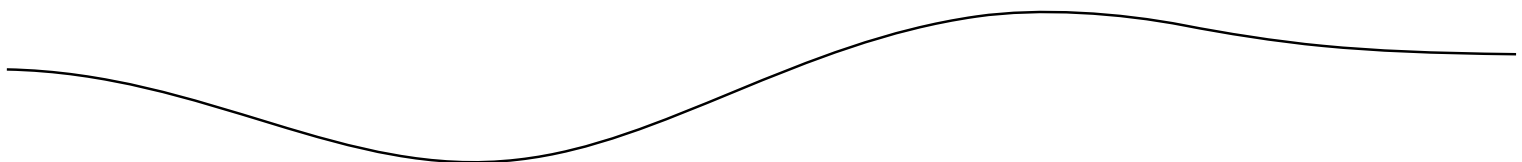


*A Flexible Unified Model Environment*

I/O Review, Design and Implementation  
v1.1

Rupert W. Ford and Graham D. Riley

©CROWN COPYRIGHT 2003, Met Office, All Rights Reserved.



# FLUME I/O Review, Design and Implementation

## v1.1

Rupert W. Ford & Graham D. Riley  
Manchester Informatics Ltd.  
The University of Manchester,  
Manchester, M13 9PL, U.K.  
email: [rupert,griley]@cs.man.ac.uk

August 2003

## 1 Introduction

This document comprises deliverables 4B, 4C, 4D, 5B, 5C, 5D, 6B, 6C and 6D. These deliverables constitute the review, design and implementation of the restart, diagnostics and archiving systems. Archiving issues are restricted to the archiving interface as the archive system itself is out of scope.

The next section gives an overview of the best practise techniques employed in restart, diagnostics and archive interfacing. This is followed by restart, diagnostics and archive interface design choices and recommendations which conform to the requirements identified in Flume document, *I/O requirements* [5] and uses the best practises defined in the review section. The final section presents any associated implementation issues.

## 2 Review

This section presents relevant work from related projects. We first separately discuss each project and then conclude with a summary of the emergent issues.

A thorough restart review paper [1] which categorises the different types of restart system may also be of interest. In summary, a coordinated checkpoint is most suited to the Met Office Flume systems requirements. The paper also provides references to un-coordinated techniques which may be useful if an attempt was made to integrate external models with their own restart system into the Flume restart system. However the authors do not recommend this, a manual approach is likely to be simpler in such cases.

### 2.1 ESMF

The Earth System Modelling Framework (ESMF) [3] requirements document [2] has some relevant discussions on I/O issues. Section 12 discusses I/O requirements. This section is limited to requirements for the reading from and writing to files from/to disk. It does not directly discuss diagnostics, archiving or restart. The consortium plan to implement an ESMF specific I/O API. They require software implementing this to be able to write to and read from many different formats including netCDF, HDF, GRIB and native binary files (with associated XML based metadata). They also require the software to be able to input and output data in parallel (to and from single or multiple files) and for the input and output to be synchronous or asynchronous. They plan to input and output “high level” objects such as fields.

Section 1.6 first mentions checkpoint and restart. In ESMF the checkpoint and restart will be coordinated by the ESMF superstructure (what Flume terms the control layer) and it is the responsibility

































The computation in transformations is expected to be relatively simple. Therefore it is important to minimise any overhead associated with using the put and get interface for such transformations. Furthermore they may also need to be memory efficient. The single model document discusses ways in which the efficiency of the put/get interface may be improved by using inline models and how the memory use and data copying may be reduced by sharing memory and running sequentially [10].

If this were still not efficient enough the transformations could also be implemented as standard sub-routines, passing data by argument (in some standard convention) and controller code could be written (either by hand, or automatically if the appropriate metadata were kept) to call these appropriately.

#### 4.4 Restart implementation

Restart must be coordinated over all models in a coupled system. In order to highlight the relevant implementation issues, the next two sections present parallel and sequential restart examples, respectively. This section then concludes with a discussion of restarts with models that have different timestep rates.

##### 4.4.1 Parallel restart example

A parallel restart example with two models, an atmosphere model and an ocean model, is presented in figure 6. As discussed in section 3.1.1, computational models have four phases, init, run, dump and shutdown.

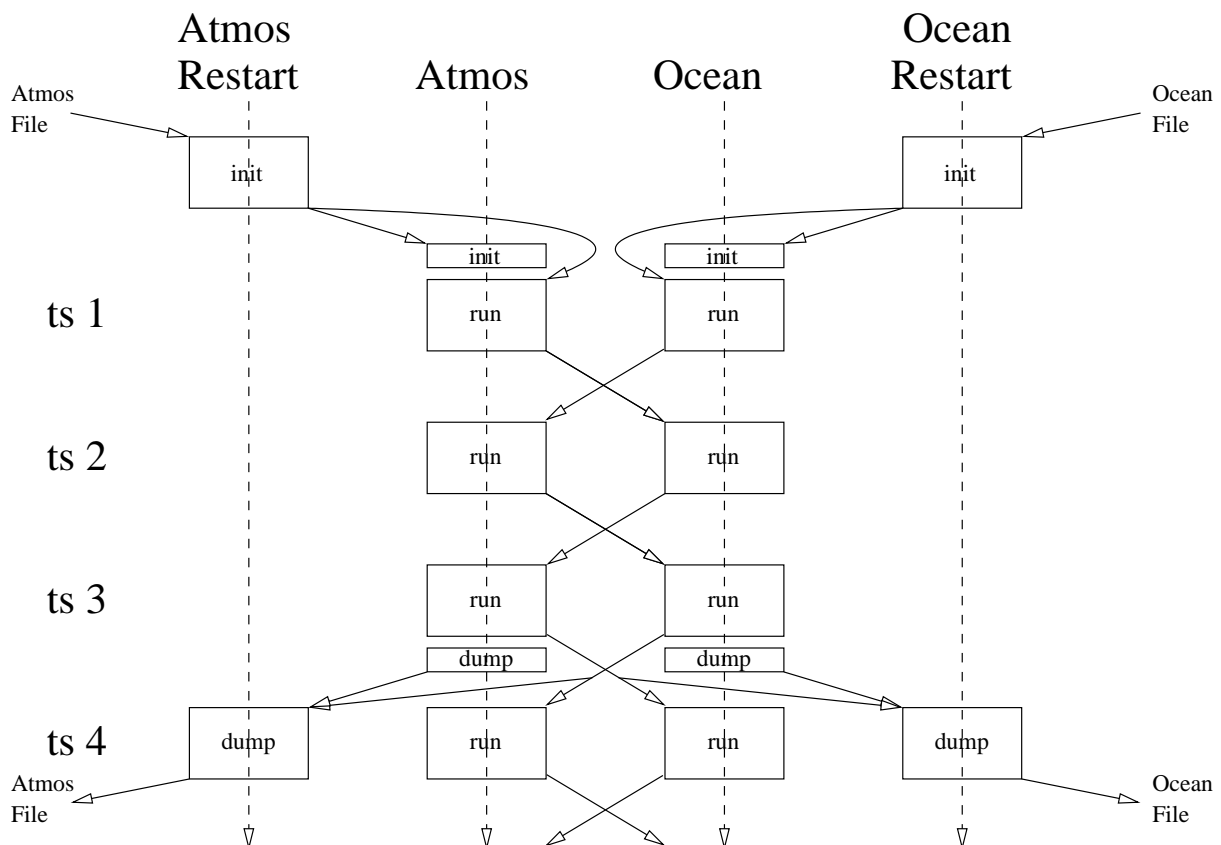


Figure 6: Parallel Restart Example

The flow of data in the system is represented by arrows. If the init and dump phases of the models are ignored then it can be seen that the models communicate with each other (are two way coupled) on each timestep. Also notice that the models continue after a restart dump has been performed.









