Requirements

The template for each requirement is as follows:

Number		Title			
Description					
Source		Priority		Build/Release	
Notes					

Note that some of the wording has changed from earlier versions of the Requirements Document, however, the content of all requirements has remained the same. As an example, 'wrapper' is now called 'component wrapper' and 'interface' is now called 'coupler'.

Software

Physics Component Architecture

Number	S1	Title	Consistent Architecture for Physics Components (PC)		
Description	All physics components in the framework will have consistent architecture. The architecture consists of: • Physics Model (PM) • Physics Component Wrapper • Coupling Interface (aka Coupler) Note that the multiplicity of component wrapper to PM mapping is 1:1, and that coupler to PM mapping is 1:many. The functionality of the partitions is further described below. The implementation of the partitions is described in the related Design Document.				
Source	Michigan	Priority	B	Build/Release	All
Notes	Done. Code had been converted in Milestone 7I, Interoperablility Prototype.				

Number	S1.0	Title	Consistent Architecture for Physics Model (PM)		
Description	Physics models will provide the following functionality: • Solution to space weather simulation in specific subdomain.				
Source	Michigan	Priority	A	Build/Release	All
Notes	Done. By definition of Physics Model.				

Number	S1.1	Title	Consistent Architecture for Component Wrappers		
Description	Component Wrappers will provide the following functionality: • Unit conversion to standard units; • Data transformation, as needed.				
Source	Michigan	Priority	C	Build/Release	Milestone 7I
Notes	Done. Code had been converted in Milestone 7I. Have created a library which can be used for these functions. Future Physics Components can use these library functions.				

Number	S1.2	Title	Consistent Language for PM and Component Wrapper		
Description	Component Wrapper will be written in same programming language as PM				
Source	Michigan	Priority	B	Build/Release	All
Notes	All PM and Component Wrappers are in F77/90.				

Number	S1.3	Title	Consistent Architecture for Coupler		
Description	Coupler will provide the following functionality: • Data exchange between PMs; • Generic data mapping between PMs, including linear interpolation and integration of grid data; • Control of execution of Processing Elements (PEs); • Fine-grained communication between PEs; • Interaction with Control Module during execution (e.g., starting/stopping execution, synchronization, allocation of PE's, etc.).				
Source	Michigan	Priority	C	Build/Release	Milestone 7I
Notes	Done. Data mapping is not generic, does what PMs require.				

Number	S2	Title	Consistent Architecture for Newly Developed Code		
Description	Newly developed PMs will have the architecture described above, prior to being included in the framework.				
Source	Michigan	Priority	B	Build/Release	All
Notes	PMs in the framework were developed in parallel with architecture above and now comply.				

Number	S3	Title	Consistent Architecture for Pre-existing Code		
Description	Pre-existing, coupled software will be decoupled to have the architecture described above, prior to being included in the framework.				
Source	Michigan	Priority	A	Build/Release	All
Notes	GM, IE, and IM preexisting coupling removed to use framework architecture.				

Number	S4	Title	Physics Component Replaceability		
Description	Physics components developed for the framework must be easily extendible and replaceable to include new components for statistical models, data assimilation models, etc. That is, physics components must supply/consume data as described in the Coupler for each physics component so that they can be substituted with alternative physics components.				
Source	Michigan	Priority	D	Build/Release	All
Notes	This has been implemented, but not thoroughly tested.				

Control Architecture

Number	S5	Title	Control of Execution on Hardware Platform		
Description	The Control module for the framework will allow control of execution of the space weather simulation. The types of control parameters include: • Choice of PMs • Run parameters for chosen PMs • Choice of parallel architecture on which to execute • Number of processors on which to execute				
Source	Michigan	Priority	B	Build/Release	All
Notes	The control module utilizes two text file, LAYOUT.in and PARAM.in which provide these parameters to the Control module.				

Number	S6	Title	Monitoring of Execution		
Description	The Control module for the framework will provide monitoring of progress of the space weather simulation. The types of progress parameters include: • Execution progress • Error logging • Other run-time information				
Source	Michigan	Priority	C	Build/Release	All
Notes	Done. Some PMs duplicate some elements of logging.				

Number	S7	Title	Control of Specific Build		
Description	The Control module for the framework will perform actions based partially upon the control parameters received. The types of actions performed include: • Selection of correct coupler, component wrapper, and PMs • Active control of PMs during execution • Synchronization of timesteps/iterations of PMs • Distribution of PEs to PMs, however each PM will load balance and distribute its own workload on its group of PEs				
Source	Michigan	Priority	C	Build/Release	All
Notes	Related to Requirement H1, Allocation of PEs to PMs. The Control module does perform these actions.				

Number	S8	Title	Functionality by User Level		
Description	The functionality of the GUI will be discretized at three levels: Beginner User, Intermediate User, and Advanced User				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	The GUI is not yet completed, but this functionality is being implemented.				

Number	S8.1	Title	Functionality for Beginner User Level		
Description	The functionality of the framework for the Beginner User is as follows: • Select and control postprocessed data • View postprocessed data				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. Description should state "The functionality of the GUI …"				

Number	S8.2	Title	Functionality for Intermediate User Level		
Description	The functionality of the framework for the Intermediate User is as follows: • All functionality of Beginner User, in addition • Create makefile • Set run parameters • Submit and monitor queue • Control and monitor run • Monitor data • Choose output file • View raw data • Control postprocessing				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. Description should state "The functionality of the GUI …"				

Number	S8.3	Title	Functionality for Advanced User Level		
Description	The functionality of the framework for the Advanced User is as follows: • All functionality of the Intermediate User, in addition • *tbd, see discussion in the section entitled' Use Case Diagram for the Space Weather Modeling Framework'*				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. Description should state "The functionality of the GUI …" Advanced users will be given more advanced monitoring and control of the system.				

Number	S9	Title	GUI for Selection of Physics Components		
Description	The GUI for the framework will allow the user to select physics components features at compile time to compose the executable for a space weather simulation run				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. This functionality is build into S8.2.				

Number	S10	Title	GUI for Selection of Run Parameters		
Description	The GUI for the framework will allow the user to select control parameters for the submission of a space weather simulation run, including model initialization parameters and restart files				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. The GUI will control creation of PARAM.in, which drives the simulation.				

Number	S11	Title	GUI for Monitoring of Job Execution		
Description	The GUI for the framework will allow the user to monitor the execution of compiled code based on the selection of physics modules				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. Monitoring via "in progress" look at run logs.				

Number	S12	Title	GUI for Viewing Results		
Description	The GUI for the framework will support the interactive viewing of plots derived from space weather simulation runs				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	In progress. Plotting will make use of batch scripts to drive IDL and Tecplot.				

Input/Output Formats

Number	S13	Title	File Formats for Post-processing		
Description	The framework will maintain the ability to read from/write to standard file formats for gridded data, including HDF-5 and documented ASCII files				
Source	Michigan	Priority	D	Build/Release	All
Notes	Old formats maintained. No HDF-5 support yet.				

Grids and Communications between Mixed-Species Grids

Number	S14	Title	Support for SWMF Grids		
Description	The framework must provide standard grid implementations for those grids most commonly used by the Space Weather modeling community, including Cartesian, block Cartesian, spherical, and block spherical, in two and three dimensions				
Source	Michigan	Priority	C	Build/Release	All
Notes	PM couplers contain implementations for all these grids.				

Number	S15	Title	Support for Operations on SWMF Grids		
Description	The framework must provide operations associated with standard grids, including transforming one grid into another. The framework must provide operations for interpolation/integration of mixed species grids as defined for the Physics Model Coupler				
Source	Michigan	Priority	C	Build/Release	All
Notes	Related to Requirement S14, Support for SWMF Grids. All grid operations required for existing PMs have been completed.				

Number	S16	Title	Support for New Grid Types		
Description	The framework must provide the ability to support new grid types and the associated re-grid transformations				
Source	Michigan	Priority	C	Build/Release	
Notes	Grid support implemented as needed by PMs. No new grid types tested.				

Hardware

Number	H1	Title	Allocation of PEs to PMs		
Description	The framework must be able to configure the number of processing elements (PEs) allocated to each physics model (PM)				
Source	Michigan	Priority	B	Build/Release	All
Notes	Related to Requirement S7, Control of Specific Build. This is done in the Control module via input from LAYOUT.in.				

Number	H2	Title	Execution on Various Hardware Platforms		
Description	The framework must be able to compile for, and execute on, each parallel architecture machine in a specified set, as defined in the Operating Environment/Hardware Platforms section of this document				
Source	Michigan	Priority	C	Build/Release	All
Notes	Related to Requirement H3.5, Performance of Customer Delivery. This has been done.				

Number	H3	Title	Performance		
Description	Conversion to framework architecture from current architecture must show eventual performance improvements for obtaining similar results with similar resolution				
Source	Michigan	Priority	A	Build/Release	Baseline
Notes	Framework performance requirements have been met.				

Number	H3.1	Title	Baseline Performance		
Description	The framework will provide at least P/2 scaling to 256 processors on the ESS testbed and to as many nodes available on the Beowulf				
Source	Michigan	Priority	A	Build/Release	Baseline
Notes	Completed in Milestone 3E.				

Number	H3.2	Title	Performance of First Code Improvement		
Description	The framework will provide an improvement over baseline of 5X with same resolution on the ESS testbed and the ESS Linux cluster				
Source	Michigan	Priority	B	Build/Release	Milestone 6F
Notes	Completed in Milestone 6F.				

Number	H3.3	Title	Performance of Second Code Improvement		
Description	The framework will provide an improvement over baseline of 15X with same resolution on the ESS testbed and the ESS Linux cluster. Measure accuracy against a fully explicit run				
Source	Michigan	Priority	C	Build/Release	Milestone 9G
Notes	Completed in Milestone 9G.				

Number	H3.4	Title	Performance of Full Model		
Description	The framework will support faster than real-time full interoperation of PMs in framework using at least 256 nodes on Teraflops Scalable Testbed				
Source	Michigan	Priority	C	Build/Release	Milestone 10J
Notes	Completed in Milestone 10J.				

Number	H3.5	Title	Performance of Customer Delivery		
Description	The framework will be portable to alternative architecture hardware, such as the ESS Linux cluster and machines used by CCMC and NOAA SEC, and demonstrate framework's operation				
Source	Michigan	Priority	C	Build/Release	Milestone 11K
Notes	Related to Requirement H2, Execution on Various Hardware Platforms. The SWMF has been successfully installed and is running at the CCMC.				

Non-Functional Requirements

Security

See Software Requirements S8, S8.1, S8.2, and S8.3.

Portability

See Hardware Requirement H2, H3.5.

Extensibility

See Software Requirements S2, S3, and S4.

Maintainability

Number	M1	Title	Source Code Documentation		
Description	The developers of the framework will provide appropriate documentation to maintain the framework: • the framework will provide a User Manual • the framework will provide a Maintenance Manual • the source code in the framework will be commented				
Source	Michigan	Priority	B	Build/Release	All
Notes	This documentation is included with the framework code.				

Number	M2	Title	Method of Source Code Delivery for Milestones		
Description	The source code will be delivered to the evaluation community using current, standard formats, such as TAR				
Source	Michigan	Priority	A	Build/Release	All
Notes	A tar distribution is available on the web.				

Number	M3	Title	Method of Source Code Delivery to Users		
Description	The source code will be available to the user community using current, standard techniques, such as SourceForge or FTP				
Source	Michigan	Priority	A	Build/Release	All
Notes	Code available through http/ftp on web page only.				

Leveraging Requirements from Other Sources

Earth System Modeling Framework

Number	E1	Title	ESMF Requirements, General		
Description	In coordination with Earth System Modeling Framework, design framework specifications, based upon analysis of requirements [4]				
Source	Michigan	Priority	E	Build/Release	N/A
Notes	Due to differing timeframes for milestone completion, very little coordination with the ESMF.				

Number	E1.1	Title	Language Interoperability Policy		
Description	Our language interoperability policy is based on the following document: • Earth System Modeling Framework: Implementation report, NASA High Performance computing and Communications Program, Earth and Space Sciences Project, UCAR, Boulder, Colorado, 2002				
Source	Michigan	Priority	D	Build/Release	All
Notes	Only F77/90 code has been used. No interoperability issues exist now.				

Operational Scenarios

The Operational Scenarios for the SWMF are described from the perspective of the Graphical User Interface (GUI) using the Unified Modeling Language (UML), v1.3 [2, 3]. Specifically, a Use Case Diagram is used to describe the long-term vision for the GUI. Several Sequence Diagrams are used to describe possible sequences of execution in the system.

Use Case Diagram Notation Description

The following is a description of Use Case Diagram notation. An *actor* is shown as a stick figure. The stick figure often represents a human interacting with the system, however, the actor is not necessarily human and may be an external hardware or software system. Actors may be related to each other through *inheritance*, as indicated by a line with a triangle at one end. The actor at the end of the relationship without the triangle inherits the properties of the actor at the end with the triangle. In addition, the inheriting actor typically has additional specialization(s). In **Figure 1**, the Intermediate User actor inherits from the Beginner User actor.

A *use case* is shown as an oval with a label. A use case is a collection of related behaviors. As an example, two use cases from **Figure 1** are Set Run Params and Submit & Monitor Queue. A *system boundary* is used to group families of use cases and/or actors, such as Create Executable and Queue being grouped together into the Operating System boundary. Finally, *associations* are shown with solid, connecting lines between actors and use cases or between two use cases. As an example, the Intermediate User actor is associated with the Control & Monitor Run use case.

Use Case Diagram for the Space Weather Modeling Framework

As mentioned, **Figure 1** is the Use Case Diagram for the SWMF. As such, the diagram offers a starting point for a discussion regarding the system functionality and operational scenarios. The figure is meant to be inclusive of eventual framework functionality, and additional details about specific use cases may be found in appropriate versions of the framework Design Document. That is, **Figure 1** shows the eventual functionality of the overall system, and not all functionality indicated in the diagram is implemented as part of Phase 1. As an example, the Control and Monitor Run use case will be thoroughly described in the Design Document covering the use cases implementation.

In **Figure 1**, there are three user actors: beginner user, intermediate user, and advanced user. Each user type inherits from the previous user type. The functionality that is inherited is indicated by which use cases are associated with each actor. Note that there is currently no discretization between Intermediate and Advanced User. In future versions of the framework, several use cases may migrate from Intermediate to Advanced User.

The use cases are as follows:

- Create Makefile: create a conditional makefile that compiles, links, and builds the executable code requested by the user. That is, the Create Makefile use case creates an executable from some subset of the physics modules;
- Set Run Params: set the parameters for a science run using a specific executable on a specific target machine;
- Submit & Monitor Queue: submit a job to the queue for later execution. Monitor the queue to observe the status of the submitted job;

- Control & Monitor Run: observe the job as it executes. Make any modifications to the submitted job at specific checkpoints;
- Monitor Data: observe the data produced by a specific science run during execution;
- Choose OutFile: choose from several output data files that have been produced;
- View Raw Data: observe the raw data produced by a specific science run after execution is complete;
- Control Postproc: filter data, as appropriate, for specific postprocessors;
- Select & Control Postproc Data: choose from several postprocessed data files and control parameters related to viewing data;
- View Postproc Data: observe data that has been postprocessed in postprocessed form.

The associations between users and use cases are indicated by solid lines connecting the two, and are not individually described here for brevity.

In addition to the GUI system, there is also an Operating System (OS) boundary in **Figure 1**. The OS contains an actor that creates the executable and the queue for maintaining submitted jobs. There is a Physics Module (PM) subsystem. The PM contains an actor that is the physics model executable. That is, it contains the actual physics software for executing science runs.

Finally, there exists the Output Data Processing subsystem that contains three actors: Raw Output Data, Postprocessors, and Graphics Software. The Raw Output Data actor is the data produced by the physics runs in it raw, unfiltered form. The Postprocessor actor(s) are tools that filter the data and make it usable. Finally, the Graphics Software actor is a set of tools for viewing the data in a usable, graphic format.

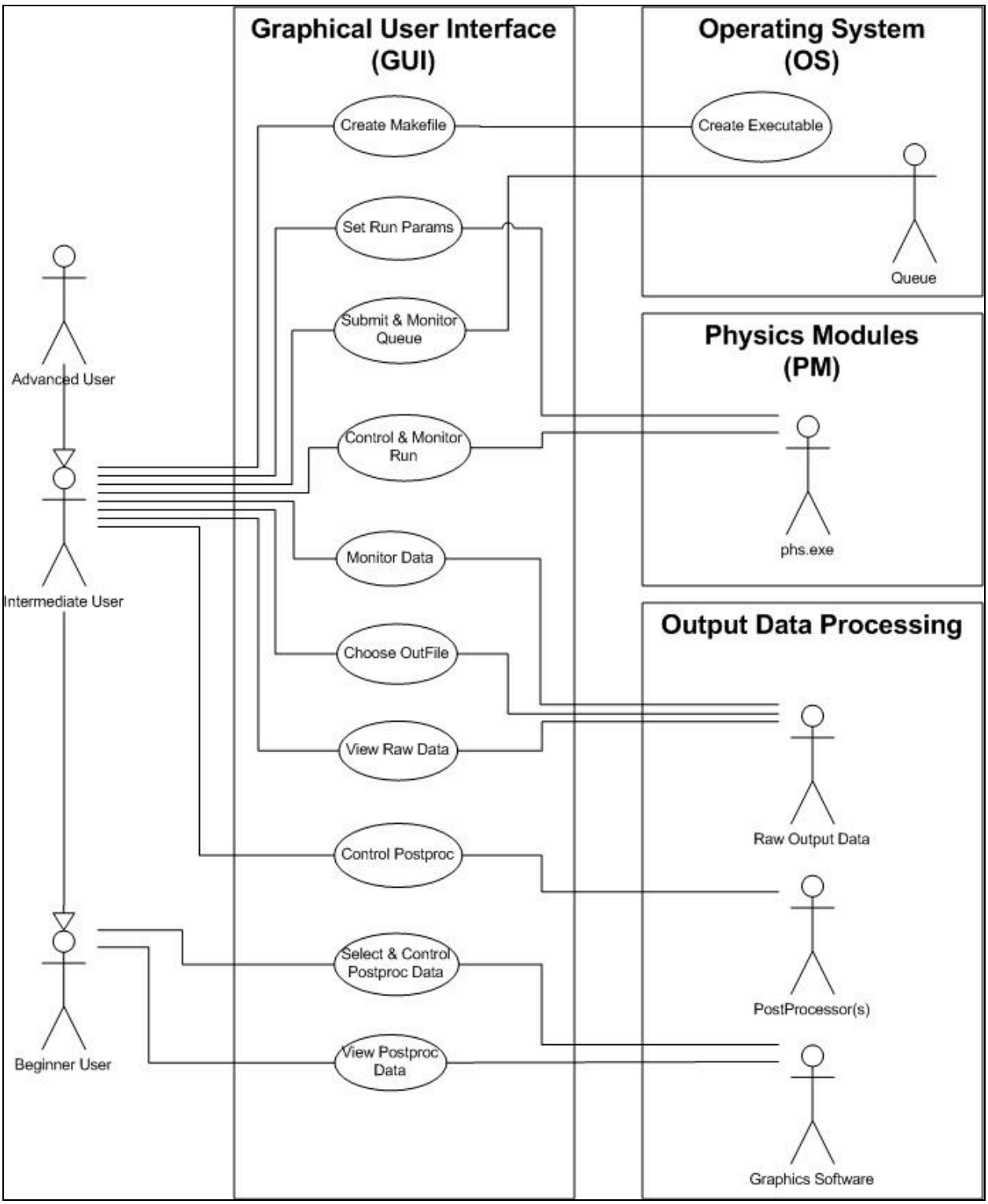

Figure 1. Use Case Diagram for web-based Graphical User Interface.