INL/EXT-05-00107

The Application Programming Interface For The PVMEXEC Program And Associated Code Coupling System

Walter L. Weaver III

March 2005

The INL is a U.S. Department of Energy National Laboratory
operated by Battelle Energy Alliance

INL/EXT-05-00107

The Application Programming Interface for the PVMEXEC Program and Associated Code Coupling System

Walter L. Weaver III

March 2005

Idaho National Laboratory
Idaho Falls, Idaho 83415

Prepared Under DOE/NE Idaho Operations Office
Contract No. DE-AC07-05ID14517

ABSTRACT

This report describes the Application Programming Interface for the PVMEXEC program and the code coupling systems that it implements. The information in the report is intended for programmers wanting to add a new code into the coupling system.

CONTENTS

1 Introduction

The purpose of this report is to document the Application Programming Interface (API) of the PVMEXEC computer program. The PVMEXEC program facilitates the coupling of several different simulation codes for the unified analysis of a system. The current application of the PVMEXEC code is for the simulation of nuclear reactor power plants. The PVMEXEC code is a general purpose code in the sense that it can be used to couple any set of simulation codes for any purpose. Any code that implements the coupling API can be used in a coupled simulation. The purpose of this report is to describe how a code must be modified to allow it to be used in a coupled simulation being controlled by the PVMEXEC code.

1.1 Background

Several codes have been developed for the analysis of nuclear reactor power plants. These codes focus on the simulation of the reactor core, the primary cooling system, and the secondary cooling systems. Other codes have been developed for the analysis of the containment systems or for the analysis of severe accidents. These codes focus on the thermal-hydraulic behavior of the systems being simulated. Still other codes have been developed for a detailed analysis of the neutronic behavior of the reactor code or of the power plant control and safety systems. Depending on the primary focus of a system simulation code, e.g., the thermal-hydraulic behavior of the system, it will usually have simple models for other aspects of the simulation, i.e. point kineitcs for the kinetics, and trips and simple control models for the control and safety systems.

The PVMEXEC code and the coupling system that it implements was developed so that the most appropriate simulation code can be used to model that part of the reactor system for which it was developed, e.g., a thermal-hydraulic code for the fluid behavior in the reactor core (such as COBRA[1]), a different thermal-hydraulics code to the simulation of the primary and secondary cooling systems [such as RELAP5-3D©[2], TRAC-PWR[3], or TRACE(TRAC-M)[4]], a kinetics code for the simulation of the neutronic behavior of the reactor core (such as NESTLE[5] or PARCS[6]), and a containment simulation code (such as CONTAIN[7] or MELCOR[8]) for the simulation of the reactor containment and auxiliary systems. Each of the coupled codes needs data from the other codes in a coupled simulation. For example, the code performing the thermal hydraulic analysis of the reactor core needs to know the power generation rate in the fuel rods in the core. This data is obtained from the reactor kinetics code. The kinetics code in turn needs to know the state of the fluid in the core region of the reactor system. This data is obtained from the thermal-hydraulics code that is modeling the behavior of the coolant in the reactor core. These data are exchanged using a message passing system. If an analyst wishes to use a particular code as part of the coupled simulation, it must be modified to implement the message passing system. This report describes the functional requirements of the message passing (or coupling) interface. These requirements are embodied in the messages that are exchanged between the PVMEXEC program and the simulation codes. The Parallel Virtual Machine[9] (PVM) software library is used to accomplish the physical transfer of data between the PVMEXEC program and the simulation codes and between the simulation codes.

1.2 General Principles

This section describes the general principles used in the design of the coupling system as implemented by the PVMEXEC program. First, the coupling system as embodied by the PVMEXEC program and the modified simulation codes is a batch simulation system. This means that once a coupled simulation has been initiated, it should proceed to completion without user intervention, although user interaction with the simulation codes is not precluded. The simulation is expected to terminate at the end of the simulation or fail gracefully if any of the simulation codes encounter difficulties from which it cannot recover. Second, any message sent from or received by a simulation code must be acknowledged to the sender or receiver. This handshaking helps keep the simulation codes synchronized. Thirdly, each message has a wait time associated with it so that if a simulation code is waiting to receive a message, it will not wait indefinitely thereby never terminating gracefully. Finally, the data items (or macros that are expanded to a list of data items) that are to be exchanged between any two simulation codes are listed twice in the input file of the PVMEXEC program, once in the specification of the message that sends the data items and once in the specification of the message that is to be received. This is needed because a data item represents data that is located in a single location in the complete system model but is represented (stored) in two different coupled models of the system in a coupled simulation. Each of the couled simulation models needs to know what the data item represents in it's simulation model.

1.3 Description of PVM

This section provides a general description of the PVM message passing methodology and is not intended to replace the detailed PVM documentation[9]. The PVM software implements a general message passing methodology. The entities (tasks in PVM parlance) passing messages back and forth may be executing on the same host or may be communicating with each other over a network. The host's being used may be of same type of architecture (all PC's or all DEC Alpha's) or may be different architectures (a PC communicating with a DEC Alpha). The PVM software handles all of the details of the physical exchange of the data so that the programmer may concentrate on the data being exchanged to implement his computational algorithm. The PVM software is implemeted in the C programming language but has wrapper functions that implement a FORTRAN interface so that the C function or subroutine may be called using FORTRAN semantics. Each computational process in a coupled simulation is assigned a unique task identification number (a tid) and each message is assigned a message identification number (a message tag) so that a specific message may be directed to a specific process. Messages are sent by initializing a buffer (i.e., reserving storage for the data to be sent), loading the data into the buffer, and then sending the messsage by specifying the message tag for the message and the task identifier for the intended receiver of the message. The receiver is expected to check for messages by specifying the message tag and the task identifier of the sender to the PVM software. Once the receiver is notifed by the PVM software that the message has been received, the receiver is expected to unload the data from the PVM buffer into his own computational database.

1.4 Responsibilities of the PVMEXEC Program

The PVMEXEC code has several responsibilities in performing a coupled computation. The PVMEXEC code must start up each of the simulation codes. It tells each of the simulation codes what data they are to send and to whom to send the data. It specifies what data to receive and from whom to receive the data. It coordinates the production of printable output, plottable output, and restart data, so that the output of the several simulation codes occur at the same simulation times. It may coordinate the time step sizes used by the several simulation codes depending upon the type of coupling being used for the simulation. Finally, it coordinates the graceful termination of a simulation, both for normal and abnormal terminations.

2 Taxonomy of Coupling

This section describes the different types of coupling that can be used for coupled simulations. There are two dimensions in the classification of coupling type. The first dimension specifies the way in which the time steps are chosen by the simulation codes. There are two types of coupling within this dimension, synchronous and asynchronous. In synchronous coupling, the simulation codes all use the same time step sizes and perform the same advancement at the same time. In asynchronous coupling, the simulation codes are free to choose their own time step sizes subject to the restrictions imposed by the requirement to produce restart data at the same simulation times and to exchange data at fixed intervals during the simulation. The second dimension of coupling classification is the type of mathematical solution algorithm being used by the coupled simulation. Solution algorithms are classified as explicit, semi-implicit, and fully implicit. Some types of solution algorithms are inherently synchronous like the semi-implicit and fully implicit solution algorithms while others such as an explicit solution algorithm may be either synchronous or asynchronous. The second dimension of coupling type can be further subdivided based on the part of the system being modelled, i.e., thermal-hydraulic coupling for modeling of the fluid portions of the system, kinetics coupling for the simulation of the power production in the system, and control systems coupling for the simulation of the reactor control and protection systems. Therefore there may be explicit, semi-implicit, or fully implicit thermal-hydraulic coupling; explicit, semi-implicit, or fully implicit kinetics coupling, etc.

Explicit coupling is characterized by data that has been computed by one simulation code in a particular time advancement being used by another simulation code in subsequent advancements. One code may compute the pressure in a volume and the other code will use that pressure as a boundary condition during its time advancements. Explicit coupling means that the value remains constant (is explicitly known) while being used by the other code. Semi-implicit coupling is characterized by data being computed by one simulation code during an advancement and that data being used by another simulation code during the same advancement. The data being used by both codes also allows for the effect of changes to the state of that portion of the system being simulated by one simulation code to be felt in that portion of the system being simulated by the other simulation code during the same time advancement. One further wrinkle in explicit coupling concerns the data at the coupling locations between the two simulation domains. In thermal-hydraulic coupling, pressure boundary conditions may be exchanged between the two codes and then both codes compute the flow rates of mass and energy across the

boundaries. These flow computations may be done in parallel since each code has all of the information needed to advance its solution, hence the name parallel explicit coupling. However, there is nothing to ensure that the flow rates computed by the two simulation codes at the same physical location in the simulated system will have the same value. This implies that the amount of mass (or energy) that leaves one computational domain will not be the same as the amount of mass (or energy) that enters the adjacent computational domain. An alternate way of explicitly coupling two computational domains is for only one of the codes to compute the flow rates between the domains and for the other code to use the flows rates as its boundary condition. This implies that the code computing the flow rates between the domains must compute the flow rate first and the other code must wait for the flow rates to be computed before it can advance its solution. This imposes an order on the computations; they must be done is sequence, hence the name sequential explicit coupling. The coupling is still explicit, because data from the one code is held constant while being used by the other code.

The PVMEXEC code implements a subset of coupling types. Within the asynchronous dimension, only explicit thermal-hydraulic coupling has been implemented. Within the synchronous dimension, explicit and semi-implicit thermal-hydraulic coupling, explicit kinetics coupling, and explicit control systems coupling have beem implemented. Within explicit thermal-hydraulic coupling, both parallel and sequential coupling are allowed. Asynchronous explicit kinetics coupling and asynchronous explicit control systems coupling are planned to be implemented in the future. Strictly speaking, the kinetics and control systems coupling are sequential explicit synchronous coupling. The thermal-hydraulic advancement is performed first, then the kinetics model is advanced during the same time step using the results of the thermal-hydraulic advancement. The same is true for the control systems coupling. The thermal-hydraulic and kinetics models are advanced, then the control systems model is advanced during the same time step using the results of the thermal-hydraulic and kinetics model advancement. **Table 2.0-1** shows the types of coupling that have been implemented by the PVMEXEC program.

Table 2.0-1 Types of Coupling

Type	Thermal-Hydraulic			Explicit Kinetic	Explicit Control System
	Explicit		Semi-Implicit		
	Parallel	Sequential			
Synchronous	X	X	X	X	X
Asynchronous	X	X	NA	not implemented	not implemented

3 PVMEXEC Application Programming Interface

The execution of a simulation code in the context of a coupled simulation under the control of the PVMEXEC program consists of two phases of computation. The first phase of a coupled computation consists of code startup, code initialization, and coupling initialization. This phase of a coupled computation is the same for all modes of coupling. The second phase of a coupled computation consists of

the time advancements and is different for the several types of coupling. The common startup and initialization phases of a coupled computation will be described first and the several types of time advancements will be described individually. There are similarities between the several different types of time advancements, particularly in the production of output, that will only be described once at the beginning of the sections of this report that describe the time advancements.

3.1 Initialization Phase of Coupled Simulation

A user performs a coupled simulation by executing the PVMEXEC code. The PVMEXEC code reads its input file to determine which hosts are to be used to perform the coupled computation and to determine which codes are to be executed on each of the hosts. The PVMEXEC program initiates the execution of the simulation codes on the designated hosts using functions in the PVM software library. Each individual code in the coupled simulation is executed as specified in the input file of the PVMEXEC program. The specification in the PVMEXEC input file contains the name of the executable file for the simulation code along with any command line parameters that are to be used by the simulation code. The PVMEXEC program adds an additional command line parameter to the end of the command line contained in the input file. This parameter is '-PVM'. This command line parameter has been added so that the simulation code may check to see if it has been executed in coupled mode. Some codes are unable to examine the command line and ignore it. There is another way that a simulation code can determine if it has been executed under the control of the PVMEXEC code. It can use a PVM function to determine if it has a 'parent' process. In PVM parlance, a 'parent - child' relationship exists between the codes in a coupled simulation. The PVMEXEC code (or its surrogate) is the parent process and the individual simulation codes are child processes. In either case, it is the first responsibility of a simulation code to determine whether or not it is being executed in coupled mode. If the code is being executed in coupled mode, the simulation code needs to determine the task identifier of its parent and its own task identifier using calls to PVM functions ('pvmfparent' and 'pvmfmytid', respectively). These functions are the FORTRAN wrappers around the underlying C routine. This report will use the FORTRAN interface to the PVM software functions when specifying function names. The C interface can be found in the PVM documentation[9]. Once a simulation code knows the task identifier of its parent, it must listen to receive a message with messsage tag 7001 from its parent. This message contains a single integer, the task identifier of the PVMEXEC process. The simulation code must acknowledge the receipt of the 7001 message by sending its parent a message with message tag 7001 containing a single integer, the task identification number for the simulation code. Appendix A contains a list of all of the messages sent or received by the PVMEXEC code listed in numerical order along with a description of the data items in each message. Appendix B contains another list of all of the messages sent or received by the PVMEXEC code listed in the order in which they are sent by the PVMEXEC program.

There are several housekeeping details that must be explained before the message exchange described above can be implemented in a new simulation code, new in the coupling sense. These details are PVM and implementation specific. First, the sender of a message must specify the format for the message. There are two formats for data exchange between codes, raw and xdr. Raw format can be used for data exchanges between hosts of the same architecture (PC to PC) and xdr format must be used for data

exchanges between hosts of different architectures (PC to DEC Alpha). Xdr is the default data format for data exchanges. Second, PVM has established several data types that should be used to exchange data between processes. The PVMEXEC program and the coupling methodology uses the PVM data types INTEGER4 for integer data (four byte integers), the PVM data type REAL8 for floating point data (eight byte real numbers), and the PVM data type STRING for character strings. The receiver of a message does not need to worry about the data format because the sender specifies the format and the PVM routines that receive the data manage the conversion to the receivers format. However, the receiver needs to know the number of data items and their types when fetching the data out of the receive buffer. The PVM software library contains macro definitions in a header that should be 'included' in all of the routines that access the PVM library routines. The first message with message tag 7001 is sent in default data format by the PVMEXEC program, and contains a single four byte integer (PVM data type INTEGER4). The message acknowledgement should be sent to the parent in default format and also contain a single four byte integer.

The next bit of housekeeping is to ask the PVM software to send the requestor a message with message tag 10001 if the PVMEXEC program terminates during a simulation. This is accomplished by a call to the PVM function 'pvmfnotify'. The PVM function 'pvmfnotify' can send messages for many types of events. The event that we need to know about is if the PVMEXEC program has terminated. This allows the simulation code to shut itself down if there is a problem with the PVMEXEC program. The simulation code should periodically query the PVM system periodically to see if a message with 10001 has been received so that it can shutdown gracefully if such a message is received. The simulation code should also periodically check to see if a message with message tag 10003 has been received. This message is sent to all simulation codes in the coupled simulation by the PVMEXEC program if one of the simulation codes terminates abnormally, e.g., fails with a floating point exception, etc.. Finally, the simulation code is expected to send a message to the PVMEXEC program with message tag 10002 if the wait time is exceeded while waiting to receive a message from another simulation code, either to receive data from the other simulation code or to receive an acknowledgement from the other simulation code that the data it sent has been received by the other simulation code. A default wait time of at least 60.0 sec should be used by each simulation code at the beginning of the simulation. A global wait time will be defined by the PVMEXEC program in one of the initial messages to the simulation codes. The PVMEXEC code will broadcast (send to all child processes) a 10003 message if it receives a 10002 message from any of the simulation codes. These messages ensure that the coupled simulation will be terminated if any of the simulation codes or the PVMEXEC program fail. This requirement of notifiying the PVMEXEC program in the event of a timeout (in a message with message tag 1002) and of periodically checking for messages with message tag 10001 and 10003 imply that the simulation code should check for messages 10001 and 10003 while waiting for other messages. This will help keep the simulation codes synchronized and avoid a deadlock, e.g. waiting for a message from a simulation code that has failed.

Once the initial data exchange with the parent has been acomplished and the simulation code has received the task identifier of the PVMEXEC code, the simulation code can begin to process its input file.

3.1.1 Restart Control and Simulation Start Time

It is assumed that a coupled simulation run is either the first execution of a simulation or is the extension of a previously performed simulation (i.e., a restart). Messages must be received from the PVMEXEC program early in the processing of input data that specify whether this is a new simualtion or a restart run. The first control message from the PVMEXEC program is contained in a message with message tag one. This message contains the following data. The first data item is an integer that specifies if the process is synchronously or asynchronously coupled. This is followed by a real number specifying the start time of the simulation. Start times are normally zero for an initial run and the restart time for a restart but the start time can be reset for both initial runs and restart runs using this value. The next data item in the message is the restart time for a restart run. This time specifies where a previous run is to be resumed. A value of zero specifies that this is a new simulation and not a restart run. A value of -1.0 denotes that the restart is to be performed from the last set of restart data in the restart file and a value greater than zero specifies the simuation time on the restart file from which to initialize this run. The default values for both the start time and the restart time are zero denoting a new run starting from time zero. This is followed by an integer that is RELAP5-3D© specific and can be ignored by other simulation codes. Next comes an integer that specifies the number of characters in a PVM string variable followed by the string of the specified length. If the simulation is a new simulation, the codes are expected to write the character string to the beginning of their restart file. If the run is a restart run, the simulation codes are expected to compare the string in the message to the string in their restart file. The simulation codes are expected to acknowledge this message from the PVMEXEC program with a message with message tag one containing the following data. The first data item is a integer restart status flag and the second data item is a real number specifying the start time determined by the simulation code. A restart status other than zero denotes an error terminating the coupled computation. A value of one denotes that the code cannot perform a restart at the time specified by the PVMEXEC program because no restart data exists for the time specified. A value of two means that the string found on the restart file does not match the string specified by the PVMEXEC program. If a task returns a restart status of two, the message from the task must contain two additional variables, an integer specifing the length of a string variable and a string variable containing the string found on the restart file. This string is printed on the printed output file of the PVMEXEC program and can be used to detemine why the incorrect restart file has been specified to the task.

If the restart status is zero, either because the simulation is a new run or because the correct restart file and restart record have been found, the start times reported by the several codes are compared to determine if they are the same. For initial runs, the simulation code should set its start time to the start time contained in the message from the PVMEXEC program and return the same value. For restart runs, the start time returned by the simulation codes depends upon how the restart time and start time are specified in the input file of the PVMEXEC program. The restart time can be specified as -1.0 or as a positive real value and the start time can be set to a positive value. If the restart time is specified in the input file of the PVMEXEC program, the start time is set to the restart time if the start time is not specified in the input file of the PVMEXEC program, otherwise the start time sent to the simulation codes is the start time specified in the input file. This leads to several different combinations of restart time and start time. If the restart time is specified as -1.0 and the start time is not specified, the start time is set to -1.0. These values denote

that the restart is to occur from the last set of restart data in the restart file and the start time is to be set to the restart time, whatever value the simulation code has found in its restart file. If the restart time is specified as -1.0 and the start time is specified in the input file of the PVMEXEC program, the simulation code is to restart from the last set of restart data in the restart file and to replace the simulation time found in the last set of restart data with the value specified by the PVMEXEC program. If the restart time is a value greater than zero and the start time in not input in the input file of the PVMEXEC program, the simulation code is to restart from the restart data in the restart file at the specified time and is to use the value of the restart time as the start time (they will be the same in the message from the PVMEXEC program). Finally, if the restart time is a number greater than zero and the start time is specified in the input file of the PVMEXEC program, the simulation code is to restart from the data in the restart file at the specified restart time and is to replace the simulation time found on the restart file with the value contain in the message from the PVMEXEC program. The PMVEXEC code compares the start times reported by all of the simulation codes and then sends another message with message tag one. This message contains a single integer, the global restart status. A value of zero denotes that there are no errors and that the computations should be continued and a value of one denotes an error condition and that the codes should terminate gracefully. An acknowledgement containing the task identifier must be sent to the PVMEXEC program to acknowledge the receipt of the global restart status. The restart status flag and the start time returned by the simulation codes are used to ensure that start times are the same if the user specifies the restart time as -1.0 and does not input the value of the start time in the input file of the PVMEXEC program.

If there are no restart errors, the next message from the PVMEXEC program is a message with message tag two. This message contains a single integer, the number of threads that this task should use if it can be executed in parallel. The message must be acknowledged with a message with message tag two containing the task identifier of the receiving task even if the simulation code cannot be executed in parallel. RELAP5-3D© may be executed using multiple threads and this message overrides the default number of threads that RELAP5-3D© will use for its execution.

3.1.2 Coupling Data Specification

After reading and processing its input data, the simulation codes must receive messages from the PVMEXEC program that specify the data that they are to exchange with the other simulation codes. The first data specification message has message tag 1000. This message contains two data items, a real number specifying a global wait time that is to replace the default wait time of 60.0 seconds and an integer (zero or one) that can be used to activate debugging output in the simulation code. The global wait time is used for the receiving of all messages unless specifically overwritten for individual messages. The debugging flag is sent to all coupled processes and can be set in the input file for the PVMEXEC program. This message is to be acknowledged by a message to the PVMEXEC program with the same message tag (message tag 1000) and must contain a single integer, the value of the task identifier of the simulation code.

Next come a series of messages that specify the data that is to be sent or received for the different types of coupling. Message 1001 specifies the number of messages that are to be sent or received by the simulation code for explicit thermal-hydraulic coupling. Message 1001 contains two integers, the number of messages to be sent for explicit thermal-hydraulic coupling and the number of messages that are to be received for explicit thermal-hydraulic coupling. This message is acknowledged by sending a message to the PVMEXEC code with message tag 1001 that contains a single integer, where the integer should be the task identifier of the simulation code. Although strictly speaking, any integer may be used in the acknowledgement to this and all other received message except the message with message tag 1000, it is recommended that the tid of the task receiving the message be used. The number of messages to be sent or received for explicit thermal-hydraulic coupling are then used to receive multiple messages with message tags 1002 and 1003, message tag 1002 for the send messages, and message tag 1003 for the receive messages. Each of these messages are to be acknowledged to the PVMEXEC program using message tags 1002 and 1003, respectively, where each acknowledgement message contains a single integer, the task identifier of the simulation code as recommended.

Messages with message tag 1002 contain the following data. The first data item is an integer that specifies the message tag to be used when sending the data. This is followed by another integer that specifies the task identifier of the task to which the data is to be sent. Next comes an integer that specifies the temporal type of explicit coupling, a value of zero means synchronous explicit coupling and a value of one means asynchronous explicit coupling. Next comes another integer that specifies the subtype of explicit coupling being used to couple this simulation code to the other simulation code. A value of minus one denotes parallel explicit coupling and a values of zero and one denote the leader and follower tasks in sequential explicit coupling respectively. This is followed by a real number that specifies the wait time for the receipt of the acknowledgement that the message has been received by the process to which it has been sent. Next comes an integer that specifies the number of characters in the string that names the data items to be sent in the message. This is followed by a string of the specified length. The string contains the identifiers of the data that are to be sent to the other process. The identifiers are simulation code specific and the simulation code needs to understand the identifiers as contained in the string. Only the simulation code needs to understand the meaning of the identifiers because it is the only entity that must associate the identifier with the location in memory where the data item is stored. The simulation code uses the identifiers to access the data and load the values into the send buffer. Examples of identifiers are the pairs of values used by the RELAP5-3D© code. These pairs consist of the plot/print name of a variable and the volume or junction identifier of the RELAP5-3D© component containing the specified variable. RELAP5-3D© also recognizes macro names that are expanded to a list of variables. Appendix C lists the macros that RELAP5-3D© understands. Each data item to be exchanged between two processes is specified twice, once for a sending process and once for the corresponding receiving task. Each task needs to know what the descriptor in a message means, the sending code needs to know how to access the value of the data item from its database to load it into the send buffer and the receiving task needs to know where to store the value received into its database after retrieving the data item from the receive buffer. If macros are used to specify a list of variables for RELAP5-3D©, the other code needs to either implement the same macro definition or the user must specify each variable individually in the input file of the PVMEXEC

program. The message specifications contained in messages with message tag 1002 and 1003 do not specify the format in which the messages are to be sent for the case of the send messages or the format in which the acknowledgement is to be sent for the case of the receive messages. It is the responsibility of each simulation code to determine the proper format for the send messages and the format of the acknowledgements to the receive messages. There are PVM functions that can be used to determine the architecture of the host on which the simulation code is being executed and the architecture of the host on which the other code is being executed. If the architectures are the same, raw mode should be used for the data exchanges and xdr mode must be used if the architectures are different. Raw mode should be used if possible because it eliminates the loss of precision for real numbers when using xdr for the conversion of real numbers between different architectures. The simulation code must determine the format for sending all of the messages that it sends, either to the PVMEXEC program or to other processes in the coupled simulation on a task to task basis.

Messages with message tag 1003 describe the contents of receive messages and contain the following data. The first data item is an integer that specifies the message tag of the message to be received. This is followed by another integer that specifies the task identifier of the task from which to receive the data. Next comes an integer that specifies whether the message is for synchronous or asynchronous explicit coupling. Next comes another integer that specifies the subtype of explict coupling (parallel or sequential explicit coupling) being used to couple this simulation code to the other simulation code. This is followed by a real number that specifies the wait time for this message. Next comes an integer that specifies the number of characters in the string variable that names the data items in the message. This is followed by a string of the specified length. Like the string in the send messages, the string contains the identifiers for the data items that are to be received in the message. Each of the messages must be acknowledged by sending a message with message tag 1003 to the PVMEXEC program that contains the task identifier of the simulation code.

The next set of messages contain data for semi-implicit thermal-hydraulic coupling. The number of send and receive messages for semi-implicit thermal-hydaulic coupling is contained in a message with message tag 1004. Next follow messages with message tag 1005 that specifiy the send messages followed by messages with message tag 1006 that specify the receive messages for semi-implicit thermal-hydraulic coupling. Messages with message tag 1005 contain the following data. The first data item is an integer that specifies the messages tag for the send message. This is followed by another integer that specifies the task identifier of the task to which to send the data. Next comes a real number that defines the wait time for the acknowledgement to the send message. This is followed by an integer that specifies the length of the string containing the identifiers of the data that is to be sent. The message ends with a string of the specified length. As with the strings for the specification of the data for the explicit thermal-hydraulic coupling, these strings contain the data specifications for semi-implicit thermal-hydraulic coupling. Each message must be acknowledged to the PVMEXEC program with a message containing a single integer, the task identifier of the simulation code.

Messages with message tag 1006 contain the same data as messages with message tag 1005 except that they specify the data that is to be received for semi-implicit thermal hydraulic coupling. In contrast to

explicit thermal-hydraulic coupling, semi-implicit thermal-hydraulic coupling requires two data exchanges per time advancement instead of a single data exchange per time step. Therefore, the data specifications contained in the 1005 and 1006 messages must be used to generate another set of send and receive messages. The message tags to be used by these internally generated messages are the message tags defined in the 1005 and 1006 messages plus 2000. The task identifiers and wait times are the same for the implied send and receive messages as contained in the 1005 and 1006 messages. The data identifiers contained in the strings in the 1005 and 1006 messages are used to determine the data to be exchanged in the implied send and receive messages. This will be explained further in the Section 3.2.2.2 of this report.

The semi-implicit thermal-hydraulic coupling send and receive message specification messages are followed by similar messages for the kinetics coupling messages. The numbers of send and receive message for kinetics coupling are contained in a message with message tag 1007. The kinetics messages for sending data are defined in messages with message tag 1008 and the kinetics messages for receiving data are defined in messages with message tag 1009.

Messages with message tag 1008 contain the following data. The first data item is an integer that specifies the message tag for the send message. This is followed by another integer that specifies the task identifier of the process to which the data are to be sent. Next comes a real number that defines the wait time for the acknowledgement to the send message. This is followed by an integer that specifies the length of the string that defines the data items to be sent in the message. The messages end with a string of the specified length. The kinetics receive messages are defined in messages with message tag 1009. The receive messages have the same format as the send messages except that the wait time is for how long to wait to receive the data rather than for how long to wait for an acknowledgement.

The last set of message contain the specifications for the data exchanges for control systems coupling. These messages use message tags 1010, 1011, and 1012. The message with message tag 1010 contains the number of control system send and receive messages. The messages with message tag 1011 contain the specification for control system send messages and the messages with messsage tag 1012 contain the specification for control system receive messages. The format of the 1011 and 1012 messages is the same as the 1008 and 1009 messages.

Once the last messages with message tag 1012 has been received, the simulation code can proceed to the initialization phase of its startup proceedure. During initialization, the coupled codes exchange data to define the correct initial state of the coupling components. These exchanges are only performed for explicit thermal-hydraulic coupling. This is done to be consistent with the data exchange process for the time advancements for explicit thermal-hydraulic coupling where data are exchanged at the end of an advancement to prepare for the next advancement. Because there is no previous advancement for the first time step in a simulation, a data exchange is performed during initialization to emulate the data exchanges that would have been performed in the non-existent previous time step.

The reader will have noticed that the PVM coupling methodology makes many assumptions about the sequence of computations that are performed by the simulation codes. The assumed sequence is taken

from the sequence of computations performed by the RELAP5-3D© code since the coupling methodology was originally developed for coupling codes to RELAP5-3D©. It is the programmers responsibility to ensure that the simulation codes send and receive data at the appropriate times so that the coupled computation can proceed smoothly. Deadlocks are easily encountered if all codes are waiting to receive data and no code is sending data or vise-versa. Keeping the codes synchronized in the sense of sending and receiving data at the appropriate times is the hardest part of adding a new simualtion code to the coupling methodology.

3.1.3 Initial Data Exchanges

The data exchanges during initialization are coordinated by the PVMEXEC program so that each code is allowed to send its data to the other codes and the other codes are listening to receive data while the specified code is sending data. The PVM data exchange methodology as implemented by the PVMEXEC program behaves like an old style telephone partyline. Only one person can talk at a time and the others must listen while someone else is talking. This data exchange is coordinated by the PVMEXEC program sending and the simulation codes receiving control messages from the PVMEXEC program with different message tags for the different types of explicit thermal-hydraulic coupling and for semi-implicit coupling.

The first data exchange is for synchronous parallel explicit thermal-hydraulic coupling. The PVMEXEC program broadcasts a series of messages with message tag 8002. This message contains the task identifier of the task that is to send data. The message is sent to all tasks that are participating in synchronous parallel explicit thermal-hydraulic coupling and all tasks participating in synchronous parallel explicit thermal-hydrualic couping are expected to listen for this message. When a task receives this message, it should determine if the task identifier contained in the message is its task identifier. If the identifier is its identifier and if it has synchronous parallel explicit thermal-hydraulic send messages in its message database, it should send the appropriate data to the appropriate task and wait for an acknowledgement from the receiving task. This is the first data exchange between simulation tasks. The acknowledgement to messages from the PVMEXEC program consist of a message containing a single integer value, the task identifier of the acknowledging task. The acknowledgement between simulation tasks should contain a single integer value, either zero or one, because the value in the message is not significant, only the fact that a message has been received from the expected task is significant. The task should keep sending data and receiving acknowledgements until all of its send messages for synchronous parallel explicit thermal-hydraulic coupling have been sent and acknowledged. It should then send a message with message tag 8002 to the PVMEXEC program containing its task identifier to signify that it is finished sending data. When a simulation code receives a message with message tag 8002 that contains the task identifier of another task, it must examine its message database to determine if it is to receive synchronous parallel explicit thermal-hydraulic data from the task specified in the message. If it is to receive data from the specified task, it must listen to receive the data, process the received data, and acknowledge the receipt of the data to the sending task. It must keep listening until all data has been received from the sending task and each message has been acknowledged by sending an acknowledgement to the sending task. Once all data from the designated task has been received and acknowleged, the receiving task should send a message to the PVMEXEC program with message tag 8002 containing its task

identifier. This signifies to the PVMEXEC program that it is finished receiving data from the designated task. When the PVMEXEC program receives acknowledgements from all of the tasks that are participating in synchronous parallel explicit thermal-hydraulic coupling that they are finished processing the data that is to be sent by the designated task, the PVMEXEC program sends the next message with message tag 8002 allowing the next task to send its data to the other tasks participating in synchronous parallel explicit thermal-hydraulic coupling. This continues until all tasks that are participating in synchronous parallel explicit thermal-hydraulic coupling have been allowed to send their respective data. Once all tasks that participate in synchronous parallel explicit thermal-hydraulic coupling have been allowed to send data, the PVMEXEC program sends a final message with message tag 8002. This message contains the integer zero and denotes that the data exchanges for synchronous parallel explicit thermal-hydraulic coupling are finished and that the tasks do not need to listen for more messages with message tag 8002 during initialization.

The second series of data exchanges is for asynchronous parallel explicit thermal-hydraulic coupling. These data exchanges are controlled using messages with message tag 8003. The process for the data exchanges is the same as for the exchanges for synchronous parallel explicit thermal-hydraulic coupling.

The next series of data exchanges is for synchronous sequential explicit thermal-hydraulic coupling. Sequential explicit thermal-hydraulic coupling implies an order in which the several simulation codes perform their computations. This order is specified in the input file for the PVMEXEC program. Tasks are labelled as ’leader’ or ’follower’ tasks. The ’follower’ tasks compute the properties in the ’volume’ components and the ’leader’ tasks compute the properties in the ’junction’ components between the two computational domains. The ’leader’ tasks need to know the properties in the ’volumes’ before they can compute the properties in the ’junction’ components (the junction properties are ’donored’ from the volume properties). Therefore, the ’follower’ tasks are directed to send their ’volume’ data to the ’leader’ tasks first. This exchange of data is coordinated using messages from the PVMEXEC program with message tag 8004 that contain the task identifier of the task that is to send data. All other tasks participating in synchronous sequential explicit thermal-hydraulic coupling are expected to listen for the data being sent by the designated ’follower’ task. The series of 8004 messages is terminated by a message with message tag 8004 containing a integer value of zero denoting that all ’follower’ tasks participating in synchronous sequential explit thermal-hydraulic coupling have sent their data. The ’leader’ tasks should then use the data obtained from the ’follower’ tasks to compute the properties in the coupling ’junction’ components. Then the data exchange process is repeated with another series of messages with message tag 8004 except that the ’leader’ tasks are directed to send their data to the appropriate ’follower’ task. This series of messages is also terminated with a message from the PVMEXEC program with message tag 8004 that contains the integer zero. This designates the all of the ’leader’ tasks in synchronous sequential explicit thermal-hydraulic coupling have had a chance to send their data.

The next series of data exchanges is for asynchronous sequential explicit thermal-hydraulic coupling. The process is the same as that described for synchronous sequential explicit thermal-hydraulic coupling in the preceeding paragraph except that the control messages from the PVMEXEC program use message tag

8005 instead of message tag 8004. This completes the data exchanges for explicit thermal-hydrualic coupling.

The next set of data exchanges for initialization purposes are exchanges between tasks that are participating in semi-implicit coupling. The messages are sent (and received) autonomously by the semi-implicitly coupled tasks and are not under the control of the PVMEXEC program. This is possible because there is a one to one relationship between tasks that are semi-implicitly coupled tasks and the roles of each task in semi-implicit coupling determine the order in which messages are sent and received. Each task in semi-implicit coupling is a 'master' task or a 'slave' task (See Section 3.2.2.2 for an explanation of the 'master' and 'slave' tasks). First, the 'master' task sends it's data to its counterpart 'slave' task, waits to receive the expected acknowledgement, and then listens to receive data from its 'slave' task, finishing the data exchanges by sending its acknowledgement to the 'slave' task. The 'slave' task first listens to receive a message from its 'master' process, sends an acknowledgement to the 'master' task, then sends its data to the 'master' task, then finishing the data exchange by waiting and receiving the acknowledgement from it's 'master' process.

The last set of data exchanges for initialization purposes are exchanges between tasks that are participating in kinetics coupling. These messages are exchanged after the initialization of the thermal-hydraulic models has been finished by the previous exchanges of messages. The messages are sent (and received) autonomously by the tasks participating in kinetics coupling and are not under the control of the PVMEXEC program. This is possible because there is a one to one relationship between tasks that are participating in kinetics coupling and the roles of each task in kinetics coupling determine the order in which messages are sent and received. Each task in kinetics coupling is a 'server' task or a 'client' task (See Section 3.2.2.3 for an explanation of the 'server' and 'client' tasks). First, the 'client' task sends it's data (thermal-hydraulic data) to its counterpart 'server' task, waits to receive the expected acknowledgement, and then listens to receive data from its 'server' task (power data), finishing the data exchanges by sending its acknowledgement to the 'server' task. The 'server' task first listens to receive a message from its 'client' process (thermal-hydraulic data) and sends an acknowledgement to the 'client' task. It then uses the data it received from the 'client' task to initialize the its kinetics model. It then sends its data (the initialized power data) to the 'client' task, finishing the initial data exchange by waiting and receiving the acknowledgement from its 'client' process.

Once the last series of initialization data exchanges have been performed, the simulation code may finish its initialization. At the end of the initialization process, each code is expected to send a message to the PVMEXEC program with message tag 9000. This message must contain three integers. The first integer is the tid of the task sending the message. The second integer specifies whether the initialization process has been completed successfully. A value of zero denotes the successful completion of initialization and a value greater than zero denotes an error condition. The programmer may develop his (or her) own set of error conditions for each individual simulation code. These values are printed on the output file of the PVMEXEC program and may be used to diagnose errors. The PVMEXEC program only tests for zero or nonzero. The third integer in message 9000 denotes whether the simulation code is prepared to execute a transient. A value of zero means that the simulation code wants to terminate the

execution after the initialization phase and a value of one denotes that the simulation code wants to continue the coupled computation to the transient phase of the computation. Some codes have the ability to stop after input processing and initialization. Stopping after input processing and initialization is helpful when developing a new input model for a simulation code or for making sure that the coupling information contained in the input file of the PVMEXEC program is correct. In any case, the PVMEXEC program examines the two values from all of the simulation codes and sends a integer run flag to the simulation codes in a message with message tag 9001. A value of zero means that the computation is to be terminated and a value of one denotes that the coupled simulation should preceed to the transient phase of the simulation. The receipt of the message with message tag 9001 completes the PVM initialization related responsibilities of a simulation code that wishes to be part of a coupled simulation.

3.2 Transient Phase of Coupled Simulation

There are two distinct phases of data exchange during each time step of a transient computation. The first set of data exchanges control the choice of the time step sizes along with the production of output, and the second set of data exchanges facilitate the coupled solution algorithms. These two phases will be discussed separately because the time step selection and production of output is the same for each type of coupling while the data exchanges during the time step that are used to effect the coupling are different for the several different types of coupling.

3.2.1 Time Step Selection and the Production of Output

The PMVEXEC program controls the production of output for all types of coupling for the synchronous mode of coupling and the production of the restart file for asynchronous mode of coupling. It also controls the time step size for synchronous coupling and the coupling interval for asynchronous coupling. The production of output and the choice of the time step size are assumed to occur at the end of a time step before proceeding to the next time step (this is the RELAP5-3D© computational sequence). As with the initialization of explicit thermal-hydraulic coupling and the production of output, time step selection must also occur at the beginning of the first time step of a transient computation to emulate the normal computations that occur at the end of all time steps. It is assumed that a set of output times has been obtained from the PVMEXEC program during previous time steps for time steps after the first time step in a simulation run.

3.2.1.1 Production of Output

At the end of a time step, the coupled simulation codes should compare the local simulation time to the minimum of the set of output times obtained from the PVMEXEC program to determine if any type of output needs to be generated. The default values of these times should be set to the start time of the simulation either during initialization or at the beginning of the first pass through the coupling control logic. This ensures that output will be obtained at the beginning of the first time step of the new simulation or restart run. There are four types of output that correspond to the types of output produced by RELAP5-3D©. The output are the minor edit output, the plot output, the major edit output, and the restart output. Each code may not produce all types of output. All output times are determined by the PVMEXEC

program for synchronous mode of coupling while only the restart times are controlled by the PVMEXEC program for asynchronous coupling. This means that the minor edit, major edit, and plot times are controlled by the individual simulation codes for asynchronous coupling. The code must check to determine if an output, restart, or asynchronous coupling interval time has been reached. If an output time has been reached, the simulation code should produce the desired type of output, if possible (not all codes produce minor edits, etc.). If the restart time has been reached, the code should write the restart data to the restart file.

After the specified output has been produced, the codes exchange messages for explicit thermal-hydraulic coupling if required. These data exchanges are coordinated by the PVMEXEC program and are explained in Section 3.2.2.1.

After the production of any output or data exchanges for explicit thermal-hydraulic coupling, the code must listen for a message with message tag 8000 from the PVMEXEC code. This message will contain eight real values. The values are new minor edit time, new plot time, new major edit time, new restart output time, new explicit coupling interval time, and end time for the simulation, followed by the maximum and minimum time step size for use in synchronous coupling. Depending on which of the times in the previous set of times has been reached at the end of the current time step, many of the new times will be the same as the corresponding time in the previous set. This message should be acknowledged with a message containing the task identifier of the receiving task. This process of checking for the production of output, the production of output if indicated, and the receipt of a new set of times if needed occurs at the end of each time advancement until the simulation reaches the end time specified by the PVMEXEC program. When the local simulation time reaches the end time for the simulation, the simulation code is expected to terminate after the production of output without listening to receive a message with message tag 8000 in the same way that it would terminate if it reached the end of a simulation while being executed in uncoupled mode.

3.2.1.2 Time Step Selection for Synchronous Coupling

After the output and aysnchronous coupling control logic described in the previous section has been executed, the time step size is determined for synchronous coupling. After the code determines the time step size that it wants to use for the next time step advancement based on its own internal time step selection algorithm, it sends this value (as a real number) to the PVMEXEC program in a message with message tag 8001 if it is participating in synchronous coupling (a code may be participating in synchronous coupling, in asynchronous coupling, or both types of coupling to different other simulation codes). This message must contain an integer specifying the task identifier of the simulation code followed by the value of the time step desired (as a real number). The PVMEXEC program will determine the synchronous time step size based on the values reported by all of the codes participating in synchronous coupling. The time step is chosen by halving or doubling the current time step size (limited by the user input maximum and minimum time step sizes) so that it is the largest value just below the minimum of the time steps requested by the code participating in synchronous coupling. This global time step size is sent to the synchronously coupled codes in a return message with message tag 8001. This message will contain the following four real values: the time step size to be used by all codes participating in synchronous

coupling, a new major edit time, a new minor edit time, and a new plot time. The new edit times are included in the time step message so that the PVMEXEC program may request output at the end of every time step advancement. The ability to produce output at the end of every time step has been adapted from the RELAP5-3D© code.

3.2.2 Data Exchanges During Time Steps

The data exchanges during the transient computations for the several types of synchronous coupling are described in the following sections.

All codes participating in synchronous coupling, either synchronous explicit thermal-hydraulic coupling, semi-implicit thermal-hydraulic coupling, or control systems coupling, are required to send a message with message tag 10000 to the PVMEXEC program at the end of the thermal-hydraulic portion of their computations. The order of the computations in each couled code is assumed to be thermal-hydraulic computation, then kinetics computations, then control system computations (this is the order of computations in RELAP5-3D©). Only a failure in the thermal-hydraulic computations can cause a failure of a coupled computation, subsequent backup, and retry of the advancement. The kinetics and control systems computations are assumed to always be successful and no failure and backup logic is provided for these types of computations. The message with message tag 10000 must contain two integer values, the first integer is the task identifier of the sending task followed by a status flag. This status flag specifies whether the thermal-hydraulic portion of their time step advancement has been successful or whether the code wishes to repeat the advancement. RELAP5-3D© may want to repeat the advancement for any number of reasons like excessive mass error during the time step advancement, a velocity flip-flop during the advancement, etc. A value of zero denotes that the advancement was successful and a value greater than zero represents an ‘error’ condition and requests either a time step repeat or a simulation termination. The ‘error’ codes understood by the PVMEXEC code are values of one and two that denotes that the code wishes to repeat the time step with a smaller time step size and values greater than two that denote that the code wishes to repeat the time step with the same time step size. A value of one denotes that the computations may continue even if the reduced time step size to be used during the repeat is the minimum time step size while a value of two denotes that the code only wishes to repeat the time step if the current time step size is greater than the minimum time step size (the time step size can only be reduced to the minimum time step size and it would be futile to repeat the advancement if the time step size were already at the minimum time step size). RELAP5-3D© can continue computations at the minimum time step size for excessive mass error during the time step (status flag equals one) but must stop if a water property failure occurs at the minimum time step size (status flag equals two). A value of six denotes the the code cannot continue with any time step size however small and that the coupled computation should be terminated. The PVMEXEC program will reply with a message with message tag 10000 that contains an integer variable specifying a global success flag. A value of zero for the global success flag denotes that the time step was successful and that all of the codes participating in synchronous coupling should proceed with the remainder of their computations for the time step. A value greater that zero denotes that the time step advancement has been unsuccessful for some reason and must be repeated. The codes that are participating in synchronous coupling are expected to do whatever is necessary for them to prepare for a

repeated attempt of the time step advancement if the global success flag is greater than zero. If the global success flag indicates a failed attemped advancement, the coupled codes are expected to proceed to the time step selection logic, choose a new time step size for the next attempt at the advancement, and to send their requested time step size to PVMEXEC in messages with message tag 8001. The global time step size for the next attempt at the advancement is chosen as explained in Section 3.2.1.2.

If the advancement is successful, additional messages are exchanged between the PVMEXEC program and the several coupled codes. The messages for each type of synchronous coupling are described in the following subsections.

3.2.2.1 Explicit Thermal-hydraulic Coupling

The additional data exchanges for explicit thermal hydraulic coupling are performed at the end of a successful time step advancement just before the execution of the output control logic described in Section 3.1.3. The data exchanges are controlled by messages 8002, 8003, 8004, and 8005. Control messages 8002 and 8004 are exchanged with the PVMEXEC program each and every time step because these messages control the data exchanges for synchronous explicit coupling. The actual data exchanges between the tasks that are partipating in synchronous explicit coupling are as described in the Section 3.1.3 of this report that describes the initialization of a coupled computation. Control messages 8003 and 8005 are exchanged with the PVMEXEC program only if the local simulation time has reached the explicit coupling interval time and the simulation code is partipating in aysnchronous explicit coupling. The data exchanges between the tasks participating in asynchronous explicit coupling are as described in the Section 3.1.3 of the report describing the initialization of a coupled computation.

3.2.2.1.1 Sequential Explicit Thermal-hydraulic Coupling

The data exchanges described above are performed at the end of the time step for all types of explicit thermal-hydraulic coupling. Addtional computations and data exchanges are needed for sequential explicit coupling, both synchronous and asynchronous. In sequential explicit coupling, the ’leader’ process performs its advancement (or multiple advancements for asynchronous coupling) and sends the ’junction’ properties to the appropriate ’follower’ task. For synchronous sequential coupling, the values sent are the values at the end of the time step. For asynchronous sequential coupling for the data items that are flow rates of mass and energy through the ’junction’, the values sent must be the average value of the junction flow rates over the multiple time steps that constitiute the current asynchronous coupling interval. This means that the ’leader’ tasks must perform the averaging of the ’junction’ flow rates over the coupling interval for asynchronous coupling. The ’follower’ task must wait and listen to receive the ’junction’ properties from the appropriate ’leader’ task before performing its thermal-hydraulic advancements, either a single advancement for synchronous sequential coupling or a series of advancements to span the current coupling interval for asynchronous coupling. The computation of ’junction’ flow rates ensures that the total amount of mass and energy that leaves (or enters) the ’leader’ solution domain is the same as the amount of mass and energy that enters (or leaves) the adjacent ’follower’ solution domain. The extra data exchange between the ’leader’ task and its ’follower’ task are performed autonomously by the two processes and not under the control of the PVMEXEC program. The exchange occurs after the ’leader’

process has completed its advancements and before the 'follower' process begins its advancements. These data excahnges are coordinated by the PVMEXEC process using 8004 and 8005 control messages for synchronous and asynchronous explicitly coupled processes, respectively.

3.2.2.2 Semi-implicit Thermal-hydraulic Coupling

The data exchanges between tasks participating in semi-implicit coupling are autonomous in the sense that they are not controlled by messages from the PVMEXEC program. Because semi-implicit thermal-hydraulic coupling is synchronous by definition, the data exchanges are performed each and every time step. The order of the data exchanges is determined by the role the individual simulation codes perform in the semi-implicit coupling solution algorithm. The semi-implicit coupling algorithm defines a one to one relationship between the semi-implicitly coupled processes whereas a task may be coupled explicitly to any number of other tasks. In this one to one relationship, one task is designated as the 'master' and the other task is denoted as the 'slave' task. The slave task is the task that computes the flow rates of mass and energy in the 'junctions' between the two semi-implicitly coupled computational domains. The master task is the task that determines the functional relationship between the boundary pressures and the coupling 'junction' flow rates (See **Reference 10** for a description of the semi-implicit coupling solution algorithm). Each task must determine whether it is the 'master' process or the 'slave' process if it is participating in semi-implicit thermal-hydraulic coupling. Once a time advancement has begun, the 'master' process computes the functional relationship between the boundary pressures and the 'junction' flow rates (i.e., the boundary pressure coefficients) and sends these values to the appropriate 'slave' process. It then must wait to receive the fluid flow rates in the several coupling 'junctions' . The 'slave' process must listen to receive the values of the pressure coefficients from the 'master' process, receive the data, and send an acknowledgement to the 'master' process. Once the 'slave' process has received the boundary pressure coefficients, it can proceed with its solution. Once the 'slave' process has finished its advancement, it must send the 'junction' flow rates to its 'master' process. The master process has been listening to receive the coupling 'junction' flow rates while the 'slave' process performs its advancement. Once the master receives and acknowledges the coupling 'junction' flow rates, it can complete its advancement. These data exchanges are autonomous and are not controlled by the PVMEXEC program. These two data exchanges constitute the 'indirect' or 'implied' data exchanges mentioned in the initialization section of this report for semi-implicit thermal-hydraulic coupling. The 'direct' data exchanges are performed at the end of the advancement. First the master process sends the properties in the boundary 'volumes' to the appropriate 'slave' process and then listens to receive the properties in the coupling 'junctions' for its 'slave' process. The 'slave' task listens to receive the properties in the boundary 'volumes' from the 'master' task. It then uses these data to compute the fluid properties in the coupling 'junctions' and sends these values to the appropriate 'master' process. The data in both messages are end of time step values. The variables to be exchanged for semi-implicit thermal-hydraulic coupling are defined in Appendix C. Appendix C also lists the macros that can be used to specify these variables to RELAP5-3D© in the 1005 and 1006 messages used during the initialization phase of a semi-implicitly coupled simulation.

3.2.2.3 Kinetics Coupling

Kinetics coupling is a type of synchronous sequential explicit coupling, sequential in the sense that the thermal-hydraulic advancements are performed first and then the kinetics advancements. It is a form of explicit coupling in that the values computed by one task and sent to the other task are held constant while being used by the other task to perform its advancement. As with semi-implicit coupling, each task participating in kinetics coupling is assigned a specific role by the user and these roles determine the data that is to be exchanged between the tasks and when the data are exchanged during the advancement. Because each task in kinetics coupling knows it role, the data exchanges are autonomous and are not controlled by the PVMEXEC program. Each task in kinetics coupling is either a 'client' task or a 'server' task. These names are arbitrary and only denote the roles that the tasks perform in kinetics coupling. The 'server' task is the task that computes the power in the nuclear reactor core using some form of neutron kinetics model, either a simple point kinetics model or a more complicated multi-dimensional kinetics model. Once the tasks have begun a time step advancement, the 'client' task computes the fluid properties and heat structure temperatures in the reactor core using the power computed by the 'server' task to advance its thermal-hydraulic solution. The thermal-hydraulic advancement may be an uncoupled advancement, an explicitly coupled advancement, or a semi-implicitly coupled advancement. The type of thermal-hydraulic advancement is immaterial to kinetics coupling except that the thermal-hydraulic advancement must be synchronous because the time step sizes must be the same for the thermal-hydraulic advancement and the kinetics advancement (asynchronous kinetics coupling may be implemented in the future). First the 'client' task performs its thermal-hydraulic solution using the power computed by the kineitcs solution that was performed at the end of the previous time step. It then sends the values of fluid properties and heat structure temperatures in the appropriate core regions to the 'server' process. The 'server' process must listen to receive these data. It acknowledges receiving the data and then uses the data in its computation of the reactor power. After the reactor power has been computed, it sends the appropriate power data to the 'client' process. The 'client' process must listen to receive the power data after receiving the acknowledgement to its sending the fluid property and heat structure data to the 'server' process. It processes the data received from the 'server' task and sends an acknowledgement to the 'server' process. Both processes then process to the next stage of the advancement. These computations are performed before any production of output or the determination of the time step size for the next advancement.

3.2.2.4 Control Systems Coupling

Control systems coupling, like kinetics coupling, is a form of synchronous explicit coupling that is performed after any thermal-hydraulic coupling and kinetics coupling. Like kinetics coupling and semi-implicit thermal-hydraulic coupling, the data exchanges between tasks participating in control systems coupling are autonomous and are not coordinated by the PVMEXEC program. The control systems coupling has been implemented in RELAP5-3D© by a new type of control component. This new control component can send data, receive data, or receive data after sending data. The control components in RELAP5-3D© are advanced in time in numerical order. If a control component is a coupling component, it first examines its data base to determine if it is to send data to another task. If it determines that it is to send data to another task, it sends the appropriate data and then waits to receive an acknowledgement. After receiving an acknowledgement, or if it determines that it is not expected to send data to another task,

it examines its data base to determine it it is to receive data from another task. If it determines that it is expected to receive data from another task, it listens to receive the appropriate message from the other task. Once it receives the message from the other task, it sends an acknowledgement to the sending task, processes the data in the message and then proceeds to the processing of the next control component. The same type of processing must be implemented in any code wanting to participate in control systems coupling. Because the control systems coupling messages are autonomous, the user must be extremely careful to order the control system computations to ensure that one task is listening while its counterpart is sending and visa-versa so that deadlocks do not occur (e.g., both tasks are listening or both tasks are sending at the same point in the control systems computations).

4 Summmary

The Application Programming Interface (API) for the coupling of several different simulation codes as implemented by the PVMEXEC program has been described in the previous sections of this report. The data that can be exchanged with RELAP5-3D© for the several different type of coupling are listed in Appendix C..

5 References

1 C. Y. Paik and L. E. Hochreiter, *Analysis of FLECHT SEASET 163-Rod Blocked Bundle Data Using COBRA-TF*, NUREG/CR-4166, US Nuclear Regulatory Commission, 1986.

2 The RELAP5 Code Development Team, *RELAP5-3D Code Manuals*, Vol I, II, IV, and V, INEEL-EXT-98-00834, Idaho National Engineering and Environmental Laboratory, Revision 2.2, October 2003.

3 Safety Code Development Group, *TRAC-PF1/MOD1: An Advanced Best-Extimate Computer Program for Pressurizer Water Reactor Thermal-Hydraulic Analysis*, LA-10157-MS, NUREG/CR-3858, Los Alamos National Laboratory, July 1986.

4 J. W. Spore, et. al., *TRAC-M/FORTRAN 90 (Version 3.0) Theory Manual*, NUREG/CR-6724, Los Alamos National Laboratory and Pennsylvania State University, July 2001.

5 P. J. Turinsky, R. M. K. Al-Chalabi, and P. Engrand, *NESTLE: A Few-Group Neutron Diffusion Equation Solver Utilizing the Nodal Expansion Method for Eigenvalue, Adjoint, Fixed Source Steady State and Transient Problems*, EGG-NRE-11406, Idaho National Engineering Laboratory, June 1994.

6 T. Downar et al., *PARCS: Purdue Advanced Reactor Core Simulator*, PU/NE-98-26, Purdue University, West Lafayette, IN, September 1998.

7 K. K. Murata et al., *Code Manual for CONTAIN 2.0: A Computer Code for Nuclear Reactor Containment Analysis*, SAND97-1735, NUREG/CR-6533, Sandia National Laboratory, December 1997.

8 R. O. Gauntt et al., *MELCOR Computer Code Manuals: Version 1.8.5,* SAND2000-2417, NUREG/CR-6119, Sandia National Laboratory, October 2000.

9 A. Geist et al., *PVM (Parallel Virtual Machine) User's Guide and Reference Manual,* ORNL/TM-12187, Oak Ridge National Laboratory, May 1993.

10 W. L. Weaver, E. T. Tomlinson, D. L. Aumiller, "A Generic Semi-Implicit Coupling Methodology for Use in RELAP5-3D©," *Nuclear Engineering and Design, 211*, 2003, pp 13-26.

Appendix A Message Tags

This appendix lists all of the message tags used by the PVMEXEC program along which a short description of the data in the message. Unless otherwise specified, the message specification is for the message that is sent from the PVMEXEC program to the designated simulation task. For these messages, the message acknowledgment is returned to the PVMEXEC in a message with the same message tag that contains a single data item in the message, the task identifier of the task sending the acknowledgement. Messages that are listed as pairs of send and receive messages with the same message tag define the handshaking for that message.

Message Tag 1

W1(R) Task transient type (-). A value of zero means a synchronous task and a value of 1 means an asynchronous task.

W2(R) Transient start time (s).

W3(R) Transient restart time (s)

W4(I) Not used (RELAP5-3D specific).

W5(I) Length of string containing simulation name (-).

W6-(A) Simulation name (-).

Message Tag 2

W1(I) Number of threads for multi-threaded simulation codes (-).

Message Tag 1000

W1(R) Global wait time (s).

W2(I) Debug flag (-).

Message Tag 1001

W1(I) Number of explicit send messages.

W2(I) Number of explicit receive messages.

Message Tag 1002

W1(I) Message tag (-).

W2(I) Task identifier of task receiving message (-).

W3(I) Message transient type (-). Zero means synchronous message, one means asynchronous message.

W4(I) Explicit coupling type (-). A value of -1 means parallel explicit coupling, a value of 0 means ‘leader’ in sequential explicit coupling, and a value of 1 means ‘follower’ in sequential explicit coupling.

W5(R) Wait time for acknowledgement (s).

W6(I) Length of character string defining thermal-hydraulic data to be sent for explicit coupling(-).

W7-(A) Character string defining thermal-hydraulic data to be sent for explicit coupling.

Message Tag 1003

W1(I) Message tag (-).

W2(I) Task identifier of task sending message (-).

W3(I) Message transient type (-). Zero means synchronous message, one means asynchronous message.

W4(I) Explicit coupling type (-). A value of -1 means parallel explicit coupling, a value of 0 means ‘leader’ in sequential explicit coupling, and a value of 1 means ‘follower’ in sequential explicit coupling.

W5(R) Wait time to receive message (s).

W6(I) Length of character string defining thermal-hydraulic data to be received for explicit coupling (-).

W7-(A) Character string defining thermal-hydraulic data to be received for explicit coupling.

Message tag 1004

W1(I) Number of send messages for semi-implicit coupling.

W2(I) Number of receive messages for semi-implicit coupling.

Message tag 1005

W1(I) Message tag (-).

W2(I) Task identifier of task receiving the message (-).

W3(R) Wait time for the message acknowledgment (s).

W4(I) Length of character string defining the thermal-hydraulic data to be sent for semi-implicit coupling (-).

W5-(A) Character string defining the thermal-hydraulic data to be sent for semi-implicit coupling.

Message tag 1006

W1(I) Message tag (-).

W2(I) Task identifier of task sending the message (-).

W3(R) Wait time to receive the message (s).

W4(I) Length of character string defining the thermal-hydraulic data to be received for semi-implicit coupling (-).

W5-(A) Characte r string defining the thermal-hydraulic data to be received for semi-implicit coupling(-).

Message tag 1007

W1(I) Number of send messages for kinetics coupling.

W2(I) Number of receive messages for kinetics coupling.

Message tag 1008

W1(I) Message tag (-).

W2(I) Task identifier of task receiving the kinetics message (-).

W3(R) Wait time to receive the message acknowledgment (s).

W4(I) Length of character string defining the kinetics data to be sent(-).

W5-(A) Character string defining the kinetics data to be sent.

Message tag 1009

W1(I) Message tag (-).

W2(I) Task identifier of task sending the kinetics message (-).

W3(R) Wait time to receive the message (s).

W4(I) Length of character string defining kinetics data to be received (-).

W5-(A) Character string defining kinetics data to be received.

Message tag 1010

W1(I) Number of send messages for control system coupling.

W2(I) Number of receive messages for control system coupling.

Message tag 1011

W1(I) Message tag (-).

W2(I) Task identifier of task receiving the control system message (-).

W3(R) Wait time to receive the message acknowledgment (s).

W4(I) Length of character string defining the control system data to be sent (-).

W5-(A) Character string defining the control system data to be sent.

Message tag 1012

W1(I) Message tag (-).

W2(I) Task identifier of task sending the control system message (-).

W3(R) Wait time to receive the control system message (s).

W4(I) Length of character string defining the control system data to be received (-).

W5-(A) Character string defining the control system data to be received.

Message tag 7000

W1(A) Fixed length string containing name of executable file for simulation task.

W2(A) Fixed length string containing command line parameters for simulation task.

W3(A) Fixed length string containing the name of the host in the virtual machine on which the execute the task.

W4(A) Fixed length string containing the name of the output file for the simulation task.

W5(I) Debugging flag (-).

Message tag 7001

W1(I) Task identifier of PVMEXEC task (-).

Message tag 8000

W1(R) Time for next minor edit (s).

W2(R) Time for next plot output (s).

W3(R) Time for next major edit (s).

W4(R) Time for next restart output (s).

W5(R) Time for next explicit data exchange (s).

W6(R) End time of transient (s).

W7(R) Maximum time step size for synchronously coupled tasks (s).

W8(R) Minimum time step size for synchronously coupled tasks (s).

PVMEXEC receive / analysis code send message tag 8001

W1(R) Desired time step size for synchronously coupled simulation task sending the message (s).

PVMEXEC send / analysis code receive message tag 8001

W1(R) Global time step size for synchronously coupled simulation tasks (s).

W2(R) Current time for a major edit (s).

W3(R) Current time for a minor edit (s).

W4(R) Current time for a plot record (s).

Message tag 8002

W1(I) Task identifier of explicitly coupled task that is to send data to other explicitly coupled tasks for synchronous parallel explicit coupling. (s)

Message tag 8003

W1(I) Task identifier of explicitly coupled task that is to send data to other explicitly coupled tasks for asynchronous parallel explicit coupling (s).

Message tag 8004

W1(I) Task identifier of explicitly coupled task that is to send data to other explicitly coupled tasks for synchronous sequential explicit coupling (-).

Message tag 8005

W1(I) Task identifier of explicitly coupled task that is to send data to other explicitly coupled tasks for asynchronous sequential explicit coupling. (-)

Message tag 9000

W1(I) Task identifier of task sending message (-).

W2(I) Initialization status flag for task (-).

W3(I) Run status flag for task (-).

Message tag 9001

W1(I) Global initialization flag (-).

W2(I) Global run status flag (-).

Receive message tag 10000

W1(I) Task identifier of simulation task sending the message (-).

W2(I) Success flag for synchronously coupled task (-).

Send message tag 10000

W1(I) Global success flag (-).

Message tag 10001

W1(I) Task identifier of simulation code that has failed (-).

Message tag 10002

W1(I) Task identifier of simulation task that has exceeded its wait time for a message from another simulation task (-).

Message tag 10003

W1(I)	Shutdown flag (-).

Appendix B
Order of Messages

This appendix lists the messages sent and received by the PVMEXEC program. Also listed are the messages between the several simulation codes that are not coordinated by the PVMEXEC program. The messages are listed in the order that they are sent for the two phases of a coupled computation, the initialization phase and the transient phase. The original message is described, and the acknowledgment is implied. Acknowledgments are contained in messages with the same message tag where the single data item in the message is the task identifier of the task asknowledging the receipt of the message. **Figure 2.3-1** shows the sequence of messages for the transient phase of a coupled computation.

1 Initialization Phase Messages

The messages sent by the PVMEXEC program to the coupled simulation codes during the initialization phase of a coupled computation are listed in the order that they are sent. The contents of the messages are listed in Appendix A.

1. Message 7001. This message is sent to all coupled simulation codes, regardess of the type of coupling in which they are participating.

2. Message 1. This message is sent to all coupled simulation codes

3. Message 2. This message is sent to all coupled simulation codes.

4. Message 1000. This message is sent to all coupled simulation codes.

5. Message 1001. This message is sent to all coupled simulation codes.

6. Message 1002. This message is only sent the simulation codes participating in synchronous, parallel, explicit thermal-hydraulic coupling.

7. Message 1003. This message is only sent the simulation codes participating in synchronous, parallel, explicit thermal-hydraulic coupling.

8. Message 1004. This message is sent to all coupled simulation codes.

9. Message 1005. This message is only sent the simulation codes participating in semi-implicit thermal-hydraulic coupling.

10. Message 1006. This message is only sent the simulation codes participating in semi-implicit thermal-hydraulic coupling.

11. Message 1007. This message is sent to all coupled simulation codes.

12. Message 1008. This message is only sent to simulation codes participating in kinetics coupling.

13. Message 1009. This message is only sent to simulation codes participating in kinetics coupling.

14. Message 1010. This message is sent to all coupled simulation codes.

15. Message 1011. This message is only sent to simulation codes participating in control systems coupling.

16. Message 1012. This message is only sent to simulation codes participating in control systems coupling.

17. Message 8002. This message is only sent the simulation codes participating in synchronous, parallel, explicit thermal-hydraulic coupling.

18. Message 8003. This message is only sent the simulation codes participating in asynchronous, parallel, explicit thermal-hydraulic coupling.

19. Message 8004. This message is only sent the simulation codes participating in synchronous, sequential, explicit thermal-hydraulic coupling.

20. Message 8005. This message is only sent the simulation codes participating in asynchronous, sequential, explicit thermal-hydraulic coupling.

21. Initialization message for kinetics coupling. The initialization messages for kinetics coupling have the same message tags and contents as the kinetics messages for transient kinetics coupling. The message tags for these messages lie in the range 1-999. These messages are autonomous messages not coordinated by the PVMEXEC program. They are sent between the tasks participating in kinetics coupling. The message is from the ’server’ process to the ’client’ process.

22. Initialization messages for semi-implicit coupling. The initialization messages for semi-implicit coupling have the same message tags and contents as the direct messages for semi-implcit coupling. The message tags for these messages lie in the range 1-999. These messages are autonomous messages not coordinated by the PVMEXEC program. They are sent between the tasks participating in semi-implicit coupling. The order of these messages depends upon the role that the simulation code sending the message is playing, i.e., the ’master’ or ’slave’ roles for semi-implicit coupling. The simulation code portraying the ’master’ role sends its message first and then the simulation code playing the ’slave’ role sends its message.

23. Initialization messages for kinetics coupling. The initialization messages for kinetics coupling have the same message tags and contents as the kinetics messages for transient kinetics coupling. The message tags for these messages lie in the range 1-999. These

messages are autonomous messages not coordinated by the PVMEXEC program. They are sent between the tasks participating in kinetics coupling. The order of these messages depends upon the role that the simulation code sending the message is playing, i.e., the 'client' or 'server' roles for kinetics coupling. The simulation code portraying the 'client' role sends its message first and then the simulation code playing the 'server' role sends its message after initializing its kinetics model using the data from the 'client' task.

24. Message 9000. This message is received from all coupled simulation codes. It contains the initialization status of the simulation tasks. The global initialization status is computed from the data in these messages and is and sent to all of the coupled simulation codes.

25. Message 9001. This message is sent to all coupled simulation codes. It contains the global run status and is only sent if the global initialization status indicates a successful initialization of all coupled simulation codes.

2 Transient Phase Messages

The messages sent by the PVMEXEC program for the transient phase of a coupled computation are listed in the order that they are sent by the PVMEXEC program. The contents of the messages are listed in Appendix A. Messages are sent at several different times during the time step advancements. The initial transient messages are sent to all coupled tasks at the beginning of the first time step of a coupled simulation. Synchronous messages are sent to every task participating in synchronous coupling at the end of every time step. Finally, asynchronous messages are sent to asynchronously coupled tasks at the end of the last time step of a coupling interval and whenever a restart record is to be written to the restart file. The messages will be described separately.

2.1 Initial Transient Messages

The initial transient messages are sent to all coupled simulation codes based of the type of coupling in which they are participating.

1. Message 8002. These messages are sent to all simulation codes participating in synchronous, parallel, explicit thermal-hydraulic coupling.

2. Message 8003. These messages are sent to all simulation codes participating in asynchronous, parallel, explicit thermal-hydraulic coupling.

3. Message 8004. These messages are sent to all simulation codes participating in synchronous, sequential, explicit thermal-hydraulic coupling.

4. Message 8005. These messages are sent to all simulation codes participating in asynchronous, sequential, explicit thermal-hydraulic coupling.

5. Message 8000. This message is sent to all coupled simulation codes

6. Message 8001.This message is received from all simulation codes participating in synchronous coupling. Its contain the desired time step size for each code. After the

global time step is computed, it is sent to all of the coupled codes participating in synchronous coupling.

2.2 Messages During Time Step

There are several different sets of autonomous messages that are sent during the course of a time step advancemet. These messages are sent between the participating tasks and are not coordinated by the PVMEXEC program. The first sets of messages are for thermal-hydraulic coupling, followed by messages for kinetics coupling, followed by messages for control systems coupling. Thermal-hydraulic messages are divided into messages for semi-implicit thermal-hydraulic coupling and for sequential, explicit thermal-hydraulic coupling. The messages are listed in the order that they are sent. The message tags for these messages are assigned by the PVMEXEC program as part of the initialization phase of the coupled computation.

1. Indirect semi-implicit messages. The message tags for these message lie in the range 2001-2999. These messages are exchanged by simulation codes participating in semi-implicit coupling. First, the ’master’ task computes the pressure coefficients and sends them to the ’slave’ task. The ’slave’ task uses the pressure coefficients to compute the flow rates in the coupling junctions and sends these flow rates to the ’master’ task. This completes the exchange of indirect semi-implicit messages.

2. Direct semi-implicit messages. The message tags for these messages lie in the range 1-999. These messages are exchanged between simulation codes participating in semi-implicit coupling after the thermal-hydraulic solution for the time step has been completed by both the ’master’ and ’slave’ tasks. First, the ’master’ task sends the fluid conditions in the coupling volumes. Then, the ’slave’ task computes the fluid conditions in the coupling junctions and sends them to the ’master’ task.

3. Sequential leader message. The message tag for this message lies in the range 1-999. The ’leader’ in sequential, explicit thermal-hydraulic coupling sends the average flow rates in the coupling junctions to the ’follower’ task at the end of the last time step for the explicit coupling interval.

4. Sequential follower message. The ’follower’ task waits at the beginning of the first time step of an explicit coupling interval to receive the average flow rates in the coupling junctions before advancing its thermal-hydraulic solution through the explicit coupling interval.

5. Kinetics messages. The message tags for these messages lie in the range 1-999. The kinetics messages are exchanged after the thermal-hydraulic portion of the time step advancement have been completed. First, the simulation code portraying the ’client’ role in kinetics coupling sends its messages to the simulation code portraying the ’server’ role in kinetics coupling. The ’server’ task computes the reactor power using the data from the ’client’ task. Finally, the ’server’ task sends the power data to the ’client’ task.

6. Control system messages. The message tags for these messages lie in the range 1-999. Control system messages are exchanged after the thermal-hydraulic and kinetics portions

of the computations for a time step advancement have been completed. The messages are sent in the order specified in the input data for the PVMEXEC program.

2.3 Messages At End Of TIme Step

The messages sent by or received by the PVMEXEC code at the end of a time step depend on the type of coupling, i.e. synchronous coupling or asynchronous coupling. The messages for synchronous coupling will be described first and then the messages for asynchronous coupling.

2.3.1 Synchronous Coupling

Four types of messages are sent at the end of a time step to simulation codes that are participating in synchronous coupling. These types of messages are; 1) advancement status message, 2) data exchange control messages, 3) output control messages, and 4) time step control messages. The first type of message is:

1. Message 10000. This message is received by the PVMEXEC program from each of the coupled simulation codes participating in synchronous coupling. It contains the status flag for the thermal-hydraulic portion of the attempted advancement. The global status flag is determined for the individual status flags and sent to all of the simulation codes participating in synchronous coupling. The flag indicates whether the thermal-hydraulic portion of the advancement was successful or not successful.

The data exchange control messages are only sent at the end of a successful advancement;

1. Message 8002. These messages are sent to all simulation codes participating in synchronous, parallel, explicit thermal-hydraulic coupling.

2. Message 8004. These messages are sent to all simulation codes participating in synchronous, sequential, explicit thermal-hydraulic coupling.

The third type of message is a output control message. It is only sent to the simulation codes participating in synchronous coupling at the end of a successful advancement. This message is sent whenever the current time is an output time. The output times are the times at which minor edits, plot records, major edit, and restart records are to be produced by the simulation codes. The message contains a new set of output times;

1. Message 8000. This message is sent to all simulation codes participating is synchronous coupling at output time.

The last type of message for synchronous coupling is the time step control message. It is received at the end of every attempted advancement, successful or not. If the advancement has been successful, the messages contains the desired time step for the next advancemet. If the attemped advancement was unsuccessful, the message contains the desired time step for the repeated attempt for the time step. In any case, the message is received by the PVMEXEC program from each simulation code participating in synchronous coupling. After the global time step size has been computed from the several desired time step sizes, the global time step size is sent to all of the synchronously coupled codes. This message also contains a set of three output times. The output times are the current time for a major edit, the current time for a minor edit and the current time for the writing of plottabel out to the plot file.;

1. Message 8001. Time step control message.

2.3.2 Asynchronous Coupling.

There are two types of messages for asynchronous coupling; 1) data exchange control messages, and 2) output control messages.

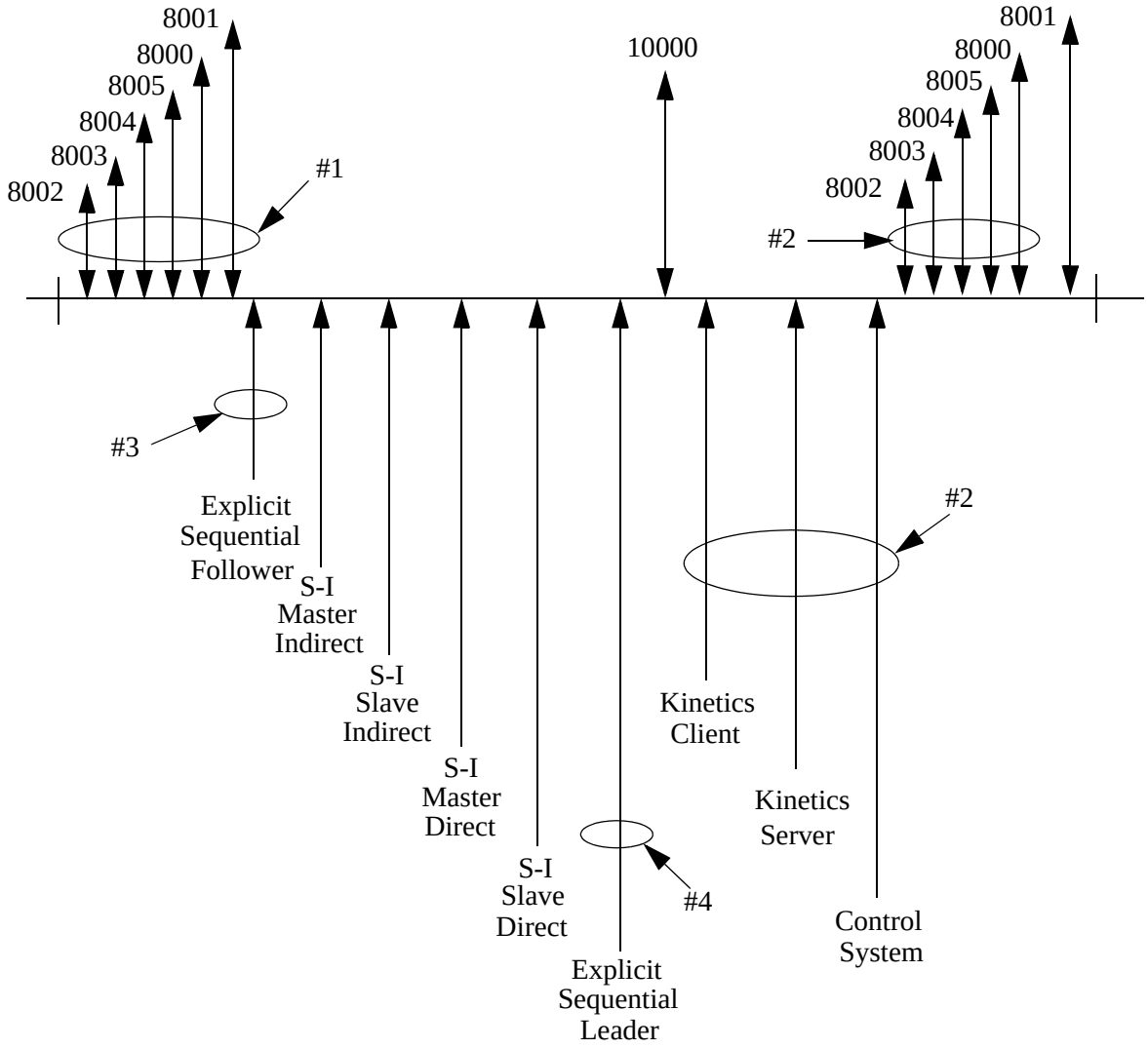
Data exchange control messages are sent at the end of the last time step of a coupling interval to simulation codes that are participating in asynchronous, explicit thermal-hydraulic coupling.

1. Message 8003. These messages are sent to all simulation codes participating in asynchronous, parallel, explicit thermal-hydraulic coupling.

2. Message 8005. These messages are sent to all simulation codes participating in asynchronous, sequential, explicit thermal-hydraulic coupling.

Output control messages are sent to all simulation codes participating in asynchronous coulpling. Asynchronously coupled simulation codes only receive this message at the end of the last time step of an explicit coupling intervals or at restart times.

1. Message 8001. This message contains a set of output times and the explicit coupling interval time. One or more of the times in the message will be different from the set of times in the provious output control message. Asynchronously coupled simulation codes only honor the restart time and the explicit coupling interval..



Notes:

#1 Only at beginning of first time step

#2 Only for successful thermal-hydraulic advancement

#3 Only at beginning of explicit coupling interval

#4 Only at end of explicit coupling interval

Above the line are messages with the executive, below the line at messages between processes

Figure 2.3-1 Sequence of messages during a transient time step.

Appendix C
RELAP5-3D© Coupling Data Items

This appendix describes the data that RELAP5-3D© can send or receive from another simulation code. The appendix is divided into sections describing the data that can be exchanged with other codes for the several different types of coupling.

1 Explicit Thermal-hydraulic Coupling

There are two ways to describe the data that RELAP5-3D© can exchange with other codes for explicit thermal-hydraulic coupling. The first way is to enter pairs of descriptors for each data item. These pairs of descriptors are the same as the descriptors used to tell RELAP5-3D© to add variables to its plot file or to the minor edits. A complete description of these descriptors can be found in Section 4 of Appendix A of Volume II of the RELAP5-3D© manual. The second way to specify data items is to specify a component name and a volume or junction number in the component (the volume or junction number is not the packed number used to specify the data items for the plot file but is the volume and junction number within the component). If the component named is a volume type component, such as a single volume or time dependent volume, a set of fifteen properties for the volume are entered into the database for explicit thermal-hydraulic coupling as if they had been specified individually. The volume variables are pressure (Pa) , vapor fraction (-), liquid fraction (-), vapor specific internal energy (J/kg), liquid specific internal energy (J/kg), total mass quality of noncondensable gas (-), vapor density (kg/m^3), liquid density (kg/m^3), vapor temperature (degrees K), liquid temperature (degrees K), and the mass qualities (-) of the five individual noncondensable gases. If a junction component such as a single junction or a time dependent junction is named, a set of fourteen fluid property variables and fifteen flow rates are entered into the database for explicit thermal-hydraulic coupling as if they had been specified individually. The fluid property variables are, vapor fraction (-), vapor density (kg/m3), vapor specific internal energy (J/kg), vapor non condensable quality (-), vapor velocity (m/sec), liquid fraction (-), liquid density (kg/m^3), liquid specific internal energy (J/kg), liquid velocity (m/sec), and the mass qualities (-) of five individual non condensable gases. The fifteen flow rate variables are total mass flow rate of non condensable gas (kg/sec), the flow rate of vapor internal energy (J/sec), the flow rate of liquid internal energy (J/sec), the mass flow rate of vapor (kg/sec), the mass flow rate of liquid (kg/sec), the volumetric flow rate of vapor (m^3/sec), the volumetric flow rate of liquid (m^3/sec), the flow rate of vapor enthalpy (J/sec), the flow rate of liquid enthalpy (J/sec), the total flow rate of enthalpy in the noncondensable gases (J/sec), and the mass flow rates (kg/sec) of the five individual non condensable gases. The units for the variables as SI, the internal system of units for RELAP5-3D©. These macro definitions are used for both parallel and sequential explicit thermal-hydraulic coupling.

The volume property macro expands to the following quantities:

W1(R) Pressure (Pa).

W2(R) Vapor fraction(-).

W3(R) Liquid fraction (-).

W4(R) Vapor specific internal energy (J/kg).

W5(R) Liquid specific internal energy (J/kg).

W6(R) Mass quality of noncondensable gas (-).

W7(R) Vapor density (kg/m^3).

W8(R) Liquid density (kg/m^3).

W9(R) Vapor temperature (°K).

W10(R) Liquid temperature (°K).

W11(R) Mass quality of first noncondensable gas(-).

W12(R) Mass quality of second noncondensable gas (-).

W13(R) Mass quality of third noncondensable gas (-).

W14(R) Mass quality of fourth noncondensable gas (-).

W15(R) Mass quality of fifth noncondensable gas (-).

The junction property macro expands to the following quantities:

W1(R) Junction vapor fraction (-).

W2(R) Junction vapor density (kg/m^3)

W3(R) Junction specific internal energy (J/kg).

W4(R) Junction mass quality of noncondensable gas (-).

W5(R) Junction vapor velocity (m/s).

W6(R) Junction liquid fraction (-).

W7(R) Junction liquid density (kg/m^3).

W8(R) Junction liquid specific internal energy (J/kg).

W9(R) Junction liquid velocity (m/s).

W10(R) Junction mass quality of first noncondensable gas (-).

W11(R) Junction mass quality of second noncondensable gas (-).

W12(R) Junction mass quality of third noncondensable gas (-).

W13(R) Junction mass quailty of fourth noncondensable gas (-).

W14(R) Junction mass quality of fifth noncondensable gas(-).

W15(R) Junction mass flow rate of noncondensable gas (kg/s).

W16(R) Junction flow rate of vapor internal energy (J/s).

W17(R) Junction flow rate of liquid internal energy (J/s).

W18(R) Junction flow rate of vapor (kg/s).

W19(R) Junction flow rate of liquid(kg/s).

W20(R) Junction volumetric flow rate of vapor (m^3/s).

W21(R) Junction volumetric flow rate of liquid (m^3/s).

W22(R) Junction flow rate of vapor enthalpy (J/s).

W23(R) Junction flow rate of liquid enthalpy (J/s).

W24(R) Junction flow rate of noncondensable enthalpy (J/s).

W25(R) Junction flow rateo of first noncondensable gas (Kg/s).

W26(R) Junction flow rate of second noncondensable gas (kg/s).

W27(R) Junction flow rate of third noncondensable gas (kg/s).

W28(R) Junction flow rate of fourth noncondensable gas (kg/s).

W29(R) Junction flow rate of fifth noncondensable gas(kg/s).

2 Semi-implicit Thermal-hydraulic Coupling.

The data items for semi-implicit coupling are specified solely by the use of macros. The user specifies a component name and volume or junction number in the component. The macros are expanded to lists of variables for two different kinds of messages, the direct messages and the indirect (or implied) messages. The direct messages are exchanged at the end of the computations for semi-implicit coupling and the indirect (or implied) messages are sent during to course of the computations for the semi-implicit solution algorithm. The indirect messages are actually exchanged first during the time step advancement while the direct messages are exchanged at the end of the advancement before the production of output and time step selection. The direct messages specify volume fluid properties or junction fluid properties as described for explicit thermal-hydraulic coupling. The volume fluid properties are the same fifteen properties listed in the previous subsection of this appendix and are sent by the 'master' process of the semi-implicit coupling algorithm. The junction fluid properties are the set of fourteen junction fluid properties listed in the previous subsection of this appendix and are sent by the 'slave' process for the semi-implicit coupling algorithm.

The volume property macro for the direct semi-implicit messages expands to the following properties.

W1(R) Pressure (Pa).

W2(R) Vapor fraction(-).

W3(R) Liquid fraction (-).

W4(R) Vapor specific internal energy (J/kg).

W5(R) Liquid specific internal energy (J/kg).

W6(R) Mass quality of noncondensable gas (-).

W7(R) Vapor density (kg/m^3).

W8(R) Liquid density (kg/m^3).

W9(R) Vapor temperature (°K).

W10(R) Liquid temperature (°K).

W11(R) Mass quality of first noncondensable gas(-).

W12(R) Mass quality of second noncondensable gas (-).

W13(R) Mass quality of third noncondensable gas (-).

W14(R) Mass quality of fourth noncondensable gas (-).

W15(R) Mass quality of fifth noncondensable gas (-).

The junction property macro for direct semi-implicit coupling messages expands the the following quantities:

W1(R) Junction vapor fraction (-).

W2(R) Junction vapor density (kg/m^3)

W3(R) Junction specific internal energy (J/kg).

W4(R) Junction mass quality of noncondensable gas (-).

W5(R) Junction vapor velocity (m/s).

W6(R) Junction liquid fraction (-).

W7(R) Junction liquid density (kg/m3).

W8(R) Junction liquid specific internal energy (J/kg).

W9(R) Junction liquid velocity (m/s).

W10(R) Junction mass quality of first noncondensable gas (-).

W11(R) Junction mass quality of second noncondensable gas (-).

W12(R) Junction mass quality of third noncondensable gas (-).

W13(R) Junction mass quailty of fourth noncondensable gas (-).

W14(R) Junction mass quality of fifth noncondensable gas(-).

The data items for the indirect messages for volume components are the pressure coefficients for each of the coupling volumes and are sent by the ’master’ process. The coefficients for a coupling volume consist of a constant followed by sets of seven coefficients for each coupling junction, each set of seven coefficients representing the derivatives of the pressure in the coupling volume with respect to flow rate of noncondensable gas, with respect to the flow rate of vapor internal energy, with respect to the flow rate of liquid internal energy, with respect to the mass flow rate of vapor, with respect to the mass flow rate of

liquid, with respect to the volumetric flow rate of vapor, and with respect to the volumetric flow rate of liquid in the first coupling junction. The second set of seven coefficients represent the derivative of the pressure in the coupling volume with respect to the seven flow rates in the second coupling junction, etc. The full set of coefficients for the first coupling volume is followed by the set of coefficients for the second coupling volume, etc., until the sets of coefficients for all of the coupling volumes had been specified.

The volume macro for indirect semi-implicit coupling messages expands to the following quantities for each coupling volume:

W1(R) Pressure constant for coupling volume.

W2R) Derivative of pressure in coupling volume with respect to the flow rate of noncondensable gas in the first coupling junction

W3(R) Derivative of pressure in the coupling volume with respect to the flow rate of vapor internal energy in first coupling junction.

W4(R) Derivative of pressure in coupling volume with respect to the flow rate of liquid internal energy in first coupling junction.

W5(R) Derivative of pressure in coupling volume with respect to flow rate of vapor in first coupling junction.

W6(R) Derivative of pressure in coupling volume with respect to flow rate of liquid in first coupling junction.

W7(R) Derivative of pressure in coupling volume with respect to volumetric flow rate of vapor in first coupling junction.

W8(R) Derivative of pressure in coupling volume with respect to volumetric flow rate of liquid in first coupling junction

W9-16(R) Derivatives of pressure in coupling volume with respect to flow rate in second coupling junction.

W17-24(R) Derivatives of pressure in coupling volume with respect to flow rates in third coupling junction.

.....

The data items for the indirect messages for junction components are the sets of seven flow rates for each coupling junction as listed in a previous paragraph and are sent by the ’slave’ process in the semi-implicit coupling algorithm.

The junction macro for indirect semi-implicit coupling messages expands to the following fifteen quantities for each coupliing junction:

W1(R) Junction mass flow rate of noncondensable gas (kg/s).

W2(R) Junction flow rate of vapor internal energy (J/s).

W3(R) Junction flow rate of liquid internal energy (J/s).

W4(R) Junction flow rate of vapor (kg/s).

W5(R) Junction flow rate of liquid(kg/s).

W6(R) Junction volumetric flow rate of vapor (m^3/s).

W7(R) Junction volumetric flow rate of liquid (m^3/s).

W8(R) Junction flow rate of vapor enthalpy (J/s).

W9(R) Junction flow rate of liquid enthalpy (J/s).

W10(R) Junction flow rate of noncondensable enthalpy (J/s).

W11(R) Junction flow rateo of first noncondensable gas (kg/s).

W12(R) Junction flow rate of second noncondensable gas (kg/s).

W13(R) Junction flow rate of third noncondensable gas (kg/s).

W14(R) Junction flow rate of fourth noncondensable gas (kg/s).

W15(R) Junction flow rate of fifth noncondensable gas(kg/s).

3 Kinetics Coupling

The data items to be exchanged for kinetics coupling may be specified using pairs of descriptors for the individual data items to be exchanged or can be specified through the use of macros. The macros that RELAP5-3D© understands for send messages are 'power', 'zone', 'heatstr', 'table', 'cntrlvar', and the name of a volume component. The macro name is followed by a component number. The component number may be followed by an optional '-' and another component number. The optional '-' and additional component number specify a range of component numbers for which data is to be sent. The macro names 'power', 'table', and 'cntrlvar' are expanded to a list of the following variables; 'rkpow' for total reactor power, 'rkpowg' for total reactor decay power, 'rkpowf' for total reactor fission power, 'rkpowk' for the

total power from the decay of fission products, and 'rkpowa' for the total power from the decay of actinides. If the macros 'table' or 'cntrlvar' are used, the five variables in the message have the same value because there is only one source for the value, i.e., the value from the table or control variable. These three macros are used to determine if RELAP5-3D© is the 'client' process or the 'server' task in kinetics coupling. If RELAP5-3D© is to send the variables defined by these three macros, it is the 'server' task in kinetics coupling (i.e., it computes the reactor power using its kinetics model). These variables are used if the 'server' task is using a point kinetics model to describe the power in the reactor core. The macros 'zone' and 'power' are used if RELAP5-3D© is using a multi-dimensional model for the computation of the reactor power and also designates that RELAP5-3D© is the 'server' task in the coupled kinetics simulation. In multi-dimensional kinetics, the reactor core is divided into a number of regions called zones and the power in each zone is computed from the results of the solution of the kinetics model. The 'zone' macro defines a list of the following variables; 'rkoztp' for total reactor power is a kinetics zone, 'rkozfp' for total fission power in a kinetics zone, 'rkozgp' for total decay power in a kinetics zone', 'rkozkp' for total power from the decay of fission products in a kinetics zone, and 'rkozap' for the power from the decay of actinides in a kinetics zone. If the macro name is 'heatstr', the macro is expanded to a single data item, the average temperature in the fuel portion of a heat structure (heat structure are used in RELAP5-3D© to represent the fuel rods in the reactor core). The set of variables defined by the macro is added to the kinetics coupling database for the first component of a component range, followed by the set of variables for the second component in the component range, etc., until a list of variables has been entered for each of the components listed in the range of component numbers. If the component identifier is not a macro name it is assumed to be a volume component name. In this case the component name is expanded to the following list of volume variables: 'voidf' for liquid fraction, 'voidg' for vapor fraction', 'boron' for soluble poison concentration, 'rho' for fluid density, 'rhof' for liquid density,' rhog' for vapor density, 'tempf' for liquid temperature, and 'tempg' for vapor temperature. As with the other macros, a range of component number may be specified.

The 'power' macro expands to the following five variables:

W1(R) Total reactor power (W).

W2(R) Total reactor decay power (W).

W3(R) Total reactor fission power (W).

W4(R) Total reactor power from decay of fission products (W).

W5(R) Total reactor power from decay of actinides (W).

The 'zone' macro expands to the following five variables:

W1(R) Total reactor power in kinetics zone (W).

W2(R) Total reactor decay power in kinetics zone (W).

W3(R) Total reactor fission power in kinetics zone (W).

W4(R) Total reactor power from decay of fission products in kinetics zone (W).

W5(R) Total reactor power from decay of actinides in kinetics zone (W).

The 'heatstr' macro expands to the following quantity:

W1(R) Volume average temperature in heat structure (°K).

The volume properties macro for kinetics coupling expands to the following quantities:

W1(R) Volume liquid fraction (-).

W2(R) Volume vapor fraction (-).

W3(R) Solute concentration (ppm).

W4(R) Fluid density (kg/m^3).

W5(R) Vapor density (kg/m^3).

W6(R) Liquid density (kg/m^3).

W7(R) Liquid temperature (°K).

W8(R) Vapor temperature (°K).

Some of the same macros used in send messages for kinetics coupling are used by RELAP5-3D© in kinetics receive messages: 'power', 'zone', 'volume', and 'heatstr'. The 'power' and 'zone' macros expand to the same list of variables defined send messages. The use of these two macros in receive messages designate that RELAP5-3D© is the 'client' process in the coupled kinetics simulation. The macro 'heatstr' expands to a single value as explained in the previous section. The macro 'volume' expands to the same list of variables as specified for volume components in the previous section. The macros 'volume' and 'heatstr' are used internally in RELAP5-3D© to designate that the properties in the messages are for volumes ot heat structures that are not part of the RELAP5-3D© thermal-hydraulic solution domain but are part of the domain computed by the other coupled code. RELAP5-3D© needs to know this so that it can reserve storage for these variables in its coupling database (they cannot be stored in the regular database because they are not specified in the RELAP5-3D© input file, only by the receive

messages). The macro 'heatstr' in receive messages must also be stored in the coupling database for the same reason.

4 Control System Coupling

The data items for control systems couling are specified by pairs of descriptors. The descriptors that are understood by RELAP5-3D© for control systems coupling are the same descriptors that are used to specify the plot and minor edit variables in RELAP5-3D©. These descriptors may specify any variable that can be used as an input variable to the control blocks in RELAP5-3D©. If a RELAP5-3D© control component is specified in a send message, then the output of the control component is to be sent to the other task. If a RELAP5-3D© control component is specified in a receive message, the value received from the other task is to be used as an input to the designated control component.