Appendix D

Techniques for Connecting Between Modules

Appendix D--Techniques for Connecting Between Modules

One of the key factors in the success of the FRAMES-HWIR Technology Software System is the interconnections between modules in the Multimedia Multipathway Simulation Processor (MMSP). This appendix discusses how file names and variable index parameters are accessed by the modules and provide a variety of techniques to assist in the connection.

Each module has a set of call arguments, which serves two purposes: 1) inform the module about which invocation of a module is occurring and 2) set the permissions for that module to read and write files. The following arguments are passed to a module:

GetArgString(1) Always returns the time a simulation is run
GetArgString(2) Always returns the date a simulation is run
GetArgString(3) Always returns the explicit ssf directory location
GetArgString(4) Always returns the explicit grf directory location
GetArgInt(5) Always returns the storage level for a simulation (see Section 8.2 for details)
GetArgString(6) Always returns the Header Datagroup SSF Filename
This will be hd.ssf
GetArgString(7) Always returns the Site Layout Datagroup SSF Filename
This will be sl<SiteId>.ssf where SiteId will be pulled directly from the Site Based Database (i.e., slwp123123.ssf for Waste Pile with Site Id 123123)
GetArgString(8) Always returns the Current Modules Datagroup SSF Filename
This will be <Medium File Prefix><MedId>.ssf where <Medium File Prefix> is the two letter model and <MedId> is the Medium Identifier for a specific medium. The use of MedId is described in detail later. (e.g., aqlaXYZXYZaq1.ssf would be the aquifers datagroup for site laXYZXYZ aquifer #1 thus the Medium Identifier is laXYZXYZaq1)
GetArgString(9) Always returns the Chemical Properties Datagroup SSF Filename
This will be cp<Model>.ssf for all media except surface water (e.g., cpar.ssf is the chemical properties for the air module). The surface water will use arguments 9, 10, and 11 for the three chemical properties datagroups it will use (e.g., cpWBNStrm.ssf cpWBNLake.ssf, cpWBNWtlnd.ssf). But since this is consistently named most modules will not need to use this argument to get the Chemical Properties.
GetArgString(10 - Last-1) Varies depending on modules but is not used by module developers
GetArgString(Last) The output datagroup a module is expected to write its results.

In general a module is expected to be aware of its media identifier. This media identifier is in the site layout as "<Medium Variable Prefix>Id". For example, the aquifer media identifier is "AquId". For a specific invocation on the module, a module could locate the media identifier by using the GetArgString(8) call. This call will always return the currently executing module's site simulation file (SSF) file name, which contains the module's media identifier, which is shown between the <Medium File Prefix> at the beginning and ".ssf" at the end. The Medium Variable Prefixes and Medium File Prefixes are described in Table D.1.

Table D.1 Mapping of Medium to File Prefix and Variable Prefix

Medium	Medium File Prefix	Medium Variable Prefix
Landfill	lf (SSF) sr (GRF)	Src
Waste Pile	wp (SSF) sr (GRF)	Src
Surface Impoundment	si (SSF) sr (GRF)	Src
Aerated Tank	at (SSF) sr (GRF)	Src
Land Application Unit	la (SSF) sr (GRF)	Src
Vadose Zone	vz	Vad
Aquifer	aq	Aqu
Air	ar	Air
Watershed	ws	WSSub
Surface Water (or Waterbody Network)	sw	WBN
Farm Foodchain	ff	FarmFC
Terrestrial Foodchain	tf	TerrFC
Aquatic Foodchain	af	AquaFC
Ecological Exposure	ee	EcoRcp
Human Exposure	he	HumRcp
Ecological Risk	er	EcoRcp
Human Risk	hr	HumRcp

For clarity in the rest of this appendix, the following abbreviations are used for the different behaviors of modules:

- O = "One at a Time" modules that expect to be executed for each medium of that type.
- A = "All at once" modules that execute all their media at once.
- M = "Modified All At Once" modules that simulate some subset of all the media with each execution.

The current behavior of the models requires a number of different techniques to be used in the FRAMES-HWIR Technology Software System. Two queries must be answered for each technique. The first query is "Give me the information on medium (O) or media (A and M) that impact my module." The

second query is "Give me the informaton on the medium (O) or media (A and M) that are impacted by my module." The first query looks from the current medium/media being simulated and back toward the source to determine some information. The second query looks from the current medium/media being simulated toward the risk. The techniques and example code for both the forward and backward looking queries are given below for the six different techniques required by the modules in the FRAMES-HWIR Technology Software System.

Code similar to the examples or "pseudocode" is expected to be used by all modules of all types. All Medium Identifiers need to have the Medium Prefix and File Extension added when they are going to be used as a Datagroup Name. For example, if an aquifer has a Medium Identifier of "XYZXYZaq1," then its SSF Data Group would be "aqXYZXYZaq1.ssf," and its Global Results Files (GRF) Data Group would be "aqXYZXYZaq1.grf". Table D.1 shows the relationship between Medium, Medium File Prefix, and Medium Variable Prefix for the Site Layout Data Group.

```
!Standard Preamble Code
!The psuedo code leans heavily towards being written in FORTRAN but I assume the C/C++
! programmers can adapt.
!Get Site Layout datagroup name
Call GetArgString(7,SLGroup)
!Get Medium Identifier
Call GetArgString(8,MedId)
!Remove the first two characters of the MedId and the last 4
MedId=MedId(3:Len(MedId)-4)
!Find media Index and Sub Media Index
NumMed=ReadInt(SLGroup,"Num<Med>","")
!For all the media of this type
do I=1,NumMed
!Get the Medium Identifier
        ReadString(SLGroup,"<Med>Id","",TestMedId)
        if (TestMedId.eq.MedId) MedIndex=I
end do
! At the end of this code the variable SLGroup, MedId and MedIndex are defined
```

Given the behavior of the current modules, a number of different Site Layout Data Group access techniques will need to be used. Table D.2 shows the modules using the Medium File Prefix and the O, A, or M specifier to identify the behavior of each module. The table also shows which of the techniques will be needed for each module. For example, Table D.2 states that for the Aquifer Module, the following techniques might be needed:

1) Vadose Zone -> Aquifer uses Technique 1 Backward
2) Aquifer -> Surfacewater uses Technique 3 Forward
3) Aquifer -> Farm Foodchain uses Technique 2 Forward
4) Aquifer -> Human Exposure uses Technique 2 Forward

For the waterbody network or surface water module the following techniques might be needed:

1) Aquifer -> Waterbody Network uses Technique 3 Backward
2) Air -> Waterbody Network uses Technique 3 Backward
3) Watershed -> Waterbody Network uses Technique 4 Backward
4) Waterbody Network -> Farm Foodchain uses Technique 6 Forward
5) Waterbody Network -> Terrestrial Foodchain uses Technique 6 Forward
6) Waterbody Network -> Aquatic Foodchain uses Technique 6 Forward
7) Waterbody Network -> Ecological Exposure uses Technique 6 Forward
8) Waterbody Network -> Human Exposure uses Technique 6 Forward

Table D.2 Mapping of Modules to Techniques for Reading Data [(a)]

		From														
		O	O	O	O	O	O	O	O	A	M	A	A	A	A	A
To		lf	wp	si	at	la	vz	aq	ar	ws	sw	ff	tf	af	ee	he
O	vz	1	1	1	1	1										
O	aq	?1	?1	?1	?1	?1	1									
O	ar	1	1	1	1	1										
A	ws								2							
M	sw							3	3	4						
A	ff							2		5	6					
A	tf									5	6					
A	af										6					
A	ee										6		5	?5		
A	he							2	2	5	6	5		5		
A	er														5	
A	hr															5

(a) O = "One at a Time"
A = "All at once"
M = "Modified All At Once"
1-6 Represent Techniques 1-6 described below
(? Represents not sure of connection)
wp,la,lf,..etc. Represent the two letter prefix for the modules

For each of the techniques detailed below, the <Src> and <src prefix> represents a medium or media that impacts a module, and <Out> and <out prefix> represents a medium or media that a module impacts. <Med> and <med prefix> represent the medium represented by the current module. <Med>, <Src>, and <Out> are the Medium Variable Prefix described in Table D.1. <med prefix>, <src prefix>, and <out prefix> are the Medium File Prefixes also in Table D.1.

D.1 Technique 1: O -> O

D.1.1 Technique 1, Backward Looking

To look "backward" in the site layout, the program must read the media's information in the Site Layout Data Group. Because the data are stored with a bias toward backward looking, this is the simpliest technique that will be used.

```
!How many of these media impact my medium
NumSrc=ReadInt1(SLGroup,"<Med>Num<Src>","",MedIndex)
do i=1,NumSrc
        !What are the indices for those media
        SrcIndex(i)=ReadInt2(SLGroup,"<Med><Src>Index","",MedIndex,i)
        !What are the fractions for those media
        SrcFract(i)=ReadReal2(SLGroup,"<Med><Src>Fract","",MedIndex,i)
        !What are the Names of the datagroups for those media
        ReadString1(SLGroup,"<Src>Id","",SrcIndex,SrcGroup(i))
        !Add module prefix and .grf toSrcGroup
        SrcGroup(i)="<src prefix>"+SrcGroup(i)+".grf"
end do
! Now SrcGroup(i) is an array of datagroups to read from

!Example use of data:
! As an example read the flux rate from all the source
Con=1
Time=1
do I=1,NumSrc
        Flux(i,Con,Time)=ReadReal2(SrcGroup(i),"Flux","g/yr",Con,Time)
end do
```

D.1.2 Technique 1, Forward Looking

To look forward with this type of connection, the program needs to read the Site Layout Data Group information for the module's output medium. Remember that the Site Layout Data Group is intended to be used by any module that needs it.

```
! How many of a particular output media type are there
NumAllOut=ReadInt(SLGroup,"<Out>Num","")
! Loop through all the output media  type
```

```
NumOut=0
do I=1,NumAllOut
        !How many sources of the same type as my module are there
        NumSrc=ReadInt1(SLGroup,"<Out>Num<Med>","",i)
        !Loop through all the sources of this type for the output media type
        do j=1,NumSrc
                !Is this media impacted by my module
                SrcIndex=ReadInt2(SLGroup,"<Out><Med>Index","",i,j)
                !Store the indices of module of this type that use my output
                If (SrcIndex.eq.MedIndex) then
                        OutIndex(NumOut)=I
                        OutCount=NumOut+1
                End if
        end do
end do
!At the end of this you have NumOut which represents the number of a type
!of output media that are impacted by your module and OutIndex which are the
!indices of those output media.

!Example use of data:
! As an example if the vadose zone needed to know gw flow direction
do I=1,NumOut
        GwDir(i)=ReadReal1(SLGroup,"GWDir","degrees",OutIndex(i))
end do
```

D.2 Technique 2: O -> A

D.2.1 Technique 2, Backward Looking

This is slightly more complicated than the O->O because the SrcGroup array in this technique has the additional index of media. The additional index requires an additional loop in reading the pertainent Site Layout Data Group information.

```
!How many of these media impact my medium
NumMed=ReadInt(SLGroup,"Num<Med>","")
do i=1,NumMed
        NumSrc=ReadInt1(SLGroup,"<Med>Num<Src>","",i)
        do j=1,NumSrc
                !What are the indices for those media
                SrcIndex(i,j)=ReadInt2(SLGroup,"<Med><Src>Index","",i,j)
                !What are the fractions for those media
                SrcFract(i,j)=ReadReal2(SLGroup,"<Med><Src>Fract","",i,j)
                !What are the Names of the datagroups for those media
                ReadString1(SLGroup,"<Src>Id","",SrcIndex,SrcGroup(i,j))
                !Add module prefix and .grf toSrcGroup
```

```
                SrcGroup(i,j)="<src prefix>"+SrcGroup(i,j)+".grf"
        end do
! Now SrcGroup(NumMed,NumSrc) is an array of datagroups to read

!Example use of data:
! As an example read the air concentration results
!Con and Time represent the Chemical and Time indices of interest
! Even though for HWIR 99 only one air simulation is expected be
Con=1
Time=1
do i=1,NumMed
        do j=1,NumSrc
                Flux(i,j,Con,Time)=ReadReal2(SrcGroup(i,j),"SoilConc", +
                        "g/yr",Con,Time)
        end do
end do
```

D.2.2. Technique 2, Forward Looking

This technique is the same as that described in Section D.1.2.

D.3 Technique 3: O -> M

D.3.1 Technique 3, Backward Looking

The backward-looking version of this technique is a mixture of the backward-looking versions of Techniques 1 and 2 (see Sections D.1.1 and D.2.1, respectively). There is an additional index on the SrcGroup variable; however, the technique makes use of the MedIndex read in the Standard Preamble Code. <SubMed> in this psuedo code represents the media that are executed together for a given media. To be explicit <Med> would be replace with WBN and <SubMed> would be replaced by Rch for the waterbody network module.

```
!How many of these media impact my medium
NumSubMed=ReadInt1(SLGroup,"<Med>Num<SubMed>","",MedIndex)
do i=1,NumSubMed
        NumSrc=ReadInt2(SLGroup,"<Med><SubMed>Num<Src>","",MedIndex,i)
        do j=1,NumSrc
                !What are the indices for those media
                SrcIndex(i,j)=ReadInt3(SLGroup,"<Med><SubMed><Src>Index", +
                "",MedIndex,i,j)
                !What are the fractions for those media
                SrcFract(i,j)=ReadReal3(SLGroup,"<Med><SubMed><Src>Fract", +
                        "",MedIndex,i,j)
                !What are the Names of the datagroups for those media
                ReadString1(SLGroup,"<Src>Id","",SrcIndex,SrcGroup(i,j))
```

```
            !Add module prefix and .grf toSrcGroup
            SrcGroup(i,j)="<src prefix>"+SrcGroup(i,j)+".grf"
        end do
! Now SrcGroup(i,j) is an array of datagroups to read from by NumSubMed and NumSrc

!Example use of data:
! The example would be similar to that of O->A Backward
```

D.3.2 Technique 3, Forward Looking

This technique is the same as that in Section D.1.1, Technique 1, Forward Looking.

D.4 Technique 4: A -> M

D.4.1 Technique 4, Backward Looking

This technique is simpler than the the backward-looking versions of Techniques 2 or 3 because the indices on the SrcGroup are reduced. This technique also makes use of the MedIndex read in the Standard Preamble Code. <SubMed> in this psuedo code represents the media that are executed together for a given media.

```
!How many of these media impact my medium
NumSubMed=ReadInt1(SLGroup,"<Med>Num<SubMed>","",MedIndex)
do i=1,NumSubMed
        NumSrc(i)=ReadInt2(SLGroup,"<Med><SubMed>Num<Src>","",MedIndex,i)
        do j=1,NumSrc(i)
            !What are the indices for those media
            SrcIndex(i,j)=ReadInt3(SLGroup,"<Med><Src>Index","",MedIndex,i,j)
            !What are the fractions for those media
            SrcFract(i,j)=ReadReal3(SLGroup,"<Med><Src>Fract", +
            "",MedIndex,i,j)
            !What are the Names of the datagroups for those media
            ReadString1(SLGroup,"<Src>Id","",SrcIndex,SrcGroup(i,j))
            !Add module prefix and .grf toSrcGroup
        end do
SrcGroup=<src prefix>+".grf"
! Now SrcGroup is the datagroup to read from for results from this media

!Example use of data:
!A Waterbody Network reading loading from the Watershed
!Con and Time represent the Chemical and Time indices of interest
Con=1
Time=1
! For each waterbody in this waterbody network
do I=1,NumSubMed
```

```
        ! For each source to that waterbody
        do j=1,NumSrc(i)
            ! Read loadings
            SoilCon(i,j,Con,Time)=ReadReal3(SrcGroup,"Loading",
+                                    "g/y",SrcIndex(i,j),Con,Time);
        end do
    end do
```

D.4.2 Technique 4, Forward Looking

To look forward with this type of connection, the program needs to read the Site Layout Data Group information for the module's output medium.

```
    ! Loop through all the output media  type
    do I=1,NumMed
        NumOut(i)=0
    end do
    ! How many of a particular output media type are there
    NumAllOut=ReadInt(SLGroup,"<Out>Num","")
    do I=1,NumAllOut
        !How many Sub-Media are there in the output
        NumSubOut=ReadInt1(SLGroup,"<Out>Num<SubOut>","",i)
        do j=1, NumSubOut
            !How many sources of the same type as my module are there
            NumSrc=ReadInt2(SLGroup,"<Out>Num<Med>","",i,j)
            !Loop through all the sources of this type for the output media type
            do k=1,NumSrc
                SrcIndex=ReadInt3(SLGroup,"<Out><Med>Index","",i,j,k)
                OutIndex(SrcIndex,NumOut(i))=j
                OutSubIndex(SrcIndex,NumOut(i))=k
                NumOut(i)=NumOut(i)+1
            end do
        end do
    end do
    ! At the end of this you have NumOut(i) which represents the number of a type
    ! of output media that are impacted by your module, OutIndex(NumMed,NumOut) which
    ! are the indices of those output media, and OutSubIndex(NumMed,NumOut) which is the
    ! Sub Media Index.  Explicitily OutIndex would be WBNIndex and OutSubIndex would
    ! be WBNReach Index

    !Example use of data:
    ! As an example if watershed needed the waterbody type for each waterbody reach
    do I=1,NumMed
        do j=1,NumOut(i)
            ReadString2(SLGroup,"WBNRchType","",
+                                OutIndex(i,j),OutSubIndex(i,j))
        end do
    end do
```

D.5 Technique 5: A -> A

D.5.1 Technique 5, Backward Looking

This is a simpler case of reading the Site Layout Data Group that is used in the backward-looking version of Technique 4 because the loop on NumSubMed is not needed.

```
!How many of these media impact my medium
NumMed=ReadInt(SLGroup,"Num<Med>","")
do I=1,NumMed
        NumSrc(i)=ReadInt1(SLGroup,"<Med>Num<Src>","",i)
        do j=1,NumSrc
                !What are the indices for those media
                SrcIndex(i,j)=ReadInt2(SLGroup,"<Med><Src>Index","",i,j)
                !What are the fractions for those media
                SrcFract(i,j)=ReadReal2(SLGroup,"<Med><Src>Fract","",i,j)
        end do
end do
SrcGroup=<src prefix>+".grf"
! Now SrcGroup is the datagroup to read from for results from this media

!Example use of data:
!A Farm Foodchain reading soil concentrations from a Watershed result
!Con and Time represent the Chemical and Time indices of interest
Con=1
Time=1
! For each farm foodchain
do I=1,NumMed
        ! For each source to that foodchain
        do j=1,NumSrc(i)
                ! Read soil concentration
                SoilCon(i,j,Con,Time)=ReadReal3(SrcGroup,"SoilConc",
+                                       "g/kg",SrcIndex(i,j),Con,Time);
        end do
end do
```

D.5.2 Technique 5, Forward Looking

To look forward with this type of connection, the program needs to read the Site Layout Data Group information for the module's output medium.

```
! Loop through all the output media  type
do I=1,NumMed
        NumOut(i)=0
end do
! How many of a particular output media type are there
```

```
NumAllOut=ReadInt(SLGroup,"<Out>Num","")
do I=1,NumAllOut
        !How many sources of the same type as my module are there
        NumSrc=ReadInt1(SLGroup,"<Out>Num<Med>","",i)
        !Loop through all the sources of this type for the output media type
        do j=1,NumSrc
                SrcIndex=ReadInt2(SLGroup,"<Out><Med>Index","",i,j)
                OutIndex(SrcIndex,NumOut(i))=j
                NumOut(i)=NumOut(i)+1
        end do
end do
!At the end of this you have NumOut(i) which represents the number of a type
!of output media that are impacted by your module and OutIndex(i,j) which are the
!indices of those output media.

!Example use of data:
! As an example if farm foodchain needed the human receptors type
do I=1,NumMed
        do j=1,NumOut(i)
                FarmFract(i,j)=ReadReal2(SLGroup,"HumRcpType","",
+                                       OutIndex(i,j))
        end do
end do
```

D.6 Technique 6: M -> A

D.6.1 Technique 6, Backward Looking

This technique is more complicated than the backward-looking version of Technique 5 because the SrcGroup array in this technique has the additional index of media. The additional index requires an additional loop in reading the pertainent Site Layout Data Group information.

```
!How many of these media impact my medium
NumMed=ReadInt(SLGroup,"Num<Med>","")
do I=1,NumMed
        NumSrc(i)=ReadInt1(SLGroup,"<Med>Num<Src>","",i)
        do j=1,NumSrc(i)
                !What are the indices for those media
                SrcIndex(i,j)=ReadInt2(SLGroup,"<Med><Src>Index","",i,j)
                SrcSubIndex(i,j)=ReadInt2(SLGroup,"<Med><Src><SubSrc>Index",
+                                       "",i,j)
                !What are the fractions for those media
                SrcFract(i,j)=ReadReal2(SLGroup,"<Med><Src>Fract",
+                                       "",i,j)
                ReadString2(SLGroup,"<Src>Id","",i,j,SrcGroup(i,j))
```

```
                !Add module prefix and .grf toSrcGroup
                SrcGroup(i,j)="<src prefix>"+SrcGroup(i,j)+".grf"
            end do
        end do
        ! Now SrcGroup(i,j) is an array of datagroups to read from by NumMed and NumSrc

        !Example use of data:
        ! For example a Farm Foodchain reads Waterbody Network Results water concentrations
        ! For all farms
        !Con and Time represent the Chemical and Time indices of interest
        Con=1
        Time=1
        do I=1,NumMed
            !For all sources to those farms
            do j=1,NumSrc(i)
                WatConc(i,j)=ReadReal3(SrcGroup(i,j),"WatConc","g/ml",SrcIndex(i,j),
+                                   SrcSubIndex(i,j),Con,Time)
            end do
        end do
```

D.6.2 Technique 6, Forward Looking

To look forward with this type of connection, the program needs to read the Site Layout Data Group information for the module's output medium. It is basically the same as the forward-looking portion of Technique 5, with an additional index for the sub-medium. This technique is only to be used by the waterbody network (surface water) module, and thus the sub-medium is the Waterbody Reach.

```
        ! Loop through all the output media  type
        do I=1,NumMed
            NumOut(i)=0
        end do
        ! How many of a particular output media type are there
        NumAllOut=ReadInt(SLGroup,"<Out>Num","")
        do I=1,NumAllOut
            !How many sources of the same type as my module are there
            NumSrc=ReadInt1(SLGroup,"<Out>Num<Med>","",i)
            !Loop through all the sources of this type for the output media type
            do j=1,NumSrc
                SrcIndex=ReadInt2(SLGroup,"<Out><Med>Index","",i,j)
                SrcSubIndex=ReadInt2(SLGroup,"<Out><SubMed>Index","",i,j)
                If (SrcIndex.eq.MedIndex) then
                    OutIndex(SrcSubIndex,NumOut(i))=j
                    NumOut(i)=NumOut(i)+1
                end if
            end do
        end do
        !At the end of this you have NumOut(i) which represents the number of a type
```

```
!of output media that are impacted by your module and OutIndex(i,j) which are the
!indices of those output media.

!Example use of data:
! As an example if  the farm foodchain needed the human receptors
!  type
do I=1,NumMed
        do j=1,NumOut(i)
                FarmFract(i,j)=ReadReal2(SLGroup,"HumRcpType","",
+                                        OutIndex(i,j))
        end do
end do
```