107030200-TD005-R02

Spallation Neutron Source

Instrument Systems Data Acquisition File Formats Technical Reference Manual

September 2005

INSTRUMENTS SYSTEMS
DATA ACQUISTION
FILE FORMATS TECHNICAL REFERENCE MANUAL

Date Published: September 2005

Contents

1 Scope

This document describes the intermediate data file format know as pre-nexus files. This is the format in which data is saved as files by the SNS data acquisition system.

2 Definitions and conventions

2.1 Definitions

Satellite Computer: A satellite computer is a computer that responds to control message from a master computer (control computer). The satellite computer communicates with controllers to affect their actions.
Control Computer: The control computer is responsible for the coordination of an experiment. It sends control messages to various satellite computers that are directly responsible for control of hardware.
NEXUS file: Nexus is a neutron community standard binary file format based on HDF.

2.2 Abbreviations, acronyms, and symbols.

DAS: Data Acquisition System
SNS Spallation Neutron Source
TBD: To Be Determined
TOF: Time-Of-Flight
GUI: Graphical User Interface
XML: Extended Markup Language
RTDL: Real Time Data Link

2.3 Conventions

2.3.1 Control Protocol XML element overview

2.3.1.1 The following is an XML element. It consists of a beginning tag, an end tag and content. (the tagname and attributes comprise the beginning tagname , i.e. <TAGNAME AttName="AttValue"> Content </TAGNAME>

2.3.1.2 Tag and attribute names **canno**t *contain white space characters. In addition all names are* **case sensitive**. *For example the tag name SampleEnvironment is not the same as sampleenvironment. For clarity the word may consist of multiple words i.e. SampleEnvironment.*

2.3.1.3 All XML files must contain a standard xml file description header and can contain ***only one*** *parent or root element.*

2.3.1.4 The root element must contain a "version" attribute with the file version number.

3 Introduction

Intermediate or Pre-Nexus files are generated by the data acquisition system and stored on a server class computer known and the data/file server. To gain an understanding into the reasons for the particular hardware architecture and reasons for the file formats one must first be introduced to some important concepts about the data acquisition system. The data acquisition system at SNS is a distributed control network with a central control point known appropriately as the control computer. The control computer does not directly control hardware; rather it directs the sequence of an experiment either through a local GUI or through python scripts. Hardware control is the responsibility of many satellite computers. For example the chopper satellite computer is responsible for the control and monitoring of choppers, while the sample

environment satellite computer would control and monitor aspects related to the sample environment. Each satellite computer makes known the control variables for a particular hardware configuration that it is responsible for through the use of an XML file known as a configuration file which contains a list of what is known as "friendly names" and their properties. The friendly name is how the satellite computer and the control computer make known to each other what parameter needs to be controlled or monitored. This same friendly name is used in the python scripting language. For example suppose one wanted to change the sample temperature to 30 degrees Kelvin. The friendly name for this control variable may be known as "sampletemp". In a GUI on the control computer one would find an edit box with the title "sampletemp". Typing in the value 30 would change this value appropriately. This change in value would also be logged by the control computer and recorded in a local file. If one was doing an experiment and directed the control computer to "save"[i], the control computer would transfer this information along with all other registered variables to the data/file server. This method is used for all variables associated with slow controls where variables are monitored and change slowly with respect to the time between neutron pulses (16.6msec). For high bandwidth data, such as neutron scattering data the source of the data is the preprocessor cluster. During an experimental run the preprocessor cluster sends event scattering data to the data/file server. On the data/file server the data is either histogrammed or directly stored in memory. In both cases, when the data/file server is directed to save the data, andy remaining data in volatile memory is flushed to disk and the file closed. Because of its size, the data is sent as a binary file. This is also true of other "real time" data such as chopper phases that may be measured every pulse. Information describing the data in the binary files (meta data) is found in an xml file. The content of the different files both binary and xml is described in the sections below. The separation of low and high bandwidth data into different files types allows great flexibility in what is finally rendered to a Nexus file.

4 Directory Structure/Files

Access to data files is in general restricted to the user that created the data files. This restriction implies that a user's data files be placed in directory whose access is limited to those who log into the data/file server appropriately. In this document, the high level directory is denoted by *proposalID*, where *proposalID* is the proposalID the current operator is operating under. The network shared name is not determined at this time nor is its location on the disks directory structure. For example *proposalID* in this context may refer to the actual location c:\userfiles\proposalID or to a network share mapping \\servername\proposalID. Refer to newer revisions of this document for information on this. Access to the data files in the *proposalID* directory is read only except for processes logged in with write-modify authorization.
At present this is limited to processes running locally on the data file server. It is anticipated that this process will need network access to a metadata data base for the inclusion of information not made available by the control computer. This mechanism is not defined in this revision. Within the proposalID directory there are a set of files, and a set of folders.
Files:
There are two files in the proposalID directory; these are "xxx_beamtimeinfo.xml" and "xxx_cvlist.xml". (xxx is an instrument specific identifier known as the instrument neumonic. One should not infer that xxx limits the neumonic to three characters). The

[i] Data is also periodically sent to the data/file server while the run is in progress. The frequency of the transfer depends upon the amount of data being transferred and the available link bandwidth.

"xxx_beamtimeinfo.xml" file contains information about the current beam time session including a list of operators, a list of runfile names created during this run, and any other pertinent information related to the entire beam time for a particular user. The "xxx_cvlist.xml" file (cv stands for control variable) contains information about the friendly names currently in use. A detailed description of the content of these files appears in the section below.

Folders:
Within the proposalID directory there are a variable number of folders denoted as xxx_*runnumber*. (xxx is the instrument neumonic). *Runnumber* refers to a unique number assigned to each run by the data acquisition system. A run in general is a particular setting of experimental parameters and scattering data that will be saved as a file set. However the run number does not increment unless an explicit save data command is give either via script or GUI button. The run number is automatically assigned by the data acquisition system and is not editable by the user. There is a separate and unique xxx_*runnumber* folder for each experimental run. Within each xxx_*runnumber* folder there are the following files with the file extension xml: (Note that all files in the folder start with the folder's name *xxx_runnumber*).
*xxx_runnumber*_runinfo.xml, *xxx_runnumber*_cvinfo.xml. and *xxx_runnumber*_alarms.xml. The runinfo file contains information about the run, such as a complete list of files generated for the run (excluding itself), operator name, run duration, binary file format etc. Some of the information is not unique and is also found in the cvinfo file. The cvinfo file contains all the information generated by the satellite computers, and the standard set of control variables made available by the control computer. It contains information such as set points, data logs, time focusing information etc. The alarms file contains a list of any alarm information generated by the data acquisition system, including power failures, and equipment or communication failures. If there were no alarms, then the xxx_runnumber_alarms.xml file will not exist. The format of these xml files is described in a section below.

Depending upon the acquisition mode, there will be one or more binary files with file extension dat in the same xxx_runnumber folder. If acquisition is run in histogramming mode, at least one binary file is generated, that is the xxx_runnumber_neutron_histo.dat file. This file is a flat binary file containing the histogrammed data of the run. Other histogrammed data related to other control variables may also be present. These would be listed as xxx_runnumber_cvname_histo.dat. (Where cvname is the friendly name of the control variable). For example one may be interested in a histogram of all the chopper phases. These would appear in a set of n binary files, one for each phase. Information needed to interpret the binary information is found in the runinfo file, under the tag "fileformats". More information on this appears in the section below.
If the data acquisition system is operated in event mode, there are at least two binary files created in the xxx_*runnumber* folder. These files are xxx_*runnumber*_neutron_events.dat file and the xxx_*runnumber*_neutron_events_pulseid.dat file. The neutron events file is a binary file of every neutron event detected, while the neutron_events_pulseid is a file that allows one to associate a pulse id with a given set of neutron events that appears in the neutron_events file. There may be additional pairs of dat files. For example one may be interested in the pulse by pulse value of chopper phases, or even the subpulse values of applied stress. Each variable would have a pair of files, a list of the data, and a pointer list which would allow one to determine which pulse id goes with what binary data. The name of the files would be xxx_*runnumber*_name_events.dat and xxx_*runnumber*_name_events_pulseid.dat. Where name

is a defined name found in this document. The format of these files is described in the section below.

Additional xml files may be included in the xxx_*runnumber* directory for logging of particular control variables. This data would appear in a file xxx_*runnumber*_cvname_log.xml (The xml files are typically used for data that would be sampled every one second or longer. Whether or not there is a log file associated with a variable can be found by examining the variable's entry in the xxx_*runnumber*_cvinfo.xml file, which informs the user about the data associated with the control variable.

5 File Formats, syntax.

5.1 Binary files:

5.1.1 Event Mode.

Event Mode consists of set of file pairs, a data file, and a file that associates a pulseid with the data. The pulse id is the 64 bit value that is sent on the accelerator RTDL network prior to a pulse being generated. It uniquely identifies each pulse. Each binary file has the extension .dat The first file is a binary file containing a list of event data. For neutron data, the name and format is fixed. The file name for neutrons is xxx_*runnumber*_neutron_events.dat and this file contains a flat list of events stored in little-endian format as the following C structure.

Uint32 timestamp (in 100nsec ticks)

Uint32 positionid (see appendix A for more information).

The position id is interpreted via information contained in another file which contains enough information to recover space information on the pixel (mapping file). The timestamp is the actual time of flight. The second file is a file that allows one to associate a pulse id with every neutron event. The name of this file for neutrons is fixed and given by xxx_*runnumber*_neutron_events_pulseid.dat. The format of every pulseid file is fixed and is binary data stored in little-endian format as a sequence of the following C structure

Uint64 pulseid

Uint64 mempointer (the upper four bits of the mempointer are reserved, setting of the upper bit is an indication of data associated with a particular condition encoded by the three remaining bits. (The meanings of these bits is undefined but are guaranteed to be zero as of this revision)

Mempointer is an index into the corresponding event file and contains the **zero based** index of the **first event structure**[ii] that is associated with the given pulseid.

A number of other files may also be saved as binary file. Some of these may have different event file formats (i.e. not UINT32). The format of any binary file (other than the pulseid file which is fixed) must be included in the runinfo.xml (see below for a description) file associated with the run. The naming of the file is required to follow the convention xxx_*runnumber*_name_events.dat , where xxx is the instrument identifier, *runnumber* is the run number (as define in the previous section) and name is the variable's friendly name (these names must be defined in this document). The ordering of the event structure and its

[ii] Note, the index is not a byte offset, rather a structure index. So, for the case of a neutron file a mempointer of 10 points to the 11th entry at byte offset 80 from the beginning of the file. To find a byte offset, one would require the use of the sizeof operator on the neutron event structure.

endianness are required to follow the format of the neutron case, i.e. little endian, with the timestamp proceeding the data. Has of this revision the following names are defined:
1) neutron (all neutron events, including beam monitors operating in event mode)
2) chopperphase (all chopper phase errors in 100nsec ticks)
3) beamcurrent (proton beam current for pulses in TBD units).

5.1.2 Histogram Mode.

Histogram Mode consists of at least one file a binary file of histogrammed neutron data. The file name for neutrons is xxx_*runnumber*_neutron_histo.dat Has of this revision all neutron scattering detectors (not beam monitors) must be contained in one histogram. The size of the file may change depending upon the experimental setup. Information needed to extract the binary data is found in the xxx_*runnumber*_runinfo.xml file. The histogram file for neutron data contains n*t uint32 values arranged as a C array of n pixel i.d.s where n is the total number of pixel i.d.s in the detector system, by t time channels. The number of pixels and number of time channels are listed in the xxx_*runnumber*_runinfo.xml file. The C array is stored as array[n][t], i.e. the t index varies the fastest. In Matlab t would be equivalent to the number of rows. Additional histogram files may be saved. Information about these files must be included in the xxx_*runnumber*_runinfo.xml file. Each beam monitor has its own histogram file. The file name is *xxx_runnumber*_beammon*yyy*_histo.dat, where *yyy* is the beammonitor's id number found in the BeamMonitorInfo element of the runinfo file. Again the information required to interpret the histogram is found in the runinfo file. Currently all beam monitors are either run in histogram or event mode with the default mode being histogram. One will typically find histo beam monitor files along with event neutron scattering data.

5.2 XML Files:

There are three required xml files associated with each run, and two xml files associated with each beam time allotment to a user. This section describes the format of these xml files. In addition, the formats of additional xml log files (which are optional) are also described.

5.2.1 xxx_*runnumber*_runinfo.xml

The runinfo.xml file contains information about a run number. The run number is a number assigned by the data acquisition system to uniquely identify a run that occurs on a particular instrument. Any run at the facility is uniquely identified by the xxx_runnumber value where xxx uniquely identifies the instrument the experiment is run on. The format of runinfo.xml is as follows.

```
[1]<?xml version="1.0"?>
[2]<RunID instrument="xxx" runnumber="runnumber"  version="versionnumber">
[3]<GeneralInfo operatorname="name" proposal="id">
        <Title> run title  </Title>
        <Notes> user supplied information </Notes>
        <ELogName> elog file name/entry </ELogName>
        <ScriptID> script id </ScriptID>
        <RunStatus> run status info </RunStatus>
        <SpecialDesignation> background, normalization or other </SpecialDesignation>
</GeneralInfo>
[4]<DetectorInfo id="dectorid" >
```

<CalibrationID> calibrationID </CalibrationID>
<MaxScatlPixelID>maximum scattering pixelID </MaxScatPixelID>
<MaxBMPixelID>maximum beam monitor pixelID </MaxBMPixelID>
<Scattering id="moduleID" type="detector type" description="verbose description" name="freindlyname">
<Mode combine="true or false"> histogram or event </Mode>
<NumTimeChannels width="xxxusec" scale="linear or log"
startbin="usec" endbin="usec">
number of time channels </NumTimeChannels>
<MappingFile> filename </MappingFile>
<NumPixels> total pixels number,pixel offset </NumPixels>
<Position>
<SampleDetDis value="distance" units="units reference="T.B.D"
cvname="associated cv name"/>
<DetAngle value="angle" units="units" reference="T.B.D"
cvname="associated cv name" />
</Position>
</Scattering>
<BeamMonitorInfo id="beammonitor id" type="type" description="verbose description" name="friendlyname">
<Mode combine="true or false"> histogram or event </Mode>
<NumTimeChannels width="xxxusec" scale="linear or log"
startbin="usec" endbin="usec">
number of time channels </NumTimeChannels>
<NumPixels> number of pixels, pixel offset </NumPixels>
<MappingFile> filename </MappingFile>
<Position TBD info />
</BeamMonitorInfo>
</DetectorInfo>
[5]<OperationalInfo>
<AcceleratorPulses> number of neutron pulses </AcceleratorPulses>
<PCurrent units="*units*"> integrated proton current units =TBD</PCurrent>
<TotalVetos> number of pulses vetoed </TotalVetos>
</OperationalInfo>
[6]<SampleInfo Name="*samplename*">
</SampleInfo>
[7]<DateTime>
<StartTime> *run startdate and time* </StartTime>
<EndTime> *enddate and time* </EndTime>
<LastUpdate> *last update date and time* </LastUpdate>
</DateTime>
[8]<ProcessList>
processID1
processID2
…..other active processes
</ProcessList>
[9]<FileList>
xxx_runnumber_cvinfo.xml

xxx_runnumber_alarms.xml
xxx_runnumber_neutron*id*_histo or event.dat*
xxx_runnumber_neutron*id*_event_pulseid.dat
xxx_runnumber_bmon*id*_histo or event.dat
xxx_runnumber_bmon*id*_event_pulseid.dat*
xxx_runnumber_other_histo or event.dat
xxx_runnumber_neutron_event_pulseid.dat
Other file names in this directory.
</FileList>
[10]<Associations>
<ScanInfo sequencenumber="sequence in scan" cv="cvparm1,cvparm2.."> *scan association ID* </ScanInfo>
<BackGround> *background run number* </BackGround>
<Normalization> *normalization run number* </Normalization>
</Associations>
[11]<FileFormats>
<neutron dims="n,m" vartype="uint32,uint32>
<Names> tof, pixelid </Names>
<Units> 100nsec,none </Units>
</neutron>
<cvname dims="n,m,…" vartype="uint32,double,….>
<Names> tof, voltage, … </Names>
<Units> 100nsec, mvolts </Units>
</cvname>
……*other variables that are found as .dat files(except _pulseid) in the file list*
</FileFormats>
</RunID>

*Note that after run completeion, that all intermediate event files are placed in one event file with the name xxx_runnumber_neutron_event.dat and xxx_runnumber_bemon_event.dat. Histograms remain separate.

Tag descriptions:

1) xml header is required exactly as shown.
2) RunID—required tag with three required attributes. "instrument" is the neumonic used to identify the instrument that the run is done on, the "runnumber" is the DAS assigned run number for the run, while "version" is the file version for this file. The combination *xxx_runnumber* must exactly match the folder name.
3) GeneralInfo—required tag with two required attributes: "operatorname" is the name of the current operator known to the control computer. The neumonic NA will be placed in this field if it is unknown. "proposal" is the proposal identifier or NA if this is unknown. There are two optional content tags defined: <Notes> which contains user supplied information about the run, and <ELogName> which contains information about any electronic log information pertinent to the run. The content tag title, is required, and contains the title of this particular run. If the run is run from a script, then <ScriptID> must contain the ID value of the Python script. <RunStatus> contains information about the run's status. This may be stalled, running, paused, etc. The exact semantics of the content is T.B.D. Further information about errors may be found in the alarms file associated with the run. <SpecialDesignation> is an optional tag that is present whenever

the current run is a normalization or background file. The field may not be current until the final file save. The content of this element contains information about what is special about this run. The content is fixed by the DAS system and typically would arise from a radio button or other list. The possible content values are TBD. Currently the defined names are.. Normalization_VAN, Normalization_H2O, Background_ECAN, Background_NSAM . for a vanadium normalization, water normalization, empty can background and background with no sample respectively.

4) DetectorInfo—required tag with one required attributes. The single required attribute is "id". This number is an identifier that can be used to find out additional information about the detector system. There are three required content tags, and an optional beam monitor tag. However if there are beam monitors, then this tag will be present.

<MaxScatPixelID> is a required tag that contains the value that all position/pixel IDs are less than this number. The minimum ID is always 0 for scattering detectors.

<MaxBMPixelID> is required if there are beam monitors. It contains the pixel ID that all beam monitor pixel IDs from all beam monitors are less than. The starting pixel ID for beam monitors is required to be 0x40000000 (hexadecimal).

<Scattering> is a required tag and contains information about the main set of neutron scattering detectors. In most cases, there will only be one scattering element, but it is possible to have more if such a separation makes sense. For example if an instrument has a moveable detector and a high angle scattering bank, it may make sense to have two sections, one describing the moveable detector, and a second describing the non-moveable detector. If the detector system is non-movable, then there are three requirement sub-elements, otherwise there are four. There are four attributes: "moduleID" is a number uniquely identifying this scattering bank, "type" which provides a description of the detector type, the possible values are TBD, "description" is a short easily understood description of the detector bank and "name" is the friendly name used by the python scripting language.

<NumTimeChannels> is a required tag that contains, when running in histogram mode, four attributes: "width"(if the scale attribute=linear) is the width of the time bins in microseconds or (if the scale attribute=log) the $\Delta t/t$ value. "scale" is either linear or log. "startbin" is the value of the minimum time (in usec) of the first time bin. Because of this, startbin is not the bin center of the first bin rather it is the minimum time of flight value the DAS system will record for this run. "stopbin" is the value of the maximum time (in usec) of the last time bin. Because of this, stopbin is not the bin center of the last bin rather it is equivalent to the maximum time of flight value the DAS system will record for this run. The content of this tag is the number of time bins. This is redundant information since this value can be calculated from the values of the attributes. When running in event mode, this element contains only the two attribute startbin and stopbin and contains no content as such information is meaning less in the context of event mode.<NumPixels> is a required tag containing the number of pixels described by this entry along with the pixel offset (Typically this will be zero). The values are comma separated.

<Mode> is a required tag that is either histogram or event. If the mode is event, the a single optional attribute may be included. The combine attribute indicates when true (default if not included) that all event mode files will be combined at the end of a run to a single event file, neutron_event for scattering listings, and beammon_event for beam monitors. A single true overrides any false attributes.

<MappingFile> is an optional tag that contains the url or file name of the file used to map pixel IDs to user coordinates.
<CalibrationID> is a required tag that contains the current calibration ID of the detector. This ID is a unique identifier that associates a calibration file or set of files with the instruments detector system.
<Position> Movable detector systems contain the "Position" tag that contains information about the detector's position. This is used with the mapping file to calculate the physical coordinates of the detector's pixels. Two sub-element tags are defined as of the revision.
<SampleDetDis> is the sample to detector distance. The reference attribute is TBD but would typically have the reference pixel ID the measurement is taken to.
<DetAngle> is the sample angle relative to the reference value. The reference nomenclature is TBD.
Typically both these values are equivalent to control variables which can be found in the cvinfo list. If they are, then the tags must contain the cvname attribute. Generally these would be used with a mapping file to determine the physical coordinates of detector pixels for movable detector systems.
<BeamMonitorInfo> if there are beam monitors in the system, this is a required tag and contains information about the beam monitor. There are four attributes: "moduleID" is a number uniquely identifying this beammonitor, "type" provides a description of the beam monitor type, the possible values are TBD, "description" is a short easily understood description of the detector and "name" is the friendly name used by the python scripting language. There are five requirement sub-elements.
<NumTimeChannels> is a required tag that contains, when running in histogram mode, four attributes: "width"(if the scale attribute=linear) is the width of the time bins in microseconds or (if the scale attribute=log) the $\Delta t/t$ value. "scale" is either linear or log. "startbin" is the value of the minimum time (in usec) of the first time bin. Because of this, startbin is not the bin center of the first bin rather it is the minimum time of flight value the DAS system will record for this run. "stopbin" is the value of the maximum time (in usec) of the last time bin. Because of this, stopbin is not the bin center of the last bin rather it is equivalent to the maximum time of flight value the DAS system will record for this run. The content of this tag is the number of time bins. This is redundant information since this value can be calculated from the values of the attributes. When running in event mode, this element contains only the two attribute startbin and stopbin and contains no content as such information is meaning less in the context of event mode.<NumPixels> is a required tag containing the number of pixels described by this entry along with the pixel offset (Typically this will be zero). The values are comma separated.
<Mode> is a required tag that is either histogram or event. All scattering elements must have the same mode!
<MappingFile> is an optional tag that contains the url or file name of the file used to map pixel IDs to user coordinates.
<CalibrationID> is a required tag that contains the current calibration ID of the detector. This ID is a unique identifier that associates a calibration file or set of files with the instruments detector system.
<Position> The format of this section is TBD.

5) OperationalInfo—Required tag containing information on various operational parameters. There are no required attributes. There are four required content tags:

<Mode> is a required tag that describes the data collection mode for all neutron scattering detectors. The two valid content names are histogram and event. The content name mixed is reserved for future use.
<MonitorMode> is an optional tag that describes the data collection mode for all beam monitors. The two valid content names are histogram and event. The content name mixed is reserved for future use.
<AcceleratorPulses> is a required tag that contains the total number of accelerator pulses delivered during the run time of the experiment.
<PCurrent units="*units*"> is a required tag that contains the integrated proton current delivered to the target during the run. It has one required attribute, "units" which describes the units used with the content uses the class,units format in this instance, this is fixed at "E_M,uA" where the class refers to electromagnetic and units is microamps.
<TotalVetos> is a required tag that contains the total number of vetos due to all sources.
6) SampleInfo—Required tag containing information about the sample. There is one require attribute and no required content. "Name" required attribute which is the name of the sample measured during the run. If this is unknown then the neumonic NA shall be placed here.
7) DateTime—Required tag containing time and date information about the experiment. There are six required content tags:
<StartTime> required tag with no required attributes containing the experiment's starting time and date in the ISO8601 format YYYY-MM-DDTHH:MM:SS-xxx
<EndTime> required tag with no required attributes containing the experiment's end time and date in the ISO8601 format YYYY-MM-DDTHH:MM:SS-xxx
<LastUpdate> required tag with no required attributes containing the time and date the current run files where updated. When the run ends, it contains the end time. During operation it contains the last update time in the ISO8601 format YYYY-MM-DDTHH:MM:SS-xxx
8) ProcessList—Required tag containing a list of process Id numbers. Each GUI or DAS process on the control computer is assigned a fixed ID number to uniquely identify the application. This number must be included in the process list for each GUI active during the experiment. The ID list is white character delimited.
9) FileList—Required tag containing a list of the files contained in this run's directory. Every file except runinfo.xml itself must be listed here. The file list is white character delimited.
There are a minimum number of files depending upon the experimental collection mode and whether or not there were alarm conditions during the run. If the operational mode is histogram, then the following files will be listed:

xxx_*runnumber*_cvinfo.xml
xxx_*runnumber*_alarms.xml (only included if there are alarms)
xxx_*runnumber*_neutron(*id)*_histo.dat.* Where *id* is the id attribute value found in the detectors listing in the detector section above.

If there are beammonitors they are listed as

xxx_runnumber_bmon(*id)*_histo.dat* Where bmonid is the string bmon+"beammonitorid". There is a separate file for each beam monitor listed in the detector section.

**id* occurs only if there is more than one corresponding entry in the detector section. Beammonitors and scattering entries are treated separately in this respect.

If the operational mode is event, then the following files will be listed:

xxx_*runnumber*_cvinfo.xml
xxx_*runnumber*_alarms.xml (only include if there are alarms)
xxx_*runnumber*_neutron_event(*id*).dat*
xxx_*runnumber*_neutron_event(*id*)_pulseid.dat*
xxx_*runnumber*_beammon*(id*)_event.dat*
xxx_*runnumber*_beammon_event(*id*)_pulseid.dat*
xxx_*runnumber*_chopperphase_event.dat
xxx_*runnumber*_chopperphase_event_pulseid.dat

*(id) appears only if there is more than one entry in the detector section. During an active run, multiple event files may be present. If the mode attribute "combine" is set to true (default), then all event files of a given type will be combined into a single event file (neutron_event and/or beammon_event) without an (*id*) appended to the filename. Note that additional pairs of of event mode data describing other variables mayl also be present. In all cases, xxx_*runnumber* refers to the unique instrument, run number tag for this experiment. (Note alarms file may not be present if there have been no alarms or errors.)

10) Associations—Optional element containing a list subelements with information about other associations with this run number. If "Associations" is found, then at least one of the subelements must be present. ScanInfo is the associated scan information for this file. Other files in the scan will also have the same ID, and the ID is the runnumber of that begins the scan. There are two optional attributes: 1) cv contains a comma separated list of friendly names that represent the parameter(s) the scan is run over. 2) sequencenumber is the sequence in the scan, starting at one the run represents. This may be missing in implicit scans where a variable may change during a run without saving the data data set. This may occur during event mode. In this case it is up to the analysis software to associated event data with the parameter that changes. BackGround is the run number or comma separated run numbers of the file(s) having the background to be used with this file. Normalization is the run number or comma separated run numbers of the file(s) having the normalization to be used with this file. There may be more than one background or normalization file associated with a run. If this is the case, then the *runnumbers* are separated by commas. Information about each of these files can be found by looking at the SpecialDesignation section.

11) FileFormats—Required tag containing a list of formatting information needed to read binary files (i.e. .dat files). There are no required attributes. Every …histo.dat file and every ….event.dat file listed in the FileList section must also contain an entry here. ….event_pulseid.dat files have a fixed format so do not need to be included here. The listing format is as follows: There is a one to one correspondence between the tagnames in the content and the file names listed in the FileLists section. The tagname is found by stripping off the xx-runnumber prefix and the _histo.dat or _event.dat postfix. For example , the file name xxx_runnumber_neutron_histo.dat must have the tag name <neutron> (Case must be retained). If the file contains a listing xxx_runnmber_othervar_event.dat there must be a tag name <othervar>. Each of the tag names in the content has two required attributes "dims", and "vartype". "dims" contains a list of the lengths of each dimension. (Comma separated), while "vartype" contains the variable type of the array. Structures are designated by the keyword "struct" followed by a comma separated list of variable types contained in the structure. If a structure is

defined, care must be taken to avoid padding as this is assumed to be turned off. Arrays are not allowed to be inside a structure. An example of a structure definition is for the neutron event, i.e. varitype="struct,uint32,uint32", while vartype="struct,uint32,float" might describe stress information. The content of each tag is require to have two sub-tags <Names> and <Units>. <Names> supply information about each dimension and/or structure elements. <Units> supply information for the same.

5.2.2 xxx_*runnumber*_cvinfo.xml

The cvninfo.xml file contains information about the control variables (or friendly names). The file is organized by satellite computer names. Each friendly name has one entry in the xml file and contains information about its value, whether or not an external log file is associated with the variable, the variables units and statistical data. The format of cvinfo.xml is as follows.

[1]<?xml version="1.0"?>
[2]<RunID instrument="*xxx*" runnumber="*runnumber*" version="*versionnumber*">
[3]<chopper_xxx>
<*Friendlyname1* deviceID="deviceID" device="device" value="value"
starttime="date/time" units="class,neumonic" ave="average" stdev="stedev"
max="maxvalue" min="min value">
<![CDATA[
timestamp value
timestamp value
]]>
Or if file type
<LogFile type="histogram or xmllog or event>
filename
</LogFile>

</*Friendlyname1*>
<*Friendlyname2* deviceID="deviceID" device="device" value="value"
starttime="date/time" units="class,neumonic" ave="average" stdev="stedev"
max="maxvalue" min="min value">
<![CDATA[
timestamp value
timestamp value
]]>
Or if file type
<LogFile type="histogram or xmllog or event>
filename
</LogFile>

</*Friendlyname2*>
….other chopper_xxx friendly names
</chopper_xxx>
[4]<samplenv_xxx>
<*Friendlyname1* deviceID="deviceID" device="device" value="value"
starttime="date/time" units="class,neumonic" ave="average" stdev="stedev"
max="maxvalue" min="min value">

<![CDATA[
timestamp value
timestamp value
]]>
Or if file type
<LogFile type="histogram or xmllog or event>
filename
</LogFile>

</*Friendlyname1*>
….other sample environment friendly names
</sampleenv_xxx>
[5]<ancillary_xxx>
<*Friendlyname1* deviceID="deviceID" device="device" value="value" starttime="date/time" units="class,neumonic" ave="average" stdev="stedev" max="maxvalue" min="min value">
<![CDATA[
timestamp value
timestamp value
]]>
Or if file type
<LogFile type="histogram or xmllog or event>
filename
</LogFile>

</*Friendlyname1*>
….other ancillary friendly names
</ancillary_xxx>
[6]<epics>
<*Friendlyname* deviceID="deviceID" value="value" datetime="date/time" units="unitsclass, unitsneumonic" ave="average" stdev="stedev" max="maxvalue" min="min value">
<LogData>
timestamp,value
timestamp,value
</LogData>

Or if file type
<LogFile type="histogram or xmllog or event>
filename
</LogFile>
</*Friendlyname*>
….other epics friendly names
</epics>
[7]<das>
<das.mode deviceID="das version" value="event or histogram" timestamp="date/time" units="class,neumonic" />

<das.monitormode deviceID=”das version” value=”event or histogram” Timestamp=”date/time” units=”class,neumonic” />
<das.counts deviceID=”das version” value=”total counts” timestamp=”date/time” units=”class,neumonic” />
<das.runtime deviceID=”das version” value=”total runtime” timestamp=”date/time” units=”class,neumonic” />
<das.totalvetos deviceID=”das version” value=”total runtime” timestamp=”date/time” units=”class,neumonic” />
<das.totalpulses deviceID=”das version” value=”total runtime” timestamp=”date/time” units=”class,neumonic” />
</das>
<detector>
<*friendlyname*.counts detectorID=”id” value=”counts” timestamp=”date/time” units=”class,neumonic” />
< *friendlyname*.mode deviceID=”id” value=”event or histogram” timestamp=”date/time” units=”class,neumonic” />
<*friendlymane*.tofbin deviceID=”id” value=”value” timestamp=”date/time” units=”class,neumonic” />
< *friendlymane* .tofstart deviceID=”id” value=”minimum tof value” timestamp=”date/time” units=”class,neumonic” />
< *friendlymane* .tofend deviceID=”id” value=”maximum tof value” timestamp=”date/time” units=”class,neumonic” />
< *friendlymane* .toftype deviceID=”id” value=”linear or log” timestamp=”date/time” units=”class,neumonic” />

…repeat above for each scattering listing and for each beam monitor listing in the detector info section of the runinfo.xml file.
</detector>

There may be additional information for additional satellite computers, including the user computer and other ancillary computers.
</RunID>

Tag descriptions:

1) xml header is required exactly as shown.
2) RunID—required tag with three required attributes. “instrument” is the neumonic used to identify the instrument that the run is done on, the “runnumber” is the DAS assigned run number for the run, while “version” is the version number of the file. The combination *xxx_runnumber* must exactly match the folder name.
3,4,5) Satellite computer tag names—required tags for each satellite computer. (Note, the xxx refers to the process controlling the actual variable, there may be more than one process related to any of the satellite computers). Listed are the required examples of typical active satellite computers. There may be other listings for other satellite computers, for example if a user computer was active, then there would be a tag entry <user>. Each satellite computer section contains elements whose tag names are the friendly names is has made known to the system. There is one element entry for each friendly name and each element contains any data or points to the data associated with that friendly name. Each friendly name tag has two required attributes: “starttime” and

"units", and may have seven other optional attributes: "deviceID", "device", "ave","stdev","max", and "min" and "value". "deviceID" is the device ID which is a unique identifier (from the configuration file) used to identify the equipment that generated the data."device" is a user recognizable name from the device. "value" is the l value for this friendly name, "starttime" is the date and time in ISO8601 format when the run started according to the controlling process. "units" contain two comma separated neumonics, the units class followed by the units. "ave",stdev","max" and "min" are other statistical entries that may be included. If there is only one entry in a log file (i.e. a constant during a run), it may be listed in the "value" attribute. If statistical data is listed rather than a log of the data is given, then the "value" attribute would be the value at the start of the run. There is no required content for the friendly name tags. Content will only exist if the variable in question is logged, if this is the case, then the log data appears in a ![CDATA[section. The log data appears as a whitespace separated date, time, value triplets. The time format is pseudo ISO8601 YYYY-MM-DD white space HH:MM:SS.millisec. If the logged data appears in a separate file, the file name is listed as the content of the sub-element <LogFile> If the data is binary, formatting information must appear in the xxx_runnumber_runinfo.xml file. Although one may either have a log of the data or statistical information along with the starting value, one or the other or both sets of data must be present.

6) epics—required tag with no required attributes. The epics element contains information gained from the accelerator epics network and any safety related variables for a particular instrument. Included in the epics information would be pulse currents, moderator temperature and accelerator fault information. The content of this tag is identical to the satellite computers, i.e. a number of friendly name sub-elements.

7) das—required tag with no required attributes. The das element contains information gained from the preprocessor cluster and is related to the variables associated with the das system. These variable names are fixed. In addition, their nature is such that (for this revision) they are never listed in the log format. The timestamp attribute is in standard IEC8601 format YYYY-MM-DDTHH:MM:SS-xxx. There following tags are defined: "das.counts" (this value is the sum of all scattering detector counts, i.e. it does not include beam monitors)., "das.mode", "das.monitormode" "das.runtime", "das.totalvetos", and "das.totalpulses"..

8) detector—required tag with no required attributes. The detector element contains information about the various scattering banks and beam monitors. The following tags are defined: "*friendlyname*.mode" "*friendlyname*.counts", "*friendlyname*.tofbin", "*friendlyname*.tofstart", "*friendlyname*.tofend", "*friendlyname*.toftype". Tof tags provide information needed for historgramming. The counts tag is the total counts from this detector bank or beam monitor. As of this revision, there is only one scattering bank allowed. Mode is either histogram or event. All detectors of the same class must have the same mode, ie. all beam monitors must either be histogram or event and scattering detectors must all be histogrammed or event mode.

5.2.3 xxx_*runnumber*_alarms.xml

The alarms.xml file contains information about alarms and/or errors that occurred during the run. If no alarms or errors occurred during a run, then this file will be absent. Alarms can be associated with a particular control variable or an alarm indication from a satellite computer indicating user intervention will be or is required. Note that removal of an alarmed condition

is considered an alarm. This provides a record of when an alarm condition was removed as well as set. The format of alarms.xml is as follows.

```
[1]<?xml version="1.0"?>
[2]<RunID instrument="xxx" runnumber="runnumber" version="versionnumber">
[3]<Alarm001 DateTime="datetime">
        <FriendlyName> friendlyname or NA </FriendlyName>
        <ValueAtAlarm> value or NA </ValueAtAlarm>
        <AlarmType> typeID <AlarmType>
        <AlarmMessage> message <AlarmMessage>
</Alarm001>
<Alarm002 DateTime="datetime">
        <FriendlyName> friendlyname or NA </FriendlyName>
        <ValueAtAlarm> value or NA </ValueAtAlarm>
        <AlarmType> typeID <AlarmType>
        <AlarmMessage> message <AlarmMessage>
</Alarm002>
….other alarms denote as Alarmxxx where xxx are sequential numbers.
<Error00n DateTime="datetime">
        error information
</Error00n>
</RunID>
```

Tag descriptions:

1) xml header is required exactly as shown.

2) RunID—required tag with three required attributes. "instrument" is the neumonic used to identify the instrument that the run is done on, the "runnumber" is the DAS assigned run number for the run, while "version" is the file version of the file. The combination *xxx_runnumber* must exactly match the folder name.

3) Alarm001—each alarm will have an entry denoted as Alarmxxx where xxx is a number from 001 to 999. The Alarm tag has one required attribute "DateTime" which is the time and date of the alarm in ISO8601 format. Alarms are numbered sequentially in the order they are received.

Each alarm has three required sub-elements: "FriendlyName", "ValueAtAlarm" and "AlarmType". There is one optional sub-element: "AlarmMessage". The content of the FriendlyName sub-element is the friendlyname associated with the alarm. NA is used if the alarm is not related to a friendlyname. The content of the ValueAtAlarm is the value of the control variable at the time of alarm or NA if it not associated with an alarm. The units must be the same as in the friendlyname's unit as listed in the xxx_runnumber_ cvinfo.xml file. The AlarmType is the alarms type. Details about this number can be found in the SNS document 107030200-TD002-Rxx. The optional AlarmMessage contains a message string with further information about the alarm.

In addition to alarms, there may be one or more errors. Typically these are associated with the script itself. The content of this element is T.B.D.

This file along with the <RunStatus> element of the runinfo.xml file provide information about the current run state.

5.3 XML Files (in username directory):

There are two files within the username directory; these are "xxx_beamtimeinfo.xml" and "xxx_cvlist.xml". (xxx is an instrument specific identifier).

5.3.1 xxx_beamtimeinfo.xml

The xxx_beamtimeinfo.xml file contains information about the current beam time. Other than the run numbers, tehemeta-data of this file is not generated by the DAQ system but is read by the DAQ system and included in the file. The format of xxx_beamtimeinfo.xml is as follows.

```
1<?xml version="1.0"?>
2<Instrument name="xxx"  version="versionnumber">
3<Proposal>
        <ID>proposal id </ID>
        <Title> proposal title </Title>
 </Proposal>
4<Users>
        <PI> pi login name </PI>
        <Operators> login name1, login name2,  … </Operators>
</Users>
5<Runnumbers>
        <Current> current runnumber </Current>
        <Previous> runnumber1, runnumber2, ….. </Previous>
</Runnumbers>
6<Schedule>
        <Start> scheduled start date and time </Start>
        <End> scheduled end date and time </End>
</Schedule>
</Instrument>
```

Tag descriptions:

1) xml header is required exactly as shown.

2) Instrument—required tag with two required attributes. "name" is the instrument identifier, while "version" is the file version number.

3) Proposal—required tag containing information about the proposal. There are two required sub-elements in the content. <ID> and <Title> whose content is the proposal ID or beam time ID if there is no proposal, and the title of the proposal respectively.

4) Users—required tag containing information about the users during the scheduled beam time. There are two required sub-elements in the content. <PI> and <Operators> whose content is the login name of the principle investigator, and a comma separated list of the login names of authorized operators respectively.

5) Runnumbers—required tag containing information about the current and previous run numbers associated with this beam time. There are two required sub-elements in the content. <Current> and <Previous>. Current, is current active run number, or if there is no active run the runnumber that will be assigned to the next saved data set. Previous is a comma separated list of previous run numbers associated with the proposal or beam time. All runnumbers in the previous list will have been saved to disk.

6) Schedule—required tag containing information about the scheduled time for the user. There are two required sub-elements in the content. <Start> and <End> whose content is the users scheduled start and scheduled stop date/tim ein ISO 8601 format respectively.

5.3.2 xxx_cvlist.xml

The xxx_cvlist.xml file contains a listing of every control variable active at the time of the inquiry. It simply provides a list of active friendly names and does not include other information about the variable such as units, this information is found in the xxx_runnumber_cvinfo.xml file. This cvlist file is useful whenever an outside application needs to build a python script file for controlling or setting up an experiment. The format of this file is very similar to the xxx_runnumber_cvinfo.xml file found in the xxx_runnumber sub-directory. All active variables that have a scriptable name are required to be included in this file, and every entry in this file is guaranteed to have a value listing in the xxx_runnumber_cvinfo.xml file. The format of xxx_cvlist.xml is as follows:

[1]<?xml version="1.0"?>
[2]<Instrument name="*xxx*" valid="true or false" version="*versionnumber*" >
[3]<chopper_xxx>
Friendlyname1
Friendlyname2
….other chopper friendly names
</chopper_xxx>
[4]<samplenv_xxx>
Friendlyname
….other sample environment friendly names
</sampleenv_xxx>
[5]<ancillary_xxx>
Friendlyname
….other ancillary friendly names
</ancillary_xxx>
[6]<epics>
Friendlyname
….other epics friendly names
</epics>
[7]<das>
tofbin
tofstart
tofend
toftype
mode
counts
runtime
totalvetos
totalpulses
</das>
</Instrument>

Tag descriptions:

1) xml header is required exactly as shown.
2) Instrument—required tag with three required attributes. "name" is the instrument identifier, "valid" indicates whether the list is currently valid, and "version" is the file

version number. Processes reading this list should verify that this attribute is true before creating a control script with any of the friendly names.
3,4,5,6) Satellite computer tag names—required tags for each satellite computer. (Note, the xxx refers to the processed controlling the actual variable, there may be more than one process related to any of the satellite computers). Listed are the required examples of typical active satellite computers. There may be other listings for other satellite computers, for example, if a user computer was active then there would be a tag entry <user>. Each satellite computer element has a white character separated list of friendly names that it controls. Data, data logs, units and other information for every entry may be found in the xxx_runnumber_cvinfo.xml file. Because friendly name are scriptable this file can be used as a source for useable script names for the current instrument configuration.
7 das tag names---required tags for predefined systems that are not satellite computers. Each element has a white character separated list of friendly names that are associated with the system.

6 Appendix A pixelIDs:

Pixel ID is a 32 bit number used to identify a detector element. The detector element can be from a scattering bank, or a beam monifor. In general a mapping file is needed to determine the spatial location of any pixel id. In many cases, the pixel ID is related to the x-y local coordinates of detector especially x-y detectors. For example a Brookhaven 2D detector operating at normal resolution has pixel id's mapped as pixelid=256x+y where y varies from 0-255 and x varies from 0 to 303.
Certain bits of the pixelID are reserved. Bit 31 (most significant bit) is set if there was an error condition in the determination of the neutrons position. This bit is useful for diagnostics. Typically these are suppressed at the preprocessor. The next three bits (30,29 and 28) are interpreted differently depending up bit 30. If bit 30 is not set, then they are interpreted at pixel ID bits. Bit 30 is clear for all scattering type detectors, providing for about 2 billion unique scattering pixels. If bit 30 is set, indication a special designation for the detector then bits 29 and 28 are interpreted as the type of special detector. Beam monitors are one type of special detector. Bits 29 and 28 are both zero for beam monitors. This implies that 0x40000000 is the base offset for all beam monitors. Bits 0-27 are then interpreted as pixelIDs for beam monitors.