SIAM Team Discussion of "MOOS Behaviors Required by SSDS"

Tom O'Reilly

June 6, 2003

Date: June 6, 2003
Present: Dan Davis, Kent Headley, Bob Herlien, Tom O'Reilly
Group discussed document entitled "MOOS Behaviors Required by SSDS", authored by John Graybeal in May 2002:
http://abalone.shore.mbari.org/siam/Requirements/BehaviorsExpectedBySSDS.pdf

A. General discussion notes

1. "ISI ID" applies to any "source" of data to be logged by SIAM and ingested into SSDS. These sources included but are not limited to an Instrument. An application which synthesizes and logs data from multiple instruments should also be assigned an ISI ID.

2. ISI DevicePacket includes a sequenceNumber field. It is not sufficient to simply reset sequenceNumber to '0' following a service restart; instead sequenceNumber must persist across resets. Otherwise, cannot detect malfunction/packet outage prior to reset.

3. Service software version ID is part of service metdata.

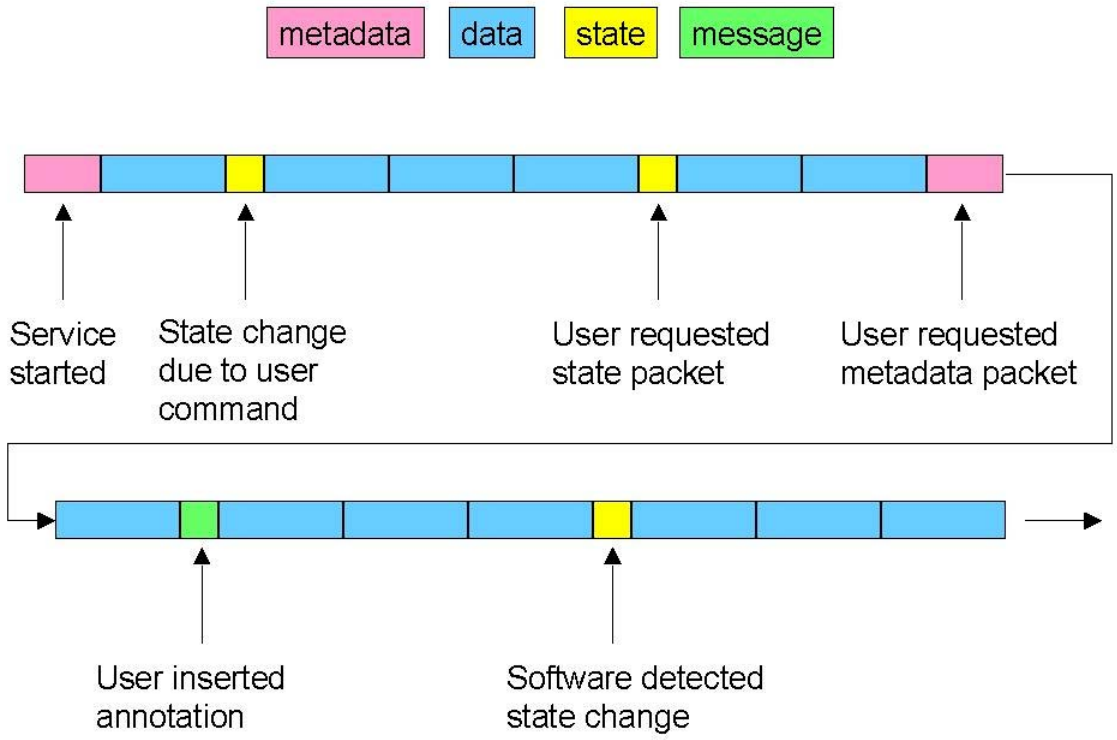
4. Discussed whether ISI ID should have mnemonic element. Consensus is "no"; difficult to come up with useful mnemonic given desired brevity of ID (ID accompanies every packet).

5. Discussed the "scope" of an ISI ID. In the context of an instrument, the ISI ID applies to any attached sensor which itself does not have a puck. Information about these non-pucked sensors must be included in the parent instrument's metadata.

6. A node (i.e. a platform) itself has an ISI ID and a log, to which packets are written. The log contains packets which list the instruments installed on the node. An updated packet is written to the node log when an instrument is removed or added. (Are these lists in the form of properties?)

7. ISI model is that each ISI ID generates a single data stream. The data stream contains metadata/state packets interleaved with data packets, as shown in the time-sequence

diagram below:



8. An instrument's metadata packet includes the ID of the parent node (platform), to enable efficient determination of the deployment platform of an arbitrary instrument data packet.

9. Each metadata packet in the above diagram is the "full" metadata for the instrument.

10. Are "state packets" distinct from "metadata packets"?

11. Each data packet includes the sequence number of the relevant metadata packet.

12. Science has specified that "raw data" always be available. What this really means is that any transformation of raw data to telemetry must be loss-less.

13. Instrument data packet can be parsed based on description within instrument's metadata. Metadata will specify parse token, field widths, types, names, etc. For special cases which do not fit into the description scheme, metadata will include a URL locator for parse/processing algorithms.

14. Instrument's position on platform (e.g. it's depth) is not specified in the instrument's metadata (e.g. depth is not a property stored in the puck). Rather, instrument position is a property of the parent node. This position is not necessarily accurate enough for

scientific purposes. SIAM will provide tools for operators to specify instrument position on mooring.

15. Will be able to query SIAM log based on packet sequence numbers as well as time range.

16. Filling out metadata forms for instrument puck is a critically important process, but SIAM does not control the metadata content. Rather, the metadata content is human-entered, and may entail policies and procedures that are outside the scope of the SIAM project.

B. Comments on "MOOS Behaviors Required by SSDS".

SIAM comments shown in red.

Requirement 1: A standard, flexible format must be used to describe all *data records* (whether science or operational data, events, or *metadata*) received by SSDS.

Required to support scaling MOOS to many sensors and moorings (and important to minimize programming)

Yes, we agree.

Requirement 2: All received data records and data stores must be accompanied by:
1. unique *data source ID* (including versions, for SW)
2. time of data ingest into ISI/MOOS
3. reference to (i.e., a way to access) the unique *record content description metadata*

Provision of this information must be automatic and guaranteed if data management is to have any hope of scaling up.

Every packet will contain data source ID (= ISI ID), packet creation time, sequence number, and sequence number of relevant metadata packet. Metadata packet will also contain software version ID, and packet description metadata. If more than one data packet type, data packet will contain indicator of which type it is.

Requirement 3: Process must be supplied that guarantees unique data source IDs identify different hardware components. (The data source ID for hardware is referring to the same thing as the ISI Device ID, by definition and assumption.)

Needs to be embedded in overall MOOS architecture, for same reasons as above (must be automatic and guaranteed if data management is to have any hope of scaling up).

We assume that ISI IDs for instruments will be handed out by SSDS database.

Requirement 4: Process must be supplied that guarantees unique data source IDs identify different software components and versions (software changes on ISI must change data source ID)

Needs to be embedded in ISI's SW architecture somehow, or else anyone can just put another SW component out there with an errant software ID. (Is a different version of software a different ID? Would be nice to have ID and version both.)

We assume that ISI IDs processes will be handed out by SSDS database. Processes, like Instruments, will also generate metadata packets. These metadata packets will contain the process software version ID. SIAM will provide a framework for developing these process applications.

Requirement 5: All received data records and data stores must be accompanied by
1. *data record type* (for multiple *data sets* from one *data source*) as indicated by originator (data source or its proxy, the driver)

Scientist (instrument) needs to be able to specify the relationships of its data records, e.g., define that a new set has started (typically SSDS will define buckets (where we put data) based on metadata, this is an additional rule)

See comments on requirement #2. Not clear how requirement #5 applies to "buckets".

Requirement 6: For every type of received data set, provide content description metadata containing
1. data contact person
2. comments about the data set
3. basic format of data item(s) (the *use metadata*: type, size, units, unit reference, and descriptive name for each described item)
4. characterization of the content description itself
a. record content description metadata ID
b. record content description metadata version
c. a timestamp indicating creation time of this version of the description
5. if applicable, identification of any predecessor data sets from which this data set was created

The characterization information is needed so that updated descriptions can be recognized and utilized effectively (knowing which is most recent, and to which record types it applies). Note that a single record of a given origin and type defines a data set of that type, so this requirement is also associated with every type of received data record. Who will define the form of this description (SSDS?), and how will it get into the MOOS system so that it eventually comes back to SSDS (puck via instrument developer?)?

The metadata packet will include all of the above information. Note that metadata packets in a given data stream will be identifiable by packet sequence number.

Requirement 7: Users/operators must be able to correct errors in data's metadata/accompanying information that are associated with data records or data sets

If a bug is incorporated into the "on-board" (instrument-associated) description, the architecture must at least support its suppression (correction would be better)

SIAM will provide capability to update puck contents, but will not address corrections to data packets that have already been logged. Note that SIAM supports creation of "annotation packets", which are human-readable packets that can be appended to the data stream.

Requirement 8: Provide necessary information to automatically determine where data was collected

For some data, this changes with every record; for other data, the entire data set is at the same location.

Instrument's metadata packet specifies the parent node ID to which instrument is attached; the node location might be part of node's metadata, or node may also contain a positional sensor (e.g. GPS). In general, an instrument data packet does not contain positional data (e.g. depth) unless the instrument includes position sensor (e.g. pressure sensor).

Requirment 9: Provide notifications of observatory physical configuration changes at and above instrument level (installation/removal of instruments, nodes, and other data sources)

Need to know configuration of systems (location of instruments relative to mooring(s)), both current and historical, to answer the question: "Give me all the data of type X from (location Z at) mooring Y"

Node (e.g. mooring) has an ISI ID to which it logs packets. These packets include messages which are generated by the plug-and-work mechanism, and which specify the IDs of all current ISI instruments. Thus these "configuration packets" are generated when an ISI device is plugged in or removed. This log also contains other types of events (e.g.

mooring reset events). Finally, human operators can append annotations to the node's data stream.

Requirement 10: Provide notifications of physical changes to configuration of instruments (installation/removal/reconfiguration of sensors)

Can instrument sensors be changed in situ? This is a common occurrence presently (see UC-CTD introduction), and is documented in numerous sources over the past several years.

List of sensors attached to an instrument could either be stored in instrument's metadata (i.e. in the puck), or indicated by a human-generated annotation packet. Note that human operator is responsible for generating annotation packet.

Requirement 11: Provide notifications of software configuration changes to instruments

When ADCP is commanded its configuration changes. When instrument loses its mind, previous software configuration changes are lost. In either case, changed configuration must be detectable in the context of the data stream.

Specific instrument service implementations are responsible for inserting appropriate metadata/state packets into data stream when instrument configuration changes.

Requirement 12: Raw data formats (i.e., formats produced by instrument) must be recoverable from information in delivered data records

SSDS must be able to provide all data in its raw format

Agreed.

Requirement 13: Provide notification of asynchronous events.

MOOS will include science event detection, SSDS must be able to save and present all raw science data (which a newly detected event is)

Specific instrument service (or process) implementation is responsible for logging message packets when appropriate.

Requirement 14: Be capable of providing a data set for an instrument, at worst case through a non-ISI communication channel, such as through physical recovery.

Saving all raw data is only meaningful if it can then be retrieved/accessed. (How is this done mechanically? What is on-shore interface to SSDS?)

Data can be retrieved from node data logs, as all node data streams on are logged persistently on node, within storage limitations.

Requirement 15: Provide information within the data stream that can be used to determine when data is missing

Must be able to save all raw data (interpreted here as "all transmitted raw data," since some systems won't transmit all their data to shore) – the SSDS must know when we haven't seen all the data. (This behavior is also implied by the fundamental nature of a "connection.")

Each packet includes a packet sequence number; the sequence number is incremented by 1 whenever a packet is generated. Sequence number variable will have TBD bits.

Requirement 16: Provide deterministic access to data that was identified as missing.

The SSDS must request any missing data in order to be able to save all the raw data. Note that this gets tricky if the data subscription and retrieval models are not deterministic (i.e., if the models do not allow unambiguous specification of the data known to be missing, a particular request might not return all needed data without that being detected).

SIAM will include methods to retrieve logged packets by sequence number as well as by time.

Requirement 17: Either guarantee that data is in order, OR provide information to detect out of sequence data and resequence it.

(Note this does not include issues of missing data, only out of order data.) SSDS must be able to provide data in its raw format, which means original byte stream order, so it must have a way to ensure its data is in the correct order.

Data packets can be ordered by SSDS on the basis of provided sequence number.

Requirement 18: SSDS must be able to subscribe to a stream of information from MOOS data sources (both by data source, and by all sources) in such a way that it can be sure no records are missed from the time the data source began transmitting data.

Polling won't scale to the multiple platforms and instruments that MOOS will encompass, so a subscription ("notify me when new data arrives") mechanism is required. (Note data will be arriving from more than just moorings.) Also, the mechanism must be capable of picking up the very first data record and supplying all subsequent records, or, SSDS would often miss critical configuration information (because it gets sent to portal before SSDS has subscribed to it).

Each packet includes sequence number. SSDS can request SIAM logged data packets on sequence number criteria or timestamp criteria.

Requirement 19: SSDS must be able to get list of current data sources, and request notification of new data sources

This is generally important to know if a data source of interest is on-line, but it is especially important to SSDS if no general means to subscribe to all data sources is possible.

SIAM must provide mechanism to "discover" what portals are present – perhaps Jini? Each portal will provide access to corresponding node log, which contains lists of installed instruments (see notes on requirement 9).

Requirement 20: MOOS must allow for integrated, single-pass data entry of system configuration and log info by operators

Requiring multiple passes, or excessive interaction in a single pass, will be unreliable, will discourage users from adopting MOOS processes, and prevent the MOOS and SSDS designs from scaling up

We do not understand this requirement.

Requirement 21: Provide a protocol for requesting historical data from devices which can apply equally well to historical data request from SSDS

— SSDS should be able to implement the portal's "give me the data" request format, so that requests can go to SSDS instead (saving unnecessary accesses across low-bandwidth

This seems to be a requirement for SSDS, not SIAM.

Requirement 22: Defined mechanism for cleanly handling the loss of multiple data sources (e.g., given node or platform failure, some software should report or generate corresponding events at the system level, so that all processes using data from those components handle the failure gracefully)

If it's just a communication failure, this should not produce 'disconnect' messages for the devices. But if a node has died and/or is physically removed, system must recognize that logical device connections are gone.

There is no plan for Operations software to distinguish between "temporary" versus "permanent" disconnection. See comments on requirement 19.

Requirement 23: Data timestamps must be accurate to within 1 ms (absolute accuracy).

This is a fundamental MOOS requirement, but the SSDS depends on it for optimal integration of disparate data sets. It also forms the basis of operational and other metadata the SSDS uses to answer science queries. (Data that arrives with a significantly wrong timestamp may be grouped with the wrong data sets.) Note that if timestamps are

sufficiently precise, they may serve to order data records. (Although, since 2 records could be submitted within the same millisecond for some fast data submitters, this may mean sub-millisecond timestamping)

Note that the only timestamp provided by SIAM is the timestamp on packet creation, which is not necessarily the time at which the data was actually acquired. Note also that due to the non-realtime nature of the JVM being employed on the node, there is no guarantee on the accuracy of the SIAM-supplied packet timestamp. Applications that require aforementioned MOOS-specified clock accuracy must make special arrangements to acquire that timestamp and embed it within the data packet.