MOOS *Deployed* Subsystem Software Requirements Specification

Tom O'Reilly

Revision 3.0
November 4, 2002

1. Introduction

1.1. Purpose

This document describes the software functional requirements for the Deployed subsystem of MOOS. This document is intended primarily for MOOS system operators, system engineers, software designers, and project managers.

1.2. Scope

The MOOS Deployed subsystem is comprised of software that executes at at sea on unmanned sensor platforms, including moorings and cabled observatories. This software will enable control and diagnosis of sensors and other devices, as well as data acquisition, storage, and retrieval of science sensor data and metadata. The Deployed subsystem also enables operation of the deployed sensor platforms, including functions such as power and communications management.

1.3. Definitions, acronyms, and abbreviations

client – software which makes requests of a *service*
data sample – a logically associated sequence of data bytes generated by an instrument
packet stream – sequential time-tagged packets of an instrument's data and metadata
ISI – Instrument Software Infrastructure
MOOS – MBARI Ocean Observation System
instrument driver – software which sends commands to an instrument and retrieves data from the instrument through a communications port (e.g. serial port)

instrument node – a deployed computer to which instruments are directly connected
remote access – access to an instrument node from an external software component
platform – a physical structure (mooring, observatory instrument node, vehicle) which contains one or more *instrument nodes*
service – software which takes some action (e.g. controls a device, provides data) on request from a *client*. A service may have multiple clients.
SSDS – Shore-Side Data System
UI – user interface

1.4. References

M. Chaffey, *MOOS Requirements*
M. Chaffey, *MOOS System Design*
D. Davis, *ISI Requirements*
J. Graybeal, *SSDS Requirements*
J. Graybeal et al, *MOOS Use Cases*
T. O'Reilly, *MOOS Operations Software Requirements*
T. O'Reilly*, ISI Users Guide*
W. Radachonsky, *SideARM Functional Description*
IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998)

1.5. Overview

Section 2 of this document contains an overall description of the MOOS Deployed subsystem, describing general factors that affect the subsystem and providing background for requirements. Section 3 describes specific requirements in detail, organized by "system feature".
".

2. Overall description

2.1. Product perspective

Figure 1 is a simple block diagram of the MOOS system, showing data flow between the Deployed, Operations, and SSDS subsystems, as well as interfaces with users and systems which are external to MOOS.

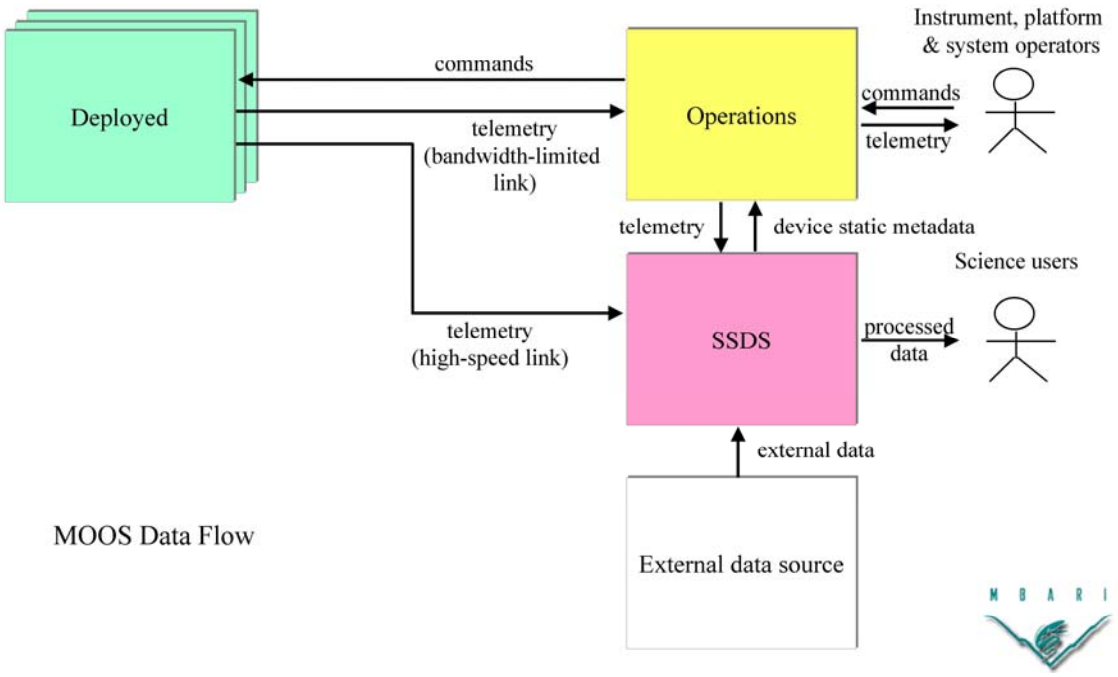

Figure 1. Block diagram of MOOS

The MOOS Deployed subsystem consists of at-sea instruments and instrument platforms (moorings, vehicles, observatory nodes) that acquire data at sea. The Operations subsystem is located on ship or shore, and provides interfaces and software by which users may interact with the Deployed subsystem, i.e. sending commands to deployed devices and retrieving data from deployed devices. The ship- or shore-based SSDS provides long-term archive for data and metadata generated by deployed elements, and provides data access and other services to science users.

2.1.1. System interfaces

As can be seen from Figure 1, Deployed has interfaces with the Operations and SSDS subsystems.

2.1.1.1. Interfaces with Operations subsystem

These interfaces have the following functionality:

2.1.1.1.1. Commands are sent from Operations to Deployed devices

2.1.1.1.2. Instrument and device data are transferred from Deployed components to Operations

2.1.1.2. Interfaces with SSDS subsystem

2.1.1.2.1. SSDS can retrieve data and metadata from Deployed components, in cases of high-speed communications links to the Deployed component (e.g. cabled observatory).

2.1.2. User interfaces

User interfaces (UIs) are located on Deployed nodes; in some cases these UIs are distributed elsewhere for execution while in other cases the UI may be executed “natively” on the node itself.

2.1.2.1. Native user interfaces

UIs for diagnostic tools shall be provided on instrument nodes. Users who are connected to the node via serial console or “telnet” login will utilize these UIs.

2.1.2.2. Distributed user interfaces

Deployed subsystem provides onboard storage of device user interface software; if allowed by link bandwidth, this software can be retrieved and executed on nodes within the Operations subsystem. These UIs enable control of devices, and retrieval of data from devices

2.1.3. Hardware interfaces

The instrument node is provided by the SideARM processor board. Communication with instruments, modems, and other devices is primarily through RS-232 and RS-485 interfaces. The SideARM provides these interfaces through the Dual Port Adapter (DPA); each SideARM board contains a number of serial ports which may be interfaced through the DPA.

2.1.4. Software interfaces

Deployed interfaces and implementation rely on ISI and ISI extensions. Thus the Deployed subsystem must be implemented in a software environment that is compatible with ISI. ISI provides classes for instrument and node “services”, data containers, and driver/service implementation frameworks. ISI also defines interfaces which conform to device puck communication and container protocols. As described in Section 3, Deployed components shall utilize device pucks where present.

2.1.5. Communications interfaces

The ocean observing system will utilize a variety of communication links to connect Deployed subsystems with Operations and SSDS. A wide range of communications bandwidth and latency characterizes these links. At one end of this spectrum are high-speed/low-latency links (e.g. T1 links); instrument nodes on a mooring benthic network or cabled observatory nodes will likely use links of this sort. At the other end of the spectrum are low-speed/high-latency links such as acoustic links and store-and-forward satellite; long-range wireless platforms are likely to utilize links of this type.

	Low latency < 500 msec	High latency > 500 sec
Low rate < 2400 bps		Acoustic: Underwater vehicle, Non-cabled benthic instrument
Medium rate 2400 – 19200 bps	RF, bent-pipe satellite: Mooring, vehicle on surface	Store-and-forward satellite: Mooring, vehicle on surface
High rate > 19200 bps	Copper, optical fiber: Mooring benthic network, cabled observatory	

Deployed communication elements must accommodate this variety of communication link characteristics.

2.1.6. Memory constraints

The SideARM provides a maximum of 64 Mbytes SRAM memory.

2.1.7. Operational considerations

2.2. Product functions

The Deployed subsystem performs the following functions:

2.2.1. Acquisition of sensor data/metadata

2.2.2. Logging of sensor data/metadata

2.2.3. Distribution of sensor data/metadata to Operations and SSDS

2.2.4. Instrument node management

2.3. User characteristics

System engineers and science instrument technicians are the primary direct users of the Deployed subsystem; these users are responsible for insuring that new instrument hardware and software conform to MOOS standards and interfaces, and that deployed instruments are functioning properly.

2.4. Constraints

Deployed interfaces and implementation rely heavily on ISI and ISI extensions. Thus the Deployed subsystem must be implemented in a software environment that is compatible with ISI.

Very low bandwidth and/or high latency characterize some communication links between Deployed and Operations. Protocols to be utilized between Operations and Deployed subsystems must take these link qualities into account.

2.5. Assumptions and dependencies

It is assumed that the Linux operating system will be available on deployed nodes. It is assumed that deployed nodes are capable of supporting the Java Connected Device Configuration (CDC).

3. Specific requirements

3.1. External interface requirements

Deployed nodes interact with other MOOS subsystems via interfaces implemented on the connecting communication links. ISI provides mechanisms to define and implement these interfaces, and shall be used when the communication link supports TCP/IP (e.g. for links with speed > 9600 bps and latency < 500 millisec). For very slow long-latency links (e.g. acoustic, store and forward satellite) which cannot support the ISI underlying TCP/IP protocol, other protocols and interfaces must be adopted or designed and implemented.

3.2. System features

3.2.1. Instrument packet streams

Each instrument has an associated *packet stream*, which consists of time-sequential *packets*. Each packet contains a unique device identifier, a time-tag, and other information. *Data packets* are packets that contain instrument data (raw and/or processed), while *metadata packets* contain “data about the data”. Metadata can be further categorized as *static* or *dynamic*; static metadata does not change while an instrument is running, while dynamic metadata may change while the instrument/driver is running.

3.2.1.1. Missing packet detection

The packet stream shall contain information that allows unambiguous detection of missing data packets.

3.2.1.2. Static metadata

A packet that contains static instrument metadata shall be added to the instrument’s stream at instrument initialization. The precise content of this

metadata is **TBD**, but will include information that describes how to process data packets from this instrument, and an identifier indicating which node this instrument is currently attached to.

3.2.1.3. Dynamic instrument state

The instrument packet stream shall contain time-tagged metadata, which reflects the instrument "state" at that point in time. Metadata shall be generated and added to the stream whenever relevant instrument state changes occur, or when requested by a client.

3.2.1.4. Data association with metadata

Each data packet shall contain information that allows that packet to be associated with an appropriate metadata packet in the same stream.

3.2.2. User control of instruments

There shall be capability to remotely control instruments and their drivers.

3.2.3. Instrument power management

Instrument drivers shall manage power in a manner consistent with the power budget allocated to the instrument.

3.2.4. Instrument data acquisition

3.2.4.1. Polled instruments

Data acquisition from instruments which must be explicitly "asked" for each data sample shall be supported

3.2.4.2. Streaming instruments

Data acquisition from instruments that asynchronously or continuously stream data to their data port shall be supported.

3.2.5. Instrument data acquisition scheduling

It shall be possible for users to specify and modify the schedule by which data will be acquired from each instrument.

3.2.5.1. Schedule specification

3.2.5.2. Polled data acquisition scheduling

3.2.5.3. Streaming data acquisition scheduling

3.2.5.4. Immediate data acquisition scheduling

It shall be possible to modify an instrument's schedule such that it begins to acquire a sample(s) immediately (as in response to an event).

3.2.6. Instrument data logging

Each deployed node must log all of its associated instrument packet streams to onboard persistent storage

3.2.6.1. Log data archive

It is required that all data packets from all instruments be archived in the data logs. There must be sufficient storage to contain all data/metadata for a specified period. This period is the interval between ship visits to the node or node recovery.

3.2.6.2. Log data retrieval criteria

It shall be possible to remotely retrieve instrument data/metadata packets, based on instrument ID and time-tag window.

3.2.7. Device pucks

Instrument descriptive metadata and instrument driver software shall be stored physically close to the instrument on a *device puck* or in the instrument itself if the instrument recognizes the puck protocol.

.

3.2.7.1. Self-contained instrument software

The instrument's driver software shall be stored on the puck

3.2.7.2. Self-contained instrument UI

A user interface for the instrument may optionally be stored in the puck

3.2.8. Data and metadata retrieval

It shall be possible to remotely retrieve instrument data/metadata packets from an instrument node, criteria for retrieval being the instrument's unique device ID and a time-tag window.

3.2.9. Data snapshot capability

Deployed subsystem must provide "data snapshot" capability; a data snapshot is a compact summary of an instrument's data.

3.2.10. Asynchronous event handling

3.2.11. Autonomous event response applications

Deployed subsystem must support "autonomous event response applications"; these applications execute on the deployed node, and autonomously initiate response to events or environmental conditions as detected by platform instrumentation.

3.2.12. Instrument node state tracking

3.2.12.1. Node location

It shall be possible to retrieve instrument node geographic location at a given point in time.

3.2.12.2. Instrument node event log

Relevant time-tagged instrument node events shall be logged to retrievable persistent storage. These events include but are not limited to node resets and node state (initializing, running, error, etc.)

3.2.13. Instrument node power management

3.2.13.1. Autonomous node power management

Autonomous node power management shall ensure that the node conserves power in a manner consistent with that node's power budget.

3.2.13.2. Remote power management

It shall be possible to remotely "wake up" a node from low-power state, and to remotely put a node into low-power state.

3.2.14. Multiple instrument nodes

A mooring shall be capable of accomodating multiple instrument nodes (12 science ports on each node). Each node shall support any remote operation described in this document, even though only one of the nodes actually has a telecommunications device.

3.2.15. Diagnostic tools

Tools shall be provided to remotely diagnose instrument problems. Tools will include software which allows user to interact directly with an instrument's data port for diagnostic purposes (e.g. "minicom"). It shall be possible to easily start and stop instrument drivers for troubleshooting purposes.

3.2.16. System resets

Users shall have capability to remotely "reset" instruments on a node and to reset/reboot the node itself.

3.2.17. Dynamic node reconfiguration/plug-and-work

Instrument node shall detect when the complement of instruments changes, and respond accordingly.

3.2.17.1. Response to instrument plug-in event

The node shall detect when a device is plugged into a node data port, at which point the node will attempt to retrieve driver code and other information from the assumed puck on the other side of the port. If the retrieval is successful, the node

shall execute the driver code. The plug-in event and its consequences shall be recorded in the node's log.

3.2.18. .Communication link management

3.2.19. Communications fault-tolerance

In the event of a communication link failure, the instrument node shall continue to acquire and log instrument data in accordance with the current schedule.

3.2.20. Time-synchronization

Where nodes are connected by high-speed/low-latency network links (e.g. a cabled network), it shall be possible to synchronize all node clocks with a "master clock" on one of the nodes, to an accuracy of 1 millisecond.

3.2.21. Device synchronization

It shall be possible to synchronize the operation of multiple devices with one another, e.g. instrument data acquisition shall be synchronized with operation of shutters and pumps. Timing requirement is TBD.

3.3. Performance requirements

3.3.1. Time-keeping

Each node shall have capability to maintain time accurate to 1 millisecond relative to an established standard time.

3.3.2. Instrument data rates

Deployed node infrastructure shall accommodate instruments which acquire data at a rate of **TBD**.

3.3.3. Execution timeliness

Particular drivers and other instrument node applications may require "realtime determinacy", meaning that certain operations execute within specific time boundaries in order for the application to be "correct". Deployed node infrastructure must designed so that timeliness requirements of **TBD** may be met.

3.3.4. Time-stamping accuracy and precision

The time-stamp on each packet reflects the time of "packet creation". Note that this is not necessarily the time at which the enclosed data was actually "acquired" by the instrument's sensor. **Requirement TBD**.

3.3.5. Event notification

3.4. Design constraints

3.5. Software system attributes

3.5.1. Portability

3.5.2. Maintainability

Deployed subsystem must enable remote installation and update of deployed software components

3.6. Other requirements

3.6.1. . Scalability

The Deployed subsystem of an ocean observing system may contain just a few instruments on a single instrument node, or many hundreds of instruments deployed on a variety of platforms; the Deployed subsystem must be *scalable* to these different scenarios.

3.7. Security