Open Geospatial Consortium, Inc.

Date: 2010-06-30

Reference number of this document: OGC 10-073r1

Category: Public Engineering Report

Editor: Scott Fairgrieve

OWS-7 CCSI-SWE Best Practices Engineering Report

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Document type: OpenGIS® Engineering Report
Document subtype: NA
Document stage: Approved for public release
Document language: English

Preface

Suggested additions, changes, and comments on this draft report are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in an edited copy of this document.

The changes made in this document version, relative to the previous version, are tracked by Microsoft Word, and can be viewed if desired. If you choose to submit suggested changes by editing this document, please first accept all the current changes, and then make your suggested changes with change tracking on.

Contents

Page

Figures

Page

Tables

OWS-7 CCSI-SWE Best Practices Engineering Report

1 Introduction

1.1 Scope

This document seeks to define the Best Practices for integrating Common Chemical, Biological, Radiological, and Nuclear (CBRN) Sensor Interface (CCSI) compliant and potentially other CBRN-based sensors into an OGC Sensor Web Enablement (SWE)-based environment. The document focuses on the practical application of SWE services and encodings for describing and interacting with CCSI sensors and data and draws heavily from and expands upon work performed in the OGC Web Services Phase 6 (OWS-6) testbed to define methodologies for integrating CCSI sensors into a SWE-based environment both now, by building upon the OWS-6 work, and in the future, by defining CCSI profiles of the SWE specifications.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
Scott Fairgrieve	Northrop Grumman Information Systems
Scott Wright	Northrop Grumman Information Systems
Eric Dorner	Northrop Grumman Information Systems
Alex Vagus	Joint Program Executive Office for Chemical and Biological Defense
Arthur Laudenslager	Joint Program Executive Office for Chemical and Biological Defense
Mike Botts	Botts Innovative Research, Inc.

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
05/20/10	0.1.0	Scott Fairgrieve	All	Draft Version
05/21/10	0.2.0	Scott Fairgrieve	Several	Updated Draft
05/29/10	0.3.0	Scott Fairgrieve	Several	Final Draft
06/14/10	0.4.0	Scott Fairgrieve	Several	Updated Final Draft

1.4 Future work

Improvements in this document are desirable to ensure that the concepts and best practices described in this document are fully applicable in real CBRN sensor operations. Concepts included in this document were developed and implemented using CCSI documentation and a simple CCSI sensor emulator that only supports a subset of sensors and sensor types that may be available to the broader CBRN sensor community; therefore, it would be worthwhile to implement and test the concepts described in this report with real CBRN sensors in a real world environment to ensure that the concepts are complete and accurate.

1.5 Forward

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

OGC 06-009r6, *OpenGIS® Sensor Observation Service v1.0*

OGC 06-021r4, *OpenGIS® Sensor Web Enablement Architecture*

OGC 06-028r5, *OpenGIS® Sensor Alert Service*

OGC 06-121r3, *OpenGIS® Web Services Common Specification*

NOTE This OWS Common Specification contains a list of normative references that are also applicable to this Implementation Specification.

OGC 07-000, *OpenGIS® Sensor Model Language (SensorML)*

OGC 07-006r1, *OpenGIS® Catalogue Service Implementation Specification*

OGC 07-014r3, *OpenGIS® Sensor Planning Service v1.0*

OGC 07-022r1, OpenGIS® Observations and Measurements – Part 1 – Observation Schema

OGC 07-022r3, OpenGIS® Observations and Measurements – Part 2 – Sampling Features (1.0)

OGC 07-092r1, *Definition identifier URNs in OGC Namespace*

OGC 07-110r2, *CSW-ebRIM Registry Service – Part 1: ebRIM profile of CSW (1.0.0)*

OGC 07-122r2, *OpenGIS® SensorML Encoding Standard v 1.0 Schema Corregendum 1 (1.01)*

OGC 07-165, *Sensor Web Enablement: Overview and High Level Architecture*

OGC 08-132, Event Pattern Markup Language Discussion Paper

OGC 08-133, Sensor Event Service Discussion Paper

OGC 08-176r1, OWS-6 Secure Sensor Web Engineering Report

OGC 09-000, OpenGIS® Sensor Planning Service v2.0 RFC Package

OGC 09-007, OWS-6 CCSI-SWE Engineering Report

OGC 09-032, OWS-6 Event Architecture Engineering Report

http://www.ws-i.org/Profiles/BasicProfile-1.1.html, *WS-I Basic Profile 1.1*

http://www.w3.org/TR/soap/, *SOAP*

http://www.w3.org/TR/wsdl, *WSDL*

http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf, WS-BaseNotification 1.3

http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-os.pdf, WS-BrokeredNotification 1.3

http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf, WS-Topics 1.3

JPEO-CBD CCSI v1.0 Volume I, *Common Chemical, Biological, Radiological, Nuclear (CBRN) Sensor Interface (CCSI): Volume I – Summary and Architecture*

3 Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Specification [OGC 06-121r3] and in the documents referenced in Clause 2 shall apply. In addition, the following terms and definitions apply.

3.1
channel
a logical grouping of information transmitted to or from a CCSI sensor

3.2
command
a message sent to a CCSI sensor that causes the sensor to perform a specific action

3.3
controlling application
host application

an application that connects to and interacts with a CCSI sensor

3.4
domain feature
feature of a type defined within a particular application domain.

3.5
feature
a component of a system. In geographic terms, this is often a component of the Earth, whether natural (a river) or man-made (a reservoir), concrete (the Amazon river) or abstract (a political boundary).

3.6
in situ
stationary, non-mobile

4 Conventions

4.1 Abbreviated terms

ACK	Positive Acknowledgement
API	Application Programmer's Interface
C2	Command and Control
CBRN	Chemical, Biological, Radiological, Nuclear

CCSI	Common CBRN Sensor Interface
CONOPs	Concept of Operations
CS/W	Catalog Service for the Web
EMCON	Emission Control
GPS	Global Positioning System
GUID	Globally Unique Identifier
MUC	Multi-User Chat
NAK	Negative Acknowledgement
NATO	North Atlantic Treaty Organization
NTP	Network Time Protocol
O & M	Observations and Measurements
OSI	Open Systems Interconnection
PDP	Policy Decision Point
PEP	Policy Enforcement Point
REST	Representational State Transfer
SAS	Sensor Alert Service
SensorML	Sensor Model Language
SIS	Sensor Interface Service
SOAP	Simple Object Access Protocol
SOS	Sensor Observation Service
SPS	Sensor Planning Service
STS	Security Token Service
SWE	Sensor Web Enablement
TML	Transducer Markup Language
UGS	Unattended Ground Sensor
URI	Uniform Resource Identifier
URN	Uniform Resource Name
UTC	Coordinated Universal Time
UUID	Universally Unique Identifier
WNS	Web Notification Service
WSDL	Web Service Description Language
WS-Security	Web Services Security
WS-Trust	Web Services Trust Language

XSLT eXtensible Stylesheet Language Transformation

4.2 UML notation

Some diagrams that appear in this report are presented using the Unified Modeling Language (UML) sequence diagram, as described in Subclause 5.2 of [OGC 06-121r3].

4.3 Used parts of other documents

This document uses significant parts of document [OGC 09-007]. To reduce the need to refer to that document, this document copies some of those parts with small modifications. To indicate those parts to readers of this document, the largely copied parts are shown with a light grey background (15%).

5 Overview

This report provides best practices for integrating CCSI sensors and their data into an OGC SWE-based architecture. The report begins by reviewing the CCSI standards and the key software interface concepts within those standards, particularly the concepts listed in Volume I and Volume III of the CCSI specifications. Note that Volume II is outside the scope of this report, since it defines the hardware interface for CCSI sensors. The report then provides an overview of the OGC SWE standards and the reasons for incorporating CCSI into a SWE-based architecture and harmonizing CCSI and SWE in the future. The report then reviews the previous work performed in OWS-6 with regards to integrating CCSI sensors into an OGC SWE-based architecture and uses and expands upon that prior work to define best practices. Before reviewing this document, it would be worthwhile to review and become familiar with the OWS-6 CCSI-SWE ER [OGC 09-007] (http://portal.opengeospatial.org/files/?artifact_id=33355), as the sections below draw heavily from the concepts described in the OWS-6 ER.

6 CCSI Review

6.1 Background/Current Status

CCSI is a Defense Information Standards Registry (DISR) mandated standard for CBRN sensors authored and maintained by the Joint Program Executive Office for Chemical and Biological Defense (JPEO-CBD). JPEO-CBD adopted version 1.0 of the CCSI standards on February 15, 2008. CCSI became a DISR mandated standard on July 28, 2009. JPEO-CBD is currently in the process of updating the published standard to include a number of documentation corrections and clarifications and to align the specification's data schema with the current CBRN Data Model. JPEO-CBD originally developed the CCSI standards with the goal of solving the hardware and software interoperability problems that would have emerged if today's stand-alone military CBRN sensors were network enabled. JPEO-CBD realized that CBRN sensor interoperability would support the ultimate end user of CBRN sensors, the warfighter. By maximizing the interoperability of CBRN sensors employed in the field, the government customer funding CBRN sensor programs would minimize the cost and effort associated with

developing and integrating CBRN sensor related capabilities, while contractors and systems integrators supporting those CBRN programs would reduce or eliminate the need to develop and sustain custom interface hardware and software. See http://www.jpeocbd.osd.mil/packs/Default.aspx?pg=860 for more information on CCSI standards.

6.2 Key Concepts

Volumes I and III of the CCSI specifications define several key concepts; these include sensor definition files for describing CCSI compliant sensors, channel messages used to communicate various readings and other information provided by CCSI sensors, and command messages used to send tasking commands to CCSI sensors. The following sections provide a brief overview of these concepts. For a more detailed overview, refer to the OWS-6 CCSI-SWE ER and the CCSI specifications.

6.2.1 Sensor Definitions

6.2.1.1 Concept of Operations for Creating/Using Sensor Definition Files

Sensor Definition files describe CBRN sensors within CCSI-based sensor operations. Sensor Definition files include important pieces of information necessary for controlling applications to understand and interact with CCSI sensors.

Table 1 - CCSI Sensor Definition Elements

Element	Description
Sensor	Provides general metadata about a sensor, including manufacturer contact information, program contact information, and identifying information (e.g. name, type, model, etc.)
DataDefinitions	Defines any unique data types that the sensor supports. A CCSI sensor may include unique elements in its data beyond those mandated in the CCSI specifications.
SensorCommands	Defines any unique commands that the sensor supports. A CCSI sensor may support unique commands in addition to the standard commands specified by CCSI.
SensorConfig	Includes configuration information for the sensor (e.g. alert thresholds)
SensorVersion	Describes version specific information about the sensor including the specific CCSI version to which the sensor adheres, the hardware and software versions of the sensor, sensing capabilities, supported CCSI channels and commands, along with other information.

Section 4.6.4 in Volume I of the CCSI specifications defines the process for creating and using CCSI sensor definition files. DISR requires government program managers to

direct sensor vendors to implement CCSI and to create Sensor Definition files for the CBRN sensors they develop as contract deliverables under their programs. The CCSI Sensor Definition files developed by sensor vendors are then passed through a compliance testing process before being stored in one or more Sensor Definition Repositories for use by Systems Integrators or within CBRN sensor operations under that program.

6.2.1.2 Example Sensor Definition File

```
<?xml version="1.0" encoding="utf-8"?>
<SensorDefinition xmlns="" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cbrn-ccsi.org/schema/CCSI
file:/C:/CCSI_V1.0/XML/CCSI_XML_Sensor.01.01.xsd">
    <Sensor>
        <Manufacturer>
            <PointOfContactNameText>Manufacturer</PointOfContactNameText>
            <PointOfContactJobTitleNameText>Chief
Engineer</PointOfContactJobTitleNameText>
            <PointOfContactAddressLineText>Canberra Dover</PointOfContactAddressLineText>
            <PointOfContactCityNameText>Dover</PointOfContactCityNameText>
            <PointOfContactStateNameText>NJ</PointOfContactStateNameText>
        </Manufacturer>
        <Program>
            <PointOfContactNameText>Program Manager</PointOfContactNameText>
            <PointOfContactJobTitleNameText>PD Radiological
Detectors</PointOfContactJobTitleNameText>
            <PointOfContactOfficeNameText>JPM NBC CA</PointOfContactOfficeNameText>

    <PointOfContactEmailAddressText>programmanager@us.army.mil</PointOfContactEmailAddres
sText>
            <PointOfContactAgencyAcronymText>JPEO-CBD</PointOfContactAgencyAcronymText>
        </Program>
        <SensorIdentification>
            <SensorName>AN/UDR-14</SensorName>
            <SensorType class="RAD" variant="PNT" name="SC001"/>
            <SensorModel>TBD</SensorModel>
            <SensorDescription>
          Radiation detector adapted to support CCSI within FCS platforms.
      </SensorDescription>
        </SensorIdentification>
        <SensorDescription>
      AN/UDR-14 RADIAC radiological detector.
    </SensorDescription>
    </Sensor>
    <DataDefinitions>
        <DataElement name="FilteredDoseRate" type="float" editable="false">
            <UnitOfMeasure>cGy/hr</UnitOfMeasure>
            <ValidityCheck>
                <Range>0.0 - 999.9</Range>
            </ValidityCheck>
            <HelpText>
        Current filtered dose rate.  The units reported may be micro Gy/hr, centi Gy/hr,
or Gy/hr.
      </HelpText>
        </DataElement>
        <DataElement name="MissionDose" type="float" editable="false">
            <UnitOfMeasure>cGy</UnitOfMeasure>
            <ValidityCheck>
                <Range>0.0 - 999.9</Range>
            </ValidityCheck>
            <HelpText>
        Total accumulated dose since last cleared.
      </HelpText>
        </DataElement>
    </DataDefinitions>
    <SensorCommands>
        <SensorUniqueCmd name="Clear_Mission_Dose" ack_required="true" roles="ANY">
            <HelpText>
```



```
    This command resets (zeroes) the mission accumulated dose value on the sensor.
  </HelpText>
        <TestDefinition>
            <TestCase name="NormalTest" type="NORMAL">
                <Description>
        This command tests the normal behavior of the command.
      </Description>
                <ExpectedResults>
                    <ACK/>
                </ExpectedResults>
            </TestCase>
        </TestDefinition>
    </SensorUniqueCmd>
</SensorCommands>
<SensorConfig>
    <ConfigItem>
        <ItemName>DoseAlarmSetPoint</ItemName>
        <ItemType>float</ItemType>
        <ItemValue>120.0</ItemValue>
        <!-- Units are ALWAYS centi-gray for this item. -->
        <ItemAccessRole>ANY</ItemAccessRole>
        <ItemModifyRole>ANY</ItemModifyRole>
    </ConfigItem>
    <ConfigItem>
        <ItemName>DoseRateAlarmSetPoint</ItemName>
        <ItemType>float</ItemType>
        <ItemValue>100.0</ItemValue>
        <!-- Units are ALWAYS centi-gray/hr for this item. -->
        <ItemAccessRole>ANY</ItemAccessRole>
        <ItemModifyRole>ANY</ItemModifyRole>
    </ConfigItem>
</SensorConfig>
<SensorVersion>
    <CcsiVersion major="1" moderate="0" minor="0" patch="0"/>
    <ReleaseVersion>
        <SensingUnit>SC001</SensingUnit>
        <HwVersion major="1" moderate="0"/>
        <SwVersion major="1" moderate="0"/>
        <Description>
    This is the base version of the AN/UDR-14 RADIAC adapted for FCS and CCSI.
  </Description>
    </ReleaseVersion>
    <Capability SensingUnit="SC001">
        <Capability>
            <ItemType>Gamma Radiation Dose</ItemType>
            <LevelUnits>cGy</LevelUnits>
        </Capability>
        <Capability>
            <ItemType>Gamma Radiation Dose Rate</ItemType>
            <LevelUnits>cGy/hr</LevelUnits>
        </Capability>
    </Capability>
    <Channels>
        <Channel Channel="ALERTS" HasMetaData="true">
            <Metadata>FilteredDoseRate</Metadata>
            <Metadata>MissionDose</Metadata>
        </Channel>
        <Channel Channel="CONFIG"/>
        <Channel Channel="HRTBT"/>
        <Channel Channel="IDENT"/>
        <Channel Channel="MAINT"/>
        <Channel Channel="READGS" HasMetaData="true">
            <Metadata>FilteredDoseRate</Metadata>
            <Metadata>MissionDose</Metadata>
        </Channel>
        <Channel Channel="STATUS"/>
    </Channels>
    <DataUsed>
        <Uses>FilteredDoseRate</Uses>
        <Uses>MissionDose</Uses>
    </DataUsed>
    <SensorCommandUsed>
```

```
            <Uses>Clear Mission_Dose</Uses>
        </SensorCommandUsed>
        <CcsiCmdSupport>
            <Supports>Register</Supports>
            <Supports>Deregister</Supports>
            <Supports>Render_Useless</Supports>
            <Supports>Reset_Msg_Seq</Supports>
            <Supports>Start_Stop</Supports>
            <Supports>Start_Self_Test</Supports>
            <Supports>Set_Heartbeat</Supports>
            <Supports>Set Ccsi Mode</Supports>
            <Supports>Get_Status</Supports>
            <Supports>Get_Configuration</Supports>
            <Supports>Set_Config</Supports>
            <Supports>Get_Identification</Supports>
        </CcsiCmdSupport>
        <SensorConfig>
            <Uses>DoseAlarmSetPoint</Uses>
            <Uses>DoseRateAlarmSetPoint</Uses>
        </SensorConfig>
    </SensorVersion>
</SensorDefinition>
```

6.2.2 Channels/Channel Messages

Messages sent to and from CCSI sensors are organized into channels. Table 2 below outlines the various channels defined in the CCSI specifications.

Table 2 - CCSI Channels

Channel	Channel Type Enumeration	Channel Type Abbreviation	Description
Command	CMD	c	Channel for issuing commands to a CCSI sensor
Identification	IDENT	i	Channel containing identification information from a CCSI sensor, such as the serial number and hardware and software versions
Maintenance	MAINT	m	Channel containing maintenance information from a CCSI sensor, which includes details about the supply level of certain items such as battery power.
Readings	READGS	r	Channel containing readings/observations from a sensor, which can include spectral and weather data as well as sensor unique data.
Alerts	ALERTS	a	Channel containing alerts from a sensor
Heartbeat	HRTBT	h	Channel containing periodic heartbeats from a sensor indicating that the sensor is

			still active and communicating
Configuration	CONFIG	f	Channel containing configuration information for a sensor.
Status	STATUS	s	Channel containing status information for a sensor.
Location	LOC	l	Channel containing location information for a sensor.
Time/Date	TMDT	t	Channel containing date and time information for a sensor.
Security	SECRTY	z	Channel containing security information for a sensor.

The following diagram illustrates a CCSI Alerts channel message and highlights some of the key pieces of information contained in any CCSI channel message.

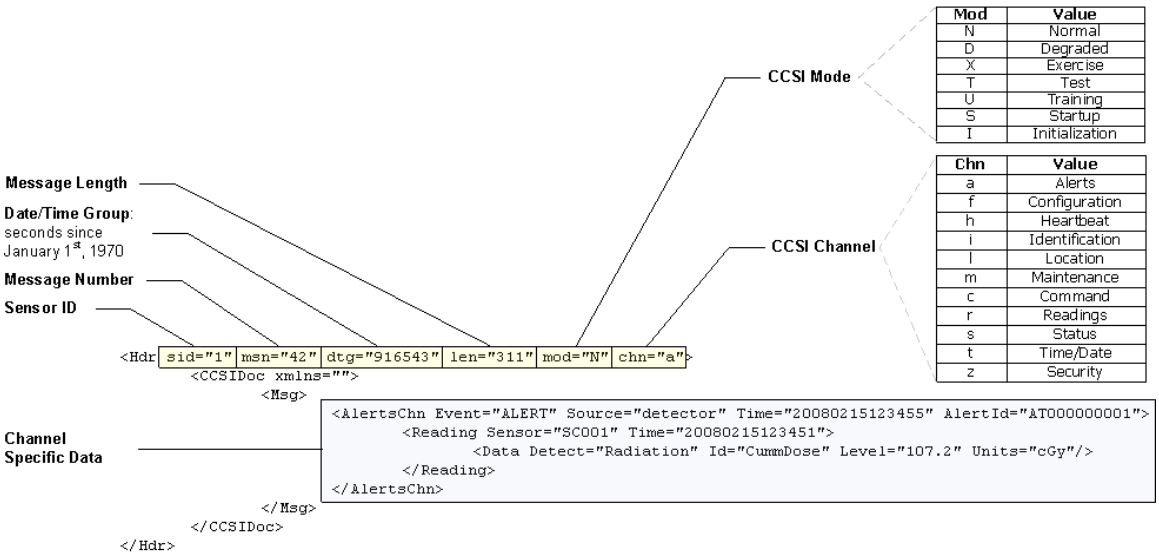

Figure 1 - The Anatomy of a CCSI Channel Message

A standard CCSI channel message contains an *Hdr* element with attributes that provide important information about the message, including the identifier of the sensor that produced the message (*sid*), the time the message was created (*dtg*), the current CCSI mode (*mod*), and the CCSI channel of the message (*chn*). The value contained in the *dtg* attribute represents seconds since 1970, which is commonly referred to as Unix time (http://en.wikipedia.org/wiki/Unix_time). The value contained in the *chn* attribute is one of the Channel Type Abbreviation values listed in Table 2 above. The *Hdr* element contains a child *CCSIDoc* element, which in turn contains one or more child *Msg* elements. The child element of the *Msg* element is a channel specific XML element, as illustrated by the *AlertsChn* element in the example above. A CCSI channel document

may contain one or more *Msg* elements with various readings from that channel, representing a time series of channel data.

6.2.3 Issuing Commands

CCSI commands are a special type of channel message that get issued from a controlling application to a sensor in order to change that sensor's configuration. CCSI command messages are issued on the Command channel, indicated through an *Hdr* element *chn* attribute value of "c". The CCSI specifications define several standard commands, and the CCSI Sensor Definition file for a particular sensor tells the controlling application what commands that sensor supports through the *CcsiCmdSupport* element.

```
<?xml version="1.0" encoding="UTF-8"?>
<SensorDefinition xmlns="http://www.cbrn-ccsi.org/schema/CCSI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cbrn-ccsi.org/schema/CCSI
file:/C:/CCSI_V1.0/XML/CCSI_XML_Sensor.01.01.xsd">

…Omitted for brevity

<SensorVersion xmlns="">

    …Omitted for brevity

        <CcsiCmdSupport>
            <Supports>Register</Supports>
            <Supports>Deregister</Supports>
            <Supports>Render Useless</Supports>
            <Supports>Reset Msg Seq</Supports>
            <Supports>Start_Stop</Supports>
            <Supports>Start_Self_Test</Supports>
            <Supports>Set_Heartbeat</Supports>
            <Supports>Set Ccsi Mode</Supports>
            <Supports>Get_Status</Supports>
            <Supports>Get Configuration</Supports>
            <Supports>Set_Config</Supports>
            <Supports>Get_Identification</Supports>
        </CcsiCmdSupport>
    </SensorVersion>
</SensorDefinition>
```

Table 3 below outlines the CCSI standard commands.

Table 3 - CCSI Standard Commands

Command	Description
Power_Off	Turns a sensor off
Render_Useless	Makes a sensor unusable by erasing its configuration, communication, user, and other information. This can be initiated immediately or in response to a tamper count.
Set_Tamper	Starts or stops the tamper mechanism of a sensor.
Set_Emcon_Mode	Sets the emission control (EMCON) mode for wireless communications with a sensor. See Table 32, page. 30 of CCSI

	Volume III for more information on EMCON modes.
Set_Encryption	Enables/disables encryption capabilities for the communication link(s) specified in the command.
Erase_Sec_Log	Erases the security-related contents of the sensor's event log.
Get_Sec_Log	Returns the security-related contents of the sensor's event log to the controlling application
Sanitize	Causes the sensor to erases all security-related information including cryptographic, user, host-specific, and communications information.
Zeroize	Erases all cryptographic information in the sensor
Set_Cmd_Privilege	Sets the user role required in order to execute each command
Reset_Msg_Sequence	Causes the controlling application and sensor to reset their message sequence numbers (i.e. provided in the msn attribute of each message) to 1.
Start_Stop	Causes the sensor to start or stop sensing
Reboot	Reboots the sensor's operating system
Set_Comm_Permission	Sets the number of controlling applications that can connect to the sensor
Set_Local_Alert_Mode	Enables/disables the local alert mode of the sensor (i.e. lighted or audible warnings that the sensor may produce locally)
Start_Self_Test	Enables the built in test functionality of the sensor. The sensor returns maintenance channel reports to the host application indicating the results of the tests.
Set_Security_Timeout	Sets timeouts and attempt counts for security events.
Clear_Alert	Clears the alert conditions specified by the command, which could include all alerts or specific alerts (identified by their Alert ID).
Silence	Silences/enables local alert mechanisms on the sensor (i.e. lighted or audible alert indicators).
Set_Heartbeat	Sets the period for heartbeat messages
Set_Date_Time	Sets the sensor's local date and time. The sensor normally gets

	date and time information using NTP or GPS.
Set_Location	Sets the sensor's current location based on latitude, longitude, altitude, and precision. This command is typically implemented on in-situ sensors that lack GPS capabilities.
Set_Data_Compression	Enables/disables data compression on the channels identified by the command.
Set_Local_Enable	Enables/disables local visual or audible indicators on the sensor
Set_Bw_Mode	Sets the bandwidth mode for communication links in use by the sensor
Get_Status	Causes the sensor to return status information to the controlling application
Get_Configuration	Causes the sensor to return configuration information to the controlling application
Install_Start	Initiates the software installation process on the sensor
Install	Provides code/data blocks to be installed on the sensor
Install_Complete	Completes the software installation process on the sensor
Set_Ccsi_Mode	Changes the mode of the sensor without requiring the sensor to reboot. Valid modes include NORMAL, EXERCISE, TEST, TRAINING, and SETUP.
Set_Config_Privilege	Sets the access roles for sensor configuration
Get_Users	Retrieves a list of all users of the sensor
User_Change	Allows a controlling application to manage users of the sensor
Register	Allows a controlling application to register or deregister for one or more information channels from a sensor and can be used as a query mechanism to retrieve the latest channel data.
Deregister	Closes the connection between a controlling application and a sensor, including deregistering any registered information channels.
Set_Config	Allows a controlling application to manage configuration options of the sensor
Get_Identification	Causes the sensor to send an identification channel message to

	the controlling application
Get_Alert_Details	Causes the sensor to return detailed data associated with a particular alert that may not have been transmitted initially (i.e. spectral readings or other large data)
Get_Reading_Details	Similar to Get_Alert_Details, except causes the sensor to return detailed data associated with a particular reading.

The XML snippet below illustrates a *Set_Ccsi_Mode* command message that sets the *Mode* to NORMAL.

```
<Hdr xmlns="" sid="1" msn="13" dtg="1227115090" chn="c" mod="T" len="145">
    <CCSIDoc>
        <Msg>
            <CmdChn Cmd="Set_Ccsi_Mode">
                <Arg>
                    <ArgName>Mode</ArgName>
                    <ArgValue>NORMAL</ArgValue>
                </Arg>
            </CmdChn>
        </Msg>
    </CCSIDoc>
</Hdr>
```

As described in the OWS-6 CCSI-SWE ER:

Each parameter of a CCSI command is included as an *Arg* element with *ArgName* and *ArgValue* key-value pair subelements. Depending on the definition of each command, *ArgName* keys are not required to be unique. For instance, the *Register* command allows a controlling application to register for channel information for one or more channels. An argument with *ArgName* Channel can be repeated several times within this command in order to list each of the channels for which a controlling application wants to register.

Command definitions are described in a separate XML document that defines all CCSI Standard Commands. An example CCSI Standard Commands document is included in Annex A for reference. The XML snippet below provides the definition of the CCSI *Register* command and illustrates the use of a *RptGrp* element with child *RepeatElement* elements for defining the ability of a command to allow repeated parameters.

```
<CCSI Std Command xmlns="" name="Register" roles="OPER" ack_required="true"
by_parameter="true" compliance="mandatory">
    <Arguments>
        <RptGrp max_repeat="7">
            <RepeatElement name="Channel" type="ChannelType" mandatory="true"/>
            <RepeatElement name="Type" type="RegistrationType" mandatory="true"/>
            <RepeatElement name="Rate" type="RegisterRateType" mandatory="false">
                <DefaultValue>0</DefaultValue>
            </RepeatElement>
            <RepeatElement name="Details" type="Boolean" mandatory="false"/>
        </RptGrp>
    </Arguments>
    …omitted for brevity
</CCSI_Std_Command>
```

CCSI support for repeatable parameters is notable in that it affects the way CCSI commands are described and encoded in SPS 1.0 formats (see section 11.4.2).

6.2.4 Interaction between a Controlling Application and a Sensor

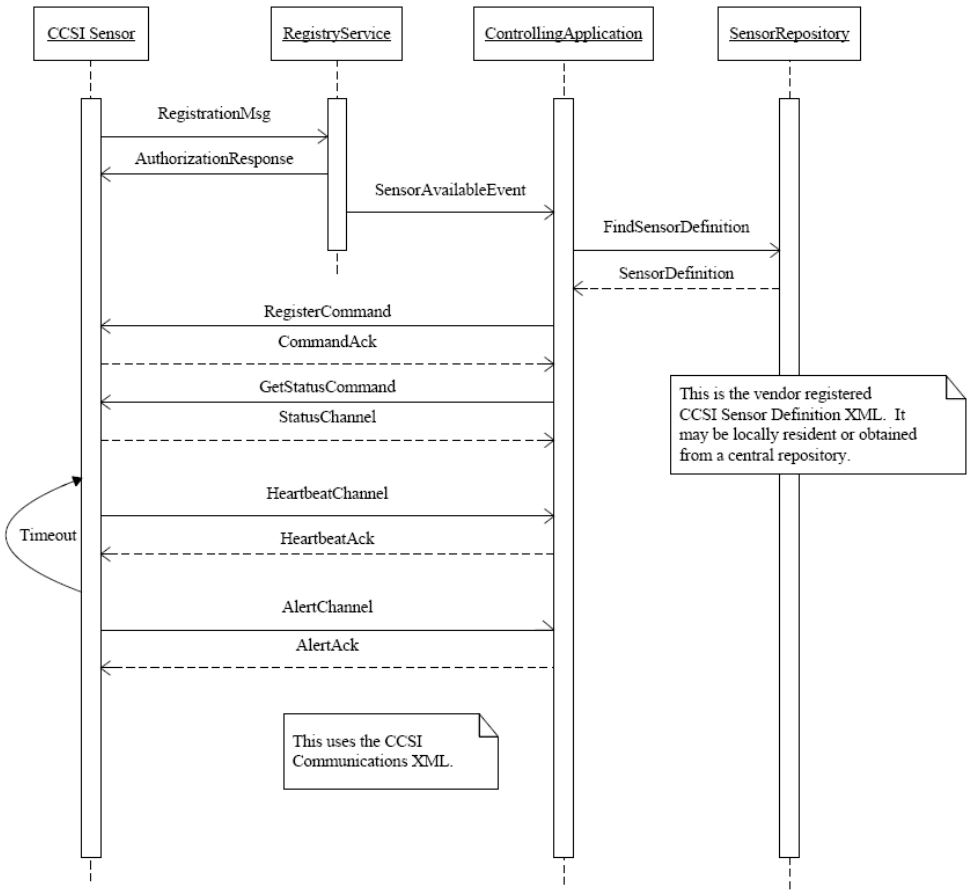

Figure 2 - Controlling Application - Sensor Communication

The OWS-6 CCSI-SWE ER describes the specifics of the diagram above in section 6.4.2. In order to minimize the need to refer to the OWS-6 CCSI-SWE ER, the section below repeats the description from that document verbatim:

After the controlling application becomes aware of a CCSI sensor, either through some automatic notification mechanism, a registry service, or a priori knowledge, the controlling application retrieves the sensor's sensor definition file from a sensor repository (which may be locally resident), connects to the sensor using the appropriate connection mechanism (e.g. opening a TCP connection), and sends a connection message like the XML example below.

```
<?xml version="1.0" encoding="UTF-8"?>
<CCSIConnection xmlns="http://www.cbrn-ccsi.org/schema/CCSI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cbrn-ccsi.org/schema/CCSI
file:/C:/CCSI_V1.0/XML/CCSI_XML_Open_Interface_Connection.00.01.xsd" sid="1">
```


The sensor then returns a connection message that contains its sensor identifier (shown below). The controlling application stores this sensor identifier (specified by the *sid* attribute) and uses it in subsequent communications with the sensor.

```
<?xml version="1.0" encoding="UTF-8"?>
<CCSIConnection xmlns="http://www.cbrn-ccsi.org/schema/CCSI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cbrn-ccsi.org/schema/CCSI
file:/C:/CCSI V1.0/XML/CCSI XML Open Interface Connection.00.01.xsd" sid="1">
```

The controlling application then sends a *Register* command containing a list of desired channels for which it would like to receive information along with desired frequencies and notification preferences (shown below). The sensor's sensor definition file contains information about particular channels that the sensor supports (within the *CcsiCmdSupport* element), and the controlling application utilizes this information to construct a *Register* command message. The CCSI specification lists specific details about the parameters of the *Register* command in Volume III section 6.2.1.

```
<Hdr xmlns="" sid="1" msn="1" dtg="1227114922" chn="c" mod="N" len="1419">
    <CCSIDoc>
        <Msg>
            <CmdChn Cmd="Register">
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>ALERTS</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>EVENT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>0</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>HRTBT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>PERIOD</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>30</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>CONFIG</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>EVENT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>0</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>IDENT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>ONCE</ArgValue>
                </Arg>
```



```
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>0</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>MAINT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>EVENT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>0</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>STATUS</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>EVENT</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>0</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Channel</ArgName>
                    <ArgValue>READGS</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Type</ArgName>
                    <ArgValue>PERIOD</ArgValue>
                </Arg>
                <Arg>
                    <ArgName>Rate</ArgName>
                    <ArgValue>15</ArgValue>
                </Arg>
            </CmdChn>
        </Msg>
    </CCSIDoc>
</Hdr>
```

Upon receiving a valid *Register* command, the sensor returns an ACK response (shown below) that lets the controlling application know that the *Register* command was successful, and the sensor then returns that latest channel information for each channel for which the controlling application registered. As described above, the controlling application must send an acknowledgement message to the sensor each time it receives a heartbeat or alert message; otherwise, the sensor may disconnect after several non-acknowledgements.

```
<Hdr sid="1" ackmsn="1" msn="1" ack="ACK" mod="N" dtg="1227114922" chn="c" len="0"
xmlns=""/>
```

At this point, the controlling application can wait to receive any periodic or asynchronous channel information it requested through the *Register* command or can send any commands it needs the sensor to execute. When the controlling application is finished using the sensor, it de-registers with the sensor (shown below), which cancels all previous channel registrations, and the sensor returns an acknowledgement indicating successful de-registration.

```
<Hdr xmlns="" sid="1" msn="5" dtg="1227115222" chn="c" mod="N" len="68">
    <CCSIDoc>
```



```
        <Msg>
            <CmdChn Cmd="Deregister"/>
        </Msg>
    </CCSIDoc>
</Hdr>
```

The controlling application then physically disconnects from the sensor using the appropriate mechanism (e.g. closing the associated TCP connection).

The current version of the CCSI specifications does not include web service specifics other than to say that all web services developed to utilize CCSI sensors and data should adhere to the WS-I Basic Profile 1.1, which implies the use of Simple Object Access Protocol (SOAP) 1.1 for messaging and Web Service Definition Language (WSDL) 1.1 for describing web service operations.

The CCSI must support the DOD network centricity requirements of the Global Information Grid (GIG). If the sensor has a network interface and is intended to operate on a DOD network that implements web services for sensors…The CCSI shall provide access to all CCSI capabilities through a web services interface compliant with WS-I Basic Profile 1.1

JPEO-CBD is in the process of defining one or more web service interfaces for use with CCSI, but these interfaces have not been developed enough to analyze in full detail at the time of this report.

7 SWE Overview

While the CCSI standards address sensor interoperability from a CBRN sensor community perspective, the OGC SWE standards address sensor interoperability from a broader, community and sensor agnostic perspective. The SWE standards provide web service interfaces and XML-based encodings that enable the emerging vision of sensor webs – heterogeneous, geographically dispersed sensors and sensor networks that are universally discoverable and interoperable over computer networks or the Internet. The SWE suite of encodings and web services provide a foundation for sensor webs by offering standard functionality for:

- Discovering sensors
- Describing sensors and the methods by which those sensors derive observations
- Retrieving sensor observations (both archived and real-time)
- Tasking sensors
- Subscribing to and being notified of sensor alerts in real-time

The SWE suite consists of three encodings and four web services, described in Table 4. The CS/W, while not traditionally classified as a SWE web service, is included in the

table, since it provides important functionality for discovering SWE and other OGC web services. Other OGC web services such as the Web Map Service (WMS) and the Web Feature Service (WFS) are important in that they provide geographic context (e.g. background imagery and geographic features) to SWE sensors and data.

Table 4 - OGC SWE Encodings and Web Services

Encoding	Description
Sensor Model Language (SensorML)	Describes and models processes, sensors, and systems of sensors
Observations and Measurements (O & M)	Format for encoding sensor observation data
Transducer Markup Language (TML)	Optimized format for streaming low-level sensor descriptions and data in real-time
Web Service	**Description**
Sensor Observation Service (SOS)	Provides archived and near real-time access to sensors and their data. Sensors are described in SensorML or TML and sensor data is described in O & M or TML
Sensor Planning Service (SPS)	Provides access to controllable sensors and the means to task those sensors in a standard way
Sensor Alert Service (SAS)	Provides the ability to subscribe to and receive sensor alerts in real-time.
Web Notification Service (WNS)	Standardized asynchronous messaging/notification mechanism for receiving messages in many ways, including e-mail, Short Message Service, phone, etc.
Catalog Service for the Web (CS/W)	Provides functionality for discovering available OGC web services and objects accessible through those web services (e.g. sensors, geographic features, etc.)

Figure 3 (re-presented from [OGC 07-165]) illustrates the SWE concept and the broader sensor web concept that SWE enables.

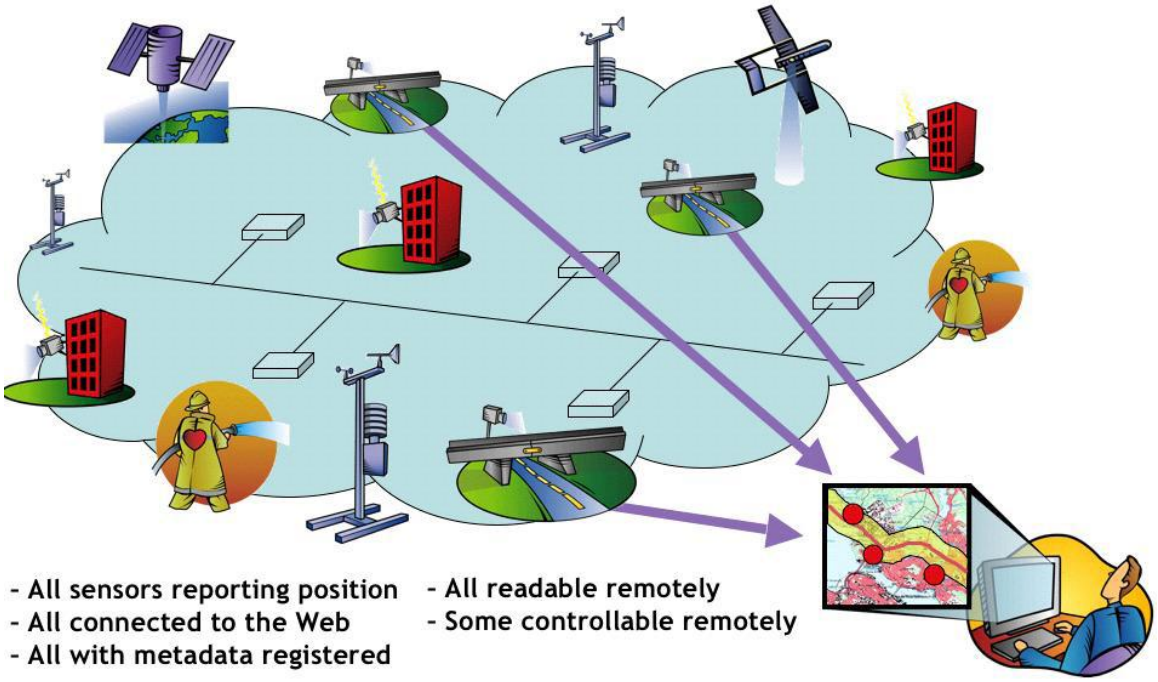

Figure 3 - The SWE Concept

The SWE standards also include a set of common data types and XML elements used across SWE encodings, called SWE Common. In the 1.0 versions of the standards, SWE Common is defined in sections 8 and 9 of the SensorML 1.0 specification and used in SensorML elements for defining data values and units of measure and used in practice in the O & M *result* element for describing sensor readings (O & M extended schemas). In the 2.0 versions of the specifications, SWE Common has been pulled out of SensorML into its own specification. In addition to SWE Common, it is also important to note that all OGC standard web services inherit common elements from the OGC Web Services (OWS) Common specification. In particular, the SWE services inherit common elements from OWS Common 1.1 [06-121r3].

8 Reasons for Integrating CCSI and SWE and Future Harmonization

Both CCSI and SWE seek to address the sensor interoperability problem but approach the problem from different perspectives and focus areas. CCSI defines a community specific standard for CBRN sensor interoperability, while SWE defines a broader, community agnostic set of standards for sensor interoperability. By integrating CCSI sensors into a SWE-based architecture, those sensors can be made available not only to the CBRN community and its applications but to the growing number of groups outside of the CBRN community that utilize SWE services and encodings. Furthermore, utilizing SWE allows additional sensor data (e.g. weather, video, or imagery data) to be integrated with CBRN sensor data without the need for converting between formats or the need for a client or command and control (C2) application to support multiple formats. In addition to electronic sensor systems, SWE is flexible enough to support describing soldiers and first responders as sensors. For example, the same SOS instance can provide access to

CCSI sensor data produced by an electronic sensor as well as visual observations recorded by one or more soldiers or first responders in the vicinity of that sensor. Given the fact that SWE messages often contain detailed, descriptive metadata and supporting information, they can easily be mapped to other common formats (e.g. Common Alerting Protocol (CAP)) to support the use of sensor data by existing applications. In terms of functionality, the CCSI and SWE encoding formats align well with one another, as illustrated in the table below. This functional alignment lends itself to exploring the potential harmonization of the two sets of standards.

Table 5 - CCSI-SWE Functional Alignment

Function	CCSI	OGC SWE
Describing Sensors	Sensor Definition File	SensorML Document
Describing Sensor Observations	Channel Message	O & M Document
Describing Tasking Commands	Standard Commands Document	SPS *DescribeTasking* Response
Submitting Tasking Messages	Command Channel Message	SPS *Submit* Request
Describing Alerts	Alerts Channel Message	SAS Alert, O & M document, others

By harmonizing CCSI standards with SWE and defining CCSI profiles of the SWE specifications in the future, competition between sensor standards can be minimized. In the military CBRN case, this reduces the burden on systems integrators and solution developers supporting CBRN and broader focused programs by eliminating the need to support multiple formats. As previously mentioned, JPEO-CBD is currently in the process of defining methodologies for web service-based access to CCSI sensors and data and is also defining the next version of the CCSI standards, meaning that there is the potential for incorporating SWE web service and encoding concepts into CCSI. The 2.0 versions of the SWE service specifications will include defined support for SOAP/WSDL, so this will comply with the CCSI requirement for CCSI web services to adhere to WS-I Basic Profile 1.1. Bearing this in mind, this report seeks to initiate and further the harmonization process by defining CCSI profiles of SWE web services and encodings that illustrate the concept that CBRN sensor information can be well represented and handled within SWE. Before defining the CCSI profiles of SWE, it is useful to review previous efforts related to integrating CCSI sensors into a SWE-based architecture conducted during OWS-6 as well as the additional work that was conducted as part of OWS-7.

9 OWS-6 Review

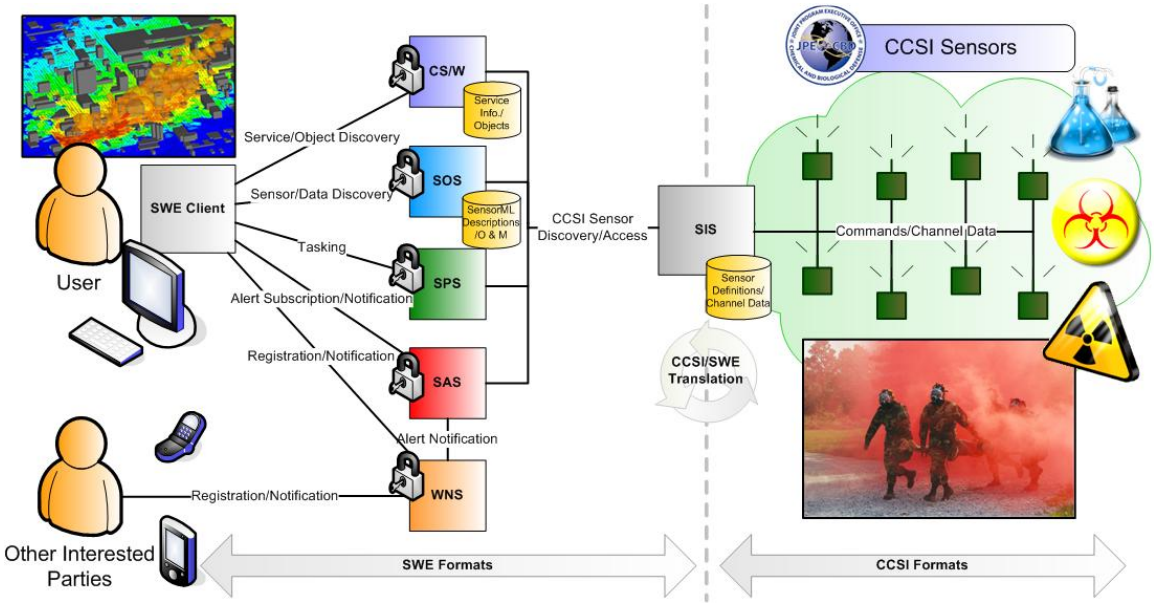

Figure 4 - OWS-6 CCSI-SWE Integrated Architecture

The OWS-6 efforts defined the architecture shown above in Figure 4 as a practical approach for integrating CCSI sensors and sensor data into an OGC SWE-based architecture. The following sections expand upon the key components within this architecture, notably the Sensor Interface Service (SIS) and the interaction between SWE services and the SIS.

9.1 Sensor Interface Service (SIS)

The OWS-6 CCSI-SWE ER describes the SIS as follows:

In order to bridge the gap between CCSI and SWE specifications and manage multiple CCSI sensors, there is a need for a common method of aggregating and discovering sensor descriptions and data from multiple CCSI sensors that can provide information to SWE services. In general, the SWE services have a common need for the following functionality:

- Access to lists of available sensors
- Access to sensor descriptions
- Access to sensor data and alerts
- Ability to send tasking commands

The SIS provides this common functionality as a middle-tier SOAP/WSDL-based web service (compliant with WS-I Basic Profile 1.1) that aggregates and connects CCSI sensors with interested consumers. The SIS in the CCSI world is somewhat analogous to the Smart Transducer Web Service (STWS) from the IEEE 1451 world in that it provides

a single web service interface that acts as an intermediary between a client, which could be a SWE web service instance, and one or more sensors and abstracts the details of how to communicate directly with the sensors themselves. In order to simplify SWE service development, the SIS also acts as a CCSI to SWE and SWE to CCSI translator and provides responses using SWE formats where possible.

In essence, the SIS provides a centralized, common mechanism for SWE services to obtain available CCSI sensors, retrieve CCSI sensor data and alerts, and task CCSI sensors. The SIS also performs CCSI to SWE and SWE to CCSI translation using eXtensible Stylesheet Language Transformation (XSLT) documents. Indeed, the primary technology enabling the SIS is the use of XSLT 1.0 transforms for translating CCSI Sensor Definition files and channel messages to SWE formats and vice versa. Since CCSI encodings and SWE encodings are both XML-based, XSLT provides a logical means for converting between formats. XSLT is also software development language independent and is supported by most modern software development languages. The OWS-6 efforts defined the SIS as a SOAP/WSDL-based web service that meets the CCSI requirements for a web service by complying with WS-I Basic Profile 1.1.

The SIS provides several operations summarized in the table below.

Table 6 - SIS Operations

Operation Name	Description
GetSensors	Returns a list of available CCSI sensors along with other relevant information such as supported channels and location
DescribeSensor	Returns a CCSI Sensor Definition file or a SensorML document describing the requested sensor
GetLatestObservation	Returns the most recent CCSI channel message or O & M document
DescribeTasking	Returns a list of supported commands for the requested sensor
SubmitCommand	Submits a command for the requested sensor

One important piece of information to note is that the SIS as defined in OWS-6 only stores the latest reading from each CCSI sensor rather than a time series of observations. It is the job of the SIS client to archive CCSI sensor observations, if archiving and retrieving multiple CCSI sensor observations is desired.

9.2 Utilizing the SIS with SWE Web Services

The following UML Sequence diagrams from the OWS-6 CCSI-SWE ER illustrate the interaction between various SWE web services and the SIS and corresponding CCSI sensors. Sequence diagrams for the CCSI SOS and SPS are provided below. In both

diagrams, a SWE client issues a *GetCapabilities* request to the corresponding SWE service, and the SWE service then invokes the *GetSensors* operation of the SIS. The response to a *GetSensors* operation request provides sufficient information to populate a SWE service Capabilities document, and SWE service generates a Capabilities document and returns the document to the SWE client in response to the original *GetCapabilities* request.

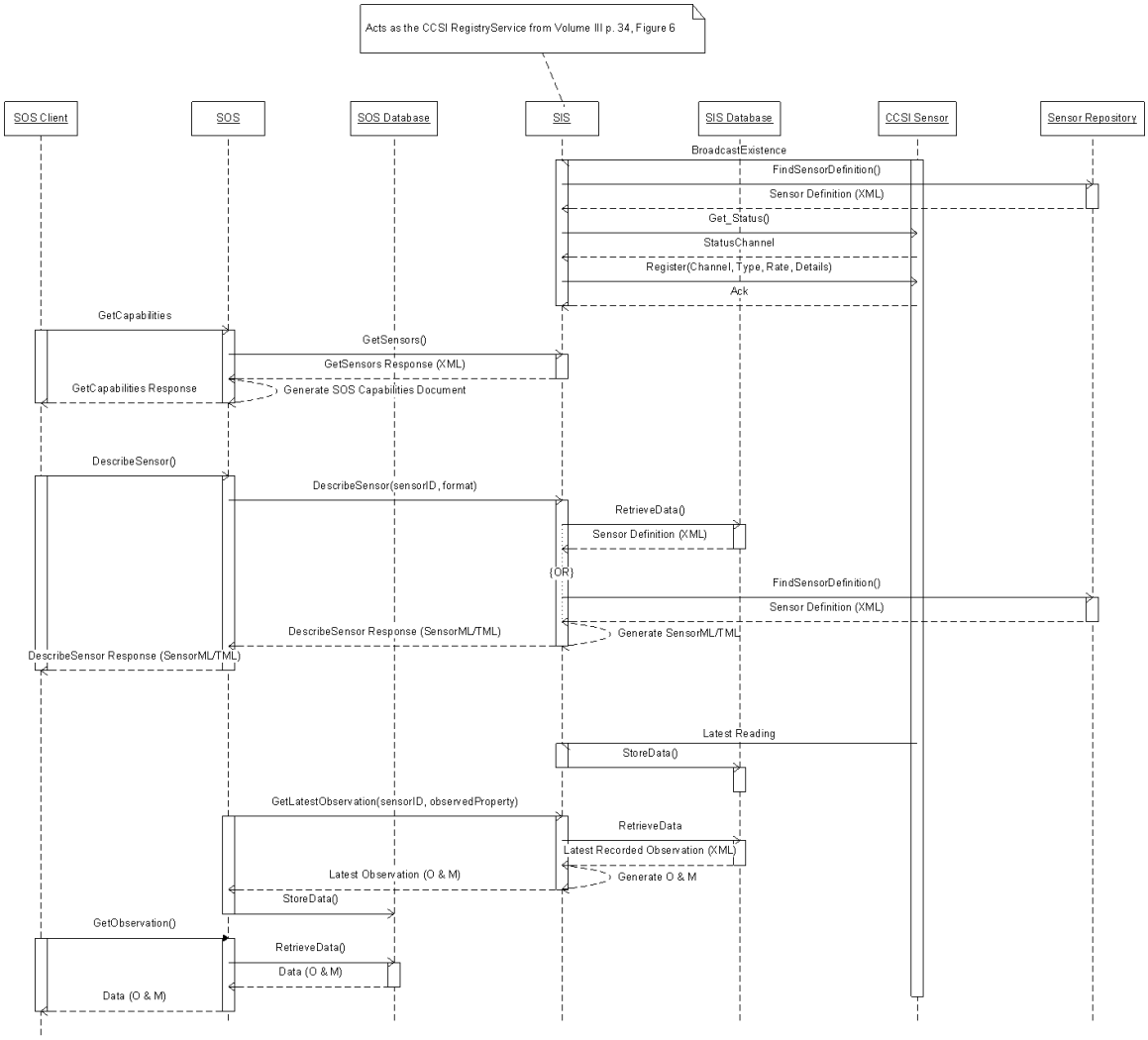

Figure 5 - SOS/SIS Interaction

When an SOS client issues a *DescribeSensor* request to the CCSI SOS, the CCSI SOS issues a *DescribeSensor* request to the SIS. The SIS retrieves a CCSI Sensor Definition file, which is either stored locally or retrieved from a Sensor Definition Repository, and uses the CCSI Sensor Definition to SensorML XSLT transform to generate a valid SensorML document. The SIS then returns this document to the CCSI SOS, and the CCSI SOS returns the document as is to the SOS client.

GetObservation requests are handled differently. Since CCSI sensor data is constantly pushed from a CCSI sensor to a controlling application, the SIS utilizes an underlying CCSI Sensor Management component to continually monitor and store CCSI channel data. When an SOS client issues a *GetObservation* request, the CCSI SOS can either issue a *GetLatestObservation* request directly to the SIS, which will cause the SIS to retrieve the latest CCSI channel data from its local datastore and convert that data to O & M using XSLT, or the CCSI SOS can utilize its own datastore for archiving CCSI channel data. The CCSI SOS can continually issue *GetLatestObservation* requests to the SIS and then archive the responses in its own datastore. This allows the CCSI SOS to provide CCSI channel data for a time range rather than just the latest recorded time.

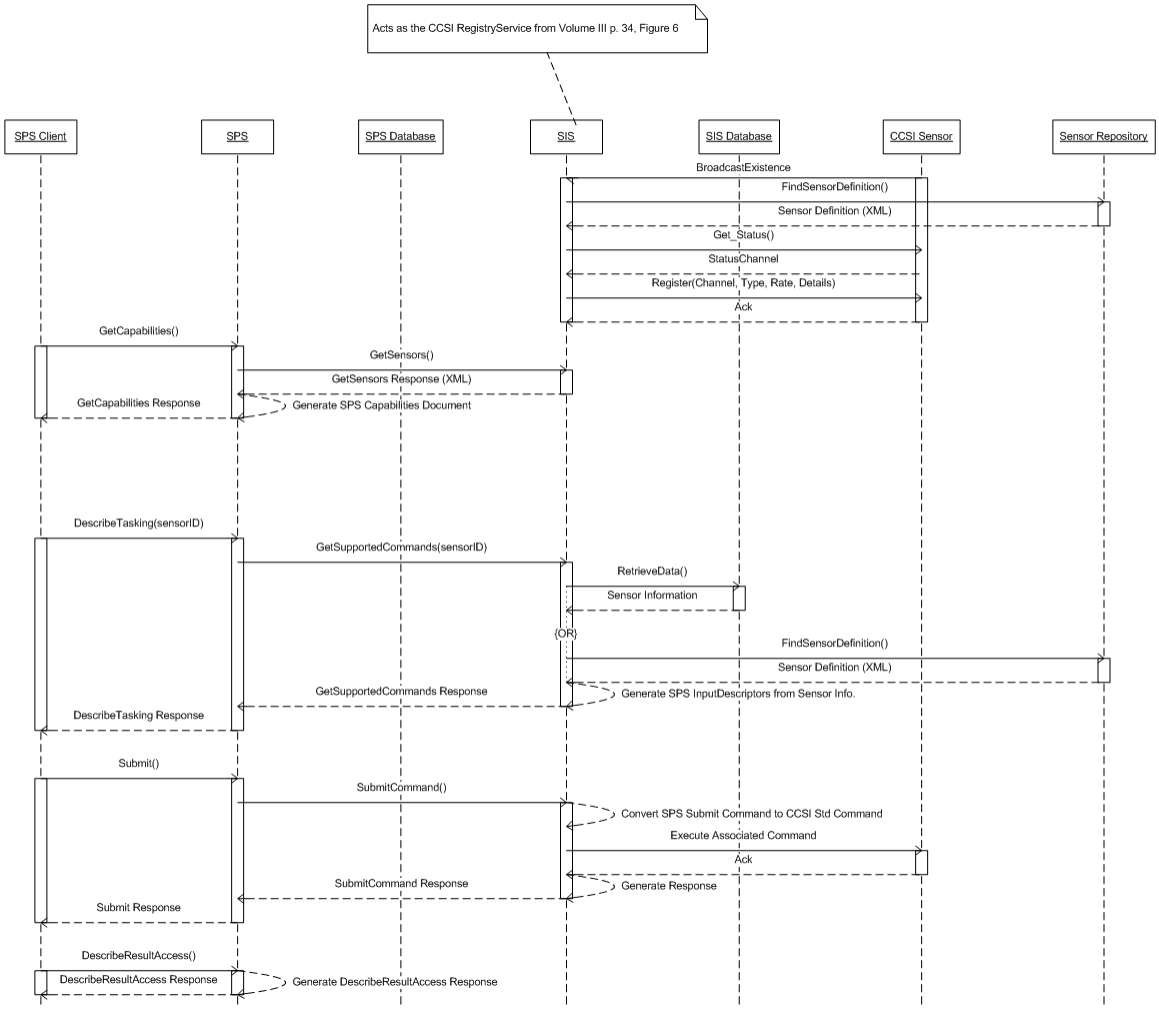

Figure 6 - SPS/SIS Interaction

When an SPS client issues a *DescribeTasking* request to the CCSI SPS, the CCSI SPS issues a *GetSupportedCommands* request to the SIS, passing in a sensorID parameter. The SIS then loads the CCSI Sensor Definition file for that sensor from a local datastore or retrieves it from a Sensor Definition Repository. The SIS utilizes the Sensor Definition file along with the CCSI Standard Commands document and passes this information through an XSLT transform, which produces a valid SPS *DescribeTasking* response document. The CCSI SPS then returns this *DescribeTasking* response

document as is to the SPS client. The SPS client can then use the *DescribeTasking* response document to generate a valid SPS *Submit* request.

When the SPS client issues a *Submit* request to the CCSI SPS, the CCSI SPS issues a *SubmitCommand* request to the SIS. The SIS converts the SPS *Submit* request to a valid CCSI command channel message using XSLT, and the CCSI Sensor Management component sends the command channel message over its connection with the CCSI sensor identified in the request. The SIS then generates an appropriate SPS *Submit* response and returns it to the CCSI SPS, which returns the same response to the SPS client without modification. The SPS client may wish to determine where to retrieve the results of any issued Submit requests. This is handled through a *DescribeResultAccess* request. When the SPS client issues a *DescribeResultAccess* request to the CCSI SPS, the CCSI SPS generates an appropriate *DescribeResultAccess* response document that points to the URL of the associated CCSI SOS and returns this document to the SPS client. This interaction does not require the use of the SIS, since the response can be a static document.

10 Integrating CCSI with SWE Now

10.1 Updates to OWS-6 Work

The OWS-6 CCSI-SWE ER identified several areas that needed to be addressed or updated in the future. The OWS-7 efforts involved reviewing the OWS-6 ER and determining areas that still needed to be addressed or updated. As part of the OWS-7 effort, the XSLT files for converting CCSI to SWE were updated to support additional CCSI information and channel data, and the Sensor Definition to SensorML conversion was updated to provide better encoding of CCSI channel outputs.

10.1.1 XSLT Updates/Additional Updates that are Necessary

As described in the OWS-6 CCSI-SWE ER, the first versions of the XSLT translations did not account for the *Hdr* element's *dtg* or *mod* attributes.

In addition, the XSLT typically does not use information contained in the *Hdr* element's attributes, except in the case of the *dtg* attribute in some instances. In most cases, this is perfectly fine, since the *Hdr* attribute information does not add any new information to the actual data contained within the channel data message, but, in the case of the *mod* attribute, this may leave off important information, particularly if the *mod* attribute is some value other than "N" representing normal operation. The *mod* attribute can vary depending on how the CCSI sensor is initialized and can be changed by a controlling application or user through a *Set_Ccsi_Mode* command. In the future, the XSLT should be modified to include the value of the *mod* attribute of the *Hdr* element so as not to leave off any important information in resulting O & M observations.

Following this recommendation, the XSLTs included in this report for converting CCSI Sensor Definitions to SensorML 1.0.1 and CCSI channel data to O & M 1.0.0 have been

updated to include the *Hdr* element's *dtg* and *mod* elements in the case of HRTBT channel messages and the *mod* attribute in the case of all other channel messages.

The XSLTs have also been updated to include support for the HRTBT channel. The channel includes a SWE Common *DataRecord* with two fields: dtg and mod, where dtg is expressed as a SWE Common *Time* data type with a *referenceTime* attribute value of 1970-01-01T00:00:00Z and a unit of measure in seconds, reflecting the fact that the *dtg* attribute represents seconds since 1970 or Unix time.

The CCSI Sensor Definition to SensorML 1.0.1 translation was also updated to ensure that the *outputs* section includes one distinct *output* element per supported CCSI channel. The translation included in the OWS-6 CCSI-SWE ER combines all channels into a single *output*. Using a separate *output* element per channel seems to align better with the way CCSI sensors work.

While the XSLTs were updated in several areas, there are currently several updates remaining that should be addressed at some point in the future to more fully integrate CCSI sensors into a SWE-based architecture. The *TimeDate* channel and the *Location* channel provide interesting cases for encoding within O & M messages, as they illustrate cases where redundancy may occur in O & M. The following XML documents provide CCSI *TimeDate* and *Location* channel examples.

```
<?xml version="1.0" encoding="utf-8"?>
<Hdr sid="RadDetector_2" msn="96355" mod="N" dtg="1272899074" chn="t" len="374">
    <CCSIDoc>
        <Msg>
            <TimeDateChn>
                <TimeDateReport Time="20100503100434" Source="network"/>
            </TimeDateChn>
        </Msg>
    </CCSIDoc>
</Hdr>
```

```
<?xml version="1.0" encoding="utf-8"?>
<Hdr sid="RadDetector_2" msn="96355" mod="N" dtg="1272899074" chn="l" len="374">
  <CCSIDoc>
    <Msg>
      <LocationChn>
        <LocationReport>
            <Location Lat="35.0657" Lon="-70.1234" Alt="0.0"/>
            <Source>manual</Source>
         </LocationReport>
      </LocationChn>
    </Msg>
  </CCSIDoc>
</Hdr>
```

The following O & M examples illustrate how location information can be encoded in SWE.

```
<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector_2_96359"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
```



```
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <gml:description>CCSI Sensor Observation Instance</gml:description>
    <gml:name>CCSI Sensor Observation Instance (RadDetector_2_96359)</gml:name>
    <samplingTime>
        <gml:TimeInstant>
            <gml:timePosition>2010-05-03T10:04:49Z</gml:timePosition>
        </gml:TimeInstant>
    </samplingTime>
    <procedure xlink:href="RadDetector_2"/>
    <observedProperty>
        <swe:CompositePhenomenon gml:id="LOC_RadDetector_2_96359" dimension="5">
            <gml:name>CCSI LOC Phenomenon</gml:name>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:OGC::latitude"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:OGC::longitude"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:OGC::altitude"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::source"/>
        </swe:CompositePhenomenon>
    </observedProperty>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>CCSI Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor_Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">35.0657 -
70.1234</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataRecord gml:id="CCSI_LOC_DATA_RECORD">
            <swe:field name="Mode">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                    <swe:value>Normal</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="Latitude">
                <swe:Quantity definition="urn:ogc:def:phenomenon:OGC::latitude"
axisID="latitude">
                    <swe:uom code="deg"/>
                    <swe:value>35.0657</swe:value>
                </swe:Quantity>
            </swe:field>
            <swe:field name="Longitude">
                <swe:Quantity definition="urn:ogc:def:phenomenon:OGC::longitude"
axisID="longitude">
                    <swe:uom code="deg"/>
                    <swe:value>-70.1234</swe:value>
                </swe:Quantity>
            </swe:field>
            <swe:field name="Altitude">
                <swe:Quantity definition="urn:ogc:def:phenomenon:OGC::altitude"
axisID="altitude">
                    <swe:uom code="m"/>
                    <swe:value>0.0</swe:value>
                </swe:Quantity>
            </swe:field>
            <swe:field name="Source">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Source">
                    <swe:value>manual</swe:value>
                </swe:Category>
            </swe:field>
        </swe:DataRecord>
    </result>
```



```
</Observation>

<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector_2_96359"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <gml:description>CCSI Sensor Observation Instance</gml:description>
    <gml:name>CCSI Sensor Observation Instance (RadDetector_2_96359)</gml:name>
    <samplingTime>
        <gml:TimeInstant>
            <gml:timePosition>2010-05-03T10:04:49Z</gml:timePosition>
        </gml:TimeInstant>
    </samplingTime>
    <procedure xlink:href="RadDetector_2"/>
    <observedProperty>
        <swe:CompositePhenomenon gml:id="LOC RadDetector_2_96359" dimension="3">
            <gml:name>CCSI LOC Phenomenon</gml:name>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:OGC::location"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::source"/>
        </swe:CompositePhenomenon>
    </observedProperty>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>CCSI Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor_Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">35.0657 -
70.1234</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataRecord gml:id="CCSI_LOC_DATA_RECORD">
            <swe:field name="Mode">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                    <swe:value>Normal</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="location">
                <swe:Vector definition="urn:ogc:def:phenomenon:OGC::location">
                    <swe:coordinate name="latitude">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:OGC::latitude"
axisID="latitude">
                            <swe:uom code="deg"/>
                            <swe:value>35.0657</swe:value>
                        </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="longitude">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:OGC::longitude"
axisID="longitude">
                            <swe:uom code="deg"/>
                            <swe:value>-70.1234</swe:value>
                        </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="altitude">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:OGC::altitude"
axisID="altitude">
                            <swe:uom code="m"/>
```



```
                        <swe:value>0.0</swe:value>
                    </swe:Quantity>
                </swe:coordinate>
            </swe:Vector>
        </swe:field>
        <swe:field name="Source">
            <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Source">
                <swe:value>manual</swe:value>
            </swe:Category>
        </swe:field>
    </swe:DataRecord>
</result>
</Observation>
```

The examples above illustrate different ways of encoding location using SWE Common in the O & M *result* element. In the first case, the location fields are included as separate fields within the *DataRecord* for the observation. In the second case, the location fields are encoded within a SWE Common *Vector* as a location field in the *DataRecord*. In the location examples above, the location of the sensor in the O & M *result* element will most likely match the *SamplingPoint position* element included in the O & M *featureOfInterest SamplingPoint* (if a *SamplingPoint* is used), so there is some redundancy, which needs to be considered. The same is potentially true for the *TimeDate* channel. The O & M *samplingTime* could match the value found in the time field in the *result* element, but it also may not, depending on how the implementer encodes this information.

The CCSI standard READGS channel supports encoding spectrum and weather information; however, this information is not supported by the available CCSI emulator and no real CCSI sensors were available to support the OWS-7 efforts, so the examples below have not been checked for validity. Given the inherent complexity of spectrum information, the lack of sample spectrum readings from the CCSI emulator makes producing an O & M encoded spectrum example difficult. Weather information is easier to generate, and so the XML example below illustrates a CCSI READGS message with weather data encoded in O & M 1.0.

```
<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector_2_96359"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <gml:description>CCSI Sensor Observation Instance</gml:description>
    <gml:name>CCSI Sensor Observation Instance (RadDetector_2_96359)</gml:name>
    <samplingTime>
        <gml:TimeInstant>
            <gml:timePosition>2010-05-03T10:04:49Z</gml:timePosition>
        </gml:TimeInstant>
    </samplingTime>
    <procedure xlink:href="RadDetector_2"/>
    <observedProperty>
        <swe:CompositePhenomenon gml:id="READGS RadDetector_2_96359" dimension="4">
            <gml:name>CCSI READGS Phenomenon</gml:name>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::weatherDataTypeSource"/>
```



```
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::weatherDataTypeTime"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::weatherDataTypeItems"/>
        </swe:CompositePhenomenon>
    </observedProperty>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>CCSI Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor_Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">35.0657 -
70.1234</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataRecord gml:id="CCSI_READGS_DATA_RECORD">
            <swe:field name="Mode">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                    <swe:value>Normal</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="weatherSource">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::weatherSource">
                    <swe:value>met_sensor</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="weatherTime">
                <swe:Time definition="urn:ogc:def:phenomenon:JPEO-CBD::weatherTime">
                    <swe:value>2010-05-03T10:04:49Z</swe:value>
                </swe:Time>
            </swe:field>
            <swe:field name="weatherItemList">
                <swe:DataRecord definition="urn:ogc:def:phenomenon:JPEO-
CBD::weatherItemList">
                    <swe:field name="location">
                        <swe:Vector definition="urn:ogc:def:phenomenon:OGC::location">
                            <swe:coordinate name="latitude">
                                <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::latitude">
                                    <swe:uom code="deg"/>
                                    <swe:value>35.0657</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                            <swe:coordinate name="longitude">
                                <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::longitude">
                                    <swe:uom code="deg"/>
                                    <swe:value>-70.1234</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                            <swe:coordinate name="altitude">
                                <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::altitude">
                                    <swe:uom code="m"/>
                                    <swe:value>4.0</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                        </swe:Vector>
                    </swe:field>
                    <swe:field name="windSpeed">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::windSpeed">
                            <swe:uom code="mph"/>
                            <swe:value>5.0</swe:value>
```



```
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="windDirection">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::windDirection">
                            <swe:uom code="deg"/>
                            <swe:value>35.0</swe:value>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="temperature">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::temperature">
                            <swe:uom code="degF"/>
                            <swe:value>74.0</swe:value>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="humidity">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::relativeHumidity">
                            <swe:uom code="%"/>
                            <swe:value>45.0</swe:value>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="precipitationType">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::precipitationType">
                            <swe:value>DRZLE</swe:value>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="precipitationRate">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC::precipitationRate">
                            <swe:uom code="inph"/>
                            <swe:value>3.0</swe:value>
                        </swe:Quantity>
                    </swe:field>
                </swe:DataRecord>
            </swe:field>
        </swe:DataRecord>
    </result>
</Observation>
```

As mentioned earlier, a CCSI channel message can contain one or more data readings by providing multiple *Msg* elements. If a CCSI channel message contains multiple *Msg* elements, then the O & M document describing that message should include a *result* element with a SWE Common *DataArray* with a SWE *DataRecord elementType* and a *TextBlock* encoding. The O & M *samplingTime* element should include a GML *TimePeriod* that contains a *beginPosition* element that corresponds to the first reading time and an *endPosition* element that corresponds to the last reading time in the array. Using the *DataArray* with *TextBlock* encoding approach eliminates redundant XML metadata and ensures that the O & M document is as compact as possible. This approach will be described in more detail in the CCSI profile of O & M that appears later in this document. The following example illustrates O & M describing multiple readings from the READGS channel.

```
<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector_1_119837"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
```



```
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <gml:description>CCSI Sensor Observation Instance</gml:description>
    <gml:name>CCSI Sensor Observation Instance (RadDetector_1_119837)</gml:name>
    <samplingTime>
        <gml:TimePeriod>
            <gml:beginPosition>2010-05-05T13:48:18Z</gml:beginPosition>
            <gml:endPosition>2010-05-05T13:58:18Z</gml:endPosition>
        </gml:TimePeriod>
    </samplingTime>
    <procedure xlink:href="urn:ogc:def:procedure:JPEO-CBD::RadDetector_1"/>
    <observedProperty>
        <swe:CompositePhenomenon gml:id="READGS_RadDetector_1_119837" dimension="10">
            <gml:name>CCSI READGS Phenomenon</gml:name>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Sensor"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Detect"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose"/>
        </swe:CompositePhenomenon>
    </observedProperty>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>CCSI Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">18.556825 -
72.297935</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataArray definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS">
            <swe:elementCount>
                <swe:Count>
                    <swe:value>3</swe:value>
                </swe:Count>
            </swe:elementCount>
            <swe:elementType name="ReadingsArray">
                <swe:DataRecord gml:id="CCSI_READGS_DATA_RECORD">
                    <swe:field name="Time">
                        <swe:Time definition="urn:ogc:def:phenomenon:ISO8601::time"/>
                    </swe:field>
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="READGS Sensor">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                    </swe:field>
                    <swe:field name="READGS_ReadingID">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                            <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                        </swe:Category>
```



```
                    </swe:field>
                    <swe:field name="READGS_Detect">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                    </swe:field>
                    <swe:field name="READGS_Level">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                    </swe:field>
                    <swe:field name="READGS_Id">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                    </swe:field>
                    <swe:field name="READGS_DetailsAvailable">
                        <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                    </swe:field>
                    <swe:field name="Sensor Unique Data">
                        <swe:DataRecord definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:SUD">
                            <swe:field name="FilteredDoseRate">
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                                    <swe:uom code="cGy/hr"/>
                                </swe:Quantity>
                            </swe:field>
                            <swe:field name="MissionDose">
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                                    <swe:uom code="cGy"/>
                                </swe:Quantity>
                            </swe:field>
                        </swe:DataRecord>
                    </swe:field>
                </swe:DataRecord>
            </swe:elementType>
            <swe:encoding>
                <swe:TextBlock decimalSeparator="." blockSeparator="@"
tokenSeparator="||"/>
            </swe:encoding>
            <swe:values>2010-05-
05T13:48:18Z||Normal||SC001||R000239709||Gamma||20.09765625||||true||1.57||20.098@2010-
05-
05T13:53:18Z||Normal||SC001||R000239710||Gamma||30.09765625||||true||2.57||30.098@2010-
05-
05T13:58:18Z||Normal||SC001||R000239711||Gamma||40.09765625||||true||3.57||40.098</swe:va
lues>
        </swe:DataArray>
    </result>
</Observation>
```

The XSLTs provided in this report have not been updated to produce output based on the examples above, but updating the XSLTs in the future should not be difficult.

10.1.2 Supporting Additional Sensor Information

In some cases, it may be useful to include additional information or sensor metadata not found in the CCSI standard Sensor Definition file when describing a CCSI sensor in SensorML. SensorML is flexible enough to support detailed sensor metadata describing CCSI sensors, including characteristics of a sensor such as size, weight, and power (often referred to as SWaP in the military and DoD communities), identification information such as name and serial number, as well as other information. This additional information could be particularly useful in military environments, where SWaP characteristics and operating temperatures might be used in mission planning to decide what sensors to deploy for particular missions. The link below provides a detailed

SensorML example for a gamma radiation detector and illustrates the different types of information that could be provided in SensorML to augment basic CCSI Sensor Definition information:

http://vast.uah.edu/downloads/sensorML/v1.0/examples/sensors/Gamma_2070.xml

In order to support additional SensorML information, one could update the XSLT translations provided in this document to allow SensorML elements to be fed in to the translation as input parameters, or one could perform the XSLT translation from CCSI to SensorML and then merge the resulting SensorML with the additional SensorML elements. These additional SensorML elements would need to be stored in flat files or in a database and would need to be referenced by sensor ID or by some combination of manufacturer and model number, since these characteristics would most likely be specific to a particular sensor model rather than a particular instance of that sensor model. Implementers utilizing CCSI in a SWE-based environment should consider adding additional sensor information to basic CCSI Sensor Definition information, if the additional information aids in discovering or determining the most appropriate sensor for a particular mission.

10.2 Usage of the SIS and Architectural Components

Rather than mandating that the SIS be developed as a separate SOAP/WSDL based web service as described in the OWS-6 CCSI-SWE ER, this document envisions the SIS as an abstract concept that can be defined and used in multiple ways. In the case where the SIS and SWE services are developed by separate organizations or where web service-based access to CCSI sensors through SWE interfaces is needed by one group or organization and SOAP/WSDL-based web service access to CCSI sensors and data in CCSI formats is needed by another group or organization, developing the SIS as a SOAP/WSDL based web service along with additional SWE services that communicate with the SIS is the best approach. This allows different organizations to develop capabilities independently and provides multiple options for accessing CCSI sensors and data. The main disadvantage of this approach is the additional overhead inherent in web service to web service communication through HTTP requests and responses and XML messaging. If a single organization is developing SWE services that interact with CCSI sensors, and there is no requirement for SOAP/WSDL web service based access to CCSI sensors and data, then it may be more beneficial to develop the SIS as a software library that the SWE services use internally. The SIS software library would be developed in whatever language the SWE services are developed. This approach eliminates the overhead of web service to web service interaction and, as a result, makes the SWE services more efficient. In this case, the SIS software library should implement the same functions and return the same information that the SIS web service would provide but without the additional SOAP wrapper around operation responses.

In general, using the SIS concepts defined in OWS-6 provides a worthwhile approach for integrating CCSI sensors and data into a SWE-based architecture now. Whether or not the SIS concepts are utilized is ultimately an implementation decision, but there is definitely a need for the functionality provided by the SIS (e.g. aggregation of sensors and data translation) in a CCSI and SWE hybrid architecture.

10.3 Other Approaches

In addition to the mandatory operations of *GetCapabilities*, *DescribeSensor*, and *GetObservation*, the SOS 1.0 specification defines an optional Transactional profile, often referred to as SOS-T, for dynamically registering new sensors through the *RegisterSensor* operation and inserting sensor data through the *InsertObservation* operation. The Transactional Profile should be implemented in cases where the SOS is not intended to be used in conjunction with an SPS, or more specifically, in cases where remotely retrieving sensor descriptions and data is of primary importance and remote tasking of CCSI sensors through web services is not required or needed. This would most likely be useful in cases where a sensor is configured by the operator before being deployed in the field or the sensor is configured by a local operator out in the field. If remote tasking of CCSI sensors is necessary, then the SIS approach described above should be used, where both the SOS and SPS utilize the SIS web service or SIS libraries to gain shared access to CCSI sensors and their data. One could implement an SPS to allow remote tasking in conjunction with an SOS-T, but this approach is less practical than the SIS approach and eliminates most of the value provided by the SOS-T for dynamically registering sensors and publishing their data, since the SPS must have a priori knowledge of the sensors to which it can send tasking requests and know where to send tasking requests. This forces the SPS to be a controlling application of the CCSI sensors in addition to the controlling application that is connected to each sensor and interacting with the SOS-T.

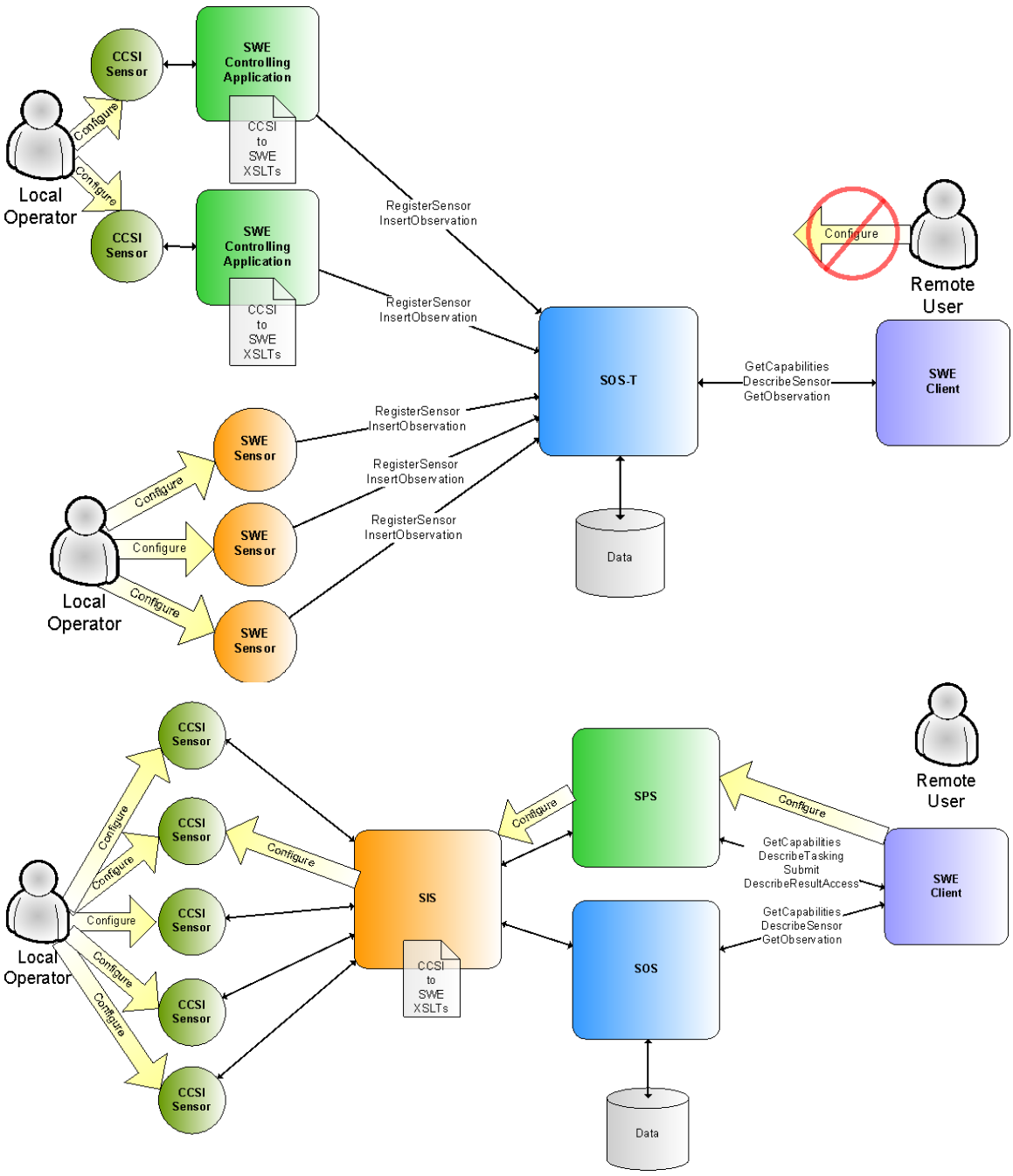

Figure 7 - SOS-T vs. SOS and SPS

In the SOS-T approach (illustrated in the top section of the figure above), one or more controlling applications connect to one or more CCSI sensors. Each controlling application will issue a *RegisterSensor* command to a centralized SOS-T when it connects to a CCSI sensor and will issue *InsertObservation* commands to the SOS-T as new channel data becomes available. The controlling application(s) can be configured to communicate with one or more CCSI sensors and can use the XSLT translations described in the OWS-6 ER and updated in this report for translating from CCSI formats to SWE formats. When each CCSI sensor successfully registers itself with the SOS-T

and begins inserting observations, it becomes discoverable to SOS clients connected to that SOS-T instance. This approach also allows for additional SWE-compliant sensors to be registered with the same SOS with which CCSI sensors are registered. In the future, one can envision a scenario where SWE-compliant sensors themselves register with an SOS and insert their observations dynamically.

The SOS 1.0.0 specification does not provide clarification on how SOS-T instances should handle registering sensors that support multiple observed properties through separate outputs. A *RegisterSensor* request only allows a single O & M template of type *Observation* to be provided that tells the SOS server what the corresponding data will look like when inserted using an *InsertObservation* request. For CCSI sensors that support multiple channels, it is uncertain how dynamic registration of a single sensor that supports multiple output channels would work. If multiple *RegisterSensor* requests are issued for the same sensor with different *ObservationTemplate* values, then the server could interpret this as there being multiple potential observation types, or the server could just overwrite the stored observation template with the latest template received. If the server interprets multiple *RegisterSensor* requests with different *ObservationTemplate* values as being different potential observation types, then the SOS could create separate offerings for each unique observation type or could create an offering for each sensor and group all unique observation types as observed properties under that offering. The *ObservationTemplate* could contain an *ObservationCollection* with an O & M template defined for each distinct output from the sensor, but there is no text in the SOS specification that mandates how this should be handled, and the use of an *ObservationCollection* most likely conflicts with the restriction of type *Observation* on the *ObservationTemplate* element. This ambiguity needs to be clarified in the SOS 2.0 specification. In most cases related to CCSI, using an O & M template for the READGS channel should suffice, since this is typically the most important channel.

10.3.1 Example RegisterSensor Request

```
<?xml version="1.0" encoding="UTF-8"?>
<RegisterSensor xmlns="http://www.opengis.net/sos/1.0"
xmlns:tml="http://www.opengis.net/tml" xmlns:om="http://www.opengis.net/om/1.0"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosAll.xsd" service="SOS" version="1.0.0">
    <SensorDescription>
        <System gml:id="CCSI_Sensor"
xsi:schemaLocation="http://www.opengis.net/sensorML/1.0.1
http://schemas.opengis.net/sensorML/1.0.1/sensorML.xsd"
xmlns="http://www.opengis.net/sensorML/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:om="http://www.opengis.net/om/1.0" xmlns:gml="http://www.opengis.net/gml"
xmlns:tml="http://www.opengis.net/tml" xmlns:swe="http://www.opengis.net/swe/1.0.1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
            <gml:description>
      AN/UDR-14 RADIAC radiological detector.
    </gml:description>
            <keywords>
                <KeywordList>
                    <keyword>CCSI</keyword>
                    <keyword>CBRN</keyword>
                    <keyword>Sensor</keyword>
                </KeywordList>
            </keywords>
            <identification>
```



```
                <IdentifierList>
                    <identifier name="uniqueID">
                        <Term definition="urn:ogc:def:identifier:OGC:uniqueID">
                            <value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</value>
                        </Term>
                    </identifier>
                    <identifier name="SensorName">
                        <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorName">
                            <value>AN/UDR-14</value>
                        </Term>
                    </identifier>
                    <identifier name="class">
                        <Term definition="urn:ogc:def:identifier:JPEO-
CBD::SensorType:class">
                            <value>RAD</value>
                        </Term>
                    </identifier>
                    <identifier name="variant">
                        <Term definition="urn:ogc:def:identifier:JPEO-
CBD::SensorType:variant">
                            <value>PNT</value>
                        </Term>
                    </identifier>
                    <identifier name="name">
                        <Term definition="urn:ogc:def:identifier:JPEO-
CBD::SensorType:name">
                            <value>SC001</value>
                        </Term>
                    </identifier>
                    <identifier name="SensorModel">
                        <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorModel">
                            <value>TBD</value>
                        </Term>
                    </identifier>
                    <identifier name="SensorDescription">
                        <Term definition="urn:ogc:def:identifier:JPEO-
CBD::SensorDescription">
                            <value>Radiation detector adapted to support CCSI within FCS
platforms.</value>
                        </Term>
                    </identifier>
                </IdentifierList>
            </identification>
            <classification>
                <ClassifierList>
                    <classifier name="intendedApplication">
                        <Term definition="urn:ogc:def:classifier:OGC:application">
                            <value>CBRN Detection</value>
                        </Term>
                    </classifier>
                    <classifier name="sensorType">
                        <Term definition="urn:ogc:def:classifier:OGC:sensorType">
                            <value>Radiological</value>
                        </Term>
                    </classifier>
                </ClassifierList>
            </classification>
            <characteristics name="HW/SW Characteristics">
                <swe:DataRecord definition="urn:ogc:def:property:powerRequirement">
                    <swe:field name="CCSI Version">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::CCSIVersion">
                            <swe:value>1.0.0.0</swe:value>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="HW Version">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::HWVersion">
                            <swe:value>1.0</swe:value>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="SW Version">
```



```
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::SWVersion">
                            <swe:value>1.0</swe:value>
                        </swe:Category>
                    </swe:field>
                </swe:DataRecord>
            </characteristics>
            <capabilities name="Gamma Radiation Dose Measurement Properties">
                <swe:DataRecord definition="urn:ogc:def:property:JPEO-
CBD::capabilityInformation">
                    <swe:field name="itemType">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::itemType">
                            <swe:value>Gamma Radiation Dose</swe:value>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="levelUnits">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::levelUnits">
                            <swe:value>cGy</swe:value>
                        </swe:Category>
                    </swe:field>
                </swe:DataRecord>
            </capabilities>
            <capabilities name="Gamma Radiation Dose Rate Measurement Properties">
                <swe:DataRecord definition="urn:ogc:def:property:JPEO-
CBD::capabilityInformation">
                    <swe:field name="itemType">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::itemType">
                            <swe:value>Gamma Radiation Dose Rate</swe:value>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="levelUnits">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::levelUnits">
                            <swe:value>cGy/hr</swe:value>
                        </swe:Category>
                    </swe:field>
                </swe:DataRecord>
            </capabilities>
            <contact>
                <ContactList>
                    <member xlink:role="urn:ogc:def:identifier:OGC::manufacturer">
                        <ResponsibleParty>
                            <individualName>Manufacturer Name</individualName>
                            <contactInfo>
                                <address>
                                    <deliveryPoint>Canberra Dover</deliveryPoint>
                                    <city>Dover</city>
                                    <administrativeArea>NJ</administrativeArea>
                                </address>
                            </contactInfo>
                        </ResponsibleParty>
                    </member>
                    <member xlink:role="urn:ogc:def:identifier:OGC::program">
                        <ResponsibleParty>
                            <individualName>Program Manager Name</individualName>
                            <organizationName>JPEO-CBD</organizationName>
                            <contactInfo>
                                <address>

    <electronicMailAddress>programmanager@us.army.mil</electronicMailAddress>
                                </address>
                            </contactInfo>
                        </ResponsibleParty>
                    </member>
                </ContactList>
            </contact>
            <spatialReferenceFrame>
                <gml:EngineeringCRS gml:id="SENSOR_FRAME">
                    <gml:srsName>CCSI Sensor Spatial Frame</gml:srsName>
```



```
                    <gml:usesCS xlink:href="urn:ogc:def:cs:xyzFrame"/>
                    <gml:usesEngineeringDatum>
                        <gml:EngineeringDatum gml:id="SENSOR_DATUM">
                            <gml:datumName>CCSI Sensor Spatial Datum</gml:datumName>
                            <gml:anchorPoint>Unknown</gml:anchorPoint>
                        </gml:EngineeringDatum>
                    </gml:usesEngineeringDatum>
                </gml:EngineeringCRS>
            </spatialReferenceFrame>
            <interfaces/>
            <inputs>
                <InputList>
                    <input name="genericCBRNHazard">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::genericCBRNHazard"/>
                    </input>
                </InputList>
            </inputs>
            <outputs>
                <OutputList>
                    <output name="ALERTS">
                        <swe:DataRecord gml:id="ALERTS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS">
                            <swe:field name="Mode">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                            </swe:field>
                            <swe:field name="ALERTS_Event">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Event">
                                    <swe:constraint>
                                        <swe:AllowedTokens>
                                            <swe:valueList>ALERT DEALERT WARN DEWARN
NONE</swe:valueList>
                                        </swe:AllowedTokens>
                                    </swe:constraint>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="ALERTS_Source">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Source">
                                    <swe:constraint>
                                        <swe:AllowedTokens>
                                            <swe:valueList>none detector bit
tamper</swe:valueList>
                                        </swe:AllowedTokens>
                                    </swe:constraint>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="ALERTS_AlertId">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:AlertId"/>
                            </swe:field>
                            <swe:field name="ALERTS_BIT_Component">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Component">
                                    <swe:constraint>
                                        <swe:AllowedTokens>
                                            <swe:valueList>CC PDIL SC UIC WCC
DPC</swe:valueList>
                                        </swe:AllowedTokens>
                                    </swe:constraint>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="ALERTS BIT Executed">
                                <swe:Time definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Executed" referenceTime="1970-01-01">
                                    <swe:uom code="s"/>
                                </swe:Time>
                            </swe:field>
                            <swe:field name="ALERTS_BIT_Result">
```



```
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Result">
                                    <swe:constraint>
                                        <swe:AllowedTokens>
                                            <swe:valueList>PASS FAIL</swe:valueList>
                                        </swe:AllowedTokens>
                                    </swe:constraint>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="ALERTS_BIT_Level">
                                <swe:Count definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Level"/>
                            </swe:field>
                            <swe:field name="ALERTS_BIT_ErrorDescription">
                                <swe:Text definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:ErrorDescription"/>
                            </swe:field>
                            <swe:field name="ALERTS Tamper">
                                <swe:Text definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Tamper"/>
                            </swe:field>
                            <swe:field name="ALERTS_FilteredDoseRate">
                                <!--Note:  Quantity is used for integer types instead of
Count, since in CCSI sensor descriptions, an integer may be associated with a unit of
measure-->
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:FilteredDoseRate">
                                    <swe:uom code="cGy/hr"/>
                                    <swe:constraint>
                                        <swe:AllowedValues>
                                            <swe:interval>0.0 999.9</swe:interval>
                                        </swe:AllowedValues>
                                    </swe:constraint>
                                </swe:Quantity>
                            </swe:field>
                            <swe:field name="ALERTS MissionDose">
                                <!--Note:  Quantity is used for integer types instead of
Count, since in CCSI sensor descriptions, an integer may be associated with a unit of
measure-->
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:MissionDose">
                                    <swe:uom code="uGy"/>
                                    <swe:constraint>
                                        <swe:AllowedValues>
                                            <swe:interval>0.0 999.9</swe:interval>
                                        </swe:AllowedValues>
                                    </swe:constraint>
                                </swe:Quantity>
                            </swe:field>
                        </swe:DataRecord>
                    </output>
                    <output name="CONFIG">
                        <swe:DataRecord gml:id="CONFIG_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::CONFIG"/>
                    </output>
                    <output name="HRTBT">
                        <swe:DataRecord gml:id="HRTBT_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::HRTBT">
                            <swe:field name="Mode">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                            </swe:field>
                            <swe:field name="DateTimeGroup">
                                <swe:Time definition="urn:ogc:def:phenomenon:JPEO-
CBD::Time" referenceTime="1970-01-01T00:00:00Z">
                                    <swe:uom code="s"/>
                                </swe:Time>
                            </swe:field>
                        </swe:DataRecord>
                    </output>
                    <output name="IDENT">
```



```
                        <swe:DataRecord gml:id="IDENT_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::IDENT"/>
                    </output>
                    <output name="MAINT">
                        <swe:DataRecord gml:id="MAINT_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT"/>
                    </output>
                    <output name="READGS">
                        <swe:DataRecord gml:id="READGS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS">
                            <swe:field name="Mode">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                            </swe:field>
                            <swe:field name="READGS_Sensor">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                            </swe:field>
                            <swe:field name="READGS_ReadingID">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                                    <swe:codeSpace
xlink:href="urn:ogc:def:property:JPEO-CBD::readingIdentificationType"/>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="READGS_Detect">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                            </swe:field>
                            <swe:field name="READGS_Level">
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                            </swe:field>
                            <swe:field name="READGS LevelConfidenceInterval">
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                            </swe:field>
                            <swe:field name="READGS_Id">
                                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                            </swe:field>
                            <swe:field name="READGS DetailsAvailable">
                                <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                            </swe:field>
                            <swe:field name="READGS FilteredDoseRate">
                                <!--Note:  Quantity is used for integer types instead of
Count, since in CCSI sensor descriptions, an integer may be associated with a unit of
measure-->
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                                    <swe:uom code="cGy/hr"/>
                                    <swe:constraint>
                                        <swe:AllowedValues>
                                            <swe:interval>0.0 999.9</swe:interval>
                                        </swe:AllowedValues>
                                    </swe:constraint>
                                </swe:Quantity>
                            </swe:field>
                            <swe:field name="READGS_MissionDose">
                                <!--Note:  Quantity is used for integer types instead of
Count, since in CCSI sensor descriptions, an integer may be associated with a unit of
measure-->
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                                    <swe:uom code="uGy"/>
                                    <swe:constraint>
                                        <swe:AllowedValues>
                                            <swe:interval>0.0 999.9</swe:interval>
                                        </swe:AllowedValues>
                                    </swe:constraint>
                                </swe:Quantity>
```



```
                            </swe:field>
                        </swe:DataRecord>
                    </output>
                    <output name="STATUS">
                        <swe:DataRecord gml:id="STATUS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS"/>
                    </output>
                </OutputList>
            </outputs>
            <components/>
            <positions>
                <PositionList>
                    <position name="sensorPosition">
                        <swe:Position localFrame="#SENSOR_FRAME"
referenceFrame="urn:ogc:def:crs:EPSG::4326">
                            <swe:location>
                                <swe:Vector
definition="urn:ogc:def:vector:OGC:location">
                                    <swe:coordinate name="latitude">
                                        <swe:Quantity axisID="Y"
definition="urn:ogc:def:phenomenon:latitude">
                                            <swe:uom code="deg"/>
                                            <swe:value>29.9850127</swe:value>
                                        </swe:Quantity>
                                    </swe:coordinate>
                                    <swe:coordinate name="longitude">
                                        <swe:Quantity axisID="X"
definition="urn:ogc:def:phenomenon:longitude">
                                            <swe:uom code="deg"/>
                                            <swe:value>-90.2583548</swe:value>
                                        </swe:Quantity>
                                    </swe:coordinate>
                                    <swe:coordinate name="altitude">
                                        <swe:Quantity axisID="Z"
definition="urn:ogc:def:phenomenon:altitude">
                                            <swe:uom code="m"/>
                                            <swe:value>0.0</swe:value>
                                        </swe:Quantity>
                                    </swe:coordinate>
                                </swe:Vector>
                            </swe:location>
                            <swe:orientation>
                                <swe:Vector
definition="urn:ogc:def:vector:OGC:orientation">
                                    <swe:coordinate name="trueHeading">
                                        <swe:Quantity
definition="urn:ogc:def:phenomenon:angleToNorth">
                                            <swe:uom code="deg"/>
                                            <swe:value>0.0</swe:value>
                                        </swe:Quantity>
                                    </swe:coordinate>
                                </swe:Vector>
                            </swe:orientation>
                        </swe:Position>
                    </position>
                </PositionList>
            </positions>
            <connections/>
        </System>
    </SensorDescription>
    <ObservationTemplate>
        <Observation gml:id="RadDetector_1_119837"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
```

```
            <gml:description>CCSI Sensor Observation Instance</gml:description>
            <gml:name>CCSI Sensor Observation Instance (RadDetector_1_119837)</gml:name>
            <samplingTime>
                <gml:TimeInstant>
                    <gml:timePosition>2010-05-05T13:48:18Z</gml:timePosition>
                </gml:TimeInstant>
            </samplingTime>
            <procedure xlink:href="urn:ogc:def:procedure:JPEO-CBD::RadDetector_1"/>
            <observedProperty>
                <swe:CompositePhenomenon gml:id="READGS_RadDetector_1_119837"
dimension="10">
                    <gml:name>CCSI READGS Phenomenon</gml:name>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose"/>
                </swe:CompositePhenomenon>
            </observedProperty>
            <featureOfInterest>
                <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
                    <gml:name>CCSI Sensor</gml:name>
                    <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
                    <sa:position>
                        <gml:Point gml:id="sensor Point">
                            <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">18.556825 -
72.297935</gml:pos>
                        </gml:Point>
                    </sa:position>
                </sa:SamplingPoint>
            </featureOfInterest>
            <result>
                <swe:DataRecord gml:id="CCSI_READGS_DATA_RECORD">
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="READGS_Sensor">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                    </swe:field>
                    <swe:field name="READGS ReadingID">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                            <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="READGS_Detect">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                    </swe:field>
                    <swe:field name="READGS_Level">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
```



```
                    </swe:field>
                    <swe:field name="READGS_Id">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                    </swe:field>
                    <swe:field name="READGS_DetailsAvailable">
                        <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                    </swe:field>
                    <swe:field name="Sensor Unique Data">
                        <swe:DataRecord definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:SUD">
                            <swe:field name="FilteredDoseRate">
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                                    <swe:uom code="cGy/hr"/>
                                </swe:Quantity>
                            </swe:field>
                            <swe:field name="MissionDose">
                                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                                    <swe:uom code="cGy"/>
                                </swe:Quantity>
                            </swe:field>
                        </swe:DataRecord>
                    </swe:field>
                </swe:DataRecord>
            </result>
        </Observation>
    </ObservationTemplate>
</RegisterSensor>
```

10.3.2 Example RegisterSensor Response

```
<?xml version="1.0" encoding="UTF-8"?>
<RegisterSensorResponse xmlns="http://www.opengis.net/sos/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosAll.xsd">
  <AssignedSensorId>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</AssignedSensorId>
</RegisterSensorResponse>
```

10.3.3 Example InsertObservation Request

```
<?xml version="1.0" encoding="UTF-8"?>
<InsertObservation xmlns="http://www.opengis.net/sos/1.0"
xmlns:om="http://www.opengis.net/om/1.0" xmlns:swe="http://www.opengis.net/swe/1.0.1"
xmlns:gml="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosAll.xsd" service="SOS" version="1.0.0">
    <AssignedSensorId>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</AssignedSensorId>
    <Observation gml:id="RadDetector 1 119837"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
        <gml:description>CCSI Sensor Observation Instance</gml:description>
        <gml:name>CCSI Sensor Observation Instance (RadDetector_1_119837)</gml:name>
        <samplingTime>
            <gml:TimeInstant>
```



```
                <gml:timePosition>2010-05-05T13:48:18Z</gml:timePosition>
            </gml:TimeInstant>
        </samplingTime>
        <procedure xlink:href="urn:ogc:def:procedure:JPEO-CBD::RadDetector_1"/>
        <observedProperty>
            <swe:CompositePhenomenon gml:id="READGS_RadDetector_1_119837" dimension="10">
                <gml:name>CCSI READGS Phenomenon</gml:name>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate"/>
                <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose"/>
            </swe:CompositePhenomenon>
        </observedProperty>
        <featureOfInterest>
            <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
                <gml:name>CCSI Sensor</gml:name>
                <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
                <sa:position>
                    <gml:Point gml:id="sensor Point">
                        <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">18.556825 -
72.297935</gml:pos>
                    </gml:Point>
                </sa:position>
            </sa:SamplingPoint>
        </featureOfInterest>
        <result>
            <swe:DataRecord gml:id="CCSI_READGS_DATA_RECORD">
                <swe:field name="Mode">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                        <swe:value>Normal</swe:value>
                    </swe:Category>
                </swe:field>
                <swe:field name="READGS_Sensor">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor">
                        <swe:value>SC001</swe:value>
                    </swe:Category>
                </swe:field>
                <swe:field name="READGS_ReadingID">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                        <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                        <swe:value>R000239709</swe:value>
                    </swe:Category>
                </swe:field>
                <swe:field name="READGS_Detect">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect">
                        <swe:value>Gamma</swe:value>
                    </swe:Category>
                </swe:field>
                <swe:field name="READGS_Level">
                    <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level">
```



```
                        <swe:value>20.09765625</swe:value>
                    </swe:Quantity>
                </swe:field>
                <swe:field name="READGS_Id">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                </swe:field>
                <swe:field name="READGS_DetailsAvailable">
                    <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable">
                        <swe:value>true</swe:value>
                    </swe:Boolean>
                </swe:field>
                <swe:field name="Sensor Unique Data">
                    <swe:DataRecord definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:SUD">
                        <swe:field name="FilteredDoseRate">
                            <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                                <swe:uom code="cGy/hr"/>
                                <swe:value>1.57</swe:value>
                            </swe:Quantity>
                        </swe:field>
                        <swe:field name="MissionDose">
                            <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                                <swe:uom code="cGy"/>
                                <swe:value>20.098</swe:value>
                            </swe:Quantity>
                        </swe:field>
                    </swe:DataRecord>
                </swe:field>
            </swe:DataRecord>
        </result>
    </Observation>
</InsertObservation>
```

10.3.4 Example InsertObservation Response

```
<?xml version="1.0" encoding="UTF-8"?>
<InsertObservationResponse xmlns="http://www.opengis.net/sos/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosAll.xsd">
  <AssignedObservationId>RadDetector_1_119837</AssignedObservationId>
</InsertObservationResponse>
```

10.3.5 Similarities between the SOS-T and CCSI Web Services

The SOS-T is essentially a dynamic sensor and sensor data registry. Sensors advertise their existence through *RegisterSensor* requests and their data through *InsertObservation* requests, and clients can discover information about available sensors and data through the SOS-T Capabilities document. This dynamic registration approach has security implications, particularly when sensors are used in hostile areas or when the sensors are used in critical military or government applications in non-hostile areas. An adversary could, for instance, gain access to the SOS-T and register false sensors and data that might mislead SOS client users. An adversary could also perform a Denial of Service attack on the SOS-T by setting up malicious software that registered a large number of false sensors and sensor data to the SOS-T, which might eventually prevent the SOS-T from functioning properly. The SOS-T approach does not currently specify the use of sensor verification or security functionality, but those functions could be added by the

SOS-T service implementer. For instance, the SOS-T implementer could configure the service to only allow *RegisterSensor* and *InsertObservation* requests for a particular sensor ID or set of sensor IDs or could require that username and password information be provided when issuing those requests; these requests could also be accepted only from particular IP addresses. This would eliminate or reduce attempts by an enemy to register false sensors and data with the SOS. Security of SWE web services as well as OGC web services in general is one area that the OGC has been examining recently both in the OWS-6 testbed and minimally in the OWS-7 testbed. The SOS-T operations are one place where security needs to be considered further. The OWS-6 Secure Sensor Web Engineering Report [08-176r1] examined the transactional operations as well as other SWE services and operations from a security standpoint and noted several vulnerabilities that need to be addressed either through the specifications themselves or through service implementations. The SOS-T approach is somewhat analogous to the use case that JPEO-CBD is exploring in developing the concept of a Network Security and Discovery Service (NSDS) web service that augments the CCSI standards. JPEO-CBD is designing the NSDS to provide the Universal Description, Discovery, and Integration (UDDI)/registry and security functions for CCSI, and the NSDS will also provide and verify client and user security credentials. The NSDS approach also seems to provide similar functionality to an OGC CS/W, which is used with SWE to provide a catalog of sensor services and sensor metadata. The following table illustrates the functional similarities that exist between the NSDS approach and the SOS-T approach and is based on initial NSDS designs that may be subject to change.

Table 7- NSDS and SOS-T Functional Comparison

Step	NSDS	SOS-T
1	The sensor advertises its existence to a registry service.	The sensor issues a *RegisterSensor* request to an SOS-T.
2	The registry service verifies the sensor using the Information Assurance functions for the sensor. It also contains or acquires the sensor metadata.	The sensor metadata is included or linked to in the *RegisterSensor* request, so the SOS-T automatically acquires sensor metadata from the sensor or from an address on the network. The SOS-T implementer can decide what sensors can register with the SOS-T using a variety of security/authentication mechanisms.
3	The service sends permission to the sensor to operate on the network.	If the sensor successfully registers with the SOS-T, the SOS-T returns a *RegisterSensor* response that contains an *AssignedSensorId* to be used when inserting data.
4	The service aggregates a list of available sensors by capability and location for the host's use, and provides permission for the host to	A client application can request the SOS-T Capabilities document, which provides a list of sensors that have been registered with the SOS-T as well as information on

	operate on the sensor network	geospatial, temporal, and observed property information for each sensor.

10.4 The Soldier/First Responder as a Sensor

In military and civilian CBRN response situations, the response to an incident involves a combination of soldiers or first responders in addition sensors. Sensors produce raw data and measurements that help to quantify characteristics of the current situation. Soldiers and first responders using those sensors or performing specific tasks in the vicinity of a CBRN incident can help to augment raw sensor measurements with additional information. In many cases, this information is qualitative rather than quantitative, and SWE supports qualitative observations through the use of SWE Common *Text* elements that allow free text values. The SWE Common *Category* element would also be particularly useful in supporting user observations describing an incident, since the *Category value* represents a particular pre-defined value from a set of pre-defined values. For instance, an observation could include an IncidentSeverity *Category* field, and the field would be populated with the category that best describes the severity of the incident from a set of pre-defined IncidentSeverity values (e.g. Moderate, Severe, etc.). The fictitious example below illustrates an O & M document that describes a user observation.

```
<?xml version="1.0" encoding="UTF-8"?>
<Observation xmlns="http://www.opengis.net/om/1.0" gml:id="OBS001"
xsi:schemaLocation="http://www.opengis.net/om/1.0
http://schemas.opengis.net/om/1.0.0/om.xsd"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <gml:description>Soldier Observation</gml:description>
    <gml:name>Observation 1</gml:name>
    <samplingTime>
        <gml:TimeInstant>
            <gml:timePosition>2010-04-13T10:34:00.564Z</gml:timePosition>
        </gml:TimeInstant>
    </samplingTime>
    <procedure xlink:href="urn:ogc:def:soldier:USArmy::"/>
    <observedProperty xlink:href="urn:ogc:def:property:USArmy::userObservation"/>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="samplingPoint1"
xmlns:sa="http://www.opengis.net/sampling/1.0"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd">
            <gml:name>Soldier Location</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="point1">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">42.500999 -
71.122829</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataRecord definition="urn:ogc:def:property:USArmy::userObservation">
            <swe:field name="IncidentSeverity">
                <swe:Category definition="urn:ogc:def:property:USArmy::incidentSeverity">
```



```
                    <swe:value>Severe</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="IncidentNotes">
                <swe:Text definition="urn:ogc:def:property:USArmy::incidentNotes">
                    <swe:value>A toxic substance was detected by a nearby sensor.  There
appears to be a yellowish smoke cloud.  I'm in the process of collecting more readings in
the area and notifying command.</swe:value>
                </swe:Text>
            </swe:field>
        </swe:DataRecord>
    </result>
</Observation>
```

When utilizing an SOS-T, soldiers and first responders could register themselves with the SOS-T and insert observations about the particular incident to which they are responding (through the use of supporting software), and the SOS-T can serve as a searchable catalog of all user information related to a particular incident. This SOS-T could be the same SOS-T used to register CCSI sensors and data.

10.5 Handling Asynchronous Sensor Data

There are currently several approaches within SWE and OGC for handling asynchronous sensor data. The most mature approach is the SAS, which is currently an OGC Best Practices document. Other, less mature but perhaps more favorable approaches, include the Sensor Event Service (SES) [08-133], which makes use of the WS-Notification family of specifications from OASIS, as well as utilizing WS-Notification with the SOAP/WSDL based future versions of the SWE web service standards. The OWS-6 CCSI-SWE integration work explored the use of the SAS for CCSI ALERT channel messages, but the SAS approach appears to be losing favor within the OGC and SWE communities, since an alert is really just a specialized form of a sensor observation. A sensor alert is merely a sensor observation that meets a specific set of conditions that are either defined by the sensor data producer or the sensor data consumer. Consequently, there have been discussions as to whether asynchronous notifications and filtering should be added to the SOS specification and other SWE specifications. Not knowing what the recommended approach will end up being, this report provides some illustrative examples using the envisioned approaches for eventing/asynchronous messaging. For more information on asynchronous notifications and eventing with respect to OGC services, refer to the OWS-6 Event Architecture ER [09-032] and the OWS-7 Event Architecture ER [10-060].

10.5.1 Using an SES

The SES acts as an intermediary between a sensor data producer and a sensor data consumer, providing notifications to consumers based on consumer-defined filter criteria. In the CCSI case, the SES would sit between a CCSI sensor and a client application or a controlling application connected to a CCSI sensor and a client application. The controlling application publishes CCSI sensor data in SWE formats to the SES. The SES then filters the incoming data and publishes it to endpoints defined by the client when subscribing to a particular topic, a logical grouping of notifications defined by the SES provider. Through WS-Notification, the SES makes use of *TopicNamespace* and

TopicSet definitions. Figure 8 below shows an overview of the topics pre-defined in the SES specification. Sensor notifications are posted to the *Measurements* topic, notifications regarding the management of sensors are posted to the *SensorManagement* topic, and notifications regarding information about expiration times of sensor registrations are posted to the *ExpirationInformation* topic.

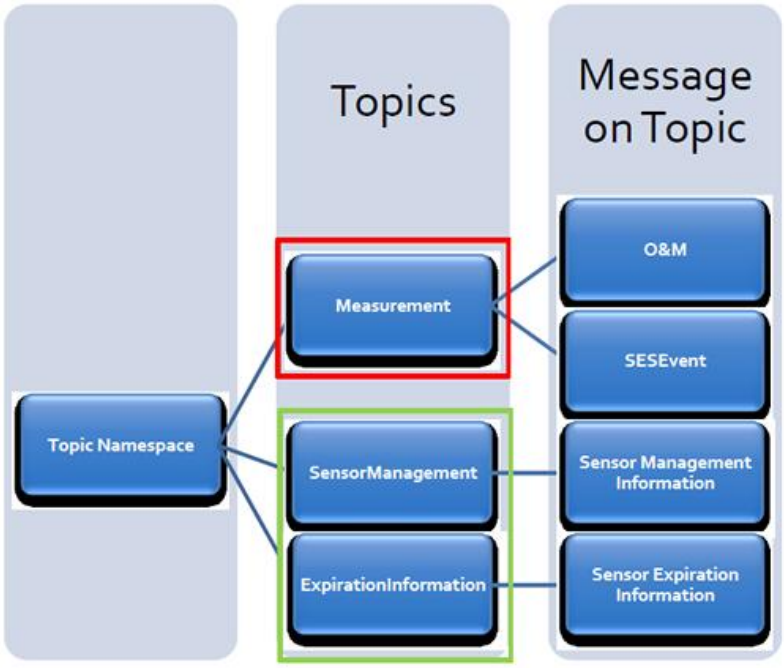

Figure 8 - SES Topic Namespaces (re-presented from [08-133])

The basic SES *TopicNamespace* is defined as follows:

```
<TopicNamespace targetNamespace="http://www.opengis.net/ses/0.0"
xmlns="http://docs.oasis-open.org/wsn/t-1" xmlns:om="http://www.opengis.net/om/1.0"
xmlns:omx="http://www.opengis.net/omx/1.0" xmlns:ses="http://www.opengis.net/ses/0.0">
    <Topic name="Measurements" messageTypes="om:Observation omx:Measurement
omx:CategoryObservation omx:CountObservation omx:TruthObservation omx:GeometryObservation
omx:TemporalObservation omx:ComplexObservation ses:SESEvent"/>
    <Topic name="SensorManagement" messageTypes="ses:SensorManagementInformation
ses:SensorExpirationInformation">
        <Topic name="Registered" messageTypes="ses:SensorManagementInformation"/>
        <Topic name="Deleted" messageTypes="ses:SensorManagementInformation"/>
        <Topic name="Paused" messageTypes="ses:SensorManagementInformation"/>
        <Topic name="Resumed" messageTypes="ses:SensorManagementInformation"/>
    </Topic>
    <Topic name="ExpirationInformation" messageTypes="ses:SensorExpirationInformation"/>
</TopicNamespace>
```

Using the SES, sensor data can be filtered in two ways. The simplest approach is based upon subscribing to one or more topics defined in the *TopicSet* of the SES. In this approach, the SES provider has already grouped events into predefined topics, and the subscriber does not care to filter events beyond the general grouping provided by the service. A more complicated approach allows a subscriber to define filter criteria for

event notifications. The SES supports multiple levels of filtering (inserted verbatim from the SES specification):

- Level 1 (mandatory): this filter uses XPath to define the filter criteria that shall be applied on each incoming notification separately.
- Level 2 (optional): this filter uses a combination of logical, spatial, temporal, arithmetic and comparison operators as well as functions. Unit of measure conversion may also be performed. Like level 1 filter which operate on single notifications, level 2 filter operate on each incoming event separately.
- Level 3 (optional): in addition to the filter capabilities of level 2, level 3 filters can handle queries on data streams. Thus, they may operate not only on one incoming event at a time but rather on the whole cloud of events that arrive at the broker and in this way make statistical queries (like average or variance of a set of events) possible.

Using Level 1 filtering, a client could subscribe for messages from a single CBRN sensor, using the following XPath expression:

```
//om:Observation/om:procedure/@xlink:href='urn:ogc:def:procedure:
JPEO-CBD::CAD_2'
```

To subscribe for messages containing observations for observed property urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Tamper, the following expression could be used:

```
//om:Observation//om:observedProperty//@xlink:href='
urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Tamper '
```

Another XPath example could be to combine the previous two expressions with a simple 'and' operator. This would result in subscribing for observations containing both the given sensor as well as the property TAMPER:

```
//om:Observation/om:procedure/@xlink:href='CAD_2' and
//om:Observation//om:observedProperty//@xlink:href='
urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Tamper '
```

The SES specification provides a description of second level filtering below:

Second level filtering involves filtering with value-thresholds, spatial operations, and time-based filters. For example,

```
<?xml version="1.0" encoding="UTF-8"?>
<Filter>
    <And>
        <PropertyIsGreaterThan>
            <PropertyName> urn:ogc:def:property:OGC:1.0:temperature </PropertyName>
            <Literal>
                <gml:Quantity uom="Cel">30</gml:Quantity>
            </Literal>
        </PropertyIsGreaterThan>
        <Contains>
```



```
            <PropertyName>geometry</PropertyName>
            <gml:Envelope srsName="urn:ogc:def:crs:EPSG:6.17:31466">
                <gml:lowerCorner>2590000 5680000</gml:lowerCorner>
                <gml:upperCorner>2590800 5680200</gml:upperCorner>
            </gml:Envelope>
        </Contains>
        <After>
            <PropertyName>startTime</PropertyName>
            <gml:TimeInstant>
                <gml:timePosition>2008-05-25T12:25:00Z</gml:timePosition>
            </gml:TimeInstant>
        </After>
    </And>
</Filter>
```

The highest level of filtering functionality at SES is level three. Here, the Event Pattern Markup Language (EML) [08-132] is used to define filters acting on (the combination of) event streams rather than on single events (as in level two).

Simple examples for stream filter use cases are:

- Counting of events that occurred inside a given time window
- Calculating the average of a measured phenomenon over a certain time period
- Being informed only when a threshold is crossed
- Counting of new events with certain property which exceeds the given threshold inside a given time window
- Averaging of a property from the last five events from sensor XYZ is greater than a given value

10.5.2 Using an SOS with WS-Notification

The *TopicNamespace* and *TopicSet* definitions for an SOS that makes use of SOAP/WSDL and WS-Notification and that provides access to CCSI sensors and data would extend the base *TopicNamespace* that will be defined in the SOS 2.0 specification, which itself should be an extension of the SWE Service Model 2.0 *TopicNamespace*. An SOS could provide the following *TopicNamespace* and *TopicSet* definitions respectively for defining notification channels from CBRN sensors. Note that the main "NewObservationAvailable" *Topic* element is intended to represent a general topic that could be defined by the SOS 2.0 specification.

```
<wstop:TopicNamespace xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:om="http://www.opengis.net/om/1.0.0" xmlns:om2="http://www.opengis.net/om/2.0.0"
name="SOS-Topic-Namespace" targetNamespace="http://www.opengis.net/sos/2.0" final="true">
    <wstop:Topic name="NewObservationAvailable">
        <wstop:Topic name="CAD_1">
            <wstop:Topic name="READGS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="ALERTS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="HRTBT" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="MAINT" messageTypes="om:Observation om2:Observation"/>
        </wstop:Topic>
        <wstop:Topic name="CAD_2">
            <wstop:Topic name="READGS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="ALERTS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="HRTBT" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="MAINT" messageTypes="om:Observation om2:Observation"/>
        </wstop:Topic>
        <wstop:Topic name="CAD_3">
            <wstop:Topic name="READGS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="ALERTS" messageTypes="om:Observation om2:Observation"/>
```



```
            <wstop:Topic name="HRTBT" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="MAINT" messageTypes="om:Observation om2:Observation"/>
        </wstop:Topic>
…
        <wstop:Topic name="ALL">
            <wstop:Topic name="READGS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="ALERTS" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="HRTBT" messageTypes="om:Observation om2:Observation"/>
            <wstop:Topic name="MAINT" messageTypes="om:Observation om2:Observation"/>
        </wstop:Topic>
    </wstop:Topic>
...
</wstop:TopicNamespace>
```

In the example above there is a parent topic defined for each CCSI sensor advertised by the SOS along with child topics for each CCSI channel that the sensor supports. There is also a parent topic defined for all sensors, and a consumer subscribing to that topic should expect to receive messages from all sensors advertised by that SOS. Each topic provides a *messageTypes* attribute that describes the type of messages consumers can expect to receive through a WS-Notification *Notify* message. In this example, notifications can be in either O & M 1.0.0 or O & M 2.0.0 formats. A CCSI SOS could advertise the following *TopicSet*, which, in this case, lists a subset of the topics defined in the associated *TopicNamespace*. In the SOS 2.0 specification, the supported *TopicSet* will most likely be defined in the SOS Capabilities document.

```
<?xml version="1.0" encoding="UTF-8"?>
<wstop:TopicSet xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:tns="http://www.opengis.net/sos/2.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://docs.oasis-open.org/wsn/t-1
http://docs.oasis-open.org/wsn/t-1.xsd"
    <tns:CAD_1 wstop:topic="true">
        <tns:READGS wstop:topic="true"/>
    </tns:CAD_1>
    <tns:CAD 2>
        <tns:READGS wstop:topic="true"/>
    </tns:CAD_2>
</wstop:TopicSet>
```

The consumer uses the service's *TopicSet* and subscribes to one or more topics. When a message occurs that matches a particular topic, the producer sends a SOAP encoded *Notify* message to the consumer. The example below illustrates a *Notify* message. Note that in the SWE case, the *wsnt:Message* field would contain the O & M or other SWE format document being sent to the consumer.

```
<s:Envelope …>
    <s:Header>
        <wsa:Action>
http://docs.oasis-open.org/wsn/bw-2/NotificationConsumer/Notify
</wsa:Action>
...
</s:Header>
    <s:Body>
        <wsnt:Notify>
            <wsnt:NotificationMessage>
                <wsnt:SubscriptionReference>
                    <wsa:Address>
http://www.example.org/SubscriptionManager
</wsa:Address>
                </wsnt:SubscriptionReference>
                <wsnt:Topic Dialect="http://docs.oasis-open.org/wsn/t-
1/TopicExpression/Simple">
tns:CAD_1
```



```
</wsnt:Topic>
                <wsnt:ProducerReference>
                    <wsa:Address>
http://www.example.org/NotificationProducer
</wsa:Address>
                </wsnt:ProducerReference>
                <wsnt:Message>
                    <om:Observation …>
                </wsnt:Message>
            </wsnt:NotificationMessage>
        </wsnt:Notify>
    </s:Body>
</s:Envelope>
```

10.6 Usage Scenarios

JPEO-CBD initially envisioned four usage scenarios when CCSI was defined: trucks, portable sensors, unattended ground sensors (UGS) and unmanned ground vehicles (UGV), and command posts. The diagram below illustrates these usage scenarios.

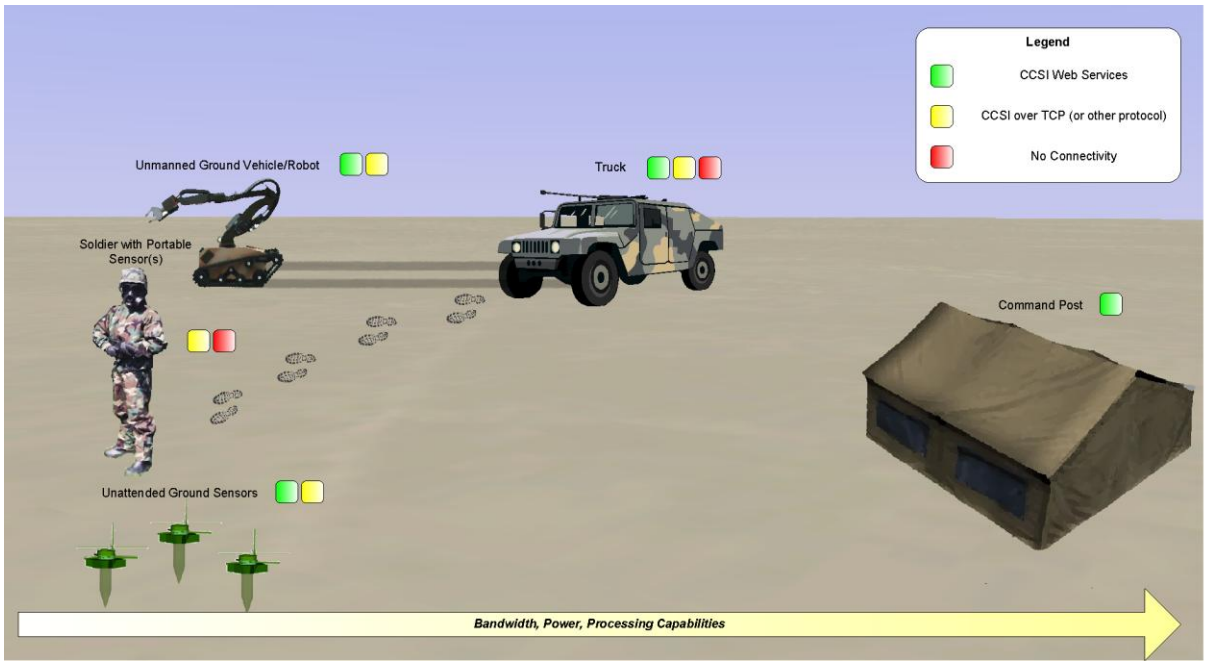

Figure 9 - CCSI Usage Scenarios

In the truck scenario, the controlling application and the CCSI sensors all reside on the same vehicle and are often hard-wired using Ethernet. The controlling application communicates with one or more CCSI sensors over Ethernet using the CCSI messaging formats described above and defined in Volume III of the CCSI specifications. The truck may or may not have network connectivity back to higher headquarters. In the case that network connectivity is not available, the Chem/Bio officer would most likely communicate sensor readings and events back to higher headquarters (via radio or other means) through a NATO standard ATP-45 Nuclear, Biological, Chemical (NBC) voice report message. In the case that network connectivity is available, the truck and hard-wired sensors could be treated as a system of sensors and could incorporate the use of web services and CCSI encodings described above. In the truck scenario where network

connectivity is available, the NSDS and other web services could be installed on the truck or at higher headquarters.

In addition to the use of sensors hard-wired on the truck, the truck also serves as a platform for transporting soldiers and portable sensors. The portable sensors are battery powered and communicate via wireless protocols, and the sensors report visually and audibly to a local operator as well as electronically to a controlling application through CCSI standard channel messages. JPEO-CBD envisions that UGS and UGV systems would communicate using wireless and would most likely include the use of web services but could also rely on basic CCSI communications with a controlling application.

The command post has access to greater bandwidth and more processing power and resources and many sensors and would implement the NSDS services and other web services so that sensor data can be easily shared with other command posts.

In general, SWE web services and encodings can be substituted or added in any of the CCSI usage scenarios that support the use of web services and CCSI standard encodings. Consideration should be given to bandwidth and processing power when deciding where and when to apply SWE web services and encodings in addition to or in place of CCSI. Based upon cursory examination, CCSI channel messages are more compact than O & M documents that utilize SWE Common, and as such will probably be delivered more efficiently over network connections. This is because SWE encodings and the underlying XML elements have been generalized to support describing multiple sensors in multiple domains. SWE elements are less specific to a particular type of sensor or sensor data, and therefore, require additional information such as *definition* attributes to be fully understood by applications making use of SWE encodings. SWE encodings also import many elements and attributes from other OGC standards and standards developed by other organizations, and therefore include many namespace declarations and extensive use of namespace prefixes, which can increase the size of SWE documents. Geographic location is also almost always included in SWE encodings by default. In contrast, CCSI messages are fairly specific to CBRN sensors and data and reference few external standards, which help to minimize XML document size. For example, the size of a CCSI READGS channel message for a radiation detector and the same channel message defined in O & M can be analyzed to compare potential performance impacts. The XML documents below illustrate the same data described in CCSI and O & M with SWE Common formats.

```
<?xml version="1.0" encoding="utf-8"?>
<Hdr sid="RadDetector_2" msn="96355" mod="N" dtg="1272899074" chn="r" len="374">
    <CCSIDoc>
        <Msg>
            <ReadingsChn>
                <ReadingReport Time="20100503100434" Sensor="SC001"
ReadingID="R000165198">
                    <Data DetailsAvailable="true" Detect="Gamma" Level="16491.677734375">
                        <SUD Name="FilteredDoseRate" Value="141.87" Units="cGy/hr"
Type="Float"/>
                        <SUD Name="MissionDose" Value="16491.678" Units="cGy"
Type="Float"/>
                    </Data>
                </ReadingReport>
            </ReadingsChn>
        </Msg>
    </CCSIDoc>
```


</Hdr>

Figure 10 - CCSI READGS Channel Message (603 Bytes)

```
<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector 2 96359"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <gml:description>CCSI Sensor Observation Instance</gml:description>
    <gml:name>CCSI Sensor Observation Instance (RadDetector_2_96359)</gml:name>
    <samplingTime>
        <gml:TimeInstant>
            <gml:timePosition>2010-05-03T10:04:49Z</gml:timePosition>
        </gml:TimeInstant>
    </samplingTime>
    <procedure xlink:href="RadDetector_2"/>
    <observedProperty>
        <swe:CompositePhenomenon gml:id="READGS RadDetector_2_96359" dimension="10">
            <gml:name>CCSI READGS Phenomenon</gml:name>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Sensor"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Detect"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose"/>
        </swe:CompositePhenomenon>
    </observedProperty>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>CCSI Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor_Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">18.556491 -
72.297463</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataRecord gml:id="CCSI_READGS_DATA_RECORD">
            <swe:field name="Mode">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                    <swe:value>Normal</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS_Sensor">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor">
```



```
                    <swe:value>SC001</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS ReadingID">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                    <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                    <swe:value>R000165204</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS_Detect">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect">
                    <swe:value>Gamma</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS_Level">
                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level">
                    <swe:value>16492.2734375</swe:value>
                </swe:Quantity>
            </swe:field>
            <swe:field name="READGS Id">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
            </swe:field>
            <swe:field name="READGS_DetailsAvailable">
                <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable">
                    <swe:value>true</swe:value>
                </swe:Boolean>
            </swe:field>
            <swe:field name="Sensor Unique Data">
                <swe:DataRecord definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:SUD">
                    <swe:field name="FilteredDoseRate">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                            <swe:uom code="cGy/hr"/>
                            <swe:value>141.87</swe:value>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="MissionDose">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                            <swe:uom code="cGy"/>
                            <swe:value>16492.273</swe:value>
                        </swe:Quantity>
                    </swe:field>
                </swe:DataRecord>
            </swe:field>
        </swe:DataRecord>
    </result>
</Observation>
```

Figure 11 - O & M Message (5,112 Bytes)

The CCSI channel message is 603 bytes (Figure 10), while the O & M message that represents the same information is 5,112 bytes (Figure 11); in this case, the O & M document is approximately 8.5 times larger than the corresponding CCSI document. Given this example, if CCSI sensors are being utilized on extremely low bandwidth networks, then it may not make sense to apply SWE on those low bandwidth networks. In the CCSI CONOPs this could mean the network connection between a soldier and a truck. It does, however, make sense to apply SWE on higher-bandwidth networks where CBRN sensor data is to be shared with other organizations or integrated with other sensor data. In the CCSI CONOPs case, this could mean applying SWE for sharing sensor data between command posts or between trucks and command posts. In the higher bandwidth

case, the value of increased interoperability will often outweigh any potential performance impacts.

Based on the simple comparison above, CCSI formats appear to be more compact than SWE formats in cases where a single observation is delivered by a sensor or web service but may be less compact than SWE formats when a time series of channel data is delivered by a sensor or web service. This case could arise when a CCSI sensor stores the last X observations in internal memory in cases where the sensor is operating without network connectivity or with intermittent network connectivity. A CCSI SOS could also provide a time series of observations in a response to a *GetObservation* request as described previously in this document. Using SWE Common encoding with O & M allows for the use of *DataArray* elements that significantly reduce the amount of redundant XML elements and metadata in a single document. For instance, an O & M document using a SWE Common *DataArray* with five readings from a radiation detector (5,600 bytes) is not much larger than an O & M document that describes one reading (5,112 bytes). Using a SWE Common *DataArray*, as the number of readings increases, the size of the O & M document increases minimally, marginalizing the impact of the additional overhead inherent in O & M and SWE Common metadata. Using the radiation detector example, a CCSI READGS channel message with 20 readings (9,055 bytes) is larger than an O & M document with 20 readings (7,130 bytes). More testing needs to be performed to determine size differences between SWE and CCSI formats for describing the same data, and the effects of message size on performance should be considered when determining where and when to apply CCSI and SWE within sensor architectures.

11 Using CCSI with SWE in the Future

11.1 Need for Profiles

The SWE standards were developed to be extremely flexible in order to support a wide variety of sensors. This flexibility has its benefits but also creates challenges. One way of eliminating some of the challenges created by the flexibility of the SWE standards is through the use of profiles. Profiles restrict the general nature of the SWE standards to define specific requirements for SWE services and encodings. The CCSI standards have restricted the past variety of vendor and program-specific CBRN messages into a standard set of messages that can be used across CBRN sensor operations and serve as the basis for the profiles defined in this document. The XSLT transforms originally developed in OWS-6 and expanded in this document provide output that complies with the profiles defined below, and these profiles can serve as the path forward for harmonizing CCSI and SWE, since they define the SWE representation of CCSI.

11.2 Common Conventions Used across Profiles

11.2.1 Naming and URN Conventions

The OWS-6 ER defined several naming and URN conventions for describing CCSI metadata and data within SWE services and encodings. URNs are used fairly frequently across SWE 1.0 implementations for defining and identifying resources like sensors and

observed properties. Note that there is on-going discussion within the OGC as to the future use of URNs versus URIs or URLs, so the 2.0 versions of the SWE standards may move away from the use of URNs.

11.2.1.1 Sensor IDs

All sensor IDs provided by CCSI SWE services should adhere to the following pattern in accordance with the OGC's URN naming convention best practices:

urn:ogc:def:procedure:JPEO-CBD::[CCSI Sensor UUID]

NOTE CCSI Sensor UUID is the sensor's unique identifier (i.e. sid) obtained through communications with the sensor.

An example URN for a Chemical Agent Detector (CAD) sensor with UUID CAD_123 is:

urn:ogc:def:procedure:JPEO-CBD::CAD_123

11.2.1.2 Observed Property URNs

Observed property URNs that describe CCSI channel data should follow a similar pattern:

urn:ogc:def:phenomenon:JPEO-CBD::[CCSI Channel Abbreviation]

urn:ogc:def:phenomenon:JPEO-CBD::[CCSI Channel Abbreviation]:[CCSI Channel Metadata]

NOTE CCSI Channel Abbreviation is the uppercase abbreviation for the channel (i.e. READGS), found in Table 2. CCSI Channel Metadata is the name of a metadata item within a particular CCSI channel (e.g. HazardLevel).

The main channel URN describes a CompositePhenomenon that is comprised of each sub-phenomenon. This is illustrated in the *GetSensors* response in the OWS-6 ER. Example observed property URNs that describe the readings (READGS) channel and the temperature property of the readings channel are:

urn:ogc:def:phenomenon:JPEO-CBD::READGS

urn:ogc:def:phenomenon:JPEO-CBD::READGS:Temperature

The OWS-6 CCSI-SWE ER recommends the following with respect to observed property URNs:

In the future, it may be more logical to include the sensor model or sensor type (e.g. CAD) within the observed property URN to better distinguish between observed properties across CCSI sensors (e.g. urn:ogc:def:phenomenon:JPEO-CBD::CAD:READGS). For instance, the READGS channel from a CAD sensor most likely will not contain the same information as the READGS channel from a biological sensor; yet, in the current implementation, the same URN is used for observations from the READGS channel regardless of the sensor involved. This is particularly true in cases where sensors utilize the Sensor Unique Data portion of the CCSI specification.

Input from the CBRN community is desired to determine the best approach for defining appropriate observed property URN values based on available sensor information. For instance, does the READGS channel look the same for all chemical agent detectors or do the fields contained in a chemical agent detector READGS channel message vary on a manufacturer by manufacturer basis when Sensor Unique Data elements are used? If so, then the manufacturer name, not the sensor type, should most likely be included in the observed property URN to distinguish READGS channel messages from different manufacturers.

11.2.1.3 Identifier URNs

Identifier URNs that describe parts of CCSI information should follow a similar pattern to the observed property URNs:

urn:ogc:def:identifier:JPEO-CBD::[CCSI Element Name]:[CCSI Element Attribute Name]

NOTE CCSI Element Name is the local name of the CCSI element, and CCSI Element Attribute Name is the name of an attribute of that element (if applicable)

An example identifier URN that describes the sensor name element from a CCSI definition file is:

urn:ogc:def:identifier:JPEO-CBD::SensorName

11.2.1.4 Parameter Names and URNs

Finally, CCSI SPS parameter definition URNs and parameter names that describe parameters of SPS tasking messages should adhere to the following patterns respectively:

Definition URN: urn:ogc:def:parameter:JPEO-CBD::[CCSI Standard Command Name]:[CCSI Standard Command Parameter Name]

Parameter Name: [CCSI Standard Command Name]-[CCSI Standard Command Parameter Name]

NOTE CCSI Standard Command Name is the name of the CCSI standard command associated with an SPS tasking input, and CCSI Standard Command Parameter Name is the name of a particular parameter of the associated CCSI Standard Command. Including both the command name and the parameter name allows software to determine with what CCSI command a particular input is associated.

NOTE A dash is used in SPS parameter names to separate CCSI command names and CCSI parameter names, since some CCSI standard command names and parameters include an underscore character in their name.

For example, the definition URN and parameter name for the *Type* parameter of the *Silence* command would be:

Definition URN: urn:ogc:def:parameter:JPEO-CBD::Silence:Type

Parameter Name: Silence-Type

Including the command name in the parameter name ensures that there is no ambiguity as to which command that particular parameter relates to, since a command name is not provided as part of an SPS 1.0 *Submit* request, only parameter names and values.

11.2.2 Expansion of Abbreviated Terms

Where possible, CCSI abbreviations for specific concepts should be spelled out to support understanding from SWE users outside of the CBRN community. The OWS-6 CCSI-SWE ER recommended this:

> In the future, such a mapping can also be used for other CCSI conventions where appropriate, such as mapping CCSI abbreviations that are understandable by those within the DOD CBRN community like "CC" and "SC" to more general, human readable terms that can be easily understood outside of the DOD CBRN community like "Control Component" and "Sensing Component".

For command messages, the CCSI sensor will be expecting the CCSI abbreviation rather than a human readable term, so the mapping from CCSI specific abbreviations should only be performed in certain cases. For cases where CCSI abbreviations cannot or should not be mapped to human readable terms, SWE Common *Category* elements can utilize the concept of *codeSpace*. *codeSpace* is often included with SWE Common *Category* elements to indicate the authority or dictionary that governs the possible terms that can be included in the value of that Category element.

11.2.3 Features/*featureOfInterest*

Both the O & M 1.0.0 and SOS 1.0 specification share the concept of *featureOfInterest*. The SOS 1.0 specification describes a *featureOfInterest* in the following way:

> Features or feature collections that represent the identifiable object(s) on which the sensor systems are making observations. In the case of an in-situ sensor this may be a station to which the sensor is attached representing the environment directly surrounding the sensor. For remote sensors this may be the area or volume that is being sensed, which is not co-located with the sensor. The feature types may be generic Sampling Features (see O&M) or may be specific to the application domain of interest to the SOS. However, features should include spatial information (such as the GML boundedBy) to allow the location to be harvested by OGC service registries.

O & M defines *featureOfInterest* as "a feature of any type (ISO 19109, ISO 19101), which is a representation of the observation target, being the real-world object regarding which the observation is made." *featureOfInterest* is intended to reference a geographic feature, which is a very broad concept. The Oceans community OOSTethys initiative provides some good examples of different features and the concept of *featureOfInterest* at the following site: http://www.oostethys.org/best-practices/foi. The OGC also provides a general document describing features [OGC 08-126]: http://portal.opengeospatial.org/files/index.php?artifact_id=29536.

For in-situ sensors, modeling the *featureOfInterest* as a *SamplingPoint* is typically the best approach, since this represents the fact that the values contained within the O & M *result* element were observations recorded at a particular time or within a particular time period about that particular location. In addition to position information, the *SamplingPoint* contains a child element called *sampledFeature* that typically links to a

domain feature (but can be a different feature type) to provide more context about what feature at that location was being observed (e.g. air, water, a particular river, etc.)

Examples of domain features in the CBRN domain might include things like the air (at a particular point or within a particular geographic area), particular bodies of water or buildings. The *sampledFeature* would need to reference these domain features, and the specific domain feature to reference depends on the current mission in which the CBRN sensors are being employed. Determining CBRN domain features is one area where CBRN community involvement would be particularly useful.

11.3 Encoding Profiles

11.3.1 CCSI Profile of SensorML 1.0.1

The SensorML document describing a CCSI sensor should be a System, since the SensorML document describes a physical sensing system. The sections below describe specific child elements of the SensorML *System* that represents a CCSI sensor.

11.3.1.1 Description

The SensorML document's GML *description* element shall be the value contained in the CCSI Sensor Definition file's *SensorDescription* element.

CCSI Sensor Definition	SensorML
<SensorDescription> Repackaged CAD chemical agent detector. </SensorDescription>	<gml:description> Repackaged CAD chemical agent detector. </gml:description>

11.3.1.2 Keywords

The *keywords* element is useful for sensor search and discovery and should be included and should include a *KeywordList* with *keyword* elements that convey the fact that the system being described is a CBRN sensor, or more specifically, a CCSI sensor. Individual keywords can also be extracted from a CCSI Sensor Definition file if desired. Common keywords include "CCSI", "CBRN", and "Sensor".

11.3.1.3 Identification

The identification section shall contain an *IdentifierList* element with *identifier* elements for each element under the CCSI *SensorIdentification* element found in the CCSI Sensor Definition file. The SensorML *identifier name* attribute shall match the local name of the corresponding CCSI *SensorIdentification* child element. The *identifier* element's *Term definition* attribute shall be of the form urn:ogc:def:identifier:JPEO-CBD::[element local name], and the value shall be the value of that particular element. For SensorIdentification elements that have attributes, each attribute shall also be a separate identifier in the SensorML IdentifierList. The identifier name should match the name of the attribute, and the Term's definition attribute should be of the form

urn:ogc:def:identifier:JPEO-CBD::[attribute name]. The value should be the value of the attribute. The table below illustrates this mapping.

CCSI Sensor Defintion	SensorML

<SensorIdentification>
  <SensorName>CAD</SensorName>
  <SensorType class="CHM" variant="PNT" name="SC001"/>
  <SensorModel>FPS</SensorModel>
  <SensorDescription>
  Chemical Agent Detector (JCAD) with CCSI capabilities for use in the FCS program.
  </SensorDescription>
</SensorIdentification>
	<identification>
	  <IdentifierList>
	…
	<identifier name="SensorName">
	  <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorName">
	  <value>CAD</value>
	  </Term>
	  </identifier>
	  <identifier name="class">
	  <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorType:class">
	  <value>CHM</value>
	  </Term>
	  </identifier>
	  <identifier name="variant">
	  <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorType:variant">
	  <value>PNT</value>
	  </Term>
	  </identifier>
	  <identifier name="name">
	  <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorType:name">
	  <value>SC001</value>
	  </Term>
	  </identifier>
	  <identifier name="SensorModel">
	  <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorModel">
	  <value>FPS</value>
	  </Term>
	  </identifier>
	  <identifier name="SensorDescription">
	  <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorDescription">
	  <value>
	  Chemical Agent Detector (JCAD) with CCSI capabilities for use in the FCS program.
	  </value>
	  </Term>
	  </identifier>
	  </IdentifierList>
	</identification>

In addition to values mapped from the CCSI Sensor Definition file, the identifiers section should also include an identifier element for the unique ID of the sensor system. The value of that identifier should be an OGC definition URN that follows the URN format described above (e.g. urn:ogc:def:procedure:JPEO-CBD::CAD_1). The definition attribute of the Term element beneath the identifier should be urn:ogc:def:identifier:OGC::uniqueID.


```
<identifier name="uniqueID">
   <Term definition="urn:ogc:def:identifier:OGC::uniqueID">
      <value>urn:ogc:def:procedure:JPEO-CBD::CAD_1</value>
   </Term>
</identifier>
```

11.3.1.4 Classifiers

The *classifiers* section shall contain a *ClassifierList* with classifier elements for intendedApplication and sensorType at a minimum. The intendedApplication should be a string that conveys the fact that the sensor is used for CBRN related operations such as "CBRN Detection". The sensor type should be populated based on the type of sensor found in the CCSI *SensorType* element's *class* attribute. An additional sensor type classifier may be added based on the *SensorType variant* attribute. In either case, the values of these classifiers should be expanded, human-readable terms (e.g. "Radiological" for "RAD" and "Point" for "PNT"). The tables below provide examples of expanded terms.

SensorType Class Abbreviation	Full Term
BIO	Biological
CHM	Chemical
NKN	Not Known
NOS	Not Specified
RAD	Radiological
NUC	Nuclear
EXP	Experimental
MET	Meterological

SensorType Variant Abbreviation	Full Term
PNT	Point
STD	Stand-Off

11.3.1.5 Characteristics and Capabilities

The *CCSIVersion* and *ReleaseVersion* elements of the *SensorVersion* element are mapped to a SensorML *characteristics* element called "HW/SW Characteristics". Each SWE *value* element of each *field* of the "HW/SW Characteristics" *DataRecord* should follow the pattern:

[major].[moderate].[minor].[patch] where each item (major, moderate, minor, and patch) corresponds to that attribute of an associated CCSI element.

CCSI Sensor Definition	SensorML
<SensorVersion xmlns=""> <CcsiVersion major="1" moderate="0" minor="0" patch="0"/> <ReleaseVersion> <SensingUnit>SC001</SensingUnit> <HwVersion major="1" moderate="0"/> <SwVersion major="1" moderate="0"/> <Description> This is the base version of the JCAD chemical agent detector. </Description> </ReleaseVersion> … </SensorVersion>	<characteristics name="HW/SW Characteristics"> <swe:DataRecord definition="urn:ogc:def:property:hwSwCharacteristics"> <swe:field name="CCSI Version"> <swe:Category definition="urn:ogc:def:property:JPEO-CBD::CCSIVersion"> <swe:value>1.0.0.0</swe:value> </swe:Category> </swe:field> <swe:field name="HW Version"> <swe:Category definition="urn:ogc:def:property:JPEO-CBD::HWVersion"> <swe:value>1.0</swe:value> </swe:Category> </swe:field> <swe:field name="SW Version"> <swe:Category definition="urn:ogc:def:property:JPEO-CBD::SWVersion"> <swe:value>1.0</swe:value> </swe:Category> </swe:field> </swe:DataRecord> </characteristics>

The SensorML description should include a *capabilities* element for each *Capability* element defined in a CCSI Sensor Definition file. Each *Capability* sub-element of the *Capability* element of the *SensorVersion* element maps to a SensorML *capabilities* element as illustrated below. Each *capabilities* element includes a SWE *DataRecord* with itemType, levelUnits, and other fields corresponding to the CCSI ItemType, LevelUnits, and other elements that may be present as part of a Capability element.

CCSI Sensor Definition	SensorML
<Capability> <ItemType>GA</ItemType> <LevelUnits>Bars</LevelUnits> </Capability>	<capabilities name="GA Measurement Properties"> <swe:DataRecord definition="urn:ogc:def:property:JPEO-CBD::capabilityInformation"> <swe:field name="itemType"> <swe:Category definition="urn:ogc:def:property:JPEO-CBD::itemType"> <swe:value>GA</swe:value> </swe:Category> </swe:field> <swe:field name="levelUnits"> <swe:Category definition="urn:ogc:def:property:JPEO-CBD::levelUnits"> <swe:value>Bars</swe:value> </swe:Category> </swe:field> </swe:DataRecord> </capabilities>

In cases where SWE Common *Quantity* elements are used along with a unit of measure, understanding whether the unit of measure provided by CCSI is a Unified Code for Units of Measure (UCUM) identifier is important. The CCSI unit of measure definition does not require the units specified to be UCUM measures, so this means that units of measure specified could either be UCUM identifiers or non-UCUM identifiers. The SWE Common *uom* element's *code* attribute is intended to specify a UCUM identifier. In cases where the CCSI unit of measure does not correspond to a UCUM identifier, use the *xlink:href* attribute to link to the unit of measure definition. However, handling UCUM versus non-UCUM identifiers dynamically is somewhat difficult.

11.3.1.6 Contacts

The Sensor Definition file *Manufacturer* and *Program* elements map to SensorML contact member elements with *xlink:role* attributes of "urn:ogc:def:identifier:OGC::manufacturer" and "urn:ogc:def:identifier:OGC::program" respectively. Each sub-element under Manufacturer and Program maps to particular elements in a SensorML *ResponsibleParty* element.

CCSI Sensor Definition	SensorML
<Manufacturer> <PointOfContactNameText>Paul Knight</PointOfContactNameText> <PointOfContactJobTitleNameText>Program Manager</PointOfContactJobTitleNameText> <PointOfContactAddressLineText>Smith's Detection</PointOfContactAddressLineText> <PointOfContactCityNameText>Watford</PointOfContactCityNameText> <PointOfContactStateNameText/> <PointOfContactCountryCode>UK</PointOfContactCountryCode> </Manufacturer>	<contact> <ContactList> <member xlink:role="urn:ogc:def:identifier:OGC::manufacturer"> <ResponsibleParty> <individualName>Paul Knight</individualName> <contactInfo> <address> <deliveryPoint>Smith's Detection</deliveryPoint> <city>Watford</city> <administrativeArea/> <country>UK</country> </address> </contactInfo> </ResponsibleParty> </member> … </ContactList> </contact>

11.3.1.7 Inputs

The *inputs* element typically defines one or more phenomena observed by the sensing system. In the CCSI case, the *inputs* section should include an *InputList* with one or more child *input* elements with child SWE Common *ObservableProperty* elements that convey the fact that the sensing system observes some CBRN related phenomenon. The

definition attribute of each *ObservableProperty* element should be an OGC definition URN with a prefix value of "urn:ogc:def:phenomenon:JPEO-CBD::". The XSLT translations provided in this document produce the following *inputs* element representing a generic CBRN hazard.

```
<inputs>
   <InputList>
      <input name="genericCBRNHazard">
         <swe:ObservableProperty
definition="urn:ogc:def:phenomenon:JPEO-CBD::toxicAgent"/>
      </input>
   </InputList>
</inputs>
```

11.3.1.8 Outputs

Each *Channel* element listed under the *Channels* element in the CCSI Sensor Definition file should map to a SensorML *output* element with a child SWE Common *DataRecord* with a definition URN that follows the previously defined URN scheme. The *Metadata* section of a *Channel* element identifies sensor unique data included with a particular channel. The CCSI specification includes required and optional information for channels like the readings and alerts channels. For instance, according to the CCSI schema, the readings channel can include Detect, Level, LevelConfidenceInterval, Id, and DetailsAvailable information in addition to sensor unique data like Temperature. SWE *constraint* and *uom* elements for each SWE *field* should be populated based on the *DataDefinitions* element found in the CCSI Sensor Definition file, which defines any sensor unique data elements (note the use of *uom* and *constraint* elements in the READGS_FilteredDoseRate and REAGS_MissionDose fields below). The XML snippet below provides an example *outputs* section for a radiological detector.

```
<outputs>
    <OutputList>
        …omitted for brevity…
        <output name="READGS">
            <swe:DataRecord gml:id="READGS DATA RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS">
                <swe:field name="Mode">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                </swe:field>
                <swe:field name="READGS_Sensor">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                </swe:field>
                <swe:field name="READGS_ReadingID">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                        <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                    </swe:Category>
                </swe:field>
                <swe:field name="READGS_Detect">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                </swe:field>
                <swe:field name="READGS_Level">
                    <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                </swe:field>
                <swe:field name="READGS_LevelConfidenceInterval">
```



```
                    <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                </swe:field>
                <swe:field name="READGS_Id">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                </swe:field>
                <swe:field name="READGS_DetailsAvailable">
                    <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                </swe:field>
                <swe:field name="READGS_FilteredDoseRate">
                    <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                        <swe:uom code="cGy/hr"/>
                        <swe:constraint>
                            <swe:AllowedValues>
                                <swe:interval>0.0 999.9</swe:interval>
                            </swe:AllowedValues>
                        </swe:constraint>
                    </swe:Quantity>
                </swe:field>
                <swe:field name="READGS_MissionDose">
                    <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                        <swe:uom code="cGy"/>
                        <swe:constraint>
                            <swe:AllowedValues>
                                <swe:interval>0.0 999.9</swe:interval>
                            </swe:AllowedValues>
                        </swe:constraint>
                    </swe:Quantity>
                </swe:field>
            </swe:DataRecord>
        </output>
        <output name="STATUS">
            <swe:DataRecord gml:id="STATUS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS">
                <swe:field name="Mode">
                    <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                </swe:field>
                <swe:field name="STATUS_BIT">
                    <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:BIT"/>
                </swe:field>
                <swe:field name="STATUS_Alert">
                    <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:Alert"/>
                </swe:field>
                <swe:field name="STATUS_Maint">
                    <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:Maint"/>
                </swe:field>
            </swe:DataRecord>
        </output>
    </OutputList>
</outputs>
```

11.3.1.9 Position

The geographic location of a CCSI sensor should be encoded in the SensorML *positions* element. The *positions* element should include a child *PositionList* element with one or more *position* sub-elements. Each *position* element should include a child SWE Common *Position* element. This allows the geographic location to be specified along with an orientation value, which is important is the sensor is emplaced facing a certain direction. The *orientation Vector* can also include *coordinate* elements for tilt and roll in addition to heading for precise orientation information.


```
<positions>
    <PositionList>
        <position name="sensorPosition">
            <swe:Position localFrame="#SENSOR_FRAME"
referenceFrame="urn:ogc:def:crs:EPSG:4326">
                <swe:location>
                    <swe:Vector definition="urn:ogc:def:vector:OGC:location">
                        <swe:coordinate name="latitude">
                            <swe:Quantity axisID="Y"
definition="urn:ogc:def:phenomenon:latitude">
                                <swe:uom code="deg"/>
                                <swe:value>35.123456</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                        <swe:coordinate name="longitude">
                            <swe:Quantity axisID="X"
definition="urn:ogc:def:phenomenon:longitude">
                                <swe:uom code="deg"/>
                                <swe:value>-70.123456</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                        <swe:coordinate name="altitude">
                            <swe:Quantity axisID="Z"
definition="urn:ogc:def:phenomenon:altitude">
                                <swe:uom code="m"/>
                                <swe:value>0.0</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                    </swe:Vector>
                </swe:location>
                <swe:orientation>
                    <swe:Vector definition="urn:ogc:def:vector:OGC:orientation">
                        <swe:coordinate name="trueHeading">
                            <swe:Quantity
definition="urn:ogc:def:phenomenon:angleToNorth">
                                <swe:uom code="deg"/>
                                <swe:value>25.0</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                    </swe:Vector>
                </swe:orientation>
            </swe:Position>
        </position>
    </PositionList>
</positions>
```

11.3.2 CCSI Profile of O & M 1.0.0

11.3.2.1 *procedure*

The *procedure* element's *xlink:href* attribute value should be the OGC definition URN identifier for the sensor that produced the observation. This should match the uniqueID identifier found in the SensorML document describing that sensor as well as the *procedure xlink:href* value for the sensor defined in an SOS Capabilities document that advertises the sensor that produced the observation, if an SOS is used to obtain the particular O & M observation.

11.3.2.2 *observedProperty*

The *observedProperty* element's *xlink:href* attribute value should be the OGC definition URN for the specific CCSI channel whose data is included in the *result* element.

11.3.2.3 *featureOfInterest*

If the sensor that produced the O & M observation is fixed at a particular location, then the *featureOfInterest* element should reference through an *xlink:href* attribute or should include as a child element a *SamplingPoint* element that includes that sensor's location. If an *xlink:href* attribute is used, then the attribute value should be a URL that resolves to a *SamplingPoint* – the same *SamplingPoint* that would be included had the *SamplingPoint* been defined as a child of the *featureOfInterest* element. The *featureOfInterest* element in the mobile case can be a GML *FeatureCollection* or one of several other GML types representing the extents of the collection of observations or can be something more abstract (e.g. a domain feature – see discussion of features below).

11.3.2.4 *samplingTime*

The *samplingTime* element should include a child GML *TimeInstant* or *TimePeriod* element depending on whether the O & M observation contains one or more values. The time value contained with the *TimeInstant*'s *timePosition* element should be an ISO 8601 value that at a minimum matches the channel message *dtg* attribute value converted to ISO 8601 or a more specific time value (e.g. the time value contained in a READGS channel message). The *TimePeriod*'s *beginPosition* and *endPosition* element values should be ISO 8601 time values that match the first and last times contained in the CCSI channel data.

11.3.2.5 *result*

A SWE Common *DataRecord* should be used within the O & M *result* element to define the collection of one or more fields contained within a CCSI channel message. If a CCSI channel message contains a single *Msg* element, then the O & M *result* element can include a child SWE Common *DataRecord* or a child SWE Common *DataArray* with a SWE Common *DataRecord elementType* and a *TextBlock* encoding. Using a *DataArray* to describe a single channel message in most cases is not necessary, since the structure of the *DataArray* adds some complexity over using a simple *DataRecord* alone. For consistency, use a *DataArray* to describe a single observation only if the sensor or service providing the O & M could return one or more observations at a time; in this case, the sensor or service will always return a *DataArray result* regardless of the number of distinct readings contained in the observation. In the single observation case, the *samplingTime* element should include a GML *TimeInstant* element with an ISO 8601 *timePosition* value the matches the time when the observation was recorded.

If a CCSI channel message contains multiple *Msg* elements or the service producing O & M returns a set of more than one CCSI channel messages for a given channel, then the O & M document describing that message or set of messages should include a *result* element with a SWE Common *DataArray* with a SWE *DataRecord elementType* and a *TextBlock* encoding. The *DataArray elementType* should include the same SWE Common *DataRecord* that defines the particular channel plus a time component. The time component ensures that the time for each observation in the *DataArray* can be determined. The time component can either be an ISO 8601 based time value with a *definition* attribute of urn:ogc:def:trs:ISO-8601::Gregorian+UTC (from urn.opengis.net)

or can be based on unix time with a *definition* attribute of urn:ogc:def:phenomenon:JPEO-CBD::dtg. If the time is based on Unix time, then the time field in the SWE Common *DataRecord* should be a SWE Common *Time* element that includes a *referenceTime* attribute of "1970-01-01T00:00:00Z" and a child SWE Common *uom* element with a *code* attribute of "s", identifying the fact that the time field represents Unix time. If the sensor is mobile, the *elementType DataRecord* of the *DataArray* should also include fields for latitude, longitude, and optionally altitude to ensure that the observation values can be matched to the location of the observation. These fields can be included as individual fields in the main *DataRecord* or can be included as individual coordinates within a *Vector* field within the main *DataRecord*. The first time and last time values in the *DataArray values* section should match the *samplingTime TimePeriod beginPosition* and *endPosition* values respectively.

11.3.2.6 *uom*

When defining a unit of measure for SWE Common *Quantity fields*, use the *code* attribute when referring to a UCUM unit of measure and the *xlink:href* attribute when referencing a non-UCUM unit of measure.

11.4 Service Profiles

The sections below specify CCSI profiles of the SOS and SPS standards. The profiles focus on the 1.0 versions of those standards and provide additional thoughts on how the 2.0 versions of those standards might change how things work in the current 1.0 versions of the standards. While the profiles below are based on CCSI, in many cases, they are general enough to utilize for general CBRN sensors not just CCSI-compliant sensors. Note that, for reasons described previously, this document does not define a profile for the SAS.

11.4.1 CCSI Profile of SOS 1.0/SOS 2.0 Concepts

An SOS that provides access to CCSI sensors and their data, hereafter referred to as a CCSI SOS, shall at a minimum implement the mandatory operations of the SOS 1.0 specification: *GetCapabilities*, *DescribeSensor*, and *GetObservation*. The CCSI SOS may optionally implement the SOS 1.0 Transactional Profile and the corresponding transactional operations: *RegisterSensor* and *InsertObservation*. The Transactional Profile should be implemented in scenarios where a CCSI SOS is not used in conjunction with a CCSI SPS as defined in section 10.3. This profile defines specific characteristics of SOS operation responses for the mandatory operations and specific characteristics of SOS operation requests and responses for the transactional operations; all SOS operation requests for the mandatory operations should conform to the SOS 1.0 specification.

11.4.1.1 *GetCapabilities* Response

11.4.1.1.1 *Keywords*

The OWS Common *Keywords* element should include individual *Keyword* elements that identify the SOS as a service that provides access to CCSI/CBRN sensors and data. Example keywords include “Chemical”, “CCSI”, “CBRN”, “Biological”, etc.

11.4.1.1.2 *OperationsMetadata*

In the *OperationsMetadata* section, the *DescribeSensor outputFormat* can optionally include CCSI Sensor Definition format. At some point, an appropriate Multipurpose Internet Mail Extensions (MIME) type format for CCSI Sensor Definition files should be created and registered with the Internet Assigned Numbers Authority (IANA), but the string text/xml;subtype=”ccsi/1.0.1” could be used for now. This is similar to the unregistered MIME types used for SWE formats in the 1.0 specifications. The MIME type string text/xml;subtype=”sensorML/1.0.1” can be used to indicate that the *DescribeSensor* outputFormat can be SensorML 1.0.1.

11.4.1.1.3 Contents - *Offerings*

In terms of offerings, there should be one offering per sensor, since the observed property for a sensor is potentially unique to that sensor, and the service could advertise a variety of sensor types from different manufacturers. The offering *responseFormat* can optionally include CCSI formats (text/xml;subtype=”ccsi/1.0.1”) and should, at a minimum, include the O & M MIME type (text/xml;subtype=”om/1.0.0”).

If the SOS serves up only the latest channel data, then define the offering time as a GML *TimeInstant* with a child *timePosition* element with an *indeterminate* attribute of “now” or with a specific ISO 8601 time value that corresponds to the time of the last recorded observation and return O & M as defined by the XSLT output included in this document.

```
<sos:time>
    <gml:TimeInstant xsi:type="gml:TimeInstantType">
        <gml:timePosition indeterminate="now"/>
    </gml:TimeInstant>
</sos:time>
```

If the SOS serves up archived channel data, then define the offering time as a GML *TimePeriod* with a fixed starting and end time corresponding to the start and end times for which observations have been recorded or with a fixed start time and an indeterminate end time of “now” corresponding to the times of the first recorded observation and the last recorded observation.

```
<sos:time>
    <gml:TimePeriod xsi:type="gml:TimePeriodType">
        <gml:beginPosition>2010-04-30T00:00:00Z</gml:beginPosition>
        <gml:endPosition indeterminate="now"/>
    </gml:TimePeriod>
</sos:time>
```


The server should use a SWE Common *DataArray* with *TextBlock* encoding in the response to a *GetObservation* request for a time period as described in the CCSI Profile of O & M 1.0.0. This will eliminate redundant XML overhead and ensure that responses to *GetObservation* requests are as compact as possible.

The offering's *boundedBy* element should be a GML *Envelope* element with a *srsName* attribute that defines the Coordinate Reference System (CRS) of the child *lowerCorner* and *upperCorner* elements. In most cases, the *srsName* attribute should reference European Petroleum Survey Group (EPSG) CRS 4326 (urn:ogc:def:crs:EPSG::4326) for WGS84 latitude and longitude coordinates or EPSG CRS 4979 (urn:ogc:def:crs:EPSG::4979) for WGS84 latitude, longitude, and altitude coordinates. The *lowerCorner* is the geographic minimum for the bounding box that includes every location where the sensor has recorded an observation and the *upperCorner* is the geographic maximum for the bounding box that includes every location where the sensor has recorded an observation.

```
<gml:boundedBy>
    <gml:Envelope srsName="urn:ogc:def:crs:EPSG::4326">
        <gml:lowerCorner>29.9837888 -90.2580453</gml:lowerCorner>
        <gml:upperCorner>29.9837888 -90.2580453</gml:upperCorner>
    </gml:Envelope>
</gml:boundedBy>
```

If the sensor is in-situ or non-mobile, then the *lowerCorner* and *upperCorner* elements will contain the same value, the location of the sensor, and the feature of interest associated with the offering should reference a *SamplingPoint* that defines the location of that sensor. In the case of mobile sensors, the offering can include multiple features of interest that reference *SamplingPoint* features for locations where that sensor has recorded an observation, or the offering can reference a more general feature (domain feature or a GML *Feature* (point, line, polygon) or *FeatureCollection*).

There should be a separate *observedProperty* element for each channel that the sensor supports. The example below illustrates a valid Capabilities document.

```
<?xml version="1.0" encoding="utf-8"?>
<sos:Capabilities version="1.0.0" updateSequence="2005-12-14T03:12:39-06"
xsi:schemaLocation="http://www.opengis.net/sos/1.0
http://schemas.opengis.net/sos/1.0.0/sosAll.xsd"
xmlns:sos="http://www.opengis.net/sos/1.0" xmlns:om="http://www.opengis.net/om/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:gml="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <ows:ServiceIdentification xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:swe="http://www.opengis.net/swe/1.0">
        <ows:Title>CCSI Sensor SOS</ows:Title>
        <ows:Abstract>A work in progress....</ows:Abstract>
        <ows:Keywords>
            <ows:Keyword>CCSI</ows:Keyword>
            <ows:Keyword>CBRN</ows:Keyword>
            <ows:Keyword>Chemical</ows:Keyword>
            <ows:Keyword>Biological</ows:Keyword>
            <ows:Keyword>Radiological</ows:Keyword>
            <ows:Keyword>Nuclear</ows:Keyword>
        </ows:Keywords>
        <ows:ServiceType codeSpace="http://opengeospatial.net">OGC:SOS</ows:ServiceType>
        <ows:ServiceTypeVersion>1.0.0</ows:ServiceTypeVersion>
        <ows:Fees>NONE</ows:Fees>
```



```
        <ows:AccessConstraints>NONE</ows:AccessConstraints>
    </ows:ServiceIdentification>
    <ows:ServiceProvider xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:swe="http://www.opengis.net/swe/1.0">
        <ows:ProviderName>SOS Provider</ows:ProviderName>
        <ows:ProviderSite xlink:href="http://www.sosprovider.com"/>
        <ows:ServiceContact>
            <ows:IndividualName>SOS Developer</ows:IndividualName>
            <ows:PositionName>Software Developer</ows:PositionName>
            <ows:ContactInfo>
                <ows:Phone>
                    <ows:Voice></ows:Voice>
                </ows:Phone>
                <ows:Address>
                    <ows:DeliveryPoint></ows:DeliveryPoint>
                    <ows:City></ows:City>
                    <ows:AdministrativeArea></ows:AdministrativeArea>
                    <ows:PostalCode></ows:PostalCode>
                    <ows:Country></ows:Country>

    <ows:ElectronicMailAddress>sosprovider@sosprovider.com</ows:ElectronicMailAddress>
                </ows:Address>
            </ows:ContactInfo>
            <ows:Role/>
        </ows:ServiceContact>
    </ows:ServiceProvider>
    <ows:OperationsMetadata xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:swe="http://www.opengis.net/swe/1.0">
        <ows:Operation name="GetCapabilities">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get
xlink:href="http://sosprovider.com/CCSISOS/GetCapabilities?"/>
                    <ows:Post
xlink:href="http://sosprovider.com/CCSISOS/GetCapabilities"/>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="service">
                <ows:AllowedValues>
                    <ows:Value>SOS</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="updateSequence">
                <ows:AnyValue/>
            </ows:Parameter>
            <ows:Parameter name="AcceptVersions">
                <ows:AllowedValues>
                    <ows:Value>1.0.0</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="Sections">
                <ows:AllowedValues>
                    <ows:Value>ServiceIdentification</ows:Value>
                    <ows:Value>ServiceProvider</ows:Value>
                    <ows:Value>OperationsMetadata</ows:Value>
                    <ows:Value>Contents</ows:Value>
                    <ows:Value>All</ows:Value>
                    <ows:Value>Filter_Capabilities</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="AcceptFormats">
                <ows:AllowedValues>
                    <ows:Value>text/xml</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
        </ows:Operation>
        <ows:Operation name="GetObservation">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Post
xlink:href="http://sosprovider.com/CCSISOS/GetObservation"/>
                </ows:HTTP>
```

```
            </ows:DCP>
            <ows:Parameter name="version">
                <ows:AllowedValues>
                    <ows:Value>1.0.0</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="service">
                <ows:AllowedValues>
                    <ows:Value>SOS</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="srsName">
                <ows:AnyValue/>
            </ows:Parameter>
            <ows:Parameter name="offering">
                <ows:AllowedValues>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector 1</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_2</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_3</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_4</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="eventTime">
                <ows:AnyValue/>
            </ows:Parameter>
            <ows:Parameter name="procedure">
                <ows:AllowedValues>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_2</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_3</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_4</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="observedProperty">
                <ows:Value>urn:ogc:def:phenomenon:JPEO-CBD::ALERTS</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:JPEO-CBD::MAINT</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:JPEO-CBD::READGS</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:JPEO-CBD::STATUS</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="featureOfInterest">
                <ows:AnyValue/>
            </ows:Parameter>
            <ows:Parameter name="result">
                <ows:AnyValue/>
            </ows:Parameter>
            <ows:Parameter name="responseFormat">
                <ows:AllowedValues>
                    <ows:Value>text/xml;subtype="om/1.0.0"</ows:Value>
                    <ows:Value>text/xml;subtype="ccsi/1.0.0"</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="resultModel">
                <ows:AllowedValues>
                    <ows:Value>om:Observation</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="responseMode">
                <ows:AllowedValues>
                    <ows:Value>resultTemplate</ows:Value>
                    <ows:Value>inline</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
        </ows:Operation>
        <ows:Operation name="DescribeSensor">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Post
xlink:href="http://sosprovider.com/CCSISOS/DescribeSensor"/>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
                <ows:AllowedValues>
```



```
                    <ows:Value>1.0.0</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="service">
                <ows:AllowedValues>
                    <ows:Value>SOS</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="outputFormat">
                <ows:AllowedValues>
                    <ows:Value>text/xml;subtype="sensorML/1.0.1"</ows:Value>
                    <ows:Value>text/xml;subtype="ccsi/1.0.0"</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="procedure">
                <ows:AllowedValues>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_2</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_3</ows:Value>
                    <ows:Value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_4</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
        </ows:Operation>
    </ows:OperationsMetadata>
    <sos:Filter_Capabilities xmlns:ogc="http://www.opengis.net/ogc"
xmlns:ows="http://www.opengis.net/ows/1.1" xmlns:swe="http://www.opengis.net/swe/1.0">
        <ogc:Spatial_Capabilities>
            <ogc:GeometryOperands>
                <ogc:GeometryOperand>gml:Envelope</ogc:GeometryOperand>
                <ogc:GeometryOperand>gml:Polygon</ogc:GeometryOperand>
                <ogc:GeometryOperand>gml:Point</ogc:GeometryOperand>
                <ogc:GeometryOperand>gml:LineString</ogc:GeometryOperand>
            </ogc:GeometryOperands>
            <ogc:SpatialOperators>
                <ogc:SpatialOperator name="BBOX"/>
                <ogc:SpatialOperator name="Contains"/>
                <ogc:SpatialOperator name="Intersects"/>
                <ogc:SpatialOperator name="Overlaps"/>
            </ogc:SpatialOperators>
        </ogc:Spatial_Capabilities>
        <ogc:Temporal_Capabilities>
            <ogc:TemporalOperands>
                <ogc:TemporalOperand>gml:TimeInstant</ogc:TemporalOperand>
                <ogc:TemporalOperand>gml:TimePeriod</ogc:TemporalOperand>
            </ogc:TemporalOperands>
            <ogc:TemporalOperators>
                <ogc:TemporalOperator name="TM_During"/>
                <ogc:TemporalOperator name="TM_Equals"/>
                <ogc:TemporalOperator name="TM_After"/>
                <ogc:TemporalOperator name="TM_Before"/>
            </ogc:TemporalOperators>
        </ogc:Temporal_Capabilities>
        <ogc:Scalar_Capabilities>
            <ogc:ComparisonOperators>
                <ogc:ComparisonOperator>Between</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>EqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>NotEqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>LessThan</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>LessThanEqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>GreaterThan</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>GreaterThanEqualTo</ogc:ComparisonOperator>
                <ogc:ComparisonOperator>Like</ogc:ComparisonOperator>
            </ogc:ComparisonOperators>
        </ogc:Scalar_Capabilities>
        <ogc:Id_Capabilities>
            <ogc:FID/>
            <ogc:EID/>
        </ogc:Id_Capabilities>
    </sos:Filter_Capabilities>
    <sos:Contents>
        <sos:ObservationOfferingList>
```

```
            <sos:ObservationOffering gml:id="urn:ogc:def:procedure:JPEO-
CBD::RadDetector_1">
                <gml:name>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</gml:name>
                <gml:boundedBy>
                    <gml:Envelope srsName="urn:ogc:def:crs:EPSG:4326">
                        <gml:lowerCorner>18.556825 -72.297935</gml:lowerCorner>
                        <gml:upperCorner>18.556825 -72.297935</gml:upperCorner>
                    </gml:Envelope>
                </gml:boundedBy>
                <sos:time>
                    <gml:TimeInstant xsi:type="gml:TimeInstantType">
                        <gml:timePosition indeterminate="now"/>
                    </gml:TimeInstant>
                </sos:time>
                <sos:procedure xlink:href="urn:ogc:def:procedure:JPEO-
CBD::RadDetector_1"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS"/>
                <sos:featureOfInterest xlink:href="RadDetector_1_Point"/>
                <sos:responseFormat>text/xml;subtype="om/1.0.0"</sos:responseFormat>
                <sos:resultModel
xmlns:ns="http://www.opengis.net/om/1.0">ns:Observation</sos:resultModel>
                <sos:responseMode>inline</sos:responseMode>
                <sos:responseMode>resultTemplate</sos:responseMode>
            </sos:ObservationOffering>

            …Omitted for brevity…

            <sos:ObservationOffering gml:id="urn:ogc:def:procedure:JPEO-
CBD::RadDetector 4">
                <gml:name>urn:ogc:def:procedure:JPEO-CBD::RadDetector_4</gml:name>
                <gml:boundedBy>
                    <gml:Envelope srsName="urn:ogc:def:crs:EPSG:4326">
                        <gml:lowerCorner>18.55624 -72.298306</gml:lowerCorner>
                        <gml:upperCorner>18.55624 -72.298306</gml:upperCorner>
                    </gml:Envelope>
                </gml:boundedBy>
                <sos:time>
                    <gml:TimeInstant xsi:type="gml:TimeInstantType">
                        <gml:timePosition indeterminate="now"/>
                    </gml:TimeInstant>
                </sos:time>
                <sos:procedure xlink:href="urn:ogc:def:procedure:JPEO-
CBD::RadDetector_4"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS"/>
                <sos:observedProperty xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS"/>
                <sos:featureOfInterest xlink:href="RadDetector 4 Point"/>
                <sos:responseFormat>text/xml;subtype="om/1.0.0"</sos:responseFormat>
                <sos:resultModel
xmlns:ns="http://www.opengis.net/om/1.0">ns:Observation</sos:resultModel>
                <sos:responseMode>inline</sos:responseMode>
                <sos:responseMode>resultTemplate</sos:responseMode>
            </sos:ObservationOffering>
        </sos:ObservationOfferingList>
    </sos:Contents>
</sos:Capabilities>
```


11.4.1.2 *DescribeSensor* Response

The response to a valid *DescribeSensor* request shall be either a SensorML 1.0.1 document that complies with the CCSI profile of SensorML defined above (when the request's outputFormat attribute indicates a value of "text/xml;subtype="sensorML/1.0.1"") or should be a CCSI-compliant Sensor Definition file when the request's outputFormat is "text/xml;subtype="ccsi/1.0.1"".

11.4.1.3 *GetObservation* Response

The response to a valid *GetObservation* request shall either be an O & M 1.0.0 document that complies with the CCSI profile of O & M defined above (when the responseFormat is "text/xml;subtype="om/1.0.0"") or should be a CCSI-compliant channel message (when the responseFormat is "text/xml;subtype="ccsi/1.0.1""). If the *GetObservation* request includes a time period for obtaining observations and the SOS advertises a time period in the Capabilities document for the offering corresponding to the *GetObservation* request, the response should be an O & M document that uses a SWE Common *DataArray* to describe each observation that has occurred within that time range. If the SOS advertises only the latest observation, then the O & M document should match the output provided by the output of the XSLT translations in this document.

11.4.1.4 *RegisterSensor* Request

The *RegisterSensor* request shall include a SensorML document in the *SensorDescription* element that complies with the CCSI profile of SensorML 1.0.1 and an empty or populated O & M document in the *ObservationTemplate* element that complies with the CCSI profile of O & M 1.0.0. The SOS service shall use the provided unique ID identifier of the sensor described in the SensorML document as the *AssignedSensorId* that is returned in the *RegisterSensor* response. At a minimum, a CCSI SOS-T client should register a CCSI sensor with an *ObservationTemplate* corresponding to that sensor's READGS channel. If multiple *RegisterSensor* requests are issued for the same sensor with *ObservationTemplate* elements that have different O & M *observedProperty* values, then the CCSI SOS should maintain a one sensor-per-offering construct and group these different observed properties under the offering for the sensor being registered. This eliminates the ambiguity that current exists in the SOS 1.0 specification for handling sensors that support multiple channels.

11.4.1.5 *RegisterSensor* Response

The *AssignedSensorId* element shall be the same unique ID identifier as defined in the SensorML that was sent to the server. If the uniqueID identifier has been omitted from the SensorML document that was sent to the server, then the server should assign a unique ID to that sensor.

11.4.1.6 *InsertObservation* Request

The *InsertObservation* request shall include an O & M document that complies with the CCSI profile of SensorML 1.0.1 defined above. The *AssignedSensorId* element shall match the value of the *AssignedSensorId* element found in the *RegisterSensor* response

when the sensor was registered with the SOS. The O & M document's procedure *xlink:href* attribute shall match the *AssignedSensorId* value.

11.4.1.7 *InsertObservation* Response

The response to a valid *InsertObservation* request shall include a unique observation ID in the *AssignedObservationId* element.

11.4.1.8 SOS 2.0, Streaming Data and the *InsertResult* and *InsertResultTemplate* Operations

SOS 2.0, which is currently being developed within an OGC Standards Working Group (SWG) and at the time of this report has not been released publicly through the OGC Request for Comments (RFC) process, defines a means for streaming sensor data efficiently by minimizing XML overhead using the optional *InsertResultTemplate* and *InsertResult* operations. These operations are designed to work in much the same way as the transactional operations described above in the SOS 1.0 specification but eliminate XML overhead when inserting observation data into the SOS. The *InsertResultTemplate* operation allows a producer (either the sensor itself or a controlling application connected to the sensor) to publish an O & M template with SWE Common encoding of result data to the SOS before inserting observations. This allows the SOS to have an understanding of the data that will be inserted by the sensor. When new observations are collected by the producer, it can issue an *InsertResult* request to the SOS and provide only the text or binary data that would be found in the O & M result element rather than inserting the entire O & M structure, making observation insertion more efficient. This approach can be used in lieu of the *RegisterSensor* and *InsertObservation* approach defined in section 10.3 as well as in section 11.4.1.

On the consumer side, an SOS client can then call *GetResultTemplate* for a particular sensor to retrieve the O & M template with SWE Common encoding, allowing it to understand the data coming from the SOS for a particular sensor. After the client has an understanding of the sensor data, it can subsequently call *GetResult* to retrieve text or binary for sensor observations without retrieving the entire O & M structure with every request, making observation retrieval more efficient. This approach is particularly applicable in cases where the bandwidth between an SOS and an SOS client is limited, as may be the case in real world CBRN sensor operations.

The SOS 2.0 specification also appears to align well with the one sensor-per-offering construct defined in the CCSI Profile of SOS 1.0.0. The SOS 2.0 specification may restrict offerings to referencing a single procedure.

11.4.2 CCSI Profile of SPS 1.0.0/SPS 2.0 Concepts

An SPS that provides access to CCSI sensors and their data, hereafter referred to as a CCSI SPS, shall at a minimum implement the mandatory operations of the SPS 1.0 specification: *GetCapabilities*, *DescribeTasking*, *Submit*, and *DescribeResultAccess*. This profile defines specific characteristics of SPS operation responses for the mandatory

operations; all SPS operation requests for the mandatory operations should conform to the SPS 1.0 specification.

11.4.2.1 *GetCapabilities* Response

The SPS *GetCapabilities* response should adhere to the SPS 1.0 specification. There should be a separate *SensorOffering* for each taskable sensor available through the SPS interface. There should also be a separate *PhenomenonOffering* for each phenomenon that the sensors support.

```
<?xml version="1.0" encoding="UTF-8"?>
<Capabilities version="1.0.0" updateSequence=""
xsi:schemaLocation="http://www.opengis.net/sps ../spsAll.xsd"
xmlns="http://www.opengis.net/sps/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">
    <ows:ServiceIdentification>
        <ows:Title>CCSI SPS</ows:Title>
        <ows:Abstract>Sensor planning service managing one or more CCSI
sensors</ows:Abstract>
        <ows:Keywords>
            <ows:Keyword>CBRN</ows:Keyword>
            <ows:Keyword>CCSI</ows:Keyword>
            <ows:Keyword>Radiological</ows:Keyword>
            <ows:Keyword>Biological</ows:Keyword>
            <ows:Keyword>Nuclear</ows:Keyword>
            <ows:Keyword>Chemical</ows:Keyword>
        </ows:Keywords>
        <ows:ServiceType codeSpace="http://opengeospatial.net">OGC:SPS</ows:ServiceType>
        <ows:ServiceTypeVersion>1.0.0</ows:ServiceTypeVersion>
        <ows:Fees>NONE</ows:Fees>
        <ows:AccessConstraints>NONE</ows:AccessConstraints>
    </ows:ServiceIdentification>
    <ows:ServiceProvider>
        <ows:ProviderName>SPS Provider</ows:ProviderName>
        <ows:ProviderSite xlink:href="http://www.spsprovider.com"/>
        <ows:ServiceContact>
            <ows:IndividualName>Provider Name</ows:IndividualName>
            <ows:PositionName>Provider Position</ows:PositionName>
            <ows:ContactInfo>
                <ows:Phone>
                    <ows:Voice/>
                    <ows:Facsimile/>
                </ows:Phone>
                <ows:Address>
                    <ows:DeliveryPoint/>
                    <ows:City/>
                    <ows:AdministrativeArea/>
                    <ows:PostalCode/>
                    <ows:Country/>
                    <ows:ElectronicMailAddress/>
                </ows:Address>
            </ows:ContactInfo>
        </ows:ServiceContact>
    </ows:ServiceProvider>
    <Contents>
        <SensorOfferingList>
            <SensorOffering>
                <AreaOfService>
                    <ows:WGS84BoundingBox>
                        <ows:LowerCorner>7.6069360670 51.9599905107</ows:LowerCorner>
                        <ows:UpperCorner>7.6075430075 51.9603508343</ows:UpperCorner>
                    </ows:WGS84BoundingBox>
                </AreaOfService>
                <Phenomenon>urn:ogc:def:phenomenon:JPEO-CBD::READGS</Phenomenon>

    <SensorDefinition>http://www.spsurl.com/DescribeSensor/CAD 1</SensorDefinition>
                <SensorID>urn:ogc:def:procedure:JPEO-CBD::CAD_1</SensorID>
```



```
            </SensorOffering>
            <SensorOffering>
                <AreaOfService>
                    <ows:WGS84BoundingBox>
                        <ows:LowerCorner>7.6059360670 51.9599905107</ows:LowerCorner>
                        <ows:UpperCorner>7.6065430075 51.9603508343</ows:UpperCorner>
                    </ows:WGS84BoundingBox>
                </AreaOfService>
                <Phenomenon>urn:ogc:def:phenomenon:JPEO-CBD::READGS</Phenomenon>

    <SensorDefinition>http://www.spsurl.com/DescribeSensor/CAD 2</SensorDefinition>
                <SensorID>urn:ogc:def:procedure:JPEO-CBD::CAD_2</SensorID>
            </SensorOffering>
        </SensorOfferingList>
        <PhenomenonOfferingList>
            <PhenomenonOffering>
                <Phenomenon>urn:ogc:def:phenomenon:JPEO-CBD::READGS</Phenomenon>
                <SensorID>urn:ogc:def:procedure:JPEO-CBD::CAD_1</SensorID>
                <SensorID>urn:ogc:def:procedure:JPEO-CBD::CAD_2</SensorID>
            </PhenomenonOffering>
        </PhenomenonOfferingList>
    </Contents>
</Capabilities>
```

11.4.2.1.1 Keywords

As in the case of the CCSI SOS, the CCSI SPS should include an OWS Common *Keywords* element that includes individual *Keyword* elements indicating that the SPS provides access to task CCSI sensors.

11.4.2.1.2 Contents – *SensorOfferingList*

The *SensorOfferingList* element should include child *SensorOffering* elements for each taskable sensor offered by the CCSI SPS. The *SensorOffering*'s *AreaOfService* element should include a child OWS Common *WGS84BoundingBox* element that defines the extents of where that sensor has been. In the in-situ case, the *WGS84BoundingBox* element's *LowerCorner* and *UpperCorner* elements should be the same value that corresponds to the fixed location of the sensor. Note that the *WGS84BoundingBox* element value should be the longitude value followed by a space followed by the latitude value, rather than latitude followed by a space followed by longitude as is typically the case for most GML coordinate types that refer to EPSG coordinate reference systems (e.g. 4326). If the sensor is mobile, then the *LowerCorner* should be the geographic minimum extent (e.g. minimum latitude and longitude) and the *UpperCorner* should be the geographic maximum extent (e.g. maximum latitude and longitude) for where that sensor has been.

The *Phenomenon* element should reference an OGC definition URN (e.g. urn:ogc:def:phenomenon:JPEO-CBD::READGS) for each CCSI channel that the sensor supports (discovered through the CCSI Sensor Definition file).

The *SensorDefinition* element should include a URL that resolves to a SensorML 1.0.1 definition for the sensor that complies with the SensorML profile defined in this document.

11.4.2.1.3 Contents – *PhenomenonOfferingList*

The *PhenomenonOfferingList* element should include a separate *PhenomenonOffering* for each unique CCSI channel (e.g. urn:ogc:def:phenomenon:JPEO-CBD::READGS) supported by the sensor(s) advertised by the CCSI SPS instance. Each sensor that supports a specific CCSI channel should be listed as a SensorID element beneath that *PhenomenonOffering*.

11.4.2.2 *DescribeTasking* Response

In response to a valid *DescribeTasking* request, a CCSI SPS should return a valid *DescribeTasking* response document (as specified in the SPS 1.0 schemas) that includes multiple *taskingDescriptor* elements. CCSI commands are clearly defined as sets of parameters; providing multiple *taskingDescriptor* elements in the *DescribeTasking* response supports this set of parameters functionality. Each CCSI command should map to a *taskingDescriptor* element with one or more input parameters. The following XML snippet illustrates a *taskingDescriptor* element for the CCSI Start_Self_Test command.

```
<taskingDescriptor>
    <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
    <gml:description>Start Self Test Command:   This command initiates self test on all
or selected CCSI components. The command also
      allows selection of a particular level of test to be performed. </gml:description>
    <InputDescriptor parameterID="Start_Self_Test-Component" updateable="true"
use="required">
        <gml:description> Select the CCSI component to be tested. </gml:description>
        <definition>
            <commonData>
                <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Start_Self_Test:Component">
                    <swe:constraint>
                        <swe:AllowedTokens>
                            <swe:valueList>CC</swe:valueList>
                            <swe:valueList>PDIL</swe:valueList>
                            <swe:valueList>SC</swe:valueList>
                            <swe:valueList>UIC</swe:valueList>
                            <swe:valueList>WCC</swe:valueList>
                            <swe:valueList>DPC</swe:valueList>
                        </swe:AllowedTokens>
                    </swe:constraint>
                </swe:Category>
            </commonData>
        </definition>
    </InputDescriptor>
</taskingDescriptor>
```

CCSI SPS implementers should consider using the XSLT translation found in Annex C Section C.4 in this document for mapping CCSI Standard Commands to an SPS 1.0.0 DescribeTasking response. For best results, CCSI Standard Commands that could affect the performance of SWE web services acting as controlling applications of CCSI sensors should be omitted from the set of *taskingDescriptor* elements advertised in the *DescribeTasking* response. For instance, the CCSI *Register* command should be omitted from the list of *taskingDescriptor* elements, since this may affect the performance of an SOS connected to one or more CCSI sensors (e.g. how often channel messages are sent or for what channels the SOS is registered).

Some CCSI commands support repeatable parameters. This can be handled by an SPS in two ways. The first way of handling repeatable parameters is for the SPS to advertise a repeatable parameter in the same way as a non-repeatable parameter in the response to a *DescribeTasking* request. Using this approach, the SPS client would need to send multiple tasking requests to the SPS in order to repeat the tasking request with multiple options. In the case of the CCSI *Register* command, the SPS client would send a *Submit* request associated with a CCSI *Register* request to the SPS for each channel to which the client wanted to subscribe. This approach is simple but requires multiple requests, which may or may not be desirable. The second way of handling repeatable parameters is by encoding a set of repeatable parameters as a SWE Common *DataArray* with a SWE Common *DataRecord elementType* and a *TextEncoding*. Each *field* in the *DataRecord* would correspond to a parameter in the set of repeatable parameters for a given command. This would inform the SPS client that the Submit request for that particular command should contain one or more values.

11.4.2.3 *Submit* Response

In response to a valid *Submit* request, a CCSI SPS should return a *Submit* response that is valid according to the SPS 1.0.0 schema.

11.4.2.4 *DescribeResultAccess* Response

In response to a valid *DescribeResultAccess* request, a CCSI SPS should return a valid *DescribeResultAccess* response document that provides information about the CCSI SOS where associated sensor data can be retrieved. The *ServiceType* element should provide a value of "SOS", and the *ServiceURL* element should provide a URL that points to a CCSI SOS instance.

```
<?xml version="1.0" encoding="UTF-8"?>
<DescribeResultAccessRequestResponse xmlns="http://www.opengis.net/sps/1.0">
    <service>
        <ServiceType>SOS</ServiceType>
        <ServiceURL>http://www.ccsiproviderurl.com/CCSISOS/SOS</ServiceURL>
    </service>
</DescribeResultAccessRequestResponse>
```

11.4.2.5 SPS 2.0

The SPS 2.0 specification has been released publicly through the OGC RFC process. SPS 2.0 appears to support CCSI command messages as sets of parameters through the use of SWE Common 2.0 *DataChoice* elements. The SPS 2.0 specification does not currently provide examples of using the *DataChoice* element to define options for different tasking parameters, but the SPS 2.0 schemas support the use of the *DataChoice* element. Each parameter set could be defined as a separate *DataRecord* within a *DataChoice* in the *DescribeTasking* response, which would inform an SPS client that there are multiple tasking options for the sensor for which the *DescribeTasking* request was issued.

The SPS 2.0 examples (listed verbatim from the SPS 2.0 RFC draft) below illustrate the response to a *DescribeTasking* request and the associated *Submit* request. Note that the examples below do not show the use of a DataChoice element but are intended to

illustrate basic SPS 2.0 functionality. Refer to the SPS 2.0 specification when it is adopted and released publicly for more detailed examples.

```
<sps:DescribeTaskingResponse xsi:schemaLocation="http://www.opengis.net/sps/2.0
../sps.xsd" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:sps="http://www.opengis.net/sps/2.0" xmlns:swe="http://www.opengis.net/swe/2.0"
xmlns:swes="http://www.opengis.net/swes/2.0" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <sps:taskingParameter name="CameraTask">
        <swe:DataRecord>
            <swe:field name="focalLength">
                <swe:Quantity definition="urn:ogc:def:property:OGC::x-FocalLength"
updatable="true">
                    <swe:uom code="mm"/>
                    <swe:constraint>
                        <swe:AllowedValues>
                            <swe:interval>12 180</swe:interval>
                        </swe:AllowedValues>
                    </swe:constraint>
                </swe:Quantity>
            </swe:field>
            <swe:field name="lookingDir">
                <swe:Category definition="urn:x-foo:property:Foo:Direction"
updatable="true">
                    <swe:constraint>
                        <swe:AllowedTokens>
                            <swe:value>North</swe:value>
                            <swe:value>East</swe:value>
                            <swe:value>West</swe:value>
                            <swe:value>South</swe:value>
                        </swe:AllowedTokens>
                    </swe:constraint>
                </swe:Category>
            </swe:field>
        </swe:DataRecord>
    </sps:taskingParameter>
</sps:DescribeTaskingResponse>


<sps:Submit service="SPS" version="2.0.0"
xsi:schemaLocation="http://www.opengis.net/sps/2.0 ../sps.xsd"
xmlns:sps="http://www.opengis.net/sps/2.0" xmlns:swe="http://www.opengis.net/swe/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <sps:sensorIdentifier codeSpace="http://www.my.org">MY-
CAM:0e1ad0a7</sps:sensorIdentifier>
    <sps:taskingParameters>
        <sps:ParameterData>
            <sps:encoding>
                <swe:TextEncoding blockSeparator="@@" tokenSeparator=","
decimalSeparator="."/>
            </sps:encoding>
            <sps:values>22.0,North</sps:values>
        </sps:ParameterData>
    </sps:taskingParameters>
    <sps:latestResponseTime>2009-11-07T20:30:47.0Z</sps:latestResponseTime>
</sps:Submit>
```

12 Next Steps/Future Work

12.1 Expanding OWS-7 Work to Incorporate Other CBRN Standards

This report develops a CBRN profile of the SWE specifications from a CCSI standpoint but may not fully address the needs of the CBRN community with respect to other CBRN standards or use of CBRN sensors outside of a military/DoD environment. One particular example that should be addressed in the future is ANSI N42.42. ANSI N42.42 is a frequently used standard for radiation detectors in US Homeland Security and other

applications. The CCSI spectrum data element supports ANSI N42.42 spectrum information, but the CCSI emulator available during the OWS-7 efforts does not generate spectrum information, so an example CCSI document with ANSI N42.42 data is not available to analyze. However, the examples below provide a general mapping showing how an ANSI N42.42 reading could be represented using O & M 1.0.0. The following ANSI N42.42 example for a simple spectrometer can be found on the ANSI N42.42 website at: http://physics.nist.gov/Divisions/Div846/Gp4/ANSIN4242/2005/annexB.n42

```
<N42InstrumentData>
    <Measurement>
        <Spectrum>
            <StartTime>2003-11-22T23:45:19</StartTime>
            <RealTime>PT60S</RealTime>
            <LiveTime>PT59.61S</LiveTime>
            <Calibration Type="Energy" EnergyUnits="keV">
                <Equation Model="Polynomial">
                    <Coefficients>-21.84 12.105214 0.0065533164771609</Coefficients>
                </Equation>
            </Calibration>
            <ChannelData>
        0 0 0 22 421 847 1295 1982 2127 2222 2302 2276
        2234 1921 1939 1715 1586 1469 1296 1178 1127 1047 928 760
        679 641 542 529 443 423 397 393 322 272 294 227
        216 224 208 191 189 163 167 173 150 137 136 129
        150 142 160 159 140 103 90 82 83 85 67 76
        73 84 63 74 70 69 76 61 49 61 63 65
        58 62 48 75 56 61 46 56 43 37 55 47
        50 40 38 54 43 41 45 51 32 35 29 33
        40 44 33 35 20 26 27 17 19 20 16 19
        18 19 18 20 17 45 55 70 62 59 32 30
        21 23 10 9 5 13 11 11 6 7 7 9
        11 4 8 8 14 14 11 9 13 5 5 6
        10 9 3 4 3 7 5 5 4 5 3 6
        5 0 5 6 3 1 4 4 3 10 11 4
        1 4 2 11 9 6 3 5 5 1 4 2
        6 6 2 3 0 2 2 2 2 0 1 3
        1 1 2 3 2 4 5 2 6 4 1 0
        3 1 2 1 1 0 1 0 0 2 0 1
        0 0 0 1 0 0 0 0 0 0 0 2
        0 0 0 1 0 1 0 0 2 1 0 0
        0 0 1 3 0 0 0 1 0 1 0 0
        0 0 0 0
      </ChannelData>
        </Spectrum>
    </Measurement>
</N42InstrumentData>
```

The ANSI N42.42 example above could be expressed as an O & M 1.0.0 document with SWE Common encoding. ANSI N42.42 appears to mix sensor metadata (e.g. calibration information) with sensor readings, which would most likely necessitate the use of both SensorML and O & M in a SWE context. An O & M document can include an optional *parameter* element that defines parameters of an observation. This would be the best place for encoding metadata like calibration information that varies frequently or on an observation to observation basis. The O & M 1.0.0 specification defines the O & M parameter element as follows:

An Observation **parameter** is a general event-specific parameter. This will typically be used to record environmental parameters, or event-specific sampling parameters that are not tightly bound to either the feature-of-interest or the procedure.

NOTE: Parameters that are tightly bound to the procedure should be recorded as part of the procedure description. For example, the SensorML model associates parameters with specific process elements or stages.

NOTE: The semantics of the parameter must be provided as part of its value.

In some applications it is convenient to use a generic or standard procedure, or feature-of-interest, rather than define an event-specific process or feature. In this context, event-specific parameters are bound to the Observation act.

Example: A time sequence of observations of water quality in a well may be made at variable depths within the well. While a comprehensive model may identify a specimen from the well taken at this depth as the feature-of-interest, a more common approach is to identify the well itself as the feature-of-interest, and add a samplingDepth parameter to the observation. The sampling depth is of secondary interest relative to the temporal variation of water quality at the site.

```
<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector_2_96359"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.cbrn-ccsi.org/schema/CCSI">
    <gml:description>ANSI N42.42 Sensor Observation Instance</gml:description>
    <gml:name>ANSI N42.42 Observation</gml:name>
    <samplingTime>
        <gml:TimePeriod>
            <gml:beginPosition>2003-11-22T23:45:19Z</gml:beginPosition>
            <gml:endPosition>2003-11-22T23:46:09Z</gml:endPosition>
        </gml:TimePeriod>
    </samplingTime>
    <resultTime>
        <gml:TimePeriod>
            <gml:beginPosition>2003-11-22T23:45:19Z</gml:beginPosition>
            <gml:endPosition>2003-11-22T23:46:08.61Z</gml:endPosition>
        </gml:TimePeriod>
    </resultTime>
    <procedure xlink:href="RadDetector_2"/>
    <observedProperty xlink:href="urn:ogc:def:phenomenon:ANSI::spectrumMeasurement"/>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>ANSI N42.42 Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">18.556491 -
72.297463</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <parameter>
        <swe:Curve
definition="urn:ogc:def:curve:ANSI::energyCalibrationCurve:polynomial">
            <swe:elementCount>
                <swe:Count>
                    <swe:value>3</swe:value>
                </swe:Count>
            </swe:elementCount>
            <swe:elementType>
```



```
                <swe:SimpleDataRecord>
                    <swe:field name="coefficient">
                        <swe:Quantity
definition="urn:ogc:def:property:ANSI::energyCoefficient">
                            <swe:uom code="keV"/>
                        </swe:Quantity>
                    </swe:field>
                </swe:SimpleDataRecord>
            </swe:elementType>
            <swe:encoding>
                <swe:TextBlock decimalSeparator="." blockSeparator="@" tokenSeparator="
"/>
            </swe:encoding>
            <swe:values>-21.84 12.105214 0.0065533164771609</swe:values>
        </swe:Curve>
    </parameter>
    <result>
        <swe:DataArray definition="urn:ogc:def:phenomenon:ANSI::spectrumMeasurement">
            <swe:elementCount>
                <swe:Count>
                    <swe:value>256</swe:value>
                </swe:Count>
            </swe:elementCount>
            <swe:elementType name="ANSIN4242Spectrum">
                <swe:Count definition="urn:ogc:def:phenomenon:ANSI::channelCount"/>
            </swe:elementType>
            <swe:encoding>
                <swe:TextBlock decimalSeparator="." blockSeparator="@" tokenSeparator="
"/>
            </swe:encoding>
            <swe:values>
0 0 0 22 421 847 1295 1982 2127 2222 2302 2276 2234 1921 1939 1715 1586 1469 1296 1178
1127 1047 928 760 679 641 542 529 443 423 397 393 322 272 294 227 216 224 208 191 189 163
167 173 150 137 136 129 150 142 160 159 140 103 90 82 83 85 67 76 73 84 63 74 70 69 76 61
49 61 63 65 58 62 48 75 56 61 46 56 43 37 55 47 50 40 38 54 43 41 45 51 32 35 29 33 40 44
33 35 20 26 27 17 19 20 16 19 18 19 18 20 17 45 55 70 62 59 32 30 21 23 10 9 5 13 11 11 6
7 7 9 11 4 8 8 14 14 11 9 13 5 5 6 10 9 3 4 3 7 5 5 4 5 3 6 5 0 5 6 3 1 4 4 3 10 11 4 1 4
2 11 9 6 3 5 5 1 4 2 6 6 2 3 0 2 2 2 2 0 1 3 1 1 2 3 2 4 5 2 6 4 1 0 3 1 2 1 1 0 1 0 0 2
0 1 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 1 0 1 0 0 2 1 0 0 0 0 1 3 0 0 0 1 0 1 0 0 0 0 0 0
            </swe:values>
        </swe:DataArray>
    </result>
</Observation>
```

When considering the use of ANSI N42.42 with SWE, it is also important to consider what information in an ANSI N42.42 message corresponds to static metadata about the sensor and what information corresponds to metadata about the particular reading within the ANSI N42.42 document, as this will determine what information needs to go in a SensorML description and what information needs to go in an O & M document. Further effort should be applied to mapping between ANSI N42.42 and SWE.

12.2 Future Directions/Recommendations

JPEO-CBD should consider incorporating SWE web services and encodings within the next version of CCSI. The profiles and examples defined in this report can serve as the starting point for creating a version of CCSI based on SWE. Additional work would be required to map channel data that is not currently provided by the emulator to SWE formats, but that should not necessitate a great deal of effort. This effort would undoubtedly require CBRN community involvement in determining the best approach for encoding CBRN sensor descriptions and data using SWE and for resolving any details that are not clear or well defined (e.g. CBRN community domain features). In addition, the OGC needs CBRN community involvement in refining the profiles defined in this

document. Perhaps the OGC should consider creating a CBRN Domain Working Group or Standards Working Group to manage CBRN profiles of the SWE specifications as well as any other items of importance. It may also make sense to establish a CBRN-focused interoperability experiment (IE) within the OGC. This IE could involve participation from CBRN community members and could examine and test areas like performance and security as well as the profiles defined in this document, perhaps with real CBRN sensors.

One of the barriers to entry with SWE is the fact that there are multiple service interfaces that need to be reviewed and understood before SWE can be fully understood, which leads to confusion. In particular, those new to SWE are often confused as to why different service interfaces need to be implemented for tasking versus observation retrieval versus alerting. The OGC should consider defining a general sensor web service interface for use in SWE rather than having multiple service interfaces for providing different sets of functionality (e.g. an SOS for observation retrieval and an SPS for sensor tasking). The sensor service Capabilities document could describe which operations are supported and could include operations from the SOS, SPS, and other SWE services. This idea has grown out of the Sensor Interface Service concept that provides common access to a set of sensors and would allow SWE service implementers to implement a single interface for observation retrieval, notifications, and tasking that would be particularly useful with taskable sensors. The sensor service interface idea has been brought up and discussed among some of the members of the SWE WGs but has not been formally considered, since it may involve a major shift in the SWE specifications. However, the use of SWE Common 2.0 for data encoding and a common SWE Service Model 2.0 for SWE web services in the 2.0 versions of the SWE specifications may make this recommendation unnecessary.

Since the SOS 2.0 specification is still being revised, it may make sense to review the latest specification to determine whether the Transactional profile, and more specifically, the *RegisterSensor* operation supports sending multiple outputs to the server to support the multiple channel functionality that CCSI provides. This may be encoded using a SWE Common 2.0 *DataChoice* element but warrants further investigation. It may also be worthwhile to consider whether an O & M template as part of the *RegisterSensor* operation is really needed at all in most cases. If the *outputs* section of the SensorML document describing a sensing system is defined in detail, then most likely, each *output* should include a SWE Common type that will match the *result* element found in an O & M document for that *output*. An SOS server would need to parse the SensorML document that is sent to the server and read each *output* in the *outputs* element to understand what the outputs for that sensor will look like, which should be enough information for an SOS server to understand the format of the data that will be inserted during an *InsertObservation* operation. For cases where SWE Common is not used within O & M, there may still be a need to register an observation template.

13 Conclusion

This report describes and analyzes CCSI and SWE specifications and recommends best practices for integrating CCSI sensors and data into a SWE-based architecture. Based on

the examination conducted in this report, there appears to be functional overlap between CCSI and SWE. This overlap should be considered by the CBRN community to determine the best approach for future CBRN sensor standards and whether or not there is a need for potentially competing standards. The report also uses the information from prior work in OWS-6 and work conducted during OWS-7 to define CCSI profiles of SWE that can be used in the future as a basis for a SWE-based version of CCSI.

Annex A

CCSI Examples

A.1 Introduction

This annex provides example SWE documents based on the CCSI profile of SWE defined in section 11.3. Note that SAS examples have been omitted given the uncertainty of SAS use in the future. Refer to Annex B in the OWS-6 CCSI-SWE ER for SAS examples.

A.2 Sensor Definition (Radiation Detector)

```
<?xml version="1.0" encoding="UTF-8"?>
<SensorDefinition xmlns="http://www.cbrn-ccsi.org/schema/CCSI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cbrn-ccsi.org/schema/CCSI
file:/C:/Tom/CCSI/XML/CCSI_XML_Sensor.01.00.xsd">
   <Sensor xmlns="">
      <Manufacturer>
         <PointOfContactNameText>Tim Taylor</PointOfContactNameText>
         <PointOfContactJobTitleNameText>VP Research and
Development</PointOfContactJobTitleNameText>
         <PointOfContactAddressLineText>Binford Sensors,
Inc.</PointOfContactAddressLineText>

   <PointOfContactCityNameText>Detroit</PointOfContactCityNameText>
         <PointOfContactStateNameText>MI</PointOfContactStateNameText>

   <PointOfContactWebPageAddressText>http://www.binford.com</PointOfCon
tactWebPageAddressText>
      </Manufacturer>
      <Program>
         <PointOfContactNameText>Albert
Borland</PointOfContactNameText>
         <PointOfContactJobTitleNameText>Program
Manager</PointOfContactJobTitleNameText>

   <PointOfContactMilitaryRankCode>O6</PointOfContactMilitaryRankCode>
         <PointOfContactOfficeNameText>JPM NBC
CA</PointOfContactOfficeNameText>

   <PointOfContactEmailAddressText>albert.borland@us.army.mil</PointOfC
ontactEmailAddressText>
         <PointOfContactAgencyAcronymText>JPEO-
CBD</PointOfContactAgencyAcronymText>
         <PointOfContactAddressLineText>Bldg
4433</PointOfContactAddressLineText>

   <PointOfContactCityNameText>Edgewood</PointOfContactCityNameText>
         <PointOfContactStateNameText>MD</PointOfContactStateNameText>
```



```
       </Program>
       <SensorIdentification>
          <SensorName>Binford Radiation Monitor</SensorName>
          <SensorType class="RAD" variant="PNT" name="SC001"/>
          <SensorModel>Model 6700</SensorModel>
          <SensorDescription>The more power rad
sensor.</SensorDescription>
       </SensorIdentification>
       <SensorDescription>
      This sensor detects gamma and neutrons and maintain personal
safety information.
    </SensorDescription>
   </Sensor>
   <DataDefinitions xmlns="">
       <DataElement name="CummDose" type="float">
          <UnitOfMeasure>cGy</UnitOfMeasure>
          <ValidityCheck>
             <Range>0.0 - 999.99</Range>
          </ValidityCheck>
          <HelpText>
        This is the cummulative dose received since the last reset.
      </HelpText>
       </DataElement>
       <DataElement name="DoseRate" type="float">
          <UnitOfMeasure>cGy/hr</UnitOfMeasure>
          <ValidityCheck>
             <Range>0.0 - 999.99</Range>
          </ValidityCheck>
          <HelpText>
        This is the rate of radiation increase averaged over one hour.
      </HelpText>
       </DataElement>
       <DataElement name="PeakRate" type="float">
          <UnitOfMeasure>cGy/hr</UnitOfMeasure>
          <ValidityCheck>
             <Range>0.0 - 999.99</Range>
          </ValidityCheck>
       </DataElement>
       <DataElement name="ClearType" type="enum">
          <ValidityCheck>
             <Enumeration>
                <Item>All</Item>
                <Item>Cumm</Item>
                <Item>Rate</Item>
                <Item>Peak</Item>
             </Enumeration>
          </ValidityCheck>
       </DataElement>
       <DataElement name="RateProfile" type="array">
          <ArrayDef nbr_dims="2" row_first="true">
             <ArrayDefinitionTypeDescription>
          This array provides a plot of the dose rate over time for the
last 24 hours.
        </ArrayDefinitionTypeDescription>

   <ArrayDefinitionTypeElementType>float</ArrayDefinitionTypeElementTyp
e>
```



```
            <ArrayDefinitionTypeDimensions order="1" name="Rate"
max_elements="24"/>
            <ArrayDefinitionTypeDimensions order="2" name="Time"
max_elements="24"/>
         </ArrayDef>
      </DataElement>
   </DataDefinitions>
   <SensorCommands xmlns="">
      <SensorUniqueCmd name="ConfTest" ack_required="false"
response="STATUS" roles="ANY">
         <DisplayName>ConfidenceTest</DisplayName>
         <HelpText>
        This command initiates the sensor confidence test that checks
for correct operation.
      </HelpText>
         <SensorMnemonic>CT</SensorMnemonic>
         <TestDefinition>
            <TestCase name="NormalTest" type="NORMAL">
               <Description>
           This test case tests normal operation of the command.
         </Description>
                <ExpectedResults>
                   <NoACK/>
                   <Channel name="STATUS" timeout="7"/>
                </ExpectedResults>
            </TestCase>
         </TestDefinition>
      </SensorUniqueCmd>
      <SensorUniqueCmd name="Reset" ack_required="true" roles="PRIV">
         <Arguments>
            <Argument name="Which" type="ClearType" mandatory="false">
               <DefaultValue>All</DefaultValue>
            </Argument>
         </Arguments>
         <HelpText>
          This command resets the sensor readings to zero.
      </HelpText>
         <TestDefinition>
            <TestCase name="NormalTestNoArg" type="NORMAL">
               <Description>
           This tests the normal operation of the command when no
argument is provided.
         </Description>
                <ExpectedResults>
                   <ACK/>
                </ExpectedResults>
            </TestCase>
            <TestCase name="NormalTestArg" type="NORMAL">
               <Description>
           This tests the command using an argument.
         </Description>
                <Argument ArgName="Which" ArgValue="Rate"/>
                <ExpectedResults>
                   <ACK/>
                </ExpectedResults>
            </TestCase>
            <TestCase name="BadArgName" type="FAIL">
               <Description>
```



```
          This tests an invalid argument name.
        </Description>
               <Argument ArgName="Witch" ArgValue="Rate"/>
               <ExpectedResults>
                  <NAK code="Msg"/>
               </ExpectedResults>
            </TestCase>
            <TestCase name="BadArgValue" type="FAIL">
               <Description>
          This tests an invalid argument value.
        </Description>
               <Argument ArgName="Which" ArgValue="None"/>
               <ExpectedResults>
                  <NAK code="Msg"/>
               </ExpectedResults>
            </TestCase>
        </TestDefinition>
    </SensorUniqueCmd>
</SensorCommands>
<SensorConfig xmlns="">
    <ConfigItem>
        <ItemName>MissionStartTime</ItemName>
        <ItemType>date_time</ItemType>
        <ItemValue/>
        <ItemAccessRole>ANY</ItemAccessRole>
        <ItemModifyRole>PRIV</ItemModifyRole>
    </ConfigItem>
</SensorConfig>
<SensorVersion xmlns="">
    <CcsiVersion major="1" moderate="0"/>
    <ReleaseVersion>
        <SensingUnit>SC001</SensingUnit>
        <HwVersion major="1" moderate="5"/>
        <SwVersion major="3" moderate="1" minor="0"/>
        <Description>
      This is the first DOD issued version of the sensor.
    </Description>
    </ReleaseVersion>
    <Capability SensingUnit="SC001">
        <Capability>
            <ItemType>Gamma</ItemType>
            <ItemSpecific>CummDose</ItemSpecific>
            <LevelUnits>cGy</LevelUnits>
        </Capability>
        <Capability>
            <ItemType>Gamma</ItemType>
            <ItemSpecific>DoseRate</ItemSpecific>
            <LevelUnits>cGy/hr</LevelUnits>
        </Capability>
        <Capability>
            <ItemType>Neutron</ItemType>
            <ItemSpecific>Rate</ItemSpecific>
            <LevelUnits>cnt/min</LevelUnits>
        </Capability>
    </Capability>
    <Channels>
        <Channel Channel="ALERTS"/>
        <Channel Channel="HRTBT"/>
```

```
          <Channel Channel="STATUS"/>
          <Channel Channel="CMD"/>
          <Channel Channel="CONFIG"/>
          <Channel Channel="IDENT"/>
          <Channel Channel="READGS" HasMetaData="true"
HasHighResData="true">
             <Metadata>CummDose</Metadata>
             <Metadata>DoseRate</Metadata>
             <Metadata>PeakRate</Metadata>
             <Metadata>RateProfile</Metadata>
          </Channel>
       </Channels>
       <DataUsed>
          <Uses>CummDose</Uses>
          <Uses>DoseRate</Uses>
          <Uses>PeakRate</Uses>
          <Uses>RateProfile</Uses>
       </DataUsed>
       <SensorCommandUsed>
          <Uses>ConfTest</Uses>
          <Uses>Reset</Uses>
       </SensorCommandUsed>
       <CcsiCmdSupport>
          <Supports>Register</Supports>
          <Supports>Deregister</Supports>
          <Supports>Clear_Alert</Supports>
          <Supports>Get_Configuration</Supports>
          <Supports>Set_Config</Supports>
          <Supports>Get Identification</Supports>
          <Supports>Get_Reading_Details</Supports>
          <Supports>Get_Status</Supports>
          <Supports>Power_Off</Supports>
          <Supports>Reboot</Supports>
          <Supports>Render_Useless</Supports>
          <Supports>Set_Bw_Mode</Supports>
          <Supports>Reset_Msg_Seq</Supports>
          <Supports>Set_Ccsi_Mode</Supports>
          <Supports>Set_Date_Time</Supports>
          <Supports>Set_Heartbeat</Supports>
          <Supports>Start_Self_Test</Supports>
          <Supports>Start_Stop</Supports>
       </CcsiCmdSupport>
    </SensorVersion>
</SensorDefinition>
```

A.3 Channel Data (Radiation Detector)

Sensor Alert Message

```
<Hdr sid="1" msn="42" dtg="916543" len="311" mod="N" chn="a">
    <CCSIDoc xmlns="">
       <Msg>
          <AlertsChn Event="ALERT" Source="detector"
Time="20080215123455" AlertId="AT000000001">
             <Reading Sensor="SC001" Time="20080215123451">
                <Data Detect="Radiation" Id="CummDose" Level="107.2"
Units="cGy"/>
```



```
            </Reading>
         </AlertsChn>
      </Msg>
   </CCSIDoc>
</Hdr>
```

A.4 Command Message

```
<Hdr sid="1" msn="3" len="902" chn="c" dtg="7654321">
   <CCSIDoc xmlns="">
      <Msg>
         <CmdChn Cmd="Clear_Alert">
            <Arg>
               <ArgName>Alert_ID</ArgName>
               <ArgValue> AT000000001</ArgValue>
            </Arg>
         </CmdChn>
      </Msg>
   </CCSIDoc>
</Hdr>
```

Annex B

SWE Samples

B.1 Introduction

This annex provides example SWE documents based on the CCSI profile of SWE defined in section 11.3. Note that SAS examples have been omitted given the uncertainty of SAS use in the future. Refer to Annex B in the OWS-6 CCSI-SWE ER for SAS examples.

B.2 SensorML 1.0.1 (Radiation Detector)

```
<?xml version="1.0" encoding="utf-8"?>
<System gml:id="CCSI_Sensor" xsi:schemaLocation="http://www.opengis.net/sensorML/1.0.1
http://schemas.opengis.net/sensorML/1.0.1/sensorML.xsd"
xmlns="http://www.opengis.net/sensorML/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:om="http://www.opengis.net/om/1.0" xmlns:gml="http://www.opengis.net/gml"
xmlns:tml="http://www.opengis.net/tml" xmlns:swe="http://www.opengis.net/swe/1.0.1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ccsi="http://www.cbrn-
ccsi.org/schema/CCSI">
    <gml:description>
      AN/UDR-14 RADIAC radiological detector.
    </gml:description>
    <keywords>
        <KeywordList>
            <keyword>CCSI</keyword>
            <keyword>CBRN</keyword>
            <keyword>Sensor</keyword>
        </KeywordList>
    </keywords>
    <identification>
        <IdentifierList>
            <identifier name="uniqueID">
                <Term definition="urn:ogc:def:identifier:OGC:uniqueID">
                    <value>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</value>
                </Term>
            </identifier>
            <identifier name="SensorName">
                <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorName">
                    <value>AN/UDR-14</value>
                </Term>
            </identifier>
            <identifier name="class">
                <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorType:class">
                    <value>RAD</value>
                </Term>
            </identifier>
            <identifier name="variant">
                <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorType:variant">
                    <value>PNT</value>
                </Term>
            </identifier>
            <identifier name="name">
                <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorType:name">
                    <value>SC001</value>
                </Term>
            </identifier>
            <identifier name="SensorModel">
                <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorModel">
```



```
                    <value>TBD</value>
                </Term>
            </identifier>
            <identifier name="SensorDescription">
                <Term definition="urn:ogc:def:identifier:JPEO-CBD::SensorDescription">
                    <value>
         Radiation detector adapted to support CCSI within FCS platforms.
      </value>
                </Term>
            </identifier>
        </IdentifierList>
    </identification>
    <classification>
        <ClassifierList>
            <classifier name="intendedApplication">
                <Term definition="urn:ogc:def:classifier:OGC:application">
                    <value>CBRN Detection</value>
                </Term>
            </classifier>
            <classifier name="sensorType">
                <Term definition="urn:ogc:def:classifier:OGC:sensorType">
                    <value>Radiological</value>
                </Term>
            </classifier>
            <classifier name="sensorType">
                <Term definition="urn:ogc:def:classifier:OGC:sensorType">
                    <value>Point</value>
                </Term>
            </classifier>
        </ClassifierList>
    </classification>
    <characteristics name="HW/SW Characteristics">
        <swe:DataRecord definition="urn:ogc:def:property:hwswCharacteristics">
            <swe:field name="CCSI Version">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::CCSIVersion">
                    <swe:value>1.0.0.0</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="HW Version">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::HWVersion">
                    <swe:value>1.0</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="SW Version">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::SWVersion">
                    <swe:value>1.0</swe:value>
                </swe:Category>
            </swe:field>
        </swe:DataRecord>
    </characteristics>
    <capabilities name="Gamma Radiation Dose Measurement Properties">
        <swe:DataRecord definition="urn:ogc:def:property:JPEO-
CBD::capabilityInformation">
            <swe:field name="itemType">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::itemType">
                    <swe:value>Gamma Radiation Dose</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="levelUnits">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::levelUnits">
                    <swe:value>cGy</swe:value>
                </swe:Category>
            </swe:field>
        </swe:DataRecord>
    </capabilities>
    <capabilities name="Gamma Radiation Dose Rate Measurement Properties">
        <swe:DataRecord definition="urn:ogc:def:property:JPEO-
CBD::capabilityInformation">
            <swe:field name="itemType">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::itemType">
                    <swe:value>Gamma Radiation Dose Rate</swe:value>
                </swe:Category>
```

```
            </swe:field>
            <swe:field name="levelUnits">
                <swe:Category definition="urn:ogc:def:property:JPEO-CBD::levelUnits">
                    <swe:value>cGy/hr</swe:value>
                </swe:Category>
            </swe:field>
        </swe:DataRecord>
    </capabilities>
    <contact>
        <ContactList>
            <member xlink:role="urn:ogc:def:identifier:OGC::manufacturer">
                <ResponsibleParty>
                    <individualName>Jiri Pesek</individualName>
                    <contactInfo>
                        <address>
                            <deliveryPoint>Canberra Dover</deliveryPoint>
                            <city>Dover</city>
                            <administrativeArea>NJ</administrativeArea>
                        </address>
                    </contactInfo>
                </ResponsibleParty>
            </member>
            <member xlink:role="urn:ogc:def:identifier:OGC::program">
                <ResponsibleParty>
                    <individualName>Mr. Program Manager</individualName>
                    <organizationName>JPEO-CBD</organizationName>
                    <contactInfo>
                        <address>

    <electronicMailAddress>programmanager@agency.gov</electronicMailAddress>
                        </address>
                    </contactInfo>
                </ResponsibleParty>
            </member>
        </ContactList>
    </contact>
    <spatialReferenceFrame>
        <gml:EngineeringCRS gml:id="SENSOR_FRAME">
            <gml:srsName>CCSI Sensor Spatial Frame</gml:srsName>
            <gml:usesCS xlink:href="urn:ogc:def:cs:xyzFrame"/>
            <gml:usesEngineeringDatum>
                <gml:EngineeringDatum gml:id="SENSOR DATUM">
                    <gml:datumName>CCSI Sensor Spatial Datum</gml:datumName>
                    <gml:anchorPoint>Unknown<gml:anchorPoint>
                </gml:EngineeringDatum>
            </gml:usesEngineeringDatum>
        </gml:EngineeringCRS>
    </spatialReferenceFrame>
    <interfaces/>
    <inputs>
        <InputList>
            <input name="genericCBRNHazard">
                <swe:ObservableProperty definition="urn:ogc:def:phenomenon:JPEO-
CBD::toxicAgent"/>
            </input>
        </InputList>
    </inputs>
    <outputs>
        <OutputList>
            <output name="ALERTS">
                <swe:DataRecord gml:id="ALERTS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS">
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="ALERTS_Event">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Event">
                            <swe:constraint>
                                <swe:AllowedTokens>
```



```
                                    <swe:valueList>ALERT DEALERT WARN DEWARN
NONE</swe:valueList>
                                </swe:AllowedTokens>
                            </swe:constraint>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="ALERTS Source">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Source">
                            <swe:constraint>
                                <swe:AllowedTokens>
                                    <swe:valueList>none detector bit
tamper</swe:valueList>
                                </swe:AllowedTokens>
                            </swe:constraint>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="ALERTS AlertId">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:AlertId"/>
                    </swe:field>
                    <swe:field name="ALERTS_BIT_Component">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Component">
                            <swe:constraint>
                                <swe:AllowedTokens>
                                    <swe:valueList>CC PDIL SC UIC WCC
DPC</swe:valueList>
                                </swe:AllowedTokens>
                            </swe:constraint>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="ALERTS_BIT_Executed">
                        <swe:Time definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Executed" referenceTime="1970-01-01">
                            <swe:uom code="s"/>
                        </swe:Time>
                    </swe:field>
                    <swe:field name="ALERTS_BIT_Result">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Result">
                            <swe:constraint>
                                <swe:AllowedTokens>
                                    <swe:valueList>PASS FAIL</swe:valueList>
                                </swe:AllowedTokens>
                            </swe:constraint>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="ALERTS BIT Level">
                        <swe:Count definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Level"/>
                    </swe:field>
                    <swe:field name="ALERTS BIT ErrorDescription">
                        <swe:Text definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:ErrorDescription"/>
                    </swe:field>
                    <swe:field name="ALERTS_Tamper">
                        <swe:Text definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Tamper"/>
                    </swe:field>
                    <swe:field name="ALERTS FilteredDoseRate">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:FilteredDoseRate">
                            <swe:uom code="cGy/hr"/>
                            <swe:constraint>
                                <swe:AllowedValues>
                                    <swe:interval>0.0 999.9</swe:interval>
                                </swe:AllowedValues>
                            </swe:constraint>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="ALERTS_MissionDose">
```



```
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:MissionDose">
                            <swe:uom code="cGy"/>
                            <swe:constraint>
                                <swe:AllowedValues>
                                    <swe:interval>0.0 999.9</swe:interval>
                                </swe:AllowedValues>
                            </swe:constraint>
                        </swe:Quantity>
                    </swe:field>
                </swe:DataRecord>
            </output>
            <output name="CONFIG">
                <swe:DataRecord gml:id="CONFIG_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::CONFIG"/>
            </output>
            <output name="HRTBT">
                <swe:DataRecord gml:id="HRTBT_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::HRTBT">
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="DateTimeGroup">
                        <swe:Time definition="urn:ogc:def:phenomenon:JPEO-CBD::Time"
referenceTime="1970-01-01T00:00:00Z">
                            <swe:uom code="s"/>
                        </swe:Time>
                    </swe:field>
                </swe:DataRecord>
            </output>
            <output name="IDENT">
                <swe:DataRecord gml:id="IDENT_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::IDENT"/>
            </output>
            <output name="MAINT">
                <swe:DataRecord gml:id="MAINT_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT">
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="MAINT_Type">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Type"/>
                    </swe:field>
                    <swe:field name="MAINT_BIT_Result">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:BIT:Result"/>
                    </swe:field>
                    <swe:field name="MAINT_BIT_Executed">
                        <swe:Time definition="urn:ogc:data:time:iso8601"/>
                    </swe:field>
                    <swe:field name="MAINT_BIT_Component">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:BIT:Component"/>
                    </swe:field>
                </swe:DataRecord>
            </output>
            <output name="READGS">
                <swe:DataRecord gml:id="READGS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS">
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="READGS_Sensor">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                    </swe:field>
                    <swe:field name="READGS_ReadingID">
```



```
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                            <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="READGS Detect">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                    </swe:field>
                    <swe:field name="READGS_Level">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                    </swe:field>
                    <swe:field name="READGS LevelConfidenceInterval">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                    </swe:field>
                    <swe:field name="READGS_Id">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                    </swe:field>
                    <swe:field name="READGS_DetailsAvailable">
                        <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                    </swe:field>
                    <swe:field name="READGS_FilteredDoseRate">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                            <swe:uom code="cGy/hr"/>
                            <swe:constraint>
                                <swe:AllowedValues>
                                    <swe:interval>0.0 999.9</swe:interval>
                                </swe:AllowedValues>
                            </swe:constraint>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="READGS MissionDose">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                            <swe:uom code="cGy"/>
                            <swe:constraint>
                                <swe:AllowedValues>
                                    <swe:interval>0.0 999.9</swe:interval>
                                </swe:AllowedValues>
                            </swe:constraint>
                        </swe:Quantity>
                    </swe:field>
                </swe:DataRecord>
            </output>
            <output name="STATUS">
                <swe:DataRecord gml:id="STATUS_DATA_RECORD"
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS">
                    <swe:field name="Mode">
                        <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                    </swe:field>
                    <swe:field name="STATUS_BIT">
                        <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:BIT"/>
                    </swe:field>
                    <swe:field name="STATUS_Alert">
                        <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:Alert"/>
                    </swe:field>
                    <swe:field name="STATUS Maint">
                        <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:Maint"/>
                    </swe:field>
                </swe:DataRecord>
            </output>
        </OutputList>
```

```
    </outputs>
    <components/>
    <positions>
        <PositionList>
            <position name="sensorPosition">
                <swe:Position localFrame="#SENSOR_FRAME"
referenceFrame="urn:ogc:def:crs:EPSG:4326">
                    <swe:location>
                        <swe:Vector definition="urn:ogc:def:vector:OGC:location">
                            <swe:coordinate name="latitude">
                                <swe:Quantity axisID="Y"
definition="urn:ogc:def:phenomenon:latitude">
                                    <swe:uom code="deg"/>
                                    <swe:value>35.123456</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                            <swe:coordinate name="longitude">
                                <swe:Quantity axisID="X"
definition="urn:ogc:def:phenomenon:longitude">
                                    <swe:uom code="deg"/>
                                    <swe:value>-70.123456</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                            <swe:coordinate name="altitude">
                                <swe:Quantity axisID="Z"
definition="urn:ogc:def:phenomenon:altitude">
                                    <swe:uom code="m"/>
                                    <swe:value>0.0</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                        </swe:Vector>
                    </swe:location>
                    <swe:orientation>
                        <swe:Vector definition="urn:ogc:def:vector:OGC:orientation">
                            <swe:coordinate name="trueHeading">
                                <swe:Quantity
definition="urn:ogc:def:phenomenon:angleToNorth">
                                    <swe:uom code="deg"/>
                                    <swe:value>25.0</swe:value>
                                </swe:Quantity>
                            </swe:coordinate>
                        </swe:Vector>
                    </swe:orientation>
                </swe:Position>
            </position>
        </PositionList>
    </positions>
    <connections/>
</System>
```

B.3 O & M 1.0.0 (Radiation Detector)

Single Observation

```
<?xml version="1.0" encoding="utf-8"?>
<Observation gml:id="RadDetector_1_119837"
xsi:schemaLocation="http://www.opengis.net/om/1.0&#xD;&#xA;http://schemas.opengis.net/om/
1.0.0/om.xsd" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <gml:description>CCSI Sensor Observation Instance</gml:description>
    <gml:name>CCSI Sensor Observation Instance (RadDetector_1_119837)</gml:name>
    <samplingTime>
        <gml:TimeInstant>
```



```
            <gml:timePosition>2010-05-05T13:48:18Z</gml:timePosition>
        </gml:TimeInstant>
    </samplingTime>
    <procedure xlink:href="urn:ogc:def:procedure:JPEO-CBD::RadDetector_1"/>
    <observedProperty>
        <swe:CompositePhenomenon gml:id="READGS_RadDetector_1_119837" dimension="10">
            <gml:name>CCSI READGS Phenomenon</gml:name>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Sensor"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Detect"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate"/>
            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose"/>
        </swe:CompositePhenomenon>
    </observedProperty>
    <featureOfInterest>
        <sa:SamplingPoint gml:id="sensor SamplingPoint"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"
xmlns:sa="http://www.opengis.net/sampling/1.0">
            <gml:name>CCSI Sensor</gml:name>
            <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
            <sa:position>
                <gml:Point gml:id="sensor_Point">
                    <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">18.556825 -
72.297935</gml:pos>
                </gml:Point>
            </sa:position>
        </sa:SamplingPoint>
    </featureOfInterest>
    <result>
        <swe:DataRecord gml:id="CCSI_READGS_DATA_RECORD">
            <swe:field name="Mode">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                    <swe:value>Normal</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS Sensor">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor">
                    <swe:value>SC001</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS ReadingID">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID">
                    <swe:codeSpace xlink:href="urn:ogc:def:property:JPEO-
CBD::readingIdentificationType"/>
                    <swe:value>R000239709</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS Detect">
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect">
                    <swe:value>Gamma</swe:value>
                </swe:Category>
            </swe:field>
            <swe:field name="READGS_Level">
                <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level">
                    <swe:value>20923.09765625</swe:value>
                </swe:Quantity>
            </swe:field>
            <swe:field name="READGS_Id">
```

```
                <swe:Category definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
            </swe:field>
            <swe:field name="READGS_DetailsAvailable">
                <swe:Boolean definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable">
                    <swe:value>true</swe:value>
                </swe:Boolean>
            </swe:field>
            <swe:field name="Sensor Unique Data">
                <swe:DataRecord definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:SUD">
                    <swe:field name="FilteredDoseRate">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:FilteredDoseRate">
                            <swe:uom code="cGy/hr"/>
                            <swe:value>124.57</swe:value>
                        </swe:Quantity>
                    </swe:field>
                    <swe:field name="MissionDose">
                        <swe:Quantity definition="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:MissionDose">
                            <swe:uom code="cGy"/>
                            <swe:value>20923.098</swe:value>
                        </swe:Quantity>
                    </swe:field>
                </swe:DataRecord>
            </swe:field>
        </swe:DataRecord>
    </result>
</Observation>
```

Multiple Observations (DataArray)

B.4 SPS 1.0.0 DescribeTasking Response

```
<?xml version="1.0" encoding="utf-8"?>
<DescribeTaskingRequestResponse xsi:schemaLocation="http://www.opengis.net/sps/1.0
http://schemas.opengis.net/sps/1.0.0/spsAll.xsd" xmlns="http://www.opengis.net/sps/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:swe="http://www.opengis.net/swe/1.0" xmlns:gml="http://www.opengis.net/gml"
xmlns:ccsi="http://www.cbrn-ccsi.org/schema/CCSI">
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
        <gml:description>Render Useless Command:   This command initiates the actions
required to render the sensor useless. It erases its
      configuration, cryptographic information, network communications information, etc.
</gml:description>
        <InputDescriptor parameterID="Render_Useless-TamperCount" updateable="true"
use="optional">
            <definition>
                <commonData>
                    <swe:Quantity definition="urn:ogc:def:parameter:JPEO-
CBD::Render_Useless:TamperCount"/>
                </commonData>
            </definition>
        </InputDescriptor>
    </taskingDescriptor>
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
        <gml:description>Reset_Msg_Seq Command:   This command resets the sensor's next
to send and expected receive message sequence
      numbers to 1. This allows a resynch when a sequence number mismatch occurs.
</gml:description>
        <InputDescriptor parameterID="Reset_Msg_Seq" updateable="false" use="required">
            <gml:description>This is a default parameter</gml:description>
            <definition>
```



```
                <commonData>
                    <swe:Boolean>
                        <swe:value>true</swe:value>
                    </swe:Boolean>
                </commonData>
            </definition>
        </InputDescriptor>
    </taskingDescriptor>
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
        <gml:description>Start_Stop Command:   This command starts sensor operation if it
has been stopped or stops it. Stopping the
      sensor operation leaves the sensor communicating, but not performing or reporting
sensing actions. </gml:description>
        <InputDescriptor parameterID="Start_Stop-StartOrStop" updateable="true"
use="required">
            <gml:description> Select the operation to be performed to either Start
sensing operations or Stop them.
        </gml:description>
            <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Start_Stop:StartOrStop">
                        <swe:constraint>
                            <swe:AllowedTokens>
                                <swe:valueList>START</swe:valueList>
                                <swe:valueList>END</swe:valueList>
                            </swe:AllowedTokens>
                        </swe:constraint>
                    </swe:Category>
                </commonData>
            </definition>
        </InputDescriptor>
    </taskingDescriptor>
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector 1</sensorID>
        <gml:description>Start_Self_Test Command:   This command initiates self test on
all or selected CCSI components. The command also
      allows selection of a particular level of test to be performed. </gml:description>
        <InputDescriptor parameterID="Start_Self_Test-Component" updateable="true"
use="required">
            <gml:description> Select the CCSI component to be tested. </gml:description>
            <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Start_Self_Test:Component">
                        <swe:constraint>
                            <swe:AllowedTokens>
                                <swe:valueList>CC</swe:valueList>
                                <swe:valueList>PDIL</swe:valueList>
                                <swe:valueList>SC</swe:valueList>
                                <swe:valueList>UIC</swe:valueList>
                                <swe:valueList>WCC</swe:valueList>
                                <swe:valueList>DPC</swe:valueList>
                            </swe:AllowedTokens>
                        </swe:constraint>
                    </swe:Category>
                </commonData>
            </definition>
        </InputDescriptor>
        <InputDescriptor parameterID="Start_Self_Test-Level" updateable="true"
use="optional">
            <gml:description> Select the level of self test to be performed on the
component. The default is a full test.
          </gml:description>
            <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Start_Self_Test:Level"/>
                </commonData>
            </definition>
        </InputDescriptor>
```



```
    </taskingDescriptor>
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
        <gml:description>Set Heartbeat Command:   This command sets the heartbeat rate
for the sensor to the specified number of seconds or
      disables the heartbeat messages. </gml:description>
        <InputDescriptor parameterID="Set_Heartbeat-Rate" updateable="true"
use="required">
            <gml:description> Enter the time between heartbeat reports from the sensor.
Zero will turn off the heartbeats.
          Valid range is 0-3600 seconds. </gml:description>
            <definition>
                <commonData>
                    <swe:Quantity definition="urn:ogc:def:parameter:JPEO-
CBD::Set_Heartbeat:Rate">
                        <swe:uom code="Sec"/>
                        <swe:constraint>
                            <swe:AllowedValues>
                                <swe:interval>0 3600</swe:interval>
                            </swe:AllowedValues>
                        </swe:constraint>
                    </swe:Quantity>
                </commonData>
            </definition>
        </InputDescriptor>
    </taskingDescriptor>
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
        <gml:description>Set_Ccsi_Mode Command:   This command sets the operating mode of
the sensor. </gml:description>
        <InputDescriptor parameterID="Set_Ccsi_Mode-Mode" updateable="true"
use="required">
            <gml:description> Enter the new mode for the sensor. </gml:description>
            <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Set_Ccsi_Mode:Mode">
                        <swe:constraint>
                            <swe:AllowedTokens>
                                <swe:valueList>NORMAL</swe:valueList>
                                <swe:valueList>DEGRAD</swe:valueList>
                                <swe:valueList>EXERCS</swe:valueList>
                                <swe:valueList>TEST</swe:valueList>
                                <swe:valueList>TRAIN</swe:valueList>
                                <swe:valueList>INIT</swe:valueList>
                            </swe:AllowedTokens>
                        </swe:constraint>
                    </swe:Category>
                </commonData>
            </definition>
        </InputDescriptor>
    </taskingDescriptor>
    <taskingDescriptor>
        <sensorID>urn:ogc:def:procedure:JPEO-CBD::RadDetector_1</sensorID>
        <gml:description>Set_Config Command:   This command allows a user to set the
value of an option in the CCSI sensor. Most options
      are set through other commands or through the sensor configuration interface, but
this allows hosts to override. </gml:description>
        <InputDescriptor parameterID="Set Config-Name" updateable="true" use="required">
            <gml:description> Enter the name of the CCSI option item to be set.
</gml:description>
            <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Set Config:Name"/>
                </commonData>
            </definition>
        </InputDescriptor>
        <InputDescriptor parameterID="Set Config-Block" updateable="true" use="optional">
            <gml:description> Select the configuration item block name that contains the
item. </gml:description>
            <definition>
```



```
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Set Config:Block">
                        <swe:constraint>
                            <swe:AllowedTokens>
                                <swe:valueList>BASE</swe:valueList>
                                <swe:valueList>GENERAL</swe:valueList>
                                <swe:valueList>ANNUN</swe:valueList>
                                <swe:valueList>TAMPER</swe:valueList>
                                <swe:valueList>COMM</swe:valueList>
                                <swe:valueList>SECURITY</swe:valueList>
                                <swe:valueList>LINKS</swe:valueList>
                                <swe:valueList>SENSDESC</swe:valueList>
                                <swe:valueList>OPERATORS</swe:valueList>
                                <swe:valueList>UNIQUE</swe:valueList>
                                <swe:valueList>LOCAL</swe:valueList>
                            </swe:AllowedTokens>
                        </swe:constraint>
                        <swe:value>
            </swe:value>
                    </swe:Category>
                </commonData>
             </definition>
        </InputDescriptor>
        <InputDescriptor parameterID="Set_Config-Key" updateable="true" use="optional">
            <gml:description> Enter the key value that distinguishes the item to be set
from other instances of the item if
            needed. </gml:description>
             <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Set Config:Key">
                        <swe:value>
            </swe:value>
                    </swe:Category>
                </commonData>
             </definition>
        </InputDescriptor>
        <InputDescriptor parameterID="Set_Config-Value" updateable="true" use="required">
            <gml:description> Enter the value to be set ensuring that it conforms to the
option item data type.
          </gml:description>
             <definition>
                <commonData>
                    <swe:Category definition="urn:ogc:def:parameter:JPEO-
CBD::Set_Config:Value"/>
                </commonData>
             </definition>
        </InputDescriptor>
    </taskingDescriptor>
</DescribeTaskingRequestResponse>
```

Annex C

CCSI to SWE Mappings

C.1 Introduction

This annex provides XSLT 1.0 mappings between CCSI encodings and SWE encodings and vice versa. When using these XSLT translations for converting from CCSI to SWE formats, the translations provide output that is compliant with the CCSI profile of SWE defined above in section 11.3. Note that the mapping from CCSI formats to SAS formats has been omitted given the uncertainty of SAS use in the future. Refer to Annex C in the OWS-6 CCSI-SWE ER for mappings to SAS formats.

C.2 CCSI Sensor Definition to SensorML 1.0.1

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <!--Converts a CCSI Sensor Definition document into a SensorML v1.0.1 document.-->
    <xsl:param name="sensorID"/>
    <xsl:param name="latitude"/>
    <xsl:param name="longitude"/>
    <xsl:param name="altitude"/>
    <xsl:param name="orientation"/>
    <xsl:template match="/">
        <System xmlns="http://www.opengis.net/sensorML/1.0.1"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:om="http://www.opengis.net/om/1.0"
xmlns:gml="http://www.opengis.net/gml" xmlns:tml="http://www.opengis.net/tml"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" gml:id="CCSI_Sensor" xsi:schemaLocation="http://www.opengis.net/sensorML/1.0.1
http://schemas.opengis.net/sensorML/1.0.1/sensorML.xsd">
            <!--============================-->
            <!--Sensor Discovery Metadata-->
            <!--============================-->
            <gml:description>
                <xsl:value-of select="//Sensor/SensorDescription"/>
            </gml:description>
            <keywords>
                <KeywordList>
                    <keyword>CCSI</keyword>
                    <keyword>CBRN</keyword>
                    <keyword>Sensor</keyword>
                </KeywordList>
            </keywords>
            <identification>
                <IdentifierList>
                    <identifier name="uniqueID">
                        <Term definition="urn:ogc:def:identifier:OGC:uniqueID">
                            <value>
                                <xsl:value-of select="$sensorID"/>
                            </value>
                        </Term>
                    </identifier>
                    <xsl:for-each select="//SensorIdentification/*">
                        <xsl:variable name="text" select="./text()"/>
                        <xsl:variable name="name" select="local-name()"/>
                        <xsl:if test="string-length($text)>0">
                            <identifier name="{$name}">
```



```
                                    <Term definition="{concat('urn:ogc:def:identifier:JPEO-
CBD::',$name)}">
                                        <value>
                                            <xsl:value-of select="$text"/>
                                        </value>
                                    </Term>
                                </identifier>
                            </xsl:if>
                            <xsl:for-each select="./@*">
                                <identifier name="{local-name()}">
                                    <Term definition="{concat('urn:ogc:def:identifier:JPEO-
CBD::',$name,':',local-name())}">
                                        <value>
                                            <xsl:value-of select="."/>
                                        </value>
                                    </Term>
                                </identifier>
                            </xsl:for-each>
                        </xsl:for-each>
                    </IdentifierList>
                </identification>
                <classification>
                    <ClassifierList>
                        <classifier name="intendedApplication">
                            <Term definition="urn:ogc:def:classifier:OGC:application">
                                <value>CBRN Detection</value>
                            </Term>
                        </classifier>
                        <classifier name="sensorType">
                            <Term definition="urn:ogc:def:classifier:OGC:sensorType">
                                <value>
                                    <xsl:variable name="sensorType"
select="//SensorIdentification/SensorType/@class"/>
                                    <xsl:choose>
                                        <xsl:when
test="$sensorType='BIO'">Biological</xsl:when>
                                        <xsl:when
test="$sensorType='CHM'">Chemical</xsl:when>
                                        <xsl:when
test="$sensorType='NKN'">Unknown</xsl:when>
                                        <xsl:when test="$sensorType='NOS'">Not
Specified</xsl:when>
                                        <xsl:when
test="$sensorType='RAD'">Radiological</xsl:when>
                                        <xsl:when
test="$sensorType='NUC'">Nuclear</xsl:when>
                                        <xsl:when
test="$sensorType='EXP'">Experimental</xsl:when>
                                        <xsl:when
test="$sensorType='MET'">Meteorological</xsl:when>
                                        <xsl:otherwise>Unknown</xsl:otherwise>
                                    </xsl:choose>
                                </value>
                            </Term>
                        </classifier>
                        <xsl:variable name="sensorVariant"
select="//SensorIdentification/SensorType/@variant"/>
                        <xsl:if test="string-length($sensorVariant)>0">
                            <classifier name="sensorType">
                                <Term definition="urn:ogc:def:classifier:OGC:sensorType">
                                    <value>
                                        <xsl:choose>
                                            <xsl:when test="$sensorVariant='STNDOF'">Stand-
Off</xsl:when>
                                            <xsl:when
test="$sensorVariant='PNT'">Point</xsl:when>
                                        </xsl:choose>
                                    </value>
                                </Term>
                            </classifier>
                        </xsl:if>
                    </ClassifierList>
```



```
            </classification>
            <characteristics name="HW/SW Characteristics">
                <swe:DataRecord definition="urn:ogc:def:property:hwswCharacteristics">
                    <xsl:variable name="CCSIVersionElement" select="//CcsiVersion"/>
                    <swe:field name="CCSI Version">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::CCSIVersion">
                            <swe:value>
                                <xsl:variable name="versionMajor"
select="$CCSIVersionElement/@major"/>
                                <xsl:variable name="versionModerate"
select="$CCSIVersionElement/@moderate"/>
                                <xsl:variable name="versionMinor"
select="$CCSIVersionElement/@minor"/>
                                <xsl:variable name="versionPatch"
select="$CCSIVersionElement/@patch"/>
                                <xsl:value-of
select="concat($versionMajor,'.',$versionModerate,'.',$versionMinor,'.',$versionPatch)"/>
                            </swe:value>
                        </swe:Category>
                    </swe:field>
                    <xsl:variable name="releaseVersion" select="//ReleaseVersion"/>
                    <swe:field name="HW Version">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::HWVersion">
                            <swe:value>
                                <xsl:variable name="hwVersion"
select="$releaseVersion/HwVersion"/>
                                <xsl:variable name="versionMajor"
select="$hwVersion/@major"/>
                                <xsl:variable name="versionModerate"
select="$hwVersion/@moderate"/>
                                <xsl:variable name="versionMinor"
select="$hwVersion/@minor"/>
                                <xsl:variable name="versionPatch"
select="$hwVersion/@patch"/>
                                <xsl:variable name="hwVersionString"
select="concat($versionMajor,'.',$versionModerate,'.',$versionMinor,'.',$versionPatch)"/>
                                <xsl:value-of select="substring-
before($hwVersionString,'..')"/>
                            </swe:value>
                        </swe:Category>
                    </swe:field>
                    <swe:field name="SW Version">
                        <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::SWVersion">
                            <swe:value>
                                <xsl:variable name="swVersion"
select="$releaseVersion/SwVersion"/>
                                <xsl:variable name="versionMajor"
select="$swVersion/@major"/>
                                <xsl:variable name="versionModerate"
select="$swVersion/@moderate"/>
                                <xsl:variable name="versionMinor"
select="$swVersion/@minor"/>
                                <xsl:variable name="versionPatch"
select="$swVersion/@patch"/>
                                <xsl:variable name="swVersionString"
select="concat($versionMajor,'.',$versionModerate,'.',$versionMinor,'.',$versionPatch)"/>
                                <xsl:value-of select="substring-
before($swVersionString,'..')"/>
                            </swe:value>
                        </swe:Category>
                    </swe:field>
                </swe:DataRecord>
            </characteristics>
            <xsl:for-each select="//Capability/Capability">
                <xsl:variable name="itemType" select="ItemType"/>
                <capabilities name="{concat($itemType,' Measurement Properties')}">
                    <swe:DataRecord definition="urn:ogc:def:property:JPEO-
CBD::capabilityInformation">
                        <xsl:if test="$itemType">
```



```
                            <swe:field name="itemType">
                                <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::itemType">
                                    <swe:value>
                                        <xsl:value-of select="$itemType"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                        </xsl:if>
                        <xsl:if test="ItemSpecific">
                            <swe:field name="itemSpecific">
                                <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::itemSpecific">
                                    <swe:value>
                                        <xsl:value-of select="ItemSpecific"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                        </xsl:if>
                        <xsl:if test="MinimumLevel">
                            <swe:field name="minimumLevel">
                                <swe:Quantity definition="urn:ogc:def:property:JPEO-
CBD::minimumLevel">
                                    <swe:value>
                                        <xsl:value-of select="MinimumLevel"/>
                                    </swe:value>
                                </swe:Quantity>
                            </swe:field>
                        </xsl:if>
                        <xsl:if test="LevelUnits">
                            <swe:field name="levelUnits">
                                <swe:Category definition="urn:ogc:def:property:JPEO-
CBD::levelUnits">
                                    <swe:value>
                                        <xsl:value-of select="LevelUnits"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                        </xsl:if>
                        <xsl:if test="SampleTime">
                            <swe:field name="sampleTime">
                                <swe:Quantity definition="urn:ogc:def:property:JPEO-
CBD::sampleTime">
                                    <swe:value>
                                        <xsl:value-of select="SampleTime"/>
                                    </swe:value>
                                </swe:Quantity>
                            </swe:field>
                        </xsl:if>
                        <xsl:if test="ROCAreaUnderCurve">
                            <swe:field name="rocAreaUnderCurve">
                                <swe:Quantity definition="urn:ogc:def:property:JPEO-
CBD::rocAreaUnderCurve">
                                    <swe:value>
                                        <xsl:value-of select="ROCAreaUnderCurve"/>
                                    </swe:value>
                                </swe:Quantity>
                            </swe:field>
                        </xsl:if>
                        <xsl:for-each select="Restrictions/OperatingRange">
                            <xsl:variable name="quantity" select="Quantity"/>
                            <xsl:variable name="units" select="Units"/>
                            <xsl:variable name="min" select="Min"/>
                            <xsl:variable name="max" select="Max"/>
                            <swe:field
name="{concat($quantity,' OperatingRangeRestrictions')}">
                                <swe:QuantityRange
definition="{concat('urn:ogc:def:property:JPEO-
CBD::operatingRangeRestrictions:',$quantity)}">
                                    <swe:uom code="{$units}"/>
                                    <swe:value>
                                        <xsl:value-of select="concat($min,' ',$max)"/>
```

```
                                    </swe:value>
                                </swe:QuantityRange>
                            </swe:field>
                        </xsl:for-each>
                    </swe:DataRecord>
                </capabilities>
            </xsl:for-each>
            <!--<validTime>
      <gml:TimePeriod>
        <gml:beginPosition indeterminatePosition="unknown" />
        <gml:endPosition indeterminatePosition="now" />
      </gml:TimePeriod>
    </validTime>-->
            <xsl:variable name="manufacturer" select="//Manufacturer"/>
            <xsl:variable name="program" select="//Program"/>
            <xsl:if test="$manufacturer and $program">
                <contact>
                    <ContactList>
                        <xsl:if test="$manufacturer">
                            <xsl:variable name="individualName"
select="$manufacturer/PointOfContactNameText"/>
                            <xsl:variable name="organizationName"
select="$manufacturer/PointOfContactAgencyAcronymText"/>
                            <xsl:variable name="voice"
select="$manufacturer/PointOfContactPhoneNumberText"/>
                            <xsl:variable name="facsimile"
select="$manufacturer/PointOfContactFaxNumberText"/>
                            <xsl:variable name="deliveryPoint"
select="$manufacturer/PointOfContactAddressLineText"/>
                            <xsl:variable name="city"
select="$manufacturer/PointOfContactCityNameText"/>
                            <xsl:variable name="administrativeArea"
select="$manufacturer/PointOfContactStateNameText"/>
                            <xsl:variable name="postalCode"
select="$manufacturer/PointOfContactPostalCodeText"/>
                            <xsl:variable name="country"
select="$manufacturer/PointOfContactCountryCode"/>
                            <xsl:variable name="electronicMailAddress"
select="$manufacturer/PointOfContactEmailAddressText"/>
                            <member
xlink:role="urn:ogc:def:identifier:OGC::manufacturer">
                                <ResponsibleParty>
                                    <xsl:if test="$individualName">
                                        <individualName>
                                            <xsl:value-of select="$individualName"/>
                                        </individualName>
                                    </xsl:if>
                                    <xsl:if test="$organizationName">
                                        <organizationName>
                                            <xsl:value-of select="$organizationName"/>
                                        </organizationName>
                                    </xsl:if>
                                    <contactInfo>
                                        <xsl:if test="$voice or $facsimile">
                                            <phone>
                                                <xsl:if test="$voice">
                                                    <voice>
                                                        <xsl:value-of select="$voice"/>
                                                    </voice>
                                                </xsl:if>
                                                <xsl:if test="$facsimile">
                                                    <facsimile>
                                                        <xsl:value-of
select="$facsimile"/>
                                                    </facsimile>
                                                </xsl:if>
                                            </phone>
                                        </xsl:if>
                                        <xsl:if test="$deliveryPoint or $city or
$administrativeArea or $postalCode or $country or $electronicMailAddress">
                                            <address>
                                                <xsl:if test="$deliveryPoint">
```



```
                                                    <deliveryPoint>
                                                        <xsl:value-of
select="$deliveryPoint"/>
                                                    </deliveryPoint>
                                                </xsl:if>
                                                <xsl:if test="$city">
                                                    <city>
                                                        <xsl:value-of select="$city"/>
                                                    </city>
                                                </xsl:if>
                                                <xsl:if test="$administrativeArea">
                                                    <administrativeArea>
                                                        <xsl:value-of
select="$administrativeArea"/>
                                                    </administrativeArea>
                                                </xsl:if>
                                                <xsl:if test="$postalCode">
                                                    <postalCode>
                                                        <xsl:value-of
select="$postalCode"/>
                                                    </postalCode>
                                                </xsl:if>
                                                <xsl:if test="$country">
                                                    <country>
                                                        <xsl:value-of
select="$country"/>
                                                    </country>
                                                </xsl:if>
                                                <xsl:if test="$electronicMailAddress">
                                                    <electronicMailAddress>
                                                        <xsl:value-of
select="$electronicMailAddress"/>
                                                    </electronicMailAddress>
                                                </xsl:if>
                                            </address>
                                        </xsl:if>
                                    </contactInfo>
                                </ResponsibleParty>
                            </member>
                        </xsl:if>
                        <xsl:if test="$program">
                            <xsl:variable name="individualName"
select="$program/PointOfContactNameText"/>
                            <xsl:variable name="organizationName"
select="$program/PointOfContactAgencyAcronymText"/>
                            <xsl:variable name="voice"
select="$program/PointOfContactPhoneNumberText"/>
                            <xsl:variable name="facsimile"
select="$program/PointOfContactFaxNumberText"/>
                            <xsl:variable name="deliveryPoint"
select="$program/PointOfContactAddressLineText"/>
                            <xsl:variable name="city"
select="$program/PointOfContactCityNameText"/>
                            <xsl:variable name="administrativeArea"
select="$program/PointOfContactStateNameText"/>
                            <xsl:variable name="postalCode"
select="$program/PointOfContactPostalCodeText"/>
                            <xsl:variable name="country"
select="$program/PointOfContactCountryCode"/>
                            <xsl:variable name="electronicMailAddress"
select="$program/PointOfContactEmailAddressText"/>
                            <member xlink:role="urn:ogc:def:identifier:OGC::program">
                                <ResponsibleParty>
                                    <xsl:if test="$individualName">
                                        <individualName>
                                            <xsl:value-of select="$individualName"/>
                                        </individualName>
                                    </xsl:if>
                                    <xsl:if test="$organizationName">
                                        <organizationName>
                                            <xsl:value-of select="$organizationName"/>
                                        </organizationName>
```

```
                                    </xsl:if>
                                    <contactInfo>
                                        <xsl:if test="$voice or $facsimile">
                                            <phone>
                                                <xsl:if test="$voice">
                                                    <voice>
                                                        <xsl:value-of select="$voice"/>
                                                    </voice>
                                                </xsl:if>
                                                <xsl:if test="$facsimile">
                                                    <facsimile>
                                                        <xsl:value-of
select="$facsimile"/>
                                                    </facsimile>
                                                </xsl:if>
                                            </phone>
                                        </xsl:if>
                                        <xsl:if test="$deliveryPoint or $city or
$administrativeArea or $postalCode or $country or $electronicMailAddress">
                                            <address>
                                                <xsl:if test="$deliveryPoint">
                                                    <deliveryPoint>
                                                        <xsl:value-of
select="$deliveryPoint"/>
                                                    </deliveryPoint>
                                                </xsl:if>
                                                <xsl:if test="$city">
                                                    <city>
                                                        <xsl:value-of select="$city"/>
                                                    </city>
                                                </xsl:if>
                                                <xsl:if test="$administrativeArea">
                                                    <administrativeArea>
                                                        <xsl:value-of
select="$administrativeArea"/>
                                                    </administrativeArea>
                                                </xsl:if>
                                                <xsl:if test="$postalCode">
                                                    <postalCode>
                                                        <xsl:value-of
select="$postalCode"/>
                                                    </postalCode>
                                                </xsl:if>
                                                <xsl:if test="$country">
                                                    <country>
                                                        <xsl:value-of
select="$country"/>
                                                    </country>
                                                </xsl:if>
                                                <xsl:if test="$electronicMailAddress">
                                                    <electronicMailAddress>
                                                        <xsl:value-of
select="$electronicMailAddress"/>
                                                    </electronicMailAddress>
                                                </xsl:if>
                                            </address>
                                        </xsl:if>
                                    </contactInfo>
                                </ResponsibleParty>
                            </member>
                        </xsl:if>
                    </ContactList>
                </contact>
            </xsl:if>
            <xsl:variable name="manufacturerURL"
select="//Manufacturer/PointOfContactWebPageAddress"/>
            <xsl:variable name="programURL"
select="//Program/PointOfContactWebPageAddress"/>
            <xsl:if test="string-length($manufacturerURL)>0 and string-
length($programURL)>0">
                <documentation xlink:role="documents">
                    <DocumentList>
```



```
                        <xsl:if test="string-length($manufacturerURL)>0">
                            <member name="manufacturerURL">
                                <Document>
                                    <gml:description>Link to information about the
sensor manufacturer</gml:description>
                                    <onlineResource xlink:href="{$manufacturerURL}"/>
                                </Document>
                            </member>
                        </xsl:if>
                        <xsl:if test="string-length($programURL)>0">
                            <member name="programURL">
                                <Document>
                                    <gml:description>Link to information about the
program utilizing this sensor</gml:description>
                                    <onlineResource xlink:href="{$programURL}"/>
                                </Document>
                            </member>
                        </xsl:if>
                    </DocumentList>
                </documentation>
            </xsl:if>
            <!--============================-->
            <!--Sensor Coordinate Frame-->
            <!--============================-->
            <spatialReferenceFrame>
                <gml:EngineeringCRS gml:id="SENSOR FRAME">
                    <gml:srsName>CCSI Sensor Spatial Frame</gml:srsName>
                    <gml:usesCS xlink:href="urn:ogc:def:cs:xyzFrame"/>
                    <gml:usesEngineeringDatum>
                        <gml:EngineeringDatum gml:id="SENSOR DATUM">
                            <gml:datumName>CCSI Sensor Spatial Datum</gml:datumName>
                            <gml:anchorPoint>origin is at the base of the mounting. Z is
along the axis of the mounting pole - typically vertical. X and Y are orthogonal to Z,
along the short and long edges of the case respectively.</gml:anchorPoint>
                        </gml:EngineeringDatum>
                    </gml:usesEngineeringDatum>
                </gml:EngineeringCRS>
            </spatialReferenceFrame>
            <!--============================-->
            <!--Sensor Interfaces-->
            <!--============================-->
            <interfaces>
                <!--unknown-->
            </interfaces>
            <!--====================-->
            <!--Sensor Inputs-->
            <!--====================-->
            <inputs>
                <InputList>
                    <input name="genericCBRNHazard">
                        <swe:ObservableProperty definition="urn:ogc:def:phenomenon:JPEO-
CBD::toxicAgent"/>
                    </input>
                </InputList>
            </inputs>
            <!--====================-->
            <!--Sensor Outputs-->
            <!--====================-->
            <outputs>
                <OutputList>
                    <xsl:variable name="modeField">
                        <swe:field name="Mode">
                            <swe:Category definition="urn:ogc:def:phenomenon:JPEO-
CBD::Mode"/>
                        </swe:field>
                    </xsl:variable>
                    <xsl:variable name="timeField">
                        <swe:field name="DateTimeGroup">
                            <swe:Time definition="urn:ogc:def:phenomenon:JPEO-CBD::Time"
referenceTime="1970-01-01T00:00:00Z">
                                <swe:uom code="s"/>
                            </swe:Time>
```

```
                        </swe:field>
                    </xsl:variable>
                    <xsl:for-each select="//Channels/Channel">
                        <xsl:variable name="channelName" select="@Channel"/>
                        <output name="{$channelName}">
                            <swe:DataRecord
gml:id="{concat($channelName,'_DATA_RECORD')}"
definition="{concat('urn:ogc:def:phenomenon:JPEO-CBD::',$channelName)}">
                                <xsl:choose>
                                    <xsl:when test="$channelName='READGS'">
                                        <xsl:copy-of select="$modeField"/>
                                        <swe:field name="READGS_Sensor">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Sensor"/>
                                        </swe:field>
                                        <swe:field name="READGS_ReadingID">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:ReadingID">
                                                <swe:codeSpace
xlink:href="urn:ogc:def:property:JPEO-CBD::readingIdentificationType"/>
                                            </swe:Category>
                                        </swe:field>
                                        <swe:field name="READGS_Detect">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Detect"/>
                                        </swe:field>
                                        <swe:field name="READGS_Level">
                                            <swe:Quantity
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level"/>
                                        </swe:field>
                                        <swe:field
name="READGS_LevelConfidenceInterval">
                                            <swe:Quantity
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:LevelConfidenceInterval"/>
                                        </swe:field>
                                        <swe:field name="READGS_Id">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id"/>
                                        </swe:field>
                                        <swe:field name="READGS_DetailsAvailable">
                                            <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:DetailsAvailable"/>
                                        </swe:field>
                                    </xsl:when>
                                    <xsl:when test="$channelName='ALERTS'">
                                        <xsl:copy-of select="$modeField"/>
                                        <swe:field name="ALERTS_Event">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Event">
                                                <swe:constraint>
                                                    <swe:AllowedTokens>
                                                        <swe:valueList>ALERT DEALERT
WARN DEWARN NONE</swe:valueList>
                                                    </swe:AllowedTokens>
                                                </swe:constraint>
                                            </swe:Category>
                                        </swe:field>
                                        <swe:field name="ALERTS_Source">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Source">
                                                <swe:constraint>
                                                    <swe:AllowedTokens>
                                                        <swe:valueList>none detector
bit tamper</swe:valueList>
                                                    </swe:AllowedTokens>
                                                </swe:constraint>
                                            </swe:Category>
                                        </swe:field>
                                        <!--Time and AlertId will be used to populate
the gml:id and om:samplingTime fields
                    <swe:field name="ALERTS Time">
                      <swe:Time definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Time"/>
```



```
                </swe:field>
                -->
                                        <swe:field name="ALERTS_AlertId">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:AlertId"/>
                                        </swe:field>
                                        <!--READGS Fields-->
                                        <!--Insert Here?-->
                                        <!--BIT Information-->
                                        <swe:field name="ALERTS_BIT_Component">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:Component">
                                                <swe:constraint>
                                                    <swe:AllowedTokens>
                                                        <swe:valueList>CC PDIL SC UIC
WCC DPC</swe:valueList>
                                                    </swe:AllowedTokens>
                                                </swe:constraint>
                                            </swe:Category>
                                        </swe:field>
                                        <swe:field name="ALERTS_BIT_Executed">
                                            <swe:Time
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:Executed" referenceTime="1970-01-
01">
                                                <swe:uom code="s"/>
                                            </swe:Time>
                                        </swe:field>
                                        <swe:field name="ALERTS_BIT_Result">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:Result">
                                                <swe:constraint>
                                                    <swe:AllowedTokens>
                                                        <swe:valueList>PASS
FAIL</swe:valueList>
                                                    </swe:AllowedTokens>
                                                </swe:constraint>
                                            </swe:Category>
                                        </swe:field>
                                        <swe:field name="ALERTS_BIT_Level">
                                            <swe:Count
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:Level"/>
                                        </swe:field>
                                        <swe:field name="ALERTS_BIT_ErrorDescription">
                                            <swe:Text
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:ErrorDescription"/>
                                        </swe:field>
                                        <!--Tamper Information-->
                                        <swe:field name="ALERTS_Tamper">
                                            <swe:Text
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Tamper"/>
                                        </swe:field>
                                    </xsl:when>
                                    <xsl:when test="$channelName='HRTBT'">
                                        <xsl:copy-of select="$modeField"/>
                                        <xsl:copy-of select="$timeField"/>
                                    </xsl:when>
                                    <xsl:when test="$channelName='STATUS'">
                                        <xsl:copy-of select="$modeField"/>
                                        <swe:field name="STATUS_BIT">
                                            <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS:BIT"/>
                                        </swe:field>
                                        <swe:field name="STATUS_Alert">
                                            <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS:Alert"/>
                                        </swe:field>
                                        <swe:field name="STATUS_Maint">
                                            <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS:Maint"/>
                                        </swe:field>
                                    </xsl:when>
                                    <xsl:when test="$channelName='MAINT'">
```

```
                                        <xsl:copy-of select="$modeField"/>
                                        <swe:field name="MAINT_Type">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Type"/>
                                        </swe:field>
                                        <swe:field name="MAINT_BIT_Result">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:BIT:Result"/>
                                        </swe:field>
                                        <swe:field name="MAINT_BIT_Executed">
                                            <swe:Time
definition="urn:ogc:data:time:iso8601"/>
                                        </swe:field>
                                        <swe:field name="MAINT_BIT_Component">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:BIT:Component"/>
                                        </swe:field>
                                    </xsl:when>
                                </xsl:choose>
                                <xsl:for-each select="Metadata">
                                    <xsl:variable name="metadataValue" select="."/>
                                    <xsl:variable name="dataElement"
select="//DataDefinitions/DataElement[@name=$metadataValue]"/>
                                    <xsl:variable name="name"
select="$dataElement/@name"/>
                                    <swe:field name="{concat($channelName,'_',$name)}">
                                        <xsl:variable name="definition"
select="concat('urn:ogc:def:phenomenon:JPEO-CBD::',$channelName,':',$name)"/>
                                        <xsl:variable name="uom">
                                            <swe:uom
code="{$dataElement/UnitOfMeasure}"/>
                                        </xsl:variable>
                                        <xsl:variable name="range"
select="$dataElement/ValidityCheck/Range"/>
                                        <xsl:variable name="minValue" select="substring-
before($range,' - ')"/>
                                        <xsl:variable name="maxValue" select="substring-
after($range,' - ')"/>
                                        <xsl:variable name="quantityConstraints">
                                            <xsl:if test="$minValue or $maxValue">
                                                <swe:constraint>
                                                    <swe:AllowedValues>
                                                        <xsl:choose>
                                                            <xsl:when test="$minValue
and $maxValue">
                                                                <swe:interval>
                                                                    <xsl:value-of
select="concat($minValue,' ',$maxValue)"/>
                                                                </swe:interval>
                                                            </xsl:when>
                                                            <xsl:when test="$minValue">
                                                                <swe:min>
                                                                    <xsl:value-of
select="$minValue"/>
                                                                </swe:min>
                                                            </xsl:when>
                                                            <xsl:when test="$maxValue">
                                                                <swe:max>
                                                                    <xsl:value-of
select="$maxValue"/>
                                                                </swe:max>
                                                            </xsl:when>
                                                        </xsl:choose>
                                                    </swe:AllowedValues>
                                                </swe:constraint>
                                            </xsl:if>
                                        </xsl:variable>
                                        <xsl:variable name="enumeration"
select="$dataElement/ValidityCheck/Enumeration"/>
                                        <xsl:variable name="enumerationItems"
select="$enumeration//Item"/>
                                        <xsl:variable name="enumerationString">
```



```
                                        <xsl:for-each select="$enumerationItems">
                                            <xsl:value-of select="."/>
                                            <xsl:value-of select="' '"/>
                                        </xsl:for-each>
                                    </xsl:variable>
                                    <xsl:variable name="enumConstraints">
                                        <swe:constraint>
                                            <swe:AllowedTokens>
                                                <swe:valueList>
                                                    <xsl:value-of
select="normalize-space($enumerationString)"/>
                                                </swe:valueList>
                                            </swe:AllowedTokens>
                                        </swe:constraint>
                                    </xsl:variable>
                                    <xsl:variable name="type"
select="$dataElement/@type"/>
                                    <xsl:choose>
                                        <xsl:when test="$type='int' or $type='float'
or $type='float' or $type='short' or $type='long' or $type='ushort' or $type='uint' or
$type='ulong'">
                                            <swe:Quantity
definition="{$definition}">
                                                <xsl:copy-of select="$uom"/>
                                                <xsl:copy-of
select="$quantityConstraints"/>
                                            </swe:Quantity>
                                        </xsl:when>
                                        <xsl:when test="$type='enum'">
                                            <swe:Category
definition="{$definition}">
                                                <xsl:copy-of
select="$enumConstraints"/>
                                            </swe:Category>
                                        </xsl:when>
                                        <xsl:when test="$type='date time'">
                                            <swe:Time definition="{$definition}">
                                                <xsl:copy-of select="$uom"/>
                                            </swe:Time>
                                        </xsl:when>
                                        <xsl:when test="$type='location'">
                                            <swe:Vector definition="{$definition}">
                                                <swe:coordinate name="latitude">
                                                    <swe:Quantity axisID="y">
                                                        <swe:uom code="deg"/>
                                                    </swe:Quantity>
                                                </swe:coordinate>
                                                <swe:coordinate name="longitude">
                                                    <swe:Quantity axisID="x">
                                                        <swe:uom code="deg"/>
                                                    </swe:Quantity>
                                                </swe:coordinate>
                                                <swe:coordinate name="altitude">
                                                    <swe:Quantity axisID="z">
                                                        <swe:uom code="m"/>
                                                    </swe:Quantity>
                                                </swe:coordinate>
                                            </swe:Vector>
                                        </xsl:when>
                                        <xsl:when test="$type='boolean'">
                                            <swe:Boolean/>
                                        </xsl:when>
                                        <xsl:otherwise>
                                            <!--
                    <xs:enumeration value="boolean"/>
                    <xs:enumeration value="byte"/>
                    <xs:enumeration value="short"/>
                    <xs:enumeration value="float"/>
                    <xs:enumeration value="int"/>
                    <xs:enumeration value="long"/>
                    <xs:enumeration value="string"/>
                    <xs:enumeration value="ubyte"/>
```

```
                    <xs:enumeration value="ushort"/>
                    <xs:enumeration value="uint"/>
                    <xs:enumeration value="ulong"/>
                    <xs:enumeration value="time period"/>
                    <xs:enumeration value="date time"/>
                    <xs:enumeration value="ip_address"/>
                    <xs:enumeration value="mac_address"/>
                    <xs:enumeration value="url"/>
                    <xs:enumeration value="enum"/>
                    <xs:enumeration value="array"/>
                    <xs:enumeration value="location"/>
                    -->
                                        </xsl:otherwise>
                                    </xsl:choose>
                                </swe:field>
                            </xsl:for-each>
                        </swe:DataRecord>
                    </output>
                </xsl:for-each>
            </OutputList>
        </outputs>
        <!--=========================-->
        <!--Sensor Detector List-->
        <!--=========================-->
        <components>
            <!--Unknown-->
        </components>
        <!--=========================-->
        <!--fields Positions-->
        <!--=========================-->
        <positions>
            <PositionList>
                <!--==================================-->
                <!--Position of Sensor in EPSG4326-->
                <!--==================================-->
                <position name="sensorPosition">
                    <swe:Position localFrame="#SENSOR_FRAME"
referenceFrame="urn:ogc:def:crs:EPSG:4326">
                        <swe:location>
                            <swe:Vector
definition="urn:ogc:def:vector:OGC:location">
                                <swe:coordinate name="latitude">
                                    <swe:Quantity axisID="Y"
definition="urn:ogc:def:phenomenon:latitude">
                                        <swe:uom code="deg"/>
                                        <swe:value>
                                            <xsl:value-of select="$latitude"/>
                                        </swe:value>
                                    </swe:Quantity>
                                </swe:coordinate>
                                <swe:coordinate name="longitude">
                                    <swe:Quantity axisID="X"
definition="urn:ogc:def:phenomenon:longitude">
                                        <swe:uom code="deg"/>
                                        <swe:value>
                                            <xsl:value-of select="$longitude"/>
                                        </swe:value>
                                    </swe:Quantity>
                                </swe:coordinate>
                                <swe:coordinate name="altitude">
                                    <swe:Quantity axisID="Z"
definition="urn:ogc:def:phenomenon:altitude">
                                        <swe:uom code="m"/>
                                        <swe:value>
                                            <xsl:value-of select="$altitude"/>
                                        </swe:value>
                                    </swe:Quantity>
                                </swe:coordinate>
                            </swe:Vector>
                        </swe:location>
                        <swe:orientation>
```



```
                                <swe:Vector
definition="urn:ogc:def:vector:OGC:orientation">
                                    <swe:coordinate name="trueHeading">
                                        <swe:Quantity
definition="urn:ogc:def:phenomenon:angleToNorth">
                                            <swe:uom code="deg"/>
                                            <swe:value>
                                                <xsl:value-of select="$orientation"/>
                                            </swe:value>
                                        </swe:Quantity>
                                    </swe:coordinate>
                                </swe:Vector>
                            </swe:orientation>
                        </swe:Position>
                    </position>
                </PositionList>
            </positions>
            <!--================================-->
            <!--Station Internal Connections-->
            <!--================================-->
            <connections>
                <!--Unknown-->
            </connections>
        </System>
    </xsl:template>
</xsl:stylesheet>
```

C.3 CCSI Channels to O & M 1.0.0

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <xsl:param name="dtgTime"/>
    <xsl:param name="sensorID"/>
    <xsl:param name="latitude"/>
    <xsl:param name="longitude"/>
    <xsl:template match="/">
        <xsl:variable name="uniqueID" select="concat(//Hdr/@sid,'_',//Hdr/@msn)"/>
        <xsl:variable name="mod" select="//Hdr/@mod"/>
        <xsl:variable name="dtg" select="//Hdr/@dtg"/>
        <xsl:variable name="chn" select="//Hdr/@chn"/>
        <xsl:variable name="mode">
            <xsl:choose>
                <xsl:when test="$mod='D'">Degraded</xsl:when>
                <xsl:when test="$mod='X'">Exercise</xsl:when>
                <xsl:when test="$mod='T'">Test</xsl:when>
                <xsl:when test="$mod='U'">Training</xsl:when>
                <xsl:when test="$mod='S'">Startup</xsl:when>
                <xsl:when test="$mod='I'">Initialization</xsl:when>
                <xsl:otherwise>Normal</xsl:otherwise>
            </xsl:choose>
        </xsl:variable>
        <xsl:variable name="readingID" select="//ReadingReport/@ReadingID |
//AlertsChn/@AlertId"/>
        <xsl:variable name="sensor" select="//ReadingReport/@Sensor"/>
        <xsl:variable name="channel" select="substring-before(local-
name(//Msg/*),'Chn')"/>
        <xsl:variable name="modifiedChannel">
            <xsl:choose>
                <xsl:when test="$channel='Readings'">READGS</xsl:when>
                <xsl:when test="$channel='Maint'">MAINT</xsl:when>
                <xsl:when test="$channel='Alerts'">ALERTS</xsl:when>
                <xsl:when test="$channel='Status'">STATUS</xsl:when>
                <xsl:otherwise>
      </xsl:otherwise>
            </xsl:choose>
        </xsl:variable>
        <xsl:variable name="time" select="//ReadingsChn/ReadingReport/@Time |
//AlertsChn/@Time | //MaintChn/Periodic/@Time"/>
        <xsl:variable name="year" select="substring($time,1,4)"/>
        <xsl:variable name="month" select="substring($time,5,2)"/>
```

```
        <xsl:variable name="day" select="substring($time,7,2)"/>
        <xsl:variable name="hour" select="substring($time,9,2)"/>
        <xsl:variable name="minute" select="substring($time,11,2)"/>
        <xsl:variable name="second" select="substring($time,13,2)"/>
        <xsl:variable name="dataTime">
            <xsl:choose>
                <xsl:when test="$time">
                    <xsl:value-of select="concat($year,'-',$month,'-
',$day,'T',$hour,':',$minute,':',$second,'Z')"/>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="$dtgTime"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:variable>
        <Observation gml:id="{$uniqueID}" xmlns="http://www.opengis.net/om/1.0"
xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:gml="http://www.opengis.net/gml"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:swe="http://www.opengis.net/swe/1.0.1" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
xmlns:ism="urn:us:gov:ic:ism:v2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/om/1.0
http://schemas.opengis.net/om/1.0.0/om.xsd">
            <gml:description>CCSI Sensor Observation Instance</gml:description>
            <gml:name>CCSI Sensor Observation Instance (<xsl:value-of
select="$uniqueID"/>)</gml:name>
            <samplingTime>
                <gml:TimeInstant>
                    <gml:timePosition>
                        <xsl:value-of select="$dataTime"/>
                    </gml:timePosition>
                </gml:TimeInstant>
            </samplingTime>
            <procedure xlink:href="{$sensorID}"/>
            <xsl:variable name="observedProperty"
select="concat('urn:ogc:def:phenomenon:JPEO-CBD::',$modifiedChannel)"/>
            <xsl:variable name="data" select="//Data"/>
            <xsl:variable name="readingsChn" select="//ReadingsChn/ReadingReport |
//AlertsChn/Reading"/>
            <xsl:variable name="alertsChn" select="//AlertsChn"/>
            <xsl:variable name="statusChn" select="//StatusChn"/>
            <xsl:variable name="maintChn" select="//MaintChn"/>
            <xsl:variable name="sudElements" select="$data/SUD | $maintChn[last()]/SUD"/>
            <xsl:variable name="maintType" select="$maintChn/@Type"/>
            <observedProperty>
                <xsl:variable name="heartbeatComponents">
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Time"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                </xsl:variable>
                <xsl:variable name="readingComponents">
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                </xsl:variable>
                <xsl:variable name="alertComponents">
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Event"/>
```



```
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Source"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:AlertId"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Sensor"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:ReadingID"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Detect"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Level"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:LevelConfidenceInterval"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:Id"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::READGS:DetailsAvailable"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Component"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Executed"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Result"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Level"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:ErrorDescription"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Tamper"/>
                </xsl:variable>
                <xsl:variable name="maintComponents">
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Type"/>
                    <xsl:choose>
                        <xsl:when test="$maintType='periodic'">
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Periodic:Time"/>
                        </xsl:when>
                        <xsl:when test="$maintType='failure'">
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Failure:Inoperable"/>
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Failure:Unit"/>
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Failure:Degraded"/>
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:Failure:Description"/>
                        </xsl:when>
                        <xsl:otherwise>
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:BIT:Result"/>
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:BIT:Executed"/>
                            <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::MAINT:BIT:Component"/>
                        </xsl:otherwise>
                    </xsl:choose>
                </xsl:variable>
                <xsl:variable name="statusComponents">
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-CBD::Mode"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:BIT"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:Alert"/>
                    <swe:component xlink:href="urn:ogc:def:phenomenon:JPEO-
CBD::STATUS:Maint"/>
                </xsl:variable>
                <xsl:variable name="componentCount">
                    <xsl:choose>
                        <xsl:when test="$modifiedChannel='READGS'">8</xsl:when>
```

```
                        <xsl:when test="$modifiedChannel='ALERTS'">17</xsl:when>
                        <xsl:when test="$modifiedChannel='STATUS'">4</xsl:when>
                        <xsl:when test="$modifiedChannel='MAINT'">
                            <xsl:choose>
                                <xsl:when test="$maintType='periodic'">3</xsl:when>
                                <xsl:when test="$maintType='failure'">6</xsl:when>
                                <xsl:otherwise>4</xsl:otherwise>
                            </xsl:choose>
                        </xsl:when>
                        <xsl:when test="$chn='h'">2</xsl:when>
                        <xsl:otherwise>0</xsl:otherwise>
                    </xsl:choose>
                </xsl:variable>
                <swe:CompositePhenomenon
gml:id="{concat($modifiedChannel,'_',$uniqueID)}"
dimension="{count($sudElements)+$componentCount}">
                    <gml:name>CCSI <xsl:value-of select="$modifiedChannel"/>
Phenomenon</gml:name>
                    <xsl:choose>
                        <xsl:when test="$modifiedChannel='READGS'">
                            <xsl:copy-of select="$readingComponents"/>
                        </xsl:when>
                        <xsl:when test="$modifiedChannel='ALERTS'">
                            <xsl:copy-of select="$alertComponents"/>
                        </xsl:when>
                        <xsl:when test="$modifiedChannel='STATUS'">
                            <xsl:copy-of select="$statusComponents"/>
                        </xsl:when>
                        <xsl:when test="$modifiedChannel='MAINT'">
                            <xsl:copy-of select="$maintComponents"/>
                        </xsl:when>
                        <xsl:when test="$chn='h'">
                            <xsl:copy-of select="$heartbeatComponents"/>
                        </xsl:when>
                    </xsl:choose>
                    <xsl:for-each select="$sudElements">
                        <swe:component
xlink:href="{concat($observedProperty,':',@Name)}"/>
                    </xsl:for-each>
                </swe:CompositePhenomenon>
            </observedProperty>
            <featureOfInterest>
                <sa:SamplingPoint gml:id="sensor_SamplingPoint"
xmlns:sa="http://www.opengis.net/sampling/1.0"
xsi:schemaLocation="http://www.opengis.net/sampling/1.0
http://schemas.opengis.net/sampling/1.0.0/sampling.xsd">
                    <gml:name>CCSI Sensor</gml:name>
                    <sa:sampledFeature xlink:href="urn:ogc:def:nil:OGC:unknown"/>
                    <sa:position>
                        <gml:Point gml:id="sensor_Point">
                            <gml:pos srsName="urn:ogc:def:crs:EPSG:4326">
                                <xsl:value-of select="concat($latitude,'
',$longitude)"/>
                            </gml:pos>
                        </gml:Point>
                    </sa:position>
                </sa:SamplingPoint>
            </featureOfInterest>
            <result>
                <xsl:choose>
                    <xsl:when test="$maintType='failure'">
                        <swe:DataArray>
                            <swe:elementCount>
                                <swe:Count>
                                    <swe:value>
                                        <xsl:value-of select="count($maintChn)"/>
                                    </swe:value>
                                </swe:Count>
                            </swe:elementCount>
                            <swe:elementType name="Maintenance Components">
                                <swe:DataRecord definition="urn:ogc:def:type:JPEO-
CBD::MAINT:Info">
```



```
                                    <swe:field name="MAINT_Type">
                                        <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Type"/>
                                    </swe:field>
                                    <swe:field name="MAINT_Failure_Inoperable">
                                        <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Failure:Inoperable"/>
                                    </swe:field>
                                    <swe:field name="MAINT_Failure_Unit">
                                        <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Failure:Unit"/>
                                    </swe:field>
                                    <swe:field name="MAINT_Failure_Degraded">
                                        <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Failure:Degraded"/>
                                    </swe:field>
                                    <swe:field name="MAINT_Failure_Description">
                                        <swe:Text
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Failure:Description"/>
                                    </swe:field>
                                    <xsl:for-each select="$maintChn[last()]/SUD">
                                        <xsl:variable name="name" select="@Name"/>
                                        <xsl:variable name="type" select="@Type"/>
                                        <xsl:variable name="definition"
select="concat('urn:ogc:def:phenomenon:JPEO-CBD::',$modifiedChannel,':',$name)"/>
                                        <xsl:variable name="uom" select="@Units"/>
                                        <xsl:variable name="uomElement">
                                            <swe:uom code="{$uom}"/>
                                        </xsl:variable>
                                        <swe:field name="{concat('MAINT_',$name)}">
                                            <xsl:choose>
                                                <xsl:when test="$type='Integer'">
                                                    <swe:Quantity
definition="{$definition}">
                                                        <xsl:if test="$uom">
                                                            <xsl:copy-of
select="$uomElement"/>
                                                        </xsl:if>
                                                    </swe:Quantity>
                                                </xsl:when>
                                                <xsl:otherwise>
                                                    <swe:Category
definition="{$definition}"/>
                                                </xsl:otherwise>
                                            </xsl:choose>
                                        </swe:field>
                                    </xsl:for-each>
                                </swe:DataRecord>
                            </swe:elementType>
                            <swe:encoding>
                                <swe:TextBlock decimalSeparator="." tokenSeparator="||"
blockSeparator="@@"/>
                            </swe:encoding>
                            <swe:values>
                                <xsl:for-each select="$maintChn">
                                    <xsl:value-of select="./@Type"/>||<xsl:value-of
select=".//Failure/@Inoperable"/>||<xsl:value-of select=".//Failure/@Unit"/>||<xsl:value-
of select=".//Failure/@Degraded"/>||<xsl:value-of select=".//Failure/Description"/>
                                    <xsl:for-each select=".//SUD">||<xsl:value-of
select="@Value"/>
                                    </xsl:for-each>@@</xsl:for-each>
                            </swe:values>
                        </swe:DataArray>
                    </xsl:when>
                    <xsl:otherwise>
                        <swe:DataRecord
gml:id="{concat('CCSI_',$modifiedChannel,'_DATA_RECORD')}">
                            <xsl:variable name="modeField">
                                <swe:field name="Mode">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::Mode">
                                        <swe:value>
```

```
                                        <xsl:value-of select="$mode"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                        </xsl:variable>
                        <xsl:variable name="timeField">
                            <swe:field name="DateTimeGroup">
                                <swe:Time definition="urn:ogc:def:phenomenon:JPEO-
CBD::Time" referenceTime="1970-01-01T00:00:00Z">
                                    <swe:uom code="s"/>
                                    <swe:value>
                                        <xsl:value-of select="$dtg"/>
                                    </swe:value>
                                </swe:Time>
                            </swe:field>
                        </xsl:variable>
                        <xsl:variable name="heartbeatFields">
                            <xsl:copy-of select="$modeField"/>
                            <xsl:copy-of select="$timeField"/>
                        </xsl:variable>
                        <xsl:variable name="readingFields">
                            <xsl:copy-of select="$modeField"/>
                            <swe:field name="READGS_Sensor">
                                <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Sensor">
                                    <swe:value>
                                        <xsl:value-of
select="$readingsChn/@Sensor"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="READGS_ReadingID">
                                <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:ReadingID">
                                    <swe:codeSpace
xlink:href="urn:ogc:def:property:JPEO-CBD::readingIdentificationType"/>
                                    <swe:value>
                                        <xsl:value-of
select="$readingsChn/@ReadingID"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="READGS_Detect">
                                <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Detect">
                                    <swe:value>
                                        <xsl:value-of select="$data/@Detect"/>
                                    </swe:value>
                                </swe:Category>
                            </swe:field>
                            <swe:field name="READGS_Level">
                                <swe:Quantity
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Level">
                                    <xsl:if test="$data/@Units">
                                        <swe:uom code="{$data/@Units}"/>
                                    </xsl:if>
                                    <xsl:variable name="levelConfidenceInterval"
select="$data/@LevelConfidenceInterval"/>
                                    <xsl:if test="$levelConfidenceInterval">
                                        <swe:quality>
                                            <swe:Quantity
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:LevelConfidenceInterval">
                                                <swe:uom code="%"/>
                                                <swe:value>
                                                    <xsl:value-of
select="$levelConfidenceInterval"/>
                                                </swe:value>
                                            </swe:Quantity>
                                        </swe:quality>
                                    </xsl:if>
                                    <swe:value>
                                        <xsl:value-of select="$data/@Level"/>
```



```
                                        </swe:value>
                                    </swe:Quantity>
                                </swe:field>
                                <swe:field name="READGS_Id">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:Id">
                                        <xsl:variable name="id" select="$data/@Id"/>
                                        <xsl:if test="$id">
                                            <swe:value>
                                                <xsl:value-of select="$id"/>
                                            </swe:value>
                                        </xsl:if>
                                    </swe:Category>
                                </swe:field>
                                <swe:field name="READGS_DetailsAvailable">
                                    <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::READGS:DetailsAvailable">
                                        <xsl:variable name="detailsAvailable"
select="$data/@DetailsAvailable"/>
                                        <xsl:choose>
                                            <xsl:when test="$detailsAvailable">
                                                <swe:value>
                                                    <xsl:value-of
select="$detailsAvailable"/>
                                                </swe:value>
                                            </xsl:when>
                                            <xsl:otherwise>
                                                <swe:value>false</swe:value>
                                            </xsl:otherwise>
                                        </xsl:choose>
                                    </swe:Boolean>
                                </swe:field>
                                <!--TODO:  Insert Spectrum and Wx info. here-->
                            </xsl:variable>
                            <xsl:variable name="alertFields">
                                <xsl:copy-of select="$modeField"/>
                                <swe:field name="ALERTS_Event">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Event">
                                        <swe:constraint>
                                            <swe:AllowedTokens>
                                                <swe:valueList>ALERT DEALERT WARN
DEWARN NONE</swe:valueList>
                                            </swe:AllowedTokens>
                                        </swe:constraint>
                                        <swe:value>
                                            <xsl:value-of select="$alertsChn/@Event"/>
                                        </swe:value>
                                    </swe:Category>
                                </swe:field>
                                <swe:field name="ALERTS_Source">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Source">
                                        <swe:constraint>
                                            <swe:AllowedTokens>
                                                <swe:valueList>none detector bit
tamper</swe:valueList>
                                            </swe:AllowedTokens>
                                        </swe:constraint>
                                        <swe:value>
                                            <xsl:value-of select="$alertsChn/@Source"/>
                                        </swe:value>
                                    </swe:Category>
                                </swe:field>
                                <!--Time and AlertId will be used to populate the gml:id
and om:samplingTime fields
            <swe:field name="ALERTS_Time">
              <swe:Time definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:Time"/>
            </swe:field>
            -->
                                <swe:field name="ALERTS_AlertId">
```

```
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:AlertId">
                                        <swe:value>
                                            <xsl:value-of select="$alertsChn/@AlertId"/>
                                        </swe:value>
                                    </swe:Category>
                                </swe:field>
                                <!--READGS Information-->
                                <xsl:copy-of select="$readingFields"/>
                                <!--BIT Information-->
                                <swe:field name="ALERTS_BIT_Component">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:Component">
                                        <swe:constraint>
                                            <swe:AllowedTokens>
                                                <swe:valueList>CC PDIL SC UIC WCC
DPC</swe:valueList>
                                            </swe:AllowedTokens>
                                        </swe:constraint>
                                        <swe:value>
                                            <xsl:value-of
select="$alertsChn/BIT/@Component"/>
                                        </swe:value>
                                    </swe:Category>
                                </swe:field>
                                <swe:field name="ALERTS_BIT_Executed">
                                    <swe:Time definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Executed" referenceTime="1970-01-01">
                                        <swe:uom code="s"/>
                                        <swe:value>
                                            <xsl:value-of
select="$alertsChn/BIT/@Executed"/>
                                        </swe:value>
                                    </swe:Time>
                                </swe:field>
                                <swe:field name="ALERTS_BIT_Result">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::ALERTS:BIT:Result">
                                        <swe:constraint>
                                            <swe:AllowedTokens>
                                                <swe:valueList>PASS
FAIL</swe:valueList>
                                            </swe:AllowedTokens>
                                        </swe:constraint>
                                        <swe:value>
                                            <xsl:value-of
select="$alertsChn/BIT/@Result"/>
                                        </swe:value>
                                    </swe:Category>
                                </swe:field>
                                <swe:field name="ALERTS_BIT_Level">
                                    <swe:Count definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:Level">
                                        <swe:value>
                                            <xsl:value-of
select="$alertsChn/BIT/@Level"/>
                                        </swe:value>
                                    </swe:Count>
                                </swe:field>
                                <swe:field name="ALERTS_BIT_ErrorDescription">
                                    <swe:Text definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:BIT:ErrorDescription">
                                        <swe:value>
                                            <xsl:value-of
select="$alertsChn/BIT/ErrorDescription"/>
                                        </swe:value>
                                    </swe:Text>
                                </swe:field>
                                <!--Tamper Information-->
                                <swe:field name="ALERTS_Tamper">
                                    <swe:Text definition="urn:ogc:def:phenomenon:JPEO-
CBD::ALERTS:Tamper">
```



```
                                        <swe:value>
                                            <xsl:value-of select="$alertsChn/Tamper"/>
                                        </swe:value>
                                    </swe:Text>
                                </swe:field>
                            </xsl:variable>
                            <xsl:variable name="statusFields">
                                <xsl:copy-of select="$modeField"/>
                                <swe:field name="STATUS_BIT">
                                    <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS:BIT">
                                        <swe:value>
                                            <xsl:value-of select="$statusChn/@BIT"/>
                                        </swe:value>
                                    </swe:Boolean>
                                </swe:field>
                                <swe:field name="STATUS_Alert">
                                    <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS:Alert">
                                        <swe:value>
                                            <xsl:value-of select="$statusChn/@Alert"/>
                                        </swe:value>
                                    </swe:Boolean>
                                </swe:field>
                                <swe:field name="STATUS_Maint">
                                    <swe:Boolean
definition="urn:ogc:def:phenomenon:JPEO-CBD::STATUS:Maint">
                                        <swe:value>
                                            <xsl:value-of select="$statusChn/@Maint"/>
                                        </swe:value>
                                    </swe:Boolean>
                                </swe:field>
                            </xsl:variable>
                            <xsl:variable name="maintFields">
                                <xsl:copy-of select="$modeField"/>
                                <swe:field name="MAINT_Type">
                                    <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:Type">
                                        <swe:value>
                                            <xsl:value-of select="$maintChn/@Type"/>
                                        </swe:value>
                                    </swe:Category>
                                </swe:field>
                                <xsl:choose>
                                    <xsl:when test="$maintType='periodic'">
                                        <swe:field name="MAINT_Periodic_Time">
                                            <swe:Time
definition="urn:ogc:data:time:iso8601">
                                                <swe:value>
                                                    <xsl:value-of select="$dataTime"/>
                                                </swe:value>
                                            </swe:Time>
                                        </swe:field>
                                    </xsl:when>
                                    <xsl:otherwise>
                                        <swe:field name="MAINT_BIT_Result">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:BIT:Result">
                                                <swe:value>
                                                    <xsl:value-of
select="$maintChn/BIT/@Result"/>
                                                </swe:value>
                                            </swe:Category>
                                        </swe:field>
                                        <xsl:variable name="extime"
select="$maintChn/BIT/@Executed"/>
                                        <xsl:variable name="exyear"
select="substring($extime,1,4)"/>
                                        <xsl:variable name="exmonth"
select="substring($extime,5,2)"/>
                                        <xsl:variable name="exday"
select="substring($extime,7,2)"/>
```

```
                                        <xsl:variable name="exhour"
select="substring($extime,9,2)"/>
                                        <xsl:variable name="exminute"
select="substring($extime,11,2)"/>
                                        <xsl:variable name="exsecond"
select="substring($extime,13,2)"/>
                                        <xsl:variable name="exdataTime">
                                            <xsl:choose>
                                                <xsl:when test="$extime">
                                                    <xsl:value-of
select="concat($exyear,'-',$exmonth,'-
',$exday,'T',$exhour,':',$exminute,':',$exsecond,'Z')"/>
                                                </xsl:when>
                                                <xsl:otherwise>
                                                    <xsl:value-of select="$dtgTime"/>
                                                </xsl:otherwise>
                                            </xsl:choose>
                                        </xsl:variable>
                                        <swe:field name="MAINT_BIT_Executed">
                                            <swe:Time
definition="urn:ogc:data:time:iso8601">
                                                <swe:value>
                                                    <xsl:value-of
select="$exdataTime"/>
                                                </swe:value>
                                            </swe:Time>
                                        </swe:field>
                                        <swe:field name="MAINT_BIT_Component">
                                            <swe:Category
definition="urn:ogc:def:phenomenon:JPEO-CBD::MAINT:BIT:Component">
                                                <swe:value>
                                                    <xsl:value-of
select="$maintChn/BIT/@Component"/>
                                                </swe:value>
                                            </swe:Category>
                                        </swe:field>
                                    </xsl:otherwise>
                                </xsl:choose>
                            </xsl:variable>
                            <xsl:choose>
                                <xsl:when test="$modifiedChannel='READGS'">
                                    <xsl:copy-of select="$readingFields"/>
                                </xsl:when>
                                <xsl:when test="$modifiedChannel='ALERTS'">
                                    <xsl:copy-of select="$alertFields"/>
                                </xsl:when>
                                <xsl:when test="$modifiedChannel='STATUS'">
                                    <xsl:copy-of select="$statusFields"/>
                                </xsl:when>
                                <xsl:when test="$modifiedChannel='MAINT'">
                                    <xsl:copy-of select="$maintFields"/>
                                </xsl:when>
                                <xsl:when test="$chn='h'">
                                    <xsl:copy-of select="$heartbeatFields"/>
                                </xsl:when>
                            </xsl:choose>
                            <xsl:if test="$sudElements">
                                <swe:field name="Sensor Unique Data">
                                    <swe:DataRecord
definition="{concat('urn:ogc:def:phenomenon:JPEO-CBD::',$modifiedChannel,':SUD')}">
                                        <xsl:for-each select="$sudElements">
                                            <xsl:variable name="name" select="@Name"/>
                                            <xsl:variable name="type" select="@Type"/>
                                            <xsl:variable name="definition"
select="concat('urn:ogc:def:phenomenon:JPEO-CBD::',$modifiedChannel,':',$name)"/>
                                            <xsl:variable name="uom" select="@Units"/>
                                            <xsl:variable name="uomElement">
                                                <swe:uom code="{$uom}"/>
                                            </xsl:variable>
                                            <xsl:variable name="value" select="@Value"/>
                                            <xsl:variable name="valueElement">
                                                <swe:value>
```



```
                                                    <xsl:value-of select="$value"/>
                                                </swe:value>
                                            </xsl:variable>
                                            <swe:field name="{$name}">
                                                <xsl:choose>
                                                    <xsl:when test="$type='Integer' or
$type='Float'">
                                                        <swe:Quantity
definition="{$definition}">
                                                            <xsl:if test="$uom">
                                                                <xsl:copy-of
select="$uomElement"/>
                                                            </xsl:if>
                                                            <xsl:copy-of
select="$valueElement"/>
                                                        </swe:Quantity>
                                                    </xsl:when>
                                                    <xsl:otherwise>
                                                        <swe:Category
definition="{$definition}">
                                                            <xsl:copy-of
select="$valueElement"/>
                                                        </swe:Category>
                                                    </xsl:otherwise>
                                                </xsl:choose>
                                            </swe:field>
                                        </xsl:for-each>
                                    </swe:DataRecord>
                                </swe:field>
                            </xsl:if>
                        </swe:DataRecord>
                    </xsl:otherwise>
                </xsl:choose>
            </result>
        </Observation>
    </xsl:template>
</xsl:stylesheet>
```

C.4 CCSI Command Definitions to SPS 1.0.0 DescribeTasking Response

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:ccsi="http://www.ccsi.org/schema/CCSI">
    <xsl:param name="sensorID">The sensor ID for the requested sensor</xsl:param>
    <xsl:param name="supportedCommands">The list of supported commands for the
sensor</xsl:param>
    <xsl:param name="filteredCommands">The list of commands to filter</xsl:param>
    <xsl:template match="/">
        <DescribeTaskingRequestResponse
xsi:schemaLocation="http://www.opengis.net/sps/1.0
http://schemas.opengis.net/sps/1.0.0/spsAll.xsd" xmlns="http://www.opengis.net/sps/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:swe="http://www.opengis.net/swe/1.0" xmlns:gml="http://www.opengis.net/gml">
            <xsl:for-each select="//ccsi:CCSI Std Command">
                <xsl:variable name="commandName" select="@name"/>
                <xsl:if test="contains($supportedCommands,$commandName) and
not(contains($filteredCommands,$commandName))">
                    <taskingDescriptor>
                        <sensorID>
                            <xsl:value-of select="$sensorID"/>
                        </sensorID>
                        <gml:description>
                            <xsl:value-of select="$commandName"/> Command:  <xsl:value-
of select="./ccsi:HelpText"/>
                        </gml:description>
                        <xsl:variable name="arguments"
select="./ccsi:Arguments/ccsi:Argument | .//ccsi:RepeatElement"/>
                        <xsl:choose>
                            <xsl:when test="$arguments">
                                <xsl:for-each select="$arguments">
                                    <xsl:variable name="use">
```



```
                                        <xsl:choose>
                                            <xsl:when
test="@mandatory='false'">optional</xsl:when>
                                            <xsl:otherwise>required</xsl:otherwise>
                                        </xsl:choose>
                                    </xsl:variable>
                                    <xsl:variable name="type" select="@type"/>
                                    <xsl:variable name="helpText"
select="./ccsi:HelpText"/>
                                    <InputDescriptor
parameterID="{concat($commandName,'-',@name)}" updateable="true" use="{$use}">
                                        <xsl:if test="$helpText">
                                            <gml:description>
                                                <xsl:value-of select="$helpText"/>
                                            </gml:description>
                                        </xsl:if>
                                        <definition>
                                            <commonData>
                                                <xsl:variable name="defaultValue"
select="./ccsi:DefaultValue"/>
                                                <xsl:variable name="dataElement"
select="//ccsi:DataElement[@name=$type]"/>
                                                <xsl:variable name="dataElementType"
select="$dataElement/@type"/>
                                                <xsl:variable name="uom"
select="$dataElement/ccsi:UnitOfMeasure"/>
                                                <xsl:variable name="definition"
select="concat('urn:ogc:def:parameter:JPEO-CBD::',$commandName,':',@name)"/>
                                                <xsl:choose>
                                                    <xsl:when
test="$dataElementType='int' or $dataElementType='float' or $dataElementType='short' or
$dataElementType='time_period' or $dataElementType='long'">
                                                        <swe:Quantity
definition="{$definition}">
                                                            <xsl:if test="$uom">
                                                                <swe:uom
code="{$dataElement/ccsi:UnitOfMeasure}"/>
                                                            </xsl:if>
                                                            <xsl:variable name="range"
select="$dataElement/ccsi:ValidityCheck/ccsi:Range"/>
                                                            <xsl:variable name="min"
select="normalize-space(substring-before($range,' - '))"/>
                                                            <xsl:variable name="max"
select="translate(normalize-space(substring-after($range,' - ')),'+','')"/>
                                                            <xsl:if test="$min or
$max">
                                                                <swe:constraint>
                                                                    <swe:AllowedValues>
                                                                        <xsl:choose>
                                                                            <xsl:when
test="$min and $max">
    <swe:interval>
    <xsl:value-of select="concat($min,' ',$max)"/>
    </swe:interval>
    </xsl:when>
                                                                            <xsl:when
test="$min">
    <swe:min>
    <xsl:value-of select="$min"/>
    </swe:min>
    </xsl:when>
                                                                            <xsl:when
test="$max">
```



```
    <swe:max>
    <xsl:value-of select="$max"/>
    </swe:max>
    </xsl:when>
                                                                    </xsl:choose>
    </swe:AllowedValues>
                                                                </swe:constraint>
                                                            </xsl:if>
                                                            <xsl:if
test="$defaultValue">
                                                                <swe:value>
                                                                    <xsl:value-of
select="$defaultValue"/>
                                                                </swe:value>
                                                            </xsl:if>
                                                        </swe:Quantity>
                                                    </xsl:when>
                                                    <xsl:when
test="$dataElementType='boolean'">
                                                        <swe:Boolean>
                                                            <xsl:if
test="$defaultValue">
                                                                <swe:value>
                                                                    <xsl:value-of
select="$defaultValue"/>
                                                                </swe:value>
                                                            </xsl:if>
                                                        </swe:Boolean>
                                                    </xsl:when>
                                                    <xsl:when
test="$dataElementType='enum' or $dataElementType='string'">
                                                        <xsl:variable name="enumItems"
select="$dataElement//ccsi:Item"/>
                                                        <swe:Category
definition="{$definition}">
                                                            <xsl:if test="$enumItems">
                                                                <swe:constraint>
                                                                    <swe:AllowedTokens>
                                                                        <xsl:for-each
select="$enumItems">
    <swe:valueList>
    <xsl:value-of select="."/>
    </swe:valueList>
                                                                        </xsl:for-
each>

    </swe:AllowedTokens>
                                                                </swe:constraint>
                                                            </xsl:if>
                                                            <xsl:if
test="$defaultValue">
                                                                <swe:value>
                                                                    <xsl:value-of
select="$defaultValue"/>
                                                                </swe:value>
                                                            </xsl:if>
                                                        </swe:Category>
                                                    </xsl:when>
                                                    <xsl:otherwise>
                                                        <swe:Boolean>
                                                            <swe:value>true</swe:value>
                                                        </swe:Boolean>
```

```
                                                    </xsl:otherwise>
                                                </xsl:choose>
                                            </commonData>
                                        </definition>
                                    </InputDescriptor>
                                </xsl:for-each>
                            </xsl:when>
                            <xsl:otherwise>
                                <InputDescriptor parameterID="{$commandName}"
updateable="false" use="required">
                                    <gml:description>This is a default
parameter</gml:description>
                                    <definition>
                                        <commonData>
                                            <swe:Boolean>
                                                <swe:value>true</swe:value>
                                            </swe:Boolean>
                                        </commonData>
                                    </definition>
                                </InputDescriptor>
                            </xsl:otherwise>
                        </xsl:choose>
                    </taskingDescriptor>
                </xsl:if>
            </xsl:for-each>
        </DescribeTaskingRequestResponse>
    </xsl:template>
</xsl:stylesheet>
```

C.5 SPS 1.0.0 Submit Command to CCSI Command

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:sps="http://www.opengis.net/sps/1.0"
xmlns:swe="http://www.opengis.net/swe/1.0">
    <xsl:param name="messageNumber"/>
    <xsl:param name="dateTimeGroup"/>
    <xsl:template match="/">
        <xsl:variable name="delimiter" select="'-'"/>
        <xsl:variable name="sensorID" select="//sps:sensorID"/>
        <xsl:variable name="sid" select="substring-after($sensorID,'::')"/>
        <xsl:variable name="firstParameter" select="//sps:InputParameter/@parameterID"/>
        <xsl:variable name="command" select="substring-
before($firstParameter,$delimiter)"/>
        <Hdr xmlns="" sid="{$sid}" msn="{$messageNumber}" dtg="{$dateTimeGroup}" chn="c"
mod="N" len="1419">
            <CCSIDoc>
                <Msg>
                    <CmdChn Cmd="{$command}">
                        <xsl:for-each select="//sps:InputParameter">
                            <xsl:variable name="parameterID" select="@parameterID"/>
                            <xsl:variable name="argName" select="substring-
after($parameterID,$delimiter)"/>
                            <xsl:variable name="argValue"
select="./sps:value/*/swe:value"/>
                            <xsl:if test="string-length(normalize-space($argValue))>0">
                                <Arg>
                                    <ArgName>
                                        <xsl:value-of select="$argName"/>
                                    </ArgName>
                                    <ArgValue>
                                        <xsl:value-of select="$argValue"/>
                                    </ArgValue>
                                </Arg>
                            </xsl:if>
                        </xsl:for-each>
                    </CmdChn>
                </Msg>
            </CCSIDoc>
        </Hdr>
    </xsl:template>
```



```
</xsl:stylesheet>
```

Bibliography

[1] Guidelines for Successful OGC Interface Standards, OGC document 00-014r1

[2] OWS-7 CCSI-SWE Engineering Report, OGC document 09-007

