OGC Sensor Web Enablement (SWE)

26th MEETING RTCA Special Committee 206 Plenary Aeronautical Information and Meteorological Data Link Services

Luis Bermudez (OGC)

lbermudez@opengeospatial.org

Washington, September 20, 2011

OGC®

© 2011, Open Geospatial Consortium

Sensors are everywhere

And .. 4 billion mobile devices can act as sensors
http://research.ict.csiro.au/conferences/ssn/ssn11

OGC®

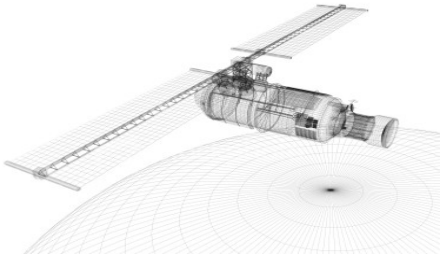
Copyright © 2011, Open Geospatial Consortium

… disconnected

Copyright © 2011, Open Geospatial Consortium

Standalone Sensors

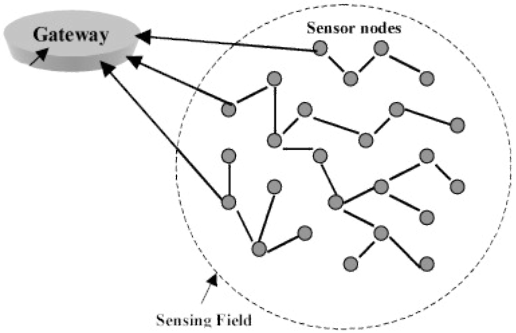
- Deployed for single purpose or function
- Integration Difficulties
  - Heterogeneous platforms
  - Disparate data sets
  - Organisationally bound
- Isolated observations



OGC®

Networked Sensors

- Allows remote access to sensors for:
  - data collection
    responsiveness
  - tasking
- Observe phenomena more completely



OGC®

SWE as a Solution

- How do I access the data
- Where is the existing data (Discovery)
- I need to create new sensor data
- What's the lineage of data (Provenance)

OGC®

Copyright © 2011, Open Geospatial Consortium

SWE as a Solution

- Error bars associated with data
- Integration of data
  - Different formats
  - Little known formats
- Real time access to data

OGC®

Copyright © 2011, Open Geospatial Consortium

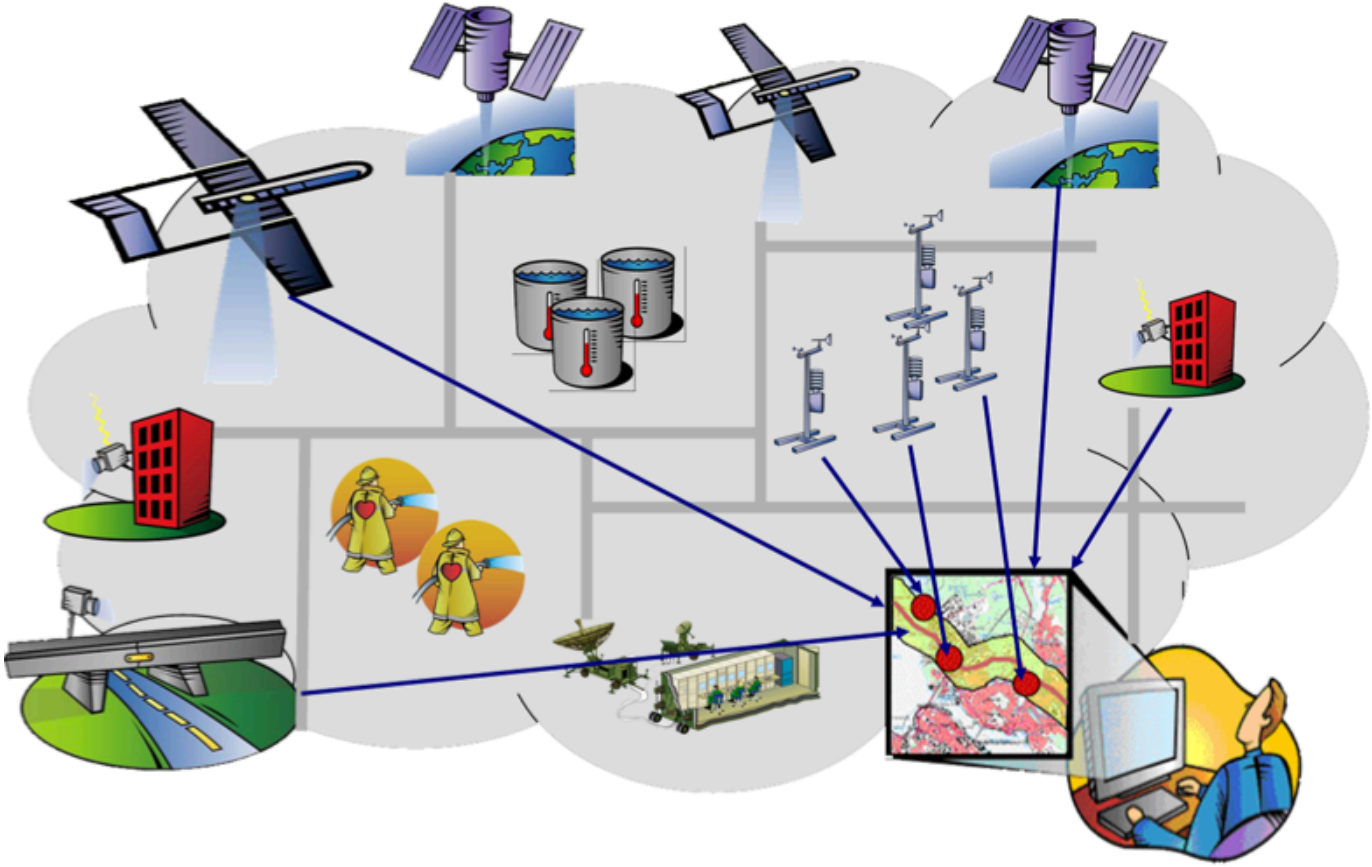
SWE as Solution

- Not enough data (Sparse)
- Too much data (Overload)
  - Subsetting
  - Filtering
- Push the data to me when its becomes available

OGC®

Copyright © 2011, Open Geospatial Consortium

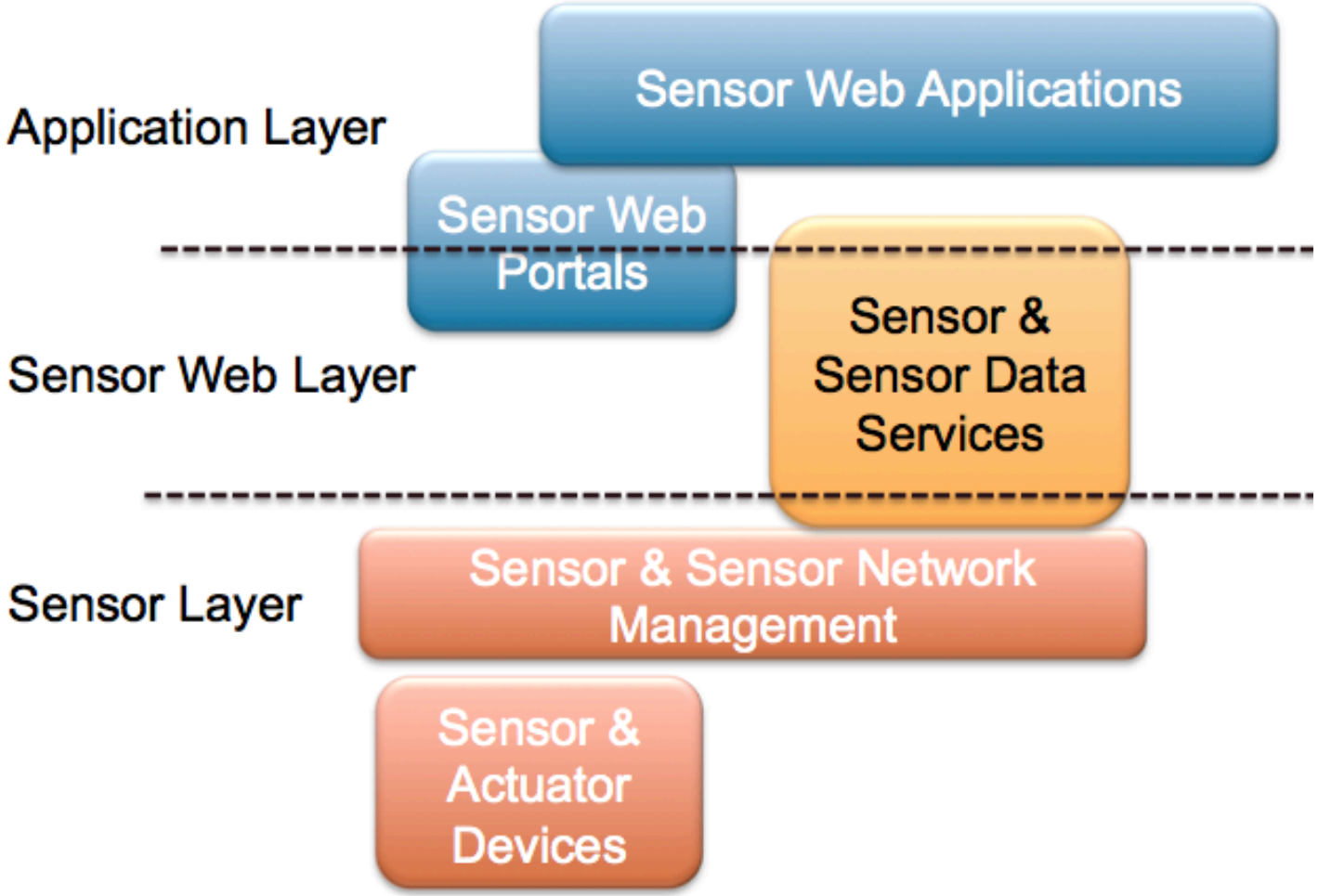
OGC Sensor Web



OGC®

Copyright © 2011, Open Geospatial Consortium

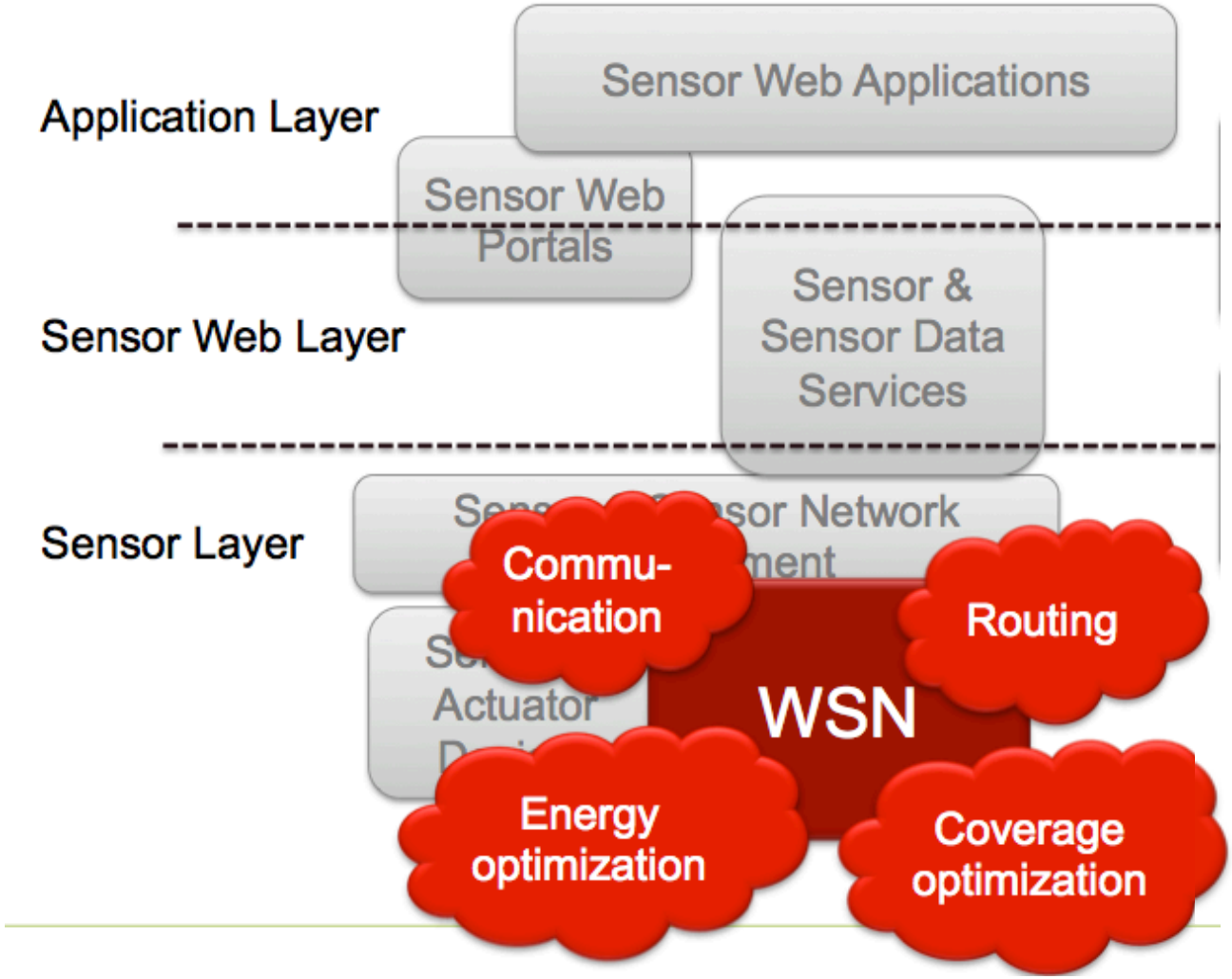
From Sensors to Applications



OGC®

Copyright © 2011, Open Geospatial Consortium

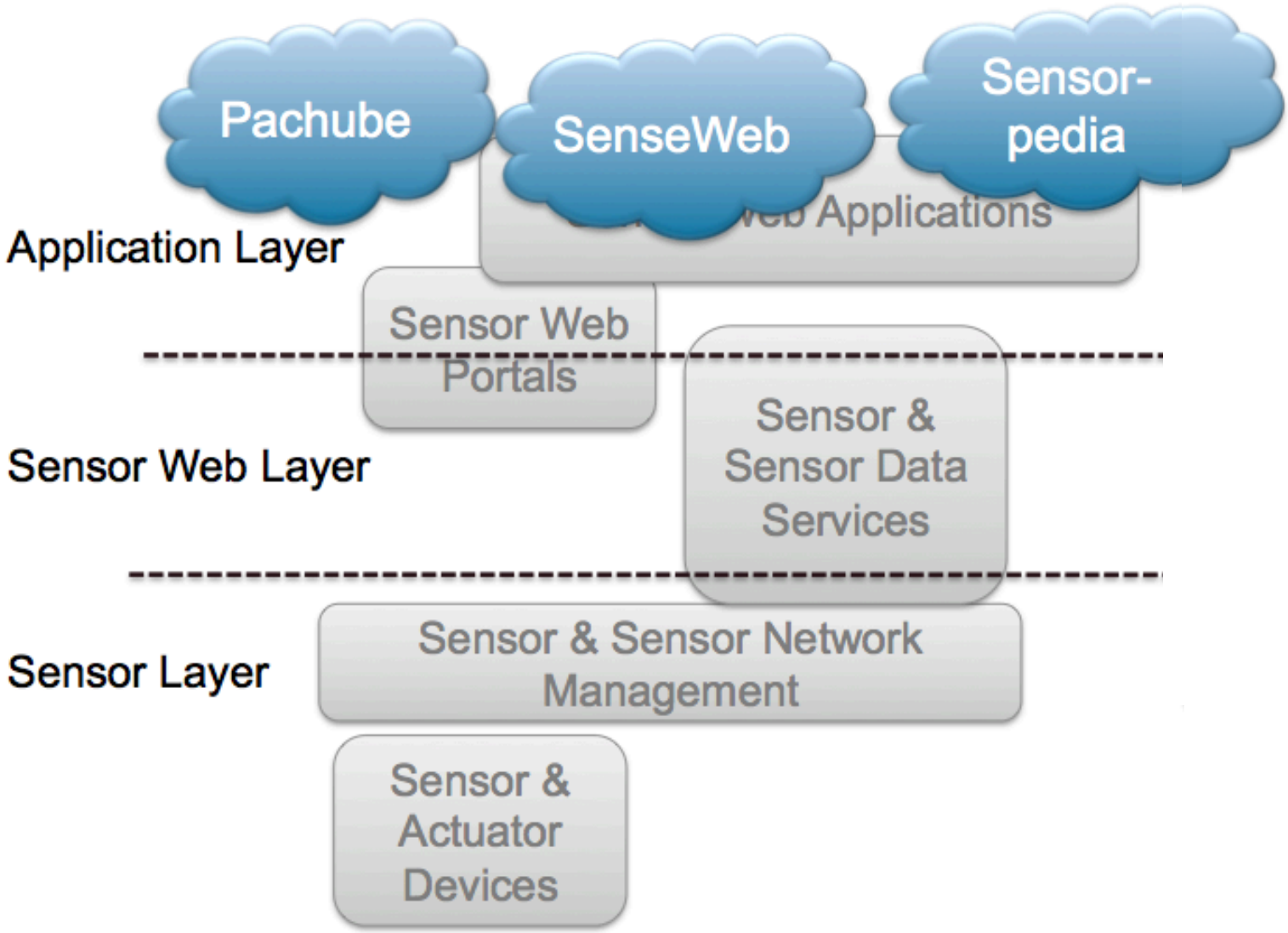
Wireless Sensor Networks (WNS) Focus



OGC®

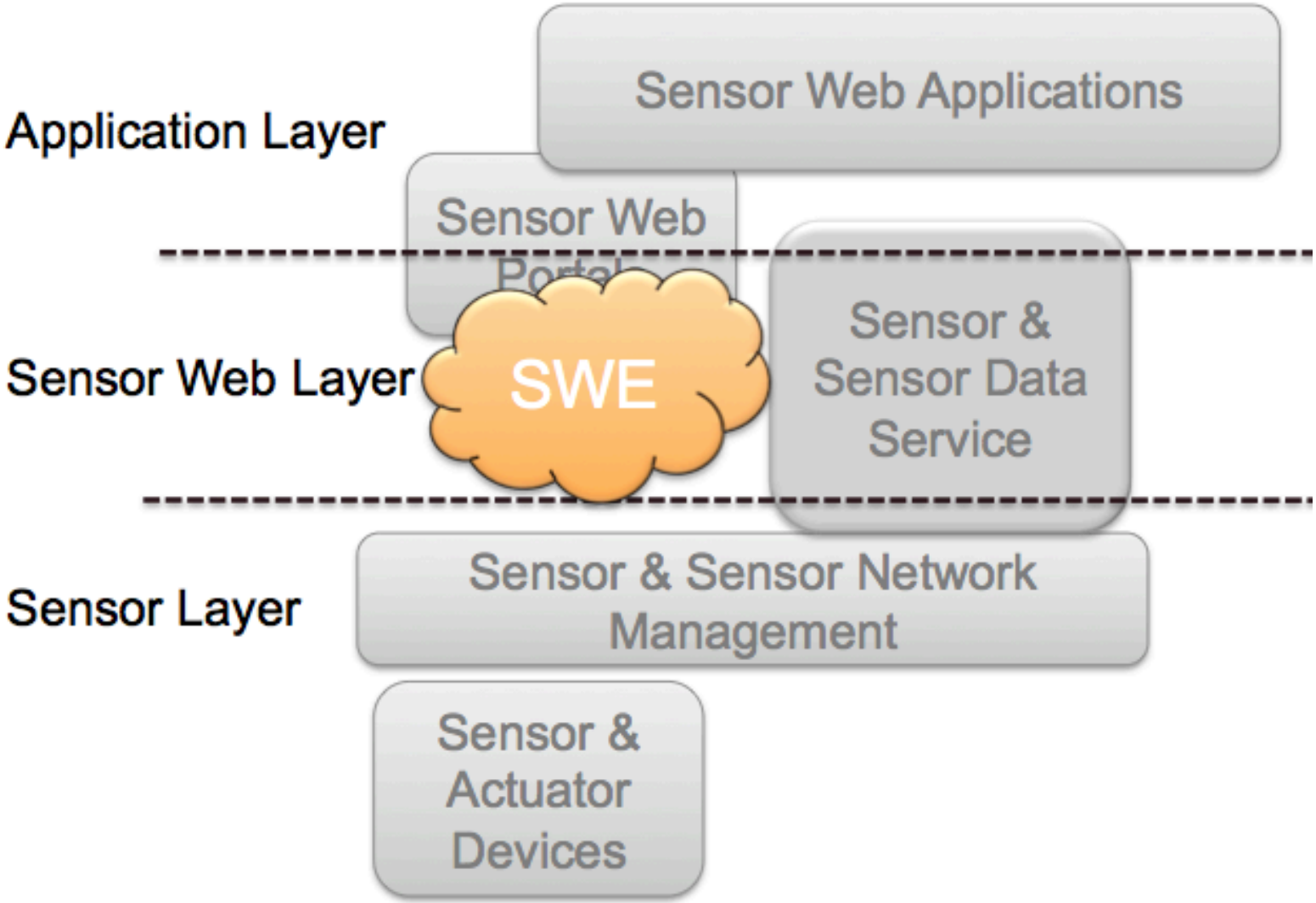
Copyright © 2011, Open Geospatial Consortium

Portal Focus



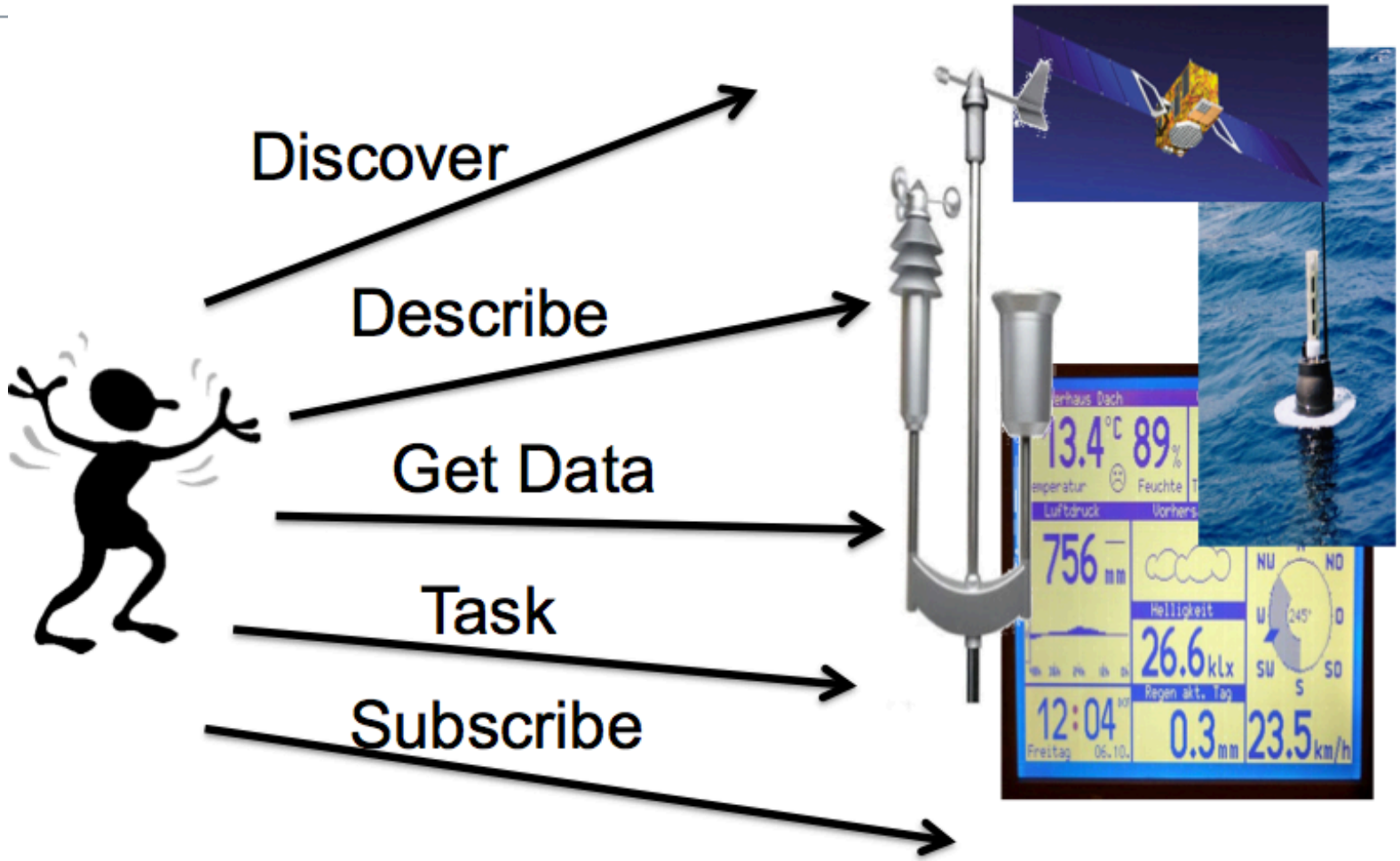
Copyright © 2011, Open Geospatial Consortium

Sensor Web Layer Focus



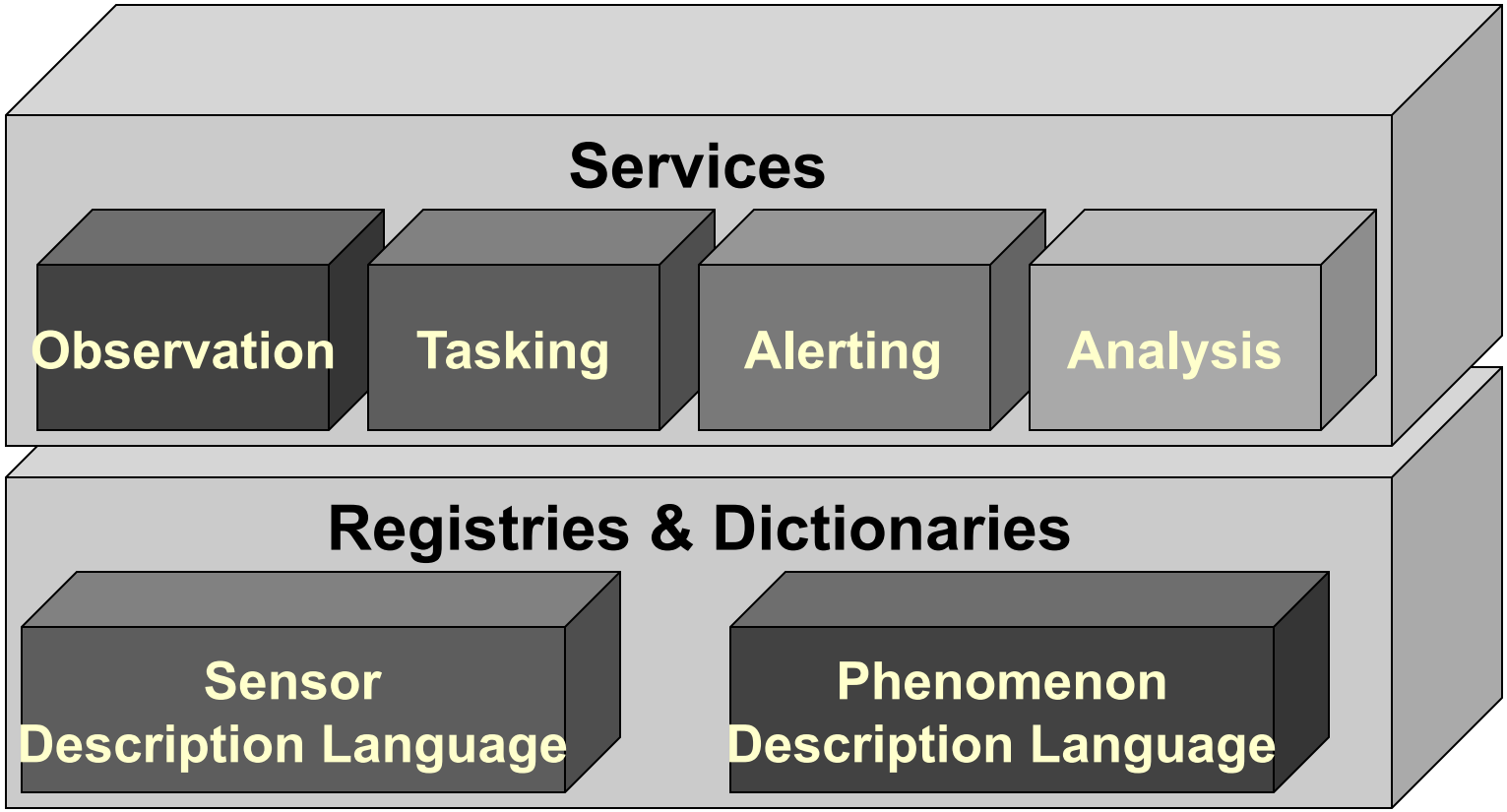
Copyright © 2011, Open Geospatial Consortium

Sensor Web Requirements



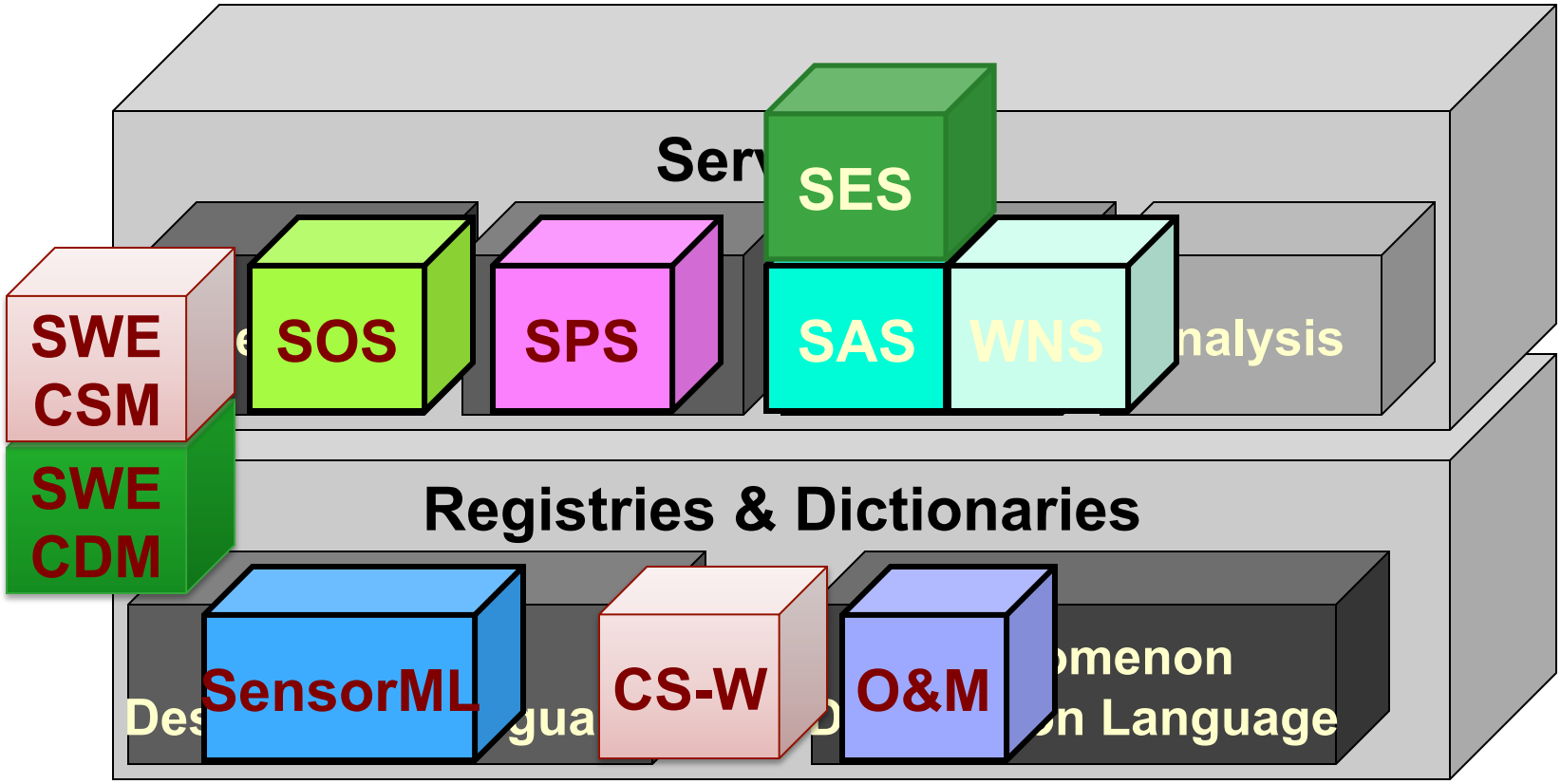
Copyright © 2011, Open Geospatial Consortium

Building Blocks: OGC SWE 2.0



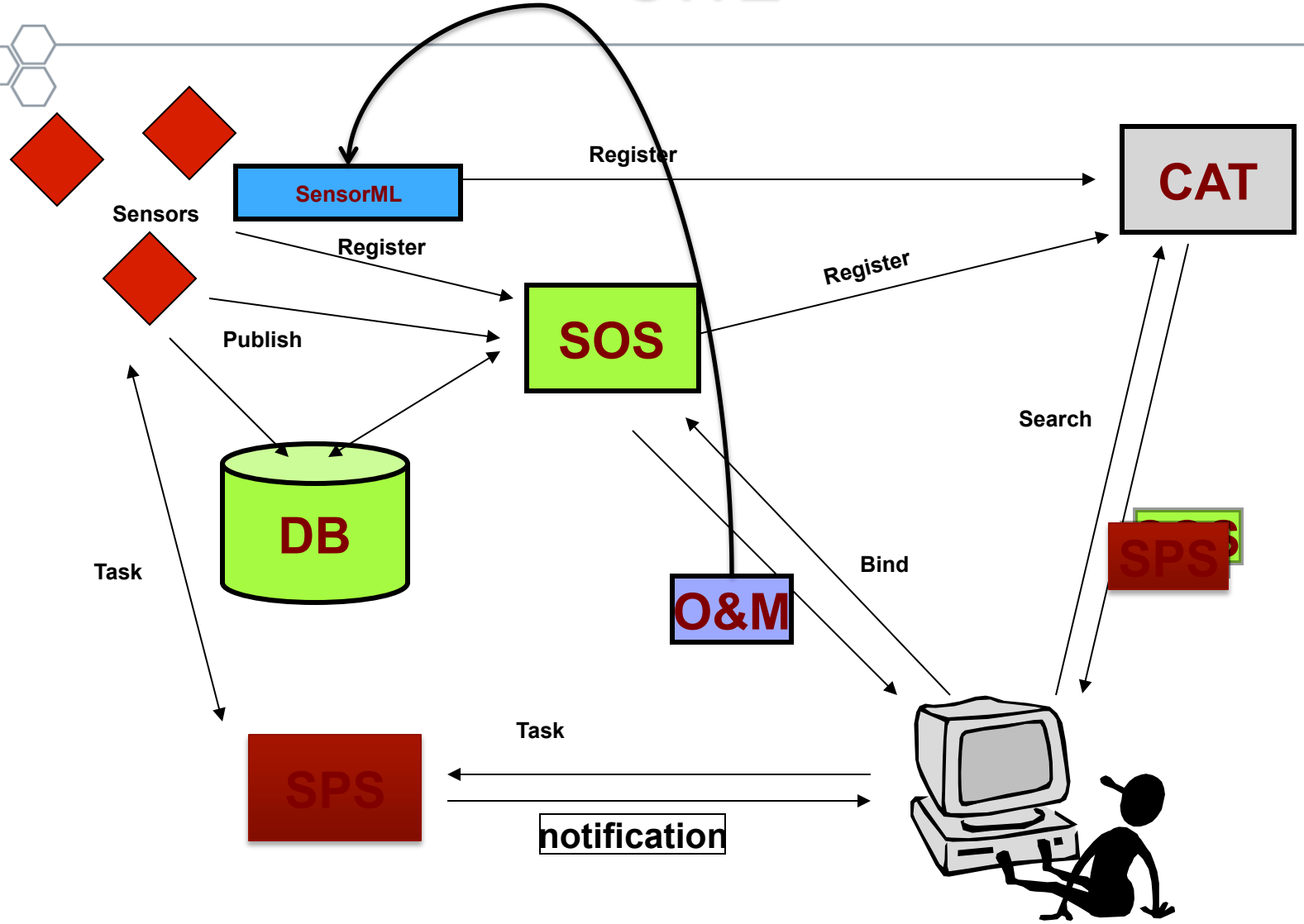
OGC®

Building Blocks: OGC SWE **2.0**



OGC®

SWE


Sensors
SensorML
Register
CAT
Register
Register
SOS
Publish
Search
DB
Task
Bind
O&M
SPS
Task
SPS
notification

OGC®

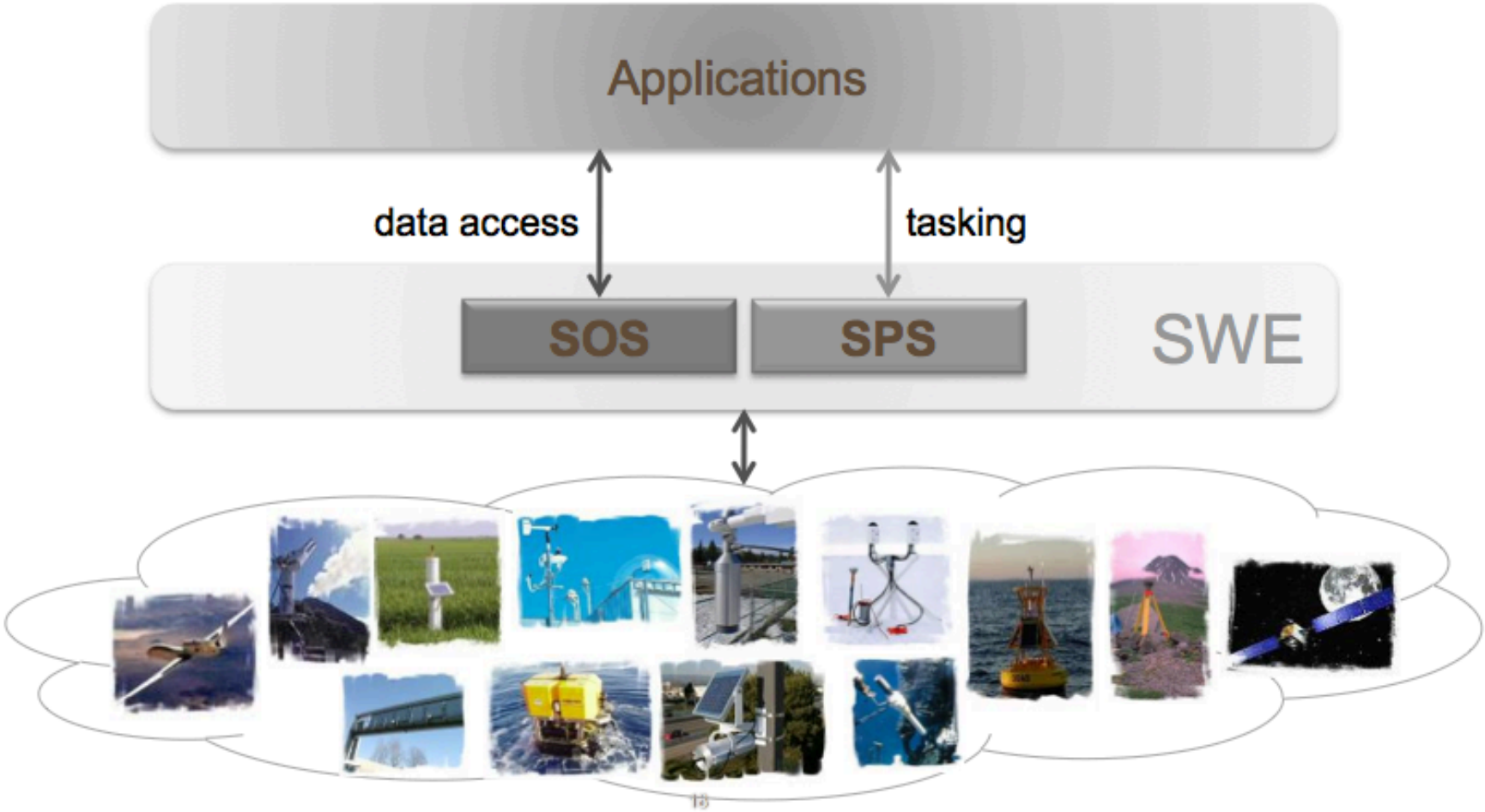
SWE Services

- XML/SOAP over HTTP
- RESTful in discussion
- Early: Provide rich data - trust users to filter
- Now: Start thinking about pre-organization of data
- All services are “self-descriptive”
  - SOS/SPS: Spatio/temporal features, observed feature, phenomenon, sensor…

OGC®

Copyright © 2011, Open Geospatial Consortium

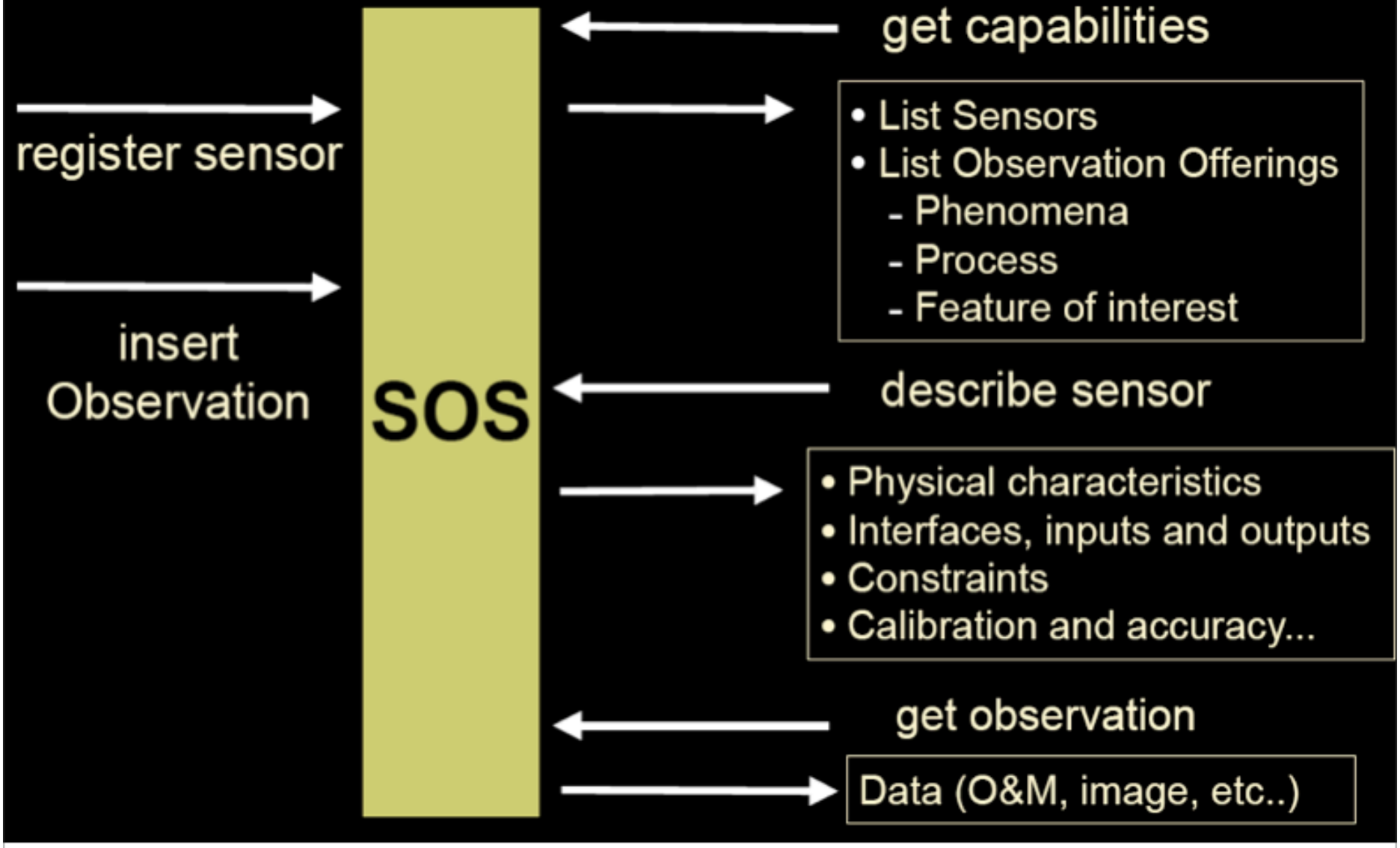
SOS & SPS: Data Access & Control



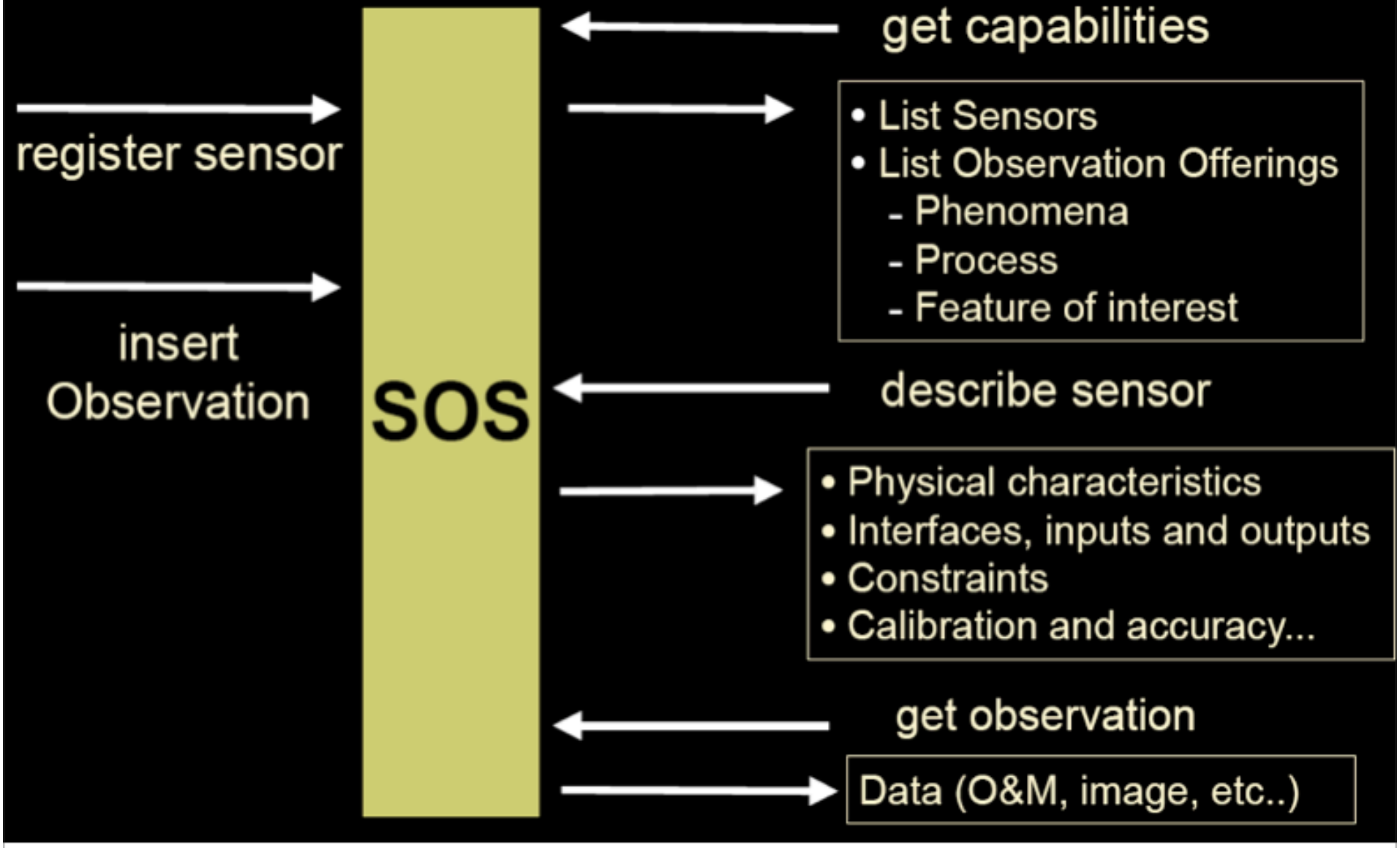
OGC®

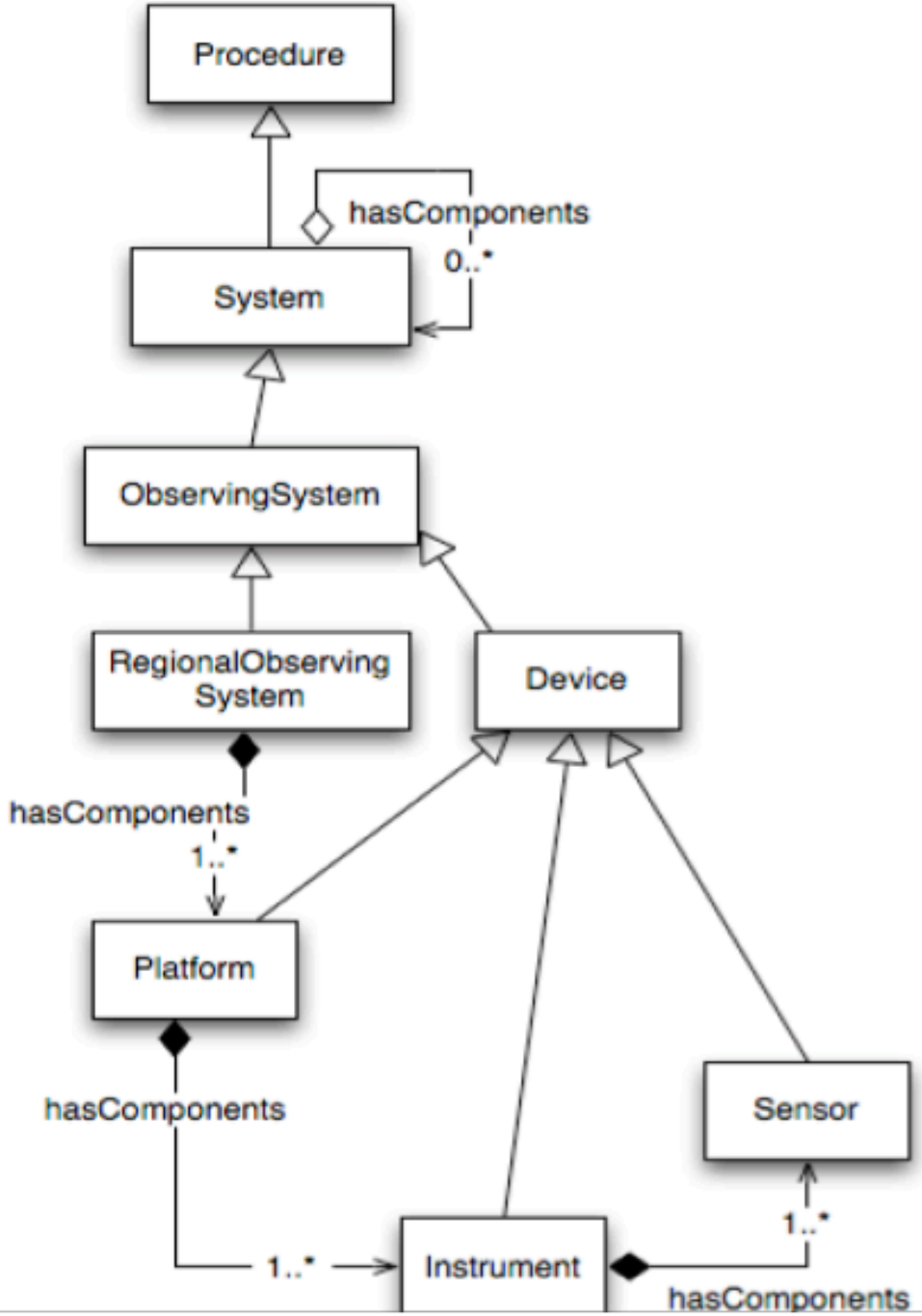
Copyright © 2011, Open Geospatial Consortium

Sensor Observation Service



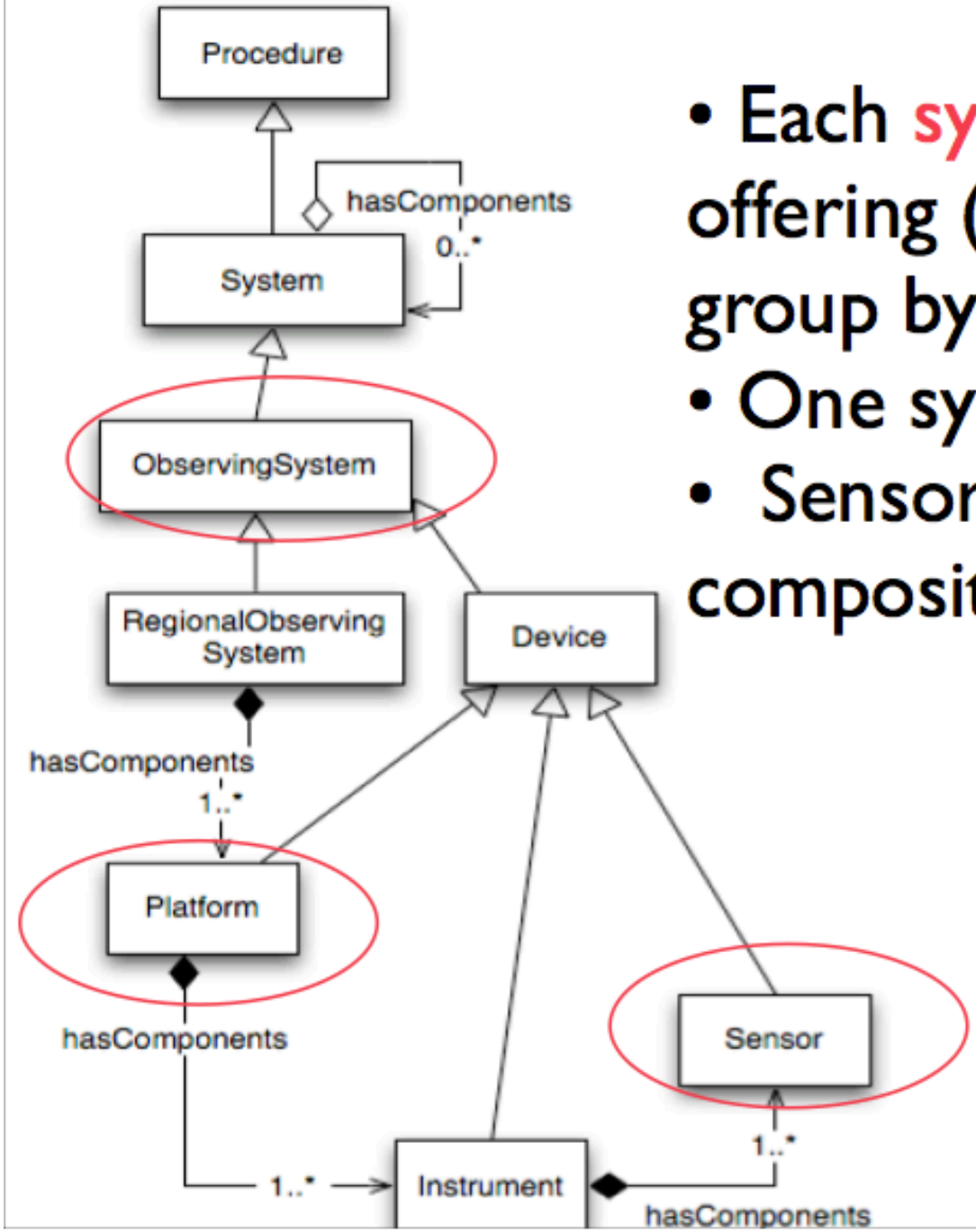
System Observation Service





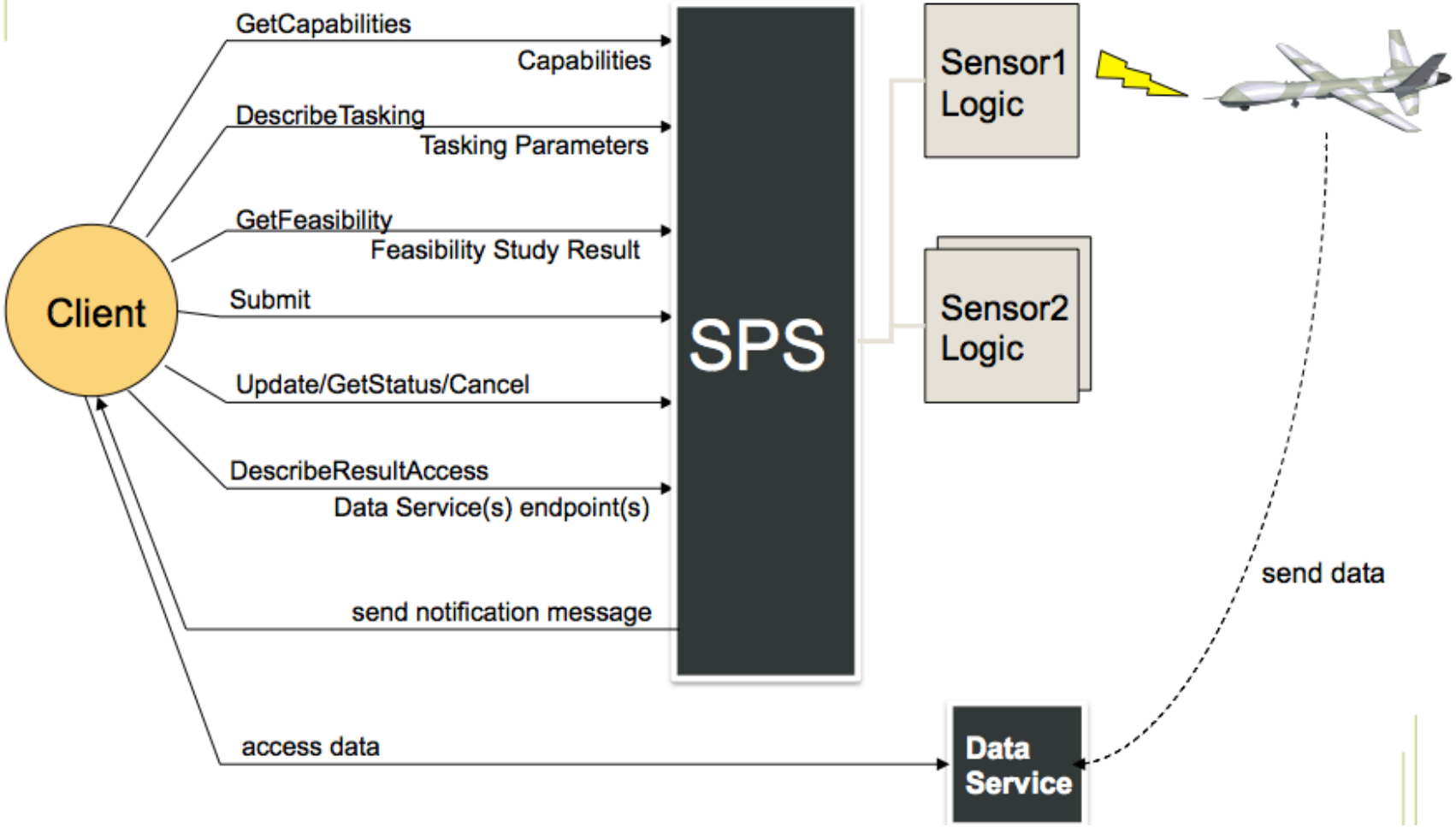
Complex systems

Lead by Luis Bermudez (SURA)



- Each system has its own offering (if too big then group by regions)
- One system per offering
- SensorML describes the composition

SPS: Tasking Sensors/Actuators



Copyright © 2011, Open Geospatial Consortium

	0	1	2	3	4	5	6	7	A	B	C	D	E	F
0	NUL	DLE	SP	0	@	P	`	p	NBSP	°	À	Ð	à	ð
1	SOH	DC1	!	1	A	Q	a	q	¡	±	Á	Ñ	á	ñ
2	STX	DC2	"	2	B	R	b	r	¢	²	Â	Ò	â	ò
3	ETX	DC3	#	3	C	S	c	s	£	³	Ã	Ó	ã	ó
4	EOT	DC4	$	4	D	T	d	t	¤	´	Ä	Ô	ä	ô
5	ENQ	NAK	%	5	E	U	e	u	¥	µ	Å	Õ	å	õ
6	ACK	SYN	&	6	F	V	f	v	¦	¶	Æ	Ö	æ	ö
7	BEL	ETB	'	7	G	W	g	w	§	·	Ç	×	ç	÷
8	BS	CAN	(	8	H	X	h	x	¨	¸	È	Ø	è	ø
9	HT	EM	)	9	I	Y	i	y	©	¹	É	Ù	é	ù
A	NL	SUB	*	:	J	Z	j	z	ª	º	Ê	Ú	ê	ú
B	VT	ESC	+	;	K	[	k	{	«	»	Ë	Û	ë	û
C	NP	FS	,	<	L	\	l	|	¬	¼	Ì	Ü	ì	ü
D	CR	GS	-	=	M	]	m	}		½	Í	Ý	í	ý
E	SO	RS	.	>	N	^	n	~	®	¾	Î	Þ	î	þ
F	SI	US	/	?	O	_	o	DEL	¯	¿	Ï	ß	ï	ÿ

SWE Encodings

SENSOR ML, O&M, SWE COMMON

OGC®

Copyright © 2011, Open Geospatial Consortium

Data Exchange Challenges

Generality

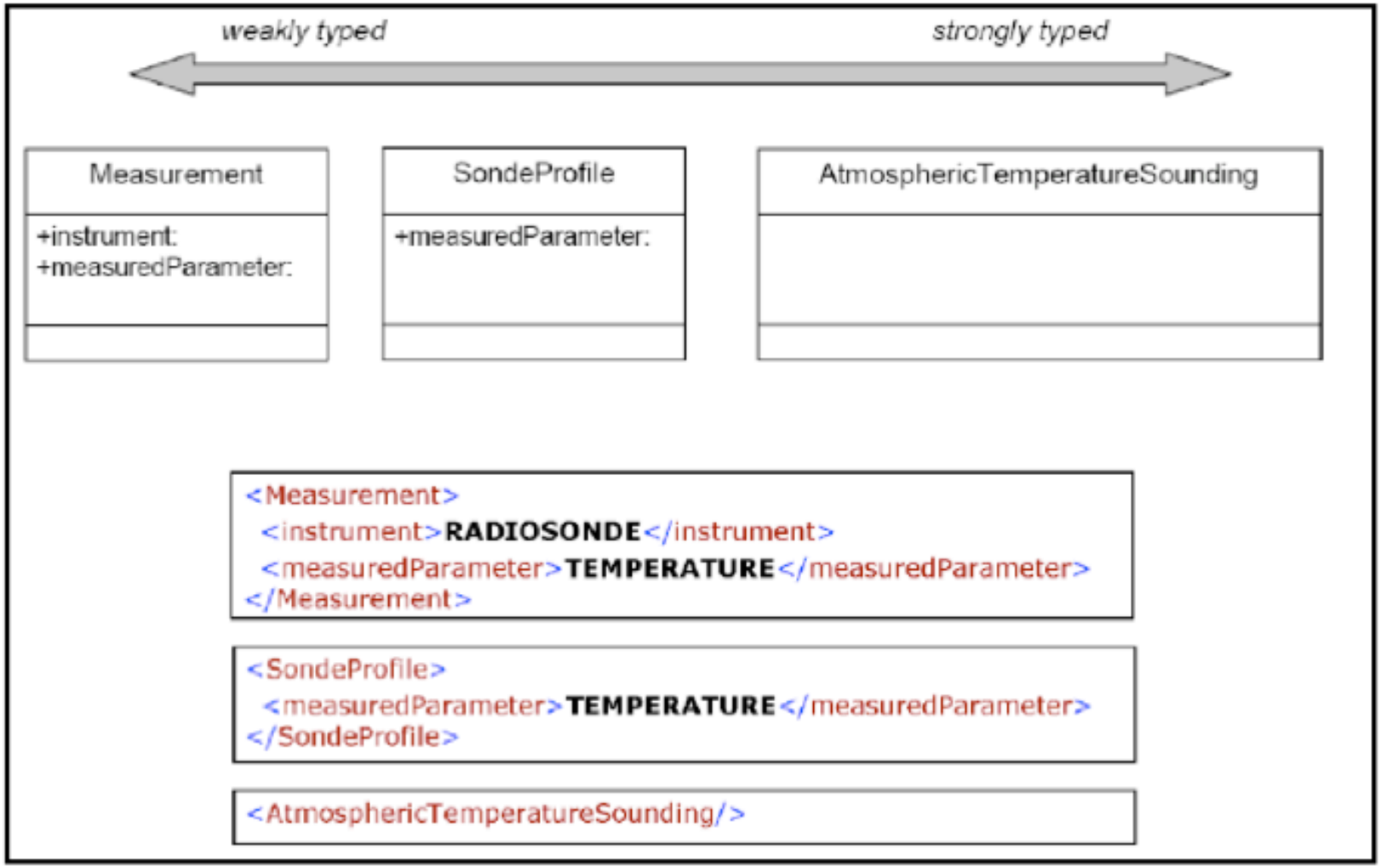
Interoperability

Soft-typed

hard-typed

OGC®

Copyright © 2011, Open Geospatial Consortium



What is the level of granularity ?

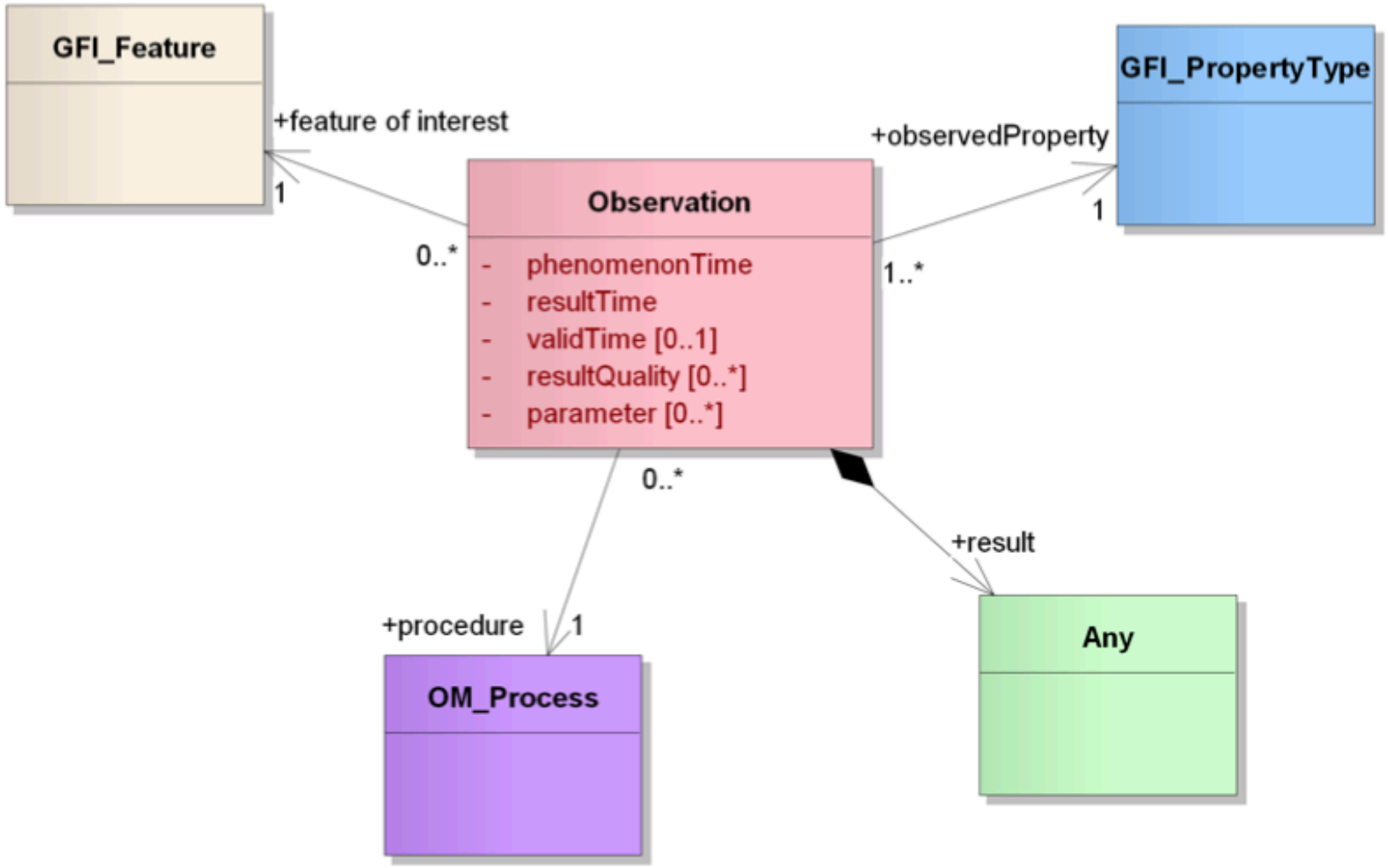
Data Structure & Encoding

- SweCommon:
  - Mixture of hard/soft-typed approaches
  - Allows to describe structure
  - Encodes values independently
- Allows for referenced and inline data

OGC®

Copyright © 2011, Open Geospatial Consortium

Observations and Measurement (O&M)

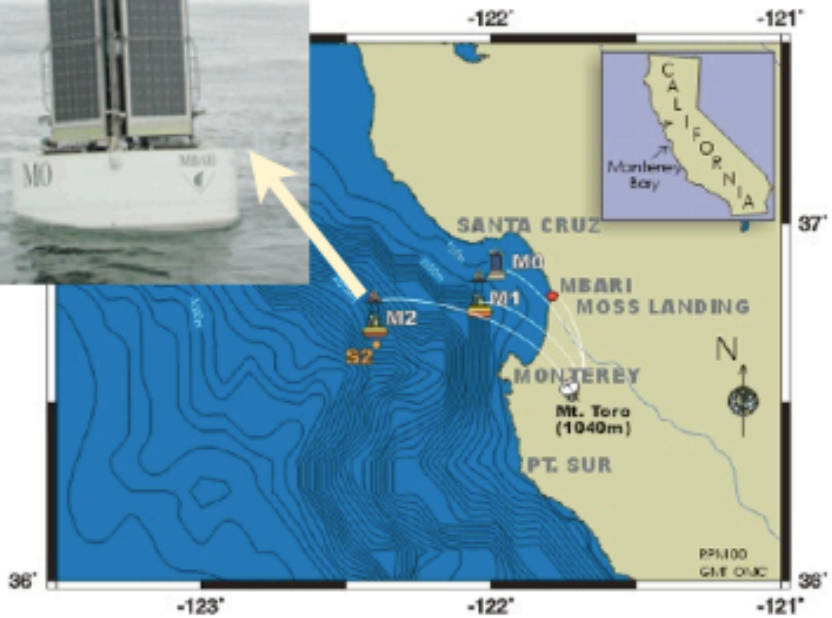


OGC®

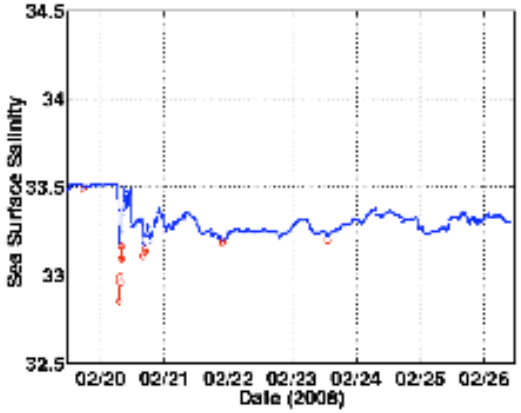
Copyright © 2011, Open Geospatial Consortium

Observation Model (O&M)

Procedure



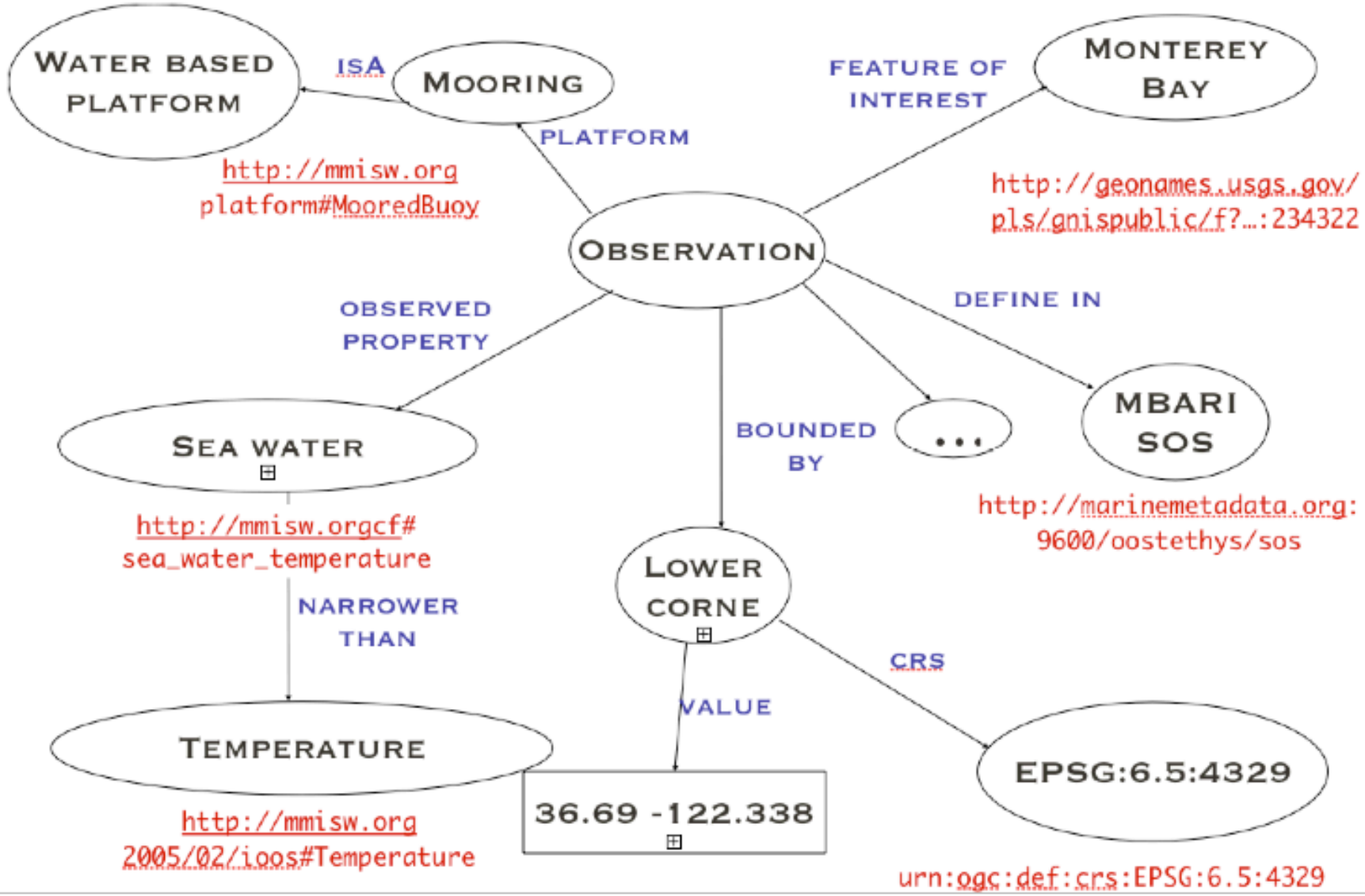
Feature of Interest = Monterey Bay



Estimation value of a property

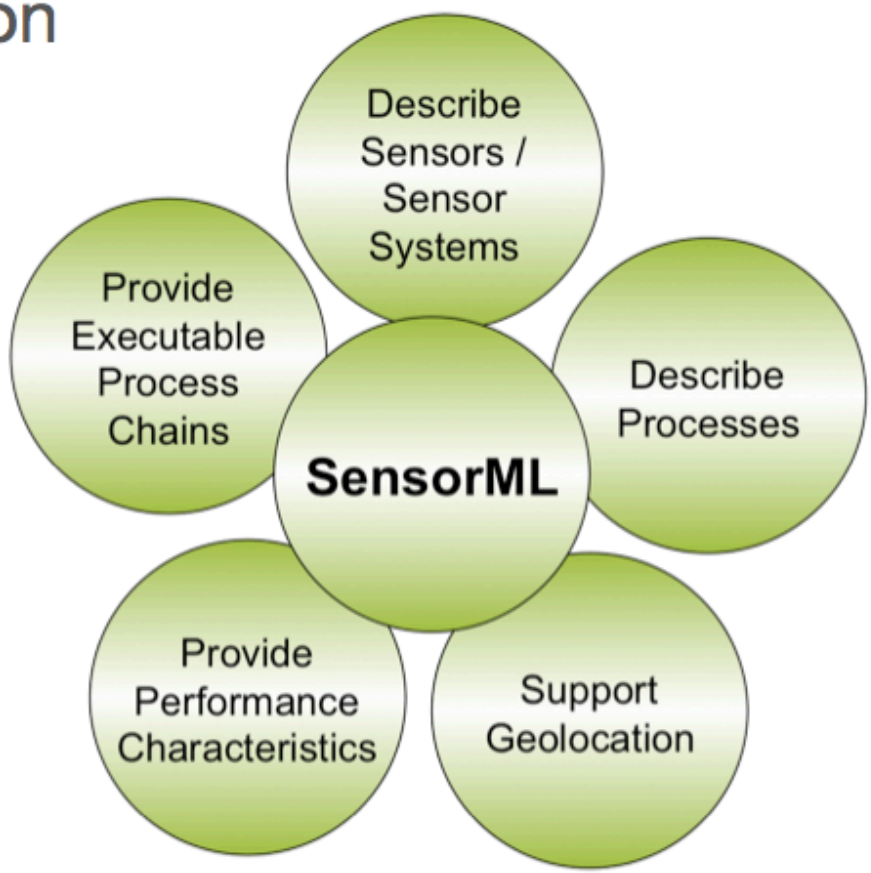
Salinity = property related to the feature of interest

O&M contains general structure, but needs community semantics



SensorML

- Sensor and Process description language
- Uses SWE Common
- Very generic



OGC®

Copyright © 2011, Open Geospatial Consortium

Current Status

- Current standards
  - SensorML – 1.0.1 approved in 2007 (V2.0 anticipated by September 2011)
  - SWE Common Data – V2.0 approved
  - SWE Common Services – V2.0 approved
  - Observations & Measurement – V2.0 approved
  - WNS – Request for Comments
  - SOS – V2.0 in final stages
  - SPS – V2.0 approved
  - SAS – being folded into SOS (with SOAP WS-N and Pub/Sub)
  - TML – no traction; looking to deprecate
- Approved SWE standards can be downloaded:
  - Specification Documents: http://www.opengeospatial.org/standards
  - Specification Schema: http://schemas.opengis.net/

OGC®

What Does v2.0 Bring

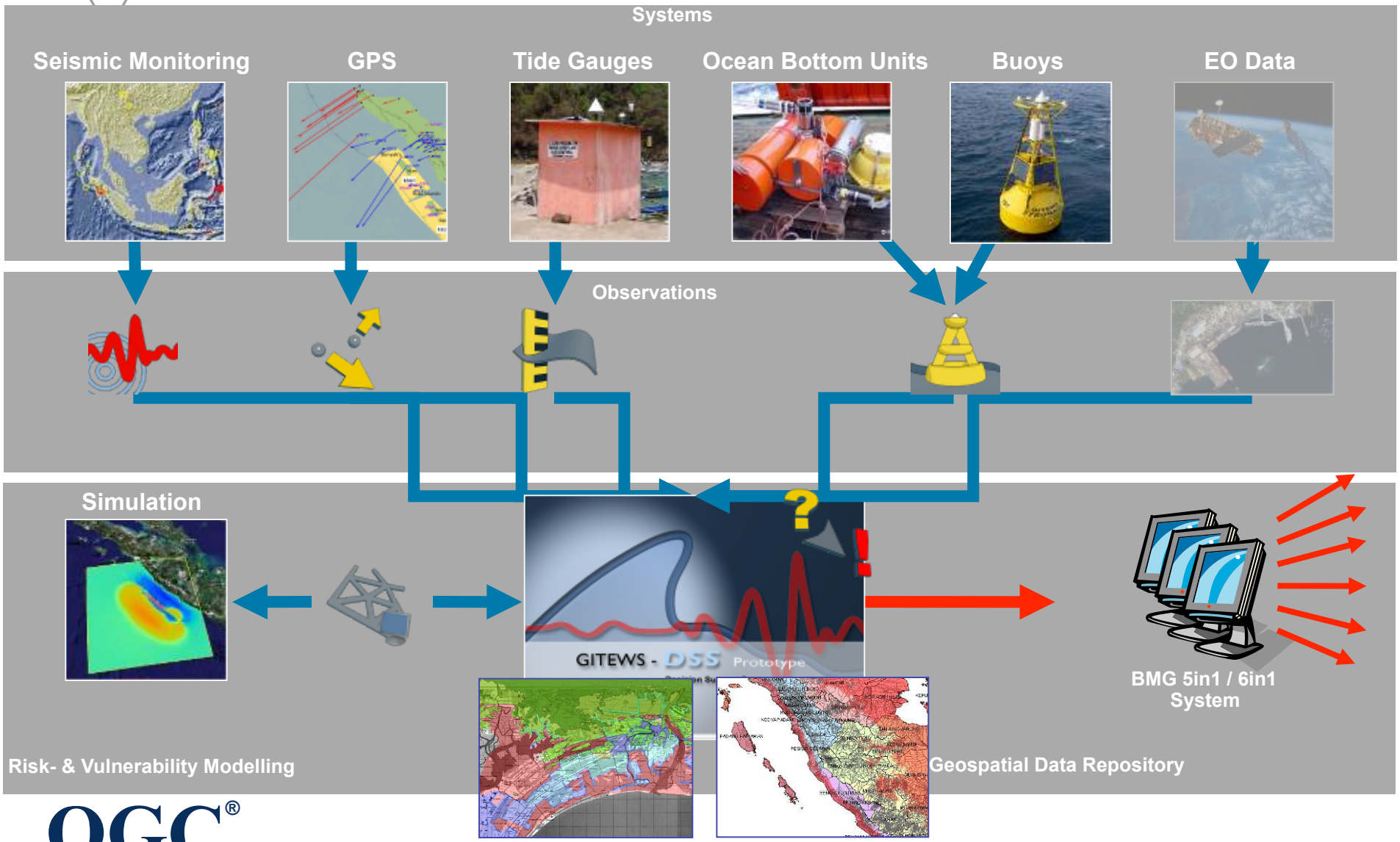
- SWE standards are more precise and testable
- Conformance classes allow for partial uptake and implementation
- More consistent use of SWE Common Data and SWE Common Services throughout SWE standards
- Added support in SWE Common Data for disparate messages, nil values, extensions (e.g. security tagging, UncertML), and required tagging to semantic definitions
- Better support for data streaming
- SOAP implementation supporting security and web notification
- Better support for inheritance and configuration in SensorML

OGC®

Examples

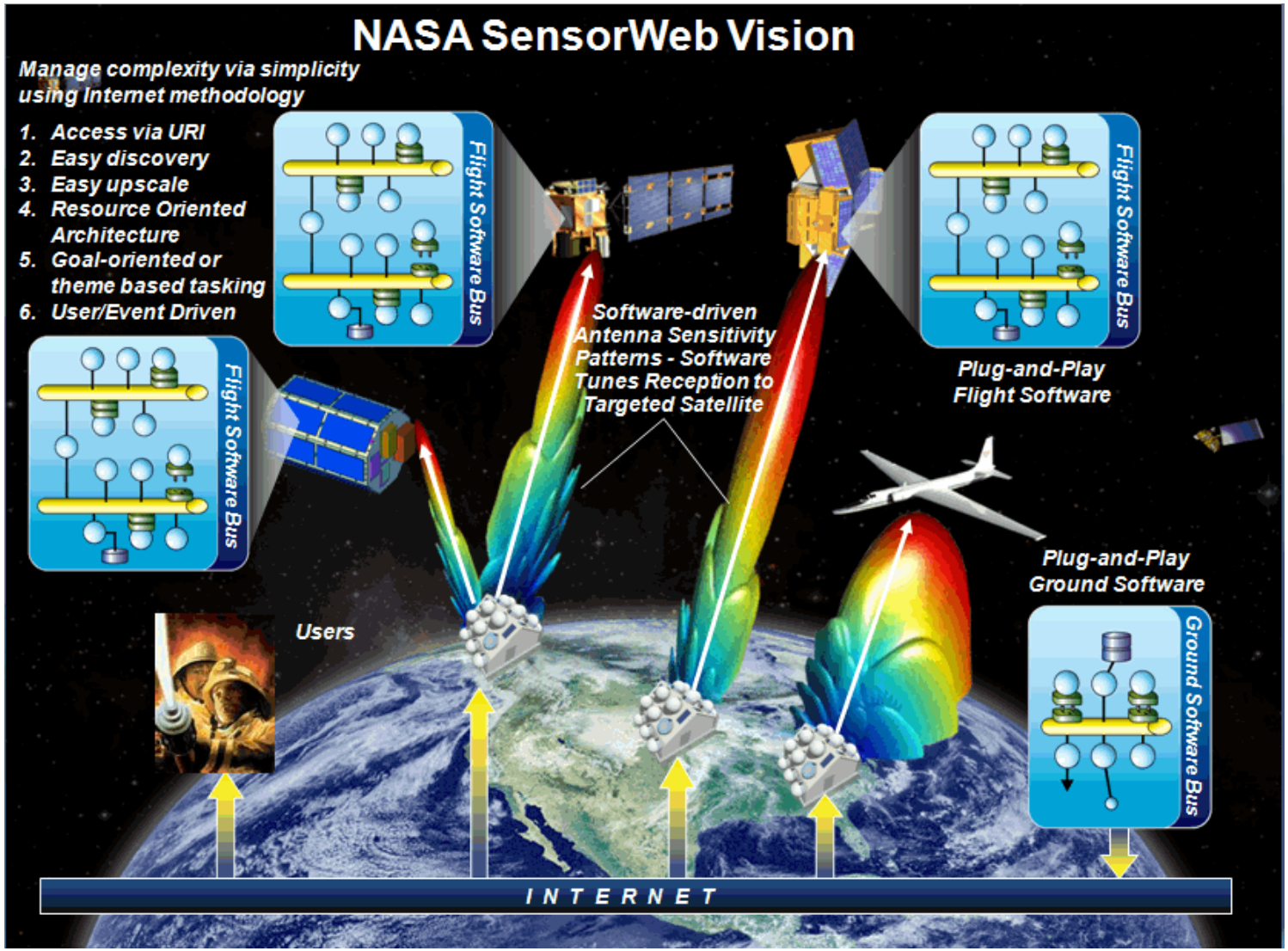
Copyright © 2011, Open Geospatial Consortium

DLR: Tsunami Early Warning & Mitigation Center



OGC®

Application: NASA Sensor Web



OGC®

PULSENet™ Applications: Atmospheric/Air Quality – Fire Monitoring/Smoke Forecasting

Charlie Neuman, San Diego Union-Tribune/Zuma Press

NOAA NESDIS

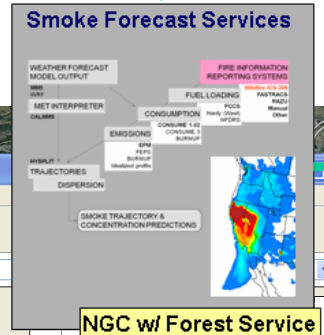
NASA Goddard

Fire Detections

NGC

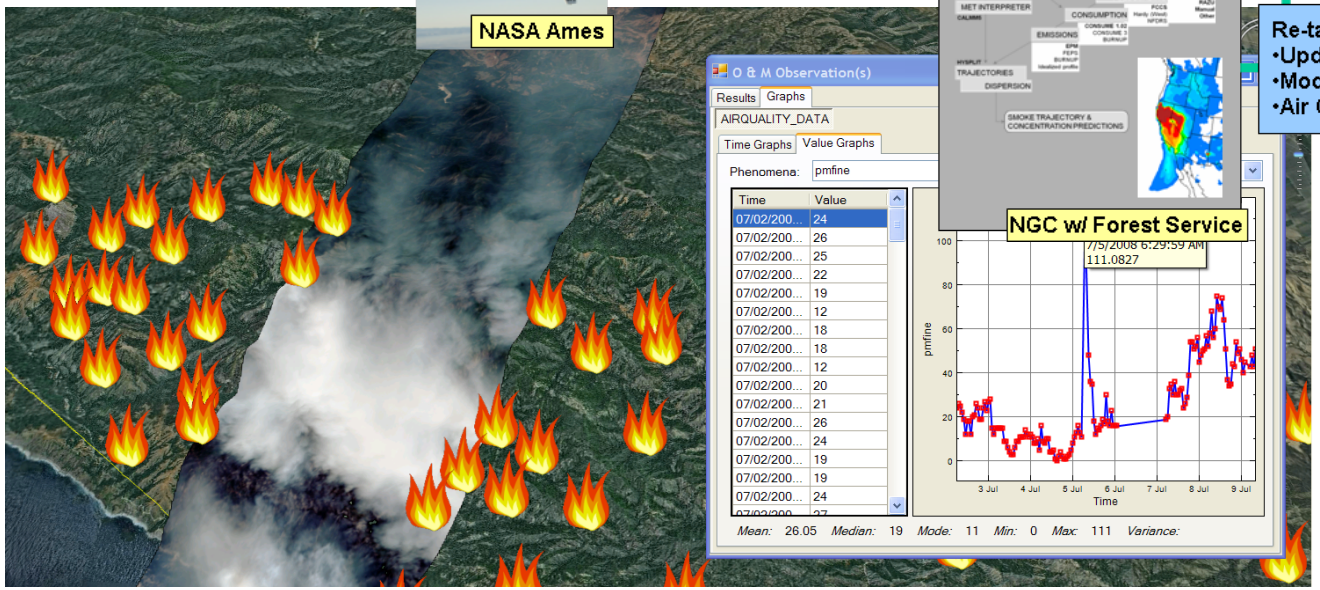
Initiate Model

NASA Ames



NGC w/ Forest Service

Re-task Sensors for
- Updated Fires
- Model Validation
- Air Quality Impacts



OGC®

SAIC Global SensorWeb

SensorWeb

- Defense/Intelligence implementation of OGC SWE standards
- SWE support for UGS and other ISR sensors
- Prominent participant in Empire Challenge since 2005

OGC®

SWE in the Oceans Community

WITH FUNDING FROM

NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

OFFICE OF NAVAL RESEARCH

OPENIOOS.ORG
AN INTEGRATED OCEAN OBSERVING SYSTEM TESTBED

WHERE STANDARDS ENABLE INNOVATION
HOME | ABOUT OPENIOOS | CONTACT US
POWERED BY SURA SCOOP IN SUPPORT OF IOOS

REAL-TIME DATA | 24/7 PREDICTIONS | RETROSPECTIVES | MODELING TEST BED | GUIDES & RESOURCES

- Real-Time Data
- Sea Surface Temperature
- OGC Sensor Web
- Conceptual Design
- System Architecture

DATA PROVIDERS

- AOOS
- COMPS/USF
- COMPUSULT
- CeNCOOS
- CenGOOS - GCOOS
- DISL - GCOOS
- GoMOOS
- MBARI
- MVCO
- NANOOS
- NASA
- NOAA / NDBC
- NOAA / NOS
- TABS TAMU
- COOA UNH
- OceanWatch
- SARTI UPC
- SmartBay

OOSTETHYS DEVELOPERS

- GoMOOS
- MMI
- MBARI
- Texas A&M
- UAH
- VIMS
- NANOOS
- SURA/SCOOP

Real-Time Data from an OGC Sensor Web

This interoperability demonstration represents an effort to develop a Web Services Architecture for Ocean Observing that is enabling observing systems to move closer to the vision of 'network as platform'. We are seeking participants who would like to serve their in-situ observation data via SOS based Web Services. To learn more, visit the OOSTethys.org website.

2061 Platforms reporting Click the station icons on the map for the latest observations.

Map | Satellite | Hybrid

1000 mi
1000 km
Imagery ©2010 TerraMetrics - Terms of Use

Zoom To: - select -

Organizations: - All -

- OOSTethys.org
- How it works
- How to participate
- Serve your data
- SOS Registry
- Google Earth KML Requires GoogleEarth™

Recent observations
No recent observations

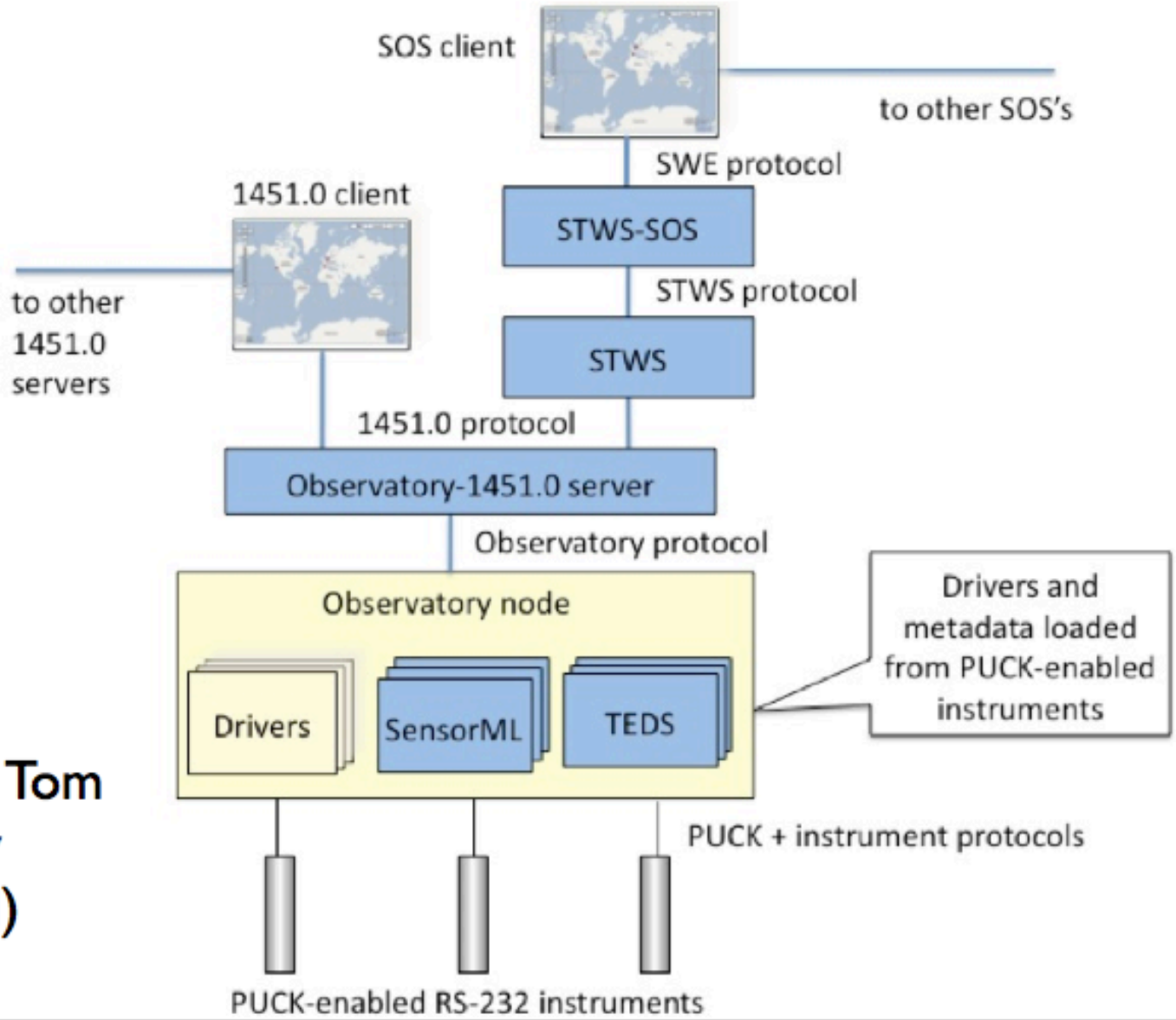
IOOS Variables: - All -

All Variables: - All Observed Properties -

ALL

OGC®

PUCK - IEEE 1451 - SOS



Lead by Tom O'Reilly (MBARI)

Simple SensorML + semantics

Fill out the definitions section and then click: Generate SensorML

Definitions | SensorML

- Service contact
- Systems
  - System: seabird-ctd
    - Contact
    - Metadata
      - type: http://mmisw.org/ont/mmi/systemtype/ctd
      - Short name: seabird-ctd
      - Long name: Seabird XXX CTD Sensor
      - Identifier: id1:idx2
    - This system has: Output / Components
    - Variables
      - Variable: Conductivity
        - URI: http://mmisw.org/ont/cf/parameter/sea_water_electrical_conductivity (Choose)
        - Name: Conductivity
        - UOM: S.m-1

Select URI

conduct

- sea_water_electrical_conductivity - http://mmisw.org/ont/cf/parameter/sea_wa
- soil_hydraulic_conductivity_at_saturation - http://mmisw.org/ont/cf/parameter/
- soil_thermal_conductivity - http://mmisw.org/ont/cf/parameter/soil_thermal_co
- air_pressure_anomaly - http://mmisw.org/ont/cf/parameter/air_pressure_anomaly
- air_pressure_at_cloud_base - http://mmisw.org/ont/cf/parameter/air_pressure_at
- air_pressure_at_cloud_top - http://mmisw.org/ont/cf/parameter/air_pressure_at_c
- air_pressure_at_convective_cloud_base - http://mmisw.org/ont/cf/parameter/air_
- air_pressure_at_convective_cloud_top - http://mmisw.org/ont/cf/parameter/air_pr
- air_pressure_at_freezing_level - http://mmisw.org/ont/cf/parameter/air_pressure_
- air_pressure_at_sea_level - http://mmisw.org/ont/cf/parameter/air_pressure_at_s

Lead by Carlos Rueda (MBARI)

SWE Oceans

Copyright © 2011, Open Geospatial Consortium

Example Implementations

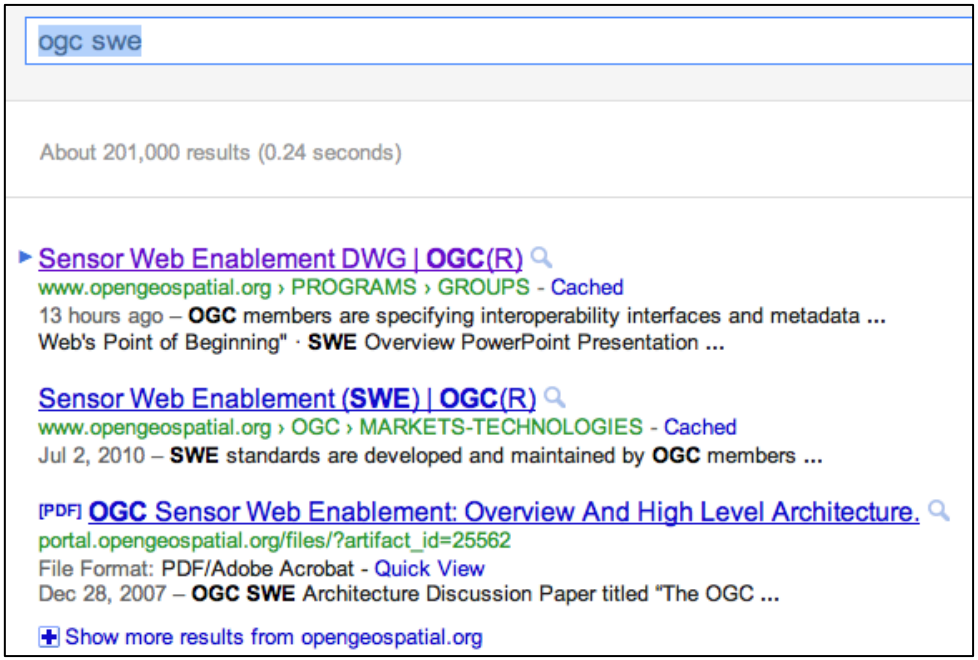
- From: http://www.opengeospatial.org/resource/products/byspec
- 1Spatial Group Ltd
- **52 North**
- Compusult Limited
- Feng Chia U.
- Geomatycs
- Inha University
- IST-SUPSI
- **Lat/lon GmbH**
- Northrop Grumman IT TASC
- SAIC
- UAH

OGC®

Copyright © 2011, Open Geospatial Consortium

Conclusions

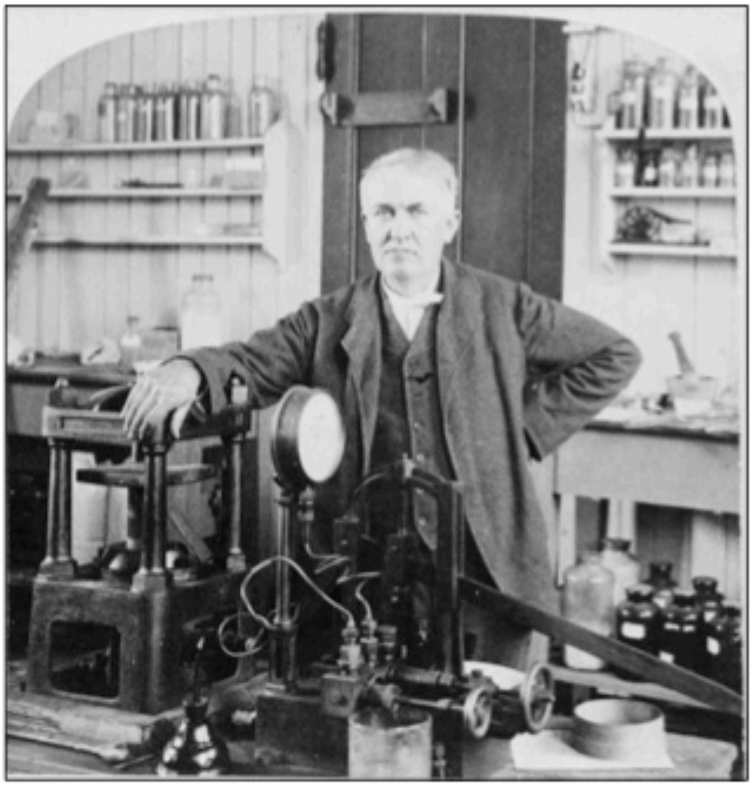
- SWE, including SensorML have been available as standards for several years (~ mature)
- SWE standards are general and can be applied to different domains.
- Google “OGC SWE” = 201,000 results



OGC®

Copyright © 2011, Open Geospatial Consortium

Try and try and try and try...



I have not failed, I've just found 10,000 ways that won't work.

Thomas Edison

OGC®

Tutorial 1

Description of a Weather Station using SensorML

Alexandre Robin - 2006

Tutorial Objectives

- Learn how to describe a simple Detector
- Learn how to use SensorML Metadata for discovery
- Learn how to group several detectors in a System
- Learn how to describe System components positions

SensorML Definitions

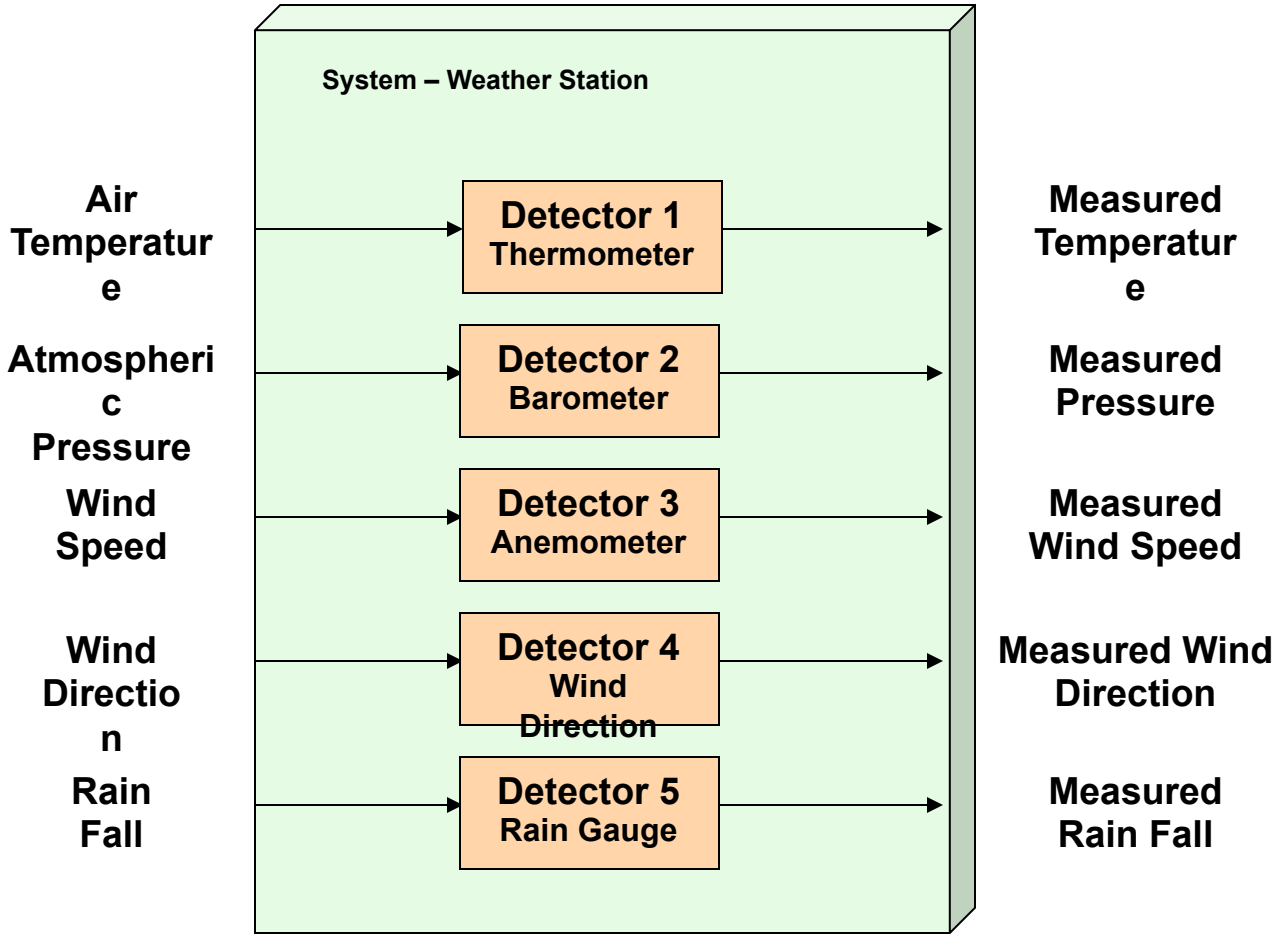
- **Detector: Atomic part of a composite Measurement System defining sampling and response characteristic of a simple detection device. A detector has only one input and one output, both being scalar quantities. In SensorML a detector is a particular type of Process Model.**

- **System: Composite model of a group or array of components, which can include detectors, actuators, or sub-systems. A System relates a Process Chain to the real world and therefore provides additional definitions regarding relative positions of its components and communication interfaces**

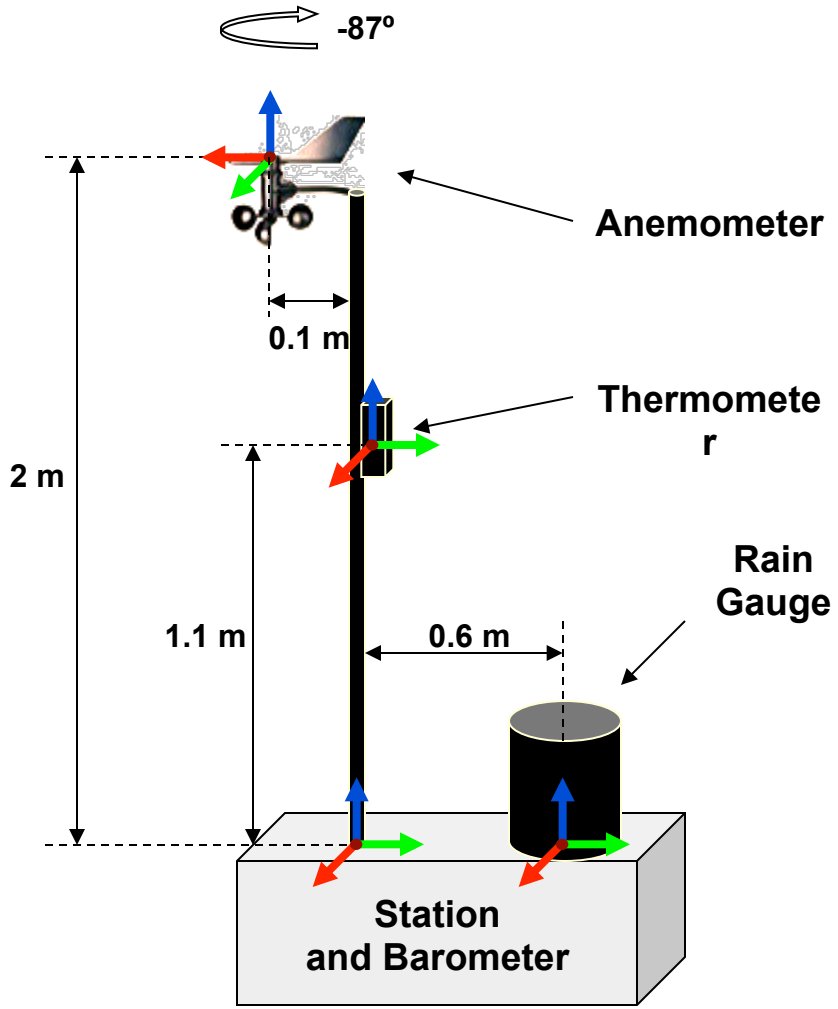
Weather Station Description

- Inspired from a Davis Instruments weather station.
- Measures 5 quantities: temperature, pressure, wind speed, wind direction and rain fall amount.
- Using 4 different modules: thermometer, barometer, wind sensor and rain gauge.
- Also includes a monitor station which reports measurements
  on an LCD screen. (barometer is included in the monitor)

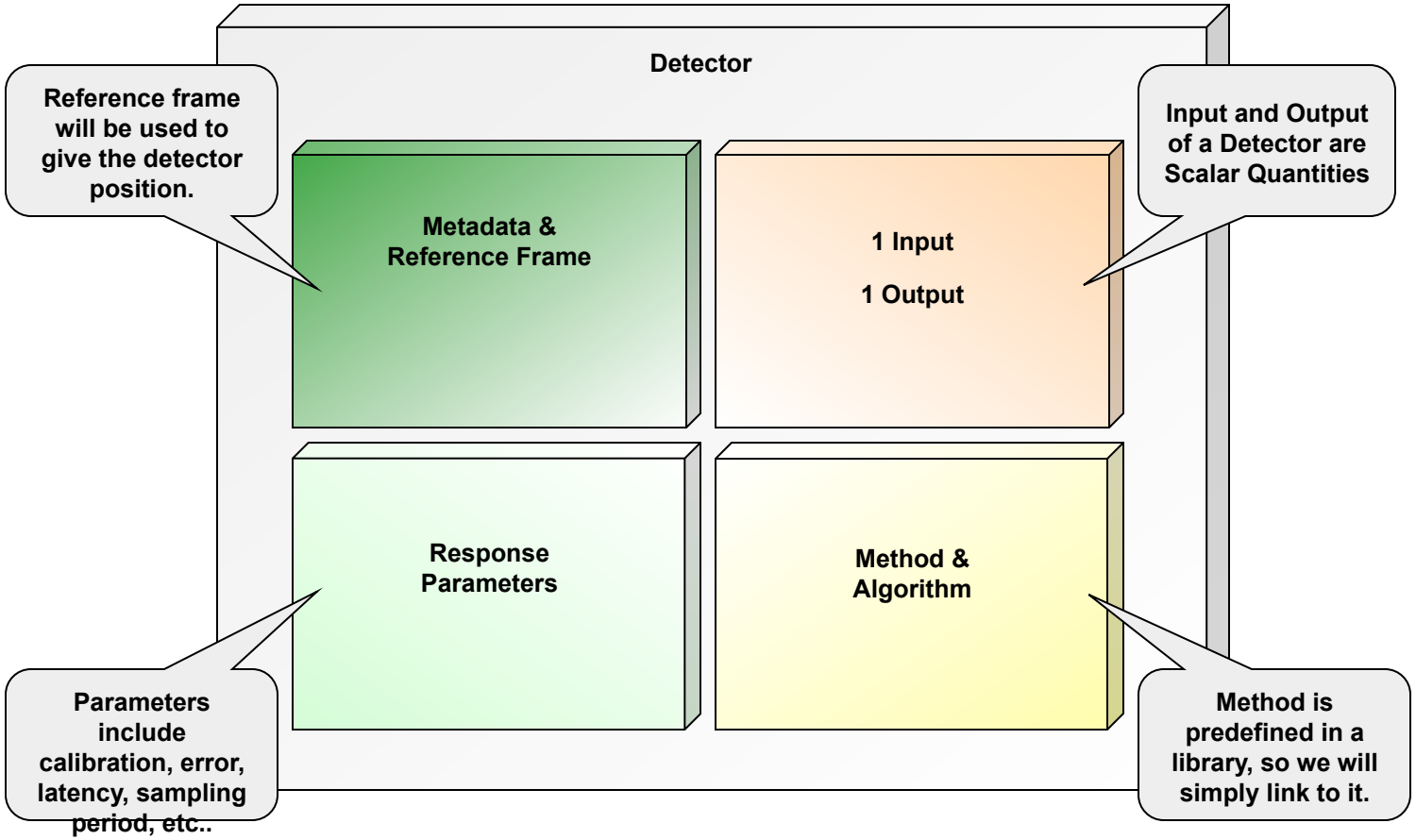
Weather Station Block Diagram



Weather Station Mounting Diagram



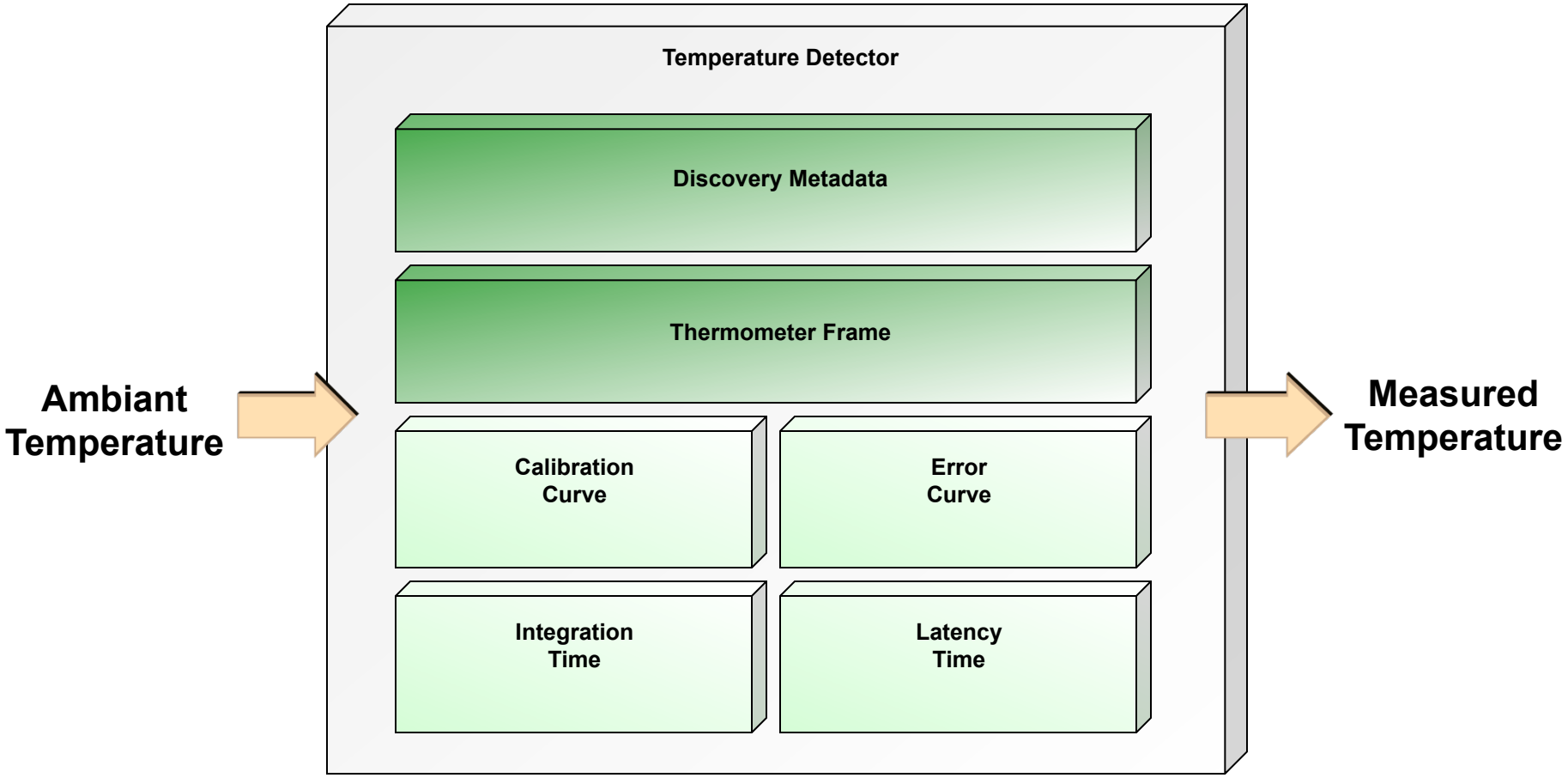
Each Detector = Process Model



Detector XML Structure

```
<Detector>
  <metadata … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <parameters>
    <ParameterList>
      <steadyStateResponse … />
      <error … />
      <frequencyResponse … />
      <integrationTime … />
      <latencyTime … />
    </ParameterList>
  </parameters>
  <method xlink:href="urn:ogc:def:process:detector"/>
</Detector>
```

Thermometer = Temperature Detector



Temperature Detector – Metadata

```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

In this example, we define a name and a model number for the detector.

Each Term is well defined by the URN.

```
<identification>
  <IdentifierList>
    <indentifier name="longName">
      <Term qualifier="urn:ogc:def:identifier:longName">Davis Thermometer</Term>
    </identifier>
    <indentifier name="modelNumber">
      <Term qualifier="urn:ogc:def:identifier:modelNumber">7817</Term>
    </identifier>
  </IdentifierList>
</identification>
```

Temp. Detector – Reference Frame

```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

It is needed to provide a textual description of how the reference frame is attached to the hardware. This is useful for future reference (i.e. mounting). The 'id' will also be used when specifying position.

```
<referenceFrame>
  <gml:EngineeringCRS gml:id="THERMOMETER_FRAME">
    <gml:srsName>Temperature Detector Spatial Frame</gml:srsName>
    <gml:usesCS xlink:href="urn:ogc:def:cs:xyzFrame"/>
    <gml:usesEngineeringDatum>
      <gml:EngineeringDatum gml:id="THERMOMETER_DATUM">
        <gml:datumName>Temperature Detector Spatial Datum</gml:datumName>
        <gml:anchorPoint>
          Origin is situated at the connector/case junction.
          X,Y,Z are undetermined since orientation is not critical.
        </gml:anchorPoint>
      </gml:EngineeringDatum
    </gml:usesEngineeringDatum>
  </gml:EngineeringCRS>
</referenceFrame>
```

Temperature Detector – Inputs

```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

A detector has only one scalar input.

It is used to define what type of phenomenon the detector measures, using the definition attribute, as well as the unit of measure in which it is expressed.

```
<inputs>
  <InputList>
    <input name="temperature">
      <Quantity
        definition="urn:ogc:def:phenomenon:temperature"
        uom="urn:ogc:def:unit:celsius"/>
    </input>
  </InputList>
</inputs>
```

Temperature Detector – Outputs

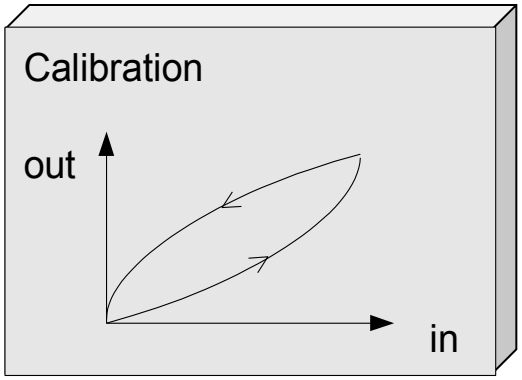
```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

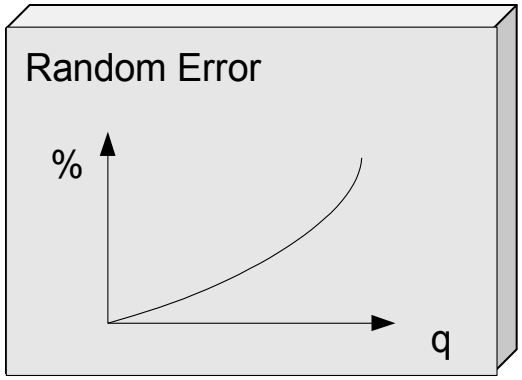
A detector has only one scalar output.

It is used to define what type of phenomenon the detector outputs, using the definition attribute, as well as the unit of measure in which it is expressed.

```
<outputs>
  <OutputList>
    <output name="measuredTemperature">
      <Quantity
        definition="urn:ogc:def:phenomenon:temperature"
        uom="urn:ogc:def:unit:celsius"/>
    </output>
  </OutputList>
</outputs>
```

Temperature Detector – Parameters





The steady state response curve (or calibration curve) gives mapping of input values to output values at a steady state regime.

The error curve characterizes the accuracy of the measure-ments with respect to some physical quantity, which can be the input itself or another external quantity.

Temperature Detector – Calibration

```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

```
<steadyStateResponse>
  <ConditionalCurve>
    <condition name="calibration_time">
      <Time>2004-10-05T04:35:00</Time>
    </condition>
    <data>
      <Curve>
        <definition>
          <Coordinates>
            <axis name="input_temp">
              <Quantity uom="urn:ogc:def:unit:celsius"/>
            </axis>
            <axis name="output_temp">
              <Quantity uom="urn:ogc:def:unit:celsius"/>
            </axis>
          </Coordinates>
        </definition>
        <tupleValues>-50,-50 140,140</tupleValues>
      </Curve>
    </data>
  </ConditionalCurve>
</steadyStateResponse>
```

Temperature Detector – Error

```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

```
<error>
  <ConditionalCurve>
    <condition name="cable_length">
      <Quantity uom="urn:ogc:def:unit:meter">10</Quantity>
    </condition>
    <data>
      <Curve>
        <definition>
          <Coordinates>
            <axis name="input_temp">
              <Quantity uom="urn:ogc:def:unit:celsius"/>
            </axis>
            <axis name="abs_max_error">
              <Quantity uom="urn:ogc:def:unit:celsius"/>
            </axis>
          </Coordinates>
        </definition>
        <tupleValues">-50,1 140,0.01</tupleValues>
      </Curve>
    </data>
  </ConditionalCurve>
</error>
```

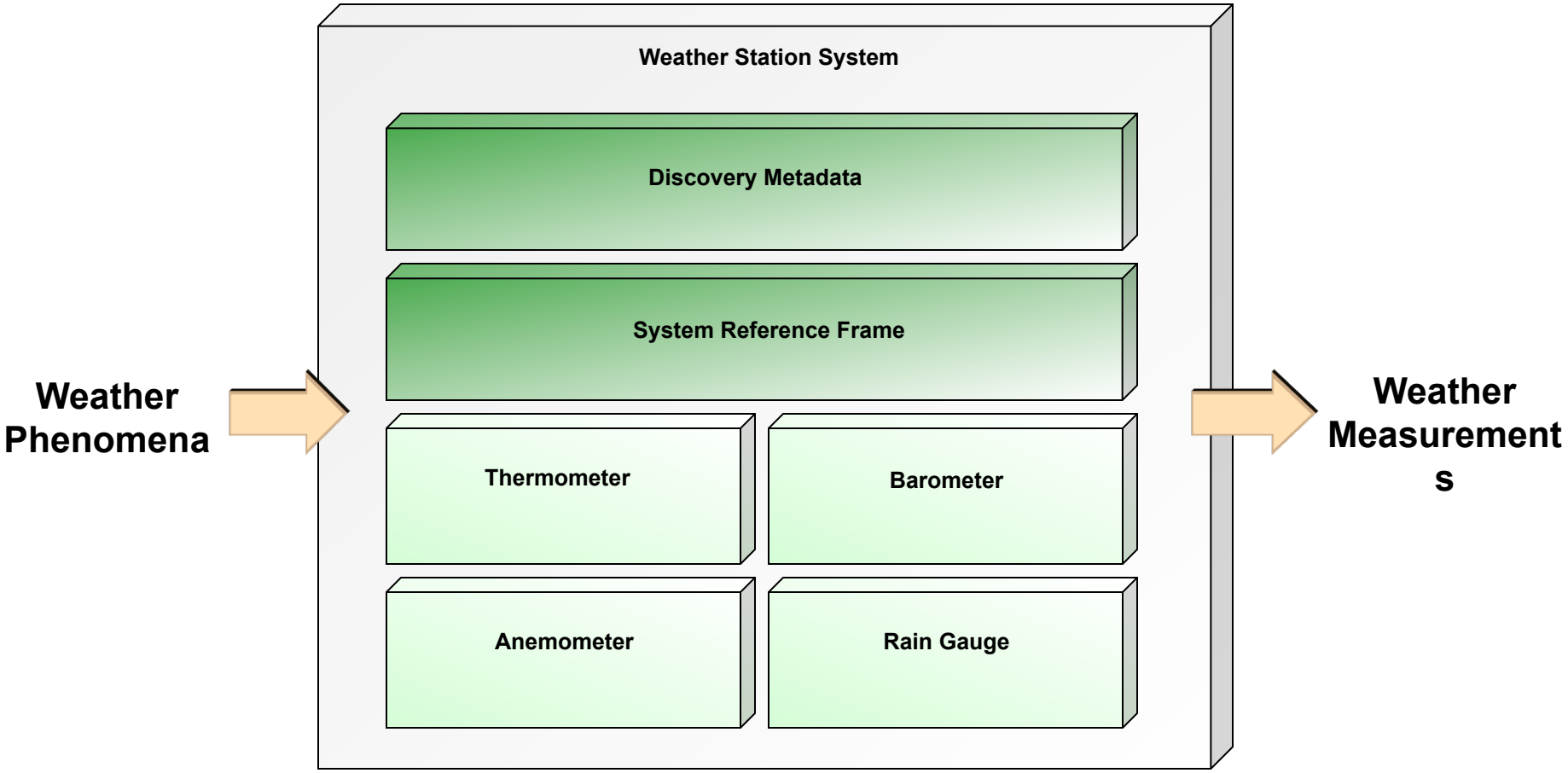
Temperature Detector – Latency Time

```
<Detector>
  <metadata/>
  <referenceFrame/>
  <inputs/>
  <outputs/>
  <parameters>
    <ParameterList>
      <steadyStateResponse/>
      <error/>
      <latencyTime/>
    </ParameterList>
  </parameters>
  <method/>
</Detector>
```

Latency time specifies the amount of time between the phenomenon is sampled and the availability of the corresponding value at the output.

```
<latencyTime>
  <ConditionalValue>
    <condition name="medium">
      <Category>air</Category>
    </condition>
    <data>
      <Quantity
        definition="urn:ogc:def:phenomenon:duration"
        uom="urn:ogc:def:unit:second">10</swe:Quantity>
    </data>
  </ConditionalValue>
</latencyTime>
```

System = Group of Detectors



System XML Structure

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contact … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

Weather Station – Identification

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

In this example, we define the name, the manufacturer and the model number of the weather station.

Each Term is well defined by the URN.

```
<identification>
  <IdentifierList>
    <indentifier name="longName">
      <Term qualifier="urn:ogc:def:identifier:longName">Davis Thermometer</Term>
    </identifier>
    <indentifier name="manufacturer">
      <Term qualifier="urn:ogc:def:identifier:manufacturer">Davis Instruments</Term>
    </identifier>
    <indentifier name="modelNumber">
      <Term qualifier="urn:ogc:def:identifier:modelNumber">7440</Term>
    </identification>
  </IdentifierList>
</identification>
```

Weather Station – Classification

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

In this example, we define the intended application of the system as well as sensor types included in this system.

```
<classification>
  <ClassifierList>
    <classifier name="intendedApplication">
      <Term qualifier="urn:ogc:def:classifier:application">Weather</Term>
    </classifier>
    <classifier name="sensorType">
      <Term qualifier="urn:ogc:def:classifier:sensorType">Thermometer</Term>
    </classifier>
    <classifier name="sensorType">
      <Term qualifier="urn:ogc:def:classifier:sensorType">Barometer</Term>
    </classifier>
      …
  </ClassifierList>
</classification>
```

Weather Station – Validity Period

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

The description of the System is valid from the start date until now.

```
<validTime>
  <StartTime>2004-10-05T04:35:00</StartTime>
  <StopTime>currentTime</StopTime
</classification>
```

Weather Station – Contacts

We define a primary contact organization for this System. Here it is the weather station manufacturer. (This is based on ISO19115)

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contact … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

```
<contact role="urn:ogc:def:identifier:manufacturer">
  <ResponsibleParty>
    <organizationName>Davis Instruments</organizationName>
    <contactInfo>
      <phone>
        <voice>+01-510-732-9229</voice>
        <facsimile>+01-510-732-9188</facsimile>
      </phone>
      <address>
        <deliveryPoint>3465, Diablo Avenue</deliveryPoint>
        <city>Hayward</city>
        <administrativeArea>CA</administrativeArea>
        <postalCode>94545-2778</postalCode>
        <country>USA</country>
        <electronicMailAddress>sales@davisnet.com</electronicMailAddress>
      </address>
    </contactInfo>
  </ResponsibleParty>
</contact>
```

Weather Station – Documentation

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

In this example, we list the user manual as a reference document for this System. The online location is specified.

```
<documentation>
  <DocumentList>
    <member name="manual">
      <Document>
        <description>
          <Discussion>Weather Station User Manual</Discussion>
        </description>
        <fileLocation xlink:href="http://www.davisnet.com/manuals/7440.pdf"/>
      </Document>
    </member>
  </DocumentList>
</documentation>
```

Weather Station – Reference Frame

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

It is needed to provide a textual description of how the reference frame is attached to the hardware. This is useful for future reference (i.e. mounting). The 'id' will also be used when specifying position.

```
<referenceFrame>
  <gml:EngineeringCRS gml:id="STATION_FRAME">
    <gml:srsName>Weather Station Spatial Frame</gml:srsName>
    <gml:usesCS xlink:href="urn:ogc:def:cs:xyzFrame"/>
    <gml:usesEngineeringDatum>
      <gml:EngineeringDatum gml:id="STATION_DATUM">
        <gml:datumName>Weather Station 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 but undetermined.
        </gml:anchorPoint>
      </gml:EngineeringDatum
    </gml:usesEngineeringDatum>
  </gml:EngineeringCRS>
</referenceFrame>
```

Weather Station – Inputs

There is an independent virtual input for each measured phenomenon.

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

```
<inputs>
  <InputList>
    <input name="ambiantTemperature">
      <Quantity definition="urn:ogc:def:phenomenon:temperature"/>
    </input>
    <input name="atmosphericPressure">
      <Quantity definition="urn:ogc:def:phenomenon:pressure"/>
    </input>
    <input name="windSpeed">
      <Quantity definition="urn:ogc:def:phenomenon:windSpeed"/>
    </input>
    <input name="windDirection">
      <Quantity definition="urn:ogc:def:phenomenon:windDirection"/>
    </input>
    <input name="rainFall">
      <Quantity definition="urn:ogc:def:phenomenon:rainFall"/>
    </input>
  </InputList>
</inputs>
```

Weather Station – Outputs

Output group includes all measured values and a time tag.

```
<outputs>
 <OutputList>
  <output name="weatherMeasurements">
   <DataGroup>
    <component name="time">
      <Time definition="urn:ogc:def:phenomenon:time"
          uom="urn:ogc:def:unit:iso8601"/></component>
    <component name="temperature">
      <Quantity definition="urn:ogc:def:phenomenon:temperature"
            uom="urn:ogc:def:unit:celsius"/></component >
    <component name="barometricPressure">
      <Quantity definition="urn:ogc:def:phenomenon:pressure"
            uom="urn:ogc:def:unit:pascal"/></component >
    <component name="windSpeed">
      <Quantity definition="urn:ogc:def:phenomenon:windSpeed"
            uom="urn:ogc:def:unit:meterPerSecond"/></component >
     ...
   </DataGroup>
  </output>
 </OutputList>
</outputs>
```

```
<System>
 <description … />
 <identification … />
 <classification … />
 <validTime … />
 <contacts … />
 <documentation … />
 <referenceFrame … />
 <inputs … />
 <outputs … />
 <processes … />
 <connections … />
 <positions … />
 <interfaces … />
</System>
```

Weather Station – Internal Processes

List of all detectors constituting the System.

Can be either included inline or in a separate document referenced using an xlink pointer.

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

```
<processes>
  <ProcessList>
    <process name="thermometer">
      <Detector id="DAVIS_THERMOMETER"> ... </Detector>
    </process>
    <process name="barometer">
      <Detector id="DAVIS_BAROMETER"> ... </Detector>
    </process>
    <process name="anemometer">
      <Detector id="DAVIS_ANEMOMETER"> ... </Detector>
    </process>
    <process name="windDirection" xlink:href="URI to detector document"/>
    <process name="rainGauge" xlink:href="URI to rainGauge document"/>
  </ProcessList>
</processes>
```

Weather Station – Internal Connections

The list of connections describes all internal connections in the System.

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

```
<connections>
  <ConnectionList>
    <connection name="inputToThermometer">
      <Link>
        <source ref="this/inputs/ambiantTemperature"/>
        <destination ref="thermometer/inputs/temperature"/>
      </Link>
    </connection>
    <connection name="thermometerToOutput">
      <Link>
        <source ref="thermometer/outputs/measuredTemperature"/>
        <destination ref="this/outputs/weatherMeasurements/temperature"/>
      </Link>
    </connection>
      ...
  </ConnectionList>
</connections>
```

Weather Station – Positions

The position list specifies the position (location and orientation) of each detector and the System itself.

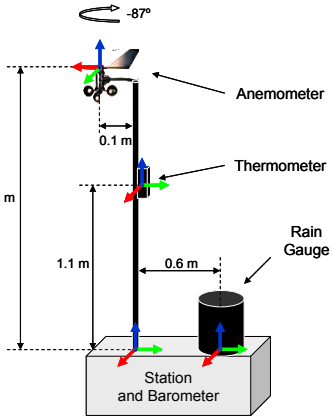
```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

```
<positions>
  <PositionList>
    <position name="stationPosition"> ... </position>
    <position name="thermometerPosition"> ... </position>
    <position name="barometerPosition"> ... </position>
    <position name="anemometerPosition"> ... </position>
    <position name="windDirectionDetectorPosition"> ... </position>
    <position name="rainGaugePosition"> ... </position>
  </PositionList>
</positions>
```

Weather Station GeoPosition

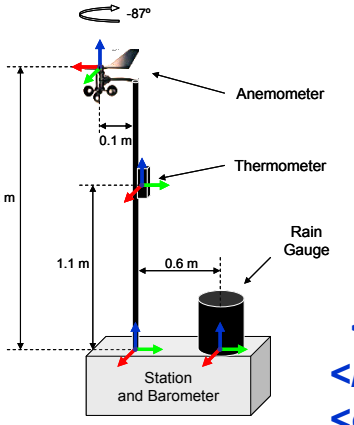
```
<position name="stationPosition">
  <Position localFrame="#STATION_FRAME" referenceFrame="urn:ogc:def:crs:epsg4329">
    <location>
      <GeoLocation>
        <latitude>
          <Quantity uom="urn:ogc:def:unit:degree" axisCode="Y">34.724</Quantity>
        </latitude>
        <longitude>
          <Quantity uom="urn:ogc:def:unit:degree" axisCode="X">-86.945</Quantity>
        </longitude>
        <altitude>
          <Quantity uom="urn:ogc:def:unit:meter" axisCode="Z">20.11</Quantity>
        </altitude>
      </GeoLocation>
    </location>
    <orientation>
      <Orientation>
        <coordinate name="trueHeading">
          <Quantity uom="urn:ogc:def:unit:degree" axisCode="Z">87.0</Quantity>
        </coordinate>
      </Orientation>
    </orientation>
  </Position>
</position>
```

Thermometer Position




```
<position name="thermometerPosition">
 <Location localFrame="#THERMOMETER_FRAME" referenceFrame="#STATION_FRAME">
  <coordinate name="x">
   <Quantity uom="urn:ogc:def:unit:meter" axisCode="X">0.02</Quantity>
  </coordinate>
  <coordinate name="y">
   <Quantity uom="urn:ogc:def:unit:meter" axisCode="Y">0.02</Quantity>
  </coordinate>
  <coordinate name="z">
   <Quantity uom="urn:ogc:def:unit:meter" axisCode="Z">1.1</Quantity>
  </coordinate>
 </Location>
</position>
```

Wind Direction Detector Position




```
<position name="windDirectionDetectorPosition">
 <Position localFrame="#WIND_DETECTOR_FRAME" referenceFrame="#STATION_FRAME">
  <location>
   <Location>
    <coordinate name="x">
     <Quantity uom="urn:ogc:def:unit:meter" axisCode="X">0.0</Quantity>
    </coordinate>
    <coordinate name="y">
     <Quantity uom="urn:ogc:def:unit:meter" axisCode="Y">-0.1</Quantity>
    </coordinate>
    <coordinate name="z">
     <Quantity uom="urn:ogc:def:unit:meter" axisCode="Z">2.0</Quantity>
    </coordinate>
   </Location>
  </location>
  <orientation>
   <Orientation>
    <coordinate name="theta">
     <Quantity uom="urn:ogc:def:unit:degree" axisCode="Z">-87</Quantity>
    </coordinate>
   </Orientation>
  </orientation>
 </Position>
</position>
```

Weather Station – Interface

Defines a serial interface for communicating with the station (get measurements).

```
<System>
  <description … />
  <identification … />
  <classification … />
  <validTime … />
  <contacts … />
  <documentation … />
  <referenceFrame … />
  <inputs … />
  <outputs … />
  <processes … />
  <connections … />
  <positions … />
  <interfaces … />
</System>
```

```
<interfaces>
  <InterfaceList>
    <interface name="serialPort">
      <InterfaceDefinition>
        <applicationLayer>
          <Protocol definition="urn:davis:def:weatherLink"/>
        </applicationLayer>
        <physicalLayer>
          <Protocol definition="urn:ogc:def:protocol:RS232">
        </physicalLayer>
        <mechanicalLayer>
          <Connector definition="urn:ogc:def:connector:DB9">
        </mechanicalLayer>
      </InterfaceDefinition>
    </interface>
  </InterfaceList>
</interfaces>
```

More Advanced Features

- Advanced properties of detectors such as Frequency Response, Spatial Response and Temporal Response.
- Possibility of using index mechanism to describe large arrays of detectors (several thousands in the case of a frame camera) with variable parameters within the array.
- Possibility of using processes to describe variable positions of components (like in the case of a scanner)

Acknowledgements

- Some slides were provided by:
  - Ingo Simons (International Geospatial Services Institute (iGSI) GmbH)
  - Mike Botts (Botts Innovative Research, Inc.)
  - Alex Robin (EADS Astrium)

OGC®

Copyright © 2011, Open Geospatial Consortium