MP 00B0000055

MITRE PRODUCT

A Space Surveillance Ontology

Captured in an XML Schema

October 2000

Mary K. Pulvermacher
Daniel L. Brandsma
John R. Wilson

Sponsor: ESC/ND
Dept. No.: D310
Contract No.: F19628-99-C0001
Project No.: 03014000

Approved for public release; distribution unlimited.

MITRE
Center for Air Force C2 Systems
Bedford, Massachusetts

MITRE Department
and Project Approval:

John J. Shottes
Project Lead, 4000

Abstract

Achieving data interoperability is a necessary element of realizing the United States government vision of interoperability across all the services. This paper describes an Extensible Markup Language (XML) approach invented to capture data structure, content, and semantics in a targeted military domain of space surveillance. The resulting Space Surveillance Ontology could become a standard, shared space surveillance vocabulary as a step toward achieving data interoperability for military space data across domains. The ontology was created using the World Wide Web Consortium (W3C) emerging standard called XML Schema.

KEYWORDS: XML, XML Schema, Ontology, Space Surveillance

Acknowledgments

The authors of this document gratefully acknowledge Mr. Mark J. Heller, Dr. Scott A. Renner, Mr. Alan D. Skillicorn, and Mr. John J. Shottes in helping to develop the concept for this task and in gaining sponsor approval for its execution. We would also like to thank Dr. Roger L. Costello, who provided an excellent XML course, helped us obtain the software tools needed, and graciously answered several technical questions. To all of you, a sincere thank you.

Executive Summary

Interoperability across the services is a fundamental premise of United States military visions, such as those captured in Joint Vision 2020 and the United States Air Force (USAF) Scientific Advisory Board concept of a Joint Battlespace Infosphere (JBI). This widespread interoperability is dependent upon the ability to not only transmit data across the enterprise, but also interpret that data correctly. Correct interpretation of data requires that the meaning of the data (i.e., data semantics) be made available both within a specific domain and across domains. This paper describes one approach for capturing data structure, content, and meaning, in what we call an ontology, using the World Wide Web Consortium (W3C) emerging standard called XML Schema. Capturing this information is a necessary step toward achieving enterprise-wide data interoperability.

XML is a family of new technologies and open standards for web-based information management. XML has its roots in other markup languages (e.g., Hypertext Markup Language (HTML), Standard Generalized Markup Language (SGML)) that deal with data format. XML separates data format from data content and uses its family of technologies to make data portable. The power of XML lies in its simplicity, its support in the commercial community, and its relationship to the Internet.

XML is already being used in the government and could be applied to the military space domain. (See Appendix B for information on related XML efforts.) The current space Command and Control (C2) concept is based upon tightly coupled, message-based systems. Future military space architectures propose more flexible information-based systems that incorporate information technologies present in the commercial world. XML is a technology that could assist in migrating to this new architecture.

This paper describes a Space Surveillance Ontology we developed to provide insight into both the applicability of XML to the space domain and the process needed to describe and document space information objects using XML. An ontology is the mechanism that captures the definitions of information elements, the individual data items, and the associated interrelationships. We invented an ontology approach that captures data structure, content, and semantics all in the XML Schema vocabulary. We used this new approach to develop the Space Surveillance Ontology. The paper also describes the process used to develop the ontology, related XML efforts, lessons learned, and XML transformations we used to test our ontology approach. The paper includes details on a government XML registry that is the likely destination for the Space Surveillance Ontology.

We draw several conclusions from this effort:

1. Capturing data semantics is vital to achieve data interoperability as envisioned by military vision documents.

2. XML is a technology of choice for the military, but it is only one aspect of the total solution. XML is an internet standard for data exchange, but it does not address things like how the data will be used, how the data will be stored, security approach, etc.
3. XML Schema definition language is the XML vocabulary of choice and is preferred over Document Type Definitions (DTDs).
4. XML Schemas are relatively easy to build; we developed ours in approximately two staff weeks.
5. Namespaces are a powerful enabler of standard data definitions; they allow naming disambiguation and information characterization in a way most appropriate to a sub-domain.
6. The commercial market is embracing XML, and the military can take advantage of this significant economic force.
7. XML could be a powerful tool for system migration in that it creates a layer of transparency that makes data portable.
8. The Space Surveillance Ontology results should be shared, at least across the government community. This ontology could be the foundation for future military XML efforts, not just for the specific domain it addresses, but also as a pathfinder for other government efforts, including the government XML registry.

Table of Contents

List of Figures

List of Tables

Section 1

Introduction

1.1 XML in the Space Domain

Joint Vision 2020 [JV2020] describes a vision that includes all services operating together as a seamless whole. The United States Air Force (USAF) Scientific Advisory Board envisions a combat information management system called the Joint Battlespace Infosphere [JBI]. Each of these visions rely upon key information being made available to the right people, at the right time, and in the right place. Space is a key enabler of achieving military objectives across the services; therefore, space mission requirements continue to grow.

To achieve these visions, future space systems will need to be more flexible and responsive. The current space Command and Control (C2) concept is based on tightly coupled, message-based systems. Future architectures propose more flexible information or content-based systems that incorporate information technologies present in the commercial world. Extensible Markup Language (XML) is a new technology that may help realize this new architecture.

XML is a family of new technologies and open standards for web-based information management. (See appendix A for more information on the XML family of technologies.) The concepts underlying XML are not new. The power of XML lies in its simplicity and its support in the commercial community. As stated by Dr. Scott Renner,[1] "XML is powerful because it is part of the *Internet phenomenon.* XML is based on open standards. There are plenty of inexpensive (or free) tools. It is present everywhere the web is present – it will soon be part of every web browser."

The commercial industry is making significant investments in XML, especially in the area of electronic business. The Air Force can capitalize upon this investment and possibly use XML to assist in migrating from tightly coupled, message-based systems to flexible, information-based systems. This could be done with a migration strategy that mitigates the impact to ongoing operations. Further, because XML is a key enabler of the JBI architecture, use of XML may also position the space community to easily integrate into the JBI.

1.2 What is an Ontology?

The definition of information, in terms of both syntax and semantics, is an integral part of the establishment of enterprise architectures. The mechanism that captures the definitions of the information elements, the individual data items, and the associated interrelationships is

---

[1] Dr. Scott Renner, Principal Information Systems Engineer, The MITRE Corporation, August 16, 2000.

referred to as an *ontology*. The term "ontology" has its roots in the fields of philosophy, artificial intelligence, and knowledge engineering. [Gruber] defines an ontology as "a specification of a conceptualization." The main purpose of an ontology is to "enable communication between computer systems in a way that is independent of individual system technologies, information architectures, and application domain" [Gruber].

A key point is that capturing the data structure alone is not sufficient. Even mnemonic tags for data elements are insufficient to provide the meaning of the data content. XML is touted as a means for capturing data structure and content. We wished to capture data meaning as well (i.e., data syntax and semantics). Combining the data structure, content, and meaning together to provide consistent and unambiguous data definitions and relationships is what we refer to as an ontology. This point is captured graphically in Figure 1-1. Note that we refer to these components collectively as an ontology. Some would consider only the topmost layer, comprised of an organized collection of terms and definitions, to be the ontology.

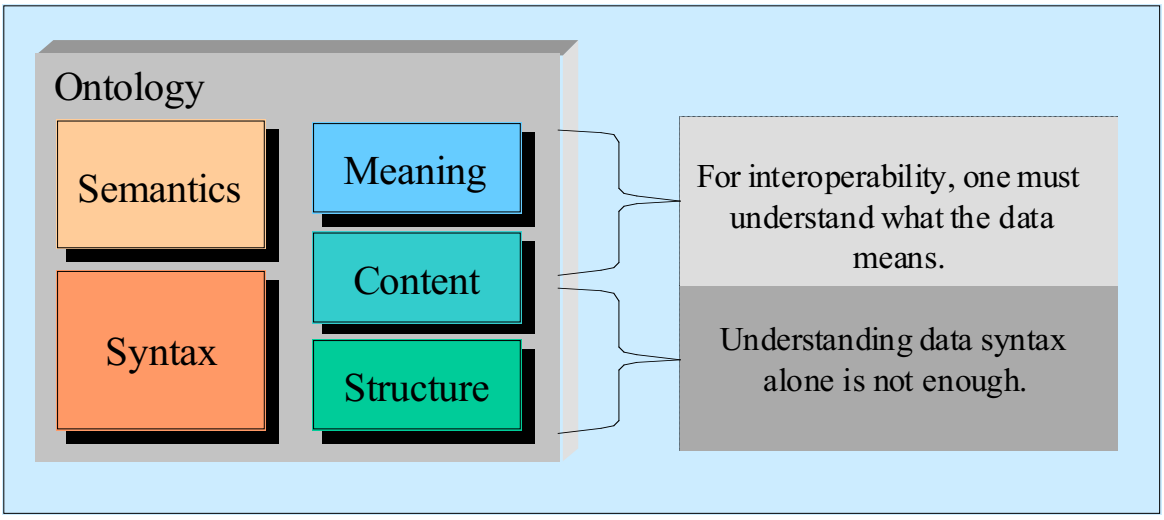


Figure 1-1. Necessary Components of Data Interoperability

1.3 Ontology Task Goals

The goal of this task was to develop a preliminary space ontology using XML technology, focusing on the space surveillance domain. The purpose of developing the ontology was to provide insight into both the applicability of XML to the space domain and the process needed to describe and document space information objects using XML. We expected to provide information to our Electronic System Center (ESC) sponsor (ESC/ND) that could be used by their contractors as they determine the applicability of XML to future architectures

and their constituent space mission systems. We also hoped to begin a collaborative process to identify where XML use could have the biggest payoff and areas where use of XML had the highest risk.

XML is a key enabler of the JBI and an important step in realizing the JBI is to collect and expose XML data definitions. Therefore, another goal of this effort was to develop a preliminary ontology that could serve two purposes: 1) it could be the foundation upon which a larger space ontology could be built and 2) it could be used to pioneer a process for exposing space data definitions across the military, and potentially to the commercial world as well. In this way it could become a pathfinder for cross-community standardization of military space data.

Finally, a goal was to investigate and share information on related efforts and leverage these efforts as much as possible in the development of our preliminary space ontology.

1.4 Document Structure

The rest of the document describes what we did, our lessons learned, and our conclusions. Section 2 describes the ontology development process. Section 3 describes the Space Surveillance Ontology developed. Section 4 provides our conclusions. The body of the document is followed by a list of references and several appendices. Appendix A provides XML background. Appendix B gives a brief synopsis of several related efforts. Appendix C summarizes the XML schema registration process. Appendix D contains the actual schema files that comprise the Space Surveillance Ontology. Appendix E discusses the space surveillance data transformations we developed using the Extensible Stylesheet Language (XSL). The appendices are followed by an acronym list.

Section 2

Ontology Development Process

This section describes the major steps in the process we used to develop the Space Surveillance Ontology.

2.1 Developed Preliminary Ontology Approach

Our first step was to define a proposed ontology approach. As mentioned in Section 1.2, an ontology provides consistent and unambiguous data definitions and relationships. Originally, we planned to capture the syntax, semantics, and interrelationships of domain information elements using the three-tiered approach described below.

1. **Concept Dictionary**: A repository for common terms and definitions. For the space ontology it was to contain terms such as satellite, element set, etc. This highest-level dictionary would provide the foundation upon which the abstract dictionary was built.
2. **Abstract Dictionary**: A repository for major object/element definitions. This would define a taxonomy of the objects, to include a description of the key objects, the relationships between these objects, and definitions for the aggregation of elements into objects. For example, the satellite observation object would contain data for azimuth, elevation, etc.
3. **Abstract Schema**: A repository for object definitions of sufficient detail that it could be instantiated into XML schemas and database tables. The abstract schema would also define the element attributes (e.g., units used) and map equivalent data elements across objects.

2.2 Researched Relevant Efforts

To launch our ontology development effort, we researched relevant ontology and XML efforts. We hoped to uncover good examples of ontologies and information on how to capture an ontology using an XML approach. We discovered many articles that touted the importance of defining ontologies and capturing semantic information within them. A recurring theme in these articles was that data structure, without understanding the meaning of the data, was of marginal value. However, we were unable to find sample ontologies relevant to our work.

We did discover examples of efforts to create standard schemas. For example, Interface Control Systems, Inc. is developing a Spacecraft Markup Language (SML) Specification to provide standard definitions of spacecraft and support data objects (e.g., packets, commands, messages). Also, NASA Goddard Astronomical Data Center (ADC) created an XML

markup language for documents containing major classes of scientific data. However, we were not able to discover any efforts that addressed how to capture semantic information as well. See Appendix B for more information on these and other related efforts.

2.3 Selected Target Domain

Our next step was to select the target domain. We chose space surveillance for a preliminary space ontology for five reasons. First, space surveillance is a key mission for United States Space Command (USSPACECOM). Second, maintaining a catalog of where objects are in space is a prerequisite for every other operation that uses space assets. Therefore, we viewed space surveillance as a foundation area in the military space domain. Third, the data content and usage within this domain is well understood. Consequently, we would not complicate the effort by trying to define how data is to be used. Fourth, this is a domain where data content, although vital, does not have stringent delivery time requirements. We know XML tags add overhead and did not want the value of an XML-based approach obscured by stringent performance constraints. Finally, this is the mission area targeted by a related XML demonstration effort.

Once we determined that our target domain was space surveillance, we developed a "contextual framework" document for the preliminary space surveillance ontology. This document described our target domain in a military space mission context and summarized the linkage of the space surveillance ontology effort to two key efforts: the Defense Information Infrastructure Common Operating Environment (DII COE) XML Registry and the NORAD/USSPACECOM Warfighting Support System (N/UWSS) Operational Architecture. For more information see [Pulvermacher].

Space surveillance is only one component of a broad military space domain. Military space is only one piece of a much larger space domain, as shown in Figure 2-1.

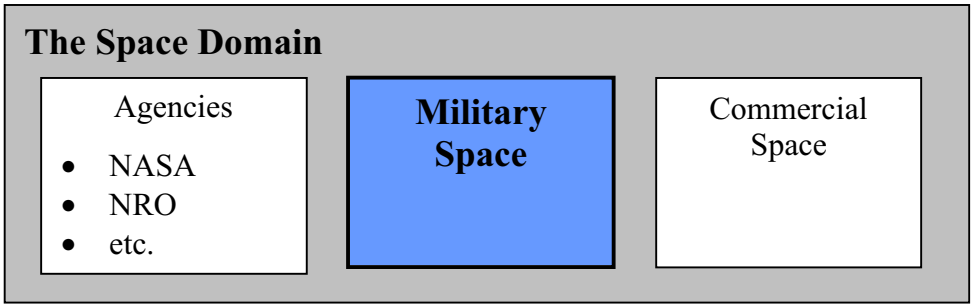


Figure 2-1. The Space Domain

This military space domain can be decomposed to the target space surveillance mission, as shown in Figure 2-2.

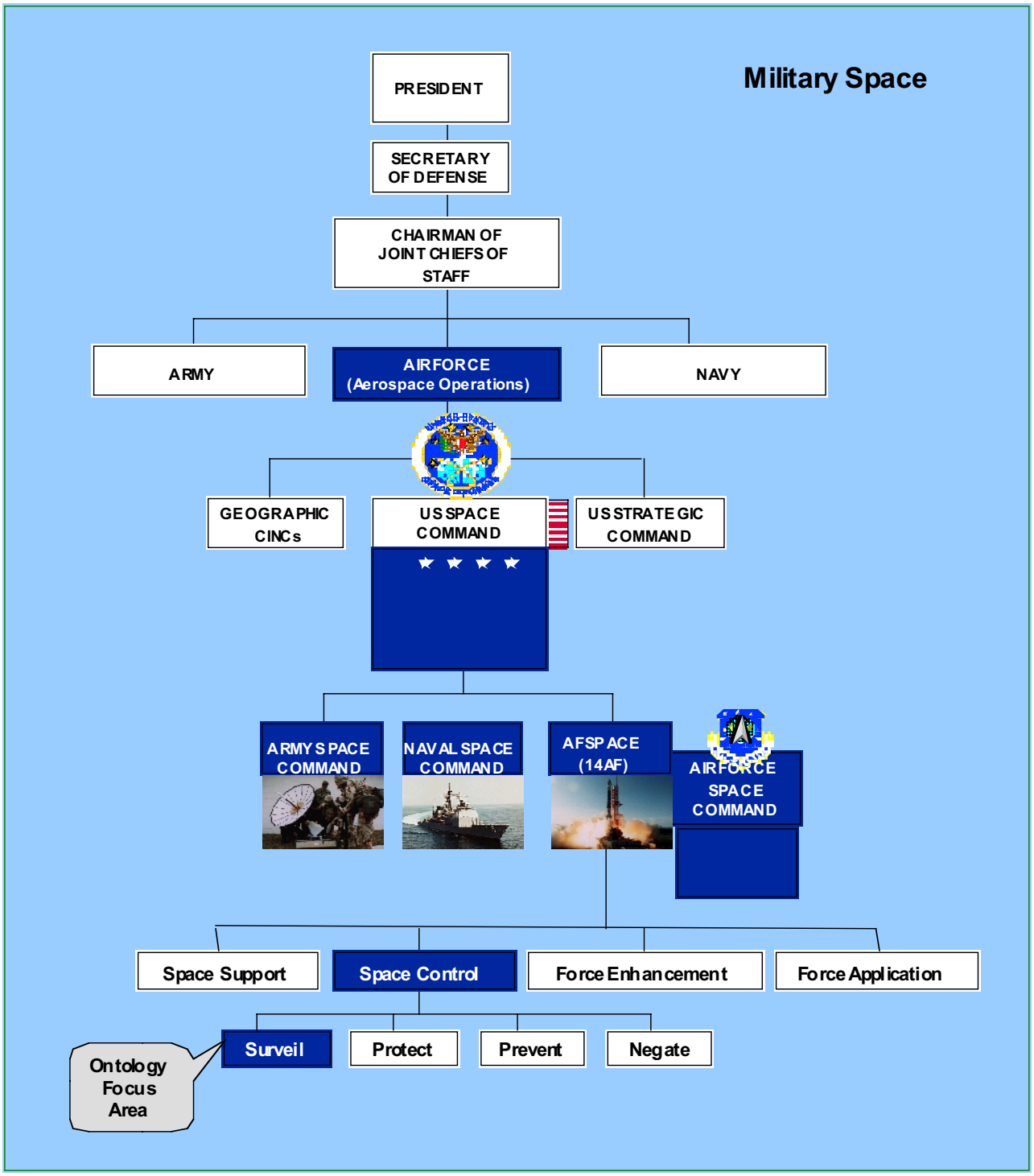


Figure 2-2. Decomposition of Military Space to Target Mission

As shown in Figure 2-2, space surveillance is a sub-mission of space control. The U.S. Space Command Brochure[2] lists four USSPACECOM missions:

1. Space Forces Support—Launching and operating satellites.
2. Space Force Enhancement—Supporting joint-service military forces worldwide with intelligence, communications, weather, navigation, and ballistic missile attack warning information.
3. Space Force Application—Engaging adversaries from space.
4. Space Force Control—Assuring U.S. access to, and operation in, space—and denying enemies that same freedom.

Both the 14th Air Force and Air Force Space Command inherit these USSPACECOM missions. Therefore, although the preliminary space surveillance ontology is being developed by the Air Force, it applies to a unified mission of USSPACECOM as well.

The preliminary ontology does not address all aspects of the space surveillance mission. Instead, it focuses on aspects related to space satellite catalog maintenance. This includes the development and distribution of satellite element sets (elsets), the receipt and processing of satellite observations used to maintain the satellite catalog, and tasking sent to ground-based sensors to collect observations to support satellite catalog maintenance.

The DII COE XML Registry (see Appendix B for more information on this XML registry) has a namespace called "Aerospace Operations." Space surveillance is one subdomain of Aerospace Operations. Therefore, one could depict our target domain relative to this namespace, as shown in Figure 2-3.

---

2 Available online at: http://www.spacecom.af.mil/usspace/newfctbk.htm

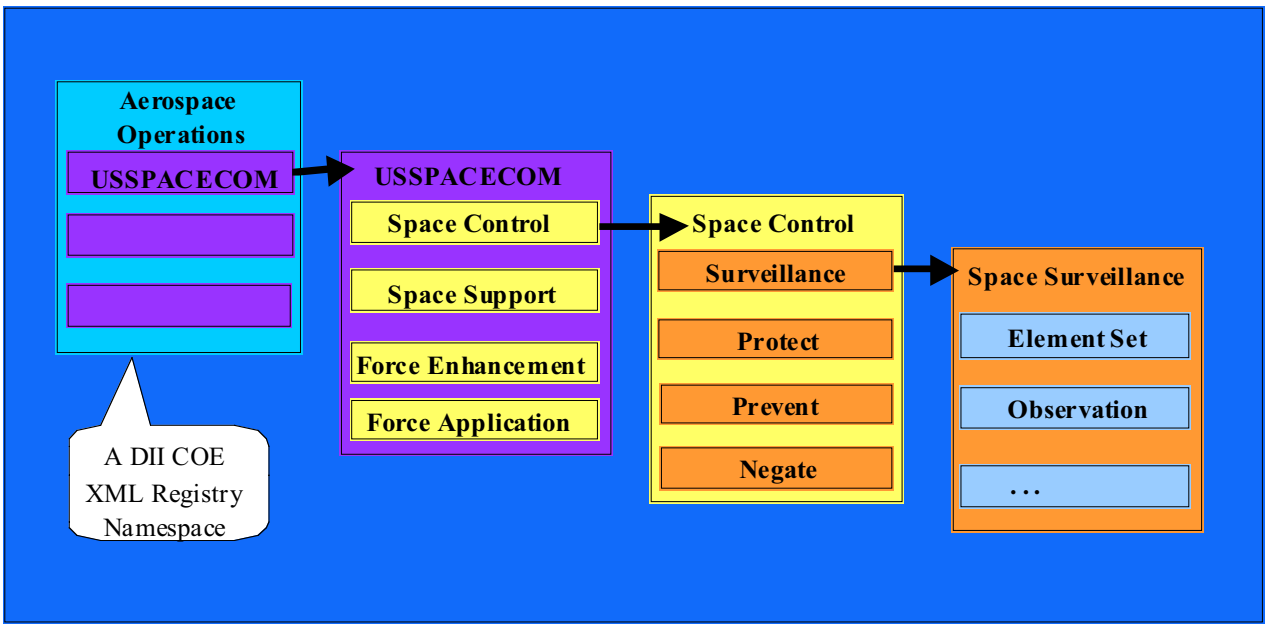


Figure 2-3. Targeted Subset of Aerospace Operations

2.4 Selected XML Schema Rather Than DTD Vocabulary

In our research on XML, we read several articles about the relative merits of defining XML elements using XML Schema definition language instead of DTDs. These articles convinced us that the preferred approach was to use the XML Schema standard being proposed by the World Wide Web Consortium (W3C). Mikula and Levy authored a particularly good article on this [Mikula]. While DTDs are in wide use today, we strongly believe XML schemas will supplant them.

Some of the advantages that XML schemas provide over DTDs include:

- **Strong data typing**: The XML Schema specification provides over 40 built-in data types. Further, you can create your own data types. These data types allow automatic error checking for things like range of values checks, membership in an enumerated list, or conformity to a data pattern (e.g., legal zip code pattern of numbers).
- **Same syntax**: The XML Schema document (".xsd" file) is written in the same syntax as an XML instance document. Both are well formed XML documents. This means that many of the same tools can operate on both files (i.e., on both ".xsd" and ".xml" files). This makes it easier and cheaper for vendors to provide tools and makes it easier to write code to operate on these files. In fact, as discussed later, we use this feature to allow access to semantic information captured in the schema file.

- **Support a type of inheritance**: With schemas one can define types that extend or restrict a previously defined type, whether that type is built-in or defined by the user. Inheritance of elements is also supported. This provides an approach similar to that used in the object-oriented programming domain.
- **Namespace aware**: XML schemas also support the use of namespaces as defined by the W3C. XML namespaces provide a simple method for associating element, attribute, and user defined type names to a particular domain. In this way, the same name can be used in different domains without name collision. These names can be used together in the same XML document without confusion. Also, namespaces facilitate the reuse of previously defined elements, attributes, and types.
- **Allows equivalent elements**: XML schemas allow elements to be substituted for other elements. Basically, this allows one element to be called by two different names. Note that in the newest version of the XML Schema specification, this feature has been renamed to *substitutionGroup.*

The W3C has not yet approved the XML Schema specification. Figure 2-4 shows the maturity of the XML Schema specification which has just been issued as a W3C Candidate Recommendation. This specification is sufficiently stable that many tool vendors are using it to start their development process.

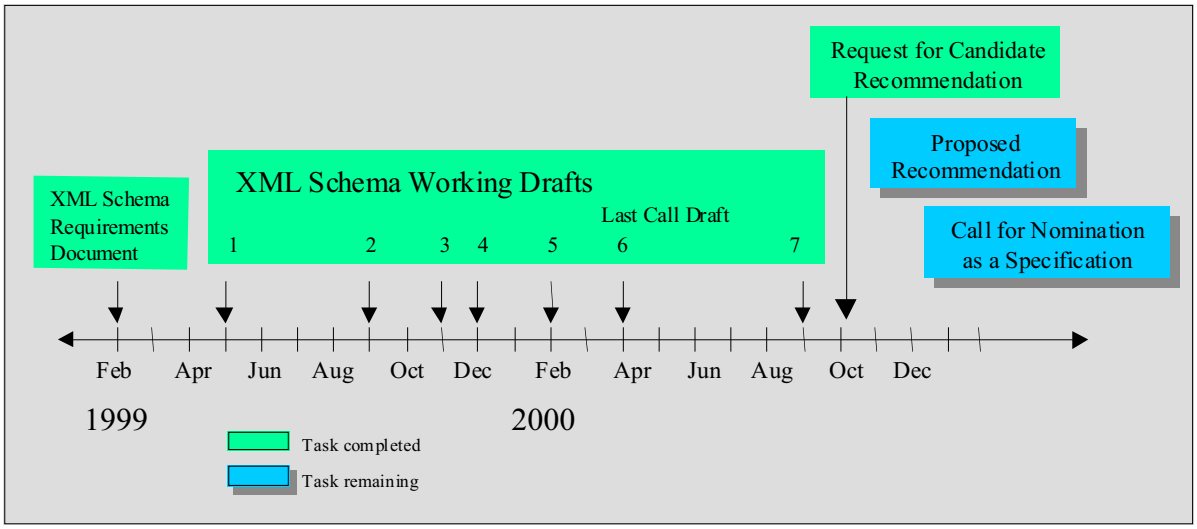


Figure 2-4. W3C Process – XML Schema Status

2.5 Participated in XML Training

Our next step was to extend our knowledge with hands-on training. We took a one-week XML training course offered by MITRE's internal training organization, The MITRE

Institute. Dr. Roger Costello, MITRE, taught this introduction to XML and its associated family of technologies. The course provided the foundation of knowledge needed to determine our final ontology development approach and actually develop a space surveillance schema. The in-class exercises were especially helpful for cementing these new concepts.

2.6 Developed XML Ontology Approach Using Only XML Schemas

Although we began drafting a concept dictionary, we were uncomfortable with capturing this information in a document separate from the data syntax. Therefore, we continued to seek ways to capture the structure and content of data, as well as definitions and elaborating information, in an integral way.

We considered several possible approaches for combining syntax and semantics. For example, we discussed using Resource Description Format (RDF) since it was designed to capture metadata of resources on the web. However, it uses a different syntax and must be associated with a web address. We also considered creating a separate description element or attribute and populating it with default content. One problem with this approach is that we did not want all the description information to be included in every instance document. Although the XML schema specification is not final, our research indicated that a validator should populate instance documents with default content.

Our selected approach was to capture our Space Surveillance Ontology using XML Schema definition language. (Note: Because of this we use the phrases Space Surveillance Ontology and space surveillance schema interchangeably in this paper.) This approach allowed us to capture data semantics within the XML Schema supplied *annotation* element. We adopted a convention that every element, attribute, and type must include an annotation that provides descriptive information. The specifics on how this approach works are described in Section 3.2. We shared this approach with Dr. Costello, the instructor for our XML training course, who incorporated it into his course as a possible strategy for defining the semantics of XML elements.

Although this approach is not perfect, we believe it addresses all key requirements we identified for the Space Surveillance Ontology. Table 2-1 summarizes these requirements and describes how our XML Schema approach satisfies them.

There are several advantages of this approach, many of which are summarized in Table 2-1. First, the XML schema will inherently capture the data structure, metadata, and relationships between the elements. Second, use of strong typing will capture much of the data content restrictions. Third, because annotations will be used to capture definitions and other explanatory information, the overall structure of the "definitions" will always be consistent with the schema's structure. Finally, since the schema itself is an XML document, we can use XML transformation tools to extract the annotations and transform this "semantic" information into a format suitable for human consumption.

Table 2-1. Space Surveillance Ontology Requirements

Requirement	XML Schema Approach
Capture data structure and content	Inherent part of XML schema (DTDs define structure; schema data types add content information)
Capture relationships between data elements	Inherent part of XML schema
Capture data semantics	Use XML annotation element to capture data definitions Use data attributes to capture metadata
Make data, metadata, and semantics accessible to a wide variety of clients	Use XSL to transform this XML tagged data to make it readable to humans or other computers
Ensure parallelism between data structure definition and data semantics	Consistency ensured by having both parts of the same XML schema

2.7 Researched XML Tools

While researching related XML efforts, we also began researching the types of XML tools available. Once we decided that we would develop the Space Surveillance Ontology using XML Schema definition language, we began focusing our search on tools that would support XML schema development. There are several tools available that are DTD based. However, because the XML Schema specification is still a working draft, there are few XML schema aware tools available. Understandably, these XML schema aware tools are not mature.

One tool we examined in detail is XML Authority by Extensibility. (See http://www.extensibility.com/ for more information.) MITRE had previously purchased several licenses for this commercial tool and made some of these licenses available to our team. We found this tool useful for developing the overall structure of an XML schema file. One very nice feature was the ability to graphically depict elements and their inter-relationships. However, we found that the current version of XML Authority does not support many of the XML Schema features we wished to use. In fact, the tool deleted or altered code for features not yet implemented. Once we had a preliminary structure in place, we abandoned use of this tool and reverted to a simple editor. We do believe that XML Authority will be a powerful XML schema development tool once the XML Schema features are fully implemented.

2.8 Developed Space Surveillance Schema

Our approach for developing the Space Surveillance Ontology using XML Schema description language relied heavily upon strong domain knowledge. Our domain expert has strong operational knowledge acquired from over 10 years of experience in this field. We used this domain knowledge to determine our key data objects (i.e., major elements) and their contents. Once we defined the major elements, we informally stepped through a scenario of how the elements would be used to confirm that the proposed structure supported current and anticipated future operations. We then described these elements in an XML schema and validated the schema against an instance document. The validation process checked that the schema and instance document were well formed, that the instance document conformed to the rules described in the schema, and that the schema conformed to the XML Schema description language rules (sometimes called the schema for schemas).

Through this entire process we maintained a space surveillance domain perspective. This means that not all aspects of the major elements are described; rather, we captured only those that are relevant to space surveillance. For example, our *Satellite* element includes child elements for *InternationalDesignator, CommonName, Owner, SatelliteObject, LaunchDate,* and *DecayDate.* However, it does not include mission specific information or other information needed for satellite telemetry, tracking, and commanding (TT&C).

In the development of our Space Surveillance Ontology, we tried to harness the power of the XML Schema definition language. To do this we experimented with many of the key features of the XML Schema vocabulary. We tried to be consistent in our approach across all major elements in the schema; however, in some cases, we tried multiple approaches to learn more about the tradeoffs between the various options. The conventions we used and design decisions we made in developing the Space Surveillance Ontology are captured in Section 3.

2.9 Developed Ontology Demonstration Approach

Our next step was to develop an ontology demonstration approach to allow our sponsor to view this XML schema and our proof-of-concept stylesheets. Our demonstration approach allows viewing of the XML schema files, the relationships among elements within a file, the XML instance documents (i.e., instances of XML tagged space surveillance data), and the results of applying XSL to both the schema files and instance files.

Throughout the development of the Space Surveillance Ontology, we recognized the need to represent the ontology graphically and textually. To satisfy this need we used a combination of commercial off-the-shelf (COTS) and custom solutions. XML Authority is the COTS solution, and it provides the capability to easily navigate and explore the ontology. This application allows the user to view the hierarchical relationships present in a schema, as well

as view the attributes and constraints placed on the elements. As noted earlier, XML Authority has preliminary releases available, but not all XML Schema features are implemented.

A more basic and accessible viewing tool is Microsoft's Internet Explorer 5 (IE5). IE5 is XML aware and, as such, is able to create a collapsible view of a well-formed XML document. The XML Schema documents are well formed, and IE5 uses color coding to differentiate between data tags and data content. A collapsible tree accurately represents the element relationships, but navigation through the schema is not as easy as it is with XML Authority.

An advantage of using XML schemas to capture the ontology is the ability to use other XML tools to manipulate the presentation of the ontology. This is an advantage that DTDs do not enjoy. We developed an XSL stylesheet to transform the XML Schema documents. Using XSL, we generated a schema view that includes all schema element definitions and any documentation associated with the elements. The transformation extracts this information and wraps it with Hypertext Markup Language (html) tags to make it viewable on any browser (i.e., even those that are not XML aware). This was done with approximately 25 lines of XSL instructions. The source code listing for this example is included in Appendix E. Additional XSL stylesheets can be developed in order to extract specific components of the metadata, such as attributes with relatively few lines of instructions.

Although we used XSL to transform the schema documents, stylesheets are more frequently used to transform instance documents. We developed samples of instance document transformations and included an example in Appendix E.

2.10 Researched Registration Process for DISA XML Registry

One of our goals in developing the Space Surveillance Ontology was to be able to share space data structure and content, as well as data meaning. In our research on related efforts, we found that XML registries do exist, and the one most relevant to the military space domain was developed by the Defense Information Systems Agency (DISA). (See Appendix B for more information on XML schema repositories.)

DISA has developed an XML registry called the DII COE XML Registry. The DISA DII COE XML Registry is intended to help ensure interoperability and expose XML data definitions. It provides a baseline set of XML tags developed through coordination and approval among the COE community. The registry is organized into eleven namespaces, with the "Aerospace Operations" namespace most relevant to the military space domain.

We talked to the MITRE staff supporting DISA on this effort and shared information on our approach with them and their DISA sponsor. We determined that, at the time of our space surveillance schema development, no other organization was developing an XML schema for the Aerospace Operations namespace. We also found that no one had shared with them an

ontology approach (i.e., way to capture data syntax and semantics) using XML Schema definition language. Consequently, DISA is very interested in the results of our effort.

We concur with DISA's goals in creating this registry and hope the Space Surveillance Ontology becomes a part of it. Therefore, we researched the process for registering with the DISA XML Registry and captured this information in Appendix C.

2.11 Documented Space Surveillance Ontology

The final step in developing the Space Surveillance Ontology was to document our results. We wrote this paper to document not only the ontology itself, but also the process we used to develop it. We do this in the hopes that some of this information will prove useful to three categories of users: 1) those who may take this work and extend it, 2) those in the space domain who may wish to use the ontology, and 3) others who are attempting to create XML schemas within their own domains.

Section 3

Space Surveillance Ontology

This section describes the Space Surveillance Ontology we developed. It describes the tools we used, how the XML schema serves as an ontology, the major elements and their inter-relationships, selected features that make schemas powerful tools, the conventions and design decisions used in constructing the schema, and lessons learned during development.

3.1 Tools Used

The primary tools used in the development of the Space Surveillance Ontology were:

- A simple **editor**: We used Microsoft Windows 95 Notepad.
- An XML **viewer**: We used Microsoft's Internet Explorer version 5.5 to view the XML instance documents. We could also use it to check schema files for "well-formedness" if we changed the file extension from ".xsd" to ".xml."
- An open source **validator**: We used the Apache Xerces Validating Parser for Java. (See http://www.apache.org/ for more information.)
- A commercial **XML schema development tool**: We used XML Authority by Extensibility. (See http://www.extensibility.com/ for more information.)
- The latest version of the **XML Schema specification**: The space surveillance schema contained in this document is compliant with the April 2000 working draft because that is the version to which our validator was compliant. (See http://www.w3.org/XML/Schema for more information.)
- An open source **XSL translator**. We used XT, written by James Clark. (To obtain a copy, see http://www.jclark.com/xml/xt.html.)

3.2 XML Schema as the Ontology Repository

We developed an approach for capturing the Space Surveillance Ontology using XML Schema definition language. Our rationale for selecting this approach is described in Section 2.6. This section describes the planned approach.

Basically, the approach is to capture the data structure, metadata, and data definitions within the XML schema itself. The data structure and relationships between data elements are an intrinsic part of an XML schema. We capture data content through the use of strong data typing, metadata through the use of element attributes, and data definitions through the use of the XML Schema provided *annotation* element. This annotation element has two possible child elements: *appInfo* and *documentation*. The appInfo child element is intended to

provide information for tools, stylesheets, and other applications. In other words, it is intended for nonhuman readers. The documentation child element is the recommended location for human readable material. Therefore, we created a convention that all elements, attributes, and types must include a documentation annotation. In these documentation annotations, we tried to include the kind of information a user of the ontology would need to understand the ontology (i.e., definitions and explanatory information).

One advantage of this approach is that the structure of the "definitions" will always be consistent with the structure used in the schema, since the definitions are an integral part of the schema. Also, because the schema itself is an XML document, we can use XSL to transform this information into a more concise, easier to read format.

Figure 3-1 shows a cartoon of our Space Surveillance Ontology approach. As shown in this figure, the XML schema contains the data structure, metadata, and definitions. A file containing XML tagged data is shown as an "instance of data content." An XML validator operates on an instance document. This instance document tells the validator where to obtain the XML schema that contains the rules for describing data in this namespace. The validator validates that the instance document is *well formed* (i.e., follows the rules of XML) and *valid* (i.e., follows the rules defined in the schema). The result, in our case, is a valid instance of space surveillance data (i.e., data and metadata).

This instance document can then be operated upon using XML tools. For example, one could view this instance in Microsoft's IE5, which displays XML documents in a collapsible tree format. Or, as shown in Figure 3-1, one can apply XSL to transform this data. One could have many stylesheets that operate on the instance to develop data presentations. We depict these data presentations as files in html format because this is the presentation format currently supported by Internet browsers. However, we anticipate that more browsers will become "XML aware" and allow viewing of XML (.xml) and XML Schema (.xsd) formatted files.

The interesting thing to note is that because an XML Schema file is a well-formed XML document, the same set of tools used for *.xml* files can be used to operate on *.xsd* files. Therefore, we can create a data transformation to extract XML schema definitions contained in the documentation annotations and store them in a format readable by a browser. We developed a stylesheet to prove this approach works. (See Section 2.9 and Appendix E for more information on our proof-of-concept stylesheets.)

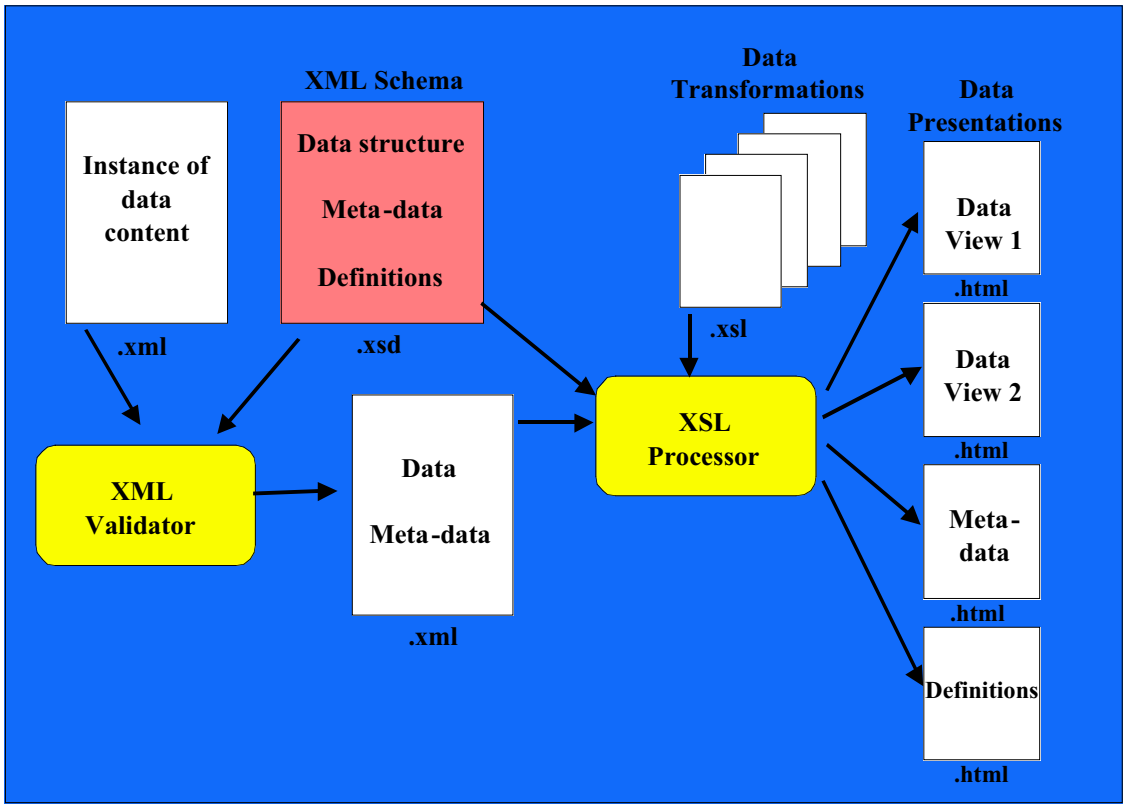


Figure 3-1. Space Surveillance Ontology Approach

3.3 Major Elements and Their Interrelationships

In this first endeavor at a space surveillance ontology, we defined several major objects and their relationships. For each major object, we developed a *.xsd* schema document to define elements and attributes of the object within the context of the space surveillance domain. The schema files developed for the ontology are intended to provide a framework for future development. The very nature of XML is eXtensibility, and our intent is that the schema be extended in the future. Below is a diagram that shows the relationships between the major objects, followed by a brief description of these objects. Appendix D contains the space surveillance XML schema files.

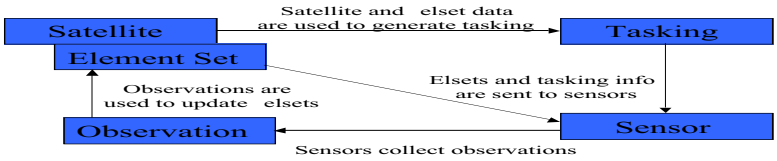


Figure 3-2. Space Surveillance Schema Element Relationships

3.3.1 Satellite

The *Satellite.xsd* schema file describes the satellite object within the Space Surveillance Namespace. A satellite number uniquely identifies each USSPACECOM catalogued object. The *Satellite* element also has child elements for international designator, common name, piece type, owner, mission(s), payload status, launch information, and decay date. The satellite object contains information currently stored in the Satellite Catalog (SATC) file in the Space Defense Operations Center (SPADOC). Although each satellite has an orbit that is characterized by a satellite elset, we choose to store elset information in a separate XML schema file. One reason for this is that satellite elsets are very dynamic data, whereas satellite object data is less dynamic. Therefore, we anticipate these data objects would be used in very different ways. Space defense type data would be stored in yet another XML schema file due to the nature and classification of the data. Currently satellite, element set, and space defense data are all stored in the same SATC file on SPADOC. The aggregation of this data in a single SPADOC database file has led to problems with database access performance and classification levels.

3.3.2 Satellite Elset

The satellite elset schema file (*SatElset.xsd*) describes the general perturbations (GP) elset data (often referred to as a two-line elset). Satellite elset data can be uniquely characterized in one of two ways: 1) by satellite number (*SatellieNumber*) and epoch time (*Epoch*) or 2) by satellite number and elset number (*ElementNumber*). Each satellite has a current satellite elset and a history of past satellite elsets.

3.3.3 Sensor

The *Sensor.xsd* schema file describes elements and attributes of a ground-based, space surveillance network (SSN) sensor. In the future, the sensor schema should be extended to

include other types of sensors (e.g., Air Force Satellite Control Network (AFSCN), Intelligence, Missile Warning). The sensor schema could become part of a larger schema, ground sites, for example. This schema contains elements for sensor number, sensor name, sensor short name, sensor mission(s), sensor location, sensor device(s), and the country of ownership. A sensor receives elsets and tasking and produces satellite observations.

3.3.4 Satellite Observation

The *SatObservation.xsd* schema file describes elements and attributes of a SSN observation in what is commonly called "B3 format." "B3 format" is a phrase used by SPADOC and implies a particular format for the data, including the units used in the observation elements. The schema contains child elements for the types of satellite observations used today in the SSN. These are observation types 1-5 and 9. Other types of observations can be added to the schema in the future. Each observation element contains an attribute of units; this allows the *SatObservation* schema to process non-B3 format observations simply by adding the proper units and field descriptors to the schema as needed. For example, the SatObservation schema could be extended to include AFSCN observations without creating a new schema or message format. A sensor collects observations and sends them to a correlation center (e.g., SPADOC) where they are used to produce new elsets.

3.3.5 Sensor Tasking

The *SensorTasking.xsd* schema file describes the data required to task SSN sensors to perform the space track mission. The current sensor tasking message sent to the SSN sensors contains satellite numbers and tasking codes. Tasking codes describe to the sensor the priority and amount of data to collect on each satellite. In the SensorTasking schema, we added optional information for tasking metrics and tasking control data. This was done to demonstrate the flexibility XML provides for adding optional data to schemas for completeness. This approach could enable a migration strategy for new functionality where data could be added to schemas as optional until all users could process the additional data.

3.4 Selected Schema Features

This section describes some powerful XML Schema features and provides examples of how we used them in the Space Surveillance Ontology.

3.4.1 Strong Typing

XML Schema definition language is a "strongly typed" language. It provides many built-in types, as well as the ability for users to define their own data types. These user defined data types can be simple types or complex types. Simple types are derived from built-in types and may not contain element content or carry attributes. Complex types allow elements in their content and may carry attributes.

Strong data typing allows one to more precisely define a piece of data. For example, with XML Schema data types, you can define a data item to be of type "integer," and also bound the allowable range of values using "minInclusive" and "maxInclusive" (called data type facets in XML schema parlance). This ability to precisely define the data means that data errors can be found at the time of data input using code that is an intrinsic part of the technology. This saves money because incorrect data is found at the start of the process and the amount of code needed to perform data type checking is drastically reduced.

In this section we provide an example of the use of XML Schema built-in types, defined types, and derived types.

3.4.1.1 Built-In Types

XML Schema definition language provides over 40 built-in types. These include types such as string, boolean, date, time, positiveInteger, nonNegativeInteger, etc. Figure 3-3 shows an example of using the *nonNegativeInteger* built-in type in an element definition.

```
<element name = "TaskCapacity" type = "nonNegativeInteger">
        <annotation>
                <documentation>TaskCapacity is the maximum number of tracks a sensor
                                        can perform in a 24-hour period.
                </documentation>
        </annotation>
</element>
```

Figure 3-3. Built-In Type Usage Example

3.4.1.2 User Defined Types

XML Schema definition language allows a user to define their own types. These types can be simple or complex, and can also be defined to be global or anonymous. Global types can

be reused both within the defined namespace or other namespaces. Anonymous types are embedded within another type definition or an element declaration. Anonymous types prohibit reuse, but save the overhead of having to be explicitly named and referenced.

We show four examples: 1) Figure 3-4 is an example of defining a simple type with a legal range of values of 1-99999, 2) Figure 3-5 is an example of the use of enumerated types (i.e., the data must be one of the listed enumeration values); 3) Figure 3-6 shows an example of a complex type that includes a documentation annotation and two child elements, and 4) Figure 3-7 is an example of an anonymous type definition. In this example, the enumeration type is defined within the element declaration.

```
<simpleType name = "satelliteNumberType" base = "positiveInteger">
        <maxInclusive value = "99999"/>
</simpleType>
```

Figure 3-4. User Defined Simple Type Example

```
<simpleType name = "satellitePayloadStatusType" base = "string">
        <enumeration value = "Active"/>
        <enumeration value = "Inactive"/>
        <enumeration value = "MissionEnded"/>
        <enumeration value = "Dead"/>
</simpleType>
```

Figure 3-5. User Defined Enumerated Type Example

```
<complexType name = "dateTimeType">
        <annotation>
                <documentation>Definition of a Date and Time element group.
                                Format of Date is: CCYY-MM-DD
                                Format of Time is: hh:mm:ss.sss
                </documentation>
                </annotation>
        <element name = "Date" type="date"/>
        <element name = "Time" type="time"/>
</complexType>
```

Figure 3-6. Complex Type Example

```
<element name = "EphemerisType">
        <simpleType base = "string">
                <enumeration value = "SGP"/>
                <enumeration value = "SGP4"/>
        </simpleType>
        <annotation>
                <documentation>Theory used to calculate elements.  Values may be:
                                        SGP (Simplified General Perturbations) or
                                        SGP4 (Simplified General Perturbations 4)
                                SGP is an analytic method of generating ephemerides
                                 for satellites in earth-centered orbits.
                </documentation>
        </annotation>
</element>
```

Figure 3-7. Anonymous Type Example

3.4.1.3 Types Derived by Extension

Each user defined type described above is derived from a base type. This ability to start with something already defined and alter it stems from the object-oriented world concept called *inheritance*. A more powerful example of the use of an object-like approach in XML Schema definition language is the ability to derive types by extension or restriction.

Derive by extension means that the new type inherits the content of the base type, but extends it in some way. It is also possible to derive new types by restricting the content of existing types. Figure 3-4 shows a simple example of this capability where the base type of positive Integer was restricted to no more than five digits in length.

Figure 3-8 shows an example of how the Space Surveillance Ontology uses derived by extension. A *Type1* satellite observation contains two child elements: *Elevation* and *Azimuth*. A *Type2* satellite observation inherits the *Type1* elements and extends this type. Thus, a *Type2* satellite observation contains three child elements: 1) *Elevation*, 2) *Azimuth*, and 3) *Range*.

```
<complexType name = "obType1Type">
        <annotation>
            <documentation>Type 1 satellite observations include:
                Elevation and Azimuth</documentation>
        </annotation>
        <sequence>
            <element ref = "ss:Elevation"/>
            <element ref = "ss:Azimuth"/>
        </sequence>
    </complexType>

    <complexType name= "obType2Type"
            base= "ss:obType1Type" derivedBy="extension">
        <annotation>
            <documentation>Type 2 satellite observations include:
                Elevation, Azimuth, and Range</documentation>
        </annotation>
        <sequence>
            <element ref = "ss:Range"/>
        </sequence>
    </complexType>
```

Figure 3-8. Type Derived by Extension Example

3.4.2 Attribute Groups

Another capability provided by XML Schema definition language is the ability to define groups of attributes that are used together. Figure 3-9 contains an example of how the Space Surveillance Ontology uses an attribute group to declare classification attributes.

```
    <attributeGroup name = "classification">
        <attribute name = "classLevel" type = "ss:classificationLevelType"
                use="required"/>
        <attribute name = "dissemination" type = "ss:disseminationType"
                use="optional"/>
        <attribute name = "classificationGuide" type = "string"
                use="optional"/>
    </attributeGroup>          </sequence>
```

Figure 3-9. Attribute Group Declaration Example

3.4.3 Defined Namespaces

Another key feature of XML schemas is its support of namespaces. As stated in the XML Schema Specification [Schema0], "A schema can be viewed as a collection (vocabulary) of type definitions and element declarations whose names belong to a particular namespace called a target namespace. The target namespace enables us to distinguish between definitions and declarations from different vocabularies." This allows elements to be understood in the context of their target namespace. For example, a "Target" in an Air Operations namespace would be different from a "Target" in a commercial marketing namespace. Yet one could use both "Target" elements together with namespace qualifications ensuring there are no conflicts. This is called naming disambiguation.

Namespaces allow one to declare elements and define types relevant to a particular namespace without concern that they may conflict with declarations and definitions in other namespaces. Namespaces also facilitate reuse. Why reinvent an element declaration if it already exists? Rather, one can "import" element definitions from other namespaces, thereby saving effort and promoting standardization.

For the Space Surveillance Ontology, we defined a target namespace for space surveillance and included all element and attribute declarations and type definitions in this target namespace. We named the target namespace "urn:schemas-gov:SpaceSurveillance." At the time we began our effort, the XML Namespace Registry contained no XML schemas; therefore, we did not reuse any previously defined elements, attributes, or user-defined types by importing them into the space surveillance schema.

Using namespaces and XML schemas could be a powerful enabler. Exposing data definitions in a repository like the XML Namespace Registry could lead to common data abstractions, common data definitions, and common metadata. This would be an important step toward achieving the JV2020 vision of Integrated C2 across all services.

3.5 Schema Conventions Used and Design Decisions

This section describes the conventions we developed and design decisions we made in developing the Space Surveillance Ontology. In some cases, we tried multiple approaches as a way to examine alternatives.

3.5.1 Naming Convention

Our naming convention was to begin element names with an initial capital letter and begin attribute and type names with lower case. Also, all type names end with the word *Type.* For example, *SatelliteNumber* is the name of an element, *satelliteNumber* is the name of an attribute, and *satelliteNumberType* is the name of a type. Our name choice was also biased toward having self-describing names over short, bandwidth conserving names.

3.5.2 Annotation Convention

To capture definitions of terms and explanatory information, we developed a convention that all elements and attribute declarations and type definitions must have a documentation annotation.

3.5.3 Namespace URI

The XML Schema Specification states that the namespace location must be a valid Universal Resource Indicator (URI). A URI may be either a Universal Resource Locator (URL), Universal Resource Name (URN), or both (for more information on URIs, see [URI]). We thought using a URL would be confusing because many people associate a URL with a web address. Therefore, we defined the space surveillance target namespace, using URI syntax, as *urn:schemas-gov:SpaceSurveillance.*

3.5.4 Atomic Elements

Another convention we used was to make the elements as atomic as possible. By this we mean that each element should be broken down to its fundamental pieces with a bias toward making the elements global so they can be reused. However, we did create some anonymous attributes and types. For example, the *Sensor* element and its children are declared to be global. However, the global element *Latitude* has a *hemisphere* attribute with an anonymous enumerated type with allowable values of "*N*" or "*S.*"

3.5.5 Overall Schema Architecture

Our approach for the Space Surveillance Ontology design was to partition the schema into modules where each module contains a logical grouping of declarations and definitions. Thus, each major element is a separate module and that file contains all the element and attribute declarations and type definitions that are specific to that element or its child elements. Type definitions and attribute declarations that apply across the major elements are stored in a separate file called *SpaceSurveillanceGlobals.xsd*. These globals were stored in a separate file to facilitate reuse and isolate items that may change as the schema evolves.

An overall container element, called *SpaceSurveillanceData,* is declared in the root file called *SpaceSurveillance.xsd*. All modules are combined into a coherent whole through the use of the *include* statement. One prerequisite of using the *include* statement is that the schema files must declare the same target namespace.

Our decisions on how to segment the space surveillance schema relied heavily on our anticipated usage of the schema. For example, although a satellite has an orbit and an orbit is characterized by an elset, we did not make the element called *SatelliteElementSet* a child of the *Satellite* element. Rather, we made *SatelliteElementSet* a peer to *Satellite.* We made this

design decision because we anticipate that many instances of *SatelliteElementSet* will be created and shared. It would be inefficient to burden this high volume element with the overhead of the rest of *Satellite* element data.

3.5.6 Qualified Element Name and Attribute Name Default

Another decision we made was to require the qualification of locally declared elements and attributes. This can be done explicitly by setting two attributes on the *schema* element, *attributeFormDefault* and *elementFormDefault,* to *qualified.* This can also be done implicitly since *qualified* is the default value if *unqualified* is not specified.

Setting these attributes to *qualified* means that locally declared elements and attributes must have a namespace prefix. It also means that elements and attributes in the instance document must be namespace qualified (i.e., use a namespace prefix). However, if in the instance document you declare a default namespace, then elements and attributes from that default namespace need not be qualified. This makes an instance document that is easier to read and that clearly indicates any elements that are external to the target namespace.

3.5.7 Element Versus Attribute Declarations

There is no correct answer on whether something should be declared as an element or an attribute. A general rule of thumb is that attributes capture metadata (i.e., data about the data). We tried both approaches in our Space Surveillance Ontology. For example, the *Satellite* element has an attribute of *satelliteNumber.* However, *SatelliteNumber* is a child of the element called *SatelliteElementSet*.

3.5.8 Classification Approach

The space surveillance domain deals with information at many classification levels, from unclassified to top secret. We made it a priority to ensure that we addressed classification marking of data. However, we determined that there are many approaches to handling classification marking, each with its own set of tradeoffs.

The approach we implemented was to develop an attribute group called *classification* that contains three attributes, *classLevel, dissemination,* and *classificationGuide,* with use of the last two attributes being optional. We then assigned this attribute group to any element that could be classified.

The next question was how to handle those items that must be unclassified. For these elements we defined a *classLevel* attribute within the element, but fixed its value to be *U* for unclassified. This means that an instance of this element must contain ‘classLevel=”U”’ as an attribute.

The question remaining was what should be the convention on how to use these attributes? Should we require that all elements have a classification attribute? This would satisfy a

multilevel security environment where everything must be tagged with its classification level. However, this means that the size of the instance document could grow to be very large. This could be a problem with high-volume data sets.

Our approach was to tag only those items that could be classified or must be unclassified. Clearly, this is a design decision that is dependent on the overall security architecture and therefore should be readdressed once the security architecture is understood.

3.5.9 Enumerated List Value Definitions

One strength of the XML Schema definition language is the ability to define enumerated types. There are several ways to capture elaborating information on each enumeration item. For example, if the enumeration is an acronym, how do you capture what that acronym means? We used several approaches to demonstrate the variety of options available.

The easiest approach is to capture the elaborating information in an annotation element in the simple type definition. This was our standard approach. However, one can also use the source attribute on the documentation element to reference a document with elaborating information. You can specify a source document for the documentation element with the following syntax <*documentation source= "URL"*>. We demonstrate this approach in our *LaunchSite* element. Finally, you can also use an anonymous enumerated type within the element or attribute that will use it. The elaborating information is then included in the annotation element for that element or attribute. We demonstrate this approach in the *hemisphere* attribute of the *Latitude* element where the legal values may be either *N* or *S*.

3.6 Lessons Learned During Development

This section describes both the lessons learned and truisms that were reinforced during our ontology development effort. These lessons are captured in two categories: 1) Lessons Learned about the Space Surveillance Domain, and 2) Lessons Learned about XML Technology.

3.6.1 Lessons Learned About the Space Surveillance Domain

3.6.1.1 An Ontology Can Capture Needed Data Semantics

Every domain has data semantics, and it is important to capture these semantics to facilitate data interoperability. Data interoperability is a necessary step in achieving the visions touted in JV2020 and the JBI. An ontology deals with documenting and exposing data semantics. Therefore, an ontology is not an end unto itself, but rather supports the goal to expose data structure and semantics.

3.6.1.2 The Most Difficult Part of XML Schema Development is Determining the Major Information Objects

The XML Schema vocabulary relies upon an object-oriented conceptual framework. Decomposing the target domain and determining the major objects was the most difficult part of XML Schema design. Defining the major objects requires not only a clear understanding of how data can be logically grouped (i.e., what seems natural), but also how the data is expected to be used operationally. Object representation and content will vary depending upon the domain perspective. For example, one's view of a satellite will differ depending upon whether one has a space surveillance, AFSCN, or satellite product user perspective. Selection of the major information objects was less difficult for us because we used a well-defined subset of the space domain (i.e., space surveillance is well understood and content definition is stable), and we had strong domain knowledge. Major object determination would be much harder if the data content or usage were not well understood.

3.6.2 Lessons Learned About XML Technology

3.6.2.1 XML is Not About Terseness

XML is an open standard for understanding the structure and content of data. Adding XML tags to data adds volume, but the XML tagged data becomes easier to understand. Therefore, XML is not about terseness, but rather about labeling data to make it comprehensible. Although there is a tradeoff between understandability and bandwidth consumption, in most cases, this is a price well worth paying. Add to the equation two additional considerations: 1) there are many inexpensive (or even free) XML tools available with more emerging daily; and 2) because XML is being used extensively, commercial companies are addressing areas such as compression and security/encryption. This makes it easy to tap a huge commercial market, strengthening the benefits relative to the costs.

3.6.2.2 Our Ontology Approach is "Pushing the Technology Envelope"

Because we were working with an emerging technology, XML Schemas, there were few examples available to guide our efforts. This is not surprising because the specification, while stable, has not yet been approved. None of the examples we found addressed the question of how to use XML to capture sufficient semantic information to develop an ontology; therefore, we invented our own approach.

Also, because the XML Schema specification is still draft, XML Schema development tools were immature. This meant added complexity because of error prone tools. For example, XML Authority deleted code for features not yet implemented. Also, the XSL tool did not recognize the namespace header in the XML documents; we had to delete it to make the tool execute correctly. Nevertheless, the tools, while immature, promise great power and ease of use in the future.

When working on the leading edge of an emerging technology, change happens quickly. For example, a new XML Schema specification working draft was published while we were developing the Space Surveillance Ontology.

3.6.2.3 XML Schemas are Superior to DTDs

In the course of this task, we learned about developing DTDs and XML Schemas. We also saw several examples of DTDs. While DTDs are in wide use today, we believe XML Schemas are supplanting them. Further, we expect that once the XML Schema specification is approved, XML schemas will be embraced even more strongly. XML schemas are powerful tools. They support strong data typing, use XML syntax, allow element and type inheritance, and support the use of namespaces. All these factors combine to teach us the lesson that XML schemas are superior to DTDs.

3.6.2.4 Schema Development is Relatively Easy with Strong Domain Knowledge Available

This lesson is predicated upon the assumption that the XML schema development team has some knowledge or experience with XML schemas. For us this meant one day of hands-on training during our week-long XML training course and some experience with object-oriented programming. The hands-on training we received gave us the foundation needed to develop our space surveillance schema. If we do not count the time it took to get the validator working or the final review of annotations, it took us only about two staff weeks to develop the space surveillance schema.

3.6.2.5 Reuse is Powerful and Easy to Do with XML Schemas

With XML schemas, it is easy to define global elements or global types, or to declare global attributes. If these schema components are defined globally, then they can be reused either within the same namespace or in other namespaces. Reuse saves time and fosters standardization. Our Space Surveillance Ontology has several examples of reuse; the *classification* attribute group, *SatelliteNumber* element, and *dateTimeType* provide a reuse example for each category of schema component.

3.6.2.6 XML Schema Definition Language is Flexible

The XML Schema vocabulary allows several approaches to accomplish an objective. For example, one might capture element data as a child element or an element attribute. We found that the "best" approach is often dependent upon how the schema will be used. How to handle element classification marking provides an illustration. The XML Schema definition language is flexible enough to allow several approaches for how to handle classification marking of data items. We believe the design decision on which approach to use should be founded upon a determination of what best supports the overall security architecture.

3.6.2.7 Incremental Integration Eased Development

Our approach of creating a space surveillance data container made it easy to create an element, add it to the container, and debug both the associated schema and instance document. Once the major element was error-free, we could easily add another major element by including that element's schema, adding the element to the space surveillance data container, and including that element in the instance file. We found this incremental approach made schema development easier.

3.6.2.8 Use of Optional Elements Can Be Powerful

One feature of the XML Schema definition language (and DTDs) is the ability to declare elements to be optional. This means that a data instance document that conforms to the XML schema need not include instances of the optional element(s). This feature allows both a flexible migration path and an ability to accommodate different user needs. An example migration path could be to add new data as optional elements until all data consumers can handle this new data. (An alternative approach would be to use XSL to extract the data of interest and send only that subset to the consumer.) Optional elements also allow the same XML schema to be used to send vastly different sets of data that can be dependent upon data consumer needs. For example, our space surveillance schema has optional elements in the *SensorTask* element so that input parameters for the tasking software and tasking metrics need not be sent to a sensor. This accommodates an expectation that operational sensors would not want to receive this data with every set of satellite tasking they receive.

3.6.2.9 It is Relatively Easy to Transform XML-Tagged Data

Given a well-formed XML document, it is relatively easy to transform the data content into many different presentation formats. Our proof-of-concept transformations, described in Section 2.9 and Appendix E, provide that lesson. One transformation allows capturing XML-tagged space surveillance definitions to create an html formatted document with a stylesheet containing only approximately 25 lines of code (i.e., XSL instructions). Our sample instance document transformation stylesheet uses only approximately 40 lines of XSL instructions. These transformations can operate on instances that contain few or many elements. Although the runtime will change, the XSL transformation stylesheet remains the same.

Section 4

Conclusions

This section includes a synopsis of our most important conclusions and is followed by details on several conclusions we reached in the execution of this task. Finally, we provide a summary of next steps to include what MITRE will do and what we hope will happen as a result of this task.

4.1 Synopsis

The major conclusions we reached can be distilled into four areas:

1. This work reinforced our belief that it is vital to capture data semantics to achieve data interoperability as envisioned by Department of Defense (DoD) and USAF vision documents. This provides clarity within a domain and facilitates interoperability with other domains. We called the combination of captured data syntax and data semantics an ontology.
2. XML is a technology of choice for the military space domain, but it is only one aspect of a total solution. XML is an open-source standard for exchanging data that enjoys tremendous commercial support. The huge commercial investment makes XML strong due to economic forces. There are many inexpensive and free tools, and more are available every day. However, even with a full family of XML technology standards available, XML only addresses data structure, presentation, and exchange. It does not address many topics, such as a security approach, how the data is to be stored, or how the data is to be used operationally. Also, a strong data exchange standard does not negate the need to address data semantics. Therefore, while XML is a powerful technology to be considered in the development of future military space architectures, it is only one part of the technology tool chest.
3. XML Schema definition language is the XML vocabulary of choice. To define the structure and content of space data, XML Schema definition language is preferred over the use of DTDs. DTDs lack robustness for data definition and consistency checking. Also, we defined an approach that captures data semantics with the XML Schema vocabulary. The use of XML Schema definition language allows the same XML toolset to be used on both XML instance documents and XML schema documents. This simplifies implementation, thereby easing maintenance and decreasing costs.

4. The Space Surveillance Ontology results should be shared, at least across the government community. There is government community interest in having this space schema registered to expose the definitions to others. Also, others are interested in how we have captured our data semantics using XML Schema definition language. MITRE has already been asked by DISA representatives and the Object Management Group (OMG) Space Domain Task Force to share information on our approach.

4.2 Capturing Semantics is Vital

Understanding the meaning of data, not just capturing its structure and content, is vital to data interoperability. ESC/DIV defines data interoperability as "the ability to correctly interpret data that crosses system or organizational boundaries" [Renner]. This means that data must not only be exchanged, it must also be properly understood before one can achieve interoperability. The hard part is understanding the data so that humans and automated systems do the right thing. Therefore, capturing data structure and content, whether it be in an XML schema or DTD, is insufficient. One must also be able to understand the meaning of that data. Thus, an ontology is vital.

An ontology captures the semantics (i.e., data meaning) needed to allow data to be properly interpreted. One benefit of an ontology is that it captures semantics not readily available. This semantic information is often intrinsic knowledge captured in the minds of domain experts or documented in separate reports. For example, the space surveillance domain captures interface specifications in reports called Interface Control Documents (ICDs). Having this data recorded separately presents a significant challenge in keeping the implementation and documentation consistent. Appropriate development of an ontology, using XML Schema definition language, can eliminate or diminish this challenge. Also, an XML approach to ontologies means that arcane conventions necessitated by standard length messages can be eliminated. For example, in the satellite elset message, the rightmost digit of the range field is not part of the value. Rather, it describes the position of the decimal point. This convention is not intuitive, and means that one digit of precision is lost because only a fixed number of digits are allowed.

4.3 XML Schema Vocabulary is the Preferred Approach

We found XML Schema definition language to be the preferred approach for not only capturing data structure and content, but for capturing data semantics as well. As mentioned in Section 2.4, our decision to use XML Schema definition language over DTDs is well supported in the literature. Also, we believe the approach we developed for capturing an XML ontology using only XML Schema definition language is reasonable. With this approach data semantics can be captured and exposed in four ways: 1) mnemonic XML tag names give some insight into data meaning, 2) the element parent/child relationships provide further clues, 3) attributes can be used to describe metadata (e.g., units of measure), and

4) annotations can capture definitions and other elaborating information. Annotations are not unlike software comments; however, annotations can be easily accessed and exposed to users. Use of these four methods together capture the information we believe constitutes the data semantics we refer to as an ontology.

With this approach, the same XML toolset can be used to operate on the data (i.e., instance documents) as on the schema itself since both are well-formed XML documents (i.e., conform to XML syntax). Using the same set of tools simplifies maintenance, which lowers maintenance costs. We have started sharing our approach with the wider XML community and, thus far, we have received only positive feedback.

Also, commercial tools are available to help in XML Schema development. These tools, while immature, promise great power in the near future.

4.4 XML Schemas are Relatively Easy to Build

We found that XML schemas are relatively easy to create, provided the development team has a basic knowledge of the XML Schema vocabulary and strong domain knowledge. Our schema knowledge came from some experience in object-oriented programming and one day of hands-on training as part of a week-long XML course. With this foundation, and a subject matter expert, it took us approximately two weeks to develop the space surveillance schema. This includes our learning curve on the XML Schema vocabulary. Even if we include the complications caused by immature tools and the time spent reviewing annotations for accuracy, our development time is still no more than three weeks. Our resulting schema includes five key objects comprised of:

- 93 elements
- 39 user-defined types
- 1 attribute group used 16 times
- 34 attributes

Our conclusion is that using XML entails a relatively small investment that could pay strong dividends for facilitating data interoperability either within the space domain, or between space and other domains. This interdomain interoperability is what is envisioned in JV2020 and the JBI.

4.5 Namespaces are a Powerful Enabler of Standard Data Definitions

Namespaces are a powerful tool in the move toward standard data definitions. Namespaces can be used to develop XML schemas in well-defined subdomains. This allows multiple domains to examine the same object from different perspectives without the risk of naming collisions. For example, one can examine a satellite from many perspectives. Space surveillance is concerned with where an object is in space and keeping an accurate

characterization of its location. The AFSCN cares about sending commands to a satellite for station keeping and querying the health and status of onboard components. Theater commanders care about satellites from the perspective of the products they produce and how quickly those products can be obtained. One could develop an XML schema in each of these domains, with each containing an element called "Satellite." Since each "Satellite" element is in a different namespace, there would be no naming conflict.

Thus, using namespaces allows one to avoid ambiguity and focus on characterizing elements in a way most appropriate in one's own domain. In this way, one can make progress in one's own subdomain without waiting for a standard definition to be agreed upon across all relevant subdomains. However, if these XML schemas are registered and exposed to others, this generates the opportunity for reuse and creates the potential for data standardization. For instance, in our satellite example, if each subdomain exposed their data definitions (both syntax and semantics), appropriate domain representatives could examine the various satellite elements to determine whether it makes sense to develop a standard satellite element definition. If a standard satellite element was defined, it could then be inherited in each sub-domain and either be extended or restricted as needed.

4.6 XML is Being Embraced by the Commercial Market

XML enjoys widespread support in the commercial world. This is evidenced by the support of XML by:

- The World Wide Web Consortium (W3C), http://www.w3.org/,
- Nonprofit interoperability consortiums like the Organization for the Advancement of Structured Information Standards (OASIS), http://www.oasis-open.org/news/registryrelease.htm,
- Industry leaders, such as Microsoft, Sun Microsystems, International Business Machines (IBM), AT&T, and Oracle,
- XML supported COTS products, such as IE5, Oracle 8i, and Microsoft Office 2000, and
- The family of XML technology standards.

XML is being embraced, not because the technology upon which it is founded is new; rather, it is XML's penetration across the internet that makes it powerful. Also, because XML is founded upon open standards, there are many inexpensive (and free) commercial tools available that support XML. For example, there are already tools like Oracle's Extensible Structured Query Language (XSQL) that can be used to extract data from an Oracle database and populate instance documents with properly tagged XML data. These tools, combined

with the strong data typing available in XML schemas, enable data errors to be found at the time of instance creation. This reduces the number of potential errors introduced, thereby decreasing costs.

The XML schema development tools currently available are immature. This is to be expected since the XML Schema specification has not been approved. However, our experience with just one of these tools leads us to anticipate that powerful tools will soon be available that make schema development even easier.

4.7 XML Could be a Powerful Tool for System Migration

Employing XML could be a powerful tool for system migration. XML not only describes data content, it also adds a layer of data transparency between the data and how it is stored. XML tagged data is built upon an open standard with many tools available to assist in structuring, presenting, and exchanging the data. However, the details on how that data is stored, be it in a relational database, an object-oriented database, or a flat file, is transparent to the user. This, in essence, makes data portable. Without this transparency, a user would need to know how the data is stored to know what kind of tool to use to extract the content they desire. Once data is XML tagged, tools based upon open standards can be used to extract the desired data.

Further, changes to the data structure may have little to no impact on the end user. For example, once XML tagged data is produced, a consumer can use XSL to extract the content they desire. If new fields are added to the XML schema, consumers may be able to ignore the new content. If this is not possible, it is usually a simple matter to change the XML tool used to extract the content of interest (e.g., XSL stylesheet or Application Program Interface (API)). One could also make new elements optional so that instance documents could be created and sent that do not include the new elements.

These approaches are vastly different from today's approach. Currently, in the space surveillance domain, a change to data content means a change to a standard message format. Because these messages are used at many locations, even a simple change means a vertical release with implementation of the change occurring at the same time at all the affected locations. Therefore, even simple changes require a long lead time. An approach based on an open commercial standard could change this.

4.8 There Was Value in MITRE Performing This Task

An unsurprising conclusion we reached is that there was value in MITRE performing this task. Execution of this task allowed us to build an understanding of this emerging technology and how and where it could benefit our sponsor. We have quickly built an understanding of the family of XML technology standards and tools, know which are mature, and realize where significant risks remain. Armed with this knowledge we can help achieve synergy between different ESC/ND System Program Office contracts and between the

military space domain and other domains. These other domains could include other Air Force efforts using XML (e.g., JBI) and XML efforts in other services (e.g., Joint Intelligence Virtual Architecture Knowledge Management effort).

4.9 Next Steps

We have divided next steps into two categories: 1) what MITRE will do next, and 2) what we hope will happen next.

4.9.1 What MITRE Will Do Next

MITRE currently has plans to do three things:

1. Meet with the newly awarded Integrated Space Command and Control (ISC2) contractor who will be developing the future architecture for military space systems. We plan to share the results of this effort with this contractor.
2. Establish a collaborative approach with this contractor for reviewing, registering, and extending the Space Surveillance Ontology, and discussing how XML could be used in the modernization of the military space C2 architecture. We would like this effort to not only form the basis for collaboration within the SPO, but also serve as a catalyst for further collaborations within the USAF and across the DoD.
3. Work with the Aerospace Operations namespace manager to understand and define how to populate a portion of this namespace, and hopefully become part of a team working with DISA to evolve the XML Registry, making its population and use easier.

4.9.2 What MITRE Hopes Will Happen

MITRE hopes this Space Surveillance Ontology task will lead to several results: 1) that the Space Surveillance Ontology will be registered in DISA's XML Registry (and possibly commercial schema registries as well) to expose the data definitions across the DoD and beyond; 2) that the Space Surveillance Ontology will be broadened to include other portions of space domain; 3) that the contractor can use this ontology to decrease costs for ESC/ND SPO development and migration efforts; and 4) that the Space Surveillance Ontology can facilitate having military space become an even more integral part of a future information architecture like that used in the JBI.

List of References

Benkley	Benkley, C. W., *An XML Framework for Development of a Scheduler Product Line (SPL),* MITRE White Paper, August 2000.
Guarino	Guarino, N., and Giaretta, P., *Ontologies and Knowledge Bases: Towards a Terminological Clarification,* Towards Very Large Knowledge Bases, IOS Press 1995. Available online at: http://www.ladseb.pd.cnr.it/infor/ontology/Papers/OntologyPapers.html
Gruber	Gruber, T, *What is an Ontology?* Available online at: http://www.ontology.org/main/papers/faq.html
JBI	Air Force Scientific Advisory Board Report on Building the Joint Battlespace Infosphere, SAB-TR-99-02, December 1999.
JV2020	*Joint Vision 2020*, June 2000. Available online at: http://www.dtic.mil/jv2020/jvpub2.htm
Mikula	Mikula, N. and Levy, K., *Schemas Take DTDs to the Next Step.l*Available online at: http://www.xmlmag.com/upload/free/features/xml/1999/01win99/nmwin99/nmwin99.asp
W3CNamespace	*Namespaces in XML*, World Wide Web Consortium, 14 Jan 1999. Available online at: http://www.w3.org/TR/1999/REC-xml-names-19990114/#sec-intro
Pulvermacher	Pulvermacher, M., *Preliminary Space Surveillance Ontology Contextual Framework,* delivered to Ms. Roxanne Groenewoud (ESC/NDC) and Mr. Mike Findley (ESC/NDW) via electronic mail on 30 August 2000.
Renner	*Data Issues,* a briefing by Dr. Scott Renner of the DII-AF Common Data Environment (CDE) Product Area Directorate (PAD) to the Air Force Architecture Workshop, sponsored by the Air Force Communications and Information Center (AFCIC), 18 November 1999.
Schema0	*XML Schema Part 0: Primer*, W3C Working Draft, 22 September 2000. Available online at: http://www.w3.org/XML/Schema
Schema1	*XML Schema Part 1: Structures*, W3C Working Draft, 22 September 2000. Available online at: http://www.w3.org/XML/Schema
Schema2	*XML Schema Part 2: Datatypes*, W3C Working Draft, 22 September 2000. Available online at: http://www.w3.org/XML/Schema

Uschold	Uschold, M. and Gruninger, M., *Ontologies: Principals, Methods and Applications,* Artificial Intelligence Applications Institute, AIAI-TR-191, February 1996.
URI	*Naming and Addressing: URIs, URLs, ...,* W3C Architecture Domain, Available online at: http://www.w3.org/Addressing/

Appendix A

XML Background

The Extensible Markup Language (XML) is an international industry standard for describing and sharing structured information. XML is receiving enormous support from the commercial community and is considered one of the leading information technologies. The primary reason XML is receiving so much support is because of its key role in eCommerce. XML is making information exchange easier on the web and giving users more flexibility on information exchange.

XML has taken a new approach to information. XML, like Hypertext Markup Language (HTML), is a markup language. But unlike the fixed HTML tags that describe how to display data, XML tags are extensible and describe the meaning of the data. Coupling the meaning of the data with the data facilitates improved data search, data sharing, and information interoperability.

As shown in Figure A-1, XML is a family of technologies that provide the basis for information exchange. Each of these technologies is briefly described below:

XML Schema – provides facilities for describing the organization and information structure of associated XML documents that are based on the schema. Capabilities include validation of structure and content, as well as constraint checking.

XSL – Extensible Style Sheet Language – provides a language for presenting the content of an XML document to a user or application. Capabilities including selecting, filtering, reordering, manipulation, and presentation of the information contained in XML document or documents.

XQL – Extensible Query Language – provides a general capability to query XML documents. The current XQL specification is under consideration by the World Wide Web Consortium (W3C).

DOM and SAX – Documented Object Model and Simple API for XML – each provide a standardized application program interface (API) for accessing data in an XML document. These APIs are used to access and navigate XML documents.

RDF – Resource Description Format – provides a generic way of describing resources (collections of content) in the XML vocabulary. The purpose of RDF is to describe metadata (i.e., data about the data) of resources on the world wide web (www). The resources must be associated with a www address (i.e., a URL). RDF could be a powerful tool to assist search engines. For example, RDF could be used to associate a collection of information about a Mustang to information about automobiles, rather than information about the particular variety of horse.

Xlink – provides generalized hypertext linking capabilities from within XML documents.

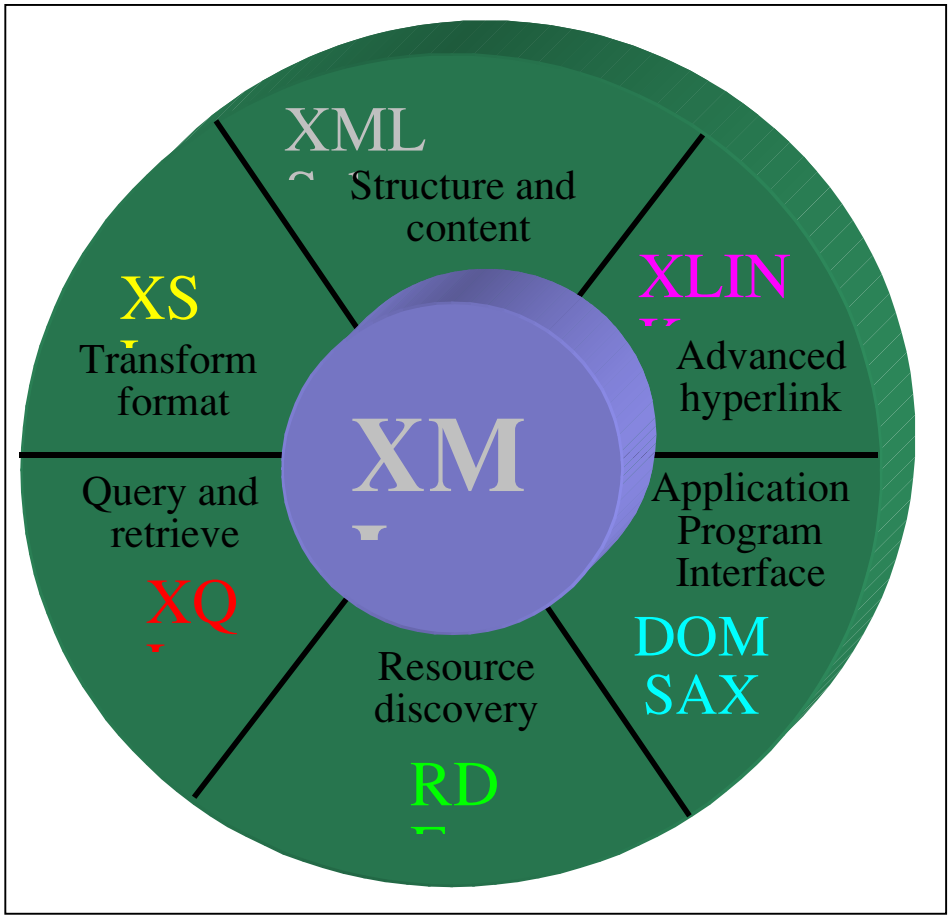


Figure A-1. XML Family of Technologies

For more information see: www.oasis-open.org or www.w3c.org or www.xml.org.

Appendix B

Related Efforts

This appendix contains a brief summary of several Extensible Markup Language (XML) efforts we found relevant to our Space Surveillance Ontology initiative. We divided these efforts into three categories: 1) Related XML Efforts, 2) XML Schema Repositories, 3) and Related MITRE Efforts. For each effort we provide a short description of the effort, an explanation of its relevance to the Space Surveillance Ontology, and where to go for more information on the effort.

B.1 Related XML Efforts

B.1.1 Spacecraft Markup Language

Description: Spacecraft Markup Language (SML) is an extension of XML providing the Space Community with a standard definition of XML tags and concepts of structure to allow the definition of spacecraft and other support data objects (e.g., packets, commands, messages). The SML reserved tags are defined within the SML Draft Specification document, see http://www.interfacecontrol.com/sml/Sml/mainpage.htm.

Relevance: We reviewed the SML Draft Specification and found little direct relevance to our Space Surveillance Ontology. The SML fits within the Astronomy and Space domain, but does not address specific data types required by the space surveillance subdomain. The Interface & Control System, Inc. developed document does provide a good example of a draft specification.

For More Information: See the Interface & Control System, Inc., web site at http://www.interfacecontrol.com/.

B.1.2 Spacecraft XML for TT&C

Description: The CEnter for REsearch Support (CERES) at Schriever Air Force Base in Colorado is investigating the use of XML to support spacecraft ground system database interchange. CERES supported the development of a set of Document Type Definitions (DTDs) to capture information related to spacecraft telemetry, tracking, and commanding (TT&C) operations. The goal is to standardize the structure and content of TT&C database file contents.

Relevance: This TT&C XML effort could become part of a standard space XML repository. The Defense Information Systems Agency (DISA) has developed the Defense Information Infrastructure Common Operating Environment (DII COE) XML Registry to facilitate

interoperability by exposing data definitions. This TT&C XML effort could become part of this DISA XML Registry. In fact, it could potentially be part of the same DISA namespace targeted by the Space Surveillance Ontology.

For More Information: Contact: Major Doug Howse,
Chief On Orbit Combined Test Force
SMC/TEOC - Center for Research Support (CERES)
(719) 567-6023
douglas.howse@afmc.schriever.af.mil

B.1.3 XML-MTF Initiative

Description: DISA, North Atlantic Treaty Organization (NATO), and the United States Air Force (USAF) Aerospace Command, Control, Intelligence, Surveillance, and Reconnaissance Center (AC2ISRC) have agreed to jointly work on an XML-Message Text Format (MTF) standard. The purpose of the XML-MTF initiative is to specify and standardize an XML representation for USAF and NATO MTFs, USMTF and Allied Data Publication-3, respectively. An international development team is accomplishing this work on behalf of the combined MTF community. MITRE, under the sponsorship of the AC2ISRC, is leading the development team. The XML-MTF Development Team has identified XML Schema as an important technology for supporting XML-MTF and is implementing XML Schemas to represent MTF metadata.

Relevance: XML-MTF is another pioneer effort in the application of XML technology to military information exchange. Approximately 6000 developmental XML-MTF elements have been submitted to the DII COE XML Registry. XML-MTF principals have also identified XML Schema development as an important step in specifying an XML vocabulary for the military.

For More Information: Contact XML-MTF Development Team leader, Mike Cokus,
msc@mitre.org, (757) 896-8553
or
Visit the USMTF Program XML-MTF web site at
http://www-usmtf.itsi.disa.mil/private/xml-mtf/xml-mtf.html.
This web site contains sensitive information and is password protected. Please follow the instructions on the associated home page to gain access.

B.1.4 Instrument Markup Language (IML)/Astronomical IML (AIML)

Description: IML is a vocabulary based on the W3C standard for XML. IML is under development by National Aeronautics and Space Administration Goddard Space Flight Center (NASA/GSFC) and Century Computing, a division of AppNet, Inc. The approach taken by the development team is to create a very general and highly extensible framework

that applies to virtually any kind of instrument that can be controlled by a computer. To this end, the AIML is the first implementation of what is to become the more general IML. The IML will be created based on experience gained by focusing on the astronomy domain (and infrared instruments in particular). NASA/GSFC believes this approach is feasible because the key aspects of the AIML approach to instrument description and control apply to many domains (medical instruments (e.g., microscopes), printing presses, machine assembly lines).

Relevance: This effort is not directly relevant, but is another example of attempting to develop an XML data standard for a particular domain.

For More Information: See http://pioneer.gsfc.nasa.gov/public/iml/.

B.1.5 The Astronomical Data Center: eXtensible Data Format (XDF)

Description: XDF is a common scientific data format based on general mathematical principles, object models, and XML that can be used throughout the scientific disciplines. It includes these key features: hierarchical data structures, any dimensional arrays merged with coordinate information, high dimensional tables merged with field information, variable resolution, easy wrapping of existing data, user specified coordinate systems, searchable ASCII metadata, and extensibility to new features.

Relevance: The XDF project demonstrates methods of handling and storing different data types; however, it is not directly relevant to the Space Surveillance Ontology because their data is stored using DTDs.

For More Information: See http://tarantella.gsfc.nasa.gov/xml/XDF_home.html

B.1.6 W3C Namespaces in XML

Description: W3C envisions applications of XML where a single XML document may contain elements and attributes that are defined for, and used by, multiple software modules. One motivation for this modularity is that it is better to reuse markup than reinvent it. This requires that document constructs have universal names, whose scope extends beyond their containing document. W3C has specified a mechanism, XML namespaces, to accomplish this [W3CNamespace].

Relevance: In our space surveillance schema, we defined a space surveillance namespace. Use of namespaces are needed to ensure software modules are able to recognize XML element tags and attributes without the threat of "collisions" occurring from using the same name. Namespaces enable disambiguation by extending the XML syntax to allow element names and attribute names to be qualified with a namespace; in our case, the space surveillance namespace.

For More Information: http://www.w3.org/

B.2 XML Schema Repositories

B.2.1 Commercial Schema Repositories

B.2.1.1 XML.ORG

Description: In June 2000, the Organization for the Advancement of Structured Information Standards (OASIS),[3] the nonprofit XML interoperability consortium, announced public access to the first phase of its XML.ORG Registry. Central to XML.ORG is an open registry and repository offering automated public access to XML specifications such as vocabularies, DTDs, schemas, and namespaces. Another portion of XML.ORG is the XML Catalog that lists organizations known to be producing industry-specific or cross-industry XML specifications. This catalog currently lists three entries under the Industry activity area of "Astronomy and Space." These are:

- Spacecraft Markup Language (SML)
- NASA –AIML
- NASA – The Astronomical Data Center: XDF

Relevance: This is a potential XML schema repository for the Space Surveillance Ontology.

For More Information: See http://www.xml.org/

B.2.1.2 BizTalk

Description: BizTalk™ is an industry initiative started by Microsoft and supported by a wide range of organizations, from technology vendors like SAP Aktiemgesell (SAP AG) and CommerceOne to technology users like Ariba and Business and Accounting Software Developers Association (BASDA). BizTalk is not a standards body. Instead, they are a community of standards users, with the goal of driving the rapid, consistent adoption of XML to enable electronic commerce and application integration. BizTalk is using the BizTalk Framework, a set of guidelines for how to publish schemas in XML and how to use XML messages to easily integrate software programs together in order to build rich, new solutions.

Relevance: This is a potential XML schema repository for the Space Surveillance Ontology.

---

[3] See "XML.ORG Goes Live with First Phase of Open Registry & Repository for XML Specifications" available online at: http://www.oasis-open.org/news/registryrelease.htm

For More Information: See http://www.biztalk.org

B.2.2 U. S. Government Schema Repository

B.2.2.1 DII COE XML Registry

Description: DISA has developed an XML registry called the DII COE XML Registry. The DISA DII COE XML Registry is intended to help ensure interoperability and provides a baseline set of XML tags developed through coordination and approval among the COE community. DISA is organizing XML tags in Namespaces. A namespace may be thought of as a community of interest. Figure B-1 quotes DISA's definition of a Namespace from their web page.[4]

> **What is a Namespace?**
>
> A Namespace is a collection of people, agencies, activities, and system builders who share an interest in a particular problem domain or practical application. This implies a common world view, as well as common abstractions, common data representations, and common metadata. The COE Data Emporium, including the XML Registry, allows Namespaces to publish their existence and their available information resources so that outsiders may discover them and assess whether or not they want to share.

Figure B-1. DISA Namespace Definition

The XML Registry *exposes* your data definitions to other people. It does not mean that your definitions must *match* theirs. The DII COE XML Registry has defined the 11 namespaces listed below:

1. AOP – Aerospace Operations
2. COE – COE Enterprise
3. CSS – Combat Support
4. GEO – Geospatial and Imagery
5. GMI – General Military Intelligence
6. GOP – Ground Operations
7. MET – Meteorological and Oceanographic

---

[4] See http://diides.ncr.disa.mil/xmlreg/namespace_list.cfm for more information.

8. MSG – Messages
9. TAR – Tracks and Reports
10. FIN – Finance and Accounting
11. PER – Personnel

Relevance: The preliminary Space Surveillance Ontology linkage to the DII COE XML Registry is under the Aerospace Operations namespace. The Aerospace Operations namespace manager is Mr. Ray Spinosa (ESC/DIVD).[5] Our goal is to work with ESC/ND contractors to submit the space surveillance schema to Mr. Spinosa for inclusion in the Aerospace Operations Namespace.

For More Information: See http://diides.ncr.disa.mil/xmlreg/index.cfm.

B.3 Related MITRE Efforts

B.3.1 GMTI Virtual Warehouse & Joint Exploitation Tools

Description: This is a MITRE Mission-Oriented Investigation and Experimentation (MOIE) research effort. The purpose of the effort is to enable greater data interoperability between Ground Moving Target Indicator (GMTI) producers and consumers, enable sharing of tools that exploit GMTI data, and advance the concepts of the Integrated Command and Control System (IC2S) and the Battlespace Infosphere. This effort addresses the problem of how to provide access to the ever expanding amount of GMTI information available from current and emerging sensors, such as the Joint Surveillance & Target Attack Radar System (JSTARS), various Unmanned Aerial Vehicles (UAVs), and coalition forces. This research is addressing the issues of how to fuse similar information from multiple sensors into a cohesive set of data stored in a virtual warehouse.

Relevance: Because space command and control (C2) is part of the IC2S, space surveillance could become another producer of data for the Battlespace Infosphere. Space data consumers could conceivably use this approach to access relevant data.

For More Information: Contact Mr. Robert Case, rcase@mitre.org

B.3.2 Fuselet Architecture for Adaptive Information Structures

Description: This was another MITRE MOIE research project. This effort developed a proof-of-concept that showed one can semiautomatically detect different types of mappings

---

[5] This is the Air Force Defense Information Infrastructure (DII-AF) Common Data Environment (CDE) Office, a part of the Information Management Product Area Directorate (PAD).

between two different XML files. The effort also demonstrated that one can document the mappings using XLink, which would enable any application to read the mappings and use them as required.

Relevance: If completed, the results of this research could be used to define mappings between XML tagged space data files, and quickly adapt these mappings as changes occur in the data structures.

For More Information: Contact Mr. Michael W. Ripley, (781) 271-4998, rip@mitre.org

B.3.3 An XML Framework for Development of a Scheduler Product Line

Description: Under the sponsorship of ESC, MITRE developed a scheduling taxonomy that is capable of representing any schedule, including constituent tasks and resources, from any scheduling domain. This scheduling taxonomy was implemented in the form of an XML schema, in accordance with the XML schema language. MITRE showed that this scheduling taxonomy can be used to partition the scheduling infospace to facilitate development of a scheduler product line [Benkley].

Relevance: Although this effort is outside the Aerospace Operations namespace, it provides a great example of an XML schema that is relevant in the government domain. It may also become the foundation for a standard scheduling data markup approach.

For More Information: Contact Mr. Carl W. Benkley, **(**781) 271-8687, cbenkley@mitre.org

Appendix C

DII COE XML Registry Registration Process

We assume one of our sponsor's goals will be to have the Space Surveillance Ontology registered in the Defense Information Systems Agency (DISA) Defense Information Infrastructure Common Operating Environment (DII COE) Extensible Markup Language (XML) Registry. This appendix summarizes the process for registering what DISA calls an Information Resource (IR) and provides some comments on future directions.

C.1 DISA Registration Process

The DISA Registry is intended to be the authoritative source for COE developers for approved XML data and metadata components. To register what DISA calls IRs, government system developers develop a "submittal package." An IR is a component or product of an information system that can be defined, scoped, and managed for reuse. The IR can be either physical (e.g., file, database column, database record, query) or conceptual (e.g., entity, attribute, content model). IRs can be documents, XML schema(s), or a set of element and attribute definitions. Below is an outline of the steps for the submitter.

1. Go to the DISA XML Registry page at: http://diides.ncr.disa.mil/xmlreg/index.cfm
2. Register as an XML Registry user
3. Review the XML Registry IRs to determine if existing tags meet your needs
4. Review namespaces on the Registry and coordinate with namespace manager(s) to determine appropriate namespace
5. Build a submission package
   (a) Examples and instructions available online
   (b) Result is a valid XML document conforming to the DTD established by DISA for a registry submission
6. Validate and submit package using online submission form

C.2 Future Directions

Most IRs in DISA's registry are Document Type Definition (DTD)-based, and the underlying model for the registry is a relational database. XML Schemas, on the other hand, use a different vocabulary and have an object-oriented conceptual model. Currently, one can register an XML schema, but it is unclear how well the XML Registry accommodates some XML Schema features, such as type and element inheritance, importing from other namespaces and attribute groups. MITRE has brought these concerns to DISA's attention

who will consider them as they continue to evolve the registry. Also, the space surveillance schema assumes an ability to define subdomains within a DISA namespace. MITRE is working with the Aerospace Operations namespace manager to determine if the namespace can accommodate subdomains.

Appendix D

Space Surveillance Schema

This appendix contains the actual files that constitute the Space Surveillance Ontology. The files were created with the April 2000 draft version of the XML Schema specification. Although a newer version of the specification was published during our development effort, we used this version because the validator we used was compliant with the April 2000 specification working draft. Each file is contained in a separate subsection. The files included in this appendix, along with a brief description of each, is listed below.

1. SpaceSurveillance.xsd: Schema for space surveillance data sets. This is a container element for the space surveillance elements.
2. SpaceSurvGlobals.xsd: Schema file that contains global attribute declarations and type definitions for the space surveillance namespace.
3. Satellite.xsd: Schema for a satellite from a space surveillance perspective.
4. LaunchSiteDefintions.doc: Microsoft Word file that contains satellite launch site short names and the associated satellite launch site long name. This file is referenced in the documentation annotation for the *LaunchSite* element in the satellite schema.
5. Sensor.xsd: Schema for a sensor from a space surveillance perspective. Presumes a ground-based sensor.
6. SensorTasking.xsd: Schema that describes the priority and amount of data a sensor is to collect on each tasked satellite.
7. SatObservation.xsd: Schema for a satellite observation by a Space Surveillance Network sensor.
8. SatElset.xsd: Schema for satellite general perturbations orbital element set.
9. ss.xml: Instance document that contains instances of the *SpaceSurveillanceData* elements: *Satellite, Sensor, SatObservation,* and *SatelliteElementSet.*
10. tasking.xml: Instance document that contains an instance of the *SensorTask* element.

D.1 SpaceSurveillance.xsd

```
<?xml version="1.0"?>
<schema xmlns="http://www.w3.org/1999/XMLSchema"
      targetNamespace="urn:schemas-gov:SpaceSurveillance"
      elementFormDefault="qualified"
```

```
   xmlns:ss="urn:schemas-gov:SpaceSurveillance"
   xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
   xsi:schemaLocation=
               "http://www.w3.org/1999/XMLSchema
                http://www.w3.org/1999/XMLSchema.xsd"
      version="1.0">

<!--Schema for Space Surveillance Sets
      **************
      *** File name:  SpaceSurveillance.xsd ***
      **************
      *** As of:       October 6, 2000
      **************
      -->

<!--Conventions used:
               Element names begin with a capital letter
               Type and Attribute names begin with lowercase
               Make elements as atomic as possible
               Bias toward self-describing names over short, bandwidth conserving ones
               Collect global element/attribute declarations and global type definitions
                 in a separate subschema
               Include a documentation annotation in each element or attribute declaration
                 or type definition to define terms and provide explanatory information.
       -->

      <!-- include Space Surveillance Namespace global type and element definitions -->
      <include schemaLocation = "SpaceSurvGlobals.xsd"/>

      <!--Include Space Surveillance Element Files -->
      <include schemaLocation="Satellite.xsd"/>
      <include schemaLocation="Sensor.xsd"/>
      <include schemaLocation="SensorTasking.xsd"/>
      <include schemaLocation="SatObservation.xsd"/>
      <include schemaLocation="SatElset.xsd"/>

      <element name = "SpaceSurveillanceData">
            <complexType>
                  <annotation>
                        <documentation>
                              A container for space surveillance elements.
```

```
                    Major object relationships.
                    - Satellite and SatelliteElementSet data are used
                      to generate sensor tasking.
                    - Sensors receive SatelliteElementSets and
                      SensorTasks and collect SatObservations
                    - SensorTasks describe to the Sensor the priority and
                      amount of data to collect on each Satellite.
                    - SatObservations are used to update
                      SatelliteElementSets.
                    - SatelliteElementSets are sent to Sensors to locate
                      objects in space.
                    Uniqueness - Satellite Element:
                        Specified so that the satelliteNumber
                        attribute must contain a unique value across
                        all instances of the Satellite element that
                        appear as children of a particular space
                        surveillance data element.
                    Uniqueness - Sensor:
                        Specified so that the value of SensorNumber
                        must be unique across all the instances of
                        the Sensor element that appear as children
                        of a particular space surveillance data
                        element.
                </documentation>
        </annotation>
        <sequence>
                <element ref = "ss:Satellite"
                        minOccurs = "0" maxOccurs = "unbounded"/>
                <element ref = "ss:Sensor"
                        minOccurs= "0" maxOccurs = "unbounded"/>
                <element ref = "ss:SensorTask"
                        minOccurs= "0" maxOccurs = "unbounded"/>
                <element ref = "ss:SatObservation"
                        minOccurs= "0" maxOccurs = "unbounded"/>
                <element ref = "ss:SatelliteElementSet"
                        minOccurs= "0" maxOccurs = "unbounded"/>
        </sequence>
</complexType>
<unique name="satelliteUnique">
        <selector>SSData/Satellite</selector>
        <field>@satelliteNumber</field>
```

```
            </unique>
            <unique name="sensorUnique">
                <selector>SSData/Sensor</selector>
                <field>SensorNumber</field>
            </unique>
        </element>
</schema>
```

D.2 SpaceSurvGlobals.xsd

```
<?xml version="1.0"?>
<schema xmlns="http://www.w3.org/1999/XMLSchema"
      targetNamespace="urn:schemas-gov:SpaceSurveillance"
      xmlns:ss="urn:schemas-gov:SpaceSurveillance"
        elementFormDefault="qualified"
        attributeFormDefault="qualified"
        version="1.0">

        <!-- Space Surveillance Namespace Global
                Attribute Declarations and Type Definitions
        **************
        *** File name:  SpaceSurvGlobals.xsd ***
        **************
        *** As of:      October 6, 2000
        **************
        -->

        <complexType name = "dateTimeType">
                <annotation>
                        <documentation>Definition of a Date and Time element group.
                                        Format of Date is: CCYY-MM-DD
                                        Format of Time is: hh:mm:ss.sss
                        </documentation>
                </annotation>
                <element name = "Date" type="date"/>
                <element name = "Time" type="time"/>
        </complexType>

        <attributeGroup name = "classification">
                <attribute name = "classLevel" type = "ss:classificationLevelType"
                                        use="required"/>
```

```
        <attribute name = "dissemination" type = "ss:disseminationType"
                                use="optional"/>
        <attribute name = "classificationGuide" type = "string"
                                use="optional"/>
</attributeGroup>

<simpleType name = "classificationLevelType" base = "string">
        <annotation>
                <documentation>Classification Level enumerated values.
                                Legal values are:
                                        "Unclassified" "U"
                                        "Secret" "S"
                                        "Secret Talent-Keyhole" "S//TK"
                                        "TopSecret" "TS"
                                        "TopSecret Talent-Keyhole" "TS//TK"
                                        "Confidential" "C"
                                        "Proprietary" "PROP"
                                        "For Official Use Only" "FOUO"
                </documentation>
        </annotation>
        <enumeration value = "Unclassified"/>
        <enumeration value = "U"/>
        <enumeration value = "Secret"/>
        <enumeration value = "S"/>
        <enumeration value = "Secret Talent-Keyhole"/>
        <enumeration value = "S//TK"/>
        <enumeration value = "TopSecret"/>
        <enumeration value = "TS"/>
        <enumeration value = "TopSecret Talent-Keyhole"/>
        <enumeration value = "TS//TK"/>
        <enumeration value = "Confidential"/>
        <enumeration value = "C"/>
        <enumeration value = "Proprietary"/>
        <enumeration value = "PROP"/>
        <enumeration value = "For Official Use Only"/>
        <enumeration value = "FOUO"/>
</simpleType>

<simpleType name = "disseminationType" base = "string">
        <annotation>
                <documentation>Dissemination control marking used to limit the
```

```
                                distribution of classified information.  This is an
                                enumerated list.
                                Legal values are:
                                  "Releasable"
                                  "ORCON"  (Originator controlled)
                                  "RELCAN" (Releasable to Canada)
                                  "NOFORN" (Not releasable to Foreign Nationals)
                                  "PROPIN" (Caution-Proprietary Information Involved)
                        </documentation>
                </annotation>
                <enumeration value = "Releasable"/>
                <enumeration value = "ORCON"/>
                <enumeration value = "RELCAN"/>
                <enumeration value = "NOFORN"/>
                <enumeration value = "PROPIN"/>
        </simpleType>
</schema>
```

D.3 Satellite.xsd

```
<?xml version ="1.0"?>
<schema targetNamespace = "urn:schemas-gov:SpaceSurveillance"
        elementFormDefault="qualified"
      xmlns:ss="urn:schemas-gov:SpaceSurveillance"
        xmlns = "http://www.w3.org/1999/XMLSchema"
        version="1.0">

  <!--Schema for Satellite (from a Space Surveillance perspective)
        **************
        *** File name:  Satellite.xsd ***
        **************
        -->

        <element name = "Satellite">
                <complexType>
                        <annotation>
                                <documentation>Manmade space object.
                                </documentation>
                        </annotation>
                        <sequence>
                                <element ref = "ss:InternationalDesignator"/>
```

```
                    <element ref = "ss:CommonName"/>
                    <element ref = "ss:Owner"/>
                    <element ref = "ss:SatelliteObject"/>
                    <element ref = "ss:LaunchData"/>
                    <element ref = "ss:DecayDate"/>
                </sequence>
        <attribute name="satelliteNumber" type="ss:satelliteNumberType"
                                        use="required"/>
        </complexType>
</element>

<element name = "SatelliteNumber" type = "ss:satelliteNumberType">
        <annotation>
                <documentation>
                        Cataloged satellite numeric identifier assigned by
                        USSPACECOM.  </documentation>
        </annotation>
</element>

<!-- In September 2000 version XML Schema specification, the equivClass has been
        renamed to substitutionGroup -->

<element name="Satno" equivClass="ss:SatelliteNumber"
                                type="ss:satelliteNumberType">
        <annotation>
                <documentation>An equivalent name for SatelliteNumber.  The two
                        terms can be used interchangeably.  </documentation>
        </annotation>
</element>

<element name = "InternationalDesignator">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Internationally recognized designator for a
                                satellite.  Includes LaunchYear, LaunchNumber, and
                                PieceIdentifier.  Must be unclassified.
                        </documentation>
                </annotation>
                <sequence>
                        <element ref = "ss:LaunchYear"/>
                        <element ref = "ss:LaunchNumber"/>
```

```
                    <element name="PieceIdentifier"
                                type="ss:satellitePieceIdentifierType"/>
                </sequence>
                <attribute name="classLevel" use="fixed" value="U">
                    <annotation>
                        <documentation>This element MUST be unclassified
                            and so classLevel attribute MUST be set to "U"
                        </documentation>
                    </annotation>
                </attribute>
        </complexType>
</element>

<element name = "LaunchYear" type = "year">
        <annotation>
            <documentation>Four-digit year of launch</documentation>
        </annotation>
</element>

<element name = "LaunchNumber" type = "positiveInteger">
        <annotation>
            <documentation>Sequential launch number for that year.
            </documentation>
        </annotation>
</element>

<element name = "CommonName">
        <complexType type = "ss:satCommonNameType" content="textOnly">
            <annotation>
                <documentation>Common name for the satellite (e.g., USA
                    148, CSL-04 DEB, SL12 R/B(1)).  Name may be no
                    more than 24 characters long.  Must be unclassified.
                </documentation>
            </annotation>
            <attribute name="classLevel" use="fixed" value="U">
                <annotation>
                    <documentation>This element MUST be unclassified
                        and so classLevel attribute MUST be set to "U"
                    </documentation>
                </annotation>
            </attribute>
```

```
                <attribute name = "source" type = "string" use="optional">
                        <annotation>
                                <documentation>Source for the common name (e.g.,
                                        NASA, NRO, TASS, USSPACECOM).
                                </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<element name = "Owner">
        <complexType type = "ss:satOwnerType" content="textOnly">
                <annotation>
                        <documentation>Country or organization that owns
                                satellite.  Must be unclassified.
                        </documentation>
                </annotation>
                <attribute name="classLevel" use="fixed" value="U">
                        <annotation>
                                <documentation>This element MUST be unclassified
                                        and so classLevel attribute MUST be set to "U"
                                </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<element name = "DecayDate" type = "date">
        <annotation>
                <documentation>Estimated date of satellite decay.
                                Format is: CCYY-MM-DD
                </documentation>
        </annotation>
</element>

<element name = "LaunchData">
        <complexType>
                <annotation>
                        <documentation>Container for LaunchDate, LaunchTime,
                                LaunchSite, and LaunchingAgency.
                                        LaunchDate format is: CCYY-MM-DD
```

```
                                        LaunchTime format is: hh:mm:ss.sss
                                        LaunchSite and LaunchingAgency are strings
                                Must be unclassified.
                        </documentation>
                </annotation>
                <sequence>
                        <element name="LaunchDate" type="date"/>
                        <element name="LaunchTime" type="time"/>
                        <element name="LaunchSite"
                                                type="ss:satelliteLaunchSiteType"/>
                        <element ref= "ss:LaunchingAgency"/>
                </sequence>
                <attribute name="classLevel" use="fixed" value="U">
                        <annotation>
                                <documentation>This element MUST be unclassified
                                        and so classLevel attribute MUST be set to "U"
                                </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<element name = "LaunchingAgency" type = "string">
        <annotation>
                <documentation>Agency responsible for the satellite launch (e.g.,
                                USAF, ESA, NRO, NASA, Intelsat)
                </documentation>
        </annotation>
</element>

<element name = "SatelliteObject">
        <annotation>
                <documentation>Characterization of the manmade object.
                                Can be either a PayloadObject or a NonPayloadObject.
                </documentation>
        </annotation>
        <complexType content = "elementOnly">
                <choice>
                        <element ref = "ss:PayloadObject"/>
                        <element name="NonPayloadObject"
                                        type="ss:satelliteObjectType"/>
```

```
                </choice>
                <attributeGroup ref="ss:classification"/>
        </complexType>
</element>

<element name = "PayloadObject">
        <complexType>
                <annotation>
                        <documentation>A manmade payload.  A satellite payload has
                                a:  PayloadType (string), SatelliteStatus (must be
                                one of satellitePayloadStatusType's enumerated
                                types), and SatelliteMission (string)
                        </documentation>
                </annotation>
                <sequence>
                        <element ref = "ss:PayloadType"/>
                        <element ref = "ss:SatelliteStatus"/>
                        <element ref = "ss:SatelliteMission"/>
                </sequence>
        </complexType>
</element>

<element name = "PayloadType" type = "string">
        <annotation>
                <documentation>Type of payload (e.g., communications, navigation,
                        reconnaissance, weather, scientific).
                </documentation>
        </annotation>
</element>

<element name = "SatelliteStatus">
        <complexType type="satellitePayloadStatusType" content="textOnly">
                <annotation>
                        <documentation>Status of payload.
                                Must be one of satellitePayloadStatusType's
                                enumerated types. </documentation>
                </annotation>
                <attributeGroup ref="ss:classification"/>
                <attribute name = "source" type = "string" use="optional">
                        <annotation>
                                <documentation>Source for the payload status (e.g.,
```

```
                                        OpenSource, Owner/Operator, Intelligence).
                                </documentation>
                        </annotation>
                </attribute>
                <attribute name = "lastUpdated" type = "date" use="optional">
                        <annotation>
                                <documentation>Date the payload status was last
                                        updated. </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<element name = "SatelliteMission">
        <complexType>
                <sequence>
                        <element name="PrimaryMission" type="string"
                                minOccurs="1" maxOccurs="unbounded"/>
                        <element name="SecondaryMission"
                                minOccurs="0" maxOccurs="unbounded"/>
                </sequence>
                <attributeGroup ref = "ss:classification"/>
                <attribute name="source" type="string" use="optional">
                        <annotation>
                                <documentation>Source for the mission information
                                        (e.g., OpenSource, Owner/Operator, Initial
                                        Launch Alert Message, Intelligence).
                                </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<!--
**********
*** Satellite Element Related Type Definitions ***
**********
-->

<simpleType name = "satelliteNumberType" base = "positiveInteger">
        <annotation>
```

```
                <documentation>
                        Cataloged satellite numeric identifier assigned by
                        USSPACECOM.  Must be unique and a positive integer no
                        more than 6 digits long.  (Note:  As of October 2000, the legal
                        range of values is 5 digits long.  Because we expect this
                        range will soon be exceeded, we are specifying a legal range
                        of 6 digits.
                </documentation>
        </annotation>
        <maxInclusive value = "999999"/>
</simpleType>

<simpleType name = "satellitePieceIdentifierType" base = "string">
        <annotation>
                <documentation>Sequential piece identifier with range of values of
                                up to 3 alpha characters but not including
                                i or o or I or O</documentation>
        </annotation>
        <pattern value = "[a-h|j-n|p-z|A-H|J-N|P-Z]{1,3}"/>
</simpleType>

<simpleType name = "satCommonNameType" base = "string">
        <annotation>
                <documentation>Common name for the satellite (e.g., USA 148,
                        CSL-04 DEB, SL12 R/B(1)).  Name may be no more than
                        24 characters long.
                </documentation>
        </annotation>
        <maxLength value = "24"/>
</simpleType>

<simpleType name = "satOwnerType" base = "string">
        <annotation>
                <documentation>Country or organization that owns
                        satellite.  String may be no more than 5 characters
                        long.  Examples include:
                                USX (United States)
                                CIS (Commonwealth of Independent States)
                                PRC (Peoples Republic of China)
                                ESA (European Space Agency)
                                GLOB (Globalstar)
```

```
                </documentation>
        </annotation>
        <maxLength value = "5"/>
</simpleType>

<simpleType name = "satellitePayloadStatusType" base = "string">
        <annotation>
                <documentation>Status of satellite payload.  Must be one of
                        enumerated values of:
                                Active, Inactive, MissionEnded, or Dead.
                </documentation>
        </annotation>
        <enumeration value = "Active"/>
        <enumeration value = "Inactive"/>
        <enumeration value = "MissionEnded"/>
        <enumeration value = "Dead"/>
</simpleType>

<simpleType name = "satelliteObjectType" base = "string">
        <annotation>
                <documentation>Type of object.  Must be one of enumerated values
                        of: rocketBody, debris, platform, or unknown.
                </documentation>
        </annotation>
        <enumeration value = "rocketBody"/>
        <enumeration value = "debris"/>
        <enumeration value = "platform"/>
        <enumeration value = "unknown"/>
</simpleType>


<simpleType name = "satelliteLaunchSiteType" base = "string">
        <annotation>
                <documentation source="LaunchSiteDefinitions.doc">
                        Alphabetic string of no more than six characters indicating
                        launch site.  Must be one of satelliteLaunchSiteType's
                        enumerated values.  Definition of these abbreviations can be
                        found in the noted source file.
                </documentation>
        </annotation>
        <enumeration value = "AFETR"/>
```

```
                <enumeration value = "AFWTR"/>
                <enumeration value = "ALA"/>
                <enumeration value = "FRGUI"/>
                <enumeration value = "HGSTR"/>
                <enumeration value = "KSCUT"/>
                <enumeration value = "KYMTR"/>
                <enumeration value = "PKMTR"/>
                <enumeration value = "SCTMR"/>
                <enumeration value = "SNMLP "/>
                <enumeration value = "SRI"/>
                <enumeration value = "TNSTA"/>
                <enumeration value = "TTMTR"/>
                <enumeration value = "WOMRA"/>
                <enumeration value = "WUZ"/>
                <enumeration value = "XIC"/>
                <enumeration value = "YAVNE"/>
        </simpleType>

        <!-- NOTE:  FILE WON'T VALIDATE IF THIS ATTRIBUTE GROUP IS MOVED
                TO THE GLOBALS FILE.  IT'S A BUG IN THE VALIDATOR.  -->

        <attributeGroup name = "classification">
                <attribute name = "classLevel" type = "ss:classificationLevelType"
                                                        use="required"/>
                <attribute name = "dissemination" type = "ss:disseminationType"
                                                        use="optional"/>
                <attribute name = "classificationGuide" type = "string"
                                                        use="optional"/>
        </attributeGroup>
</schema>
```

D.4 LaunchSiteDefinitions.doc

Satellite Launch Sites

List of satellite launch sites used in the Space Surveillance schema. The launch site short names are included in an enumerated list in the defined type called "satelliteLaunchSiteType"

Launch Site Short Name	Launch Site Long Name
AFETR	Air Force Eastern Test Range
AFWTR	Air Force Western Test Range
ALA	Al-Andar
FRGUI	French Guiana
HGSTR	Hammaguira Space Track Range
KSCUT	Kagoshima Space Center University of Tokyo
KYMTR	Kapustin Yar Missile and Space Complex
PKMTR	Plesetsk Missile and Space Complex
SCTMR	Shuangchenzi Missile Test Range
SNMLP	San Marco Launch Platform
SRI	Sr. Harikota
TNSTA	Tanegashima Space Center
TTMTR	Tyuratam Missile Test Range
WOMRA	Woomera
WUZ	Wuzhai Launch Facility or Taiyuan
XIC	Xi Chang Launch Facility
YAVNE	Yavne Launch Facility

D.5 Sensor.xsd

```
<?xml version ="1.0"?>
<schema targetNamespace = "urn:schemas-gov:SpaceSurveillance"
        elementFormDefault="qualified"
      xmlns:ss="urn:schemas-gov:SpaceSurveillance"
        xmlns = "http://www.w3.org/1999/XMLSchema"
        version="1.0">

        <!--Schema for Sensor from a Space Surveillance perspective
                Presumes a ground-based sensor.  Should be extended to include
                other types of sensors.
        **************
        *** File name:  Sensor.xsd ***
```

```
**************
*** As of:      October 6, 2000
**************
-->

<element name = "Sensor">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Sensor used to collect data.
                        For the preliminary space surveillance ontology, these
                        ground-based sensors are used to collect observations on
                        satellites.</documentation>
                </annotation>
                <sequence>
                        <element ref= "ss:SensorNumber"/>
                        <element ref= "ss:SensorName"/>
                        <element ref= "ss:SensorShortName"/>
                        <element ref= "ss:SensorMission"/>
                        <element ref= "ss:SensorLocation"/>
                        <element ref= "ss:OwningCountry"/>
                        <element ref= "ss:Device" minOccurs="1"
                                                maxOccurs="unbounded"/>
                </sequence>
                <attributeGroup ref="ss:classification"/>
        </complexType>
</element>

<element name = "SensorNumber" type = "ss:sensorNumberType">
        <annotation>
                <documentation>Unique identifier for the sensor complex.
                </documentation>
        </annotation>
</element>

<element name = "SensorName">
        <complexType type = "string" content="textOnly">
                <annotation>
                        <documentation>Name of the sensor; often referred to as
                                        sensor long name.</documentation>
                </annotation>
                <attributeGroup ref="ss:classification"/>
```

```
        </complexType>
</element>

<element name = "SensorShortName" type = "ss:sensorShortNameType">
        <annotation>
                <documentation>A 3-character abbreviation for the sensor name.
                </documentation>
        </annotation>
</element>


<element name = "SensorLocation" type = "ss:geodeticLocationType">
                <annotation>
                        <documentation>General location of the sensor complex.
                                        Not to be used for observation calculations.
                                        (see DeviceLocation).
                        </documentation>
                </annotation>
</element>

<element name = "OwningCountry" type = "string">
        <annotation>
                <documentation>Country that owns the sensor.</documentation>
        </annotation>
</element>

<element name = "Device">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Designation of a specific device at the
                                sensor location.  A sensor may have multiple
                                devices.</documentation>
                </annotation>
                <sequence>
                        <element ref= "ss:DeviceNumber"/>
                        <element ref= "ss:DeviceName"/>
                        <element ref= "ss:DeviceType"/>
                        <element ref= "ss:DeviceLocation"/>
                </sequence>
        </complexType>
</element>
```

```
<element name = "SensorMission">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Mission of the sensor -- a container element
                                for the sensor primary, secondary, and tertiary
                                mission.</documentation>
                </annotation>
                <sequence>
                        <element ref= "ss:PrimaryMission"
                                                minOccurs="1" maxOccurs="1"/>
                        <element ref= "ss:SecondaryMission"
                                                minOccurs="0" maxOccurs="1"/>
                        <element ref= "ss:TertiaryMission"
                                                minOccurs="0" maxOccurs="1"/>
                </sequence>
                <attributeGroup ref="ss:classification"/>
        </complexType>
</element>

<element name = "PrimaryMission">
        <complexType type = "ss:sensorMissionType" content="textOnly">
                <annotation>
                        <documentation>Sensor's primary mission.
                                        Must be one of sensorMissionType's
                                        enumerated types.</documentation>
                </annotation>
                <attributeGroup ref="ss:classification"/>
        </complexType>
</element>

<element name = "SecondaryMission">
        <complexType type = "ss:sensorMissionType" content="textOnly">
                <annotation>
                        <documentation>Sensor's secondary mission.
                                        Must be one of sensorMissionType's
                                        enumerated types.</documentation>
                </annotation>
                <attributeGroup ref="ss:classification"/>
        </complexType>
</element>
```

```
<element name = "TertiaryMission">
        <complexType type = "ss:sensorMissionType" content="textOnly">
                <annotation>
                        <documentation>Sensor's tertiary mission.
                                        Must be one of sensorMissionType's
                                        enumerated types.</documentation>
                </annotation>
                <attributeGroup ref="ss:classification"/>
        </complexType>
</element>

<element name = "DeviceNumber" type = "ss:sensorDeviceNumberType">
        <annotation>
                <documentation>Unique numeric identifier for specific device at the
                                sensor location.  May be no more than 3 digits in
                                length. </documentation>
        </annotation>
</element>

<element name = "DeviceName" type = "string">
        <annotation>
                <documentation>Name given to a specific device at the sensor
                                location.</documentation>
        </annotation>
</element>

<element name = "DeviceType" type = "ss:sensorDeviceType">
        <annotation>
                <documentation>Type of the device at the sensor location.
                                Must be one of sensorDeviceType's enumerated types.
                </documentation>
        </annotation>
</element>

<element name = "DeviceLocation" type = "ss:geodeticLocationType">
        <annotation>
                <documentation>Geodetic location of the specific device at the
                                sensor location.
                </documentation>
        </annotation>
```

```
</element>

<element name = "Latitude">
        <complexType type="ss:latitudeType" content = "textOnly">
                <annotation>
                        <documentation>The distance of a point on the earth's
                                surface from the equator.
                        </documentation>
                </annotation>
                <attribute name="units" use="fixed" value="degrees" type="string">
                        <annotation>
                                <documentation>Latitude units are measured in
                                        degrees.</documentation>
                        </annotation>
                </attribute>
                <attribute name="precision" type="ss:locationPrecisionType"
                                use="optional">
                        <annotation>
                                <documentation>Indicator of the precision provided.
                                </documentation>
                        </annotation>
                </attribute>
                <attribute name = "hemisphere">
                        <annotation>
                                <documentation>Designation of latitude's hemisphere
                                        as either N (North) or S (South)
                                </documentation>
                        </annotation>
                        <simpleType base = "string">
                                <enumeration value = "N"/>
                                <enumeration value = "S"/>
                        </simpleType>
                </attribute>
        </complexType>
</element>

<element name = "Longitude">
        <complexType type="ss:longitudeType" content="textOnly">
                <annotation>
                        <documentation>Distance on the earth's surface east or west
                                of the Greenwich meridian.</documentation>
```

```
                </annotation>
                <attribute name="units" use="fixed" value="degrees" type="string">
                        <annotation>
                                <documentation>Longitude units are measured in
                                        degrees.</documentation>
                        </annotation>
                </attribute>
                <attribute name="precision" type="ss:locationPrecisionType"
                                use="optional">
                        <annotation>
                                <documentation>Indicator of the precision provided.
                                </documentation>
                        </annotation>
                </attribute>
                <attribute name = "hemisphere">
                        <annotation>
                                <documentation>Designation of longitude's
                                        hemisphere as either W (West) or E (East)
                                </documentation>
                        </annotation>
                        <simpleType base = "string">
                                <enumeration value = "W"/>
                                <enumeration value = "E"/>
                        </simpleType>
                </attribute>
        </complexType>
</element>

<element name = "Altitude">
        <complexType type="float" content = "textOnly">
                <annotation>
                        <documentation>The vertical elevation of an object above
                                mean sea level measured in meters.</documentation>
                </annotation>
                <attribute name="units" use="fixed" value="meters" type="string">
                        <annotation>
                                <documentation>Altitude units are measured in meters
                                        above mean sea level. </documentation>
                        </annotation>
                </attribute>
                <attribute name="precision" type="ss:locationPrecisionType"
```

```
                                        use="optional">
                        <annotation>
                                <documentation>Indicator of the precision provided.
                                </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<!--
**************
*** Sensor element related type definitions ***
**************
-->

<simpleType name = "sensorNumberType" base = "positiveInteger">
        <annotation>
                <documentation>Unique identifier for the sensor complex.
                        Must be an integer in the range 1-9999.
                </documentation>
        </annotation>
        <maxInclusive value = "9999"/>
</simpleType>

<simpleType name = "sensorDeviceNumberType" base = "positiveInteger">
        <annotation>
                <documentation>Sensor Device Number must be an integer
                        with legal range of values 1-999.
                </documentation>
        </annotation>
        <maxInclusive value = "999"/>
</simpleType>

<simpleType name = "sensorShortNameType" base = "string">
        <annotation>
                <documentation>Sensor short name -- a 3 character abbreviation for
                                the sensor name.
                </documentation>
        </annotation>
        <pattern value = "[a-z|A-Z]?[a-z|A-Z]?[a-z|A-Z]"/>
</simpleType>
```

```
<simpleType name = "latitudeType" base = "float">
        <annotation>
                <documentation>Latitude must be in the range from 0.0-90.0
                </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxInclusive value = "90.0"/>
</simpleType>

<simpleType name = "longitudeType" base = "float">
        <annotation>
                <documentation>Longitude must be in the range from 0.0-180.0
                </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxInclusive value = "180.0"/>
</simpleType>

<simpleType name = "sensorMissionType" base = "string">
        <annotation>
                <documentation>Sensor Mission Type enumerated values.
                        Legal values are:
                                "AFSCN" (Air Force Satellite Control Network)
                                "Intelligence"
                                "MissileWarning"
                                "SSN"  (Space Surveillance Network)
                </documentation>
        </annotation>
        <enumeration value = "AFSCN"/>
        <enumeration value = "Intelligence"/>
        <enumeration value = "MissileWarning"/>
        <enumeration value = "SSN"/>
</simpleType>

<simpleType name = "sensorDeviceType" base = "string">
        <annotation>
                <documentation>Sensor Device Type enumerated values.
                        Legal values are:
                                "PhasedArray"
                                "MechanicalRadar"
```

```
                                "Optical"
                                "Fence"
                                "Laser"
                        </documentation>
                </annotation>
                <enumeration value = "PhasedArray"/>
                <enumeration value = "MechanicalRadar"/>
                <enumeration value = "Optical"/>
                <enumeration value = "Fence"/>
                <enumeration value = "Laser"/>
</simpleType>

<simpleType name = "locationPrecisionType" base = "string">
        <annotation>
                <documentation>Indicator of the precision provided:
                                values may be "float" or "double"
                                Placeholder until determine how to characterize.
                </documentation>
        </annotation>
        <enumeration value = "float"/>
        <enumeration value = "double"/>
</simpleType>

<complexType name = "geodeticLocationType">
        <annotation>
                <documentation>Location in a geodetic reference frame with latitude
                        and longitude measured in degrees and altitude measured in
                        meters above sea level. </documentation>
        </annotation>
        <element ref= "ss:Latitude"/>
        <element ref= "ss:Longitude"/>
        <element ref= "ss:Altitude"/>
        <attributeGroup ref="ss:classification"/>
        <attribute name = "dateUpdated" type = "date" use="optional">
                <annotation>
                        <documentation>Date the sensor location was last updated.
                                Format is: CCYY-MM-DD </documentation>
                </annotation>
        </attribute>
        <attribute name = "source" type = "string" use="optional">
                <annotation>
```

```
                        <documentation>Source for the location data (e.g., GPS,
                                survey). </documentation>
                    </annotation>
                </attribute>
            </complexType>

            <!-- NOTE:  FILE WON'T VALIDATE IF THIS ATTRIBUTE GROUP IS MOVED
                    TO THE GLOBALS FILE.  IT'S A BUG IN THE VALIDATOR.  -->

            <attributeGroup name = "classification">
                <attribute name = "classLevel" type = "ss:classificationLevelType"
                                                    use="required"/>
                <attribute name = "dissemination" type = "ss:disseminationType"
                                                    use="optional"/>
                <attribute name = "classificationGuide" type = "string"
                                                    use="optional"/>
            </attributeGroup>
</schema>
```

D.6 SensorTasking.xsd

```
<?xml version ="1.0"?>
<schema targetNamespace = "urn:schemas-gov:SpaceSurveillance"
        elementFormDefault="qualified"
      xmlns:ss="urn:schemas-gov:SpaceSurveillance"
        xmlns = "http://www.w3.org/1999/XMLSchema"
        version="1.0">

  <!--Schema for Satellite (from a Space Surveillance perspective
        **************
        *** File name:  SensorTasking.xsd ***
        **************
        *** As of:      October 6, 2000
        **************
        -->

        <element name = "SensorTask">
            <complexType content = "elementOnly">
                <annotation>
                    <documentation>Overall container for all tasking data.
                    </documentation>
```

```
                </annotation>
                <sequence minOccurs = "0" maxOccurs = "unbounded">
                        <element ref = "ss:TaskedSensor"/>
                </sequence>
        </complexType>
</element>

<element name = "TaskedSensor">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>TaskSensor element contains tasking data for
                                a particular sensor site including input parameters
                                for the tasking software, tasking metrics, and the
                                sensor's current tasking </documentation>
                </annotation>
                <sequence>
                        <element ref = "ss:Controls" minOccurs="0"
                                                        maxOccurs="1"/>
                        <choice>
                                <element ref = "ss:TaskDaily"/>
                                <element ref = "ss:TaskUpdate"/>
                        </choice>
                        <element ref="ss:TaskingMetrics" minOccurs="0"
                                                         maxOccurs="1"/>
                </sequence>
                <attributeGroup ref = "ss:classification"/>
                <attribute name="sensor" use="required"
                                                type="ss:sensorNumberType"/>
        </complexType>
</element>

<element name = "TaskDaily">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>TaskDaily element contains data for the
                                Daily Tasking Message to a particular sensor site.
                        </documentation>
                </annotation>
                <element ref="ss:Category" minOccurs="0"
                                                maxOccurs="unbounded"/>
                <attributeGroup ref = "ss:classification"/>
```

```
        </complexType>
</element>

<element name = "Category">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Tasking category describes for the sensor the
                                priority (1-5) and amount of data collected on each
                                satellite (represented by the tasking code).
                        </documentation>
                </annotation>
                <element ref = "ss:Satno" minOccurs = "0"
                                                maxOccurs = "unbounded"/>
                <attribute name="priority" type="ss:taskingPriorityType"
                                        use="required"/>
                <attribute name="code" use="required">
                        <simpleType base = "string">
                                <maxLength value = "1"/>
                        </simpleType>
                </attribute>
        </complexType>
</element>

<element name = "TaskUpdate">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>TaskUpdate element contains data for the
                                Update Tasking Message to a particular sensor site.
                        </documentation>
                </annotation>
                <element ref="ss:Category" minOccurs="0"
                                                maxOccurs="unbounded"/>
                <attributeGroup ref = "ss:classification"/>
        </complexType>
</element>

<element name = "TaskingMetrics">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Selected data for assessing sensors
                                        performance to tasking.</documentation>
```

```
                </annotation>
                <sequence>
                        <element ref = "ss:PercentOfMaxTracks"/>
                        <element ref = "ss:PercentOfNetworkLoad"/>
                        <element ref = "ss:PercentResponseToTasking"/>
                </sequence>
                <attributeGroup ref = "ss:classification"/>
        </complexType>
</element>

<element name = "Controls">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>Input parameters for tasking
                        </documentation>
                </annotation>
                <sequence>
                        <element ref = "ss:TaskCreation"/>
                        <element ref = "ss:TaskValidPeriod"/>
                        <element ref = "ss:AssumedDownTime" minOccurs="0"
                                                maxOccurs="unbounded"/>
                        <element ref = "ss:TaskCapacity"/>
                </sequence>
                <attributeGroup ref = "ss:classification"/>
        </complexType>
</element>

<element name = "TaskCreation" type="ss:dateTimeType">
        <annotation>
                <documentation>Date and Time the tasking data was generated.
                </documentation>
        </annotation>
</element>

<element name = "TaskValidPeriod">
        <complexType content="elementOnly">
                <annotation>
                        <documentation>TaskValidTimes specifies the tasking day for
                                each sensor site. Includes a ValidStart
                                (date and time) and a ValidEnd (date and
                                time) </documentation>
```

```
                </annotation>
                <sequence>
                        <element name = "ValidStart" type = "ss:dateTimeType"/>
                        <element name = "ValidEnd" type = "ss:dateTimeType"/>
                </sequence>

        </complexType>
</element>

<element name = "AssumedDownTime">
        <complexType content = "elementOnly">
                <annotation>
                        <documentation>AssumedDownTime represents the windows
                                of outage times entered in the tasking algorithm for
                                each sensor. Each downtime is defined by a
                                DownTimeStart (date and time) and a DownTimeEnd
                                (date and time) </documentation>
                </annotation>
                <sequence>
                        <element name="DownTimeStart" type="ss:dateTimeType"/>
                        <element name="DownTimeEnd" type="ss:dateTimeType"/>
                </sequence>
        </complexType>
</element>

<element name = "TaskCapacity" type = "nonNegativeInteger">
        <annotation>
                <documentation>TaskCapacity is the maximum number of tracks a
                                sensor can perform in a 24-hour period.
                </documentation>
        </annotation>
</element>

<element name = "PercentOfMaxTracks" type = "decimal">
        <annotation>
                <documentation>PercentOfMaxTracks is the percent of this
                                tasking as compared to the sensors max capacity.
                </documentation>
        </annotation>
</element>
```

```
<element name = "PercentOfNetworkLoad" type = "decimal">
        <annotation>
                <documentation>PercentOfNetworkLoad is the percent of this
                        particular sensor's tasking compared to all sensors' tasking.
                </documentation>
        </annotation>
</element>

<element name = "PercentResponseToTasking" type = "decimal">
        <annotation>
                <documentation>PercentResponseToTasking is the percent of actual
                        tracking compared to the tasking assigned.</documentation>
        </annotation>
</element>

<!--
**********
*** Sensor Tasking Related Type Definitions ***
**********
-->

<simpleType name = "taskingPriorityType" base = "integer">
        <annotation>
                <documentation>
                        Tasking Priority type with legal range of values 1-5.
                </documentation>
        </annotation>
        <minInclusive value = "1"/>
        <maxInclusive value = "5"/>
</simpleType>

<!-- NOTE:  FILE WON'T VALIDATE IF THIS ATTRIBUTE GROUP IS MOVED
        TO THE GLOBALS FILE.  IT'S A BUG IN THE VALIDATOR.  -->

<attributeGroup name = "classification">
        <attribute name = "classLevel" type = "ss:classificationLevelType"
                                                use="required"/>
        <attribute name = "dissemination" type = "ss:disseminationType"
                                                use="optional"/>
        <attribute name = "classificationGuide" type = "string"
                                                use="optional"/>
```

```
        </attributeGroup>
</schema>
```

D.7 SatObservation.xsd

```
<?xml version ="1.0"?>

<schema xmlns = "http://www.w3.org/1999/XMLSchema"
        targetNamespace = "urn:schemas-gov:SpaceSurveillance"
        elementFormDefault = "qualified"
        xmlns:ss = "urn:schemas-gov:SpaceSurveillance"
        version="1.0">

  <!--Schema for Satellite Observation by a ground-based sensor
        **************
        *** File name:  SatObservation.xsd ***
        **************
        *** As of:      October 6, 2000
        **************

        Reference: [AFSPCI 60-102] Air Force Space Command (AFSPC) Astrodynamic
        Standards, AFSPC Instruction 60-102, 11 March 1996.
        -->

        <element name = "SatObservation">
                <annotation>
                        <documentation>Observation of a satellite by a ground-based sensor
                        </documentation>
                </annotation>
                <complexType content = "elementOnly">
                        <sequence>
                                <element ref = "ss:ObType"/>
                                <element ref = "ss:SensorNumber"/>
                                <element ref = "ss:SatelliteNumber"/>
                                <element ref = "ss:ObDate"/>
                                <element ref = "ss:ObTime"/>
                                <element ref = "ss:OriginalTag"/>
                                <element ref = "ss:CCTag"/>
                        </sequence>
                        <attributeGroup ref="ss:classification"/>
                </complexType>
```

```
</element>

<element name = "ObType">
        <annotation>
                <documentation>Satellite Observation Types. Can be one of the
                        following types of elements:  ObType1, ObType2, ObType3,
                        ObType4, ObType5, ObType9</documentation>
        </annotation>
        <complexType content = "elementOnly">
                <choice>
                        <element ref = "ss:ObType1"/>
                        <element ref = "ss:ObType2"/>
                        <element ref = "ss:ObType3"/>
                        <element ref = "ss:ObType4"/>
                        <element ref = "ss:ObType5"/>
                        <element ref = "ss:ObType9"/>
                </choice>
        </complexType>
</element>

<element name = "ObDate">
        <complexType type = "date" content="textOnly">
                <annotation>
                        <documentation>Date observation taken.
                                        Format is: CCYY-MM-DD </documentation>
                </annotation>
        </complexType>
</element>

<element name = "ObTime">
        <complexType type = "time" content="textOnly">
                <annotation>
                        <documentation>Time observation taken.
                                        Format is: hh:mm:ss.sss </documentation>
                </annotation>
        </complexType>
</element>

<element name = "ObType1" type = "ss:obType1Type"/>
<element name = "ObType2" type = "ss:obType2Type"/>
<element name = "ObType3" type = "ss:obType3Type"/>
```

```
<element name = "ObType4" type = "ss:obType4Type"/>
<element name = "ObType5" type = "ss:obType5Type"/>
<element name = "ObType9" type = "ss:obType9Type"/>

<complexType name = "obType1Type">
        <annotation>
                <documentation>Type 1 satellite observations include: Elevation and
                        Azimuth</documentation>
        </annotation>
        <sequence>
                <element ref = "ss:Elevation"/>
                <element ref = "ss:Azimuth"/>
        </sequence>
</complexType>

<complexType name= "obType2Type" base="ss:obType1Type"
                        derivedBy="extension">
        <annotation>
                <documentation>Type 2 satellite observations include: Elevation,
                        Azimuth, and Range</documentation>
        </annotation>
        <sequence>
                <element ref = "ss:Range"/>
        </sequence>
</complexType>

<complexType name = "obType3Type" base="ss:obType2Type"
                        derivedBy="extension">
        <annotation>
                <documentation>Type 3 satellite observations include:
                          Elevation, Azimuth, Range, and RangeRate
                </documentation>
        </annotation>
        <sequence>
                <element ref = "ss:RangeRate"/>
        </sequence>
</complexType>

<complexType name = "obType4Type" base="ss:obType3Type"
                        derivedBy="extension">
        <annotation>
```

```
                <documentation>Type 4 satellite observations include:
                                Elevation, Azimuth, Range, RangeRate,
                                AzimuthRate, and ElevationRate</documentation>
        </annotation>
        <sequence>
                <element ref = "ss:AzimuthRate"/>
                <element ref = "ss:ElevationRate"/>
        </sequence>
</complexType>

<complexType name = "obType5Type">
        <annotation>
                <documentation>Type 5 satellite observations include:
                                RtAscension, and Declination</documentation>
        </annotation>
        <sequence>
                <element ref = "ss:RtAscension"/>
                <element ref = "ss:Declination"/>
        </sequence>
</complexType>

<complexType name = "obType9Type" base = "ss:obType5Type"
                                derivedBy="extension">
        <annotation>
                <documentation>Type 9 satellite observations include:
                                RtAscension, Declination, and
                                SensorLocation described as an E,F,G position
                                vector of mobile sensor measured in meters)
                </documentation>
        </annotation>
        <sequence>
                <element ref = "ss:SensorLocationStateVector"/>
        </sequence>
</complexType>

<element name = "Elevation">
        <complexType type = "ss:elevationType" content="textOnly">
                <annotation>
                        <documentation>The angular distance of a body above
                                or below the horizon measured along the great
                                circle passing through the body and the zenith.
```

```
                    [AFSPCI 60-102] Measurement units must be degrees.
                </documentation>
            </annotation>
            <attribute name="units" type="string" use="fixed" value="degrees" />
        </complexType>
</element>

<element name = "Azimuth">
        <complexType type = "ss:azimuthRtAscensionType" content="textOnly">
            <annotation>
                <documentation>The angular distance measured clockwise
                    along the horizon on the celestial sphere.
                    [AFSPCI 60-102] Measurement units must be degrees.
                </documentation>
            </annotation>
            <attribute name="units" type="string" use="fixed" value="degrees" />
        </complexType>
</element>

<element name = "Range">
        <complexType type = "double" content="textOnly">
            <annotation>
                <documentation>Distance from a sensor to a satellite.
                    [AFSPCI 60-102] Measurement units must be meters.
                </documentation>
            </annotation>
            <attribute name="units" type="string" use="fixed" value="meters" />
        </complexType>
</element>

<element name = "RangeRate">
        <complexType type = "double" content="textOnly">
            <annotation>
                <documentation>Rate-of-change of the distance from a sensor
                    to a satellite.  [AFSPCI 60-102]
                    Measurement units must be metersPerSecond.
                </documentation>
            </annotation>
            <attribute name="units" type="string" use="fixed"
                                value="metersPerSecond" />
        </complexType>
```

```
</element>

<element name = "AzimuthRate">
        <complexType type = "double" content="textOnly">
                <annotation>
                        <documentation>Rate of change of the azimuth.
                                Measurement units must be degreesPerSecond.
                        </documentation>
                </annotation>
                <attribute name="units" type="string" use="fixed"
                                        value="degreesPerSecond" />
        </complexType>
</element>

<element name = "ElevationRate">
        <complexType type = "double" content="textOnly">
                <annotation>
                        <documentation>Rate of change of the elevation.
                                Measurement units must be degreesPerSecond.
                        </documentation>
                </annotation>
                <attribute name="units" type="string" use="fixed"
                                        value="degreesPerSecond" />
        </complexType>
</element>

<element name = "RtAscension">
        <complexType type = "rtAscensionType" content="textOnly">
                <annotation>
                        <documentation>The angle between the vernal equinox
                                and the projection of the radius vector on the equatorial
                                plane, regarded as positive when measured eastward
                                from the vernal equinox. [AFSPCI 60-102]
                                Measurement units must be degrees.
                        </documentation>
                </annotation>
                <attribute name="units" type="string" use="fixed" value="degrees" />
        </complexType>
</element>

<element name = "Declination">
```

```
        <complexType type = "declinationType" content="textOnly">
                <annotation>
                        <documentation>The angle between the celestial equator and
                                a radius vector, regarded as positive when
                                measured north from the celestial equator.
                                [AFSPCI 60-102] Measurement units must be degrees.
                        </documentation>
                </annotation>
                <attribute name="units" type="string" use="fixed" value="degrees" />
        </complexType>
</element>

<element name = "SensorLocationStateVector">
        <complexType content = "elementOnly">
                <annotation>
                <documentation>A position vector describing the
                                instantaneous location of a mobile sensor in the
                                Earth-Fixed Geocentric (EFG) coordinate system.
                                Values are in meters and are epoched to the
                                observation's epoch.</documentation>
                </annotation>
                <sequence>
                        <element name = "E" type="integer"/>
                        <element name = "F" type="integer"/>
                        <element name = "G" type="integer"/>
                </sequence>
                <attribute name="units" type="string" use="fixed" value="meters" />
        </complexType>
</element>


<element name = "OriginalTag" type = "ss:satelliteNumberType">
        <annotation>
                <documentation>Satellite Number that observation originally tagged
                                to correspond to by sensor.</documentation>
        </annotation>
</element>

<element name = "CCTag" type = "ss:satelliteNumberType">
        <annotation>
```

```
                <documentation>Satellite Number observation tagged to by
                correlation center</documentation>
        </annotation>
</element>

<!--
**************
*** Satellite Observation Related Type Definitions ***
**************
-->

<simpleType name = "elevationType" base="double" derivedBy="restriction">
        <annotation>
                <documentation>Double precision elevation value with range
                        -90.0 to 90.0 degrees.  </documentation>
        </annotation>
        <minInclusive value = "-90.0"/>
        <maxInclusive value = "90.0"/>
</simpleType>

<simpleType name = "declinationType" base="double" derivedBy="restriction">
        <annotation>
                <documentation>Double precision value with range 0.0 to 180.0
                        degrees.  </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxInclusive value = "180.0"/>
</simpleType>

<simpleType name = "aziumthRtAscensionType" base="double"
                        derivedBy="restriction">
        <annotation>
                <documentation>Double precision value with range 0.0 to 360.0
                        degrees. </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxInclusive value = "360.0"/>
</simpleType>
```

```
        <!-- NOTE:  FILE WON'T VALIDATE IF THIS ATTRIBUTE GROUP IS MOVED
        TO THE GLOBALS FILE.  IT'S A BUG IN THE VALIDATOR.  -->

        <attributeGroup name = "classification">
                <attribute name = "classLevel" type = "ss:classificationLevelType"
                                                        use="required"/>
                <attribute name = "dissemination" type = "ss:disseminationType"
                                                        use="optional"/>
                <attribute name = "classificationGuide" type = "string"
                                                        use="optional"/>
        </attributeGroup>
</schema>
```

D.8 SatElset.xsd

```
<?xml version ="1.0"?>

<schema xmlns = "http://www.w3.org/1999/XMLSchema"
        targetNamespace = "urn:schemas-gov:SpaceSurveillance"
        elementFormDefault = "qualified"
        xmlns:ss = "urn:schemas-gov:SpaceSurveillance"
        version="1.0">

  <!--Schema for Satellite Orbital Element Set
        **************
        *** File name:  SatElset.xsd ***
        **************
        *** As of:      October 6, 2000
        **************
        -->

        <element name = "SatelliteElementSet">
                <complexType content = "elementOnly">
                        <annotation>
                                <documentation>
                                        The standard "2-line" format for element sets used
                                        within USSPACECOM.  The data describes a satellite's
                                        orbit using general perturbations theory (SGP or
                                        SGP4) and Keplerian elements.</documentation>
                        </annotation>
                        <sequence>
```

```
                    <element ref = "ss:SatelliteNumber"/>
                    <element ref = "ss:InternationalDesignator"/>
                    <element ref = "ss:OrbitalElements"/>
                </sequence>
        </complexType>
</element>

<element name = "OrbitalElements">
        <complexType content = "elementOnly">
                <annotation>
                    <documentation>
                        A selected set of six variables describing the
                        orbit.  The classical orbital elements are:
                        semi-major axis, eccentricity, inclination,
                        longitude of the ascending node, argument of
                        periapsis, and mean anomaly at epoch.
                    </documentation>
                </annotation>
                <sequence>
                    <element ref = "ss:ElementNumber"/>
                    <element ref = "ss:Epoch"/>
                    <element ref = "ss:NDot"/>
                    <element ref = "ss:NDotDot"/>
                    <element ref = "ss:BStar"/>
                    <element ref = "ss:EphemerisType"/>
                    <element ref = "ss:Inclination"/>
                    <element ref = "ss:RtAscension"/>
                    <element ref = "ss:Eccentricity"/>
                    <element ref = "ss:ArgPerigee"/>
                    <element ref = "ss:MeanAnomoly"/>
                    <element ref = "ss:MeanMotion"/>
                    <element ref = "ss:EpochRevolution"/>
                </sequence>
                <attributeGroup ref = "ss:classification"/>
        </complexType>
</element>

<element name = "ElementNumber" type = "positiveInteger">
        <annotation>
                <documentation>A number that is sequentially increased each time
                        the orbital elements are updated or transmitted.  Used in
```

```
                    determining data currency.</documentation>
        </annotation>
</element>

<element name = "Epoch" type="ss:dateTimeType">
        <annotation>
                <documentation>
                        An instant of time selected as a point of reference.
                        Epoch element includes a Date and Time.
                </documentation>
        </annotation>
</element>

<element name = "NDot" type = "ss:orbitNDotType">
        <annotation>
                <documentation>First derivative of satellite's mean motion with
                        respect to time, divided by 2 (Radians/Min**2).  This data
                        is used by SGP (Simplified General Perturbations) theory.
                </documentation>
        </annotation>
</element>

<element name = "NDotDot" type = "ss:orbitNDotDotType">
        <annotation>
                <documentation>The second derivative of the satellite's mean motion
                        with respect to time, divided by 6 (Radians/Min**3).  This
                        data is used by SGP (Simplified General Perturbations)
                        theory.
                </documentation>
        </annotation>
</element>

<element name = "BStar" type = "ss:orbitBStarType">
        <annotation>
                <documentation>A measure of the satellite's atmospheric drag used by
                        SGP4 (Simplified General Perturbations 4) theory.
                        (1/each radii(km))</documentation>
        </annotation>
</element>

<element name = "EphemerisType">
```

```
        <simpleType base = "string">
                <enumeration value = "SGP"/>
                <enumeration value = "SGP4"/>
        </simpleType>
        <annotation>
                <documentation>Theory used to calculate elements.  Values may be:
                                SGP (Simplified General Perturbations) or
                                SGP4 (Simplified General Perturbations 4)
                        SGP is an analytic method of generating ephemerides
                        for satellites in earth-centered orbits.
                </documentation>
        </annotation>
</element>

<element name = "Inclination">
        <complexType type = "ss:orbitInclinationType" content = "textOnly">
                <annotation>
                        <documentation>
                                The angle between the prime meridian and the
                                satellite's orbit plane and the earth's equatorial
                                plane.</documentation>
                </annotation>
                <attribute name="units" use= "fixed" value="degrees" type="string"/>
        </complexType>
</element>

<element name = "RtAscension">
        <complexType type = "ss:orbitRtAscensionType" content = "textOnly">
                <annotation>
                        <documentation>Right ascension of the ascending node.  The
                                angle between the prime meridian and the satellite's
                                ascending node measured counterclockwise in the
                                earth's equatorial plane.
                        </documentation>
                </annotation>
                <attribute name="units" use="fixed" value="degrees" type="string"/>
        </complexType>
</element>

<element name = "Eccentricity" type = "ss:orbitEccentricityType">
        <annotation>
```

```
                <documentation>A quantity describing the shape of the satellite's
                        orbit.</documentation>
        </annotation>
</element>

<element name = "ArgPerigee">
        <complexType type = "ss:orbitArgPerigeeType" content = "textOnly">
                <annotation>
                        <documentation>
                                Argument of Perigee.  The angle, in the plane of the
                                satellite's orbit, from the ascending node to
                                periapsis measured in the direction of the
                                satellite's motion.</documentation>
                </annotation>
                <attribute name="units" use="fixed" value="degrees" type="string"/>
        </complexType>
</element>

<element name = "MeanAnomoly">
        <complexType type = "ss:orbitMeanAnomalyType" content = "textOnly">
                <annotation>
                        <documentation>The angle relating the satellite's position
                                in the orbit to time, measured from perigee in the
                                direction of the satellite's motion.
                        </documentation>
                </annotation>
                <attribute name="units" use="fixed" value="degrees" type="string"/>
        </complexType>
</element>

<element name = "MeanMotion">
        <complexType type = "ss:orbitMeanMotionType" content="textOnly">
                <annotation>
                        <documentation>The average rate of motion of the satellite
                                (revolutions/day).
                        </documentation>
                </annotation>
                <attribute name="theoryUsed"
                                type="ss:orbitMeanMotionTheoryType"
                                use="required">
                        <annotation>
```

```
                                <documentation>Theory for determining Mean Motion
                                        is either "Kozai" (used in SGP) or "Brouwer"
                                        (used in SGP4).  </documentation>
                        </annotation>
                </attribute>
        </complexType>
</element>

<element name = "EpochRevolution" type = "nonNegativeInteger">
        <annotation>
                <documentation>The number of revolutions traveled by the satellite
                        while in orbit.  Range of values 0-99999</documentation>
        </annotation>
</element>

<!--
**************
*** Satellite Orbital Element Set Related Type Definitions ***
**************
-->

<simpleType name = "orbitNDotType" base = "float">
        <annotation>
                <documentation>
                        Orbit N Dot must be greater than -1.0 and less than 1.0
                </documentation>
        </annotation>
        <minExclusive value = "-1.0"/>
        <maxExclusive value = "1.0"/>
</simpleType>

<simpleType name = "orbitNDotDotType" base = "float">
        <annotation>
                <documentation>Orbit N Dot Dot must be in the range -0.01 to 0.01
                </documentation>
        </annotation>
        <minInclusive value = "-0.01"/>
        <maxInclusive value = "0.01"/>
</simpleType>

<simpleType name = "orbitBStarType" base = "float">
```

```
        <annotation>
                <documentation>Orbit BStar must be in the range -10.0 to 40.0
                </documentation>
        </annotation>
        <minInclusive value = "-10.0"/>
        <maxInclusive value = "40.0"/>
</simpleType>

<simpleType name = "orbitInclinationType" base = "float">
        <annotation>
                <documentation>Orbit inclination must be greater than or equal
                        to 0.0 and less than 180.0
                </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxExclusive value = "180.0"/>
</simpleType>

<simpleType name = "orbitRtAscensionType" base = "float">
        <annotation>
                <documentation>Orbit right ascension must be greater than or equal
                        to 0.0 and less than 360.0
                </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxExclusive value = "360.0"/>
</simpleType>

<simpleType name = "orbitEccentricityType" base = "float">
        <annotation>
                <documentation>Orbit eccentricity must be greater than or equal to
                        0.0 and less than 1.0
                </documentation>
        </annotation>
        <minInclusive value = "0.0"/>
        <maxExclusive value = "1.0"/>
</simpleType>

<simpleType name = "orbitArgPerigeeType" base = "float">
        <annotation>
                <documentation>Orbit argument of perigee must be greater than or
```

```
                    equal to 0.0 and less than 360.0
                </documentation>
            </annotation>
            <minInclusive value = "0.0"/>
            <maxExclusive value = "360.0"/>
</simpleType>

<simpleType name = "orbitMeanAnomalyType" base = "float">
            <annotation>
                <documentation>Orbit mean anomaly must be greater than or equal
                    to 0.0 and less than 360.0
                </documentation>
            </annotation>
            <minInclusive value = "0.0"/>
            <maxExclusive value = "360.0"/>
</simpleType>

<simpleType name = "orbitMeanMotionType" base = "float">
            <annotation>
                <documentation>Orbit mean motion must be greater than or equal
                    to 0.0 and less than or equal to 48.20
                </documentation>
            </annotation>
            <minInclusive value = "0.0"/>
            <maxInclusive value = "48.20"/>
</simpleType>

<simpleType name = "orbitMeanMotionTheoryType" base = "string">
            <annotation>
                <documentation>Orbit mean motion theory value must be either
                    "Kozai" or "Brouwer"
                </documentation>
            </annotation>
            <enumeration value = "Kozai"/>
            <enumeration value = "Brouwer"/>
</simpleType>

<!-- NOTE:  FILE WON'T VALIDATE IF THIS ATTRIBUTE GROUP IS MOVED
        TO THE GLOBALS FILE.  IT'S A BUG IN THE VALIDATOR.  -->

<attributeGroup name = "classification">
```

```
            <attribute name = "classLevel" type = "ss:classificationLevelType"
                                                    use="required"/>
            <attribute name = "dissemination" type = "ss:disseminationType"
                                                    use="optional"/>
            <attribute name = "classificationGuide" type = "string"
                                                    use="optional"/>
        </attributeGroup>
</schema>
```

D.9 ss.xml

```
<?xml version="1.0"?>

<SpaceSurveillanceData  xmlns="urn:schemas-gov:SpaceSurveillance"
            xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
            xsi:schemaLocation=
                "urn:schemas-gov:SpaceSurveillance
                 SpaceSurveillance.xsd">

    <Satellite satelliteNumber="17940">
        <InternationalDesignator classLevel="U">
            <LaunchYear>1987</LaunchYear>
            <LaunchNumber>39</LaunchNumber>
            <PieceIdentifier>A</PieceIdentifier>
        </InternationalDesignator>
        <CommonName classLevel="U" source="TASS">COSMOS
1843</CommonName>
        <Owner classLevel="U">CIS</Owner>
        <SatelliteObject classLevel="FOUO">
            <PayloadObject>
                <PayloadType>Scientific</PayloadType>
                <SatelliteStatus classLevel="U">Decayed</SatelliteStatus>
                <SatelliteMission classLevel="Unclassified"
                            source="TASS">
                                <PrimaryMission>Scientific</PrimaryMission>
                </SatelliteMission>
            </PayloadObject>
        </SatelliteObject>
        <LaunchData classLevel="U">
            <LaunchDate>1987-05-05</LaunchDate>
            <LaunchTime>08:30:27</LaunchTime>
```

```
            <LaunchSite>TTMTR</LaunchSite>
            <LaunchingAgency>CIS</LaunchingAgency>
        </LaunchData>
        <DecayDate>1987-05-19</DecayDate>
    </Satellite>

    <Satellite satelliteNumber="11622">
        <InternationalDesignator classLevel="U">
            <LaunchYear>1979</LaunchYear>
            <LaunchNumber>98</LaunchNumber>
            <PieceIdentifier>B</PieceIdentifier>
        </InternationalDesignator>
        <CommonName classLevel="U" source="AFSPC">OPS 9434 (DSCSII-
14)</CommonName>
        <Owner classLevel="U">USAF</Owner>
        <SatelliteObject classLevel="FOUO">
            <PayloadObject>
                <PayloadType>communications</PayloadType>
                <SatelliteStatus classLevel="U">Active</SatelliteStatus>
                <SatelliteMission classLevel="Unclassified"
                        source="USAF">
                    <PrimaryMission>USAF communications</PrimaryMission>
                </SatelliteMission>
            </PayloadObject>
        </SatelliteObject>
        <LaunchData classLevel="U">
            <LaunchDate>1979-11-21</LaunchDate>
            <LaunchTime>08:30:27</LaunchTime>
            <LaunchSite>AFETR</LaunchSite>
            <LaunchingAgency>USAF</LaunchingAgency>
        </LaunchData>
        <DecayDate></DecayDate>
    </Satellite>

    <Sensor classLevel="Unclassified" dissemination="Releasable">
        <SensorNumber>0399</SensorNumber>
        <SensorName classLevel="U">Eglin</SensorName>
        <SensorShortName>EGL</SensorShortName>
        <SensorMission classLevel="Unclassified">
            <PrimaryMission classLevel="Unclassified">SSN</PrimaryMission>
            <SecondaryMission classLevel="Unclassified">Test</SecondaryMission>
```

```
        </SensorMission>
        <SensorLocation classLevel="Unclassified"
                        dateUpdated="1999-03-02"
                source="GPS">
            <Latitude units="degrees" precision="double" hemisphere="N">30.5
                </Latitude>
            <Longitude units="degrees" precision="float" hemisphere="E">273.7
            </Longitude>
            <Altitude units="meters">17.0</Altitude>
        </SensorLocation>
        <OwningCountry>USA</OwningCountry>
        <Device>
            <DeviceNumber>399</DeviceNumber>
            <DeviceName>Eglin Near-Earth Phased Array Radar</DeviceName>
            <DeviceType>PhasedArray</DeviceType>
            <DeviceLocation classLevel="U">
            <Latitude units="degrees" precision="double" hemisphere="N">
                30.654321</Latitude>
            <Longitude units="degrees" precision="float" hemisphere="W">
                273.7654321</Longitude>
            <Altitude units="meters">17.6543210</Altitude>
            </DeviceLocation>
        </Device>
        <Device>
            <DeviceNumber>398</DeviceNumber>
            <DeviceName>Eglin Deep-Space Phased Array Radar</DeviceName>
            <DeviceType>PhasedArray</DeviceType>
            <DeviceLocation classLevel="U">
            <Latitude units="degrees" precision="double" hemisphere="N">
                30.654321</Latitude>
            <Longitude units="degrees" precision="float" hemisphere="W">
                273.7654321</Longitude>
            <Altitude units="meters">17.6543210</Altitude>
            </DeviceLocation>
        </Device>
    </Sensor>

    <Sensor classLevel="Unclassified" dissemination="Releasable">
        <SensorNumber>0354</SensorNumber>
        <SensorName classLevel="U">Ascension</SensorName>
        <SensorShortName>ASC</SensorShortName>
```

```
        <SensorMission classLevel="Unclassified">
            <PrimaryMission classLevel="Unclassified">NASA AFETR Support
                </PrimaryMission>
            <SecondaryMission classLevel="Unclassified">SSN</SecondaryMission>
        </SensorMission>
        <SensorLocation classLevel="Unclassified" dateUpdated="1999-03-02"
                source="GPS">
            <Latitude units="degrees" precision="double" hemisphere="S">
                7.9</Latitude>
            <Longitude units="degrees" precision="float" hemisphere="E">
                345.5</Longitude>
            <Altitude units="meters">30.0</Altitude>
        </SensorLocation>
        <OwningCountry>USA</OwningCountry>
        <Device>
            <DeviceNumber>354</DeviceNumber>
            <DeviceName>Ascension Primary Tracker</DeviceName>
            <DeviceType>MechanicalRadar</DeviceType>
            <DeviceLocation classLevel="U">
            <Latitude units="degrees" precision="double" hemisphere="N">
                -7.987654321</Latitude>
            <Longitude units="degrees" precision="float" hemisphere="W">
                345.54321</Longitude>
            <Altitude units="meters">31.098765</Altitude>
            </DeviceLocation>
        </Device>
        <Device>
            <DeviceNumber>355</DeviceNumber>
            <DeviceName>Ascension Secondary Tracker</DeviceName>
            <DeviceType>MechanicalRadar</DeviceType>
            <DeviceLocation classLevel="U">
            <Latitude units="degrees" precision="double" hemisphere="N">
                -7.980654321</Latitude>
            <Longitude units="degrees" precision="float" hemisphere="W">
                345.54021</Longitude>
            <Altitude units="meters">37.1234560</Altitude>
            </DeviceLocation>
        </Device>
    </Sensor>

    <SatObservation classLevel="Unclassified">
```

```
    <ObType>
        <ObType1>
            <Elevation units="degrees">68.90</Elevation>
            <Azimuth units="degrees">90.78</Azimuth>
        </ObType1>
    </ObType>
    <SensorNumber>741</SensorNumber>
    <SatelliteNumber>12345</SatelliteNumber>
    <ObDate>2000-03-02</ObDate>
    <ObTime>12:10:25.345</ObTime>
    <OriginalTag>12345</OriginalTag>
    <CCTag>99999</CCTag>
</SatObservation>

<SatObservation classLevel="Unclassified">
    <ObType>
        <ObType4>
          <Elevation units="degrees">68.90</Elevation>
          <Azimuth units="degrees">90.78</Azimuth>
          <Range units="meters">1245.78</Range>
          <RangeRate units="metersPerSecond">345.678903</RangeRate>
          <AzimuthRate units="degreesPerSecond">2.7834564</AzimuthRate>
          <ElevationRate units="degreesPerSecond">0.0345678</ElevationRate>
        </ObType4>
    </ObType>
    <SensorNumber>388</SensorNumber>
    <SatelliteNumber>1122</SatelliteNumber>
    <ObDate>2000-03-02</ObDate>
    <ObTime>22:21:05.350</ObTime>
    <OriginalTag>90023</OriginalTag>
    <CCTag>1122</CCTag>
</SatObservation>

<SatObservation classLevel="Unclassified">
    <ObType>
        <ObType5>
            <RtAscension units="degrees">184.787654321</RtAscension>
            <Declination units="degrees">78.787654321</Declination>
        </ObType5>
    </ObType>
    <SensorNumber>245</SensorNumber>
```

```
        <SatelliteNumber>23357</SatelliteNumber>
        <ObDate>2000-03-02</ObDate>
        <ObTime>22:21:02.100</ObTime>
        <OriginalTag>10000</OriginalTag>
        <CCTag>23357</CCTag>
</SatObservation>

<SatelliteElementSet>
        <SatelliteNumber>19557</SatelliteNumber>
        <InternationalDesignator classLevel="U">
                <LaunchYear>1988</LaunchYear>
                <LaunchNumber>92</LaunchNumber>
                <PieceIdentifier>D</PieceIdentifier>
        </InternationalDesignator>
        <OrbitalElements classLevel="Unclassified">
                <ElementNumber>0007</ElementNumber>
                <Epoch>
                        <Date>2000-09-14</Date>
                        <Time>23:10:15.522</Time>
                </Epoch>
                <NDot>-0.0000045</NDot>
                <NDotDot>0.0</NDotDot>
                <BStar>.00010000</BStar>
                <EphemerisType>SGP</EphemerisType>
                <Inclination>67.636987</Inclination>
                <RtAscension>258.4641</RtAscension>
                <Eccentricity>.5412831</Eccentricity>
                <ArgPerigee>274.0424</ArgPerigee>
                <MeanAnomoly>028.9720</MeanAnomoly>
                <MeanMotion theoryUsed="Kozai">2.04126354</MeanMotion>
                <EpochRevolution>8807</EpochRevolution>
        </OrbitalElements>
</SatelliteElementSet>

<SatelliteElementSet>
        <SatelliteNumber>24841</SatelliteNumber>
        <InternationalDesignator classLevel="U">
                <LaunchYear>1988</LaunchYear>
                <LaunchNumber>30</LaunchNumber>
                <PieceIdentifier>F</PieceIdentifier>
        </InternationalDesignator>
```

```
        <OrbitalElements classLevel="Unclassified">
            <ElementNumber>0416</ElementNumber>
            <Epoch>
                <Date>2000-09-14</Date>
                <Time>22:04:25.534</Time>
            </Epoch>
            <NDot>-0.00001042</NDot>
            <NDotDot>0.0</NDotDot>
            <BStar>.00036485</BStar>
            <EphemerisType>SGP</EphemerisType>
            <Inclination>86.3956</Inclination>
            <RtAscension>47.8360</RtAscension>
            <Eccentricity>.000238131</Eccentricity>
            <ArgPerigee>76.8442424</ArgPerigee>
            <MeanAnomoly>283.301720</MeanAnomoly>
            <MeanMotion theoryUsed="Kozai">14.24227259</MeanMotion>
            <EpochRevolution>16284</EpochRevolution>
        </OrbitalElements>
    </SatelliteElementSet>

</SpaceSurveillanceData>
```

D.10 tasking.xml

```
<?xml version="1.0"?>

<SpaceSurveillanceData      xmlns="urn:schemas-gov:SpaceSurveillance"
                            xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
                            xsi:schemaLocation=
                        "urn:schemas-gov:SpaceSurveillance
                         SpaceSurveillance.xsd">

<SensorTask>
    <TaskedSensor sensor="354" classLevel="U">
            <Controls classLevel="U">
                <TaskCreation>
                        <Date>2000-09-20</Date>
                        <Time>20:10:10.10</Time>
                </TaskCreation>
                <TaskValidPeriod>
                        <ValidStart>
```

```
                        <Date>2000-09-21</Date>
                        <Time>00:00:00</Time>
                    </ValidStart>
                    <ValidEnd>
                        <Date>2000-09-21</Date>
                        <Time>23:59:59</Time>
                    </ValidEnd>
                </TaskValidPeriod>
                <AssumedDownTime>
                    <DownTimeStart>
                        <Date>2000-09-21</Date>
                        <Time>12:00:00</Time>
                    </DownTimeStart>
                    <DownTimeEnd>
                        <Date>2000-09-21</Date>
                        <Time>15:00:00</Time>
                    </DownTimeEnd>
                </AssumedDownTime>
                <AssumedDownTime>
                    <DownTimeStart>
                        <Date>2000-09-21</Date>
                        <Time>17:00:00</Time>
                    </DownTimeStart>
                    <DownTimeEnd>
                        <Date>2000-09-21</Date>
                        <Time>19:00:00</Time>
                    </DownTimeEnd>
                </AssumedDownTime>
                <TaskCapacity>1000</TaskCapacity>
        </Controls>
        <TaskDaily classLevel="U">
        <Category priority="2" code="B">
                <Satno>10362</Satno><Satno>14693</Satno>
                <Satno>20399</Satno><Satno>23049</Satno>
                <Satno>23210</Satno><Satno>23817</Satno>
                <Satno>23941</Satno><Satno>26468</Satno>
                <Satno>26478</Satno><Satno>88258</Satno>
        </Category>
        <Category priority="2" code="C">
                <Satno>1328</Satno><Satno>7646</Satno>
                <Satno>16908</Satno><Satno>22076</Satno>
```

```
        <Satno>22824</Satno><Satno>25398</Satno>
</Category>
<Category priority="2" code="D">
        <Satno>16609</Satno><Satno>20436</Satno>
        <Satno>23342</Satno><Satno>23501</Satno>
        <Satno>23549</Satno><Satno>23605</Satno>
        <Satno>25063</Satno><Satno>25394</Satno>
        <Satno>25544</Satno><Satno>25674</Satno>
        <Satno>25860</Satno><Satno>25977</Satno>
        <Satno>25978</Satno><Satno>26032</Satno>
        <Satno>26040</Satno><Satno>26105</Satno>
        <Satno>26354</Satno>
</Category>
<Category priority="2" code="K">
        <Satno>23125</Satno><Satno>23126</Satno>
        <Satno>25938</Satno>
</Category>
<Category priority="2" code="O">
        <Satno>13736</Satno><Satno>18123</Satno>
        <Satno>20978</Satno><Satno>21798</Satno>
        <Satno>23233</Satno><Satno>23533</Satno>
        <Satno>24753</Satno><Satno>25991</Satno>
</Category>
<Category priority="3" code="D">
        <Satno>21987</Satno><Satno>22521</Satno>
        <Satno>22659</Satno><Satno>23757</Satno>
        <Satno>23857</Satno><Satno>23954</Satno>
        <Satno>24743</Satno><Satno>24779</Satno>
        <Satno>25064</Satno><Satno>25424</Satno>
        <Satno>25504</Satno><Satno>25616</Satno>
        <Satno>25722</Satno><Satno>25791</Satno>
</Category>
<Category priority="3" code="F">
        <Satno>12497</Satno><Satno>86724</Satno>
        <Satno>86860</Satno>
</Category>
<Category priority="3" code="G">
        <Satno>21223</Satno><Satno>22997</Satno>
</Category>
<Category priority="3" code="J">
        <Satno>2144</Satno><Satno>3597</Satno>
```

<Satno>3598</Satno><Satno>6153</Satno>
<Satno>6155</Satno><Satno>20453</Satno>
<Satno>24323</Satno>
</Category>
<Category priority="3" code="K">
<Satno>25023</Satno><Satno>25024</Satno>
<Satno>25025</Satno>
</Category>
</TaskDaily>
<TaskingMetrics classLevel="U">
<PercentOfMaxTracks>50.0</PercentOfMaxTracks>
<PercentOfNetworkLoad>5.5</PercentOfNetworkLoad>
<PercentResponseToTasking>66.7</PercentResponseToTasking>
</TaskingMetrics>
</TaskedSensor>
<TaskedSensor sensor="344" classLevel="U">
<TaskUpdate classLevel="U">
<Category priority="1" code="T">
<Satno>12547</Satno><Satno>26105</Satno>
</Category>
<Category priority="2" code="B">
<Satno>10362</Satno><Satno>14693</Satno>
<Satno>22067</Satno><Satno>23049</Satno>
<Satno>23210</Satno><Satno>23291</Satno>
<Satno>23817</Satno><Satno>23941</Satno>
<Satno>24032</Satno><Satno>25552</Satno>
<Satno>26468</Satno><Satno>26478</Satno>
<Satno>81739</Satno><Satno>86620</Satno>
<Satno>86721</Satno><Satno>88258</Satno>
</Category>
<Category priority="2" code="C">
<Satno>16908</Satno><Satno>18963</Satno>
<Satno>22076</Satno><Satno>22824</Satno>
<Satno>24646</Satno><Satno>25398</Satno>
<Satno>81447</Satno><Satno>81538</Satno>
<Satno>81622</Satno><Satno>87248</Satno>
</Category>
<Category priority="2" code="D">
<Satno>16609</Satno><Satno>20436</Satno>
<Satno>23342</Satno><Satno>23501</Satno>
<Satno>23549</Satno><Satno>23605</Satno>

```
        <Satno>25394</Satno><Satno>25544</Satno>
        <Satno>25674</Satno><Satno>25860</Satno>
        <Satno>25977</Satno><Satno>25978</Satno>
        <Satno>26032</Satno><Satno>26040</Satno>
        <Satno>26354</Satno>
</Category>
<Category priority="2" code="M">
        <Satno>86751</Satno>
</Category>
<Category priority="2" code="P">
        <Satno>7646</Satno><Satno>13736</Satno>
        <Satno>18123</Satno><Satno>20978</Satno>
        <Satno>21798</Satno><Satno>23233</Satno>
        <Satno>23533</Satno><Satno>24753</Satno>
        <Satno>25991</Satno>
</Category>
<Category priority="3" code="C">
        <Satno>831</Satno><Satno>7004</Satno>
        <Satno>24286</Satno><Satno>25942</Satno>
        <Satno>81640</Satno>
</Category>
<Category priority="3" code="D">
        <Satno>4166</Satno><Satno>19822</Satno>
        <Satno>20826</Satno><Satno>21430</Satno>
        <Satno>21574</Satno><Satno>21575</Satno>
        <Satno>21725</Satno><Satno>21840</Satno>
        <Satno>21867</Satno><Satno>22077</Satno>
        <Satno>22161</Satno><Satno>22782</Satno>
        <Satno>22825</Satno><Satno>22970</Satno>
        <Satno>23189</Satno><Satno>23317</Satno>
        <Satno>23323</Satno><Satno>23547</Satno>
        <Satno>23657</Satno><Satno>23752</Satno>
        <Satno>23851</Satno><Satno>23940</Satno>
        <Satno>24883</Satno><Satno>24920</Satno>
        <Satno>24925</Satno><Satno>24926</Satno>
        <Satno>25233</Satno><Satno>25395</Satno>
        <Satno>25396</Satno><Satno>25397</Satno>
        <Satno>25468</Satno><Satno>25560</Satno>
        <Satno>25568</Satno><Satno>25634</Satno>
        <Satno>25646</Satno><Satno>25730</Satno>
        <Satno>25731</Satno><Satno>25735</Satno>
```

<Satno>25758</Satno><Satno>26033</Satno>
<Satno>26061</Satno><Satno>26065</Satno>
<Satno>81412</Satno><Satno>81453</Satno>
<Satno>81454</Satno><Satno>81466</Satno>
<Satno>81471</Satno><Satno>81481</Satno>
<Satno>81491</Satno><Satno>81537</Satno>
<Satno>81583</Satno><Satno>81627</Satno>
<Satno>81631</Satno><Satno>81645</Satno>
<Satno>81660</Satno><Satno>81721</Satno>
<Satno>81732</Satno><Satno>81772</Satno>
<Satno>81780</Satno><Satno>81827</Satno>
<Satno>81833</Satno><Satno>81859</Satno>
<Satno>81885</Satno><Satno>81889</Satno>
<Satno>81902</Satno><Satno>81931</Satno>
<Satno>81954</Satno><Satno>81974</Satno>
<Satno>81976</Satno>
</Category>
<Category priority="3" code="H">
<Satno>3758</Satno><Satno>18807</Satno>
<Satno>81449</Satno><Satno>81542</Satno>
<Satno>81695</Satno><Satno>81729</Satno>
<Satno>81791</Satno><Satno>81884</Satno>
<Satno>81981</Satno><Satno>81986</Satno>
</Category>
<Category priority="3" code="M">
<Satno>899</Satno><Satno>10395</Satno>
<Satno>10705</Satno><Satno>12529</Satno>
<Satno>12854</Satno><Satno>13507</Satno>
<Satno>14475</Satno><Satno>15354</Satno>
<Satno>15538</Satno><Satno>18591</Satno>
<Satno>18745</Satno><Satno>20030</Satno>
<Satno>21570</Satno><Satno>21701</Satno>
<Satno>21936</Satno><Satno>21997</Satno>
<Satno>22384</Satno><Satno>22465</Satno>
<Satno>23821</Satno><Satno>23841</Satno>
<Satno>25467</Satno><Satno>25469</Satno>
<Satno>26252</Satno><Satno>81413</Satno>
<Satno>81427</Satno><Satno>81431</Satno>
<Satno>81439</Satno><Satno>81470</Satno>
<Satno>81474</Satno><Satno>81479</Satno>
<Satno>81483</Satno><Satno>81505</Satno>

<Satno>81508</Satno><Satno>81512</Satno>
<Satno>81522</Satno><Satno>81529</Satno>
<Satno>81530</Satno><Satno>81546</Satno>
<Satno>81547</Satno><Satno>81548</Satno>
<Satno>81572</Satno><Satno>81582</Satno>
<Satno>81609</Satno><Satno>81612</Satno>
<Satno>81618</Satno><Satno>81619</Satno>
<Satno>81647</Satno><Satno>81665</Satno>
<Satno>81679</Satno><Satno>81711</Satno>
<Satno>81716</Satno><Satno>81782</Satno>
<Satno>81784</Satno><Satno>81792</Satno>
<Satno>81816</Satno><Satno>81817</Satno>
<Satno>81821</Satno><Satno>81836</Satno>
<Satno>81844</Satno><Satno>81847</Satno>
<Satno>81848</Satno><Satno>81852</Satno>
<Satno>81856</Satno><Satno>81865</Satno>
<Satno>81886</Satno><Satno>81909</Satno>
<Satno>81932</Satno><Satno>81950</Satno>
<Satno>81967</Satno><Satno>81968</Satno>
<Satno>81969</Satno><Satno>81973</Satno>
<Satno>81980</Satno><Satno>81993</Satno>
<Satno>85069</Satno>
</Category>
<Category priority="3" code="P">
<Satno>21829</Satno><Satno>25018</Satno>
<Satno>81876</Satno><Satno>86724</Satno>
<Satno>86860</Satno>
</Category>
</TaskUpdate>
</TaskedSensor>
</SensorTask>
</SpaceSurveillanceData>

Appendix E

Space Surveillance Data Transformations Using XSLT

This appendix includes a description of the transformation language used to develop proof-of-concept transformations of Space Surveillance Ontology Extensible Markup Language (XML) schema and XML instance files.

E.1 What is XSLT?

The Extensible Stylesheet Language (XSL) provides a transformation language, as well as a formatting language. The transformation language, called XSL Transformation (XSLT), provides rules that enable the transformation of XML tagged data from one document to another. The XSL formatting language provides the rules for describing the presentation of the XML document content. The focus of this appendix is on XSLT and how it is used to provide selected views of the Space Surveillance Ontology.

In general, XSLT is used to transform a well-formed XML document into a user-defined format. The transformation addresses content selection, as well as presentation. The presentation can be targeted for either a user or an application. Capabilities include selecting, filtering, reordering, manipulation, and presentation of the information contained in one or more XML documents. Because XML is hierarchical, one can think of an XML document as a "tree." In XSLT, the source XML document is often referred to as the "source tree," and transformed document as the "result tree."

XSLT provides the syntax and semantics for navigating and manipulating the source tree in order to produce the result tree. The transformation rules can be captured in an XSL stylesheet and take the general form of a condition followed by, and enclosing, an action. An XSL enabled application compares the conditions defined in the stylesheet with the source tree and creates the result tree (see Figure E-1). Internet Explorer 5.0 (and later releases) is an example of an XSL-enabled application. By referencing the XSL stylesheet within the XML document, Internet Explorer applies the stylesheet to the XML document and presents the results.

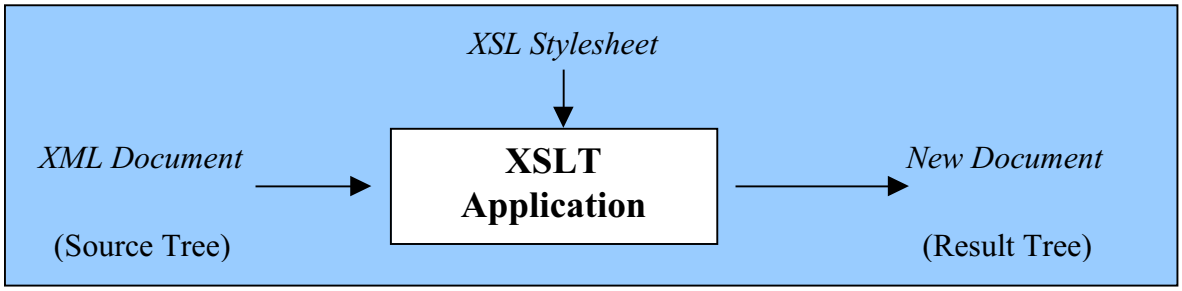


Figure E-1. XSL Transformation Process

XSL stylesheets can be used to transform both data and metadata. Typically, XSL is used to provide a tailored view of the source data. In a distributed architecture either the server or the client can perform this transformation. The only limitation is that the structure of the source document must be a well-formed XML document. XML Schemas are well-formed XML documents (i.e., use XML syntax), and the Space Surveillance Ontology is captured in an XML Schema. Therefore, XSL stylesheets can also be used to transform both XML tagged data, as well as the contents of the Space Surveillance Ontology. This provides the capability of producing multiple views of the metadata while maintaining a single metadata repository document.

E.2 Space Surveillance Schema Stylesheet Examples

This section includes two sample XSLT stylesheets. The first example uses a stylesheet to transform the XML schema document into a hypertext markup language (html) document. Note that currently the XSLT application used does not recognize the *.xsd* extension as a valid XML document extension. A workaround is to save the file with an *.xml* extension. It is our expectation that XML aware tools will quickly mature and will soon accept files with either an *.xsd or .xml* extension. The second example uses XSL to transform an XML instance document into a new format.

E.2.1 Capture Space Surveillance Ontology Definitions

Filename: Comments_to_html.xsl

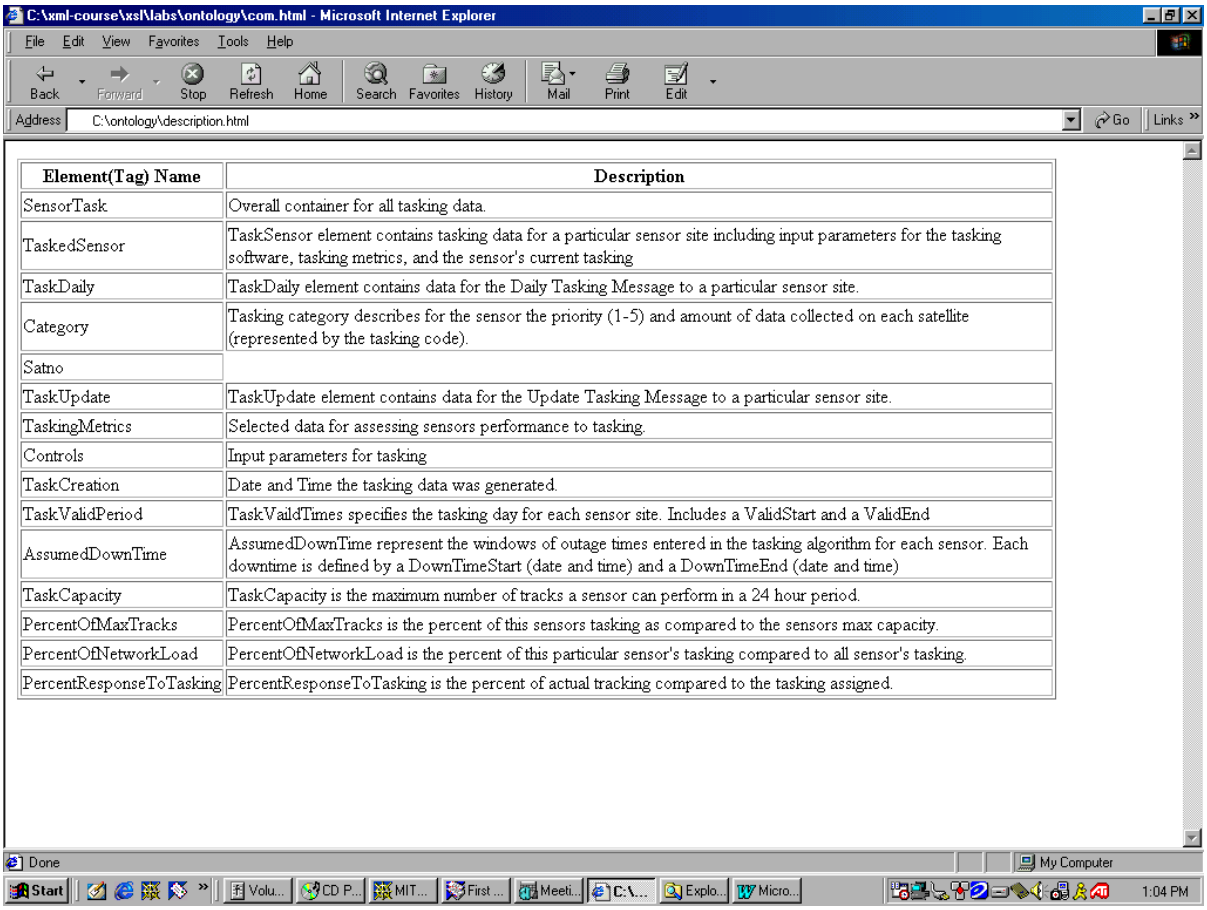
This stylesheet parses the element (or tag) name definitions and their associated descriptions from the schema document (*.xsd* file) and creates an html table. Figure E-2 contains a sample screen print of the output of the element definitions from the *SensorTasking.xsd* file.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">
  <xsl:output method="html" omit-xml-declaration="yes" />
  <xsl:template match="schema">
  <html>
    <body>
      <table BORDER="1" Width="90%">
        <th>Element(Tag) Name</th>
        <th>Description</th>
          <xsl:apply-templates select="element"/>
      </table>
    </body>
  </html>
```

```
        </xsl:template>

        <xsl:template match//element">
          <tr>
            <td><xsl:value-of select="./@name"/></td>
            <xsl:apply-templates select="annotation/documentation"/>
            <xsl:apply-templates select="complexType/annotation/documentation"/>
          </tr>
        </xsl:template>

        <xsl:template match="documentation">
          <td><xsl:value-of select="."/></td>
        </xsl:template>
</xsl:stylesheet>
```



Element(Tag) Name	Description
SensorTask	Overall container for all tasking data.
TaskedSensor	TaskSensor element contains tasking data for a particular sensor site including input parameters for the tasking software, tasking metrics, and the sensor's current tasking
TaskDaily	TaskDaily element contains data for the Daily Tasking Message to a particular sensor site.
Category	Tasking category describes for the sensor the priority (1-5) and amount of data collected on each satellite (represented by the tasking code).
Satno	
TaskUpdate	TaskUpdate element contains data for the Update Tasking Message to a particular sensor site.
TaskingMetrics	Selected data for assessing sensors performance to tasking.
Controls	Input parameters for tasking
TaskCreation	Date and Time the tasking data was generated.
TaskValidPeriod	TaskVaildTimes specifies the tasking day for each sensor site. Includes a ValidStart and a ValidEnd
AssumedDownTime	AssumedDownTime represent the windows of outage times entered in the tasking algorithm for each sensor. Each downtime is defined by a DownTimeStart (date and time) and a DownTimeEnd (date and time)
TaskCapacity	TaskCapacity is the maximum number of tracks a sensor can perform in a 24 hour period.
PercentOfMaxTracks	PercentOfMaxTracks is the percent of this sensors tasking as compared to the sensors max capacity.
PercentOfNetworkLoad	PercentOfNetworkLoad is the percent of this particular sensor's tasking compared to all sensor's tasking.
PercentResponseToTasking	PercentResponseToTasking is the percent of actual tracking compared to the tasking assigned.

Figure E-2. Sample Schema Definitions Screen Print

E.2.2 Create Tasking File

Filename: Tasking.xsl

This stylesheet parses the tasking.xml document, which conforms to SensorTasking.xsd, and builds a text file equivalent to the legacy tasking message format.

```
<?xml version="1.0"?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">

<xsl:output method="xml" omit-xml-declaration="yes" />

<xsl:template match="SensorTask">
<xsl:text>
BT

UNCLASS                         FOUO

SUBJECT:  SPADOC NOTIFICATION (U)

(U) REAL


1. (U) MESSAGE TYPE:  SENSOR TASKING

2. (U) PREPARATION DATE TIME:  010000ZJAN00


</xsl:text>
<xsl:apply-templates select="TaskedSensor" />
<xsl:text>
BT </xsl:text>
</xsl:template>

<xsl:template match="TaskedSensor">
<xsl:value-of select="@sensor"/>
<xsl:text> ASCENSION (1
```

(U) TASKING SUMMARY

```
</xsl:text>
    <xsl:apply-templates select="TaskDaily"/>
</xsl:template>

<xsl:template match="TaskDaily">
<xsl:apply-templates select="Category" />
</xsl:template>

<xsl:template match="Category">
   <xsl:text>CATEGORY </xsl:text>
   <xsl:value-of select="@priority"/>
   <xsl:value-of select="@code"/><xsl:text>

</xsl:text>
   <xsl:apply-templates select="Satno"/>
<xsl:text>


</xsl:text>
</xsl:template>

<xsl:template match="Satno">
   <xsl:value-of select="."/><xsl:text>/</xsl:text>
</xsl:template>

</xsl:stylesheet>
```

Glossary

ADC	Astronomical Data Center
AFCIC	Air Force Communications and Information Center
AFETR	Air Force Eastern Test Range
AFSCN	Air Force Satellite Control Network
AFWTR	Air Force Western Test Range
AIML	Astronomical IML
ALA	Al-Andar
API	Application Program Interface
BASDA	Business and Accounting Software Developers Association
C	Confidential
C2	Command and Control
CDE	Common Data Environment
CERES	Center for Research Support
CIS	Commonwealth of Independent States
COTS	Commercial off-the-Shelf
DII-AF	Defense Information Infrastructure Air Force
DII COE	Defense Information Infrastructure Common Operating Environment
DISA	Defense Information Systems Agency
DoD	Department of Defense
DOM	Documented Object Model
E	East
EFG	Earth-Fixed Geocentric
ELSET	Element Set
ESA	European Space Agency
ESC	Electronic Systems Center
FOUO	For Official Use Only
FRGUI	French Guiana
GLOB	Globalstar
GMTI	Ground Moving Target Indicator
GP	General Perturbations
GPS	Global Positioning System
GSFC	Goddard Space Flight Center

HGSTR	Hammaguira Space Track Range
HTML	Hypertext Markup Language
HTTP	Hyper Text Transfer Protocol
IBM	International Business Machines
IC2S	Integrated Command and Control System
ICD	Interface Control Document
IE5	Internet Explorer 5
IEEE	Institute of Electrical and Electronic Engineers
IML	Instrument Markup Language
IRs	Information Resources
ISC2	Integrated Space Command and Control
JBI	Joint Battlespace Infosphere
JSTARS	Joint Surveillance & Target Attack Radar System
JV2020	Joint Vision 2020
KSCUT	Kagoshima Space Center University of Tokyo
KYMTR	Kapustin Yar Missile and Space Complex
MOIE	Mission-Oriented Investigation and Experimentation
N	North
NASA	National Aeronautics and Space Administration
NOFORN	Not Releasable to Foreign Nationals
NRO	National Reconnaissance Office
N/UWSS	NORAD/USSPACECOM Warfighting Support System
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
ORCON	Originator Controlled
PAD	Product Area Directorate
PKMTR	Plesetsk Missile and Space Complex
PRC	Peoples Republic of China
PROP	Proprietary
PROPIN	Caution-Proprietary Information Involved
RDF	Resource Description Format
RELCAN	Releasable to Canada

S	Secret
S	South
SATC	Satellite Catalog
SAX	Simple API for XML
SCTMR	Shuangchenzi Missile Test Range
SGP	Simplified General Perturbations
SGP4	Simplified General Perturbations 4
SML	Spacecraft Markup Language
SNMLP	San Marco Launch Platform
SPADOC	Space Defense Operations Center
SPO	System Program Office
SRI	Sr. Harikota
SSN	Space Surveillance Network
SUO	Standard Upper Ontology
TNSTA	Tanegashima Space Center
TS	Top Secret
TS/TK	Top Secret Talent-Keyhole
TT&C	Telemetry, Tracking and Commanding
TTMTR	Tyuratam Missile Test Range
U	Unclassified
UAVs	Unmanned Aerial Vehicles
URI	Universal Resource Indicator
URL	Universal Resource Locator
URN	Universal Resource Name
USAF	United States Air Force
USSPACECOM	United States Space Command
USX	United States
W	West
W3C	World Wide Web Consortium
WOMRA	Woomera
WUZ	Wuzhai Launch Facility or Taiyuan
WWW	World Wide Web
XDF	EXtensible Data Format
XIC	Xi Chang Launch Facility
XML	Extensible Markup Language
XQL	Extensible Query Language
XSL	Extensible Stylesheet Language

XSLT	XSL Transformation
XSQL	Extensible Structured Query Language
YAVNE	Yavne Launch Facility