The SAOXML 5: New Format for Ionogram-Derived Data

Ivan A. Galkin, Grigori M. Khmyrov, Bodo W. Reinisch, and Jonathan McElroy

University of Massachusetts Lowell Center for Atmospheric Research
600 Suffolk Street, Lowell, MA 01854

Abstract. A new format for the unified data exchange between ionosonde data producers and users of ionogram-derived characteristics is introduced, dubbed SAOXML 5 to reflect its heritage in the previous Standard Archiving Output (SAO) format version 4 and a general-purpose computer language XML commonly used for data exchange. The SAOXML 5 specification shall serve as the reference for development of input and output interfaces for the software projects that read and write ionogram-derived data. The paper discusses motivation for introducing new format and outlines basic principles of its design and use.

Keywords: ionogram, ionospheric characteristics, data format
PACS: 93.85.Bc, 07.05.Kf, 94.05.Sd

INTRODUCTION

The Standard Archiving Output (SAO) format has been operational since 1987 in the capacity of the URSI-recommended standard for ionogram-derived data [1-2]. Current version 4.3 of the SAO format is used for delivery of the scaled ionospheric characteristics to the World Data Centers [3] and major operational space weather systems [4-6] from both unattended and supervised ionosonde observatories. As ionospheric conditions vary with time, and so does the amount of information collected and derived to describe them, the key requirement to the SAO design has been its ability to efficiently accommodate data elements of varying length and missing entries. With little support of these needed features in the mainstream scientific data formats at the time of its inception, the SAO design followed an independent development path to come up with a concept of human-readable, plain-text records of the varying length, with a lead-in index describing the rest of the content. Software libraries to read and write SAO records were made then available to the general public for incorporation in their software, and the international ionospheric data exchange was thus established.

As the need for new data items to be reported in SAO files comes up, the standard naturally undergoes appropriate modifications, and the network-wide update to the reading/writing software commences to keep up with the changes. It is the painstakingly slow process of upgrading software that inspired the SAO design team to consider new approaches to the task of storing ionogram-derived data so as to allow forward-compatibility of the standard in its next revisions.

MOTIVATION

Revision of the SAO 4.3 format began similarly to several previous revisions of the standard, with a need to store new data items for delivery to the end users. This time, however, proposed additions stressed the extensibility buffers envisaged in the SAO 4.3. Newly requested items were uncertainty boundaries for the autoscaled ionospheric characteristics and the electron density profile. These additions were lengthy, and lack of established practice in reporting them to the assimilation models made it difficult to quickly settle on one particular calculation technique and format. With this problem at hand, a generally good and common practice of providing data elements with appropriate descriptions (metadata) has been elevated to an operating requirement. While previously lack of metadata in the stored SAO records would merely make them less friendly to an outsider, now it could as well make them ambiguous and therefore incomplete.

PHILOSOPHY OF NEW SAO

Several important design concepts were formulated in order for the new SAO format to withstand time. Considering the expected amount of work needed to implement software changes at the organizations that will use the new format, a thorough peer review process was initiated, and contributions to the SAO have been made from many members of the ionospheric physics community.

Extensibility

While emerging ionospheric research brings new inspiration to the ranks of scientists, it also poses difficult technical challenges to the data engineers. Interoperability of the data repositories has been a tough requirement as it is, considering the variety of ionosonde brands operating worldwide. New observations that modern ionosondes can provide unavoidably call for extensions to the nomenclature of the ionogram-derived information. The necessary evil of changing the standard to accommodate such new applications triggers an avalanche of software compatibility issues across the network of end users.

In order to ensure completeness of the format across different ionosonde models and with time and progress in the ionospheric research, a mechanism for extending the format has to be established. Here format extensibility relates to the ability of adding new data elements with a minimal level of effort and no side effects on existing data elements. Related to the format extensibility is an important concept of forward compatibility of software.

Forward compatibility

Existing SAO reading software that encounters data records containing new, previously unknown data elements (either custom or newly introduced in the future SAO revisions) shall continue to work within the scope of their original design. Such ability is known as forward compatibility, acceptance of input data intended for future versions of software. In order to be forward compatible, the format organization shall support extensibility (i.e., allow new elements to be added without affecting existing elements) and admit the operation of skipping or ignoring unknown elements. If SAO reading software is not forward compatible, any change to the format specification incurs software redesign.

User friendliness

The format shall be user-friendly in various modes of operation, from software development to troubleshooting. The following list of requirements is commonly suggested as the guidelines for user friendliness:

Readability: Key components of the data shall be easily identifiable in the data without the need to match data records with the external format description.

Self-descriptiveness: Proper metadata shall be provided together with the data to explain properties of the stored information.

Clarity of presentation: Names, data types, units shall be clear, precise, not abbreviated, and helpful in understanding of the data contents.

One ionogram – many ionogram-derived records

As more than one scaler can interpret the same ionogram, the relationship between recorded ionograms and sets of available scaled and derived ionospheric characteristics is “one to many”. Three possible scenarios of handling this relationship in software are possible, listed in Table 1.

TABLE 1. Storage of Multiple Scaled Records Related to the Same Ionogram.

Method Name	Description
Updated storage	newly scaled data replace existing data in the same SAO record
Combined storage	newly scaled data are added to the same SAO record
Separate storage	one SAO record stores data derived by one scaler only

Let’s consider a typical situation that arises when a human scaler discovers a wrongly autoscaled foF2 value and wishes to correct it. In the updated storage approach, existing foF2 value is replaced by a new one. For combined storage, manually scaled foF2 value is added as a new entry to the existing SAO record, preserving the [wrong] autoscaled foF2 value. In the separate storage arrangement, a completely new record is created holding manually scaled foF2.

At first glance, updated or combined storage shall simplify data management as they keep all relevant data in one record, and therefore at a single point of access. However, there are subtle considerations that led us to accept the separate storage approach.

Patching ionogram interpretation provided by one scaler with supposedly better interpretation from another scaler should not be allowed, as this constitutes an irreplaceable data loss. The combined storage scenario is free from the data loss problem; however, it imposes a great technical demand on the software. Keeping an accurate account of changes made by multiple scalers inside one SAO record makes it a miniature database on its own, where each reported data element has to be linked to an appropriate description of the scaler responsible for its creation. While such architecture looks interesting and powerful, both SAO reading and writing software has to be written well to manage associated referential integrity of the document during operations of element addition and removal. It also unnecessarily complicates otherwise simple task of reading one SAO record by addition of algorithms that select one representative item out of multiple available alternatives provided by other scalers. These functions are netter taken care of by professional-strength database management systems.

Our choice of the separate storage approach has two important consequences for the SAO architecture. First, SAO record carries only one scaler description attribute that applies to the whole record. Second, our design warrants storage of the unprocessed (raw) ionogram data separately from the SAO in order to avoid duplicate storage of the same voluminous information in the multiple records submitted by different scalers. The proposed new SAO format does not include raw ionosonde sounding data.

Storage by column

Scaled ionogram traces, as well as derived ionospheric profiles, admit a natural presentation in the tabular form (Table 2):

TABLE 2. Sample Ionogram Trace Stored in Tabular Form

Frequency kHz	Virtual Height, km	Doppler shift, Hz	Amplitude dB
2.0	326	-0.098	130
2.1	335	0.098	122
2.2	350	0.293	126
2.3	366	0.293	134

In order to fulfill the forward compatibility requirement, this proposal calls for arrangement of the tabular data storage by column, thus holding only homogeneous values within one data element. In the example given in Table 2, storage by column means the following sequence of data items:

Frequency: 2.0, 2.1, 2.2, 2.3
Virtual height: 326, 335, 350, 366
Doppler shift: -0.098, 0.098, 0.293, 0.293
Amplitude: 130, 122, 126, 134

In this case it becomes considerably simpler to build software readers that can handle ionospheric data of varying richness of their contents. Unknown characteristics of traces and profiles can be skipped as a whole entity. Skipping unknown data items in other models of data presentation (such as storage by rows or by tables) requires tokenizing and identification of known/unknown token positions for every row of the stored table. The same data of Table 2 presented in the storage-by-row approach are:

Trace point #1: 2.0, 326, -0.098, 130
Trace point #2: 2.1, 335, 0.098, 122
Trace point #3: 2.2, 350, 0.293, 126
Trace point #4: 2.3, 366, 0.293, 134

Let's suppose that the Doppler frequency shift is a new element introduced to the SAO standard. Reading tabulated data by row will require the old software to count tokens and discard every 3rd token. If more than one column is unknown to the reader, it has to keep a list of unknown columns to discard. However, if data are arranged by column, unknown column elements can be easily ignored as a whole.

Design simplicity

Where possible, simpler solutions are preferred. Particular objection has been raised against internal and external links between data elements. Management of intra-element links incurs additional effort of the referential integrity preservation through operations of addition and deletion. Need for advanced software engineering shall be avoided by adopting simpler architecture of the SAO record.

SAO DATA CONTENTS

New SAO design takes one important step above existing standard in allowing new data elements to be reported to cover immediate needs of the assimilation models. While other developments are in the works that shall bring new types of geophysical observations and techniques for ionogram data processing, we emphasize the need to provide standardization solutions for the task at hand. The SAO extensibility and forward compatibility will ensure that no restrictions are imposed that would limit ability of the new design to adapt to the evolving discoveries and developments. With this in mind, the SAO version 5 data contents given in Table 3 will look conventional to ionospheric data providers and users.

TABLE 3. SAO Data Contents

Item	Comments
SAO Record Attributes	Measurement time, location, content descriptors
System Information	Additional information on measurement location and equipment
Characteristics	Scaled characteristics with qualifiers, descriptors, error bars
Traces	Traces of ionospheric reflections extracted from the ionogram image

Profiles	Height profiles of geophysical data derived from ionograms, e.g., electron density profile.

The concept of *elements* and *attributes* will be used throughout the rest of the text. Elements and attributes play a major role in defining contents of hierarchical data structures such as the SAO; they were introduced by IBM in the 1960s to create documents that needed to remain machine-readable for several decades, a very long time in the information technology. Modern simplified versions of the original IBM design, HTML and XML markup languages, have outgrown their parent in terms of worldwide acceptance and support.

The element is a data structure with well-defined start and end, some content, and attributes with additional properties of the element (Fig.1). The content of an element can be simple or complex (consisting of other sub-elements). Well-defined start and end of the elements makes them easy to skip without knowledge of internal structure and format, thus solving the problem of forward-compatibility.

Figure 2 provides a more detailed diagram of the internal SAO Record structure. In the diagram, individual data elements are shown as boxes. When an element can exist in multiple instances (e.g., more than one trace can be found on ionogram), it is shown as a box triplet. SAO elements whose content is an array of multiple items of the same kind use "List" suffix in their name.

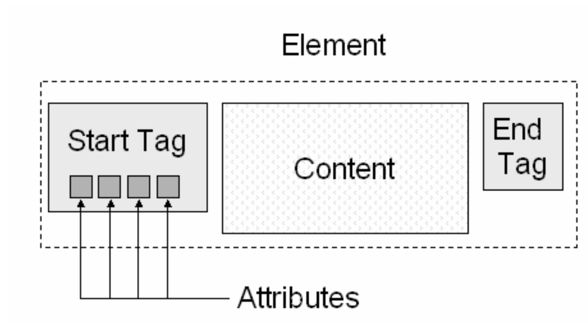

FIGURE 1. Element and its attributes.

The top level SAORecord element, its attributes, and CharacteristicList element with multiple URSI elements are shown with bold outlines as they constitute the minimum set that one SAORecord has to contain. Other data elements are optional.

We will continue description of the new SAO structure after introduction of the underlying language that we selected to hold SAO data. Then it will be easier to illustrate the structure with examples.

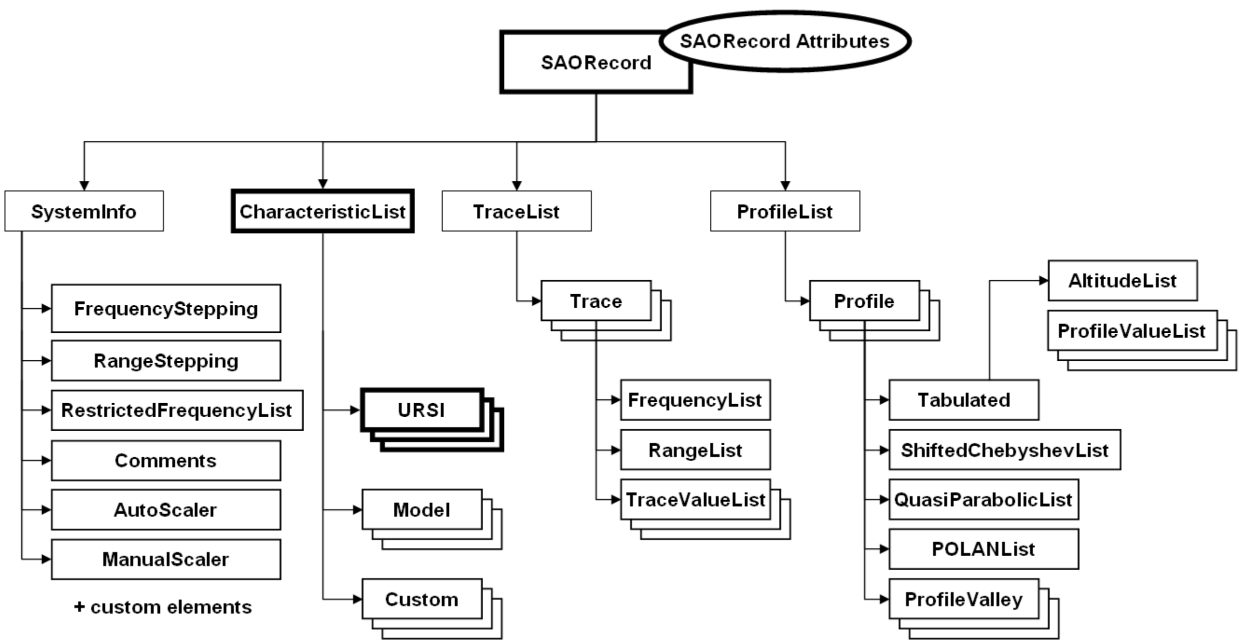

FIGURE 2. Structure of SAO version 5.

XML: NEW VEHICLE FOR SAO

We found the eXtensible Markup Language (XML) to be a good choice of the new SAO vehicle that meets the SAO design requirements discussed so far. An XML document is a plain text file that is quite human-readable. Figure 3 presents an example of the SAOXML record that contains the minimum required elements and attributes to report one foF2 value.

```
<?xml version="1.0"?>
<SAORecordList>
  <SAORecord
    FormatVersion="5.0"
    StartTimeUTC="2005-10-06 -279 20:11:12.000"
    URSICode="MHJ45"
    StationName="Millstone Hill"
    GeoLatitude="42.6"
    GeoLongitude="288.5"
    Source="Ionosonde"
    SourceType="DPS-4"
    ScalerType="auto"
    >
<CharacteristicList Num="2" >
<URSI ID="00" Name="foF2" Val="6.225" />
<URSI ID="20" Name="foE" Val="2.41" />
</CharacteristicList>
  </SAORecord>
</SAORecordList>
```

FIGURE 3. Sample SAOXML file holding the minimum-required data to report foF2 and foE values.

For comparison, Fig. 4 shows an SAO 4.3 record holding the same amount of information. While the amount of auxiliary information in Fig. 3 and Fig. 4 is not quite the same, the difference in terms of data readability is quite clear.

It is also evident that SAOXML readability is accomplished at the expense of significant additions of various markup tags. One common criticism of the SAOXML proposal was the inefficiency of data storage because of the descriptive tags that accompany values. While comparisons to the SAO 4.3 format can be made showing that SAOXML 5 offers several volume savings ideas, the ultimate argument is that the overall SAO data production from the worldwide ionosonde network is miniscule in comparison with what modern science data warehouses are handling. With a typical 15 minute schedule of observations, one digisonde produces ~200 MB of SAO files a year. Direct conversion of the SAO 4.3 content to SAOXML 5 doubles the size, which is still below the noise level for a consumer grade PC workstation server, let alone a professional strength resident archive or a data center. The inherent space-inefficiency of XML documents is not an important issue for ionogram-derived data.

The readability of SAOXML records is additionally strengthened by the commonly available software support for XML-formatted data. Regular Internet browsers have an intelligent parser for the XML documents that is useful to quickly get familiarized with their hierarchical structure.

PROGRAMMING SAOXML

While human readability is very important for SAOXML to be readily accepted outside of the expert circle, so is its machine readability. Building a robust SAOXML reader is not quite straightforward; it is important to use available standard XML API for this task.

There are two primary approaches to reading of XML, (1) using the object graph (DOM) and (2) using a stream parser (SAX). Both DOM and SAX libraries are available for many programming languages. While writing an SAOXML programmer guide is beyond the scope of this paper, the following comparison of old and new SAO reading procedures shall help application programmers to deal with the SAOXML.

Simple example shown in Fig.3 and Fig. 4 helps to understand the subtle difference between reading old and new SAO. Varying length of reported items was accommodated in the SAO 4.3 by preceding the data section with an 80-element index holding item counts. Each non-zero count would trigger an appropriate reading function (see Fig. 5). Index counts are tested for non-zero values one by one, in a linear sequence of conditional statements.

```
  5  1 19  9  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4
  1.400 71.000 42.600288.500 24.000
DPS-4 042/MHJ45, NAME Millstone Hill, ARTIST 1103 |
AA20052791006201112
   6.225 9999.0009999.000 9999.0009999.0009999.0009999.0009999.000   2.410
```

FIGURE 4. SAO 4.3 minimum-required record holding the same foF2 and foE values as example in Fig. 3.

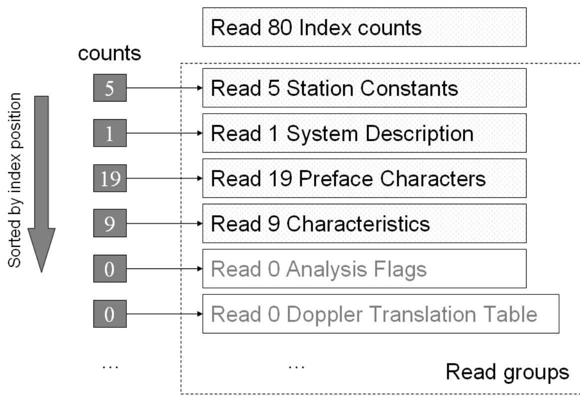

FIGURE 5. Logic diagram for SAO 4.3 reading procedure as it scans the sample file in Fig. 4.

In the stream parsing approach to XML, the record parser triggers appropriate reading functions, just like in SAO 4.3, but there is no pre-defined index, and parser uses the start tag names instead. Figure 6 shows the simplified logic diagram that would be sufficient to read example in Fig.3. As the parser detects start tags in the SAOXML record, it calls startElement() function provided in the standard SAX library. The startElement() function shall be written to recognize familiar tag names "SAORecord" and "URSI" and forwards tag attributes to appropriate user-specific reading functions. The user-specific functions look at each attribute in the provided list to find known values and populate application data structures accordingly.

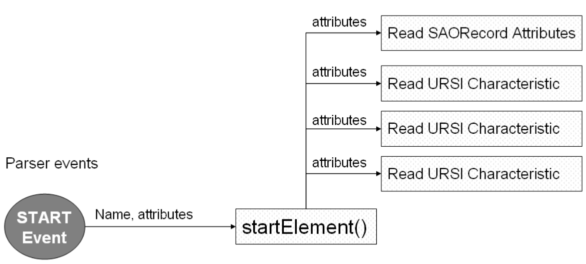

FIGURE 6. Simplified logic diagram for SAOXML reader as it parses the sample file in Fig. 3. Callback function startElement() is provided in the standard SAX library.

Simplicity of the parser logic is the main argument why we prefer to store data values in the tag attributes, not in the body of the element. There are more complex data elements stored in the SAOXML record (see Fig. 2), such as the traces and profiles, that require the element body to be retrieved, and more careful control of the nested elements. Figure 7 shows a more complete logic diagram for SAOXML parser that calls startElement(), endElement(), and characters() functions, recurrently, as it detects start and end tags and body content between them. The custom code for these three functions and the state machine that keeps track of the nested levels of SAOXML hierarchy is provided by UMLCAR, and the user-specific reading function are given as stubs to be personalized for each user application.

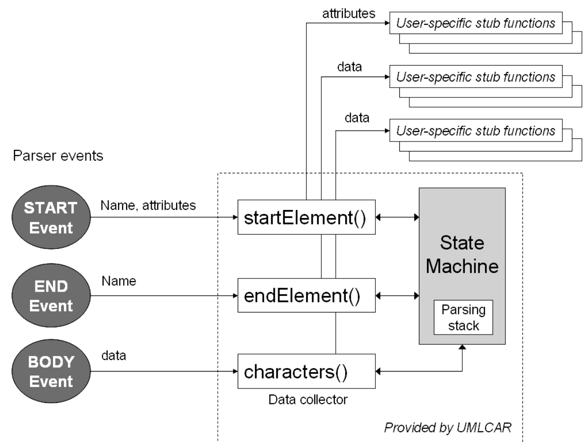

FIGURE 7. Logic diagram for SAOXML reader.

TRACES AND PROFILES

A greater degree of flexibility has to be involved in storage of the ionogram traces and ionospheric profiles whose content can be diverse, depending on the capability of the sounder and analysis software. In order to provide for such flexibility, we use the same idea of replacing enumerated types of information with named elements and attributes. The Trace element in Fig.2 has attributes Type, Layer, and Polarization to describe its contents. The Profile element has Tabulated presentation where ProfileValueList attributes define which physical quantity (electron density, drift velocity, etc) is reported. SAOXML structure is discussed at a greater detail in [7].

SAOXML EXTENSIBILITY

Logic diagrams in Fig. 5 and 6 illustrate well the extensibility and forward compatibility of the SAOXML design: only those parser events will eventually reach reading functions that are known to the reader, the rest of events will be bypassed. This feature can be compared to SAO 4.3 design in Fig. 4, where bypassing an unknown group requires knowledge of its internal organization (data item size or number of items per line). The important point here is that SAOXML extensibility and forward compatibility should not mean complete freedom of new element additions each time such need arises, even though technically it is possible. A control mechanism for additions to the SAOXML contents must be in place.

Version Control

The best way to maintain control over additions to the SAOXML is called Document Type Definition (DTD), a legal definition of XML document structure with a list of allowed elements, their attributes, and whether they are required/optional and allowed multiple instances within one record. Provision of the formal DTD proved to be most useful for building SAOXML writers, because it allows one to test created documents for compliance with the DTD. It is therefore imperative to keep master copy of the SAOXML DTD up to date with suggested extensions. Current DTD version 5.0.1e is available at http://ulcar.uml.edu/SAOXML/, further version control is expected to be transferred under management of the URSI Ionosonde Network Advisory Group.

Other Applications

SAOXML 5 can serve as a data exchange format for other sources of ionospheric measurements or modeling results. It is successfully used at UMLCAR for storage of data from incoherent scatter radars and the IRI. Presentation of the data from various sources in the same format simplifies development of software for data analysis and visualization.

SUMMARY

A new format for the unified data exchange between ionosonde data producers and users of ionogram-derived characteristics has been introduced, dubbed SAOXML 5 to reflect its heritage in the previous Standard Archiving Output (SAO) format version 4 and a general-purpose computer language XML commonly used for data exchange. The SAOXML 5 specification shall serve as the reference for development of input and output interfaces for the software projects that read and write ionogram-derived data. SAOXML 5 extensibility, forward compatibility, and commonly available software support shall be helpful in accepting it as new standard and implementing in the existing and new software applications.

ACKNOWLEDGMENTS

The authors thankfully acknowledge contributions to the SAOXML 5 development from many ionospheric data users and producers. We appreciate design suggestions from Dr. Terry Bullett of the US Air Force Research Laboratory, Mr. Rob Redmon and Ray Conkright of NOAA National Geophysical Data Center, Dr. Richard Stamper of United Kingdom Space Science Data Center, and Dr. Iwona Stanislawska of Space Research Center, Polish Academy of Sciences. We are indebted to Dr. Lee-Anne McKinnell of Space Physics Group at Rhodes University, RSA, and Dr. Ioanna Tsagouri of National Observatory of Athens/DIAS, for testing SAOXML in their field operations.

REFERENCES

1. R. R. Gamache, T. W. Bullett, Z. M. Zhang, B. W. Reinisch, and W. T. Kersey, ARTIST Data Editing and Printing Output Format - a Flexible Modular Database, in *Digital Ionogram Data Formats for World Data Center Archiving*, Proceedings of IIWG International Workshop, University of Lowell Center for Atmospheric Research, November 1989.
2. B.W. Reinisch, SAO (Standard ADEP Output): Format For Ionogram Scaled Data Archiving, *INAG Bulletin No. 62*, January 1998. Also available at http://ulcar.uml.edu/~iag/SAO-4.3.htm
3. R.O. Conkright, SPIDR on the Web: Space Physics Interactive Data Resource Online Analysis Tool, *Proc. URSI XXVI GA*, Toronto, Canada, 1999, p. 491.
4. J. Buchau, T. W. Bullett, A.E. Ronn, K.D. Scro, and J.L. Carson, The digital ionospheric sounding system network of the US Air Force Air Weather Service, *Ionosonde Networks and Stations, UAG-104*, WDC-A for STP, Boulder, CO, 16-20, (1995).
5. R. Stamper, J. Lilenstein, and N. Jakowski, Nowcasting, forcasting, and warning for ionospheric propagation: supporting databases, *Annals of Geophysics,* **42**, Supplement N2, COST 271 Action, 33-70, (2004).
6. R.S. Gardiner-Garden, Ionospheric variability in sounding data from JORN, *Proceedings of WARS 2006,* Leura, NSW, Australia, 15-17 February 2006.
7. *SAOXML 5 Format Specification*, (2007) Retrieved on September 23, 2007, from University of Massachusetts Lowell Center for Atmospheric Research website http://ulcar.uml.edu/SAOXML/