OPEN NAVIGATION SURFACE WORKING GROUP

MEETING SUMMARY

Inaugural Meeting, 21-23 January 2004
CCOM/JHC, Durham NH

DRAFT VERSION
January 28, 2004

1 Introduction

1.1 The Navigation Surface Concept

The Navigation Surface [1, 2] is a concept for re-engineering of the workflow associated with processing hydrographic data sources. In essence, it is an argument for using a grid data product to represent the bathymetric part of a hydrographic survey, which grid should include at least an estimate of the depth of the seafloor in a particular region, and an estimate of the uncertainty of this depth, along with the meta-data that allows this information to be used. The grid product is constructed in such a way that it is equivalent in terms of safety of navigation to the traditional 'points and lines' product currently used (i.e., the traditional 'smooth-sheet', also known as a 'field-sheet' or 'fair-sheet'), and hence that the grid may be certified as the legal source of the bathymetric information for the construction of nautical charts.

1.2 The Open Navigation Surface

The Open Navigation Surface (ONS) effort started as a community-led development with the aim of developing a file format that could be used to store information contained in a Navigation Surface data object. This file format was conceived explicitly as an open format, so that multiple vendors could utilize the same information without having to re-organize it, or construct import and export filters. This has the important additional goal of allowing users to adopt the technology more widely and have suitable confidence that investment in tools to manipulate the format will not be wasted later.

The ONS Working Group (ONSWG) was formed from the respondents to an e-mail message soliciting statements of interest from the community known to be interested in the topic. Initial membership consisted of representatives from the National Oceanographic and Atmospheric Administration (NOAA) and the Naval Oceanographic Office (NAVO), Science Applications International Corporation (SAIC), Interactive Visualization Systems (IVS), CARIS Ltd., 7Cs GmbH, and the Center for Coastal and Ocean Mapping and NOAA/UNH Joint Hydrographic Center (CCOM/JHC) at the University of New Hampshire (where the Navigation Surface concept was developed).

In order to aid adoption of the format and its wide dissemination, the members of the ONSWG agreed that the format should be represented in concrete form, meaning that a source-code library for the format would have to be defined. Work on this started via e-mail lists from mid-December 2003, and the ONSWG agreed to convene for a workshop early in 2004. This workshop was held from 21-23 January 2004 at CCOM/JHC's Chase Ocean Engineering Lab (Durham, NH).

1.3 Goals of the Working Group

The immediate goals of the Working Group are:

1. To define the requirements for an Open Navigation Surface data object, specifying all of the mandatory data elements required.
2. To determine suitable software technologies to implement these requirements.
3. To develop one or more source-code libraries that form a reference model for reading and writing data in this format, and provide support and guidance to potential users of the format.

The long-term goals of the Working Group are to promote continual improvement of the format, evaluate requests for modification as required, and encourage adoption of the format by vendors and users alike.

1.4 Objectives of this Meeting

The broad objectives of the meeting were to define the requirements for the ONS data elements (including a unique name for the element), and to start development of the software library to accompany the format.

2 Requirements

2.1 Distinction between the 'Navigation Surface' Processes and Data

It was agreed that we need to draw a clear distinction between the generic term 'Navigation Surface' and the data and procedural components that it encapsulated. The most commonly asked question about the Navigation Surface is 'what is it, exactly?' and it appeared to be too complex to describe it in terms of a Object Oriented programming model (i.e., as a chunk of data with associated methods for its use) although this is the clearest description of the intent.

It is clear that the most basic elements are the constituent grids of bathymetric data that are used to make composite grids suitable for navigational use. However, both source grids at the highest resolution that is supported by the survey data, and the product grids (made from suitable combination of source grids, including defocusing for horizontal error and generalization to a particular product scale) have the same requirements: both are a regularly spaced grid of depth and uncertainty pairs, plus, potentially, some auxiliary information about the grids or the hydrography of the area, and some meta-data. Therefore, the group agreed to reserve 'Navigation Surface' for a product grid (i.e., one that is certified as 'For Navigational Use'), and to use 'Bathymetric Attributed Grid' (BAG) to describe the individual grid objects at any level of detail and resolution. This name is therefore also adopted for the file format itself. The BAG format does not require the use or application of any specific algorithms, but offers processing traceability by providing mechanisms for documenting the name, version, originator, operator, and nature of any algorithmic operations applied to the data.

2.2 Meta-Data Information

It was agreed that meta-data is integral to the utility of the data, and (after some debate) that it should be kept with the data itself, rather than as a separate file. Otherwise, it would be too easy for the meta-data and data to become separated. A number of meta-data standards were discussed, including FGDC CSDGM 2.0 (http://www.fgdc.gov) and ISO19115. Since ISO19115 has been adopted by ANSI, the group agreed that it was the preferred content standard.

However, ISO19115 only defines the required contents of the meta-data, and not how the information is encoded. It was agreed that an XML encoding would be most appropriate, since freely available code libraries readily parse XML, even while it remains in human-readable form. ISO19139 defines the XML encoding of ISO19115, and the group agreed to adopt this standard for meta-data in the BAG format.

The encoding of ISO19139 is voluminous, and not all of the tags are required. The group agreed to adopt as mandatory all of the tags mandated by ISO19115, and identified a number of additional Meta-Data fields that would be considered mandatory for the BAG format. Determining which tags should be mandatory and which suggested is no small feat, and remains an on-going task for the meta-data sub-group of ONSWG. This information is being encoded in an XML schema file, which contains both the tags and the documentation of those tags. In addition, the schema can be used to verify the encoding of XML example files, which may be adopted as a requirement before accepting meta-data for inclusion in a BAG file.

A number of issues concerning whether the format should detail limitations on the provenance of data in the file, and whether there should be restrictions on the type of data that should go into the file. In all cases, the answer was that the requirement is only that the data in the file be certified to a particular level by a competent authority; the provenance of the data can be determined by the contents of the meta-data elements in the file, and the authenticity of the provenance can be determined by the digital signature associated with the file (q.v.) The format does not, however, guarantee or control the meta-data, and the group agreed that it is the user's responsibility (or the responsibility of the user-level code) to ensure that appropriate meta-data is entered whenever the data is modified.

2.3 Required Data Components

The group agreed that the data in a BAG would be represented as a regular grid at a single resolution, and that (at least initially) there would be only a single geographical grid in each file. Each grid will have regular spacing on each dimension (although not necessarily equal on each dimension), and the group agreed to allow either unprojected (geographic) grids, or grids in a limited number of projections. For unprojected grids, units of decimal degrees will be used; for projected grids, units of meters will be used. In all cases, meters will be used for depths and elevations; the group agreed to adopt a right-handed

coordinate system with reference point in the southwest corner of the grid, co-incident with the location of the data sample. This choice implies that increasing value in the grid implies motion away from the center of mass of the earth (i.e., 'up'), and hence all values in the grid are elevations, so that all values below datum (i.e., depths) are negative. (Other axiomatic definitions regarding time, etc., were decided, and are outlined in the Format Specification Document).

The required data components consist of a grid of elevations, and a co-located grid of uncertainties of the same size, etc. The meaning of 'uncertainty' was not defined by the group, and indeed may change through the life of a grid from source (where its intent is to inform the compilation of the product) to the product (where its intent is to inform the user about the reliability of the information in the grid). Note that while the specific definition for the basis of the uncertainty was not defined during the workshop, this will be addressed as ONSWG efforts continue. In addition, the group adopted the idea of a 'change list' to encapsulate the NS idea of 'designated soundings' (i.e., a sounding chosen by a hydrographer as the 'truth' about the depth in the area). The group agreed that the data presented in gridded form should be that with the designated soundings applied; for each such modification, the position of the modified grid node (in row, column space), the original depth and uncertainty, and a reason code indicating why the change was made are to be written into the change list data element. In this way, it is possible to determine where changes have been made, for what reason, and how to reverse them if required, while still ensuring that unless some action is taken, the value read from the depth grid is the 'safe' depth determined after inspection and verification by the certifying authority.

2.4 Extension Data Components

The group reviewed the tradeoffs of a purely exchange based format versus a format that could be suitable as an internal format for software products. While an exchange format would be simpler to define and more straightforward to control changes to, it would have the undesirable effect of potentially requiring format conversion in and out of the various software tools. The clear priority was placed on achieving (at least) an export format, but a high level of interest in achieving a format that would allow for interoperability between software tools was identified. In order for the format to be considered for internal use, some facilities would need to be provided to allow for extensions beyond the mandated data elements. The need for both node-based (gridded) and random point data extensions were discussed, with group consensus that the primary need would be for node-based extensions.

Use of proprietary extensions (i.e., where after a tag value, the contents of the data block would be known only to the owner of the block specification) was discussed but rejected as a means to ensure that the format remained open. Instead, the group agreed that extension blocks would be adopted into the standard at the request of any party, after review by an ONS Advisory Board (q.v.), given that the contents of the extension were specified. Procedures for making such a request are to be defined in the Format Specification Document (q.v.). The group decided that, in the interests of simplicity, no optional blocks would be defined at the first iteration of format specification.

2.5 Certification and Digital Signature

Data in the BAG file will contain information that can potentially be used for navigation. Clearly, some method is required to ensure that the data is certified by some competent authority, and that it has not been modified since that certification was made. The group agreed that certification should take place by adding an element to the meta-data (e.g., "2004-01-25. Data in this file examined by CDR J. Doe, Chief of Party and found to meet required standards per Standing Instructions. This data is certified to be complete and of appropriate quality for use in construction of Navigation Surfaces."), and that integrity of the data should be ensured by an appropriate secure hash, with the identity of the signatory being determined by a Digital Signature algorithm. This method ensures that if any modification is made to the file, the secure hash is automatically outdated, with no user intervention, and that there is a way to trace the chain of custody of the data from generation to point of use.

There are a number of different ways to achieve these ends, but cryptographic algorithms are not a playground for the unwary. The group agreed to adopt current standard methods, using FIPS 180-2 (Secure Hash Standard) for message integrity checks, and FIPS 186-2 (Digital Signature Standard) for identity confirmation.

2.6 Documentation

The format will be defined by a series of at least three documents:

- "Requirements for the Open Navigation Surface's BAG File Format"
- "Format Specification for the BAG File Format"
- "Application Programmer's Interface for the BAG File Format"

All three documents have been outlined during the workshop, but are, and will continue to be, works in progress throughout the lifetime of the project.

2.7 Advisory Board

The group agreed that there would be a requirement for a group of people to consider suggestions for improvement to the format, and to accept bug-fixes; the requirements are defined in the Format Specification Document. It was agreed that selection of the Board should be deferred until a beta release of the source code library is available.

2.8 Platform Independence

The group agreed that the BAG format and software access library need to be supported on multiple operating systems and computer architectures. The initial focus will include support for the Microsoft Win32 platform, and for Red Hat Linux 7.1 & 7.3 (gcc 2.95 & gcc 2.96). Other platforms discussed include: Macintosh, Sun, SGI, and HP-UX.

3 Implementation

3.1 Data Encapsulation Layer

The ONSWG considered several alternatives for an existing data encapsulation technology. Included were Tagged Data Format (TDF) from IVS (http://www.ivs.unb.ca/), NetCDF from the Unidata Program Center in Boulder, Colorado (http://www.unidata.ucar.edu/packages/netcdf/), and HDF5 from the National Center For Supercomputing Applications (http://hdf.ncsa.uiuc.edu/HDF5/). Important considerations in the selection process included: open unrestricted format, file access speed, platform independence (data and software), current and continuing software support, large file support, and growth potential. After considerable review and discussion the ONSWG agreed to pursue use of HDF5 for the data encapsulation and low-level file I/O. Initial efforts will be based on HDF5 version 5-1.6.1, which is the most recent official release as of January, 2004.

While it is expected that initial versions of the BAG format and access library will make use of only a limited number of the features available in HDF5, selection of HDF5 now provides a foundation that will allow for growth potential in the future. As an example, the group agreed that initial versions would support a single spatial resolution in a BAG. Existing features in HDF5 will allow for growth to support multiple spatial resolutions in a BAG. ONSWG consensus is that the existing feature set of HDF5 outweighs the complexity inherent in adopting HDF5.

3.2 File Format and Support Library

The group agreed that a software access library should be developed hand-in-hand with the development of the BAG format definition. The software access library will be an open library that is maintained with and lives with the BAG format definition. The intent of the open software access library is to ensure that the low level software for reading and writing BAG format data need only be developed and maintained once and is shared among recognized parties. The access library will provide a set of application program interfaces (APIs) that provide all the necessary functionality for reading data from and writing data to the BAG file format. The APIs will provide a straightforward set of interfaces to read and write BAG format data, isolating the application programmer from the format complexities, OS and architecture independencies, large file (> 2 gig) support, and other gory implementation details. The software access library will be developed using ANSI C.

3.3 Layout of the File

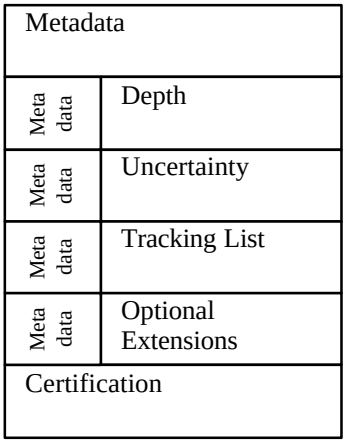

Figure 1. Conceptual BAG File Layout

A basic organization of the contents of the BAG file was agreed to and is shown in figure 1. The meta data, depth, uncertainty, tracking list and certification sections of the file are mandatory. The extension section(s) of the file are optional and will be addressed after support for the mandatory elements has been demonstrated. The metadata section contains all of the required descriptive information about the dataset itself, including the definition of the geo-spatial parameters that define the grid resolution and extents. This

includes items such as the coordinate system (i.e. geographic, or projected and projection parameters), horizontal and vertical datum, x node spacing, y node spacing, number of rows, number of columns, and the exact location of the south-west corner of the grid. The elevation (depth) section is a regularly spaced grid (rows, columns) with one value per node, where the node is defined as the georeference point for the cell (i.e., node-based rather than grid-based georeferencing). The meta-data prefix to the elevation (depth) section contains any elevation specific parameters such as minimum value, maximum value, precision, etc. In addition to its metadata prefix, the uncertainty section contains a two dimensional array (rows, columns) of vertical uncertainty values. Both elevation and uncertainty sections are stored as row-major ordering when packed. The BAG format uses a model-based approach that defines the elevation (depth) at points whose positions are exactly specified, so it is only necessary to maintain uncertainty in the vertical. The tracking list section will contain the list all original node-values that have been modified after the original surface definition. The intent of this section is to provide a facility that records the original grid values when a change is made to the surface definition such as may occur during hydrographic processing of *features*. The *feature* depth will replace the model derived surface value. When this is done, an entry is made in the tracking list to record the change, save the original model derived value (row, column, depth, uncertainty), and identify a reason code for the change. The certification section provides a facility that will allow appropriate organizations to "sign-off" on the BAG data file at an appropriate juncture in the completion of dataset validation. The signature section includes a hash code that can be used on subsequent access to verify that the certified contents of the file have not changed. The combination of the signature and the hash code provide proof of pedigree of the file contents.

3.4 Meta-Data Information

A working subgroup was formed to develop the content of the BAG meta-data and the approach to encapsulate this information in the file. XML was selected as a logical approach for encapsulating the meta data in a structured and human readable form. It was noted that some native support for XML is being developed with HDF5 (http://hdf.ncsa.uiuc.edu/HDF5/XML/). Following the decision to adopt the ISO19139 XML encoding of ISO19115, the subgroup set out to define the BAG specific mandatory meta-data tags that will be required in addition to those items mandated by ISO19115. The working subgroup made considerable progress towards defining the contents of the meta-data section of a BAG file. Sample XML schema files were developed and presented to the group for review and comment. Specific discussion was given to the approach for tracking the BAG format version and the BAG access software library version. It was decided that these version specifics would be stored in text form and would be the initial entries in the meta-data section. Mandatory meta-data items identified include: survey platform description(s), sensor description(s), navigation system(s), citation, originator, date/time of data collection, data lineage and chain of custody, data processing history including algorithm details such as name/version/originator, grid resolution in X dimension, grid resolution in Y dimension, position of reference point, number of rows in grid, number of columns in grid, horizontal and vertical datum, coordinate system, projection parameters, access level and distribution level for security attribution and certification for use of information (i.e. suitable for navigation). The sample XML schema files provide a good starting point for defining the necessary specifics of the meta-data, and the encapsulation approach.

3.5 Reprojection and Coordinate Transforms

The group agreed that it will be necessary to support grid definitions in geographic coordinates and in a limited number of projected (X, Y) coordinates. Meta-Data elements are included to fully define the required horizontal and vertical datums, coordinate systems, units, and fully describe the projection parameters. Based on recommendation from NAVOCEANO, the group agreed to adopt the use of the NGA geotrans (http://earth-info.nima.mil/GandG/geotrans/geotrans.html) software library for use in coordinate system transformations and datum conversions. The group agreed that the BAG API should allow for the definition of a BAG in either geographic or projected coordinates and the API should provide support for BAG access via both geographic and projected coordinates. The group agreed to support the following projected coordinate systems: Mercator, Polar Stereographic, Lambert Conformal, and Universal Transverse Mercator. Considerable discussion was given to supporting state plane projections. The decision was made to not support state plane projections at this point in time due in part to the significant regional variability in local datums and state plane's use of feet for distance measurement units. Coordinate system units of distance measurement will be decimal degrees for geographic grids and meters

for projected grids. The group agreed to support the following vertical datums: Approximate Lowest astronomical tide, Equatorial Springs Low Water, Highest Astronomical Tide, Half tide level, Indian Springs Low Water, Lowest equatorial tide, Lowest Low Water, Lowest Normal Low Water, Low Water Datum, Mean Higher High Water, Mean Higher Low Water, Mean High Water, Mean High Water Neap, Mean High Water Springs, Mean Lower High Water, Mean Lower Low Water, Mean Lower Low Water Springs, Mean Low Water, Mean Low Water Neap, Mean Sea Level, and ellipsoidal height. The group agreed to support the following horizontal datums: North American 1983, World Geodetic System 1972, and World Geodetic System 1984.

3.6 Certification and Digital Signatures

Certification will be implemented by use of SHA-1 (FIPS 180-2) for computing a secure hash (message digest) of the HDF5 file, which is then signed using the DSA (FIPS 186-2). The group discussed implementing the signatures as a separate file, but discarded this idea since it would be too simple for this information to become 'detached'. The Secure Hash Standard (SHS) works by computing a number from the data that, with very low probability of error, can only come from that message. Hence, the SHA-1 message digest cannot be stored inside the HDF5 file, and the group consensus was that it should be stored at the end of the file as a separate fixed-length section added using binary mode append. A simple experiment was run to ensure that this did not affect the HDF5 library, and this appears to be the case.

This mode of operation ensures that the file is essentially tamper-proof with no further action from the user. To modify the file using the library, the user will have to read the source and reconstruct a destination file. In this case, the digital certification section will not be copied, and the certification check will fail. The user might, under certain circumstances, be able to make a direct binary copy of the file (which would preserve the certification), and then modify it using a binary editor or other HDF5 data editor (e.g., to change a depth). In this case, if the length of the file did not change, the certification data would be preserved, but the modification in the file would cause the SHA-1 message digest to fail, and the user would be able to determine that the file had been modified. It is possible, but unlikely, that the file can be modified to over-write the data section, and hence infiltrate the certification section. In this case, the integrity markers at the start of the section would fail to check, and the certification damage would be recognized (and in any case, the certificate would not check). Hence, almost any change to the file would allow the user to determine that something had happened, although not necessarily *what* had happened. The group consensus was that it was the job of the user level code to ensure that the meta-data for the file reflects changes to the file appropriately. There is a slight concern that a multiply modified file might accumulate a number of certification blocks. However, the size of these blocks is trivial (on the order of hundreds of bytes), and the group consensus is that this is not significant relative to the size of the other elements in the file.

The group discussed the implications of this scheme on workflow within a production system, and the requirements for re-certification at each stage. This was felt to be cumbersome, and would make for delays in getting data from place to place. Therefore, the group consensus was that certification should not be made mandatory for the code to operate on the file, and that it should be the user level code's responsibility to check and apply certifications as required. Our belief is that certification is something that should be carried out at checkpoints in the processing of the data, e.g., as the data is first made into a surface, when it is checked by competent shore-bound authority, as it is accepted for ingestion into a database, and as it is made into a product surface, etc.

There is some concern that the key-length of SHA-1 (160 bits) may not be adequate for all uses (in the sense that the key-length affects how likely we are to have a message digest collision, and how easy it would be to forge a digital signature. There are a number of variants of the SHS and DSA that use longer key lengths, up to 512 bits for SHA-512, and more powerful signature algorithms, and adoption of these methods may be implemented at a later date.

The ONSWG recognizes that the implementation of a crypto-key security system is not a trivial concern since the security and verisimilitude of the system depends on the security and verification of the private keys of the participants. Key generation, certification and distribution is the most difficult and important part of the process, but it is also the one segment that the ONSWG cannot mandate, since it is properly an operational requirement of the agencies involved. The implementation, therefore, will be carried out assuming that the user level code will provide for the key management portion of the system, providing both public and private keys as required for the code. Agencies interested in assuring their data are encouraged to investigate the requirements for a secure key distribution system, and in particular the

requirement for a trusted third party to verify the identities of the entities involved by issuing an appropriate certificate file.

3.7 Unit Testing and Datasets

The group agreed that a set of unit and regression test code will be maintained for the BAG software access library. Initially, these test modules will be provided by the contributing members based on test code written to demonstrate access library functionality. It is envisioned that the unit and regression test code and test datasets will provide a mechanism to help ensure required functionality is maintained as software updates and maintenance is performed. New features added by contributing organizations, should be accompanied by appropriate unit test software modules. Initial datasets made available for testing include the NOAA Woods Hole dataset and the NAVOCEANO Loihi dataset.

4 Workshop Progress

4.1 Software Technology Identification

The workshop participants considered a number of different technologies for each element of the overall design. The following components were identified:

1. HDF5-1.6.1. This library is available as binary or source files for all platforms of interest. The group initially attempted to compile the source distribution, and found that this proved to be more than a trivial exercise. There are a sufficient number of problems in compiling on both Win32 and Linux that some care will be required in implementation of an overall build system that will support both of these key operating systems, and the others required for the project (see the Requirements Document). In order to speed testing, a binary distribution was obtained for Win32, which contains both static and dynamic libraries; a similar distribution was obtained for Linux, and both systems were successfully installed and used with example code. Note however that tests against a realistically complex build environment (in this case IVS'sFledermaus) showed that there may be considerable difficulty in resolving dependencies between libraries, which may require significant work. Custom code to generate the BAG file format in HDF was developed, and use of h5dump (a utility that is provided with the distribution) showed that the format was in fact correct. We are therefore confident that the file format can be readily constructed, although with some caveat on the complexity of the build process. The HDF sub-group were, however, confident that this was essentially a matter of a little concentrated effort to get everything into the correct location. It would be possible to go forward with binary distributions for all of the systems required, but this might become cumbersome in the future. The sub-group felt that, in the spirit of Open Source, the build should be required to be 'clean' from source directly.
2. zlib 1.1.4. This is an auxiliary library required by HDF5. Similar comments to those for HDF5 apply to the process of building the software, but a binary distribution was available and was utilized for testing.
3. szip 1.1. This is an auxiliary library required by HDF5. Similar comments to those for HDF5 and zlib apply here too. A binary implementation was also used for testing.
4. libgcrypt-1.1.44. This is a freely available encryption library, and contains all of the code for SHS and DSS. A source distribution was obtained and compiled under Win32 (using CygWIN, Gnu Make and gcc(1)), and under Linux. A build system for Win32 native (i.e., Visual C++) will have to be built at some point, since this is essentially required for proper inter-operation with some vendor's software build environments. Example code that came with the distribution was compiled and tested, and passed all of the build-in self-checks.
5. libgpg-error-0.6. This is an auxiliary library required by libgcrypt-1.1.44. A source code distribution was obtained and compiled under Win32 (using CygWIN) and Linux. Again, a build system for Win32 native will have to be built eventually.
6. geotrans 2.2.3. This is an application program that supports conversion of geographic coordinates to a wide variety of coordinate systems, map projections, and datums.

The group agreed that for builds of software under Linux, there was a concern about the particular version of the Gnu C Library (and hence gcc(1)) that was being used. There are two different versions, gcc 2.95 (or 2.96) and gcc 3.x. For support under various versions of Linux used by some of the development group, gcc 2.95/2.96 are preferred, and the group adopted this as a goal.

The sole technology not fully identified is the XML parser required to build and unpack the meta-data associated with the format. There are a number of potential solutions, including the Xerces parser/validator associated with the Apache project (`http://www.apache.org`).

4.2 Software Implementation

Much of the software development was focused around identifying the correct technologies, and checking that they would work in the manner expected. Therefore, much of the progress was exploration and test code of limited utility in the remainder of the project. All of the available code was checked into the CVS repository (q.v.) after the workshop, using the `openns` module. All filenames given here are taken relative to the root directory where this module is checked out.

The builds are:

1. HDF File Format. Code was built to write the BAG file format in HDF, and is provided in `$(ROOT)/tests/hdftest2/hdftest2.cpp`. A Visual Studio project file is also provided. The code is known to build the BAG format prototype structure, as confirmed by `h5dump`.
2. libgcrypt Encryption. A test example was identified and modified to show the basic functionality required to implement the certification (SHS) and signature (DSS) elements of the format. The example is `$(ROOT)/libgcrypt-1.1.44/tests/bag_test.c`.
3. API Functionality. An initial implementation of the API structure was constructed, outlining the calls required to implement the format. The file is in `$(ROOT)/ons/api_defs/bag.h`. Note that not all calls are fully populated, and the formal parameters list for some are still in development.
4. XML Sheet and Example. An example XML file was created to illustrate the components considered to be mandatory in a BAG file. In current form, it has the majority of the required tags, but lacks some of the extensions which will be required for the BAG meta-data format. It is available as `$(ROOT)/ons/xml/BAGMetadata/smXML/bag.xml`. A corresponding XSD file describing the XML is available as `$(ROOT)/ons/xml/BAGMetadata/smXML/BAG.xsd`. Note that many sub-files are required to support ISO19139 style XML tags, and these are available in `$(ROOT)/ons/xml/gml3.0` and `$(ROOT)/ons/xml/smXML`.

4.3 Documentation

The group agreed on a minimal requirements document, and this was constructed and verified during the workshop. A draft of the Format Specification Document was also developed, summarizing many of the decisions outlined here, and outlining the methods to request modifications for the format. Both documents were provided for the CVS repository (q.v.) and were checked in to the `openns` module, in `$(ROOT)/docs`.

4.4 Assignment of Modules

The modules required to build the code were portioned out after the API was defined. In total, there are four main modules to the API implementation (in addition to the API definition header file), and a number of support libraries. The CVS repository (q.v.) was designed to reflect the structure:

- `hdf5-1.6.1` [Lead: Paton]. All code required to build the HDF5 library.
- `zlib` [Lead: Paton]. Support code for HDF5.
- `szlib` [Lead: Paton]. Support code for HDF5.
- `libgcrypt-1.1.44` [Lead: Calder]. Encryption base library.
- `libgpg-errror-0.6` [Lead: Calder]. Encryption library error reporting.
- `geotrans` [Lead: Fabre]. Geotranslation engine for reprojection of data.
- XML_parser [Lead: Lamey]. Parser library still to be determine; will be added when available.
- `ons` [primary code interface API for user level code]
    - `api_defs/bag.h` [Lead: Byrne]. Definitions file for whole user level library.
    - `hdf` [Lead: Paton, Depner]. Wrapper code for dealing with the raw file.
    - `crypto` [Lead: Calder]. Wrapper code for implementation of certification/signatures.
    - `xml` [Lead: Lamey, Ladner]. Wrapper code for meta-data interface.
    - `error` [Lead: -]. Error reporting and control. No lead coder has been assigned against this module, since it will depend heavily on the development of the other modules. There was general agreement to defer this decision until more of the API was built.
- High-level API wrappers [Lead: Gallagher, Riley]. SWIG (http://www.swig.org/) wrappers for BAG library API to provide unit test scripts.
- `docs` [Lead: Calder, Moggert, Byrne]. Any and all documentation, particularly the Format Specification Document, Requirements Document and auxiliary information.
- `lib` [Lead: -]. An empty directory designed to hold the compiled library modules for the auxiliary and API libraries.
- `include` [Lead: -]. An empty directory designed to hold header files for the API and other auxiliary libraries.

- `bin` [Lead: -]. An empty directory designed to hold the compiled binary executables for the BAG format examples, plus any build helpers.
- `tests` [Lead: TBD]. Example of regression test code and data. Decision on lead deferred by general agreement until more of the API is built.
- `examples` [Lead: TBD]. Example source code designed as tutorials for potential users. Decision on lead deferred by general agreement until more of the API is built.

The group agreed that our initial target should be to ensure that code builds on at least the Win32 and Linux platforms, since those will be the most immediate and pressing concern. Since it is not possible for everyone to compile the code and test it on each hardware system, the group also agreed that each person in charge of an auxiliary library component would ensure the compilation on Win32 and Linux, and stand-by to act as advisor, mentor and guide for anyone else wishing to re-compile on another system.

4.5 CVS Repository Construction

The code repository was build on 2004-01-25 by Calder using the CCOM/JHC external CVS server at `:pserver:<user>@cvs.ccom.unh.edu/projects/external` (`cvs.ccom.unh.edu` is a synonym for `ccom.unh.edu`, or 132.177.44.120), user `openns`. Interested users should contact Calder (brc@ccom.unh.edu) for details of access. When making requests, please identify:

1. The IP address of the machine you'll be connecting from. We filter by host name or number to limit our `pserver` exposure. This is required to interface with the server at all.
2. The user name that you want to use.
3. Any preferred password.

If no preferred password is given, a default one will be created. There are three modules available for checkout:

- `openns`. The base code, as described above.
- `openns-data`. The regression data units provided by various vendors. This initial checking contains GSF and PFM data for the Loihi Seamount (provided by NAVO), and a depth/uncertainty surface pair for Woods Hole, MA (provided by CCOM/JHC using data from NOAA). This is a significant amount of data, and may before a problem in the future.
- `openns-wg`. A collection of all of the document associated with the meetings of the Open Navigation Surface Working Group. This contains agendas, meeting notes and summaries.

5 Summary and Recommendations

The workshop identified the key elements required to represent the data for an Open Navigation Surface data object, known as the BAG (Bathymetric Attributed Grid) file format. After much discussion, the group agreed that the file should be at the very least a good transfer format to encourage and enable exchange of data between systems, but that it should also be extensible so that it can be used for other purposes too, and indeed as a general file format for processing if required.

The group agreed that the file should consist of a hierarchical arrangement of data (in HDF5) and meta-data (in XML to ISO19139 with a number of extensions and mandatory items), with the emphasis on ensuring that sufficient detail went into the file to ensure that the user was always informed about the provenance of the data, and its intended use. The group recommends the use of appropriate cryptographic procedures to ensure that this is guaranteed. A basic BAG consists of meta-data, and a triplet of depth, uncertainty and tracked changes to the surface.

Software support libraries for all of the primary components of the file format have been identified, except for the XML parser, where it was felt that a proper definition of the contents of the meta-data was more important than identifying the parser software immediately. Simple example programs using these libraries were developed to ensure that the modules worked as expected; so far, this appears to be the case. Software modules were parceled out according to necessity and as far as possible the strengths of the various available developers.

Documentation of the format was started, and is at preliminary draft stage. This documentation consists of a Requirements Document, detailing the aspirations of the project, and a Format Specification Document, which details the actual format as it is currently implemented.

All of the supporting auxiliary libraries, test examples and documents were gathered together and checked in to a CVS source control repository, which has password protection, but otherwise open access. Calder at CCOM/JHC (brc@ccom.unh.edu) will act as repository administrator.

In conclusion, the ONSWG makes the following recommendations for the continuation with the project to build a BAG implementation:

1. Encourage wide dissemination of the report on the workshop via e-mail lists; enlist aid of volunteers willing to help with development.
2. Solicit input from other agencies not involved with the project directly on any concerns or limitations that they might find in the format (e.g., NGA and USACOE).
3. Develop and implement a web site, www.openns.org, to provide information on the project, access meeting notes, design documents and other information, and provide a point of contact for requests. CCOM/JHC has volunteered to host the site if necessary; NOAA HSTP volunteered to investigate the requirements to set this up.
4. We suggest that the future development of the BAG file format should be driven primarily by e-mail connection through the navsurf_dev@ccom.unh.edu list, with an increasing interaction level of teleconferences, video conferences and (as a last resort) face-to-face meetings. Where possible, we should arrange for these meetings to take place as side-meetings to other events, e.g., the US and Canadian Hydrographic Conferences.
5. The ONSWG would not have come about without the dedication and support of the participants and their home institutions and companies, which are gratefully acknowledged. The future of the BAG format will require the same dedication, decisive leadership and support, and we would encourage the host entities to factor this into their decision and planning systems.

6 Participants

Brian Calder (CCOM/JHC)
Rick Brennan (CCOM/JHC)
Bill Lamey (CARIS Ltd)
Mark Paton (IVS Ltd)
Shannon Byrne (SAIC Newport)
Jim Case (SAIC Newport)
Dave Fabre (NAVOCEANO)
Wade Ladner (NAVOCEANO)
Barry Gallagher (NOAA HSTP)
Friedhelm Moggert (SevenCs AG & Co. KG)
Shep Smith (NOAA) [phone]
Jan Depner (NAVOCEANO) [phone]
Jack Riley (NOAA HSTP) [phone

7 References

[1] Smith, SM (2003). *The Navigation Surface: A Multipurpose Bathymetric Database*, Masters Thesis, Center for Coastal and Ocean Mapping & Joint Hydrographic Center, University of New Hampshire.

[2] Smith, SM, Alexander, L, & Armstrong, AA (2002). 'The Navigation Surface: A New Database Approach to Creating Multiple Products from High-Density Surveys', *Int. Hydro. Review*, Vol. **3**, No. 2, pp. 12-26.