ABSTRACT
In the classical triad of vector, raster, and meta data, it is the raster part which is not yet sufficiently supported in SDIs nowadays. Consequently, integration of earth observation imagery, LIDAR, legacy map scans, etc. into Spatial Data Infrastructures (SDIs) remains incomplete. In terms of standards, the OGC Web Coverage Service (WCS) Standard defines open interfaces for accessing and processing of raster data, more generally: coverages. In August 2010, the completely overhauled WCS 2.0 has been adopted by OGC. To make coverages interchangeable across all OGC-based services, WCS 2.0 has been based on Geography Markup Language (GML) 3.2.1, with a small, backwards compatible addition to achieve informational completeness. In parallel to specification writing, its reference implementation and an online demo are being pursued.

WCS 2.0 offers several advantages over previous versions, such as: support for general n-D raster data and non-raster coverage types; crisp, modular, and easy to understand; flexible and adaptive; harmonized with GML and Sensor Web Enablement (SWE); improved testability; and allows for efficient and scalable implementations.

In this paper we present WCS 2.0 and some central design rationales. Further, we inspect the reference implementation architecture discussing some features critical for scalability. Finally, we give an outlook on next steps, such as the planned WCS Earth Observation Application Profile.

Categories and Subject Descriptors

General Terms
Coverages, raster, grid, standards, OGC.

Keywords
Web Coverage Service, WCS, OGC, geo services, standards.

1. INTRODUCTION
Geo services today offer versatile, scalable access to manifold location-based information, including navigation, extraction/download, and aggregation. In the classical triad of vector, raster, and meta data, however, this is not entirely true for the raster part. Spatial Data Infrastructures (SDIs) largely tend to use raster images as backdrop only, tend to provide elevation Triangulated Irregular Networks (TINs) as pre-rendered height classification or hill shading imagery, and tend to provide atmospheric and ocean research coverage data on a file basis in ftp archives. Solutions better than that can be found occasionally, but these are ahead of the general state of the art and, hence, isolated and not interoperable with the community at large.

Scientific research has prepared the underpinnings for such services. Trenchard More first has phrased the notion of raster data as an array consisting of a function \( a: D \rightarrow V \) from index domain \( D \) to value domain \( V \) [17]. AFATL Image Algebra has proven a particularly valuable basis for eliciting the needs of domain-independent array processing. Array Algebra, which has been impacted by Image Algebra, constitutes the formal basis for the rasdaman system [7], “the most comprehensively implemented array DBMS” [15]. Further formalisms include AQL [14], AML [16], and RAM [1]. On the practical side, ad-hoc systems prevail; the GEOSS initiative (see www.earthobservations.org) aims at combining efforts. However, systems normally do not offer database-equivalent services, such as including a raster query language.

Going beyond raster data into the realms of point clouds and meshes we find much practical work in the application domains. Like with earth observation archives, many specialized systems have been developed – for example, TDS (THREDDS Data Server, www.unidata.ucar.edu/projects/THREDDS) – but usually services operate on the level of file download with retrieval language support limited to metadata. One effort to establish a general retrieval language for multi-dimensional coverages is
GridField [8]. OpeNDAP is an open-source project developing data format independent access to ease scientific data access [19].

For the subsequent use of the terms feature and coverage we rely on the OGC (Open Geospatial Consortium) and ISO provided definitions. A feature, in OGC and ISO nomenclature, is defined as an abstraction of some real-world phenomenon whereas a coverage means a space-time varying phenomenon; as such, a coverage is a special type of a feature. Raster data are common representatives of coverages, but irregular grids, point clouds, and meshes likewise constitute coverages.

The OGC Web Coverage Service (WCS) specification defines open interfaces for accessing and server-side processing of coverages. Figure 1 shows how WCS embeds itself into the corresponding triad of access services consisting of Web Feature Service (WFS), WCS, and Catalog Service (CSW).

**Figure 1. Some basic OGC standards related.**

A Web Map Service (WMS) allows for retrieval of images representing features and coverages. WMS is particular in this aspect as WFS and WCS allow retrieving data in their original, unchanged shape suitable for further processing and not just for visual inspection by humans, as is the case with WMS generated imagery.

Notably, each of WFS, WCS, and CSW nowadays has added a component for updating server-side offerings through a standards-conformant interface (in OGC jargon called “transactional” and abbreviated with a “-T” suffix, such as WCS-T). Further, all these services offer an extension to perform declarative search – Filter Encoding (FE) defines predicate-based selection of features, the Web Coverage Processing Service (WCPS) allows raster coverage filtering and processing, and the OGC Common Query Language (CQL) offers SQL-style metadata retrieval. This brief overview should emphasize the role of WCS as an extraction and download service on coverages which retain their original semantics.

WCS 2.0, which is the focus of this contribution, is a completely overhauled version currently in the final stage of adoption within OGC. It is no longer a single standard but a modular, structured suite of specifications fitting into an extensible overall concept. Technically, this WCS is based on Geography Markup Language (GML) and Sensor Web Enablement (SWE) and, hence, achieves a high degree of interoperability with related standards. In parallel to specification writing, its reference implementation has been pursued to verify viability of the concepts, together with an online demo accessible at www.earthlook.org.

WCS 2.0 constitutes a substantial step forward in geo coverage services in several respects. To understand these better we briefly skim over WCS history. Historically, WCS 1.0 issued in 2004 was the first ever attempt to standardize raster services. While metadata were described extensively the semantics on data (“pixel”) level was not at all addressed. Thus, a WCS server always delivering a black 10x10 raster could still be WCS 1.0 conformant. Of course, this affects interoperability negatively. Despite this, take-up was remarkable, and still today many implementations are in active use.

This was corrected in 2006 with the next version, WCS 1.1. Actually, many more issues were addressed there in addition so that the specification was much more complete and with a substantially higher degree of interoperability. On the other hand, the document was impressively large with its 135 pages of which about 65 pages address coordinate system issues. Consequently, this version generally was perceived as “complex” and “hard to implement”, and today very few implementations exist which claim conformance with WCS 1.1. Further, there was no differentiation in subsetting whether it reduces dimensionality of the result (in WCS 2.0 called slicing, see below) or keeps it (in WCS 2.0 called trimming); consequently, it was difficult to determine the actual result dimensionality which can lead to coordinate reference system (CRS) issues.

In the quest for the next, improved WCS version it gradually became clear that the necessary refactorization made it impossible to stay backwards compatible, and so WCS 2.0 was coined. The list of design goals was remarkable:

- **Crisp:** A specification which is crisp and easy to handle for implementers.
- **GML harmonization:** Harmonization with GML by basing the coverage model completely on GML 3.2.1.
- **General harmonization:** It should be possible to exchange coverages seamlessly between OGC services, in other words: establish a model powerful enough to unify coverage handling across all of OGC.
- **Increased domain support:** WCS should not any longer be constrained to mainly serve web mapping and remote sensing; a broad range of different domains has emerged meantime requiring homogeneous coverage support as well. Examples include atmospheric and ocean research, geology, aviation, aerosol chemistry, and finally sensor coverage data in all their variety. This had a few particular consequences, as the next two requirements show.
- **Beyond raster:** More than just raster (“quadrilateral grid”) coverages should be supported – in particular the meteorology and oceanography (“metocean”) communities expressed the need for coverage types like curvilinear grids (e.g., for littoral data) and more general meshes.
- **N-D coverages:** Homogeneous support was requested across all spatio-temporal dimensions and beyond – in WCS 1.x, coverages were constrained to 2-D, 3-D, and 4-D; 1-D time-series, for example, were not considered. Additionally, “abstract” dimension axes having no spatio-temporal semantics

---

1. [www.opengeospatial.org/standards/wcs](http://www.opengeospatial.org/standards/wcs) provides WCS versions, [www.ogcnetwork.net/wcs](http://www.ogcnetwork.net/wcs) offers background and news.
were advocated particularly by the metocean community. For example, height there frequently is substituted by pressure. Temperature axes constitute another use case of such dimensions.

- **Concise semantics**: A concise semantics down to „pixel“ level was required – compare this to the above characterization of WCS 1.0 which didn’t make any statement about the pixel payload.

- **Improved testability**: Connected to semantics, it should be easier to derive conformance statements on particular implementations under test.

- **Core/extension modularization**: Just at the time when WCS 2.0 write-up started OGC came out with the core/extension model which was to become the mandatory specification style to achieve modularity, extensibility, and increased testability. The WCS 2.0 documents were the first ones to practically apply this, hence a lot of experience had to be gained during specification writing.

- **URI identifiers**: Similarly, on the way rules for URI definition and use were beginning to be established by the OGC Naming Authority, OGC-NA. These rules had to be implemented as well, again with a learning curve on both sides.

- **Implementability**: the specification should allow for (and actually endorse) efficient and scalable implementations.

Based on extensive preparatory discussion, write-up started in May 2009. From the start, active participation of stakeholder communities was sought, although diverging legacy, viewpoints, and goals made it anything but trivial to come up with concepts acceptable across domains. The specification bundle evolved over several versions which were subject to in-depth investigation by the stakeholders involved leading to manifold improvements. In August 2010, the specification finally has been adopted by the OGC voting members and now is official OGC standard. This state is the target of presentation in this contribution. To this end, the remainder of this contribution is organized as follows. In the next section we present the conceptual coverage model. The WCS Core is addressed in Section 3, its extensions in Section 4, and Application Profiles in Section 5. Following this, in Section 6 some central design rationales are discussed, followed by a brief overview on the reference implementation architecture. We conclude the paper with a status brief and an outlook in Section 7.

The author is co-chair of the coverage-relevant working groups in OGC and editor of eight specifications, among them the coverage model [2], the WCS Core [4], several extensions, and the forthcoming Earth Observation Profile.

### 2. Coverage Model

The coverage model is laid out in the “GML Application Schema for Coverages” (ASC) specification [2]. It is based on GML 3.2.1 [22], which in turn relies on OGC Abstract Topic 6 [18] which is identical to ISO 19123 [9]. According to the latter, a coverage is a function mapping locations in space-time, the coverage’s domain, to values of some data set, its range.

GML is an XML implementation of this model. The root of the GML coverage definition is the abstract class AbstractCoverage which is a subtype of GML’s AbstractFeature. The main components of AbstractCoverage, in addition to several inherited properties, are domainSet and rangeSet. In the domainSet, the CRS (including its axes) is specified, plus the coverage’s extent. The rangeSet contains the concrete range values (such as pixel radiometries) plus some metadata. This was perceived incomplete, hence the ASC adds a third component, rangeType, which contains a complete specification of the range values, including data type, units of measure, and nil values (if any). To achieve harmonization with the OGC Sensor Web Enablement (SWE) standards family, the DataRecord definition of SWE Common 2.0 [23] has been adopted.

Range data types can include recursively nested structs and multi-dimensional arrays. Figure 2 shows the extended AbstractCoverage structure.

Figure 2. Extended AbstractCoverage structure [2].

Several concrete subtypes are derived from AbstractCoverage, including gridded coverages, point clouds, and other types of varying topological dimension. These allow representing, in addition to multi-dimensional rasters, a large class of practically relevant data structures, such as point clouds, curvilinear grids, Triangulated Irregular Networks (TINs), and general meshes.

Overall, only few, carefully pondered changes have been made over the GML model. ASC solely contains additions over GML coverages, hence it is backward compatible in the sense that existing applications using GML coverages may safely ignore the new components. For the future, it is planned to merge ASC into forthcoming GML 4.0. A sample GML coverage instance document, which validates against the ASC schema, is provided in the Appendix.

### 3. Coverage Service Core

Following the OGC core/extension model for modular specifications [21], WCS is split into a Core and an open-ended set of extensions. The Core defines properties that any WCS implementation must offer in order to be conformant with WCS overall. Extensions define further functionality which an implementation can add. In this section, we first inspect the WCS Core data model and then its service model.

#### 3.1 WCS Data Model

WCS 2.0 is based on the ASC coverage model. In the specification, the overall service offering is modeled as a single (virtual) GML document with root CoverageOfferings (cf. Figure 3). It consists of service specific parameters plus the set of coverages available. Each such OfferedCoverage in turn consists of a cover-
age as defined by the ASC plus coverage-specific service parameters, such as the CRSs in which a coverage can be requested. In the WCS Core, only a placeholder is reserved so that extensions have a canonical place to add such information.

![Figure 3. Coverage offering of a WCS](image)

In order to allow the WCS service functionality a few restrictions are placed on coverages as compared to ASC. For example, the id attribute of an AbstractCoverage must be unique across the server offering document tree. Further, every coverage must contain a bounding box definition summarizing extent and axis names.

### 3.2 WCS Service Model

WCS services offer three mandatory request types: GetCapabilities, DescribeCoverage, and GetCoverage. GetCapabilities is the canonical OGC Web Service discovery request where a server unveils its service capabilities and, optionally, a list of the offered coverage’s identifiers. Using DescribeCoverage a client can request details – i.e., metadata – about one or more coverages by providing such identifiers.

The actual workhorse is the GetCoverage request. For a given identifier, it allows to retrieve all or part of the corresponding coverage. In the Core, domain subsetting is defined. As a novelty in WCS 2.0, it is subdivided into trimming and slicing for reasons of conciseness. A trim operation reduces the extent of a coverage in the axis dimension as indicated, thereby preserving the number of dimensions. A slice operation, on the other hand, performs a cut through the coverage in the axis and at the slice coordinate specified, thereby reducing dimensionality by one. Trim and slice operations can be combined arbitrarily in a GetCoverage request. Care has to be taken, though, to indicate an output format in the request which is capable of handling the resulting dimensionality and range type. For example, the output of a trimming on a 3-D cube, which is 3-D again, can be delivered in NetCDF, but not in JPEG. A 2-D slice, on the other hand, can be delivered in JPEG.

For reasons of modularity some mandatory functionality packages have been separated into extensions. One such extension is raster subsetting based on array indexes; this is required if the server offers raster coverages. Further, a minimal complete WCS implementation must support at least one format and one protocol extension. We will discuss such extensions next.

### 4. Coverage Service Extensions

Soon during the design of WCS 2.0 it has turned out that the number of extensions would not be just two or three. Actually, the current count of extensions developed and foreseen is already at fifteen. This has prompted the need to structure the "WCS extension universe" in a suitable way, that is: allowing readers of the specifications to easily grasp the "big picture".

Figure 4 shows such a structuring using UML class diagram notation where each specification is represented as a class. Notably, this UML diagram does not capture all dependencies – UML has turned out insufficient for representing all the details of semantic relationships between the specifications. This is not a big issue, though - it is understood that the diagram serves for overview purposes only, it is not normative.

Extensions are grouped into five categories. **Data model extensions** add further information to coverages. As a representative, the Uncertainty extension is going to add error estimates, uncertainty information, and other quality relevant metadata to coverages.

**Service model extensions**, on the other hand, extend functionality on the coverages offered. WCS-T allows modifying a server’s...
offering by adding, updating, or deleting coverages [25]. The Processing extension [6] ties in the OGC Web Coverage Processing Service (WCPS) Standard [5] which defines a declarative query language on multi-dimensional raster data in the tradition of SQL. The Scaling & Interpolation extension adds scaling, resampling, and interpolation to the GetCoverage request. As such, it paves the way (and, hence, is a mandatory prerequisite) for the CRS extensions. The simple CRS extension version adds EPSG coordinate reference systems so that CRS transformation of coverages becomes possible in GetCoverage requests. The general version additionally allows ad-hoc defined and nested CRSs. 

The family of coverage format encoding extensions will support coverage delivery in the various well-known image and scientific data formats. First in line are GeoTIFF, NetCDF, and JPEG2000; more will follow pending demand and resources. A special role is played by the GML extension. On the one hand, it defines XML as an output format (GML 3.2.1. already foresees four different, semantically equivalent, encodings). On the other hand, this format defines aka wrapper around the other formats. This allows for three different variants of delivery, which all have been requested by the user community:

- Delivery of the image/data file alone. This allows existing applications to readily digest the coverage file delivered. However, metadata possibly can be represented only incompletely, depending on the particular format.
- Delivery of pure GML. This format delivers all metadata according to the standardized XML schema, plus the range value data also encoded in GML. While for in-situ sensor time series, for example, this is appropriate, it is not efficient for high-volume coverages.
- Delivery of a GML metadata record together with an encoded file, MIME-encapsulated. This allows to take advantage of the encodings while, at the same time, delivering a complete set of metadata in the canonical XML format.

The fourth extension category consists of the Protocol Bindings. Currently supported are GET/KVP, XML/POST, and SOAP; a RESTful binding might be added depending on demand and resources.

Finally, the Usability group collects extensions which add neither coverage data structures nor coverage functionality. Currently, multilingual support has been placed there.

Only a few extensions have been written as for now, some are in progress, and several more are waiting to be scheduled. In the spirit of the volunteering approach of standardization bodies like OGC there will be a natural line-up: extensions will be written in a similar way (and, hence, is a mandatory prerequisite) for the CRS extensions. The simple CRS extension version adds EPSG coordinate reference systems so that CRS transformation of coverages becomes possible in GetCoverage requests. The general version additionally allows ad-hoc defined and nested CRSs.

The family of coverage format encoding extensions will support coverage delivery in the various well-known image and scientific data formats. First in line are GeoTIFF, NetCDF, and JPEG2000; more will follow pending demand and resources. A special role is played by the GML extension. On the one hand, it defines XML as an output format (GML 3.2.1. already foresees four different, semantically equivalent, encodings). On the other hand, this format defines aka wrapper around the other formats. This allows for three different variants of delivery, which all have been requested by the user community:

- Delivery of the image/data file alone. This allows existing applications to readily digest the coverage file delivered. However, metadata possibly can be represented only incompletely, depending on the particular format.
- Delivery of pure GML. This format delivers all metadata according to the standardized XML schema, plus the range value data also encoded in GML. While for in-situ sensor time series, for example, this is appropriate, it is not efficient for high-volume coverages.
- Delivery of a GML metadata record together with an encoded file, MIME-encapsulated. This allows to take advantage of the encodings while, at the same time, delivering a complete set of metadata in the canonical XML format.

The fourth extension category consists of the Protocol Bindings. Currently supported are GET/KVP, XML/POST, and SOAP; a RESTful binding might be added depending on demand and resources.

Finally, the Usability group collects extensions which add neither coverage data structures nor coverage functionality. Currently, multilingual support has been placed there.

Only a few extensions have been written as for now, some are in progress, and several more are waiting to be scheduled. In the spirit of the volunteering approach of standardization bodies like OGC there will be a natural line-up: extensions will be written once somebody considers them important enough to write them.

### 5. Application Profiles

Given the substantial number of extensions the question arises how to achieve interoperability among implementations choosing different extensions, and also how to give guidance to implementers on a wise selection of extensions. To this end, the concept of Application Profiles (APs) is provided by OGC.

An Application Profile represents a bundle of specifications together with some suitable additional data structures and functionalities as well as bespoke rules for the envisaged application domain. For WCS, an Earth Observation Application Profile (EO-AP) is under construction which focuses, streamlines, and extends WCS specifically for remote sensing data services. At the current — preliminary — stage of discussion it is planned to incorporate at least one of GET/KVP and SOAP protocol bindings; at least one of the formats GeoTIFF, NetCDF, and JPEG2000; and the range subsetting extension. Further specializations include, in terms of data structures:

- requiring GML Earth Observation Metadata [18] to be present in the coverages delivered;
- requiring support for 2-D x/y and 3-D x/y/t coverages.

And in terms of functionality:

- requiring the Scaling & Interpolation extension;
- requiring the CRS/EPSG extension.

Another application profile foreseen for the future is aiming at atmospheric and ocean data.

### 6. Reference Implementation

In parallel to the specification writing a reference implementation is being pursued which will be released as open source once sufficiently advanced. This has greatly helped to assess feasibility of concepts. Further, it is considered advantageous for a broad take-up of the forthcoming standard to have a free, open-source implementation available by the time the standard gets adopted and released.

This reference implementation (cf. Figure 6) is based on the open-source raster server rasdaman (standing for “raster data manager”, see www.rasdaman.org). This so-called array database system stores multi-dimensional raster data in either files or inside a relational database, such as PostgreSQL. The rasdaman query language, rasql, extends standard SQL to multi-dimensional raster arrays in a declarative, evaluation-safe manner. Internally, manifold optimizations on logical and physical level help to achieve performance and scalability. To achieve best performance the rasdaman server is implemented in C++ with APIs in C++ and Java.

The petascope layer of rasdaman adds geo semantics by providing OGC based interfaces for WCS, WCPS, WCS-T, and WPS. This Java servlet package receives requests, translates them into rasql for processing by rasdaman, and sends the results generated back to the client.

An online demonstration is publicly available on the EarthLook site (www.earthlook.org). Aside from showing manifold 1-D to 4-D use cases for raster query processing through WCPS it offers a small WCS 2.0 demo to illustrate feasibility of the approach.

Performance of the architecture is very satisfactory and can be observed on EarthLook. Typically, even complex classification queries take about 400ms, and simple access is significantly below. Under lab conditions, queries involving 30 raster operators have been brought below the 100ms frontier. Figure 7 shows the results of a performance evaluation [13]. The rasdaman query under test performs \( n \) pixelwise multiplications

```
select x*x* … *x from float_matrix as x
```

on a collection holding a single 2-D 512x512 coverage, float_matrix, of double-precision floating-point numbers.
The optimization applied here – one out of several rasdaman employs – is just-in-time compilation: an incoming query is analyzed for compilable fragments. These fragments are isolated and collected in clusters. From each of these clusters C code is generated, compiled, linked dynamically into the server, and executed on the data addressed.

Figure 7 shows evaluation times in rasdaman for varying \( n \) where \( n \) denotes the number of multiplications in the query; effectively, hence, 262,144 floating-point pixel multiplications are executed. Note the logarithmic scale on the horizontal axis. ORIGINAL marks query evaluation with all optimizations switched off. With just-in-time compilation enabled the COLD displays results in the case when the query had to be compiled and linked into the server; HOT is time elapsed when the precompiled code is already available in the server executable. TAILORED, finally, is a hand-crafted expert implementation of this query's code to have a comparison with what comes close to the fastest possible implementation.

7. Discussion
In this section we review the design goals listed in the introduction to assess their level of achievement.

Crisp: ASC consists of 14 requirements, a payload of 14 pages description, plus a 6-page annex containing the Abstract Test Suite (ATS). The Core encompasses 42 requirements listed in 30 pages with an ATS of 14 pages.

GML harmonization: As planned initially, the coverage model relies on GML 3.2.1, with an addition – the range type – taken from SWE Common. Existing applications can safely ignore the range type information and use coverages as before (modulo namespace adjustments).

General harmonization: Several Standard Working Groups (WCS.SWG, GML.SWG, SWE.SWG) and Domain Working Groups (Coverages.DWG and the GALEON domain interest group) have actively collaborated in the work on WCS. Several groups have announced that they intend to base their further specification work on ASC, among them Observation and Measurements (O&M) and Web Processing Service (WPS).

Increased domain support: ASC and WCS in their new shape can support a variety of domains. We list a few representative examples:

- In-situ sensory through 1-D time series;
- Earth observation, through 2-D imagery and 3-D time series;
- Web mapping, for example, by feeding WMSs with raster layers;
- Geology, through 1-D vertical coverages, 2-D maps, and 3-D earth tomograms;
- Oceanography, through 4-D spatio-temporal raster data as well as curvilinear grids, TINs, and other meshes;
- Atmospheric sciences, with the same argument;
- A host of adjacent areas using data and methods from the earth sciences, such as: aerosol chemistry and metagenomics; further, more downstream applications like: biodiversity, environmental monitoring, aviation / air traffic control, and disaster management.

- Decision fusion and support, through versatile retrieval using the WCPS raster query language for ad-hoc derivatives like normalized indices and classifications.

Notably, cross-domain services become possible by combining hitherto separate data sets into a single service with a small set of generic operations; further, data now can be merged into new cross-domain datasets.

Beyond raster: Aside from the classical raster data (called “quadrilateral grid” in ISO 19123) the coverage model and service now supports the ISO 19123 coverage subtypes Multi-Point Coverage, Multi-Curve Coverage, and Multi-Solid Coverage which allow modeling of point clouds, nonuniformly spaced grids, curvilinear grids, TINs, and general meshes.

That said, operational support on the new coverages is rather limited for now: subsetting and WCPS queries are only possible on rasters; all other coverages for now can only be retrieved as a whole. Further work is required here.

N-D coverages: Beyond the classical x/y and x/y/t coverage types, WCS now allows for any spatio-temporal axis combination, including exotics like x/t or y/z/t (which can occur during slicing operations, for example).
Again, however, there is a drawback currently. As any coverage must have a CRS associated, this sometimes requires CRSs far outside the usual canon, such as EPSG. Additionally, on-the-fly construction of CRSs is necessary. These issues will be addressed in a CRS extension which is to be written next. ISO 19111-2 [10] with its concept of parametric coordinate systems seems supportive here.

**Concise semantics:** On syntax level, this has been achieved by rigorously using UML and XML Schema plus Schematron. On semantics level, description mostly relies on natural language (with the exception of WCPS which has a formal semantics specification). This is due to several facts: attempts towards formal specification of the standard were not well accepted by the communities (which sometimes do not have a core Computer Science education and, hence, are not familiar with specification styles like algebraic specification of abstract data types). This is not new – a similar observation in fact can be made in the fields of communication systems: LOTOS [11], an ISO Standard based on an algebraic formalization of concurrent systems, is largely forgotten today; ESTELLE [12], another ISO Standard with roughly the same functionality and target, but a graphical language, is adopted much more widely.

Further, it turns out that several aspects, like CRS semantics, are sufficiently involved so that a homogeneous formalization of the complete coverage field appears not (yet?) possible given the resources and timeframe available. An approach to use XPath or OCL at least on some parts has been abandoned due to the unwieldy results – it turns out that the resulting complexity is prohibitive not just in writing, but also user friendliness of the resulting text is sacrificed to an unacceptable extent.

As we find a formalization of the concepts at least for the Core highly desirable we pursue this further on academic level, separate from standardization. Currently we are working on a formalization of the WCS coverage structures and operations [26] using ontology concepts. Among our goals is to go towards automated test generation by using ontologies and a reasoner.

**Improved testability:** Concise formulation of one conformance test for each requirement should make testing significantly easier. Still, though, the Holy Grail of deriving tests automatically from requirements is not in sight. This is largely due to the missing formal semantics. In parallel to the standardization work we research on a formal reasoning at least on rasters so that automatic test generation in future might become reality.

**Core/extension modularization:** This has been achieved. Although there was quite some learning curve in applying this new specification style it has turned out highly advantageous – many cross dependencies which formerly have been implicit or forgotten are now explicit and, through the use of URIs (see next) sometimes even traceable automatically.

**URI identifiers:** OGC currently is under way to establish a thorough URI identification scheme for any of its entities which are of normative character, based on the OGC Naming Authority (OGC-NA). WCS 2.0 was the first specification to follow this approach. URI identification includes specification packages, documents, XML schemas, single requirements and conformance classes etc. down to CRSs, axis directions, units of measure, and the like. In future, this is intended to allow following dependencies automatically – or, at least, express them formally. Generally, this is perceived as an important step forward towards automatic service chaining and orchestration.

**Implementability:** Development of WCS 2.0 is based on 15 years of experience with raster services, embodied in the rasdaman system. This technology is running in operational installations, such as the French National Geographic Institute which is hosting a dozen-Terabyte ortho image of France more than five years at the time of this writing. Scalability as well as performance gains through optimization have been tested extensively in scientific benchmarks, such as [13]. A proof-of-concept implementation of WCS 2.0 has been accomplished based on rasdaman, with a demo online at www.earthlook.org; this implementation is being extended to the reference implementation. Hence, the raster aspects can be considered understood; experience still has to be gained on efficient services on large-scale archives for the further coverage types.

So much about the requirements matching. Next, we discuss some general experiences and observations made during the development of the specification.

Following [21], the normative statements in WCS 2.0 draft have been structured into requirements classes with corresponding conformance classes containing a testing recipe for each requirement. In most specifications, the simple case of just one requirements and one conformance class could be achieved. The approach of making normative requirements stand out from the text and to clearly visually differentiate between normative requirements and non-normative explanations has contributed significantly to making documents more concise and readable.

WCS being an Interface Standard does not define a server API, but a communication protocol between a server and a client (whereby the client does not necessarily have to be a human user interface, but can well be another fully automated process). Often requirements can be phrased in a way that they apply to “the line” rather than to one specific endpoint. Consider Requirement 5 in the Core as a simple example:

> For all WCS request types defined in this standard, the request version parameter shall have a fixed value of “2.0”.

For clients, this means that the version parameter generated needs to match. For a server, this requires accepting only this value and responding with an exception otherwise.

However, client testing currently is not in the scope of OGC – for example, the TEAM conformance testing engine used for OGC standards conformance testing only supports server examination. This reflects current practice in Web services where servers are tested extensively while clients less rigidly exercised. Explicit client testing, which is highly desirable, remains to be added in future.

Sometimes proper crafting simply was a matter of best practices. For example, ISO mandates that notes in regular text are non-normative whereas footnotes in tables are normative. Experience shows that this practice, which also holds for OGC standards, is highly confusing and error prone. Hence, in the WCS documents footnotes in tables have been omitted at all.

Another issue is the use of optional parameters and conditional functionality; as experience with WCS 1.1.2 shows this is a tempting way of adding flexibility, but increases overall complexity significantly. Maybe it even makes implementation simpler for
servers, for them it is no problem to respond flexibly depending on the incoming parameter constellation. Further, server implementers can choose whether to add some optional feature or not. Clients, however, need to be prepared to any potential constellation of features a server might support, which obviously leads to a combinatorial explosion. Optional features, therefore, are discouraged; the modular specification model indeed discourages them by enforcing separate conformance classes (which come with some overhead for the specification writer). What remains is the structure of a GetCapabilities response which is laid down in OWS Common [24] – almost every item there is optional.

Another example for the same shortcoming, albeit on another level, is the OWS Common rule that GetCapabilities delivers either metadata directly or it presents the client with a link to a catalog – however, without defining what such a catalog should contain and in which format. This way, automatic service detection and chaining is made difficult at least. Hence, recommendation to implementers is to return complete metadata in the GetCapabilities response.

GML likewise imposes some design patterns. For mapping UML associations it uses a generic relationship pattern. In the special case of a containment relationship as occurring in the rangeType part (Figure 8 left) this leads to an unnecessary overhead of five XML elements A, A-R, R, R-B, and B instead of just A and B (cf. Figure 8 right). All of these elements come with the standard GML attributes, making the resulting schema unwieldy. Simplifying the schema, on the other hand, is not possible because then GML parsers will not understand it as a relationship. Recommendation, therefore, is to extend the GML pattern set to support containment relationships.

Extensibility of the Core has turned out to have some intrinsic difficulties. On the one hand, clear normative statements need to be made as to what is enforced and what not, on the other hand extensions can add and sometimes even override features. In terms of data structures, this problem has been addressed by foreseeing “hooks” where extensions can plug in their extra structures. For example, the ASC coverage model contains an element named “extension” of XML type <any> and multiplicity zero to unbounded to this end. More difficult it was with modul-

arization of functionality. Ultimately, statements had to be used like “Property X shall hold unless an extension overrides this”.

Still, modularity has its limits. For example, the format encoding extensions add further parameters to the GetCoverage request, which requires the extension to define a mapping to all protocol encodings, something that normally would belong into the protocol extensions. However, we consider this not a design flaw (following extensive discussion of alternatives), but attribute it to the intrinsic complexity of the matter.

8. Conclusions and Outlook

Standards are a service for communities to ease their day-to-day work. In the field of geo services, this includes simple and fast access to data, wherever and on whatever system they may sit. “Access” unites manifold aspects of user friendliness in itself, such as discovery of suitable resources, availability of both metadata and data in a form that is useful for the purpose on hand, availability of derived products without a high penalty. The latter can be something as simple as subsets and scaling, something intermediate like deriving classifications on the fly, or something as complex as launching computationally involved simulations. In any case, preparing the result on the server is advantageous in view of the network bandwidth savings. For example, to obtain a Normalized Difference Vegetation Index (NDVI) defined as

$$NDVI = \frac{(a.nir - a.red)}{(a.nir + a.red)}$$

for near-infrared (nir) and red bands would require download of 2 bytes per pixel if processed on the client. Processing the NDVI on the server saves 50% of bandwidth. If thresholding is applied on the vegetation index then this will reduce the result pixel size to one bit which can be compressor very effectively. Let additionally apply scaling (because the client may be satisfied with a lower resolution) then a substantial gain in transmission speed is possible. Indispensable prerequisite is, however, that servers have an access interface intelligent enough to understand the user’s intention.

With WCS 2.0, OGC aims at providing a technologically up-to-date, extensible service definition allowing for efficient, scalable implementations. Extensive proofreading by scientists and practitioners from many different fields have contributed substantially to get as close as possible to the goals stated.

On specification level, we are working on adding further extension specifications in order of their relevance. A next significant task is establishing a WCS Earth Observation Application Profile. On scientific level, formalization of coverages with a goal of automatic consistency verification and conformance test generation is under work. Finally, efficient server implementations are being investigated.

9. ACKNOWLEDGMENTS

The author is indebted to the many people inside and outside OGC contributing discussion and constructive criticism, foremost Max Martinez, Clemens Portele, and Alex Robin. Further, the conceptual and implementation work of the Large-Scale Scientific Services Research Group at Jacobs University is gratefully acknowledged; let Andrei Aiordachioaie, Constantin Jucovschi, and

Figure 8. Containment relationship in coverage rangeType: concrete situation (left) and relationship types (right).

$$NDVI = \frac{(a.nir - a.red)}{(a.nir + a.red)}$$
Jinsongdi Yu be mentioned as repreentatives for the many con-
tributors.

10. REFERENCES
DBMS. Proc. EDBT’04, Heraklion, Greece, pp. 154-165
OGC document 09-146
OGC document 09-153r1
document 09-146r1
(WCPS) Interface Standard. OGC document 08-068r2
ing extension 1.0. OGC document 08-059r1
[7] Baumann, P.: Large-Scale Raster Services: A Case for Data-
bases (invited keynote). Proc. CoMoGIS’06, Santiago, Chile,
6 - 9 November 2006, LNCS 4231, Springer, pp. 75 – 84
[8] Howe, B. and Maier, D.: Algebraic Manipulation of Scientif-
ic Datasets. Proc. VLDB’04, Toronto, Canada, August 31 -
September 3 2004, pp. 924-935
[9] ISO (ed.): Geographic Information – coverage geometry and
functions. ISO 2005:19123
[10] ISO: Geographic information – Spatial referencing by co-
ordinates – Part 2: Extension for parametric values. ISO
2009:19111-2
connection – LOTOS – A formal description technique based
on the temporal ordering of observational behavior. ISO
1989:8807
[12] ISO: Information processing systems – Open Systems Inter-
connection – Estelle – A formal description technique based
on an Extended State Transition Model. ISO 1997:9074
[13] Jucovschi, C. and Baumann, P. and Stancu Mara, S.:
Speeding up Array Query Processing by Just-In-Time

APPENDIX: Sample GML Coverage Document Instance

```xml
<?xml version="1.0" encoding="UTF-8"?>
<gmlcov:GridCoverage xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xmns:gml='http://www.opengis.net/gml/3.2' xmlns:swe='http://www.opengis.net/swe/2.0'
xmns:gmlcov='http://www.opengis.net/gmlcov/1.0'
xmns:xlink='http://www.w3.org/1999/xlink'
xsi:schemaLocation='http://www.opengis.net/gmlcov/1.0 ../../../gmlcov/1.0/gmlcovAll.xsd'
gml:id='C0001'>
  <gml:boundedBy>
    <gml:Envelope srsName="http://www.opengis.net/def/crs/EPSG/0/4326" axisLabels="Lat Long"
      uomLabels="deg deg" srsDimension="2">
      <gml:lowerCorner>1 1</gml:lowerCorner>
      <gml:upperCorner>3 10</gml:upperCorner>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:domainSet>
    <gml:Grid gml:id="gr0001_C0001" dimension="2">
```

for Multidimensional Arrays: Design, implementation, and
Optimization Techniques. SIGMOD Rec. 25(2)1996, pp.
228-239
safety and equivalence. Proc ACM PODS’2007, Beijing,
China, June 11 – 14, 2007, pp. 175-184
arrays. Proc. VLDB’97, August 25 - 29, 1997, Athens,
Greece, pp. 46—55
Journal of Research and Development, 17(2)1973, pp. 135-
175
OGC document 06-080
access protocol, http://www.opendap.org; last seen:
September 13, 2010
erage geometry and functions. OGC document 07-111
Modular Specifications. OGC document 08-131r3
[22] Portele, C. (ed.): Geography Markup Language (GML)
Encoding Standard 3.2.1. OGC document 07-036
[23] Robin, A. (ed.): SWE Common 2.0. OGC document 08-094
Common Standard 2.0. OGC document 06-121r9
[25] Whiteside, A. (ed.): Web Coverage Service (WCS) – Trans-
action operation extension 1.1.4. OGC document 07-068r3
[26] Yu, J. and Wang, X. and Baumann, P.: A Specification-
Oriented Geospatial Coverage Ontology Study. Proc. ON-
TOSE’10, June 7 – 8, 2010, Hammamet, Tunisia, pp. 63-74