AGENCY INSTRUCTION

AI TECH 06.02.15

SERVICE INTERFACE PROFILE FOR GEOSPATIAL SERVICES – GEOPROCESSING SERVICES

Effective date:	2022-12-13 (*Precise date as per Approver's e-signature date*)
Revision No:	Original
Issued by:	Chief Joint Intelligence, Surveillance & Reconnaissance
Approved by:	Chief Technology Office

Digitally signed by Matt Roper
Date: 2022.12.16 17:07:25 +01'00'

Digitally signed by Antonio Calderon
Date: 2022.12.21 10:40:48 +01'00'

Table of Amendments

Amendment No	Date issued	Remarks

Author Details

Organization	Name	Contact Email/Phone
NCI Agency	Dr. Alexander Gruler	alexander.gruler@ncia.nato.int
52 North	Dr. Christoph Stasch, Benjamin Pross	

Table of Contents

AI TECH 06.02.15

Service Interface Profile for Geospatial Services – Geoprocessing Services

1 REFERENCES

A. NCI Agency Executive Structure – Directive 1 (Rev 3); dated Dec 2021
B. Agency Directive 01.01 - Agency Policy on Management and Control of Directives, Notices, Processes, Procedures and Instructions, dated 20 May 2014
C. Agency Directive 06.00.01 - Management and Control of Directives, Processes, Procedures and Instructions on Service Management, dated 03 June 2014

2 PURPOSE

This Agency Instruction (AI) specifies the Service Interface Profile (SIP) for Geoprocessing Services, which is a type of Geospatial Web Service used in the NATO Enterprise environment.

The purpose of the SIP for Geoprocessing Services is to ensure interoperability between providers and consumers of geoprocessing web services in the NATO IT Enterprise environment. The SIP profiles existing standards and provides directives, along with clarifications and amendments, on the use of these standards in the NATO IT Enterprise environment, and defines the mandatory and recommended interface operations to be implemented by any application, system, or Functional Area System (FAS) / Functional Service (FS) that offers or consumes geoprocessing services.

The SIP for Geoprocessing Services is not limited to software products or technologies of a particular vendor. The service interface is specified in an implementation-independent and vendor-neutral way which does not require technologies or products from one particular vendor for its implementation.

This AI specifies the first version of the SIP for Geoprocessing Services and does not supersede any previous version.

2.1 Agency Instructions

Technical AIs are living documents and will be periodically reviewed, updated, and made available to Agency staff as part of the CTO's responsibility as Design Authority. Technical content of these instructions is the shared responsibility of CTO Service Engineering and Architecture Branch and the Service Line of the discipline involved.

Technical AIs are primarily disseminated electronically and will be announced through new2know notices. Hard copies or local electronic copies should be checked against the current electronic version prior to use to assure that the latest instructions are used.

3 APPLICABILITY

This AI applies to all elements of the Agency, in particular to all NCI Agency staff involved in the development of software products for FASes, FSs, or other IT services. It is the responsibility of all NCI Agency Project Managers, Programme Managers, and Service Owners to ensure the implementation of this AI and to incorporate its content into relevant contractual documentation for external suppliers. In particular, this AI should be included as a requirement in the implementation of all future FASes and FSs which require such an interface.

4 SCOPE

4.1 In Scope

This SIP is intended to give directives, along with clarifications and amendments, on the use of mandatory and recommended interface operations and parameters to be implemented by any application, system, or FAS/FS within the NATO Enterprise that offers or consumes geoprocessing services.

This SIP addresses generic requirements that apply to any geoprocessing service, irrespectively of its concrete type or functionality, such as routing, viewshed analysis, etc.

4.2 Out of Scope

The following topics are out of scope of this SIP

- Recommendations for the use of products and platforms to implement this service interface.
- Specification of the interfaces of specific geoprocessing services that cover a specific use case, such as routing, the calculation of a viewsheds, etc. Instead, this SIP specifies the fundamental mechanism and web-service interface that is the basis for any more concrete geoprocessing service.
- Specification of any specific data required or output by the a specific geoprocessing service / WPS. Instead, this document identifies a generic mechanism to describe the data inputs required and produced by any geoprocessing service.
- Further refinement/restriction of the *complexType* in/out parameters in the `DescribeProcess` operation. Such refinements are reserved for subsequent, more specific SIPs which address specific geoprocessing services.
- Service Discovery mechanisms and related standards to discover specific geoprocessing services that are offered and hosted in an environment.
- Semantic mechanisms and concepts to determine if a specific geoprocessing services actually meets the requirements of a client, or the purpose for which the client likes to use the service.
- Orchestration concepts and mechanisms for geoprocessing services and web services (in general).
- Specification of any other interfaces that might be provided by a system which offers a geoprocessing service.

4.3 Audience

The target audience for this specification is the broad community of stakeholders who are delivering capabilities in the NATO IT Enterprise environment, or anticipate that their services may be used in this environment.

These may include (but are not limited to):

- Project Managers procuring Functional Area System (FAS), Functional Services (FS), or related systems for Bi-SC, ITM, or the NATO IT Enterprise environment.
- Architects and developers of service consumers and providers
- Coalition partners whose services may need to interact with services or systems in the NATO IT Enterprise environment, or the FMN environment.
- Systems integrators delivering systems into the NATO environment.

5 NOTATIONAL CONVENTIONS

The following notational conventions apply to this document:

The keywords "MUST", "MUST NOT", "SHALL", "SHALL NOT", "REQUIRED", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" represent requirements. Within this document these keywords are to be interpreted as described in [IETF RFC 2119, 1997].

In summary, this means that

- MUST, in the context of this SIP equivalent to SHALL or REQUIRED, means “that the specified item is an absolute requirement of the specification” [IETF RFC 2119, 1997].
- SHOULD, in the context of this SIP equivalent to RECOMMENDED, means that “there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course” [IETF RFC 2119, 1997].
- MAY, in the context of this SIP equivalent to OPTIONAL, means “that an item is truly optional” [IETF RFC 2119, 1997].

`Courier font` indicates syntactic elements, and names of web service operations.

XPath [W3C XPath 3.0, 2011] syntax is used to specifying the location of certain elements and attributes in an XML structure.

6 BACKGROUND

The majority of data used in NATO Command & Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR) systems has a spatial dimension attached to it, i.e. most of the data represents phenomena, events, artefacts, etc. which are associated with a certain geographic location or reference. In NATO, such data is typically called *geospatial data*. The processing of geospatial data requires specific algorithms, methods, knowledge, and tools, and is typically performed using *geographic information systems (GIS)* or components thereof. In this context, the processing of geospatial data is called *geoprocessing*. Geoprocessing is required in many application domains by domain specific systems, such as Command and Control (C2) systems for Air, Land, Maritime, or Logistics operations.

Geoprocessing services allow to specify and expose such geoprocessing functionality in form of web services which can be invoked by many clients and which fit into the web service enabled networking capabilities within the NATO Enterprise. Geoprocessing services are typically hosted on a GIS, such as NATO Core Geographic Information System (NATO Core GIS). This shifts the burden and cost of implementation and operating geospatial functions away from C2 systems, and avoids that the functionality that is a natural part of a GIS has to be replicated within the application logic of the C2 systems.

This SIP specifies an interface which is common to all geoprocessing services. Through this interface C4ISR systems can retrieve the exact information of how to syntactically interact with a specific instance of a particular geoprocessing service, as well as start its execution, and ultimately retrieve the results that are produced by a geoprocessing service.

ANNEX A: SERVICE INTERFACE PROFILE FOR GEOSPATIAL SERVICES: GEOPROCESSING SERVICES

This annex specifies the technical content of the Service Interface Profile for Geospatial Services - Geoprocessing Services.

The SIP specifies the web service interface, i.e. the rules and conventions that the providers (geoprocessing server) and consumers (geoprocessing client) of a geoprocessing service must follow in order to be able to interact successfully.

This SIP tries to avoid repetition, i.e. copying text passages, from the original standards as much as possible, and contains only the changes or amendments compared to the original standards which get profiled. For this reason, it is recommended to read this SIP together with the original standards.

1 INTRODUCTION

The majority of data used in NATO C4ISR systems has a spatial dimension attached to it, i.e. most of the data represents phenomena, events, artefacts, etc. which are associated with a certain geographic location or reference. In NATO, such data is called *geospatial data*.

Geoprocessing means processing of geospatial data. It considers the special characteristics of geospatial data and analysis. Geoprocessing ranges from simple GIS operations, such as handling positions in different coordinate systems, considering spatial (auto-)correlation, or computing the intersection of two spatial datasets, over more advanced GIS analysis tools, such as viewshed analysis or routing, up to complex environmental simulation models, such as a plume model for air pollutant dispersion.

The implementation of geoprocessing functionality is not trivial and typically requires geospatial experts and geographic information systems or components thereof. As an example, even a “simple” distance calculation is often implemented incorrectly for geographic coordinates, e.g. WGS84 coordinates, by using the Euclidean distance instead of the great circle distance[1].

Many FASes or FSs implement (similar) geoprocessing functions internally as part of their own application logic. In contrast, geoprocessing functions can be implemented (centrally) by a GIS, such NATO Core Geographic Information System (NATO Core GIS), and exposed via web service interfaces to FASes and FSs. These web services are specifically designed to meet the requirements of their consumers and communities of interest (COI), and enable clients to access geoprocessing functions without the need to own or manage the underlying implementation. In the context of this SIP such web services, i.e. services that implement geoprocessing functionality, are referred to as *geoprocessing services*.

The *OGC Web Processing Service (WPS)* Standard specifies a standardized web service interface for the description and execution of geoprocessing services. Thereby the WPS Standard does not address specific geoprocessing services for specific purposes, but instead defines the overarching interface and the concepts that standardize the use of (all kinds) of geoprocessing services in general, independently of their specific purpose and functionality.

Advantages of the WPS Standard are that it firstly allows to reuse and embed geoprocessing functionality in different client applications while keeping the same standardized mechanism to specify, execute, and retrieve the results from a geoprocessing service. This decouples changes in the implementation and the (syntactic) interface of a specific geoprocessing service from the way how a client uses a geoprocessing service and allows clients to flexibly consume geoprocessing services. This is in particular useful for NATO’s Command and Control Systems, where geoprocessing functionality

[1] The Euclidean distance can only be applied to projected coordinates, e.g. UTM coordinates, while the great circle distance considers geographic coordinates that are used for location information on a spheroid such as the earth.

may be added from WPS servers in a flexible manner. Secondly, many GIS applications already support the WPS interfaces for sharing geoprocessing functionality out of the box. Furthermore, the WPS facilitates the integration of different processing steps into complex geospatial information production workflows and makes the automation of such production workflows easier.

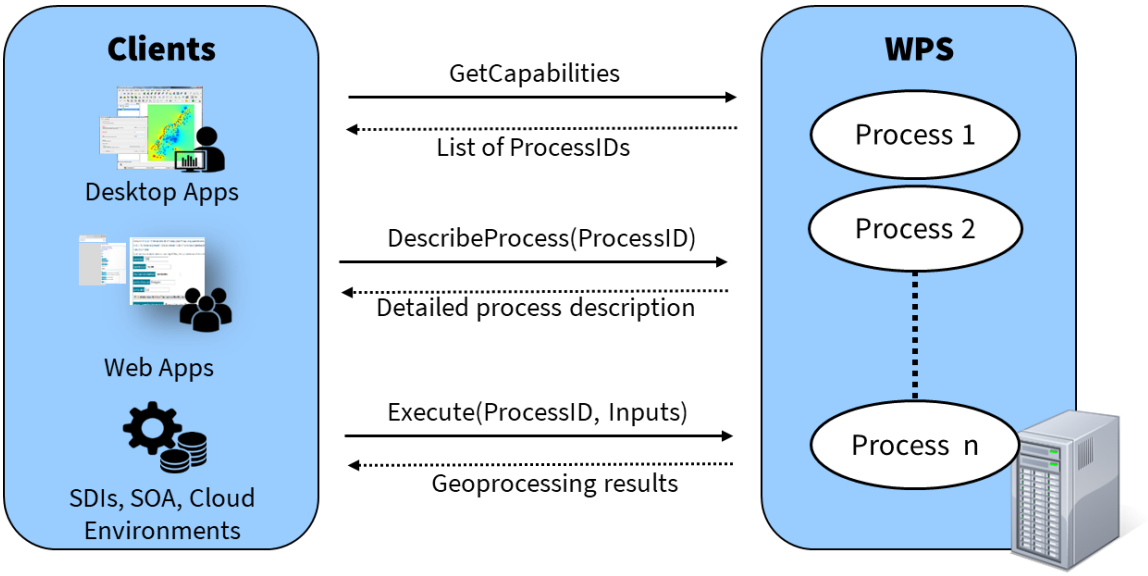

Figure 1 – OGC Web Processing Service operations and their conceptual use.

The WPS Standard defines three core operations for describing and executing processes: `GetCapabilities`, `DescribeProcess`, and `Execute`. Figure 1 illustrates the concept how these three operations are used. The `GetCapabilities` operation (Section 3.1.1) can be invoked to retrieve general service metadata, e.g. information about the service provider or access restrictions, as well as basic information about the processes offered including process identifiers and brief descriptions. If a certain process is of interest, the process identifier is used to retrieve the detailed process description using the `DescribeProcess` operation (Section 3.3). The detailed process description provides, besides general process metadata, detailed information about the mandatory and optional input and output parameters. It is used by clients to generate a form for trigging the execution of a process via the `Execute` operation (Section 3.4). To trigger an actual execution, the process identifier as well as other required input parameters need to be passed to the WPS server in an `Execute` request. Once the processing is finished, results are made available to the requesting client.

The WPS supports two parameter passing methods: Firstly, passing inputs and retrieving outputs directly encoded in requests/responses, or secondly passing them as references to web-accessible resources. Besides the synchronous execution of a process, the WPS also specifies how to run a process in an asynchronous execution mode. This is in particular useful for complex calculations on large datasets that may run for a longer time. In this case, the `Execute` operation creates a reference to a status document that is continuously updated. Once the process is completed, the process results are provided on specific location on the server from where they can be downloaded by clients.

1.1 WPS 1.0.0 vs WPS 2.0

At the time of writing this SIP there are two major versions of the OGC Web Processing Service Standard, WPS 1.0.0 [OGC WPS 1.0.0, 2007] and WPS 2.0 [OGC WPS 2.0, 2015].

WPS 2.0 is the more recent standard and a continuation of WPS 1.0. It incorporates a range of change requests that have been submitted since the release of WPS 1.0 and further follows the OGC policies for more modular specifications of standards.

WPS 2.0 addresses some shortcomings of WPS 1.0.0, especially related to the execution modes (asynchronous vs synchronous) and the passing of input and output data to and from a process ("by value" or "by reference"). While the realization of these concepts in WPS 1.0 is possible, WPS 2.0 introduces new attributes and operations which simplify the use of these concepts.

WPS 1.0 has several advantages:

- Firstly, WPS 1.0 is a widely adopted standard for which many commercial implementations exist. In contrast, only very few commercial products and client implementations are available for WPS 2.0, and this situation has not changed in the past couple of years.
- Secondly, WPS 1.0 contains in Annexes D (normative) and E (informative) a specification, together with guidance and examples, for using WPS with SOAP / WSDL [NCIA AI_06.02.06, 2015], which are fundamental standards for web services in the NATO IT Enterprise. In contrast, WPS 2.0 does not specify a mapping to SOAP/WSDL.

This SIP profiles the WPS 1.0 standard. However, the improvements that the WPS 2.0 standard makes with respect to execution modes (asynchronous vs synchronous) and parameter passing ("by value" or "by reference") have been considered in this SIP by profiling the WPS 1.0 elements in line with the WPS 2.0 philosophy, and by providing additional guidance and explanation, especially regarding the the execution modes and parameter passing methods.

1.2 Terminology

This SIP and its annexes base on the terminology introduced in the WPS 1.0 Standard (Section 4 of [OGC WPS 1.0.0, 2007]). In addition to the WPS 1.0 terminology the following terms are used

Table 1 Additional NATO SIP terminology

Term / Abbreviation	Definition
COI	Community of Interest
FAS	Functional Area System
feature (geospatial feature)	Abstraction of a real world phenomenon.
FS	Functional Service
geoprocessing	Processing of geospatial data.
geoprocessing client	Any entity, system, or application that consumes a geoprocessing service hosted by a geoprocessing server.
geoprocessing server	A server that runs one or more geoprocessing services and offers/provides them to clients.
geoprocessing service	The exposure of GIS functionality (GIS operations, algorithms operating on geospatial data) by a Geographic Information System (GIS) to clients through a web services interface. Note that a geoprocessing service may contain many sub-functions. These sub-functions are called *processes* in the OGC WPS terminology.

geospatial data	Data used to derive geospatial information, which are facts about the Earth, referenced by geographic position and arranged in a coherent structure. This includes topographic, aeronautical, hydrographic, planimetric, relief, thematic, geodetic, geo-referenced imagery, geophysical products, publications and materials. These are either in analogue or digital formats (based on MC 0296/3 "NATO Geospatial Policy").
geospatial information product	Specified data with direct or indirect spatial reference to be consumed by a user, e.g. maps, coverage data, or structured data that is returned as the result of a geospatial web service call from a producer/server to a client.
GIS	Geographic Information System
map (digital)	A portrayal of geospatial information in a graphical representation, primarily as a digital image, suitable for display on a computer screen.
NATO Cloud	See *NATO Enterprise*.
NATO Core GIS	NATO Core Geographic Information System
NATO Enterprise (environment)	Formally defined in [C3B AC/322-D, 2015] as : "The *NATO Enterprise* encompasses NATO entities and points of presence to which standardized NATO C3 capabilities must be delivered and ICT services must be provided for the fulfillment of NATO's goals and objectives; and for the conduct of NATO's daily business process activities, operations, training and exercises." Essentially this includes all NATO operational and business networks, such as the AIS, ON, PBN, or MS networks.
parameter (of a process)	A formal argument that is used in a process or subroutine to refer to an input or output that is provided to or returned by the process.
portrayal	Graphical presentation of information to humans.

1.3 **Namespaces**

The following namespaces are used to qualify XML elements in this document and its annexes, and the underlying OGC standards.

Table 2 Normative namespaces

Namespace Prefix	Namespace	Reference
nwps	nato:ncia:geo:proc	Specified in this SIP.
wps	http://www.opengis.net/wps/1.0.0	[OGC WPS 1.0.0, 2007]
ows	http://www.opengis.net/ows/1.1	OGC Web Services Common Standard Version 1.1

All OGC WPS 1.0 schemas are available online under http://schemas.opengis.net/wps/1.0.0/ .

1.4 Normative References

This SIP bases on the following public and open standards, and profiles them for the use within NATO.

Table 3 Normative references

Title	Version	Document No
OpenGIS® Web Processing Service	1.0.0	05-007r7
Corrigendum for OpenGIS® Implementation Standard Web Processing Service (WPS) 1.0.0 (0.0.8)	0.0.8	08-091r6
OGC Web Services Common Specification 1.1.0 with Corrigendum 1 *(Note that the WPS 1.0.0 Standard bases on this standard, and includes it as a normative reference, itself.)*	1.1.0	06-121r3

2 GENERAL REQUIREMENTS

This SIP specifies the interface of a geoprocessing service by profiling the OGC WPS Standard Version 1.0.0 [OGC WPS 1.0.0, 2007] with Corrigendum [OGC WPS 1.0.0 C, 2009], the OGC Web Services Common Standard Version 1.1.0 [OGC WSCom 1.1, 2007] with Corrigendum 2 [OGC WSCom 1.1 C2, 2011], and related standards.

Note that throughout this SIP the terms “WPS Standard” and “Web Services (WS) Common Standard” refer to the standards together with the respective corrigenda, and in the same version as listed in the Section 1.4.

This SIP applies to all systems in the NATO Enterprise environment that provide geoprocessing (web) services to consumers. This ensures that consumers can interact with any geoprocessing service via a WPS interface as profiled in this SIP.

Any NATO Functional Area Service (FAS), Functional Service (FS), or subsystem thereof that exposes geoprocessing services via its external interface to other systems in the NATO Enterprise environment MUST use the interfaces, standards and methods specified in this SIP.

Any geoprocessing server SHALL expose all of its geoprocessing services through a Web Processing Service (WPS) interface following the profile of the WPS Standard Version 1.0.0 [OGC WPS 1.0.0, 2007] with Corrigendum [OGC WPS 1.0.0 C, 2009] as specified in this document.

The OGC Web Services Common Standard specifies aspects that are common to all OGC Web Service interfaces, such as WMS, WMTS, and WPS. It provides additional details, makes requirements more precise, and aligns the WPS Standard with other OGC standards.

Any geoprocessing service SHALL implement and comply with the profile of the Web Services Common Standard Version 1.1.0 [OGC WSCom 1.1, 2007] and its Corrigendum [OGC WSCom 1.1 C2, 2011], as specified in this document.

Any aspects of the WPS Standard or the Web Services Common Standard that are not explicitly profiled in this SIP SHALL be realized according to the specification and requirements of the original OGC WPS and Web Services Common Standards.

NATO Core Geographic Information System (Core GIS) is one example of a system that provides geoprocessing services in a way that is conformant to the specification provided in this SIP.

2.1 Media/MIME Types for use with Geoprocessing Services

The OGC Discussion Paper [OGC WPS BPDP, 2009] contains best practices for identifying input and output data formats for use in the OGC WPS 1.0.0 by means of IANA Media / MIME Types.

The WPS 1.0.0 standard uses MIME Types [IETF RFC 2046, 1996] as one of the primary identification mechanisms for identifying the types of input and output values of geoprocessing services. However, currently there are no official MIME Types registered for most geospatial datasets (e.g. shapefile or geotiff) on the list of IANA Media Types.

Due to the lack of official IANA Media Types for most geospatial datasets this SIP recommends to use the MIME types for specific geospatial datasets in line with the OGC Best Practices Discussion Paper [OGC WPS BPDP, 2009].

In the absence of corresponding IANA Media Types (for geospatial datasets) it is RECOMMENDED to use the Media / MIME Types defined in [OGC WPS BPDP, 2009] for identifying geospatial data types when used for geoprocessing services within NATO.

2.2 Coordinate Reference Systems

Depending on the nature of a specific geoprocessing service, it might handle input or output data that is associated with a specific coordinate reference system (CRS).

In line with the WPS Standard this SIP does not mandate support for any particular CRS.

However, AGeoP-26 Edition B [NATO AGeoP-26 EdB, 202x] suggests a set of CRSs that should be supported as a minimum in order to increase interoperability with systems of NATO Nations and Partner Organizations:

- CRS:84; WGS84 geographic longitude, then latitude, expressed in decimal degrees
- EPSG:4326; WGS84 geographic latitude, then longitude, expressed in decimal degrees
- EPSG:3395; World Mercator projection
- Among the following CRSs the server shall support all those which validity zone overlaps the data published by the service:
    - EPSG:32601 to EPSG:32660; for UTM projections over WGS84 (north zones)
    - EPSG:32701 to EPSG:32760; for UTM projections over WGS84 (south zones)
    - EPSG:5041; for UPS projection over WGS84 (north zone)
    - EPSG:5042; for UPS projection over WGS84 (south zone)

It is RECOMMENDED that a geoprocessing service uses and supports the coordinate reference systems that are suggested in AGeoP-26 Edition B [NATO AGeoP-26 EdB, 202x] whenever applicable.

3 WPS INTERFACE

3.1 Operations

The WPS 1.0.0 interface contains the following three mandatory operations for the interaction between a geoprocessing server and consumers.

- `GetCapabilities`
- `DescribeProcess`
- `Execute`

A geoprocessing service SHALL offer all three mandatory operations, i.e. the `GetCapabilities`, `DescribeProcess`, and `Execute` operation, of the OGC WPS 1.0.0 interface with the changes and amendments that are specified in this SIP.

3.1.1 Operation Request and Response Encoding

Section 11 of [OGC WSCom 1.1, 2007] defines two ways of encoding operation requests, XML and keyword-value pairs (KVP). It also specifies how these encodings are used together with the HTTP GET and POST operations.

Regarding the encoding of operation requests and responses a geoprocessing service SHALL follow Section 11 of [OGC WSCom 1.1, 2007] with the following amendments and restrictions:

- HTTP GET requests where the parameters are encoded in the body of the GET request SHALL NOT be supported. The reason is that a payload within a GET request message has no defined semantics [IETF RFC 7231, 2014], and sending a payload body on a GET request might cause some existing implementations to reject the request [IETF RFC 7231, 2014].
  This means that the consumer of a geoprocessing service SHALL NOT send such requests to a server, and that a geoprocessing server SHOULD ignore such requests. Any action that a server might take in response to such requests is not further specified in this SIP and is at the discretion of the specific server implementation.
- A KVP encoding SHALL NOT be used in combination with a HTTP POST method.
  This means that the consumer of a geoprocessing service SHALL NOT send such requests to a server, and that a geoprocessing server SHOULD ignore such requests.

In summary, as a consequence of the restrictions specified above and the requirements from [OGC WSCom 1.1, 2007], the following two ways remain as the only valid possibilities to interact with a GeoProcessing Service:

- HTTP GET method with parameters encoded as KVP
- HTTP POST with parameters encoded as XML in the body of the POST request. Note that this includes SOAP encoded messages.

3.1.2 Capitalization of Parameter, Element, and Attribute Names

Regarding the capitalization of parameter, attribute, and element names a geoprocessing services SHALL follow [OGC WSCom 1.1, 2007]:

- For KVP encodings Section 11.5.2 of [OGC WSCom 1.1, 2007] applies, which states that the capitalization of parameter names shall be case insensitive.
- For XML encodings Section 11.6.2 of [OGC WSCom 1.1, 2007] applies, which states that UpperCamelCase shall be used for names of XML elements and values of XML parameters, and lowerCamelCase shall be used for names of XML attributes.

This means for example for the parameter names listed in Table 4 that the parameter “AcceptFormats” can equivalently be written as “acceptFormats” while still identifying the same parameter.

3.1.3 Exceptions

In addition to the exceptions specified in the WPS Standard a geoprocessing service MAY support more and/or more specific exceptions. These are listed in the remainder of this section.

3.1.3.1 Duplicated Parameters in a Request

When parameters of a WPS request (any operation) are duplicated with conflicting values, a geoprocessing service MAY respond with a specific exception. This applies to all three operations of the WPS Standard.

If responding with such an exception, the exception SHALL follow the requirements specified in Section 8 of [OGC WSCom 1.1, 2007] with the following restrictions:

- The exception type SHALL be *InvalidParameterValue*, i.e. the value of the `ows:Exception/@exceptionCode` element SHALL be the string *InvalidParameterValue* as specified in Table 25 of [OGC WSCom 1.1, 2007].
- The exception SHOULD contain the attribute `ows:Exception/@locator`. The value of the attribute `locator` SHALL specify the name and both conflicting values of the duplicated parameters.
- The exception SHOULD contain textual information stating that the cause for raising this exception is the existence of duplicated parameters in the operation request. The content of the element `ows:Exception/ows:ExceptionText` SHALL be used for this purpose. In order to indicate the language in which the exception text is composed the corresponding exception report SHOULD contain the attribute `ows:ExceptionReport/@xml:lang` whose value SHALL match the language in which the exception text is composed. The implementation of the `xml:lang` attribute SHALL follow the implementation of the *Language* parameter in Table 4.

3.1.4 Consistency and Completeness of Information about Process IDs

As described in Figure 1 all three WPS operations (`GetCapabilities`, `DescribeProcess`, and `Execute`) base on process identifier parameters.

A WPS server SHALL list the process identifiers of all processes which it exhibits to clients in the capabilities document by means of `wps:ProcessOfferings/wps:Process/ows:Identifier` sub-elements.

A WPS server SHALL ensure that the following sets of process identifiers are identical:

- The set of process identifiers that are returned in the response of a the `GetCapabilities` operation, using `wps:ProcessOfferings/wps:Process/ows:Identifier` sub-elements.
- The set of process identifiers that can be used in the `DescribeProcess` operation request to get more information about the corresponding process, using `wps:DescribeProcess/ows:Identifier` sub-elements.
- The set of process identifiers that can be used in the `Execute` operation request to invoke the corresponding process, using `wps:Execute/ows:Identifier` sub-elements.

3.2 GetCapabilites Operation

The purpose of the `GetCapabilities` operation is to obtain service metadata and value ranges for request parameters. Especially in a WPS service the operation is used by clients to retrieve the `ProcessID` of the geoprocessing processes hosted by a geoprocessing server.

3.2.1 **Request**

The `GetCapabilities` operation request SHALL follow the requirements specified in Section 8.2 of the WPS Standard with the modifications and amendments specified in Table 4 below.

Table 4 Parameters for a GetCapabilities request.

Parameter Name and Status	**Requirement**
service (mandatory; frozen)	No changes compared to [OGC WPS 1.0.0, 2007]: As specified in [OGC WPS 1.0.0, 2007], Table 6, Section 8.2, the value of this parameter is frozen and shall be "WPS".
Request (mandatory; frozen)	No changes compared to [OGC WPS 1.0.0, 2007]: As specified in [OGC WPS 1.0.0, 2007], Table 6, Section 8.2, the value of this parameter is frozen and shall be "GetCapabilities".
AcceptVersions (optional)	A geoprocessing server MAY support version negotiation. If version negotiation is supported the *AcceptVersions* parameter SHALL be used as specified in Sections 7.2 and 7.3 of [OGC WSCom 1.1, 2007] (No change compared to [OGC WPS 1.0.0, 2007]).
language (optional)	This parameter SHALL be a language identifier which follows the encoding specified in [IETF RFC 4646, 2006] (No change compared to [OGC WPS 1.0.0, 2007]). The language tags from the language code list that is mandated by the NGMP [NATO AGeoP-8EdB, 2019][2] SHALL be used for the value domain of this parameter. For the country code identifiers that are part of a RFC 4646 conformant language tag the NATO list of country codes from Annex A of [STANAG1059ed08, 2007] [3] SHALL be used.

[2] The language codelist required by Edition B of the NGMP [NATO AGeoP-8EdB, 2019] is based on the codelist specified by ISO 639-2, "Codes for the representation of names of languages – Part 2: Alpha-3 code" with some NATO specific adaptations. RFC 4646 also uses the ISO 639-2 language tags for representing three-character language subtags. While the preceding Edition A of the NGMP [NATO AGeoP-8, 2016] still contained a copy of the language code list in Append A-4, the current Edition B of the NGMP [NATO AGeoP-8EdB, 2019] does no longer contain a copy of the codelist itself, but refers to other standards.

[3] Note that STANAG 1059 and IETF RFC 4646 [IETF RFC 4646, 2006] both base on ISO 3166-1, "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes"". The STANAG 1059 establishes a snaphshot of the ISO country codes in ISO 3166-1 since ISO 3166-1 is updated frequently.

Sections (OPTIONAL)	This parameter is OPTIONAL for a WPS `GetCapabilities` request. If it is implemented it is RECOMMENDED to follow the guidance and specification in Sections 7.2 and 7.3 of [OGC WSCom 1.1, 2007]. Although the parameter *Sections* is specified in Sections 7.2 and 7.3 of [OGC WSCom 1.1, 2007], it is not explicitly listed in [OGC WPS 1.0.0, 2007], Table 6, Section 8.2, nor is it listed in the XML schema of the *wpsGetCapabilities_request.xs*.
AcceptFormats (OPTIONAL)	Same requirement as for parameter *sections* above.
updateSequence (OPTIONAL)	Same requirement as for parameter *sections* above.

3.2.1.1 Bindings and Encodings

A geoprocessing server exhibits the `GetCapabilities` operation in the following two ways:

- Firstly, the `GetCapabilities` operation of any geoprocessing service SHALL be offered by a geoprocessing server through an HTTP GET method using KVP encoding for the parameters.
- Secondly, the `GetCapabilities` operation of any geoprocessing service SHALL be offered by a geoprocessing server through an HTTP POST method where the parameters are encoded as XML in the body of the POST request.
  Note that this is a more restrictive requirement compared to [OGC WPS 1.0.0, 2007] in which the POST operation with XML is optional but not mandatory.

For the implementation of both encodings a geoprocessing server SHALL follow [OGC WSCom 1.1, 2007], in particular Section 11 of [OGC WSCom 1.1, 2007] with the modifications and restrictions specified in Section 3.1.1.

3.2.2 Response

The response of a `GetCapabilities` request SHALL follow the requirements specified in Section 8.3 of WPS with the exceptions and amendments specified in Table 5 below.

Table 5 Parameters for a GetCapabilities response.

Parameter Name and Status	Requirement
service (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
version (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
updateSequence (optional)	No changes compared to [OGC WPS 1.0.0, 2007], which specifies that this parameter shall be implemented as specified in [OGC WSCom 1.1, 2007].

lang (mandatory)	This parameter SHALL be implemented in the same way as the l*anguage* parameter in Table 4.
ServiceIdentification (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007], which specifies that the XML schema of this parameter shall follow the one specified in [OGC WSCom 1.1, 2007]. All data provided in this element SHALL follow the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019], i.e. the value domains, semantics, and guidance of the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019] SHALL be used for all sub-elements and attributes, wherever applicable. The values for the *xml:lang* attribute for all sub-elements of the ows:ServiceIdentification element, such as ows:Title, ows:Abstract, and ows:Keyword, SHOULD follow the encoding specified in [IETF RFC 4646, 2006] and SHOULD use values of the language code list mandated by the NGMP [NATO AGeoP-8EdB, 2019] (which is an extract of ISO 639-2).
ServiceProvider (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007], which specifies that the XML schema of this parameter shall follow the one specified in [OGC WSCom 1.1, 2007]. All data provided in this element SHALL follow the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019], i.e. the value domains, semantics, and guidance of the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019] SHALL be used for all sub-elements and attributes, wherever applicable.
OperationsMetadata (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007], Section 8.3.2. All data provided in this element SHALL follow the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019], i.e. the value domains, semantics, and guidance of the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019] SHALL be used for all sub-elements and attributes, wherever applicable.
Languages (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007], Section 8.3.4. The value of any */wps:Languages//ows:Language* sub-element SHALL be implemented in the same way as the *Language* parameter in Table 4.
WSDL (optional)	No changes compared to [OGC WPS 1.0.0, 2007].
ProcessOfferings (mandatory)	This element and its sub-elements SHALL be implemented as specified in Table 6 below.

Table 6 Elements of a ProcessBriefType structure.

Parameter Name and Status	Requirement
Identifier (mandatory)	The `wps:ProcessOfferings/wps:Process/ows:Identifier` sub-element SHALL be implemented in the same way as the I*dentifier* parameter in Table 7.
Title (mandatory)	The values for the *xml:lang* attribute SHOULD follow the encoding specified in [IETF RFC 4646, 2006] and SHOULD use values of the language code list mandated by the NGMP [NATO AGeoP-8EdB, 2019] (which is an extract of ISO 639-2).
Abstract (optional)	The values for the *xml:lang* attribute SHOULD follow the encoding specified in [IETF RFC 4646, 2006] and SHOULD use values of the language code list specified by the NGMP [NATO AGeoP-8EdB, 2019] (which is an extract of ISO 639-2).
Metadata (optional)	All data provided in this element SHALL follow the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019], i.e. the value domains, semantics, and guidance of the NATO Geospatial Metadata Profile [NATO AGeoP-8EdB, 2019] SHALL be used for all sub-elements and attributes, wherever applicable.
Profile (optional)	No changes compared to [OGC WPS 1.0.0, 2007].
WSDL (optional)	No changes compared to [OGC WPS 1.0.0, 2007].
processVersion (use is depending on context)	No changes compared to [OGC WPS 1.0.0, 2007]. Note that this attribute • MAY exist when used in a `Capabilities` structure, i.e. when used as `/wps:Capabilities/ProcessOfferings/wps:Process/@wps:processVersion`, according to [OGC WPS 1.0.0, 2007], Table 2. • SHALL exist when used in a `ProcessDescription` structure, i.e. when used as `/wps:ProcessDescriptions/ProcessDescription/@wps:processVersion`, according to [OGC WPS 1.0.0, 2007], Table 16.

3.3 DescribeProcess Operation

In order to use the `Execute` operation a WPS client needs to know details about the process which should be executed, such as the input and output parameters, the respective formats, etc. The

`DescribeProcess` operation allows WPS clients to request a comprehensive description of one or more processes, including all the information that is required to execute a process.

3.3.1 **Request**

The `DescribeProcess` operation request SHALL follow the requirements specified in Section 9.2 of the WPS Standard [OGC WPS 1.0.0, 2007] with the modifications and amendments specified in Table 7, below.

Table 7 Parameters for a DescribeProcess request.

Parameter Name and Status	**Requirement**
service (mandatory; frozen)	No changes compared to [OGC WPS 1.0.0, 2007]: As specified in [OGC WPS 1.0.0, 2007], Table 13, Section 9.2, the value of this parameter is frozen and shall be “WPS”.
Request (mandatory; frozen)	No changes compared to [OGC WPS 1.0.0, 2007]: As specified in [OGC WPS 1.0.0, 2007], Table 13, Section 9.2, the value of this parameter is frozen and shall be “DescribeProcess”.
version (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
language (optional)	This parameter is OPTIONAL for a geoprocessing service. If implemented, this parameter SHALL be implemented in the same way as the *language* parameter in Table 4.
Identifier (mandatory)	In line with [OGC WPS 1.0.0, 2007], the *Identifier* parameter SHALL be an unambiguous identifier or name of a process that is offered by the geoprocessing server. The value SHALL be unique within a geoprocessing server. It is RECOMMENDED to use a URN for encoding the value of an I*dentifier* parameter. For example, identifiers of geoprocessing services hosted by an instance of NATO Core GIS MAY have the namespace `nato:ncia:geo:proc` (as defined in Table 2) as a prefix of the value.

3.3.1.1 **Bindings and Encodings**

A geoprocessing server exhibits the `DescribeProcess` operation in the following two ways:

- Firstly, the `DescribeProcess` operation of any geoprocessing service SHALL be offered by a geoprocessing server through an HTTP GET method using KVP encoding as specified in Section 9.2.2 of [OGC WPS 1.0.0, 2007].
  The value “ALL”, i.e. Identifier=ALL, MAY be used to request process descriptions for all processes that are supported by a geoprocessing server, but – in contrary to [OGC WPS 1.0.0, 2007], Table 14, Footnote B – the use of such a value is NOT RECOMMENDED for a geoprocessing service (as it is not in line with the requirement to use a true URN for encoding the value of the *Identifier* parameter; See Table 7).

- Secondly, the `DescribeProcess` operation of any geoprocessing service SHALL be offered by a geoprocessing server through an HTTP POST method where the parameters are encoded as XML in the body of the POST request.
  Note that this is a more restrictive requirement compared to [OGC WPS 1.0.0, 2007] in which the HTTP POST method (with XML) is truly optional but not mandatory.
  If XML encoding is used for the request message it MUST comply with the XML schema specified in Section 9.2.3 of [OGC WPS 1.0.0, 2007].

For the implementation of both encodings a geoprocessing server SHALL follow [OGC WSCom 1.1, 2007], in particular Section 11 of [OGC WSCom 1.1, 2007] with the modifications and restrictions specified in Section 3.1.1.

3.3.2 **Response**

The response of a `DescribeProcess` request SHALL be a `ProcessDescriptions` XML data structure following the requirements specified in Section 9.3 of [OGC WPS 1.0.0, 2007] with the exceptions and amendments specified in the remainder of this section.

Table 8 Parameters for a DescribeProcess response.

Parameter Name and Status	**Requirement**
service (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
version (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
lang (mandatory)	This parameter SHALL be implemented in the same way as the *language* parameter in Table 4.
ProcessDescription (mandatory)	This element includes firstly the brief information (about a specific process) that is part of the service metadata (Capabilities) document of the `GetCapabilities` request, and secondly more information about the data input and output parameters. This element and its sub-elements SHALL be implemented as specified in the WPS Standard with the changes (for its sub-elements) listed in Table 9 below, and Table 6 above.

Table 9 Subset of sub-elements of a ProcessDescription parameter.

Parameter Name and Status	**Requirement**

DataInputs (optional; mandatory for all non-predefined process inputs)	Although declared optional, this element SHALL always exist and be defined, unless for processes where all inputs are predetermined fixed resources that are identified and described in the *Abstract* parameter. In that case this parameter MAY be omitted. The sub-elements *Identifier*, *Title*, *Abstract*, and *Metadata* SHALL be implemented as specified in Table 6 above. The remaining sub-elements shall be implemented as specified in Table 11.
ProcessOutputs (mandatory)	The sub-elements *Identifier*, *Title*, *Abstract*, and *Metadata* SHALL be implemented as specified in Table 6 above. The remaining sub-element *OutputFormChoice* SHALL be implemented as specified in Table 13.
storeSupported (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. Note that the boolean *storeSupported* attribute has been removed in WPS 2.0. In WPS 2.0 it is conceptually replaced by the new parameter *outputTransmission* which can take the two values "*value*" and "*reference*".
statusSupported (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. Note Section 2.2 of WPS Corrigendum [OGC WPS 1.0.0 C, 2009], which states that "*In order for a server to support status reporting it must also be able to store process outputs. I.e. "statusSupported" can be "true" only if "storeSupported" is also "true*".

The attribute *storeSupported* of the DescribeProcess response is used to declare support of the execution mode: The value "false" corresponds to synchronous execution, while the value "true" corresponds to asynchronous execution (See [OGC WPS 1.0.0, 2007], Table 16, Footnote b).

Regarding the attributes *storeSupported* and *statusSupported* (specified in Table 9) a geoprocessing server SHOULD only support the combinations of values that is specified in Table 10 below. These combinations reproduce the improved modes of execution (asynchronous vs synchronous) and the passing of input and output data to and from a process ("by value" or "by reference") which are defined in the WPS 2.0 Standard [OGC WPS 2.0, 2015] (Sections 6.5 – 6.7).

Table 10 Supported combinations of values for the attributes *statusSupported* and *storeSupported*.

Value of *storeSupported*	**Value of *statusSupported***	**Combination allowed**
true	true	allowed
true	false	not allowed
false	true	not allowed
false	false	allowed

Table 11 Subset of sub-elements of a DataInputs parameter.

Parameter Name and Status	Requirement
minOccurs (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
maxOccurs (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
InputFormChoice (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007]. The sub-elements of this parameter SHALL be implemented as defined in Table 12.

Table 12 Sub-elements of the InputFormChoice element.

Parameter Name and Status	Requirement
ComplexData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. For the value domains of all `MimeType` sub-elements, i.e. the sub-elements `ComplexData/Default/Format/MimeType` and `ComplexData/Supported/Format/MimeType`, the requirements specified in Section 2.1 SHALL be applied.
LiteralData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].
BoundingBoxData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].

Table 13 Sub-elements of the OutputFormChoice element.

Parameter Name and Status	Requirement
ComplexData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. For the value domains of all `MimeType` sub-elements, i.e. the sub-elements `ComplexData/Default/Format/MimeType` and `ComplexData/Supported/Format/MimeType`, the requirements specified in Section 2.1 SHALL be applied.

LiteralData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].
BoundingBoxData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].

3.4 **Execute Operation**

The `Execute` operation allows WPS clients to start the execution of a process on a geoprocessing server. Inputs and/or outputs can be included directly in the `Execute` request/response, or indirectly as a reference to a web accessible resource.

A process can be executed in two ways:

1. Synchronously, meaning that the response is available only after the process is finished. The client that sent the `Execute` request has to keep the connection open during execution. Since the output is returned as part of the `Execute` response it is not available anymore after the completion of the operation call.
2. Asynchronously, meaning that a response is returned directly after the `Execute` request was sent by the client. However, this response does not contain the final outputs but rather contains an URL that points to an XML element that provides the status of the computation, and – once the process execution has been completed – the outputs of the process. The client can poll this URL repeatedly until the process has finished and the server makes the final outputs available at the URL. The `Execute` response containing the outputs remains available even after the computation of the geoprocessing process is completed.

Note that the WPS 1.0 standard does not have an explicit parameter that specifies if a process should be executed synchronously or asynchronously. Instead, WPS 1.0 allows to choose between these two execution modes only indirectly by means of the Parameter `/wps:Execute/wps:ResponseForm/wps:ResponseDocument/@storeExecuteResponse`. The same parameter is also used to define if the results that are produced by the `Execute` operation should be stored as a web accessible resource instead of returning it as an XML document as part of the `Execute` operation response.

Outputs of a process execution can be requested in three different ways:

1. As raw data. This can be used when **synchronously** executing a process with **one** output. The output is returned after the process is finished directly in the HTTP response. This is useful for requesting small to large sized text-based or binary complex outputs for direct consumption.
2. As value in the `Execute` response XML. This can be used when **synchronously** or **asynchronously** executing a process with **one** or **more** outputs. This is useful for requesting text-based complex outputs (e.g. Well-Known-Text, JSON or GML), literal and bounding box outputs. This can also be used to request small to medium sized binary complex outputs. In this case the output SHOULD be encoded in Base64 [IETF RFC 4648, 2006].
3. As reference in the `Execute` response XML. This is typically used when **synchronously** or **asynchronously** executing a process with **one** or **more** outputs. This is useful for requesting one or more small to large sized binary or text-based complex outputs.

3.4.1 **Request**

The `Execute` operation request SHALL follow the requirements specified in Section 10.2 of the WPS Standard [OGC WPS 1.0.0, 2007] and the applicable corrections from [OGC WPS 1.0.0 C, 2009], with the modifications and amendments specified in the remainder of this section.

In order to trigger asynchronous execution the `storeExecuteResponse` parameter in the `Execute` request (i.e. `/wps:Execute/wps:ResponseForm/wps:ResponseDocument/@storeExecuteResponse`) SHALL be set to *true*.

Table 14 Parameters for an Execute request.

Parameter Name and Status	Requirement
service (mandatory; frozen)	No changes compared to [OGC WPS 1.0.0, 2007]: As specified in [OGC WPS 1.0.0, 2007], Table 39, Section 10.2, the value of this parameter is frozen and shall be "WPS".
Request (mandatory; frozen)	No changes compared to [OGC WPS 1.0.0, 2007]: As specified in [OGC WPS 1.0.0, 2007], Table 39, Section 10.2, the value of this parameter is frozen and shall be "Execute".
version (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
language (optional)	This parameter is OPTIONAL for a geoprocessing service. If implemented, this parameter SHALL be implemented in the same way as the *language* parameter in Table 4.
Identifier (mandatory)	This parameter is used to identify the process that shall be executed. The process SHALL be one of the process which is listed in the corresponding parts of the `DescribeProcess` operation response. For the parameter value the same requirements apply as specified for the *Identifier* parameter in Table 7.
DataInputs (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. The sub-elements *Identifier*, *Title*, and *Abstract* located at `wps:DataInputs/wps:Input/` SHALL be implemented as specified in Table 6 above. For the value domains of all `MimeType` sub-elements, i.e. the sub-element `wps:DataInputs/wps:Input/wps:Reference @mimeType` the requirements specified in Section 2.1 SHALL be applied. The sub-elements of the parameter `wps:DataInputs/wps:Input/wps:Data/` SHALL be implemented as defined in Table 15 below.
ResponseForm (optional)	No changes compared to [OGC WPS 1.0.0, 2007]. The sub-elements of the `ResponseForm` SHALL be implemented as defined in Table 16 below.

Table 15 Sub-elements of the wps:Input/wps:Data element.

Parameter Name and Status	Requirement
ComplexData (optional; conditional)	No syntactic changes compared to the WPS Standard. For the value domains of all `MimeType` sub-elements, i.e. the attribute `wps:ComplexData/@mimeType`, the requirements specified in Section 2.1 SHALL be applied.
LiteralData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].
BoundingBoxData (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].

Table 16 Sub-elements of the ResponseForm element.

Parameter Name and Status	Requirement
ResponseDocument (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. For asynchronous execution the `wps:ResponseDocument/@storeExecuteResponse` sub-element SHALL be set to true. The parameter *Output* and its sub-elements SHALL be implemented as specified in Table 17 below.
RawDataOutput (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007]. For the *Identifier* sub-element the same requirements apply as specified for the I*dentifier* parameter in Table 7. For the value domains of all `mimeType` sub-element the requirements specified in Section 2.1 SHALL be applied.

Table 17 Sub-elements of the Output element (of type wps:DocumentOutputDefinitionType).

Parameter Name and Status	Requirement

Output	No changes compared to [OGC WPS 1.0.0, 2007].
(mandatory)	The sub-elements *Title*, and *Abstract* SHALL be implemented as specified in Table 6 above. For the *Identifier* sub-element the same requirements as specified for the I*dentifier* parameter in Table 7 SHALL be applied. For the value domains of all `mimeType` sub-element the requirements specified in Section 2.1 SHALL be applied.

3.4.1.1 Bindings and Encodings

A geoprocessing server exhibits the `Execute` operation in the following two ways:

- Firstly, the Execute operation of any geoprocessing service SHALL be offered by a geoprocessing server through an HTTP POST method where the parameters are encoded as XML in the body of the POST request. The XML operation request SHALL comply with the XML schema that is provided in Section 10.2.3 of [OGC WPS 1.0.0, 2007], taking into account the corrections made in [OGC WPS 1.0.0 C, 2009], Sections 2.3.
- Secondly, the `Execute` operation of any geoprocessing service MAY be offered by a geoprocessing server through an HTTP GET method using KVP encoding as specified in Section 10.2.2 of [OGC WPS 1.0.0, 2007] and the respective sections of [OGC WPS 1.0.0 C, 2009], most noticeably Sections 2.6, 2.7, 2.8, and 2.14.

For the implementation of both encodings a geoprocessing server SHALL follow [OGC WSCom 1.1, 2007], in particular Section 11 of [OGC WSCom 1.1, 2007] with the modifications and restrictions specified in Section 3.1.1.

3.4.2 Response

The response of an `Execute` request SHALL follow the requirements specified in Section 10.3 of [OGC WPS 1.0.0, 2007] and [OGC WPS 1.0.0 C, 2009] (especially Sections 2.9, 2.10, and 2.11) with the exceptions and amendments specified in the remainder of this section.

The `Execute` operation of a (successfully finished) process can have two forms:

- Raw data, meaning that **one** output is returned directly in the response in the requested format, e.g. binary files (e.g. images, PDFs, zip-files) or text-based output like CSV, JSON or GML. Note that this kind of response is triggered by submitting a `/wps:Execute/wps:ResponseForm/wps:RawDataOutput` element in the `Execute` request.
- A response encoded in XML, meaning **one** or **more** outputs are returned wrapped in a wps:`ExecuteResponse` XML element. Each output value can be returned directly included in the XML or as reference to a web-accessible resource. Note that this kind of response is triggered by submitting a `/wps:Execute/wps:ResponseForm/wps:ResponseDocument` element in the `Execute` request.
  In this case the response message SHALL comply with the XML schema and requirements specified in Section 10.3 of the WPS Standard [OGC WPS 1.0.0, 2007], taking into account the corrections made in [OGC WPS 1.0.0 C, 2009] (especially in Sections 2.4 and 2.11), and the requirements specified in Table 18 and in the remainder of this section.

Table 18 Parameters in an ExecuteResponse element.

Parameter Name and Status	Requirement
service (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
version (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
lang (mandatory)	This parameter SHALL be implemented in the same way as the *language* parameter in Table 4.
serviceInstance (mandatory)	No changes compared to [OGC WPS 1.0.0, 2007].
statusLocation (optional; conditional)	No changes compared to [OGC WPS 1.0.0, 2007].
Process (mandatory)	This parameter SHALL contain the description of the process as advertised in the `ProcessOfferings` section of the `GetCapabilities` response. This element and its sub-elements SHALL be implemented in the same way as specified in Table 6.
Status (mandatory)	This parameter SHALL be implemented as specified in the [OGC WPS 1.0.0, 2007] with the following modifications: For asynchronous execution the WPS 1.0 standard does not specify the time how long the results will remain available under the store URL after the have been produced. It is RECOMMENDED that the geoprocessing server keeps hosting the results at the corresponding URL at least until they have been requested once since the status of the producing process has been changed to "succeeded", i.e. the `wps:ExecuteResponse/wps:Status` element has a child element called `wps:ProcessSucceeded`. If present, the sub-element `wps:Status/wpsProcessFailed/ows:ExceptionReport/@xml:lang` SHALL be implemented in the same way as the *language* parameter in Table 4.
DataInputs (optional; conditional)	This parameter is used to record the inputs that were provided when invoking the `Execute` request. It SHALL be omitted unless the attribute `wps:Execute/wps:ResponseForm/wps:ResponseDocument/@lineage` of the `Execute` request is set to "true". This parameter SHALL be implemented in the same way as the *DataInputs* parameter in Table 14 above.

OutputDefinitions (optional; conditional)	This parameter SHALL be omitted unless the attribute `wps:Execute/wps:ResponseForm/wps:ResponseDocument/@lineage` of the `Execute` request is set to "true". The sub-element *Output* and its sub-elements SHALL be implemented as specified in Table 17 above.
ProcessOutputs	In all `wps:Output` sub-elements the elements *Title*, *Abstract*, Metadata and *Identifier* SHALL be implemented as specified in Table 6 above. For the value domains of all `MimeType` sub-elements, i.e. the sub-element `wps:Output/wps:reference/@mimeType` the requirements specified in Section 2.1 SHALL be applied. The sub-elements of the parameter `wps:Output/wps:Data/` SHALL be implemented as defined in Table 15 above.

4 TEST SUITE

This section specifies a test suite for verifying the requirements of the SIP for Geoprocessing Services. The test suite consists of a set of web services operation calls and expected results. As such the tests serve as templates for executable and repeatable tests for any existing automated testing tools and platform.

The following tests represent the normative test suite for testing compliance to the SIP for Geoprocessing Services, and as such replace Annex A.3 of the WPS 1.0.0 Standard [OGC WPS 1.0.0, 2007], which is only considered informative for this SIP.

Note that this test suite does not specify any dedicated tests for **clients**, but is limited to testing server implementations. In addition to this test suite the tests defined for clients in Annex A.2 of the WPS 1.0.0 Standard [OGC WPS 1.0.0, 2007] should be considered as an informative reference for **client**-testing of this SIP.

4.1 Background

The WPS 1.0.0 Standard [OGC WPS 1.0.0, 2007], Annex A, contains an abstract test suite that specifies at a high level how server and client implementations can be checked for conformance with the WPS specification. However, the test suite is abstract and does not contain sufficient details to be used as the basis for repeatable, executable, and automated tests.

The WPS 2.0 Standard [OGC WPS 2.0, 2015] contains a test suite that bases on the tests suite of WPS 1.0. It refines it with more details and extends it with additional tests that mainly address the additional operations that are introduced in WPS 2.0.

This section

- Covers all tests defined in WPS 1.0 and refines them with the aim to make them executable in support of automated and repeatable testing.
- Covers selected test cases from the WPS 2.0 Standard that are relevant for testing execution modes (synchronous vs asynchronous) and data transmission modes (parameter data "by value" and "by reference").
- Refines all of the above tests cases and specifies a sufficient (higher) level of detail for the server implementation to be implemented as black box tests for implementations to ensure conformance to this SIP.

4.2 Basic WPS Server Test Module

4.2.1 HTTP Protocol Usage

The rules and conventions governing the use of HTTP SHALL be tested as specified in A.4.1.1 of the WPS Standard [OGC WPS 1.0.0, 2007].

4.2.2 HTTP Response Status Code

The status code in a response to a request that causes an exception on the server side SHALL be tested as specified in A.4.1.2 of the WPS Standard [OGC WPS 1.0.0, 2007].

4.2.3 Request Service Parameter

Test Purpose	Verify that a geoprocessing server correctly handles the `service` parameter.
Test Method	For each request type, send valid requests to server under test. Modulate `service` parameter: • Parameter value equal to “WPS”. Verify that request succeeds. • Parameter value not equal to “WPS”, e.g. “AnotherService”. Verify that request fails. Overall test passes if all individual tests pass.
Reference	3.1.1 3.1.2 3.2.1

4.2.4 Version Negotiation

The correct handling of the version parameter SHALL be tested as specified in A.4.2.4 of the WPS Standard [OGC WPS 1.0.0, 2007].

4.2.5 Language Selection

Test Purpose	Verify that a geoprocessing server handles the `language` parameter as specified in Section 3.2.1 of this SIP.
Test Method	See A.4.2.6 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.2.6 of the WPS Standard [OGC WPS 1.0.0, 2007] • Section 3.2.1, Table 4, of this SIP3.2.1.1

4.3 `GetCapabilities` Operation Test Module

4.3.1 HTTP GET `GetCapabilities` KVP Request

Test Purpose	Verify that a geoprocessing server accepts `GetCapabilities` requests transferred through HTTP GET as specified in Section 3.2.1.1.
Test Method	See A.4.2.1 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.2.1 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.2.1.1

4.3.2 HTTP POST `GetCapabilities` XML Request

Test Purpose	Verify that a geoprocessing server accepts `GetCapabilities` requests transferred through HTTP POST as specified in Section 3.2.1.1.
Test Method	See A.4.2.2 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.2.2 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.2.1.1

4.3.3 `GetCapabilities` Operation Response

Test Purpose	Verify that a geoprocessing server satisfies all requirements on the `GetCapabilities` operation response as specified in Section 3.2.2.
Test Method	See A.4.2.3 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.2.3 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.2.2

4.3.4 Handling `updateSequence` Parameter

Test Purpose	Verify that a geoprocessing server handles the `updateSequence` parameter as specified in Section 3.2.
Test Method	See A.4.2.5 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.2.5 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.2.2

4.3.5 Unique Process Identifier

Test Purpose	Verify that each process the server offers has an unique identifier.
Test Method	Send a valid `GetCapabilities` request to server under test. The Capabilities documents lists the processes available in the `wps:ProcessOfferings` sub-element. Check that the identifiers for the processes offered in the elements at `wps:ProcessOfferings/wps:Process/ows:Identifier` are unique.
Reference	• 3.1.4 • 3.2.2

4.4 `DescribeProcess` Operation Test Module

4.4.1 HTTP GET `DescribeProcess` KVP Request

Test Purpose	Verify that a geoprocessing server accepts `DescribeProcess` requests transferred through HTTP GET as specified in Section 3.3.1.1.
Test Method	See A.4.3.1 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.3.1 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.3.1.1

4.4.2 HTTP POST `DescribeProcess` XML Request

Test Purpose	Verify that a geoprocessing server accepts `DescribeProcess` requests transferred through HTTP POST as specified in Section 3.3.1.1.
Test Method	See A.4.3.2 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.3.2 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.3.1.1

4.4.3 `DescribeProcess` Operation Response

Test Purpose	Verify that a geoprocessing server satisfies all requirements on the `DescribeProcess` operation response as specified in Section 3.3.2.
Test Method	See A.4.3.3 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.3.3 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.3.2

4.4.4 `DescribeProcess` Supported Identifier

Test Purpose	Verify that a process description is returned for a process identifier listed in the `Capabilities` document.
Test Method	Execute the `GetCapabilities` test as specified in Test Case 4.3.1 or 4.3.2. Select an identifier value from `wps:ProcessOfferings/wps:Process/ows:Identifier` in the `Capabilities` response document and use this identifier in a `DescribeProcess` request. Verify that the server returns a valid `ProcessDescription` document and that the value of the element `ows:ProcessDescription/ows:Identifier` is the same as passed in the `DescribeProcess` request.
Reference	• 3.1.4 • 3.3.1

4.4.5 `DescribeProcess` Unsupported Identifier

Test Purpose	Verify that an exception response is returned when using the `DescribeProcess` operation for a process identifier that is not listed in the `Capabilities` document.
Test Method	Execute the `GetCapabilities` test as specified in Test Case 4.3.1 or 4.3.2. Select an identifier that is **not** listed in the `Capabilities` document in the elements `wps:ProcessOfferings/wps:Process//ows:Identifier`. Verify that the server returns an exception report message with an exception code "InvalidParameterValue" in line with the requirements of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• 3.1.4 • 3.3.1, 3.3.2

4.5 `Execute` Operation Test Module

4.5.1 HTTP GET `Execute` KVP Request

Test Purpose	Verify that a geoprocessing server accepts `Execute` requests transferred through HTTP GET as specified in Section 3.4.1.1.
Test Method	See A.4.4.1 of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.4.1 of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.4.1.1

4.5.2 HTTP POST `Execute` XML Request

Test Purpose	Verify that a geoprocessing server accepts `Execute` requests transferred through HTTP POST as specified in Section 3.4.1.1.
Test Method	See A.4.4.2 [4] of the WPS Standard [OGC WPS 1.0.0, 2007].
Reference	• A.4.4.2 [4] of the WPS Standard [OGC WPS 1.0.0, 2007] • 3.4.1.1

4.5.3 Correct Handling of Process Identifiers

Test Purpose	Verify that the `Execute` operation succeeds for a process identifier that is listed in the `Capabilities` document.
Test Method	Execute firstly a `GetCapabilities` test as specified in Test Case 4.3.1 or 4.3.2, and secondly a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create a valid `Execute` request containing the mandatory inputs that are specified in the `ProcessDescription` in the elements `wps:ProcessDescription/wps:DataInputs/wps:Input` and at least one output specified in the elements `wps:ProcessDescription/wps:ProcessOutput/wps:Output`. Modulate the `identifier` parameter in the `Execute` request as follows: • Select an identifier value from `wps:ProcessOfferings/wps:Process/ows:Identifier` in the `Capabilities` response document and use this identifier in an `Execute` request. Verify that the request succeeds. • Select an identifier value that is not listed in the `Capabilities` document. Verify that the request fails and that the exception code returned is "InvalidParameterValue". Overall test passes if all individual tests pass.
Reference	• 3.1.4 • 3.4

4.5.4 Invalid Input Parameter

Test Purpose	Verify that the `Execute` operation fails when an input parameter is passed that is not listed in the `ProcessDescription` for the process that is invoked.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create a valid `Execute` request containing the mandatory inputs that are specified in the `ProcessDescription` in the elements `wps:ProcessDescription/wps:DataInputs/wps:Input` and at least

[4] Note that the numbering of this section (incorrectly) shows as "A.4.3.2" in Annex A of the WPS Standard [OGC WPS 1.0.0, 2007]. The correct numbering should be A.4.4.2.

	one output specified in the elements `wps:ProcessDescription/wps:ProcessOutput/wps:Output`. Add an additional input parameter that is not defined in the `ProcessDescription` element. Verify that the request fails.
Reference	• 3.4

4.5.5 **Invalid Output Parameter**

Test Purpose	Verify that the `Execute` operation fails when an output parameter is passed that is not listed in the `ProcessDescription` for the process that is invoked.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create a valid `Execute` request containing the mandatory inputs that are specified in the `ProcessDescription` in the elements `wps:ProcessDescription/wps:DataInputs/wps:Input` and at least one output specified in the elements `wps:ProcessDescription/wps:ProcessOutput/wps:Output`. Add an additional output parameter that is not defined in the ProcessDescription. Verify that the request fails.
Reference	• 3.4

4.5.6 **Missing Mandatory Input Parameters**

Test Purpose	Verify that the `Execute` operation fails when a mandatory input parameter, listed in the `ProcessDescription` element for the process that is invoked, is missing.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create an `Execute` **missing** request at least one mandatory input parameter specified in the `ProcessDescription` in the elements `wps:ProcessDescription/wps:DataInputs/wps:Input` and containing at least one output specified in the elements `wps:ProcessDescription/wps:ProcessOutput/wps:Output`. Verify that the request fails.
Reference	• 3.4

4.5.7 **Synchronous Execution - Raw Data Output**

Test Purpose	Verify that the `Execute` operation returns raw data, if an output is requested as raw data.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create an `Execute` request containing the mandatory input parameter specified in the `ProcessDescription` in the elements `wps:ProcessDescription/wps:DataInputs/wps:Input`. Request an output specified in the elements `wps:ProcessDescription/wps:ProcessOutput/wps:Output` as raw data by specifying the output identifier in the element

	wps:Execute/wps:ResponseForm/wps:RawDataOutput/ows:Identifier. Verify that the output is returned as raw data.
Reference	• 3.4

4.5.8 Synchronous Execution – Output as Value

Test Purpose	Verify that the Execute operation returns output as value, if an output is requested as value in an Execute response document.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create an Execute request containing the mandatory input parameters that are specified in the ProcessDescription in the elements wps:ProcessDescription/wps:DataInputs/wps:Input. Request an output specified in the elements wps:ProcessDescription/wps:ProcessOutput//wps:Output as value by specifying the output identifier in the element wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output/ows:Identifier and setting the value of the attribute asReference of the element wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output to “false”. Verify that the requested output is provided as value in the element wps:ExecuteResponse/wps:ProcessOutputs/wps:Output, i.e. that it contains a sub-element wps:Data containing the actual output data.
Reference	• 3.4

4.5.9 Synchronous Execution - Output as Reference

Test Purpose	Verify that the Execute operation returns output as reference, if an output is requested as reference in an Execute response document.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Create an Execute request containing the mandatory input parameters that are specified in the ProcessDescription in the elements wps:ProcessDescription/wps:DataInputs/wps:Input. Define an output specified in the elements wps:ProcessDescription/wps:ProcessOutput/wps:Output as reference by specifying the output identifier in the element wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output/ows:Identifier and setting the value of the attribute asReference of the element wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output to “true”. Verify that the requested output is provided as reference in the element wps:ExecuteResponse/wps:ProcessOutputs/wps:Output, i.e. it contains a sub-element wps:Reference containing the URL to the output data.
Reference	• 3.4

4.5.10 **Asynchronous Execution – Stored Response Document**

Test Purpose	Verify that the `Execute` operation returns a reference to an `ExecuteResponse` document, if the Execute operation is invoked in asynchronous mode.
Test Method	Execute a DescribeProcess test as specified in 4.4.1 or 4.4.2. Make sure that the values of the attributes `storeSupported` and `statusSupported` in the root element of the `ProcessDescription` are set to “true”. Create an `Execute` request containing the mandatory input parameters that are specified in the `ProcessDescription` in the elements `wps:ProcessDescription/wps:DataInputs//wps:Input`. Define an output in the `Execute` request as specified in the elements `wps:ProcessDescription/wps:ProcessOutput/wps:Output` of the `ProcessDescription`. To indicate that the operation should be executed in asynchronous mode, set the values of the attributes `storeExecuteResponse` and `status` of the element `wps:Execute/wps:ResponseForm/wps:ResponseDocument` to “true”. Verify that an `ExecuteResponse` document is returned immediately, specifying the URL to the stored `ExecuteResponse` in the `executeResponseLocation` attribute.
Reference	• 3.4

4.5.11 **Asynchronous Execution – Update of Stored Response Document**

Test Purpose	Verify that the `ExecuteResponse` document is updated, if the `Execute` operation is invoked in asynchronous mode.
Test Method	Follow the test method as specified in the previous Test Case 4.5.10. If possible, include at least one input with a large dataset in order to ensure a runtime large enough to verify a status update. After the last step, use the URL given in the `executeResponseLocation` attribute to verify that the value of the element `wps:ExecuteResponse/wps:Process/wps:Status` is updated.
Reference	• 3.4

4.5.12 **Asynchronous Execution – Value Output**

Test Purpose	Verify that an output is returned as value in an ExecuteResponse document, if the Execute operation is invoked in asynchronous mode and the output is requested as value.
Test Method	Follow the test method as specified in the previous Test Case 4.5.10, but when preparing the `Execute` request, make sure that an output is requested as value by specifying the output identifier in the element `wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output/ows:Identifier` and setting the value of the attribute `asReference` of the element `wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Out`

	put to “false”. Wait until the wps:Status element of the ExecuteResponse contains the element wps:ProcessSucceeded. Verify that the requested output is provided as value in the element wps:ExecuteResponse/wps:ProcessOutputs/wps:Output, i.e. it contains a sub-element wps:Data containing the actual output data.
Reference	• 3.4

4.5.13 **Asynchronous Execution – Output as Reference**

Test Purpose	Verify that an output is returned as value in an ExecuteResponse document, if the Execute operation is invoked in asynchronous mode and the output is requested as value.
Test Method	Follow the test method as specified in the previous Test Case 4.5.10, but when preparing the Execute request, make sure that an output is requested as value by specifying the output identifier in the element wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output/ows:Identifier and setting the value of the attribute asReference of the element wps:Execute/wps:ResponseForm/wps:ResponseDocument/wps:Output to “true”. Wait until the wps:Status element of the ExecuteResponse contains the element wps:ProcessSucceeded. Verify that the requested output is provided as reference in the element wps:ExecuteResponse/wps:ProcessOutputs/wps:Output, i.e. it contains a sub-element wps:Reference containing the URL to the output data.
Reference	• 3.4

5 REFERENCES

[OGC WPS 1.0.0, 2007]

Open Geospatial Consortium Inc., *OpenGIS® Web Processing Service*, Version 1.0.0, OpenGIS® Standard, OGC 05-007r7, 2007-06-08, online at http://portal.opengeospatial.org/files/?artifact_id=24151

[OGC WPS 1.0.0 C, 2009]

Open Geospatial Consortium Inc., *Corrigendum for OpenGIS Implementation Standard Web Processing Services (WPS) 1.0.0*, OpenGIS® IS Corrigendum 08-091r6, Version 0.0.8, 2009-09-16, online at http://portal.opengeospatial.org/files/?artifact_id=32766

[OGC WPS 2.0, 2015]

Open Geospatial Consortium, *OGC® WPS 2.0 Interface Standard Corrigendum 2*, Version 2.0.2, OGC ID 14-065r2, Publication date 2015-03-05, Revised date 2018-02-16, online at http://docs.opengeospatial.org/is/14-065/14-065.html

[OGC OWS-4, 2009]

Open Geospatial Consortium Inc., *OWS-4 WPS IPR Discussions, findings, and use of WPS in OWS-4*, OGC® Discussion Paper, OGC 06-182r1, Version 0.9.1, 2007-05-10

[NCIA AI_06.02.06, 2015]

NATO Communications and Information Agency, *Service Interface Profile for Messaging*, NCIARECCEN-4-30649, NCI Agency Technical Instruction, 2015

[OGC WSCom 1.1, 2007]

Open Geospatial Consortium Inc., *OGC Web Services Common Specification*, Version 1.1.0 with Corrigendum 1, OGC Implementation Specification, OGC 06-121r3, 2007-02-09, online at http://portal.opengeospatial.org/files/?artifact_id=20040

[OGC WSCom 1.1 C2, 2011]

Open Geospatial Consortium Inc., *Corrigendum 2 for OGC Web Services Common*

Specification v 1.1.0 – Exception, Implementation Standard Corrigendum, OGC 11-158, 2011-10-18, online at https://portal.opengeospatial.org/files/?artifact_id=46436

[OGC WPS BPDP, 2009]

Open Geospatial Consortium, *Web Processing Services Best Practices Discussion Paper*, Discussion Paper, OGC 12-029, 2012-04-04, External identifier http://www.opengis.net/doc/wps1.0-best-practice-dp, online at https://portal.opengeospatial.org/files/?artifact_id=47860

[IETF RFC 2119, 1997]

Internet Engineering Task Force, Request for Comments 2119, "*Key words for use in RFCs to Indicate Requirement Levels*", S. Bradner, March 1997, online at http://tools.ietf.org/html/rfc2119.

[IETF RFC 2046, 1996]

Network Working Group, Request for Comments 2046, "*Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*", N. Freed and N. Borenstein, November 1996, online at https://tools.ietf.org/html/rfc2046.

[IETF RFC 4646, 2006]

Internet Engineering Task Force, Request for Comments 4646, "*Tags for Identifying Languages*", A. Phillips, M. Davis, September 2006, online at https://www.ietf.org/rfc/rfc4646.txt.

[IETF RFC 4648, 2006]

Internet Engineering Task Force, Request for Comments 4648, "*The Base16, Base32, and Base64 Data Encodings*", S. Josefsson, October 2006, online at https://tools.ietf.org/html/rfc4648.

[IETF RFC 7231, 2014]

Internet Engineering Task Force, Request for Comments 7231, "*Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*", R. Fielding, J. Reschke, June 2014, online at http://tools.ietf.org/html/rfc7231.

[STANAG1059ed08, 2007]

NATO Standardization Agency, STANAG 1059 INT (Edition 8), "*Letter Codes for Geographical Entities*", Promulgated 19-February-2004

[C3B AC/322-D, 2015]

NATO Consultation, Command and Control Board (C3B), Document AC/322-D(2015)0014-REV3, "*The NATO Enterprise Approach for the Delivery of C3 Capabilities and the Provision of ICT Services*", 14-Dec-2015

[NATO AGeoP-8, 2016]

North Atlantic Treaty Organization, Allied Geospatial Publication AGeoP-8, Edition A Version 2, "*NATO Geospatial Metadata Profile (NGMP)*", February-2016.

[NATO AGeoP-8EdB, 2019]

North Atlantic Treaty Organization, Allied Geospatial Publication AGeoP-8, Edition B Version 1, "*NATO Geospatial Metadata Profile*", February-2019.

[NATO AGeoP-26 EdB, 202x]

NATO Standardization Office, Allied Geographic Publication AGeoP-26, "*Defence Geospatial Web Services*", Edition B Version 1, expected for publication in 2023.

6 ABBREVIATIONS

Acronym	Definition
AIS	Automated Information System
BBOX	Bounding Box
BPEL	Business Process Execution Language
BPMN	Business Process Model and Notation
C4ISR	Command & Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance
CES	Core Enterprise Services
CRS	Coordinate Reference System
CSV	Comma-separated values
CWL	Common Workflow Language
EPSG	European Petroleum Survey Group
FAS	Functional Area System
FS	Functional Service
FMN	Federated Mission Network
GIF	Graphics Interchange Format
GIS	Geographic Information System
GML	Geography Markup Language
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICT	Information and Communications Technology
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
ITM	IT Modernization
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
KVP	Key-value pair
MIME	Multipurpose Internet Mail Extensions
NATO	North Atlantic Treaty Organization
NCIA	NATO Communications and Information Agency
NGMP	NATO Geospatial Metadata Profile
OGC	Open Geospatial Consortium
OWS	OGC Web Service, or Open Web Service
PBN	Public Business Network
PDF	Portable Document Format
PNG	Portable Network Graphics
REST	Representational State Transfer
RFC	Request for Comments

SIP	Service Interface Profile
SLD	Styled Layer Descriptor
SOAP	Simple object access protocol
STANAG	Standardization Agreement
SVG	Scalable Vector Graphics
URI	Unified Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
WGS	World Geodetic System
WMS	Web Map Service
WMTS	Web Map Tile Service
WPS	Web Processing Service
WSDL	Web Services Description Language
XML	Extensible Mark-up Language
XSD	XML schema definition