Antony K Cooper, Harold Moellering, Jan Hjelmager, Petr Rapant, Tatiana Delgado, Dominique Laurent, David M Danko, Paloma Abad, Ulrich Düren, Serena Coetzee, Adam Iwaniak, Abbas Rajabifard, Michel Huet, and Jean Brodeur (2012). A spatial data infrastructure model from the computational viewpoint, International Journal of Geographical Information Science, ISSN: 1362-3087, DOI: 10.1080/13658816.2012.741239.

A spatial data infrastructure model from the computational viewpoint

Antony K Cooper[a*], Harold Moellering[b], Jan Hjelmager[c], Petr Rapant[d], Tatiana Delgado[e], Dominique Laurent[f], Serena Coetzee[g], David M Danko[h], Ulrich Düren[i], Adam Iwaniak[j], Jean Brodeur[k], Paloma Abad[l], Michel Huet[m] and Abbas Rajabifard[n]

[a]*Built Environment, CSIR, PO Box 395, Pretoria, 0001, South Africa*

[b]*Department of Geography, Ohio State University, Columbus, OH, 43210, USA*

[c]*Kort & Matrikelstyrelsen, Rentemestervej 8, Copenhagen, DK-2400 NV, Denmark*

[d]*IT4Innovations, VSB-Technical University of Ostrava, 17. listopadu 15, Ostrava-Poruba, Czech Republic, and Institute of Geoinfomatics, Faculty of Mining and Geology, VSB-Technical University of Ostrava, 17. Listopadu 15, Ostrava-Poruba, Czech Republic*

[e]*Department of Information Systems, Industrial Engineering Faculty, ISPJAE University, 119#11901 Marianao. La Habana, Cuba*

[f]*Institut Géographique National, 4 Rue Pasteur, 94165 Saint Mandé, France*

[g]*Centre for Geoinformation Science, University of Pretoria, Pretoria, 0002, South Africa*

[h]*Environmental Systems Research Institute, Inc, 8620 Westwood Center Drive, Vienna, VA 22182-2214, USA*

[i]*Landesvermessungsamt Nordrhein-Westfalen, Muffendorfer Strasse 19-21, 53177 Bonn, Germany*

[j]*Institute of Geodesy and Geoinformatics, Wrocław University of Environmental and Life Sciences, ul. Grunwaldzka 53, 50-357 Wrocław Poland*

* Corresponding author. Email: acooper@csir.co.za

[k]*Natural Resources Canada, Centre for Topographic Information, 2144-010 King West Street, Sherbrooke, Quebec, Canada J1J 2E8*

[l]*Instituto Geográfico Nacional, General Ibañez de Ibero 3, 28003 Madrid, Spain*

[m]*International Hydrographic Bureau, 4 quai Antoine 1er, Monaco*

[n]*Department of Geomatics, University of Melbourne, Melbourne, Victoria 3010, Australia*

The Commission on Geoinformation Infrastructures and Standards of the International Cartographic Association (ICA) is working on defining models of spatial data infrastructures (SDI). SDI models from the enterprise and information viewpoints of the Reference Model for Open Distributed Processing (RM-ODP) have already been presented. Our model from the computational viewpoint identifies the main computational objects of an SDI and their interfaces, which are modelled using Unified Modelling Language (UML) component diagrams. Presented here is the first comprehensive SDI model from the computational viewpoint, which enhances the understanding of the computational objects and their interactions in an SDI. This viewpoint complements the previous two and together, the three viewpoints contribute towards a more holistic interpretation of an SDI, which is independent of specific SDI legislation, technology and implementations. For the computational viewpoint, we identified six computational objects, *SDI Registry*, *SDI Data*, *SDI Processing*, *SDI Application*, *SDI Portrayal* and *SDI Management,* and their provided and required interfaces. We describe the interactions of the computational objects in stakeholder activities and the roles they play in the different processes of SDI development and use, which we identified as *Initiation*, *Creation*, *Management*, *Manipulation*, *Access*, *Processing*, *Evaluation* and *Liaison*. Two tables summarise the SDI services that are provided by computational objects for stakeholder activities and SDI processes.

Keywords: spatial data infrastructure; SDI; analytical cartography; reference model; computational viewpoint; unified modelling language; UML

1. Introduction

Over many decades, spatial data scientists in many parts of the world have worked to

develop mechanisms to share various kinds of scientific spatial data. More recently, efforts have been organized to design and build spatial data infrastructures (SDI) at the global, regional, national and local levels. There are two major facets to this effort: organizational and technical. The organizational side is concerned with various groups developing agreements, protocols and policy strategies for the SDI, see: Groot & McLaughlin (2000), Masser (2005), Rajabifard *et al* (2006), and Delgado (2005). Another aspect of the organizational side is the building and maintenance of the organizational structures to maintain the elements of the SDI, as well as running the daily SDI business.

The second facet of the SDI effort is of a technical nature: designing and implementing the mechanisms and networks that will bring the SDI to reality. This article focuses on the technical facet. More specifically, it discusses the contribution of the Commission on Geoinformation Infrastructure and Standards of the International Cartographic Association (ICA) (hereafter called *the Commission*) by modelling the SDI through the prism of the various model views of the Reference Model for Open Distributed Processing (RM-ODP) (ISO/IEC 10746-1:1998), using the Unified Modelling Language (UML) (ISO/IEC 19501:2005, Object Management Group 2011). The work discussed here operates in a broader scientific milieu of geographical information science and analytical cartography, e.g. see Moellering (2000).

Previous work by the Commission described SDI models from the enterprise and information viewpoints (Hjelmager *et al* 2005, Hjelmager *et al* 2008). In this article, the Commission's SDI model from the computational viewpoint is presented. The computational viewpoint is a functional decomposition of a system into a set of objects that interact at interfaces – enabling system distribution (ISO/IEC 10746-1:1998). This article consolidates and refines earlier work. The initial SDI model from a computational viewpoint identified computational objects and briefly described them (Cooper *et al* 2007). Earlier work identified the processes of SDI development and use (Cooper *et al* 2009). In this article the descriptions are refined and examples are added to improve the understanding. We also consolidate earlier work by describing how stakeholders (identified in the enterprise viewpoint) interact through interfaces (identified in the computational viewpoint) during the activities associated with the information classes (identified in the information viewpoint).

The remainder of the article is structured as follows: section 2 provides a short overview of previous work by the Commission leading up to this article, as well as other work that is related to ours. Section 3 describes an SDI from the computational viewpoint. Section 4 links the enterprise, information and computational viewpoints by showing how stakeholders identified in the enterprise viewpoint interact through the interfaces identified in the computational viewpoint during the activities identified in the information viewpoint. Section 5 identifies the different processes of SDI development and use, and then shows how SDI services are provided by computational objects for these SDI processes. Examples from the European SDI, INSPIRE (INfrastructure for SPatial InfoRmation in Europe), are included to illustrate the service needs for processes. Two tables summarize the specific SDI services for the stakeholders provided by each computational object for each activity (Table 1 in section 4) and process (Table 2 in section 5) respectively. Section 7 evaluates the SDI model from a computational viewpoint and provides conclusions on this work.

2. Previous and related work

2.1 SDI modelling framework

Business and enterprise systems are becoming more and more flexible and increasingly dependent on local and global communication networks. As a result, the software systems have to become more modular and distributed throughout these networks. The design and installation of such distributed systems is a complex task, which requires comprehensive conceptual work before beginning the implementation. The International Standard ISO/IEC 10746-1:1998, *Information technology – Open Distributed Processing – Reference model: Overview*, aids in this challenge by providing a framework for the design and description of distributed software and information systems.

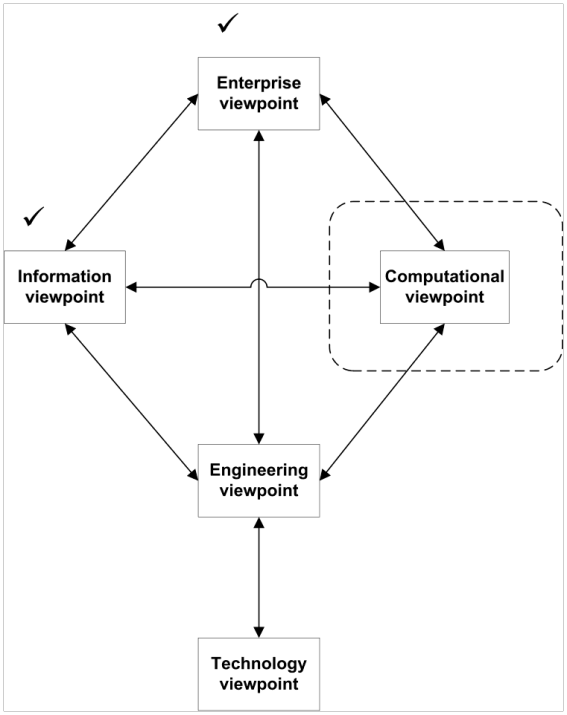

Figure 1: The five RM-ODP Viewpoints and their dependencies (from Hjelmager *et al* 2008)

RM-ODP defines five *viewpoints* for the design and description of distributed software and information systems: the *enterprise viewpoint* (purpose, scope and policies for the system); the *information viewpoint* (semantics of information and information processing incorporated into the system); the *computational viewpoint* (functional decomposition of the system into a set of objects that interact at interfaces); the *engineering viewpoint* (mechanisms and functions required to support distributed interaction between objects within the system); and the *technology viewpoint* (the specific technologies chosen for the implementation). Figure 1 shows the five RM ODP viewpoints and their dependencies, with the focus in this article being on the computational viewpoint.

Earlier research work by the Commission began with a general overview of the SDI (Aalders & Moellering 2001), and then proceeded to examine SDIs using UML (Cooper *et al* 2003). Subsequently, the Commission developed formal models of SDIs from the enterprise and information viewpoints of RM-ODP (Hjelmager *et al* 2005; Hjelmager *et al* 2008). The investigation of the enterprise viewpoint of an SDI identified the stakeholders of an SDI and their activities, i.e. the actors and the use cases, in UML; and identified the core components of an SDI and their relations.

While the enterprise viewpoint deals mainly with the administrative aspects of an SDI, the information viewpoint has its focus on the products (data and services), their specification, their description via metadata, and product registries (catalogues). For the information viewpoint, we identified *information classes* that are required for SDIs to deliver data and services and showed how stakeholders are linked to the information classes through *stakeholder activities* in an SDI.

2.2 Other attempts at modelling SDIs

There are a number of documents that publish guidelines or specifications for the software architectures of specific SDI implementations. For example, the INSPIRE Network Services Architecture (INSPIRE Network Services Drafting Team 2008) describes the INSPIRE service types that are mandated by the INSPIRE Directive (2007) and shows how these services connect to portals and applications through the INSPIRE service bus. Another example is the Canadian Geospatial Data Infrastructure (CGDI) (Geoconnections 2005), which provides a high-level overview of the CGDI architecture and describes the range of services that comprise the CGDI. The CGDI architecture endorses the RM-ODP, and describes elements that can be used to support the information, engineering and computational RM-ODP viewpoints, while the implementation details of the RM-ODP technology and engineering viewpoints are left as the responsibility of agencies collaborating in the CGDI. Both the INSPIRE Network Service Architecture and the CGDI architecture are technology independent, similar to the Commission's scientific model, but both of them are bound by the policies and legislation of the respective organizations. The Commission's model recognizes that policies and legislation play a role in an SDI but the model is independent of specific policies and legislation, an approach also taken for the SDI Cookbook (Nebert 2004).

Béjar *et al* (2008) analyze published architectural SDI models and based on this analysis, propose an architectural style for SDI software architectures. This style provides a tool and a shared vocabulary to help system architects to design SDIs, and facilitates the exchange of knowledge about them. The style is defined under the component-and-connector architectural viewtype, extending the client-server and shared-data styles. In contrast, the functional decomposition into a set of objects provided in the computational viewpoint of the Commission's SDI model provides the basis for decisions on how to distribute objects. The Commission's model is therefore not restricted to client-server architectures, but can also be implemented in emerging architectures, such as data grids and cloud environments. In addition, the Commission's work provides a holistic view of an SDI by linking components of the different viewpoints, for example, by listing SDI services that are performed by objects

(computational viewpoint) during stakeholder (enterprise viewpoint) activities associated with information classes (information viewpoint).

Coetzee (2009) presents Compartimos, a reference model for sharing address data on a data grid in an SDI environment. The computational viewpoint for Compartimos describes the essential components, and their relationships and interactions, that are required for sharing address data on a data grid. In Coetzee and Bishop (2009), commonly available technology choices for Compartimos are explored. Compartimos is, similar to the extended client-server and shared-data architectural styles of Béjar *et al* (2008) because it is specific in terms of an architectural style, namely the data grid. The Commission's model from a computational viewpoint, on the other hand, does not specify the physical distribution of the computational objects and is therefore not bound to a specific architectural style. The actual distribution of computational objects could be specified in future in one or more engineering viewpoints. In addition, while Compartimos is specific to address data, the Commission's model applies to any kind of spatial information.

While there are those who have nominally applied systems theory to SDIs (e.g. Mavima *et al* (2001)), they have not actually done so (ie: considering SDIs as self-regulating systems), but have rather applied various theories of systems to SDIs (soft systems theory, in the case of Mavima *et al* (2001)). Grus *et al* (2006) considered SDIs as complex adaptive systems, which are systems that changed from stable and predictable to unstable and unpredictable, and then exhibited more complex patterns of behaviour. Mansourian & Abdolmajidi (2011) applied system dynamics to SDIs, that is, where positive and negative feedback loops are used to model systems with multiple components that mutually interact within a common goal and where the system adjusts internally in response to external disturbances.

3. Description of an SDI from a computational viewpoint

The *computational viewpoint* is a functional decomposition of the modelled system into a *set of objects* that interact at *interfaces*, such as to provide services – enabling system distribution (ISO/IEC 10746-1:1998). In the context of an SDI, the computational viewpoint captures the details of the computational objects and their interfaces definitions without regard to the physical distribution of the objects (i.e. where they actually are located in the world). The latter is covered by the engineering viewpoint. The computational viewpoint sets the scene for distribution by decomposing the system and by specifying a binding model describing how interactions between given computational interfaces are carried out.

3.1 SDI computational objects modelled as UML components

We use a UML component diagram (Object Management Group 2010) to model the SDI computational objects. The component diagram allows one to structure the *components* of a system (autonomous and encapsulated units within the system providing one or more interfaces) and their interrelations. Briefly, the basic elements of a component diagram are (as shown in Figure 2):

- *Component* (rectangle with small symbol in upper right corner): A *component* represents a modular part of a system that encapsulates its contents and whose manifestation is replaceable within its environment. A component defines its behavior in terms of provided and required interfaces.
- *Provided Interface* (connector with circlet): An interface is a named set of operations that characterize the behaviour of a component. A *provided interface* is one that is implemented directly by the component.
- *Required Interface* (connector with arc): A *required interface* specifies services that a component needs in order to perform its function and fulfill its own obligations to its clients.
- *Dependence* (dashed arrow): A *dependency* is a relationship that signifies that a component requires services from an other component through its *provided interface* (Object Management Group 2010).

The RM-ODP defines a computational interface as being characterized by a signature, a behaviour and an environmental contract. However, since we are modelling the SDI computational objects as UML components, we specify the interfaces of the provided and required interfaces of the SDI computational objects, as defined in the list above. For simplicity reasons, we did not model the details of the signature and contract of individual interfaces. These could be provided in a further refinement of the computational viewpoint.

Figure 2 shows the six SDI computational objects in a UML component diagram: *SDI Registry*, *SDI Data*, *SDI Processing*, *SDI Application*, *SDI Portrayal* and *SDI Management*. Each SDI computational object *offers* a number of functionalities, modelled by the *provided interfaces*, as well as *uses* functionality offered by other SDI computational objects, modelled by the *required interfaces*.

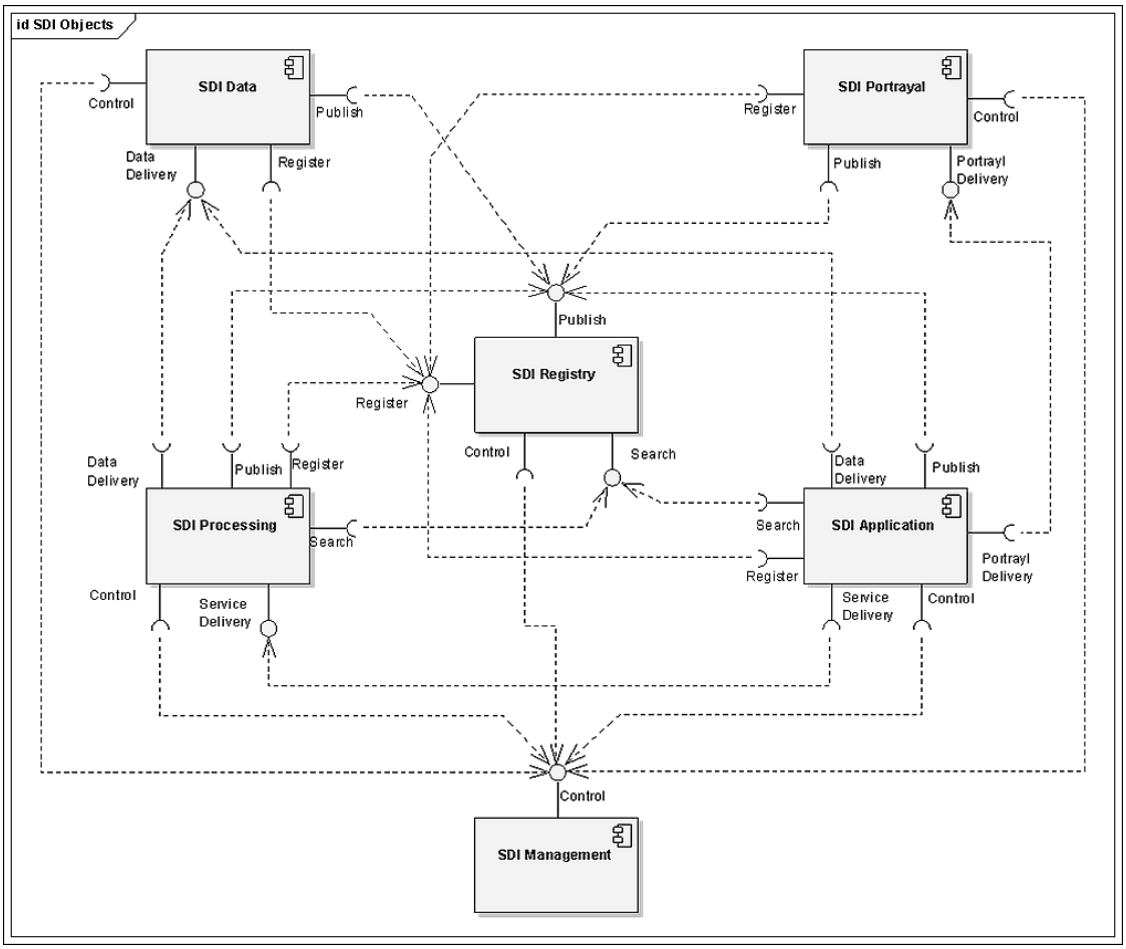

Figure 2. The SDI computational objects (adapted from Cooper *et al* 2007)

3.2 Purpose and interfaces of the SDI computational objects

3.2.1 SDI Registry

The main purpose of the *SDI Registry* is to *register* data, services (i.e. products) and other items in the catalogue, to *publish* them, and afterwards let users *search* through them. This functionality is provided by the three interfaces of *SDI Registry* described below. The *SDIManagement::Control* is the only required interface. The double colon notation '::' means Interface.

The *SDIRegistry::Register* interface provides the necessary operations to *register* information about resources available on a network, such as, Register Product Specification, Register Product, Register Metadata, Register Catalogue, Register Policy, Register Business Plan and Update Register

The *SDIRegistry::Search* interface facilitates searching for the required items in the relevant catalogue and includes operations, such as, Search Register, Search Product Specification, Search Product, Search Metadata, Search Catalogue and Search through Catalogue.

The *SDIRegistry::Publish* interface provides functionality for publishing the output information from Registers through the Internet or other media. Operations, such as, Publish Product Specification, Publish Product, Publish Metadata and Publish Catalogue are typically included.

3.2.2 SDI Data

SDI Data deals with data sets shared and registered on the Internet. For example, *SDI Data* provides access to collections of data in repositories and databases. The only provided interface is *SDIData::DataDelivery*, which is designed in such a way that it provides data to users via the *SDI Processing* or *SDI Application*. No other SDI computational object deals with data sets directly. Required interfaces are *SDIRegistry::Register*, *SDIRegistry::Publish* and *SDIManagement::Control*.

3.2.3 SDI Processing

SDI Processing provides the interfaces for data processing, such as, coordinate computation and projection system transformation. It provides only one interface, *SDIProcessing::ServiceDelivery*. However, a number of interfaces are required, including *SDIRegistry::Register*, *SDIRegistry::Publish*, *SDIRegistry::Search*, *SDIData::DataDelivery* and *SDIManagement::Control*.

3.2.4 SDI Application

SDI Application is a key part of the SDI computational architecture. It does not provide any interface, but requires a large number of interfaces in order to meet the needs of users. These include SDIRegistry::Register, *SDIRegistry::Publish*, *SDIRegistry::Search*, *SDIData::DataDelivery*, *SDIProcessing::ServiceDelivery*, *SDIPortrayal:: PortrayalDelivery* and *SDIManagement::Control*.

3.2.5 SDI Portrayal

SDI Portrayal deals with displaying the results of application services. It provides one interface for these purposes, *SDIPortrayal::PortrayalDelivery*, which facilitates output, such as, designing layouts, editing functions, specifying delivery options and formats. *SDI Portrayal* has the following required interfaces: *SDIRegistry::Register*, *SDIRegistry::Publish* and *SDIManagement::Control*.

3.2.6 SDI Management

SDI Management monitors the overall functionality of the SDI. For this purpose it has one interface, *SDIManagement::Control,* which, for example, controls the interoperability amongst services or rights of access.

3.3 Binding model

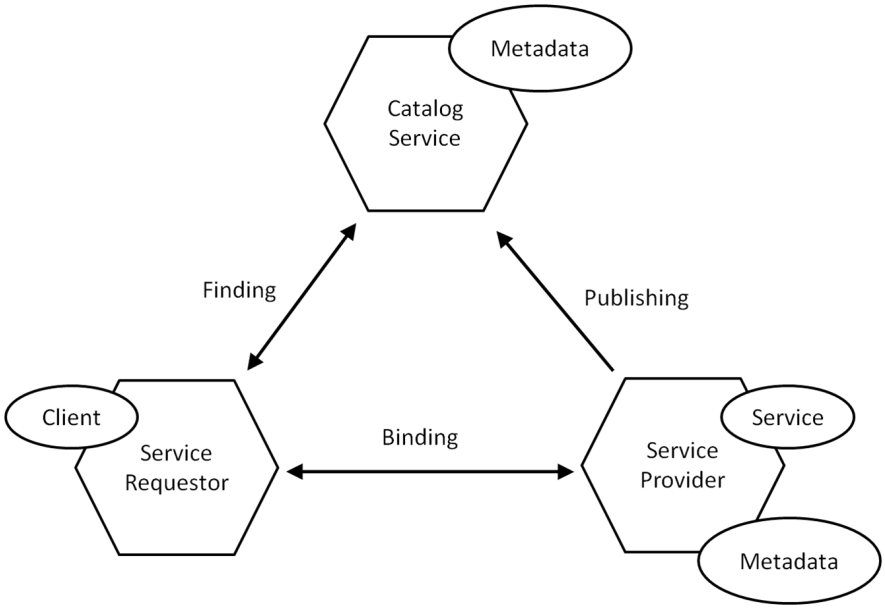

Figure 3. A service oriented architecture (adapted from W3C 2004)

According to the RM-ODP (ISO/IEC 10746-1:1995), binding should be modelled in a computational viewpoint. The purpose of the binding model is to specify how liaisons are to be created between objects in order for interactions to occur.

The ICA computational model of an SDI is a service-oriented architecture (SOA), as shown in Figure 3 (W3C 2011). An SOA is based on the concept that a service provider publishes its services at a service registry. A service requester finds details about a specific service at the registry, proceeds to bind to the service at the service provider and starts interacting with the service at the service provider. Since our model is technology independent, the binding model is inherited from whichever technology is used in the implementation (e.g. SOAP, REST, DCOM or CORBA).

4 SDI services provided by SDI computational objects for stakeholder activities

In section 3, we identified the *provided* and *required* interfaces of SDI computational objects. In this section we show how stakeholders (identified in the enterprise viewpoint) interact through these interfaces during the activities associated with the information classes (identified in the information viewpoint).

Initially, the Commission defined the concept "service" for its own purpose, starting with the definitions from *ISO 19119:2005, Geographic information – Service*. As such, an *SDI service* was defined as *the distinct part of the SDI functionality that is provided by an SDI computational object through interfaces, either manually or automatically.*

Additionally, an *interface* is defined as *a named set of operations that characterize the behaviour of an object.*

Moreover, an *operation* is defined as *something that an object may be called on to do, may do without request, or may do for internal reasons*.

By the integration of these definitions, an *SDI service* is defined as *the distinct part of the SDI functionality that is provided by an SDI computational object through named sets of operations that characterize the behaviour of the SDI computational object, where each operation may be called on to do, may do without request, or may do for internal reasons, either manually or automatically*.

Thus, in this model, an *SDI service* is the distinct part of the SDI functionality that is provided by a collection of operations from (one or more) interfaces of (one or more) SDI computational objects. An operation is part of a single interface and is performed by a single SDI computational object, but a service is more general, collectively referring to a number of operations from one or more computational object's interfaces.

Based on this definition of an SDI service, we identified more than 180 services within an SDI and classified them in a matrix showing which computational object provides them and where they are required for stakeholder activities associated with the five information classes (i.e. Policies, Product specification, Product, Metadata, and Catalogue) from the information viewpoint. Table 1 provides a detailed list of the SDI services. Column 1 contains the information classes defined in the information viewpoint (from Table 1 in Hjelmager *et al* 2008). Column 2 contains the stakeholder activities associated with these classes, also defined in the information viewpoint (from Table 1 in Hjelmager *et al* 2008). Row 1 contains the SDI computational objects. The rest of the cells in the table show the SDI services that fulfil the functionality required by the stakeholder activities. Note that there does not have to be a specific SDI service linking every activity to every object in the table.

Consider, for example, the *Publishing* service, which makes all kinds of information publicly available. The *SDI Registry* computational object provides the required interfaces. The *Publishing* service is required in all the information classes for various stakeholder activities, such as, 'Make policy' and 'Make business plan' of the Policies information class and 'Stipulate requirements', 'Translate into product specifications' and 'Obtain and implement product specifications' of the Product specifications information class.

Another example is the *Delivery* service, which delivers data, metadata and catalogues to SDI stakeholders. *SDI Data* and *SDI Processing* provide the *SDIData::Delivery* and *SDIProcessing::ServiceDelivery* interfaces for the 'Provide product', 'Use product' and 'Maintain product' stakeholder activities of the Product information class; the 'Provide metadata' of the Metadata information class and the 'Provide catalogue' of the Catalogue information class. *SDI Application* does not provide interfaces but acts as intermediary to deliver data, metadata and catalogues through the above mentioned *SDI Data* and *SDI Processing* interfaces to the SDI stakeholders in the 'Provide product' and 'Provide catalogue' of the Product and Catalogue information classes respectively.