An architectural style for spatial data infrastructures

R. Béjar \(^a\), M. Á. Latre \(^a\), J. Nogueras-Iso \(^a\), P. R. Muro-Medrano \(^a\)
& F. J. Zarazaga-Soria \(^a\)

\(^a\) Computer Science and Systems Engineering Department,
University of Zaragoza, c/María de Luna 1, 50018 Zaragoza, Spain


To link to this article: http://dx.doi.org/10.1080/13658810801905282
This work proposes an architectural style, a pattern, for spatial data infrastructures (SDIs). This style provides a tool and a shared vocabulary to help system architects to design these infrastructures, and facilitates the exchange of knowledge about them. This style is defined under the component-and-connector architectural viewtype, extending the client–server and shared-data styles. The style has been created after analyzing six of the most relevant SDIs and geo-service architectural proposals. Several architectural elements that these proposals have not properly addressed are considered. Three real projects, with published architectural views or models, have been examined to verify the applicability of the style. The proposed style offers a systematization and refinement of knowledge about SDIs, grounded in well-known concepts in software architecture.

**Keywords**: Spatial data infrastructure; Software architecture; Architectural style; Pattern

1. **Introduction**

The definitions of Spatial Data Infrastructure (SDIs) have included, directly or indirectly, the necessity to provide search, visualization and data download services (GSDI Technical Working Group and contributors 2004). Besides these services, and as anticipated by the USA Federal Geographic Data Committee (FGDC) Geospatial Applications and Interoperability Working Group (Evans 2003), SDIs have grown in complexity by including other types of services (Bernard and Craglia 2005). As SDIs become more complex, their software architecture becomes more relevant in order to facilitate their design and understanding. This paper is focused on the software architectural design of SDIs that are developed by single organizations.

There are several previous works which propose architectural models for SDIs. Most of these proposals have their technical roots in a work of the FGDC Geospatial Applications and Interoperability Working Group (Evans 2003), which describes a model of geospatial processing based on interoperable geospatial services. Among the most relevant SDI architectural models are the proposal by GeoConnections (2005) about the Canadian infrastructure and the initial proposals for the European SDI established by the AST Working Group (2002), though this initial European proposal is currently being superseded after the approval of the

*Corresponding author. Email: rbejar@unizar.es*
INSPIRE directive: the first draft of the INSPIRE technical architecture has already been published (INSPIRE Drafting Teams ‘Data Specifications’, ‘Network Services’, ‘Metadata’ 2007). These architectural models provide high-level views of SDIs and, consequently, they lack some elements that would have been defined for a detailed software architecture: they are focused on the allowed components for the architecture but they barely mention, when they are mentioned, other elements of an architecture, as the types of relationships among the components, their visible properties or necessary constraints. There are also non-obvious overlaps among the different architectural models, e.g. components with different names but similar roles are common. Finally, they are not completely grounded in well-known architectural models: a Service Oriented Architecture (SOA) (Erl 2004) is commonly mentioned as the basis, but this architectural model fails to capture many components included in these models like applications or data repositories; no other references to software architectural models are provided.

Because of the lack of more detailed architectural models for SDIs, the high-level proposals cited in the previous paragraph have been used to support the development of many of such infrastructures. As these proposals do not follow any kind of common structure or pattern, comparing them, or verifying that a given information system actually follows one of the reference models they present, is difficult and ad hoc.

This paper provides an architectural style that unifies some of the most important SDI technical architectural proposals. Besides this, the style is a template that presents the concepts and rules needed to document the architecture of an SDI, either to design one, or to communicate this architecture to others.

The rest of this paper is structured as follows. First, some definitions are given about software architecture and the concepts of view, viewtype and style. Then an architectural style for SDIs is defined after analyzing the architectures proposed by six SDIs and geographic service reference models. In section 4, three real SDI projects with published architectural descriptions are studied to determine their compatibility with the proposed style. The next section offers an architectural description of one of these projects, following the proposed style, to highlight the benefits of its use. Section 6 discusses the methodology used in this paper. In the final section, some important issues are highlighted and conclusions are drawn.

2. Software architecture

Although there is not a single accepted definition, a software architecture can be defined as the structure of a system, and is formed by elements, sometimes called components, their properties and the relationships among them and possibly with their environment. This definition roughly summarizes the main points of those proposed by Kruchten et al. (2006), Clements et al. (2003) and the IEEE Architecture Working Group (2000).

The software architecture of a system is an inherent property of this system, but documenting this architecture may be a very complex task. The approach that has been consolidated over the past few years is expressing the software architecture of a system as a set of views, each of them addressing different concerns for different users. Indeed, documenting an architecture without specifying the type of view that is being used tends to create too complex diagrams, with too much information and without a clear separation of concerns.
2.1 Architectural views

The ‘views and beyond’ proposal by Clements et al. (2003) or the IEEE Architecture Working Group (2000) recommended practice for architectural description of software-intensive systems, share a similar approach that allows us to describe the architecture of a system as a set of views which follow some defined viewtypes, viewpoints in the IEEE standard, and styles.

A view is a representation, typically a graphical representation of some of the elements and relationships that are present in a system. For instance, a view of a given Web information system could be a diagram showing its Web services as boxes and the relationships between these services, e.g. requests and responses, as arrows.

A viewtype is a definition of the allowed element types and relationship types that can be used to describe a system from a certain perspective. For instance, a viewtype for Web information systems could indicate that only Web services, as boxes, and service chaining, as arrows between these boxes, should be included (and not databases, applications and other components of the system). This viewtype could be useful to have a perspective of the high-level processes, i.e. service chaining, that occur in that system.

A style is a specialization of a viewtype; it can specify that only certain elements and relationships from the viewtype are allowed, and include also other constraints. A style for standards-based Web information systems could specify that only certain types of Web Services, e.g. those suggested by a certain standardization organization, can be included.

The architecture of a system will be documented by a set of views of this system. These views will address different concerns and will be created following the guidelines provided by the viewtypes and styles provided by the literature. The ‘views and beyond’ methodology describes a set of viewtypes and styles, but it does not propose a fixed set of views for a given system: the architects will decide the views they need to completely document their system.

In the rest of this paper, the ‘views and beyond’ proposal is followed, and therefore the style presented here is under the umbrella of one of the viewtypes defined by Clements et al. (2003): the Component-and-Connector (C&C) viewtype, which is described in the next subsection.

2.2 The C&C viewtype

Clements et al. (2003, p. 103) indicate that C&C views include elements with runtime presence, such as clients or servers, which are the components, and the pathways for their interactions, such as information flows, captured as connectors. A general C&C viewtype thus consists of allowed component and connector types, constraints for allowed relationships (i.e. which connectors are attached to which components), some properties of the components and the connectors (e.g. a name or a type) and maybe also some topological constraints (e.g. ‘a connector must not be attached to another connector’).

3. An SDI style for the C&C viewtype

This section defines an SDI architectural style for the C&C viewtype. The objective behind the definition of this style is to capture, unify and systematize the previous knowledge on SDI architectural models, and to explicitly take into account elements
that are not typically considered in these models (i.e. constraints), or considered only implicitly (i.e. data stores).

A hybrid style is defined in Clements et al. (2003, p. 201) as the combination of two or more existing styles. From the styles for the C&C viewtype in this book, those that have been considered more appropriate as a basis for this work are Shared-Data and Client-Server. The proposal in this paper is a specialization of a hybrid style which combines these two:

(i) Shared-Data: this style highlights interactions dominated by the exchange of persistent data (i.e. data that are stored in a non-volatile storage, such as a hard disk, so it remains available between executions of the processes that make use of it). It is important for SDIs because spatial datasets and metadata are persistent and relevant data, shared by different kinds of services. In this style, there are two types of components: shared-data repositories and data accessors. The possible connector types are data reading and data writing. Data accessors are attached to data repositories by means of these types of connectors.

(ii) Client-Server: this style shows asymmetric interactions among components, from clients to servers. It is important in SDIs because they follow an SOA: some of their services will act as servers, for other services or for applications, and others will act as clients for other services, and these interactions are the base of developing complex functionality. In this style, there are also two types of components: clients, which request services, and servers, which provide them. The connector type is thus request/reply. Clients are attached to servers.

The next sections describe the elements of a new style for SDIs. These elements extend those in the Shared-Data and Client-Server styles to tailor them to the necessities of a software architect designing an SDI. This style has been designed from the experience of the authors in several SDI projects (Béjar et al. 2003, 2004, Latre et al. 2005, Portolés-Rodríguez et al. 2005), taking into consideration several of the most relevant SDI and geoservice architectural descriptions in the bibliography. A discussion about the relationship between the elements proposed here and those in the bibliography is also presented.

3.1 Previous work on SDI architectural models

These are the main bibliographic references that have been taken into consideration, and the reasons to choose them:

- The International Organization for Standardization (ISO) Technical Committee 211 (TC 211) standard on geographic information services (ISO 19119, ISO/TC 211 (2003)): this is the most thorough taxonomy of geoservices available. From a technological point of view it is an abstract specification, but most, if not all, current SDI initiatives are using Web services and this technology fits very well with the ISO standard.
- The OGC Web service architectural description (Whiteside 2005): the geoservice architecture from the most active standardization organization, with ISO, in the geospatial field. It is quite similar to the ISO standard, but it is technologically specific (Web services, based on Web protocols or SOAP, and XML to transfer data).
- The FGDC Geospatial Interoperability Reference Model (GIRM) (Evans 2003): the concept of national SDI was developed in the USA, and the FGDC
set up this guide, one of the first and most relevant for these kinds of infrastructures. Besides this, this model was included in the first position paper on architecture for the Infrastructure for Spatial Information in Europe (INSPIRE) (AST Working Group 2002).

(iii) The Canadian Geospatial Data Infrastructure Architectural Description (GeoConnections 2005): the architecture of one of the leading projects in national SDIs in the world.

(iv) The final text of the European Union Directive for the establishment of a European SDI (INSPIRE, European Parliament and The European Council (2007)): relevant because it establishes the minimum requisites for all national SDIs of the EU member states to be part of a European SDI. Although it could be considered that it does not define any architecture, the truth is that although there are not any diagrams, it gives some detail on the components that national SDIs in the EU must have, in some cases more deeply than other architectural proposals.

(v) A proposal from the European Commission (EU) Joint Research Center (JRC), presenting the initial steps leading to the establishment of the European Geographic Information Portal (Bernard et al. 2005). The JRC is the institution in charge of providing scientific and technical support of the EU policies, among them INSPIRE. Although, of course, this list may never be complete, a reference to the Global SDI (GSDI) could be expected. But the GSDI cookbook (GSDI Technical Working Group and contributors 2004) does not suggest an SDI architectural reference model; it refers to other documents for this (especially ISO and OGC standards) which have been considered.

### 3.2 Component types

The component types in this section are specializations of those in the client-server and shared-data styles defined in Clements et al. (2003). Regarding to this, although the ISO 19119 standard is platform-neutral, most other bibliography on SDIs and geoservices assumes an SOA, deployed over Internet protocols with XML as the data exchange format, i.e. Web Services (Booth et al. 2004); this is also the case of this work. The component types have been chosen because they play relevant roles in SDIs, but not all of them need to be present in every SDI. Their names have been selected from the bibliography when there seemed to be a high degree of consensus. When this has not been possible, they have been chosen to highlight their main characteristic. The intention has been to capture the main structure of an SDI, so the component hierarchy is not very deep. The ISO and the OGC have done a good work specifying types of geoservices, so in this paper only the higher levels in the component hierarchy, which hold a higher level of information about the structure of an SDI, have been defined.

Figure 1 shows the hierarchical relationships among these component types, and among those in the client-server and shared-data styles. This is a Unified Modeling Language (UML) class diagram where classes represent component types. Table 1 holds a comparison of these component types with those that appear in the considered bibliography. The table shows which of the proposed component types appear in the different architectures studied. It also indicates when they appear with a different name, with a similar, but not equal, meaning, or when they do not appear but are related, even indirectly, to other explicit elements. The definitions of the proposed component types are given in the next list.
Figure 1. Hierarchical relationships among SDI style component types.
Table 1. SDI style component types compared with those in the other architectures studied.

<table>
<thead>
<tr>
<th>Component Type</th>
<th>OGC</th>
<th>FGDC</th>
<th>GIRM</th>
<th>ISO 19119</th>
<th>INSPIRE</th>
<th>EU Geoportal</th>
<th>Canadian GDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SDI Service</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓ (Geographic service)</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>(OGC Web Service)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Processing Service</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓ (Geoprocessing service)</td>
<td>✓ (Geographic processing service, Geographic communication service)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transformation Service</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Information Management Service</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Portrayal Service</td>
<td>✓ (Web map service)</td>
<td>✓ (Maps &amp; visualization service)</td>
<td>✓ (Map access service)</td>
<td>✓ (View service)</td>
<td>✓ (Rendered map service)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access Service</td>
<td>✓ (Web feature service, Web coverage service)</td>
<td>✓ (Web feature service, Web coverage service)</td>
<td>✓ (Feature access service, Coverage access service, Product access service)</td>
<td>✓ (Download service)</td>
<td>✓ (Feature service)</td>
<td>✓ (Geographic features service, geographic feature encoding service)</td>
<td></td>
</tr>
<tr>
<td>Catalog Service</td>
<td>✓ (Catalog service for Web)</td>
<td></td>
<td></td>
<td>✓ (Catalogue service) (1)</td>
<td>✓ (Discovery Service)</td>
<td>✓ (Registery)</td>
<td></td>
</tr>
<tr>
<td>Gazetteer Service</td>
<td>✓ (Web gazetteer service)</td>
<td></td>
<td></td>
<td>x (2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Knowledge Model Service</td>
<td>x (10)</td>
<td>x</td>
<td></td>
<td>x (3) (Registry service)</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Application Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDI Client</td>
<td></td>
<td>✓ (4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geoportal</td>
<td>x (5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

An architectural style for spatial data infrastructures.
Table 1. Continued.

<table>
<thead>
<tr>
<th>Component Type</th>
<th>OGC</th>
<th>FGDC GIRM</th>
<th>ISO 19119</th>
<th>INSPIRE</th>
<th>EU Geoportal</th>
<th>Canadian GDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>×</td>
<td></td>
<td></td>
<td>×</td>
<td>× (6)</td>
<td>√ (7)</td>
</tr>
<tr>
<td>Metadata Repository</td>
<td>×</td>
<td>(User application)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
</tr>
<tr>
<td>Dataset Repository</td>
<td>×</td>
<td>(Content Repository)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
</tr>
<tr>
<td>Spatial Dataset Repository</td>
<td>×</td>
<td>(Feature and Coverage Repositories)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
</tr>
<tr>
<td>Knowledge Model Repository</td>
<td>×</td>
<td>(7)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>× (7)</td>
<td>×</td>
</tr>
</tbody>
</table>

×: This architecture does not include explicitly this component type.
√: This architecture includes this component type with similar semantics, though maybe with a different name (it is given within parentheses).
1) There is also a Registry Service in the ISO 19119 standard, but with a different meaning.
2) Not mentioned explicitly in the INSPIRE directive, but an indirect reference seems implied when it indicates that the discovery services will support search by geographical locations.
3) The ISO 19119 registry service provides access to metadata about types, so it could be considered as a specialization of the knowledge model service.
4) The FGDC GIRM mentions clients in its interoperability stack, indicating that user applications will act as clients in the distributed system they propose.
5) OGC defines Web portal services, but they are application services as defined in this work.
6) The geoportal is defined in this work as an application that aggregates instances of GI services, but application is not defined and not mentioned anymore.
7) Not explicit components, but they are indirectly taken into consideration (i.e. considering [some of] the contents of the store type).
8) The architecture does not reference it, but a geospatial ontology server, a specialization of the knowledge model service, has been created for this SDI (M3GO, http://intelegeomatics.com/ogm3/).
9) A thesaurus service as defined for the EU Geoportal could be considered as a specialization of a knowledge model service.
10) The catalog in the OGC architecture can store metadata and datasets which can be schemas, models, semantic documents, etc., so it could play the role of the knowledge model service.
• **Web Service**: all kinds of Web services (Booth *et al.* 2004).

• **SDI Service**: all web services in an SDI will be a specialization of an SDI Service. The name has been chosen to reinforce the idea behind the architectural style while avoiding other names than could be understood as too restrictive (i.e. calling them geographic information services or geoservices seems to imply that they all access geographic data, and this will not be the case for some of them).

• **Processing Service**: these services are designed to make generic processing of data, typically spatial data. These data can be provided when calling their operations or the services can access to some data repositories.

• **Transformation Service**: services that allow spatial datasets to be transformed with a view to achieving interoperability.

• **Information Management Service**: they store and provide access to data and metadata.

• **Portrayal Service**: they support the visualization of spatial datasets.

• **Access Service**: these services allow downloading or accessing spatial datasets or parts of them.

• **Catalog Service**: these services make it possible to discover, explore and evaluate datasets, services, etc. by means of the metadata describing them.

• **Gazetteer Service**: they offer geocoding functionalities, which link toponyms and their spatial locations.

• **Knowledge Model Service**: they offer discovery and access to shared knowledge models in order to facilitate the semantic interoperability among different services, applications, etc.

• **Application Service**: those used to support client applications, especially thin, i.e. Web, clients.

• **SDI Client**: software that gives human users access to the services in an SDI.

• **Application**: a kind of computer software that allows users to perform a set of tasks, most of them using SDI Services.

• **Geoportal**: web sites mainly focused on geographic content, geographic services, and the tools to discover them. Although it would be possible to model a Geoportal as a type of application, the relevance of Geoportals for SDIs in the bibliography supports considering them on their own.

• **Metadata Repository**: a repository which holds identifiable metadata and structured data about other resources in the SDI (datasets, services, etc).

• **Knowledge Model Repository**: a repository which holds knowledge models, defining knowledge models as data models, schemas, ontologies, thesauri or any other explicit conceptualization of knowledge in a domain.

• **Dataset Repository**: a repository that holds datasets, defining datasets as identifiable collections of data.

• **Spatial Dataset Repository**: a repository that holds spatial datasets, which are defined as identifiable collections of spatial data (i.e. data with a direct or indirect reference to a specific location).

### 3.3 Connector types

In the bibliographic references on SDI and geoservice architecture listed before, there is little attention paid to connectors. At most there are some indications about what kinds of components can connect to others without further details. This could
be due to the fact that defining special connector types seems not necessary for SDIs; but since connectors in general are barely mentioned, it could also be possible that they have not been considered at all. After a more detailed study, neither new relevant types of connectors nor refinements of those provided by the Client-Server and Shared-Data styles have been found. Therefore, these are those included in the SDI style:

- From the Client-Server style:
  - **Request/Reply**: the invocation of a server by a client and its response go through this connector type. In the SDI style, SDI Clients (i.e. Applications) or SDI Services can make requests to SDI Services, and the latter can reply to the former.

- From the Shared-Data style:
  - **Data Reading**: data accessors read data from data repositories. In the SDI style, different types of SDI Services can read data (i.e. Information Management Services).
  - **Data Writing**: data accessors write data to data repositories. In the SDI style, different types of SDI Services can read data (i.e. Access Services).

### 3.4 Properties

As with the connector types (see Section 3.3), it has not been possible to find relevant properties for SDIs that were present in a significant number of architectural proposals, but missing in the generic C&C styles (Client-Server and Shared-Data). But among the several properties for these styles suggested by Clements et al. (2003), there are some that are used in some studied SDI architectures:

- **Name**: for components and connectors, suggesting their functionality or the nature of its interactions.
- **Type**: the type which components and connectors belong to.
- **Types of data stored**: for **Shared Data Repositories**.

This list of properties is not closed. System architects may consider it useful adding others when designing their SDIs following the proposed style. For example, properties indicating access permissions or performance indicators could be useful. They just seem a little too specific for the objectives of this work.

### 3.5 Constraints

As defined before, constraints in an architectural style are rules which specify how the elements defined for the style, especially components and connectors, can be used, and the valid interactions among them. This section defines some fundamental topological constraints, which are those that define how components relate to each other by means of connectors.

First of all, these are the allowed connector configurations (topological constraints) defined for the Client-Server and Shared-Data styles (they have already been mentioned when describing the Connector Types):

- From the Client-Server style:
  - Client **Requests** from Server.
  - Server **Replies** to Client.
From the Shared-Data style:
- Data Accessor **Reads Data** from Shared-Data Repository.
- Data Accessor **Writes Data** to Shared-Data Repository.

In the studied SDI and geoservice architectures and models, there are not many clear references to constraints. Nevertheless some can be found:

- **OGC**: this architecture describes some ideas which are constraints indeed. They would be clearer if they were separated and made explicit. The constraints designed for the style in this paper are compatible with these ideas.
- Services are organized into tiers but loosely, and it is not required to separate services that way. Services can use other services within the same tier or not.
- All kinds of services may access data, although most of data will be accessed by Information Management Services.
- **FGDC GIRM**: this model organizes its components in an ‘interoperability stack’. In this stack, user applications have access to services and to content repositories (direct data access), and services access other services and content repositories. In our proposal, **Clients** are not **Data Accessors**, so they are not allowed to read or write to **Shared Data Repositories**; this is more restrictive than the GIRM proposal, where applications can directly access content repositories. As most other SDI proposals separating clients from data by means of services, this constraint has been included in the style designed in this paper.
- **ISO 19119**: in this standard, the engineering viewpoint section establishes as a reference model of a four-tier logical architecture. This logical architecture is then mapped to different physical ones, establishing thus some constraints on the topology of interactions among services. The problem is that this architecture is designed for generic Information Technology (IT) services as well as for GIS-extended services, so it is a general proposal with a broad scope. If besides this we consider that this standard is not for SDIs but for geoservices in general, it results that the level of detail is not appropriate to extract conclusions useful for an SDI style as the one defined in this work.

When defining the SDI style, new component types have been pointed out. These component types extend those in the Client-Server and Shared-Data styles, so they inherit their constraints too. But not every component type extending Data Accessor should be allowed to read and write from/to any kind of Shared-Data Repository. New constraints are needed to explicitly capture these new rules. These constraints are given as forbidden topological connections among some component types:

- **Portrayal Service**:
  - **NOT Writes** to Shared Data Repository.
  - **NOT Reads from** Knowledge Model Repository, Metadata Repository.

- **Access Service**:
  - **NOT Reads from AND NOT Writes to** Knowledge Model Repository, Metadata Repository.

- **Catalog Service**:
  - **NOT Reads from AND NOT Writes to** Knowledge Model Repository, Dataset Repository.

- **Gazetteer Service**:
  - **NOT Reads from AND NOT Writes to** Knowledge Model Repository, Metadata Repository.
Knowledge Model Service:

• **NOT Reads from AND NOT Writes to** Metadata Repository, Dataset Repository.

These constraints intend to separate the roles of the different service types. For example, a Portrayal Service is specifically designed to portray existent spatial datasets, so, although it is basic to allow it to read Spatial Dataset Repositories, it is not allowed to modify them or to read from other types of repositories. If one of these services would be needed to read a knowledge model or some metadata, and this situation is perfectly possible, it should do it through a specialized service (a Knowledge Model Service or a Catalog). This is in order to follow good design principles, like a clean and strict separation of service roles. But in some situations, these constraints may be unnecessarily complex: for example, one could want to create a Catalog Service able to read just some data from a Spatial Dataset Repository, but without the burden of setting up an Access Service. This can be done by defining a new component type which extends Catalog Service and Access Service. This component would be thus allowed to read from a metadata repository and from a dataset repository. The idea behind constraining the data repositories which can be accessed from different components is to help to clarify their function; but a system architect may decide that for a specific SDI a catalog component which accesses metadata and datasets is the best solution. This style allows for that while making it explicit that this component is a Catalog Service and an Access Service. Making it explicit is useful because it gives roles and precise meanings to the elements in an SDI, and because it helps this system architect to document the design, relating this component to the component types defined for this style. This also makes the design easier to understand to other system architects who know the SDI style.

It is also important to notice that geoportals and Applications are not allowed to access Shared Data Repositories, because they are not Data Accessors. If this necessity arises in the process of designing an SDI, it is a clear indication that some Application Services are needed. This is one of the reasons why Application Services have been defined: to separate Applications from the Dataset Repositories, helping to enforce the usual rules of layered IT systems.

A final consideration is that Portrayal Services have been allowed to read from Dataset Repositories. It could be argued that this would be the role of an Access Service and that most Portrayal Services in SDIs would also have to be Access Services. This decision has been taken precisely because the main function of Portrayal Services is reading Spatial Datasets and portraying them. If their main function includes reading Datasets, it seems correct to allow them to read from Dataset Repositories.

As in the case of property types (Section 3.4), this list of constraints is not closed. They have been chosen because they capture the basic ideas which appear, normally in an implicit manner, in the SDIs studied, and have proven themselves to be useful in the experience of the authors with SDI projects.

4. **Analysis of real SDI architectures**

In this section, three different projects are analyzed in order to determine if real SDIs have architectures that fit the proposed style. These projects are from regions in three different European countries and have been developed by different people with different technologies, objectives and constraints. They have been chosen
because they give enough public architectural information, claim to be following SDI principles and have a view that is close to the C&C viewtype.

4.1 Architecture of the Galicia CMA SDI

Galicia is a Spanish Region, NUTS-2 in the EU terminology, located at the northwest corner of the Iberian Peninsula. The climate is warm and wet so its land is covered with many forests (69% of its surface). This fact makes forests the main concern of their environmental department (Consellería de Medio Ambiente, CMA) which is responsible for water use, disposal of waste and protected natural environments as well. This department had found the same kinds of problems with geographic information that SDIs address: incompatible data formats and information systems, difficulties in disseminating data among their users (it is a very decentralized department), difficulties in finding relevant information, etc. The solution adopted in 2001 to overcome these problems was to develop a geographic information system for this department, but following INSPIRE principles and recommendations in architecture and standards, thus effectively building an SDI. This project had to be developed using the available commercial software licenses in the CMA in a COTS (commercial off-the-shelf) approach. This infrastructure was designed to become the core of a future Galician SDI, and it is described in some detail in Béjar et al. (2003). In this paper, there is an architectural view of this SDI, referred to as a ‘Service Oriented Architecture’, which is shown in figure 2.

This architecture is depicted in a layered way, focusing on its components and some of their properties. It must be noted that there are neither explicit connectors nor constraints in this diagram, but some of them are detailed in the text of the paper. Regarding the components, and following their function as explained in the paper, they all can be matched to some of the component types proposed in the SDI style:

- In the layer ‘Data and Metadata Sources’:
  - Vector Data and Raster Data are Spatial Dataset Repositories.
  - SDI Documentation is a Shared Data Repository.
  - Metadata is a Metadata Repository.

- In the layer ‘Chainable Services’:
  - WMS-Core, WMS-Raster Core and WMS-Environmental are Portrayal Services.
  - WFS-Core, WFS-Environmental and WCS-Raster are Access Services.
  - OGC Metadata Catalog and OGC Services Catalog are Catalog Services.

- In the layer ‘Integration Services’:
  - Access Control is a specialized Application Service.

- In the layer ‘User Applications’:
  - All components in this layer are Applications.

Some connectors and constraints can be extracted. At least there is one that is quite clear and that also matches the proposed style:

- ‘User applications are built on top of distributed services’ (p. 94, emphasis added). This implies a connector between user applications and distributed services and follows two of the defined constraints:
  - Clients Request from Servers. Applications in the SDI style are Clients and the SDI Services are Servers, so Applications Request from the SDI Services, as it happens in the CMA SDI architecture.
Data Accessor **Reads and Writes Data** from Shared-Data Repository. These constraints imply that any component type that is not a Data Accessor cannot read or write data from Shared-Data Repositories. Neither Applications in the SDI style nor ‘User Applications’ in the CMA SDI architecture are Data Accessors. The reason is that they access ‘Services’ instead of ‘Data Sources’.

Finally, it is worth noting that in this architecture every component has the properties suggested for the SDI style (**name**, **type** and **types of data stored** for repositories), but the connectors do not have any.

### 4.2 Architecture of the Piedmont local SDI

The SITAD is the name of a project which points towards the creation of a local SDI in the Piedmont region, Italy. Designed according to INSPIRE principles, it
aims at facilitating the coordination of public sector organizations to collect, manage, distribute and reuse spatial data (Cipriano and Garretti 2004). This paper describes the components in the SITAD and provides the architectural diagram shown in figure 3. Although it is not indicated whether this diagram follows any existing architectural view type, it is stated that it ‘represents the presentation logic, the business logic and the data logic of the infrastructure’ (p. 4). According to the architectural principles in Clements et al. (2003), all that information should probably have been distributed among several views (i.e. in the same diagram are shown elements quite different like Web servers (software components) and metadata records (datasets)). Anyway, the information in this diagram and the text of the paper enable the evaluation of some elements in the SITAD architecture. These are the components described in the paper mapped, as far as it has been possible, to their equivalent types proposed in the SDI style:

- Application to compile metadata is an **Application**.
- Metadata catalogue (MTD in the figure) is a **Catalog Service**.
- Unique catalogue gateway is a **Geoportal**.
- Web map services are **Portrayal Services**.
- Download services are **Access Services** or **Information Management Services** if they hold non-spatial data.

![Architecture of SITAD infrastructure](image)

Figure 3. Architecture of SITAD infrastructure (taken from Cipriano and Garretti (2004, p. 5)).
Visualization services are **Information Management Services** if they show non-spatial data.

- Multi-map service viewer is an **Application**.
- User interfaces (i1 and i2 in figure 3) are **Applications**.
- DBs (from the figure) are **Dataset Repositories**. When they have the ‘Spatial Box’ over them, they are **Spatial Dataset Repositories**.

With regard to connectors or constraints, there is little information that can be extracted from the paper. The text mentions that data are accessed via on-line services and served to clients, which points out that there must be connectors between data and services (at least **Data Reading**) and between services and clients (**Request/Reply**). Probably this also implies several of the constraints defined for the SDI style, though trying to specify this would be pure speculation. There are also some connectors portrayed in figure 3, which seems to confirm this interpretation of the text.

The only property that is shown for some components is their **type**. There are not any properties for the connectors.

### 4.3 Architecture of the North Rhine-Westphalia GDI

As described by Brox *et al.* (2002), the Geospatial Data Infrastructure North Rhine-Westphalia (GDI-NRW) is an initiative of the Land North Rhine-Westphalia, in Germany. It started in January 2000, with the objective to develop a market for geographic information in that Land by connecting users, service providers and enablers, integrators, data producers and infrastructure providers. In addition to a general description of the objectives of this SDI, this paper includes an architectural model with a taxonomy of services and technical components (pp. 31–33). The component diagram is shown in figure 4. Although this diagram presents an architectural model without any architectural view, it does not make it a less valid or

---

**Figure 4.** GDI North Rhine-Westphalia component diagram (taken from Brox *et al.* (2002, p. 32).
relevant reference for the purpose of verifying the applicability of the SDI style in real projects.

First of all, instead of defining a taxonomy of services, the GDI-NRW service taxonomy adheres to the one described by ISO/TC 211 on geographic information services (ISO/TC 211 2003). Then, focusing on the technical components, they define a model based on Web services, with a number of components which support them. These geospatial services are classified into three categories:

- GDI-NRW Search and Discovery Services: organization, discovery and access of geospatial information.
- GDI-NRW Access and Retrieval Services: access to geospatial information outside the scope of the catalog services.
- GDI-NRW Web Mapping Services: distributed Web mapping.

The components described in figure 4, following OGC specifications, fall into these categories:

- Catalog Server: search and discovery of geospatial data and services through its metadata.
- Web Map Server: services for distributed Web mapping.
- Web Coverage Server: services for access to coverage data.
- Web Feature server: service for access to feature data.

The other component types in the figure are the clients, which access any kind of data distributed in the GDI-NRW through the services, and the metadata and geospatial data storages, which are not defined though some comments are given regarding their contents.

The paper ends giving some future steps to the architectural model, which include (with little detail) services for portrayal and presentation, ordering and payment, security, authentication, gazetteers and an e-commerce framework.

Regarding the component types in figure 4, and understanding their function as explained in the paper, they all can match the component types of the SDI style:

- GDI Client is an SDI Client.
- Catalog server is a Catalog Service.
- Web Feature Server and Web Coverage Server are Access Services.
- Web Map Server is a Portrayal Service.
- Metadata Storage is a Metadata Repository.
- Geospatial Data Storage is a Spatial Dataset Repository.

The proposed services for the evolution of the architectural model can match those in the SDI style, though there are some aspects that need to be clarified:

- Portrayal and Presentation Services are Portrayal Services.
- Gazetteer is a Gazetteer Service.
- Ordering and Payment Services are SDI Services. These kinds of services are quite specific and important and it could be argued that they should have been included in the style. The problem is that we are far from a consensus on the e-commerce technical aspects of an SDI. Although this issue is important, and addressed in some high level SDI specifications and regulations (i.e. in the INSPIRE directive text), the idea behind the proposed style is to capture, refine and systematize the existing knowledge about SDI architecture. The e-commerce issue has not been defined or implemented to an extent that makes
this viable. On the other hand, the SDI style does not prevent an SDI architecture from having e-commerce services, which would extend the SDI Service type, and maybe others (i.e. Access Services).

- Security and Authentication Services are SDI Services. With these kinds of services arises a problem that is very similar to the one discussed in the previous point.

From the text of the paper and figure 4, some connector types and constraints can be extracted for the GDI-NRW. There are five kinds of relationships in the figure:

(i) GDI client uses Catalog Server, Web Feature Server, Web Coverage Server and Web Map Server: this one is called a Request/Reply connector in the SDI style.

(ii) Web Feature Server and Web Coverage Server get data from Geospatial Data Storage: this one would be equivalent to the Reads Data connector.

(iii) Web Map Server displays data from Geospatial Data Storage: this is also equivalent to Reads Data. The paper does not give any indication of the difference between this connector and the get data from discussed before.

(iv) Catalog Server discovers Metadata Storage: there are no explanations about this connector, but most probably it does not mean that the Catalog Server needs to discover the location of the metadata it serves! Indeed, it seems that this connector is similar, if not identical, to the gets data from in the diagram, so equivalent to Reads Data.

There is another connector in the figure, which helps to illustrate the problems of creating an architectural diagram without defining its view type: the Metadata Storage describes Geospatial Data Storage connector. If the diagram is a style of a viewtype similar to the C&C, which seems implied in the paper, then the connectors should be among components, not among other elements. Although the depicted type of connector is undoubtedly present (i.e. some metadata in the Metadata Storage will surely describe some data in the Geospatial Data Storage), it is clearly different from the others, because it does not show a connection between components: probably it would be better placed on another diagram with a different view type.

With regard to constraints, the diagram shows a layered architecture with connectors that seem to enforce some of the constraints defined for the SDI style: clients in the GDI-NRW component diagram only use servers (Client Requests from Server), and only the servers are allowed to get data or metadata from the storages. Therefore, it could be assumed that because servers in this diagram are all Data Accessors and storages are all Shared Data Repositories, the constraint Data Accessor Reads Data from Shared-Data Repository is implicit.

The only property that is shown both for the components and the connectors is their type. Since this proposal is an architectural model instead of an actual architecture, and it includes component types instead of components, this is the only property that makes clear sense.

5. Application of the SDI style

This work would not be complete without an example of application of the proposed style to document a view of an SDI architecture. Thoroughly documenting the views of a software architecture is a complex task (see Clements et al. 2003, pp. 317–322 for some guidelines) far from the intention of this paper; this section is focused on the primary presentation, as defined in that book. There are many
different options to document views, from formal architectural description languages (ADLs) to various graphical notations. UML has been chosen because it is widely extended in the information systems community in general, and in the geospatial and SDI community in particular. As UML can be used in different ways to document an architectural view, some clarification is needed: objects will represent the different components in the view and associations among them will represent the connectors; different shapes have been used for the different types of objects (UML graphical stereotypes). Topological constraints are implied in the diagram (i.e. component types that must not be connected will not be connected).

The Galicia CMA SDI has been chosen as the example to avoid defining a new project environment. Since this architecture has been found to extend some of the component types in the SDI style, figure 5 has been included to facilitate the understanding of the architectural view that comes next. In that figure, classes represent component types, and those on top are the component types defined for the SDI style. In the rest of this section, this question will thus be answered: if the SDI style had been followed, how would a view of the Galicia CMA SDI architecture have been documented?

5.1 Galicia CMA architectural view following the SDI style

Figure 6 shows an architectural view of the Galicia CMA SDI, following the guidelines given by the SDI style. Several components projected, but not implemented, have not been included in order to have a diagram easier to understand. All the elements shown have their type: components have names and belong to a type defined in the SDI style, or belong to a type which extends one in the SDI style. Repositories include the data types they hold (one of their properties). Connectors are explicit: they have a name and their shapes indicate their type. The question is: what are the differences between this diagram and the one shown in figure 2?

![Figure 5. Galicia CMA SDI component types which extend those in the SDI style.](image-url)
The first difference is that once the SDI style is known, the meaning of this diagram is better defined. Most component and connector types, except for those defined specifically for the project, have a defined meaning. Even from those that are not defined in the SDI style, i.e. the Web Map Service, things can be immediately deduced: for example, as the Web Map Service extends the Portrayal Service, everything that is true for the Portrayal Service (definition, constraints, etc.) must also be true for the Web Map Service.

This diagram is also more complete: connectors and also the types of the components are explicit. For example, it is now clear that the services do not write to the repositories, but only read from them.

Finally, as constraints are explicit for the SDI style, it can be sure that they are fulfilled: for example, it is clear that the applications in the Galicia CMA SDI do not read data from the data repositories (this was not so clear before).

6. Methodology discussion

In this section, the methodology that led to the results presented in this paper and the rationale behind the decisions taken are explained. The objective was to facilitate the architectural description of the distributed information systems that support SDIs, i.e. their technological backbone. This was divided into two tasks: analyzing the current technological proposals for SDIs, and presenting a unified model over solid software architectural concepts.

The bibliography was analyzed in order to find the most relevant sources of information about SDI technical architectures and their components. After selecting...
some of the fundamental documents, they were studied and crossed to find the common elements and the main differences. Table 1 summarizes most of the information that was gathered.

After the analysis of the bibliography, the knowledge collected and unified had to be presented in a way that made it useful to describe software architectures. The ‘views and beyond’ methodology was chosen to provide the architectural support of this work. Architectural styles are the mechanism provided by this methodology to describe the architecture of a particular class of systems, so the creation of a new style for SDIs was decided.

Clements et al. (2003, chapter 6.5) propose several options to create new styles. The option chosen was to combine two existing styles and to specialize from this combination. Combining two existing styles, the client-server and the shared-data styles, gave us roots in fundamental software architectural concepts. The specialization made was domain-specific. This allowed us to present the information about SDI technology collected and unified in the previous phase under the terms of a solid software architectural documentation methodology. The resulting SDI style is presented in Section 3.

Finally, the bibliography was analyzed again to find examples of systems that claimed to be SDIs in order to verify if their architectures could have been described with the SDI style. This was made in order to test the applicability of the style to real systems.

The choice of the ‘views and beyond’ methodology is justified because of its flexibility: the set of viewtypes and styles it provides is extensible, and the methodology itself gives guidelines to extend it. This has allowed us to present a result that fits a tried and proven software architectural methodology, while being able to incorporate domain-specific concepts, i.e. spatial concepts.

The ISO and International Telecommunication Union Telecommunication Standardization Sector (ITU-T) Reference Model for Open Distributed Processing (RM-ODP), for example (Putman 2000), is an architectural methodology cited in some OGC documents, e.g. Whiteside (2005), so it could seem suitable to support our work. There were two reasons not to choose this option: firstly, this methodology prescribes a fixed set of views and it does not provide an extension mechanism similar to the one used in our work; and secondly, the RM-ODP is designed to specify architectures, not to describe them. Specifying is a task that goes beyond description and was out of the objectives of our work (there are discussions about the differences between description and specification in Putman (2000, p. 32) and Clements et al. (2003, p. 8)).

7. Conclusions

This work proposes a pattern to design and document distributed geographic information systems following SDI design principles. The pattern has been presented as an architectural style, defined under the component-and-connector view type, extending two well-known styles in distributed information systems: the client-server and shared-data styles. The style has been created analyzing several important SDI architecture proposals, finding their common elements, and giving them a unique name and a definition. Several elements that a software architecture should consider, which had not been addressed in these proposals yet, have also been discussed and included in the style (especially connectors and constraints). Three real SDI projects, with published architectural views or models, have been
examined to verify whether the style would have been applicable to them. For one of these projects, the style has been effectively applied to show how this could have been done.

The proposed style offers a systematization, refinement and extension of knowledge about SDI architectures. The style intends to facilitate the software creation and integration activities in SDI development, so its usefulness is independent of the SDI scope (local, regional, national, etc.). Anyone designing or communicating the architecture of an SDI may benefit from the guidelines provided by the style. The papers cited in Sections 4.1, 4.2 and 4.3 are good examples of the current necessity to communicate information about SDI architectures. The style may facilitate this communication, because it names, defines and arranges the basic common elements in SDI architectures, providing thus a common ground for understanding.

The style has been defined with its extension in mind: it is a minimum core of elements that are common to most SDI proposals, either explicitly or implicitly, but a system architect may extend it to address specific necessities of an SDI. Indeed, there are several aspects of SDIs that the style does not address: e-commerce, security, etc. These are issues which are currently under discussion, so it was considered that it was too early to include them.

There is a refinement of the SDI style that could have been considered: the use of OGC and ISO specifications for the components in the style when possible (i.e. instead of a Portrayal Service, a Web Map Service could have been included). Although this was seriously considered, a more abstract approach was decided. This decision was adopted to promote concepts before technologies and because the value of a style is larger when it can be applied to more architectures; specifying too much detail reduces its applicability. The result is a style that can be easily refined to allow only for OGC components but does not force them. In addition to this, neither ISO nor OGC has defined every component type in the SDI style, and almost none of the other elements (connectors, properties and constraints), so the style could not have been completely defined in OGC or ISO terms.

To finish these conclusions, it is important to remark two issues about the scope of this work. First of all, the style proposed refines the component-and-connector view type, but there are other view types and styles for software systems that would be interesting for SDIs. The second issue is that as well as being distributed geographic information systems, SDIs are also Information Infrastructures, composed by different independent systems working together. From this point of view, their architecture could not be designed, but ‘cultivated’ (Georgiadou 2006). This will require further advances that allow us to analyze their architectural properties not only in terms of their components, but also in terms of the systems that compose them.

Acknowledgements
This work has been partially supported by GeoSpatiumLab, the National Geographic Institute (IGN) and the Spanish Ministry of Education and Science through the project TIN 2006-00779 from the National Plan for Scientific Research, Development and Technology Innovation.

References


GSDI TECHNICAL WORKING GROUP AND CONTRIBUTORS, 2004, Developing Spatial Data Infrastructures: The SDI Cookbook v.2.0 (Global Spatial Data Infrastructure (http://www.gsdio.org)).


WHITESIDE, A., Eds, 2005, OpenGIS web services architecture description, no. OGC 05-042r1 In: OpenGIS Discussion Paper (Open Geospatial Consortium Inc.).