InterDataNet: an Infrastructural Approach to Data Interoperability to Enable Computer Supported Collaborative Applications

Maria Chiara Pettenati  
Electronics and Telecommunications  
Department, University of Florence (IT)  
mariachiara.pettenati@unifi.it

Davide Chini  
Electronics and Telecommunications  
Department, University of Florence (IT)  
davide.chini@unifi.it

David Parlanti  
Electronics and Telecommunications  
Department, University of Florence (IT)  
david.parlanti@gmail.com

Franco Pirri  
Electronics and Telecommunications  
Department, University of Florence (IT)  
franco.pirri@unifi.it

Abstract

Distributed information interoperability is crucial for the development of computer supported collaborative applications. We recognize two main factors inhibiting extensive data interoperability: one is the lack of “Web of Data” solutions allowing greater granular addressability and management of elementary data; another is the lack of suitable “Web of Data” infrastructural solutions providing the basis for the development of specific applications. We present InterDataNet (IDN) layered middleware architecture. IDN is based on the adoption of a shared Information Model (IDN-IM) and the management of such model operated by a Service Architecture (IDN-SA) providing collaboration-support functions. The InterDataNet name aims to put to evidence the goal of achieving interoperability among networks of data (interdataworking). IDN proposes itself to act as an interoperability application-independent middleware, supporting distributed information management to enhance interoperating and collaboration. Basic concepts and specifications which are grounding development of an IDN demonstrator are reported and discussed.

1. Introduction

At present distributed information interoperability is often addressed trying to share a common information model which, eventually, allows communication by pairs [1]. Sometimes interoperability is achieved through an information conversion operated at the application level [2, 3, 4] with a great amount of inefficiency in the information management. In other cases [5] interoperability is based on communication end-nodes sharing the same information model [6, 7]. From a scalability point of view, the last approach is the most efficient as a basis for building infrastructures to provide support for interoperable applications. Such approach has also been recognized as being extremely valuable in the definition of JCR API [8] - a Java API for content management. However, often at the state of the art, even when this solution is adopted, it is normally tied to the specific applications such as in [9].

Addressing data interoperability issues also requires the fundamental need of a greater granularity in addressability and management of data units. This is also referred to as the “Web of Data” approach [10, 11]. Until now we experienced the Web of Document intended primarily as a repository of interlinked documents where addressability is restrained to the document level. To shift from the current Web of Document to the Web of Data view, we need to break down complex resources into simpler information pieces which can be handled (addressed) with sufficient granularity so as to allow their effective processing, integration and, ultimately, interoperability.

Even if a shared information model was defined and manageable at an arbitrarily granular level, a concrete system capable of managing it would be nonetheless needed. As it happened in the domain of communications for internetworking (TCP/IP), in the realm of data interoperability an infrastructural level solution to interoperability is needed. The aim of this work is to address the issue of data interoperability through the definition of an interdataworking infrastructural solution which should allow connecting,
distributing and integrating data in an open environment.

Such system should provide adequate functionality to the collaborative applications for the management of generic information compliant with the information model. To this end, in this work we present IDN (InterDataNet) basic concepts and specifications to sustain the achievement of a NETwork of INTERoperable DATA.

IDN specifies the model, the languages and the architectural solution to:
1. provide a reference Information Model to enable Data interoperability;
2. provide global granular data addressability;
3. provide a set of basic collaboration-oriented services for data management through a layered, service-oriented middleware.

It is expected that CSC (Computer Supported Collaborative) applications [12, 13] can be built on top of IDN middleware thus exploiting above basic functions. InterDataNet hence can become a key enabler for applications such as distributed content production, multimedia document production and management, workflow and versioning management, groupware systems and collaborative content management, just to mention some.

The paper is organized as follows: section 2 deals with the general IDN Model Framework. Section 3 presents the IDN-IM (Information Model), while section 4 focuses on IDN approach to data global addressability. Section 5 illustrates the IDN-SA (Service Architecture) providing a macro-level view of the distributed middleware functions. Section 6 highlights how IDN middleware can satisfy Computer Supported Collaborative (CSC) applications requirements.

2. IDN Model Framework

IDN is grounded on the following conceptual and technological paradigms:

IDN is a layered middleware following SOA (Service Oriented Architecture) approach [14, 15]; this will allow the development of loosely coupled and interoperable services which can be combined into more complex systems. Layering approach allows the breaking down of the system complexity exploiting “information hiding” and “separation of concern” principles [16, 17].

IDN adopts of the REST (Representational State Transfer) approach [18, 19]; to use its uniform interface approach to resources (URIs) [20] for performance, scalability and resource abstraction in distributed systems.

IDN framework is described through the ensemble of concepts and models pertaining to the following entities:
1. IDN-IM (InterDataNet Information Model). It is the information model representing a generic document model which is independent from specific contexts and technologies. It defines the requirements, desirable properties, principles and structure of the document to be managed by IDN. In the IDN-IM, information is highly structured and endowed with specific metadata to serve collaboration purposes;
2. IDN-SA (InterDataNet Service Architecture). It is the architectural layered model to provide the services for handling IDN documents. The IDN-SA defines the reference functionalities, subsystems, protocols and interfaces for IDN document collaborative management.

3. IDN reference Information Model to enable data interoperability

An Information Model can be defined as “an abstraction and universal representation of the entities in a managed environment - their properties, operations and relationships. It is independent of any specific repository, application, protocol or platform” [21]. According to this definition, the IDN Information Model is not concerned with details but it aims at capturing abstractions and fundamental requirements of the entities to be modeled. The adoption of an Information Model thus implies the capability to support a number of concrete Data Models. This capability enables characteristics such as scalability and adaptability of the model in different contexts.

Generic information modeled in IDN is formalized as an aggregation of elementary data units and is structured as a directed graph. Each graph node is a data unit addressed by (at least) an URI and contains data and metadata. Graph nodes are related each other through two types of directed links: aggregation and reference links (see Figure 1). Such links allow defining the IDN document as “a set of nodes related each other by aggregation links”. In Figure 1, element “A” represents the root of IDN document “Doc1”. Applying the definition recursively, every document can be considered as an aggregation of IDN documents. Documents are indirectly related each other through reference links. Aggregation and reference links cannot create cycles in the graph; therefore, the graph composed of nodes, aggregation and reference links, is a DAG1. Cross references

1 It's worth recalling that a tree structure is a specific case of DAG in which each node has at most one parent. DAG structures are therefore capable of modelling all tree-structured resources [22].
between documents are handled by IDN applications. This is done through a third type of link named IDN-Application-level link. Such types of link are handled transparently by the IDN middleware and are used by the IDN Application to provide context-dependent relation between nodes.

In order to meet the collaborative data management functions to be provided by the IDN-Service Architecture, IDN-IM complies with the following properties:

1. each DAG node corresponds to an elementary information and can contain unstructured content; each node can be parent of a series of nodes and can contain a series of unrelated and heterogeneous data;
2. aggregation links express parental relations among DAG nodes inside a document (in Fig. 1 information B is parent of information D);
3. reference links express relation between two documents (in Figure 1 document A is refers to document E);
4. an entity responsible has to be associated to each node corresponding to a elementary information. This entity is entitled to guarantee correct logical relations between nodes and the accuracy of node contents;
5. the node content inherits the responsible of the node itself;
6. authorship and responsibility of a node are separated properties;
7. every DAG node can be defined inside an historical information logical structure and it is a part of it. IDN handles historical information with a built-in UEVM-compliant [23] versioning model. This definition is orthogonal and independent from DAG information logical structure, so it extends the information model to history and versioning capabilities.
8. In IDN-IM documents structure and content are completely separate from their presentation.

4. Providing global data addressability

In order to address the design and the development of an actual truly collaborative distributed information management middleware, IDN architecture has to account for the possibility to make the logical identification of the resource (nodes, authors, entities responsible etc.) independent from theirs physical locations. This is different from what happens in current “Web of document” architecture where an URL (Uniform Resource Locator) both identifies the resource and locates it. This feature is enabled by the presence in IDN of a three layers naming space (see Figure 2):

1. Uniform Resource Locator (URL) is used to identify a resource copy as well as to access it;
2. Uniform Resource Name (URN) [24, 25, 26] is used to unambiguously, univocally and persistently identify the resource independently of its physical location;
3. Human Friendly Name (HFN) [27] is used to affect flexible and descriptive names easily interpretable by people.

URLs, URNs and HFNs are subclasses of URIs. IDN middleware hides and automatically manages URNs and URLs. Humans and IDN-compliant application use HFN names to specify resources.

Each end user can name the same resource in a personalized way through many HFNs (alias). As any resource can be copied in several locations, there is a one-to-many correspondence between URNs and URLs (see Figure 2). Resolution processes are required to access a resource starting from its HFN. As all names are sub-classes of URI name category, they are natively hierarchical and their direct and inverse resolution is possible using DNS-like (Domain Name System) systems². A Logical Domain Name System (LDNS) is envisaged to resolve HFN names into a URN name; a Localization Service (LS) is instead devoted to the resolution of a URN name into URL addresses (see Figure 2). The DNS-like approach to resolution allows scalability of the system as well as hierarchical and local delegation of responsibility in the collaborative creation, management and access to data.

² This is analogous to what happens in Web environment as for DNS resolution acting on Application level addresses versus IP addresses and vice versa.
5. Providing basic collaboration-oriented services for data management

Analyzing the requisites recognized as fundamental in different areas and in different solutions currently available in the domain of computer supported collaborative applications [28, 29, 30, 31] we derive the requirements to be met by the proposed IDN-Information Model. Information should be consequently modeled basing on its native global characteristics, regardless from specific needs of the application context.

To this end, IDN layered architecture will provide the following functionalities to the IDN-compliant application:
- **Identity management services** to support users profile management, ACL (Access Control List), etc;
- **Basic structure management** to support creation, modification, access, deletion of IDN-IM nodes and their reciprocal relations;
- **Version control** to handle IDN-IM (UEVM compliant) versioning model;
- **Workflow management**; to provide basic functions to support information work processes;
- **Replica management** to enable augmented system reliability and scalability through built-in replication subsystems;
- **Naming management** to allow users to manage aliases (HFN) to IDN nodes;
- Other supplementary functions such as query management, transaction management, locking, publish and subscribe functions etc., depending which needed by specific applications.

The effective achievement of collaboration in a distributed network environment is thus enabled due to the overall set of above functions provided by the architecture, considered as a whole.

5.1 InterDataNet Service Architecture (IDN-SA)

InterDataNet Service Architecture (IDN-SA) has been designed to the end of providing the basic collaboration-oriented services described in paragraph 5. As it is shown in Figure 3, the four layers composing the IDN-SA architecture are:
1. Storage Interface Layer;
2. Replica Management Layer;
3. Information History;

The IDN-compliant Application is built on top of the Virtual Repository layer exposing the IDN interface.

![Figure 3. IDN layered architecture](image-url)

IDN-SA layers functions are hereafter briefly specified, starting the description from the bottom of the stack.

**Storage Interface Layer (SI)**; it provides the services related to the uniform data storage of distributed and heterogeneous data sources. The SI layer provides a REST-like uniform view over distributed data independently from their location and physical storage platform. This layer is eventually devoted to provide physical addressability to resources through URLs addresses. Basic SI functions provided to the upper layer are: access, creation, modification, deletion of URL-addressed resources.

**Replica Management Service Layer (RM)**; this layer provides a delocalized view of the resources to the upper layer offering URN to URL address resolution (see Figure 2). This enables a twofold consequence: from one side, the upper layer can handle only resources univocal and persistent identifiers (URN); from the other side, it allows the association of several physical resources to the same identifier. This layer is charged of treating this set of physical resources as replicas of the same logical information providing replica updating and synchronization. In principle the creation of a replica can be done with any criteria. For instance, as it is the case of P2P applications, each user can declare his availability to become provider of a replica or the level itself can apply an internal replication policy to guarantee resource availability, performance and reliability.

**Information History Service Layer (IH)**; this layer deals with the history management of information
nodes. Navigation into the versions of non-structured information is consequently allowed by the services provided by this layer. Information nodes inside the history are addressed by standardized symbolic version identifiers (“first”, “prev”, “next”, “last”, etc.) suggesting relative node position in the container starting from previous or subsequent node. Direct access to a specific version is still possible through an absolute version identifier. IH layer can also maintain multiple development branches simultaneously; branching and merging functionalities are therefore exposed to upper layer. This layer is also devoted to managing locking functions on the information to be collaboratively accessed and edited.

Virtual Repository Service Layer (VR); it exposes the IDN interface to the applications implementing all needed functionalities exploiting lower layers services. VR provides the application with the maximum abstraction of structured information. Indeed, this layer is seen from the Application level as the container-repository of all IDN-compliant information.

One of the main functions of this layer is the resolution of logical resources names (HFN) into unique identifiers optionally qualified by a symbolic version identifier (URN with version parameters) (see also Figure 2).

This layer is also in charge of user access control to the system (Identity Management Service).

The main service presented from VR layer to IDN compliant applications allows browsing, edition, creation, deletion etc. of IDN information and their aliases. VR is also responsible of the versioning management of the structured information through the propagation of the versions of the information managed. VR uses a UEVM-compliant versioning model in order to handle structured information versions. It performs versions propagation to parent node starting from the node involved in structural or data content changes.

6. From the IDN middleware towards collaborative applications

From a methodological point of view, IDN approach is oriented to information virtualization, as proposed in [32, 33, 34]. Information virtualization is achieved through the introduction of layers supporting the abstraction from lower level access, storage, processing and communication of structured information. In such a way the complexity of these processes at the application level can be reduced, and consequently new added-value applications development can be enabled.

Let’s consider three very well known social software widely used by Internet users: Flickr\(^3\), Blogger\(^4\) and Youtube\(^5\). These applications allow to manage and share multimedia content with a certain degree of freedom in their granularity: Flickr allows to manage “elementary data” such as images, photosets, users profiles, comments, tags, groups, discussions, notes etc.; Youtube is similar to Flickr but it handles mainly videos instead of images. Blogger allows managing personal diaries and logs combining elementary data such as text, images, and links to other blogs, web pages, and other media related to a certain topic. Interoperability between elementary data managed by such applications can be achieved through the use of specific APIs which are exposed by each application provider. The possibility to granularly reuse and combine elementary data stored on each provider is thus conditioned and restrained by both commercial as well as technical choices operated by the providers. For instance it is not possible to post a YouTube video into a Flickr Group; a Flickr image can be aggregated into a Blogger post through a solution (API) specifically developed for the interoperation between Blogger and Flickr. Interoperability eventually is an application-dependent issue where the Information Model adopted depends on the application and is defined by each provider and on a case basis. If one had to aggregate Flickr content with YouTube videos in a single document in the current situation one would have to implement both providers’ APIs in order to enable data fusion and develop the orchestration logic at the application level.

IDN architecture aims to free interoperability from heterogeneous APIs and ad-hoc applications by providing a middleware solution addressing this problem at an infrastructural level. In Figure 4 the example cited illustrates how two IDN-compliant applications, a Blog and a PhotoSharing service, could transparently share their elementary data with other IDN-compliant applications exploiting the services provided by the middleware. In this case, through this solution content providers become naturally interoperable and don’t need to develop and expose specific APIs in order to allow global access to their resources.

In Figure 4 IDN approach for supporting Application for collaborative multimedia content management is illustrated.

\(^3\) http://www.flickr.com/
\(^4\) http://blogger.com/
\(^5\) http://www.youtube.com/
Figure 4. IDN middleware serving generic computer supported collaborative applications

IDN Service Oriented Middleware exposes to the Application a REST interface. The layered architecture provides a set of collaboration–enabling functions (structure, versioning, provenance, workflow, replica management etc.) for the management of a shared IDN Information Model. Global addressability on elementary data units is achieved through a shared naming system allowing the attribution of aliases at the Application level, the handling of unique identifiers at the IDN middleware level and its eventual resolution into physical addresses of the resources at the bottom of the architecture.

7. Concluding remarks

The IDN (InterDataNet) conceptual system model has been presented as the grounding pattern of a middleware capable of providing extensive support for collaborative data management applications. In the IDN approach documents are broken down into simpler information units to be addressed with sufficient granularity up to allow their effective processing, integration and, ultimately, interoperability.

IDN sustains global addressability of concepts and resources as well as basic collaborative oriented services for distributed and heterogeneous data management.

This model fits the principles of Service Oriented Architectures while enhancing intrinsic information interoperability and support network collaboration functions. The IDN conceptual framework is composed of the Information Model (IDN-IM) and the layered Service Architecture (IDN-SA).

Analogously to what happened for the adoption of the HTTP protocol and HTML language as for the infrastructural management of hypertext documents, the IDN approach can shift collaboration-related issues in the management of elementary information from the application level to the infrastructural level.

The wide adoption of IDN middleware could add to current internetworking solutions interdataworking features allowing interoperability between data.

At present the IDN conceptual framework is specified through the detailed Information Model definition and the specification of the API exposed by the Virtual Repository to the IDN-compliant Application.

A prototypal implementation of the IDN Service Architecture is underway according to two different approaches: one envisages the bottom-up development of the SA layers which is functional to the a medium-long term completion of the IDN architecture implementation; the other is a top-down development aiming at considering IDN middleware as a black-box to come to a short-term partial IDN-API implementation based on the available content repository Apache-JackRabbit6. The latter approach will provide the suitable conditions for the development and testing of experimental compliant applications.

References


6 http://jackrabbit.apache.org/


Steve Vinoski, "REST Eye for the SOA Guy," IEEE Internet Computing, vol.11, no. 1, pp. 82-84.


