

Information Object Design Pattern for Modeling Domain Specific Knowledge

Nitin Arora¹, Rupert Westenthaler¹, Wernher Behrendt¹, Aldo Gangemi²

¹ Salzburg Research Forschungsgesellschaft m.b.H.,
A5020 Salzburg, Austria
{<first name>.<last name>}@salzburgresearch.at
²Laboratory for Applied Ontology, ISTC-CNR, Roma (Italy)
aldo.gangemi@istc.cnr.it

Abstract. We report on a standardized container object called *Knowledge Content Object* for the management of rich media content which is enhanced by representations of domain knowledge. *KCOs* are based on the *Information Object Design Pattern* originally developed for the DOLCE foundational ontology. We describe a methodology for stepwise refinement of the domain specific knowledge in these container objects. The work aims at bringing semantic web technologies and object-oriented software engineering closer together.

1 Introduction

Modeling domain specific knowledge plays an important role throughout the software development cycle. Since many applications nowadays are web-based, there is an increasing number of cases where domain specific knowledge is not only needed to drive a particular application, but also to describe in a formal fashion, the content that is being exchanged between service oriented applications. Knublauch et al. in [1] highlight the use of semantic web technologies (like RDF [2] and OWL [3]) for Object Oriented Software Developers and how the Semantic Web [4] can serve as a platform on which domain models can be created, shared and reused.

This paper discusses Knowledge Content Objects which are derived from the Information Object Design Pattern (IO Design Pattern) of the DOLCE foundational ontology [5]. The KCO provides a framework in the form of semantic facets. Section 2 discusses the motivation behind the IO Design pattern followed by an outline specification in Section 3. Section 4 illustrates KCOs through an example.

2 Motivation for the Information Object Design Pattern

The Information Object Design Pattern (see Figure 1) is an ontological design pattern specified in the DOLCE foundational ontology [5]. The IO Design pattern is based on the fact that *conceptualization* and the actual *realisation* of an entity are two different

aspects. For instance a website may be conceptualized by sketching out its design, and by noting down the content that needs to be put on it and the nature of it. But it is only *realized* when a web designer creates the web site in HTML, i.e. once an actual instance of it exists. The rationale behind this design pattern is that any *content object (entity)* can realize a socially constructed object called **information object (IO)**. For our example the concept of a web site will refer to information object and the actual realisation is the content object.

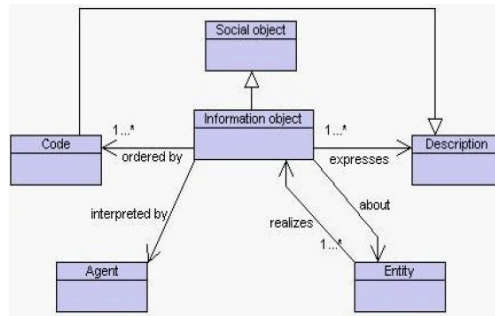


Figure 1: DOLCE Information-object design pattern

The information object carries meaning on the cognitive levels of the *agent*. *Agent* refers to an agentive role, the combination of a social or a physical agent which is playing a distinct role while interacting with entities. In the above figure, *entity* plays two distinct roles. First it refers to a *physical entity* (the actual data or content) which realizes the information object. Entities playing this role are referred to as **information-realizations**. Second it refers to something (which can be physical or virtual) which the information object is *about*. For example, a news text is an information realisation of the information object about a real event like a soccer game as interpreted by the news agency. The expressed description of an information object describes the *meaning/conceptualization*. These semantic descriptions are created and modified by agents and are therefore - like the information object - defined as social objects. Coming back to our example of a website, the graphical design and the nature of the website will be the descriptions.

3 Information Object Design Pattern: Implementation

Figure 2 shows the UML diagram of the three aspects in the Information Object Design Pattern: Information Realisation, Information Object and Description. The interface *Information Realisation* handles the actual content or data. *Information Object* is the conceptualization (i.e. the subject matter which has no direct realisation in a target system). Interface *Description* gives information about the content or subject matter. For example: a digital picture of an alpine landscape will be an *Information Realisation* whereas its meta data descriptions like the encoding format (e.g. "format = jpg")

will be the description part. *Information Object* in this case will be the mental concept of “Alpine Landscape”.

4 Usage of the Information Object Design Pattern in KCOs

The KCO is derived from the Information Object Design Pattern. It is divided into *six "facets"* which are semantically distinct components that refer to the actual content and the related domain specific meta data or knowledge (Table 1).

Table 1. Facets of a KCO.

Facet	Short Description
Content Description	This is subdivided into three facet elements: access information, meta data schemes and subject matter knowledge about the content
Presentation Description	describes of how the content (and the Knowledge) of the KCO is presented to users and specifies modes of interaction with users
Community Description	This contains descriptions of Plans, Tasks, Roles and Goals in the context of a community which uses the KCO
Business Description	This specifies how to trade the content, including the specification of business models and negotiation protocols
Trust & Security	To specify methods that ensure security and trust for KCO users
Self-description	This declares the structure of the KCO itself, including active facets, ontologies used, and other KCO system related information

The Content Facet consists of *three layers of descriptions*: the *first layer* describes access information and storage information about the content, the *second layer* describes the meta data level information like format(s), encoding, etc and the *third layer* provides knowledge-level statements about the actual subject of the content. These third layer statements refer to a domain ontology and contain instance-specific domain information. The content facet of a KCO may contain multiple content objects. ContentSegment can be any part of a ContentObject, e.g. a specific paragraph on a web page. ContentClassificationDescription holds the classification of the content. Domain specific information is part of this ContentClassificationDescription. The first and the second layers are realized through the ContentProfileDescription and ContentSegmentDescription which describe encoding, format and storage information about the actual content. The third layer is realized by the ContentClassification along with ContentClassificationDescription which describes/classifies the content subject and attaches the domain specific information. Coming back to our example of a digital picture of a landscape, the ContentClassification represents the concept of an image of an alpine landscape (with a lake and some mountains) and the ContentObject is its actual realisation (for example the image on the web or on the file system). ContentSegment is a cropped image say of a particular mountain peak visible in the image. The ContentProfile is the handler for the description that deals with the ContentObject. Analogously, ContentSegmentProfile is the handler for the description that deals with the ContentSegment. ContentProfileDescription describes the meta data informa-

tion for example the format of the landscape image, information about the image storage location (like the URL). ContentSegmentDescription gives storage information (like the URL which can be same as that of the main image) about the sub-image i.e. the mountain peak in our example. Domain specific information can be wrapped in the ContentClassification descriptions for instance expressing that the picture was taken on the family holiday to the Swiss Alps and that the peak in question is the Matterhorn, etc. Typically the Content Facet will contain content objects with multiple content segments and their appropriate profiles and descriptions.

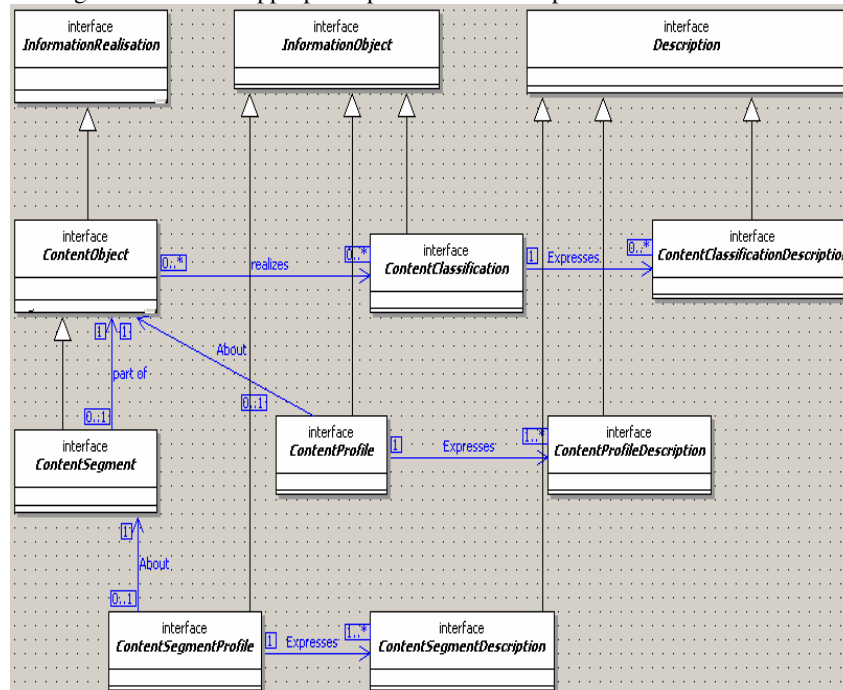


Figure 2: UML diagram of the Content Facet of a KCO. The arrows show class hierarchy and the blue lines are association links with appropriate labels and cardinality values.

References

1. Knublauch, H., Oberle, D., Tetlow, P., Wallace, E.: "A Semantic Web Primer for Object-Oriented Software Developers", W3C Working Group Note, 9 March 2006.
2. Manola, F., Mille, E.: RDF Primer, W3C Recommendation, 10 February 2004.
3. Dean, M. and Schreiber, G.: Web Ontology Language (OWL) Reference Version 1.0. Technical report, World Wide Web Consortium (2003).
4. W3C Semantic Web: <http://www.w3.org/2001/sw/>
5. Gangemi, A. et al, 2002, Sweetening Ontologies with DOLCE, in Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02), Springer, Okt 1-4, Spain, pp. 166-181.