Architecture

Meta model

This is where all the magic takes place. Inference, rules, linear prediction and many others forms of enrichment on the underlying data are provided here transparently and made available for models who populate an consume this enriched knowledge. Models below use an specification mechanism (mapping rules and transformation) to populate themselves via their ModelManager.

Models

Trying to overcome the limitation of actual semantic environments, like RDF, being is a so universal model for everything that renders itself useless when trying to model real world simple example of business entities in a real use case.

And trying to overcome all the actual needs with it of shared vocabularies and upper ontologies just to make one model understand each other here we model everything as instances of models and, our underlying knowledge layer is not exposed so one can talk in business terms instead of triples. The meta model layer as as the underlying knowledge base being mapped with bidirectional translation semantics to the model instances.

Common object model

For the three items upcoming this one, there is a need of a 'shared' domain object model for exposing two features of the supported models: Activation and Analysis, so they can talk each other and to the Application(s) through the Interface exposure in a protocol 'understandable' form.

Activation and Analysis

Activation and Analysis are all about creating, retrieving, updating and any other domain specific manipulation of business entities, plus 'analysis' like further contextual queries with dimensional and mining semantics (aggregation, etc).

Interface

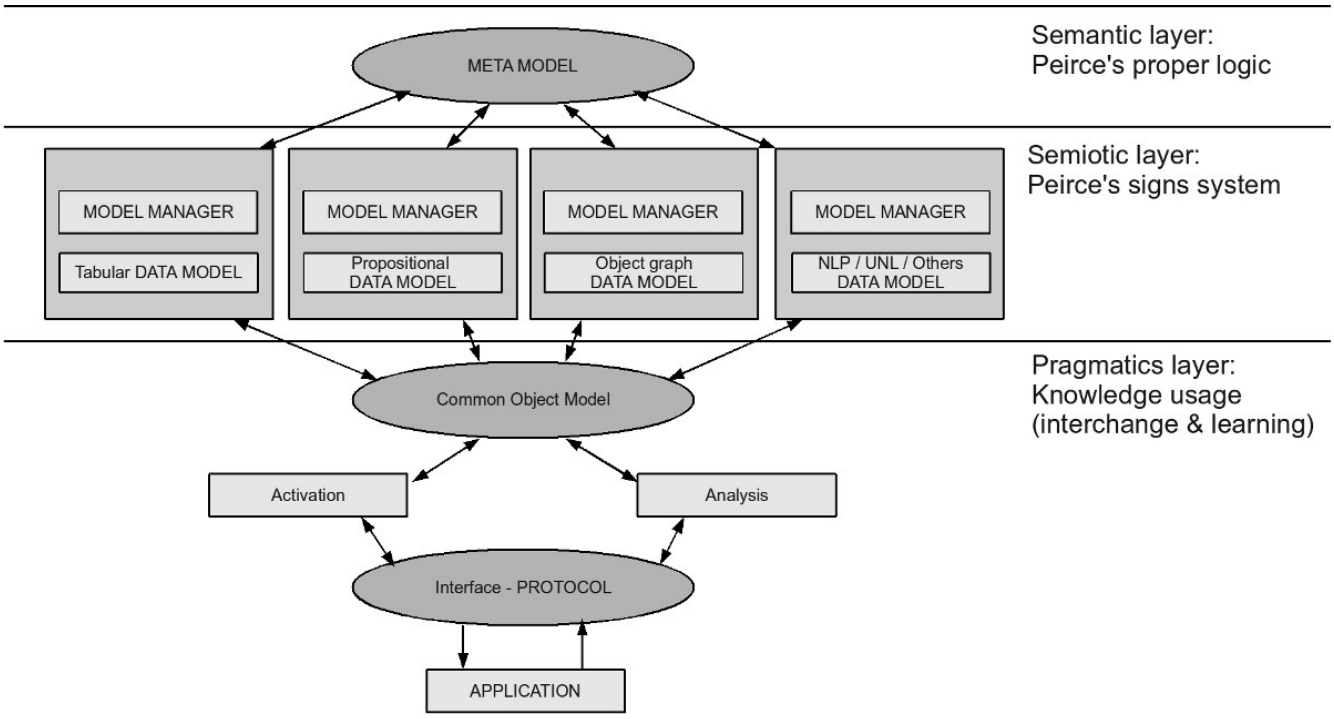
We should expose an interface to our applications so they are not tied nor bound to any of our implementation or model specific details. So there must be a protocol supporting on top of the knowledge base the whole interactions an application could need to have with it.

Application

An application could render a user interface, being exposed as a service or whatever, with whatever semantics and exposure it likes, leveraging the support brought by the lower layers of the architecture.

Diagram

The model below show the implications of each entity belonging to each layer, mostly because of their relationship between the Peirce's model of Semiotics (Semantics, Signs and Pragmatics).



On pragmatics

Pragmatics, as implemented on the systems, are to be understood as the interchange of knowledge, cross evaluation of statements and sharing of the outcomes with the purpose of learning between the system and the client, being this a user, or another knowledge base. More explanation is needed here that supports this approach but the cornerstone of the methods that will be used are the ones described above.