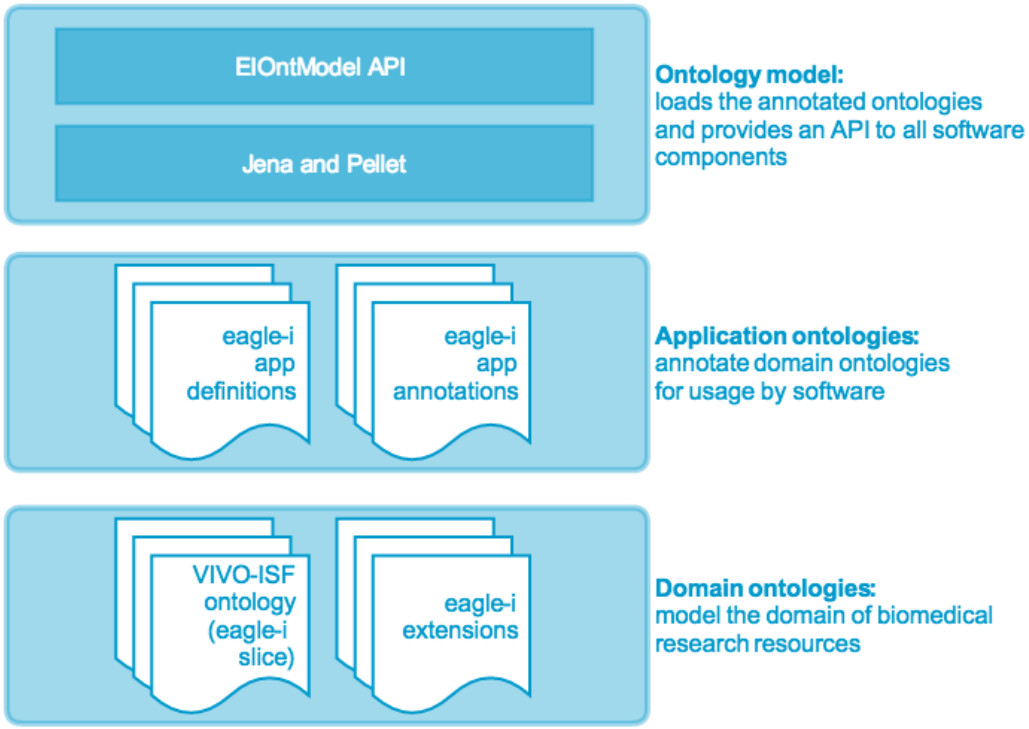
eagle-i Application Ontologies

eagle-i software is predicated upon the principle of ontology layering. Applications are driven by an *application layer ontology* that complements and annotates a *domain ontology*. An application layer ontology encapsulates the specifics of the software behavior and is not meant to be reusable in and of itself, in contrast to a domain ontology. The domain ontology used in the current eagle-i stack is the ERO (eagle-i resource ontology), as derived from the ISF (Integrated Semantic Framework). The layering principle is shown in the figure below, and is applicable to other domain ontologies.

The runtime model that represents the ontology to the applications is built by loading the domain and application ontologies and interpreting the annotations in different ways. The memory model is organized as a core model fully stored in memory and a collection of referenced taxonomy models stored in Solr indexes for rapid access.



Important files

- eagle-i-app-def.owl contains axioms that define the basic annotation vocabulary (classes and properties).
- ero-app.owl serves as a root for the software loading mechanism. It provides the chain of imports - if an owl file is reachable directly or indirectly from this file via an owl:import statement, it will be loaded as part of the core model. It is a configurable parameter in the software.
- Files named *-app.owl contain the axioms that annotate the domain ontologies. There is typically one app file per domain owl file (e.g. disease-imports-app.owl, disease-imports.owl)
    - Note that ero-app.owl serves this purpose for the ero.owl file.
    - (describe mechanism for loading referenced taxonomy files into solr)

Requirements of *-app.owl files

- Every app file must declare an ontology IRI
- Every app file must declare an ontology version
- Every app file must import its corresponding domain owl file