PROTON Ontology

Version 3.0 Beta

This document describes the third version (beta) of PROTON ontology. This new version extends the version from 2005 with new classes and properties in order to cover the conceptual knowledge encoded within the most popular datasets from Linked Open Data[1] (LOD) like DBPedia, GeoNames, etc. The document is based on the documentation of 2005 release of PROTON ontology[2].

The PROTON (PROTo ONtology) ontology has been developed in the SEKT project[3] as a lightweight upper-level ontology, serving as a modelling basis for a number of tasks in different domains. To mention just a few applications: PROTON is meant to serve as a seed for ontology generation (new ontologies constructed by extending PROTON); it can be used for automatic entity recognition and more generally Information Extraction (IE) from text, for the sake of semantic annotation (metadata generation).

The document first briefly discusses the design principles, the basic structure and the scope of the onotology. Then it presents the architecture and the modules of PROTON.

1 Design Rationales

PROTON is designed as a lightweight upper-level ontology for use in Knowledge Management and Semantic Web applications. The above mission statement has two important implications:

- PROTON is relatively un-restrictive. It specifies only a hierarchy of classes and domain and range of properties defined within it, but it does not impose any other restrictions on the meaning of the classes and properties.
- PROTON is not precise in some aspects, for instance regarding the conceptualization of space and time. This is partly because proper models for these aspects would require using a logical apparatus, which is beyond the limits acceptable for many of the tasks to which we wish to apply PROTON (e.g. queries and management of huge datasets/knowledge bases); and partly because it is very hard to craft strict and precise conceptualizations for these concepts, which are adequate for a wide range of domains and applications.

Having accepted the above drawbacks, we add two additional requirements to PROTON namely to allow for

- low cost of adoption and maintenance and
- scalable reasoning.

The first point requires an easy understanding of a set of classes and properties and the second requires a tractable algorithm for reasoning with the represented conceptual knowledge. Thus,

[1] http://linkeddata.org/

[2] For detailed description of the design principle of PORTON ontology, please, consult the original documentation: Ivan Terziev, Atanas Kiryakov, Dimitar Manov. 2005. Base Upper-level Ontology (BULO) Guidance. Deliverable 1.8.1, SEKT project, July 2005, (http://proton.semanticweb.org/D1_8_1.pdf) and Atanas Kiryakov. 2006. Ontologies for Knowledge Management In "Semantic Web Technologies: Trends and Research in Ontology-based Systems"; John Davies (Editor), Rudi Studer (Co-Editor), Paul Warren (Co-Editor). pp. 115-138 John Wiley & Sons, Europe, April 2006. ISBN: 0-470-02596-4.

[3] http://www.ontotext.com/research/sekt

the goal is to make feasible the usage of ontologies and the related reasoning infrastructure (with all their attendant advantages discussed above) as a replacement for the use of DBMSs.

Being lightweight, PROTON matches the intuition behind the arguments coming from the Information Science community, (Sparck Jones 2004)[4] and (Shirky 2005)[5], that the Semantic Web is more likely to yield solutions to real world information management problems if it is based on partial and relatively simple models of the world, used for semantic tagging.

2 Basic Structure

The PROTON ontology contains about 500 classes and 150 properties, providing coverage of the general concepts necessary for a wide range of tasks, including semantic annotation, indexing, and retrieval. The design principles can be summarized as follows:

- domain-independence;
- light-weight logical definitions;
- alignment with popular metadata standards;
- good coverage of named entity types and concrete domains (i.e. modelling of concepts such as people, organizations, locations, numbers, dates, addresses, etc.); and
- good coverage of instance data in Linked Open Data Reason-able view Fact Forge[6].

The ontology is encoded in a fragment of OWL Lite and split into four modules: System, Top, Extent, and KM (Knowledge Management). A snapshot of the PROTON class hierarchy is given on Figure 1, showing the Top and the Extent modules.

[4] Spark Jones, K. 2004. What's new about the Semantic Web? Some questions. SIGIR Forum December 2004, Volume 38 Number 2. http://www.sigir.org/forum/2004D-TOC.html

[5] Shirky, Clay. 2005. Ontology is Overrated: Categories, Links, and Tags. Clay Shirky's Writings About the Internet. Economics & Culture, Media & Community. http://www.shirky.com/writings/ontology_overrated.html

[6] http://www.ontotext.com/factforge

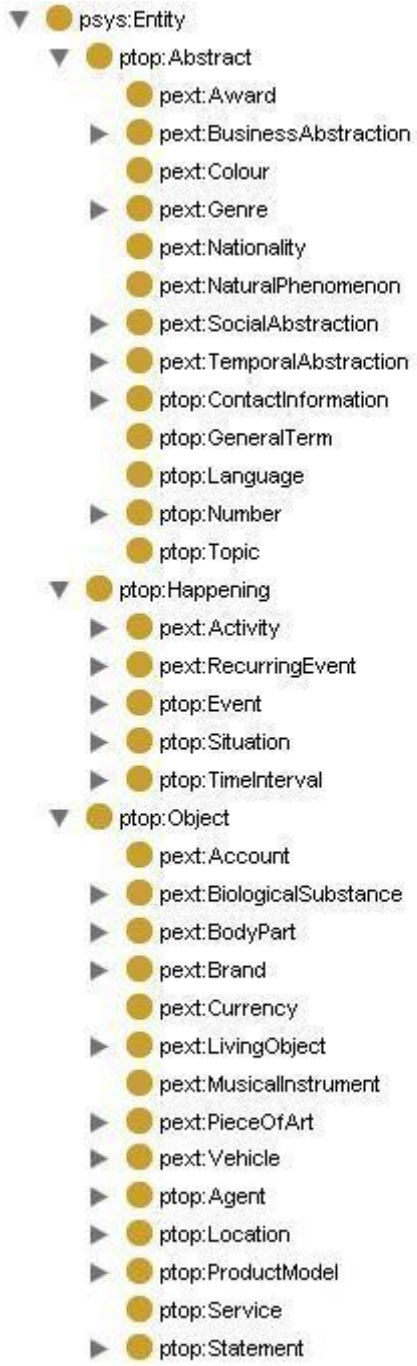

Figure 1. A view of the top part of the PROTON class hierarchy

PROTON is presented in greater detail in (Terziev et al. 2005)[7]. The development of the ontology continues under a collaborative "community process" organized in accordance with the DILIGENT methodology[8]. In the following sub-sections we provide an overview of its core module and its structure.

3 Scope, Coverage, and Compliance

[7] Ivan Terziev, Atanas Kiryakov, Dimitar Manov. 2005. Base Upper-level Ontology (BULO) Guidance. Deliverable 1.8.1, SEKT project, July 2005, (http://proton.semanticweb.org/D1_8_1.pdf)

[8] Sofia Pinto , Steffen Staab , Christoph Tempich. 2004. DILIGENT: Towards a fine-grained methodology for DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies. In Proceedings of the 16th European Conference on Artificial Intelligence. pp. 393-397.

The extent of specialization of the ontology is partly determined on the basis of case studies within the scope of the SEKT project and on a survey of the entity types in a corpus of general news (including political, sports, and financial ones). Additionally, several datasets in FactForge were studied and relevant concepts were created. The distribution of the most commonly used entity types varies greatly across domains. Still, as reported in (Maynard et al. 2003)[9], there are several general entity types that appear in the large majority of corpora (text collections) – **`Person`**, **`Location`**, **`Organization`**, **`Money`** (**`Amount`**), **`Date`**, etc. The proper representation and positioning of those basic types was one of the objectives in the PROTON design and this was accomplished, for the most part of it, at the level of PROTON Top module layer (for details see next section).

The rationale behind PROTON is to provide a minimal, but nevertheless sufficient ontology, suitable for semantic annotation, a unifying ontology for FactForge, as well as a conceptual basis for more general Knowledge Management (KM) applications. A number of upper-level resources inspired its creation and development: OpenCyc[10], Wordnet[11], DBPedia Ontology[12], UMBEL[13], GeoNames (GNS) Ontology[14], DOLCE[15], EuroWordnet Top (Peters 1998)[16], and others.

The development philosophy of PROTON is to keep it compliant, with most of the important standards and ontologies in Semantic Web and LOD.

4 The Architecture of PROTON

PROTON is organized in three levels, including four modules. In Figure 2 the levels are layered from left to right. The System ontology module occupies the first, basic layer; then the Top, and Extent, and KM ontology modules are provided on top of it to form the diacritical modular architecture of PROTON.

The System module is an application ontology, which defines several notions and concepts of a technical nature that are substantial for the operation of any ontology-based software, such as semantic annotation and knowledge access tools. It includes the class **`psys:Entity`** – the top ("master") class for any sort of real-world objects and things, which could be of interest in some areas of discourse. In the system ontology it is defined that entities (i.e. the instances of **`psys:Entity`**) could have multiple names (strings related to a given entity via the data property **`psys:name`**) that information about them could be extracted from particular **`psys:EntitySource`**-s, etc.

[9] Maynard, D.; Tablan, V.; Bontcheva, K.; Cunningham, H.; Wilks, Y. 2003. MUlti-Source Entity recognition – an Information Extraction System for Diverse Text Types. Technical report CS--02--03, Univ. of Sheffield, Dep. of CS, 2003. http://gate.ac.uk/gate/doc/papers.html

[10] http://www.opencyc.org

[11] http://www.cogsci.princeton.edu/~wn/

[12] http://wiki.dbpedia.org/Ontology

[13] http://umbel.org/

[14] http://www.geonames.org/ontology/documentation.html

[15] http://www.loa-cnr.it/DOLCE.html

[16] Peters, W. (ed). 1998. The EuroWordNet Base Concepts and Top Ontology. Version 2, Final. January 22, 1998. http://www.illc.uva.nl/EuroWordNet/corebcs/topont.html

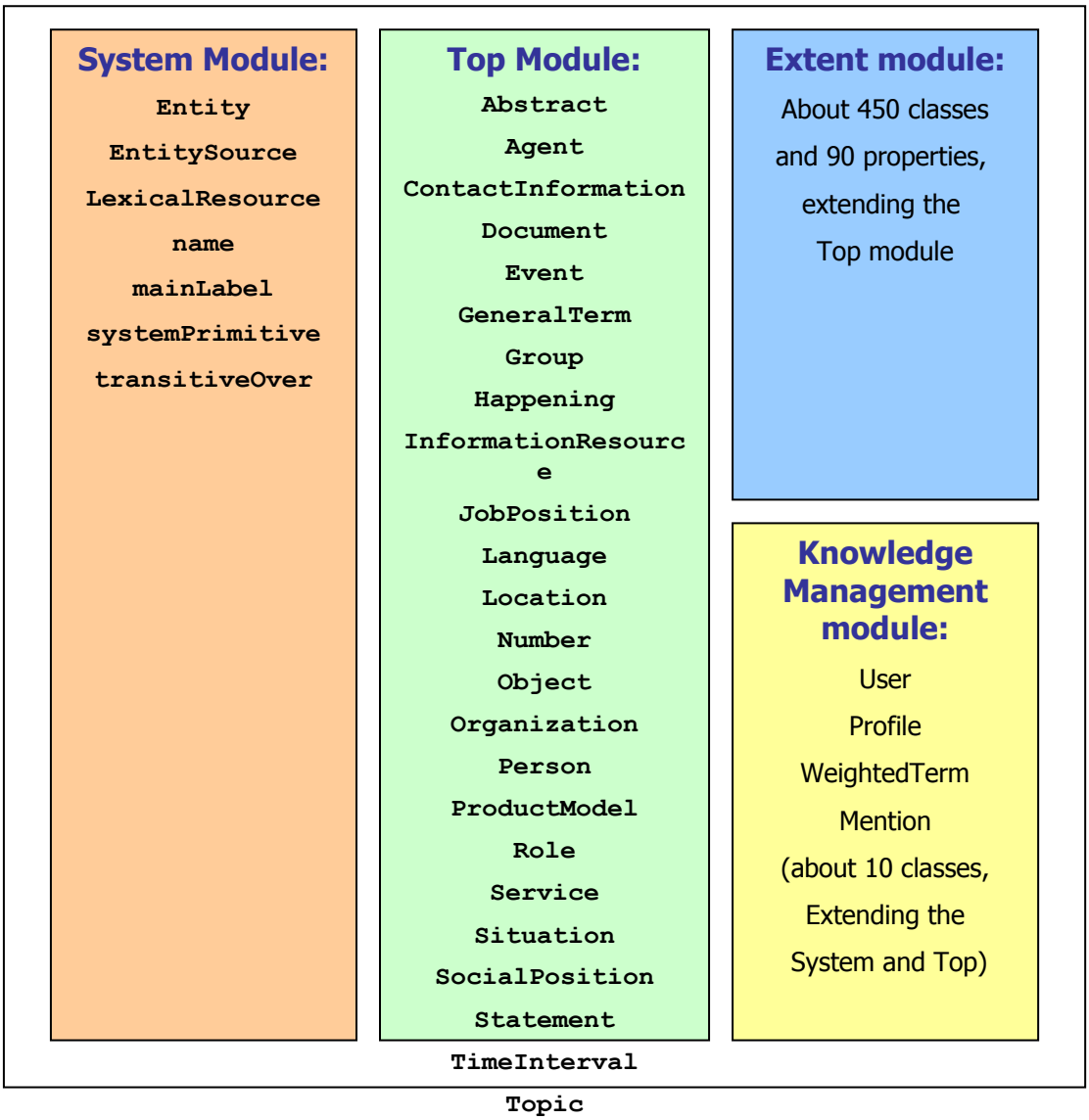

Figure 2. PROTON (PROTo ONtology) modules

The Top ontology module starts with some basic philosophically-reasoned distinctions between entity types, such as **`Object`** – existing entities (agents, locations, vehicles); **`Happening`** – events and situations; **`Abstract`** – abstractions that are neither objects, nor happenings. The design at the highest level of the Top module follows the stratification principles of DOLCE, through the establishment of the PROTON trichotomy of Objects (**`dolce:Endurant`**), Happenings (**`dolce:Perdurant`**), and Abstracts (**`dolce:Abstract`**). The same stratification is also defined in (Peters 1998). According to many experts in upper-level ontology construction, e.g. (Guarino 1998a) and (Peters 1998), an important ontology design principle is that the extensions of these three hierarchies should be disjoint, i.e. no individual should be an instance of more than one of these three top classes. One of the reasons for the introduction of this guiding principle is to avoid the "overloading" of the subsumption (sub-class-of, is-a) relation.

These three classes are further specialized by about 20 general classes. These include **`Agent`**, **`Person`**, **`Organization`**, **`Location`**, **`Event`**, **`InformationResource`**, besides abstract notions, such as **`Number`**, **`TimeInterval`**, **`Topic`**, and **`GeneralTerm`**. The featured entity types have their characteristic attributes and relations defined for them (e.g. **`subRegionOf`** property for **`Location`**-s, **`hasPosition`** for **`Person`**-s; **`locatedIn`** for **`Organization`**-s, **`hasMember`** for **`Group`**-s, etc.).

PROTON extends into its third layer, where two independent ontologies, which define much more specific classes, can be used: the PROTON Extent module and the PROTON KM (Knowledge Management) module. Examples from the Extent module are: **`Mountain`**, as a specific type of **`Location`**; **`ResourceCollection`** as a sub-class of **`InformationResource.`** Having this ontology as a basis, one could easily add domain-specific extensions.

The 2011 version of PROTON has changed in several ways. Some classes were substituted with new ones in order to facilitate the usage of the ontology. Some new classes were added in order to extend the coverage of the ontology. In the following section we present an overview of the changes for each module. We also changed the prefixes for each module. The classes and properties in PROTON KM module are the same as in the previous version of the ontology. This is why we do not provide a description of this module here. The reader has to consult SEKT Deliverable D1.8.1.

5 PROTON System module

The new URI for this module of PROTON ontology is **`http://www.ontotext.com/protonsys`** (abbreviated to **`psys:`**). Here, we first describe classes included in the module, and then we present the differences with the previous versions. The PROTON System module of PROTON provides a sort of high-level system- or meta-primitives, which are likely to be accepted and even hard-coded in particular tools that may use PROTON. The PROTON System module of PROTON includes the following classes (depicted in Figure 3): **`psys:LexicalResource; psys:EntitySource; psys:Entity`**.

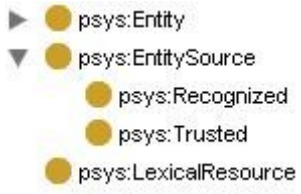

Figure 3. PROTON System module

The **`psys:Entity`** class hiearchy represents the root of the "true" ontology (the variety of entity classes) – it is the super-class of the PROTON Top module (see below), while the other class hiearchies could be considered auxiliary ones for technical use.

The instances of the **`psys:EntitySource`** class are used to separate the trusted (pre-populated) information in the KB, from the one that is extracted automatically. This is indicated by the **`psys:generatedBy`** property of the specific entity. This is done via the following two sub-classes: **`psys:Recognized`** (it serves to identify a source – like a program or a module – that is able to recognize and generate new entities from a text as part of Information Extraction (IE) tasks; typically those are not checked and therefore not trustable; and **`psys:Trusted`** used to indicate entities imported from trusted sources. An instance of **`psys:Entity`** could be linked to an instance of **`psys:EntitySource`** via the **`psys:generatedBy`** property.

The **`psys:LexicalResource`** class is mostly dedicated to the encoding of various data, related to the IE process, such as company suffixes (AG, Ltd.), first names of persons, etc.

- **`psys:description`** – this property denotes a textual description of an entity, usually a free text in some natural language; as defined in Dublin Core for **`dc:InformationResources`**; in a sense, it is a specialization of **`rdf:comment`**.
- **`psys:laconicDescription`** – denotes an extremely short (typically, a single sentence) description of an entity; a sub-property of **`psys:description`**.

- **psys:generatedBy** – this property identifies the party that introduced the entity into the respective knowledge base; it relates the **psys:Entity** and **psys:EntitySource** super-classes of the PROTON system module.
- **psys:systemPrimitive** - the system classes and properties are used to encode system specific information; those, as well as their instances, and related information, should usually not be presented to the end-user; in practice, user-interface and visualization modules can filter such primitives.
- **psys:transitiveOver** - it suggests that one property is transitive with respect to another one, thus making the modelling of a specific, but rather useful modelling pattern, possible; the semantics is defined with the following axiom:

  ```
  (p,transitiveOver,q) (x,p,y) (y,q,z) => (x,p,z).
  ```

  An usage example of **psys:transitiveOver** follows:

  ```
  (locatedIn, transitiveOver, subRegionOf)
  (Ontotext,locatedIn,Bulgaria)
  (Bulgaria,subRegionOf,Europe) =>
  (Ontotext,locatedIn,Europe)
  ```

The following classes and properties were removed from the previous release of the ontology:

psys:Alias

psys:hasAlias

psys:hasMainAlias

They were removed because in the previous version of the ontology the class **psys:Alias** was assumed to represent different names/nicknames/aliases. The properties were used to attach them to other entities. The problem was that the class required the names/nicknames/aliases to be instances with corresponding URIs, which was inconvenient. To overcome this problem we substituted the two properties with **rdfs:label,** which is a standard way to attach label to classes, properties, instances. In order to have both of the above properties we also included the following property: **psys:mainLabel,** which is a sub property of **rdfs:label**. Additionally, we added a property **psys:name,** which is a more natural way to assign names to objects in the ontology.

6 PROTON Top and Extend modules

This section describes in detail the PROTON Top and Extend modules super-classes, plus some of their more significant sub-classes. The new URIs for these modules are **http://www.ontotext.com/protontop** (abbreviated to **ptop:**) and **http://www.ontotext.com/protonext** (abbreviated to **pext:**). The modules are updated in order to incorporate the extensions necessary for the mapping to the main ontologies of Fact Forge. These top-level classes represent the most common, basic, and inclusive notions of world knowledge.

6.1 Object class hiearchy

The instances of the class **ptop:Object** are entities that could be claimed to exist (in some sense of existence). An object can play a certain role in some happenings. Objects could be material (as the Eiffel Tower or Lenin's body) or immaterial (say, an electronic document). Its proprietary relations and attributes are:

- **ptop:hasContactInfo** – this property allows for relations between the **ptop:Object** and **ptop:ContactInformation** (a sub-hierarchy of **ptop:Abstract**).
- **ptop:isOwnedBy** – this property relates a particular organization to the agents that are members of that organization. This predicate indicates a 'generic' type of membership,

although there may be specialized kinds of membership in the same organization. Typically, membership eligibility is determined by the organization and accepted with the agent's voluntary affiliation. In many cases **ptop:Person**-s that take **ptop:JobPosition**-s within an **ptop:Organization** are considered members of the organization, although this is not encoded here formally in any way.

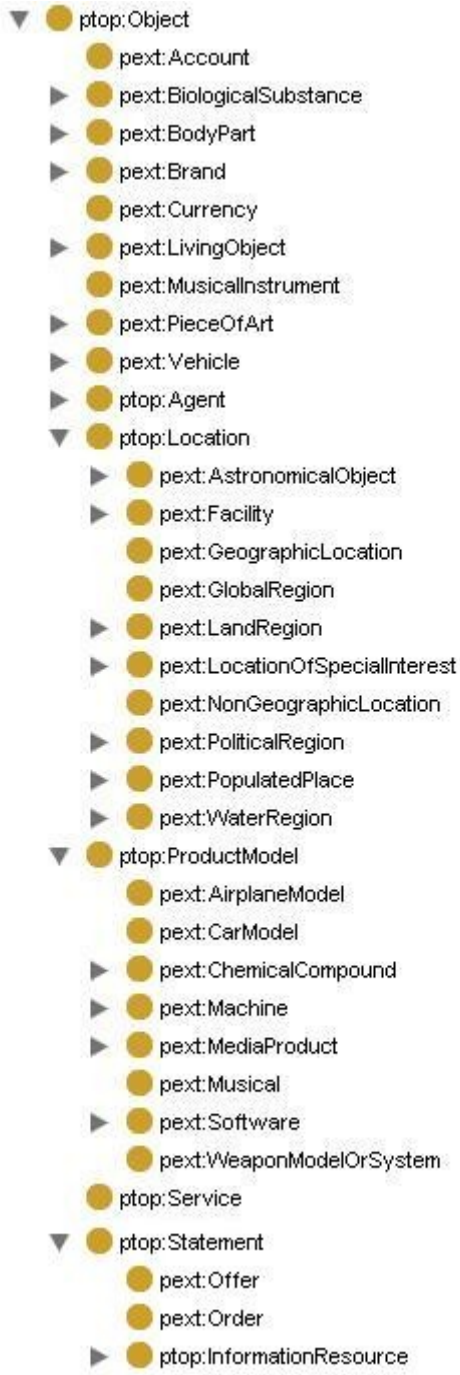

Figure 4: ptop:Object class hierarchy

The **ptop:Object** class is a super-class of the following classes – see Figure 4: **ptop:Agent** (see subsection 6.1.1), **ptop:Person** (see subsection 6.1.2), **ptop:Group** (see subsection 6.1.3), **ptop:Organization** (see subsection 6.1.3), **ptop:Location** (see subsection 6.1.4), **ptop:Statement** (see subsection 6.1.5), **ptop:InformationResource** (see subsection 6.1.6), **ptop:Document** (see subsection 6.1.6), **ptop:ProductModel** (ex. `ptop:Product`) (see subsection 6.1.7), **ptop:Service** (see subsection 6.1.8), and **ptop:CommercialOrganization** (see subsection 6.1.9). Here is a list of classes from Extend module that are direct sub-classes

of **ptop:Object**: **pext:Account**, **pext:BiologicalSubstance**, **pext:BodyPart**, **pext:Brand**, **pext:Currency**, **pext:LivingObject**, **pext:MusicalInstrument**, **pext:PieceOfArt**, and **pext:Vehicle**.

6.1.1 Agent class

ptop:Agent is something that can show (carry out) an independent action, whether consciously or not. Most animals are considered agents, in most contexts; so are most organizations. According to DOLCE 2.0[17], agents are "objects to which we ascribe intentions, beliefs, and desires". **ptop:Agent** here also denotes any automatic services, including web services and servers.

Figure 5: ptop:Agent class hierarchy

Apart from the properties inherited from the **ptop:Object** super-class, **ptop:Agent** has three proprietary ones:

- **ptop:involvedIn** – this property allows for various relations between **ptop:Agent** and members of the **ptop:Happening** class hiearchy, for instance.
- **ptop:hasParticipant** – this is a new property, introduced in order to complete the inventory of properties for happening description. It is an inverse property for **ptop:involvedIn**.
- **ptop:isLegalEntity** – this property determines whether a particular **ptop:Agent** is a legal entity or a non-legal one. Its range should be constrained to ***Boolean***. **ptop:Agent**-s, for which the value is ***True***, correspond to instances of http://www.cyc.com/2003/04/01/cyc#LegalAgent, which is defined as "Each instance

[17] http://www.loa-cnr.it/DOLCE.html

of #$LegalAgent is an agent who has some status in a particular legal system. At the very least, such an agent is recognized by some legal authority as having some kinds of rights and/or responsibilities as an agent (e.g., #$citizens of Germany), ...". In PROTON, it is modelled as a property in order to avoid multiple inheritances of classes and/or multiple classifications of instances.

This property is quite specific and its inclusion as a property of **ptop:Agent** was triggered by practical considerations, as the legal/non-legal connotation of an agent. Because of the relation it provides, it allows the user of a respective application, using the ontology, to narrow his/her search to a great extent.

- **ptop:partiallyControls** – any sort of partial control of an agent with respect to an object; this intransitive relation provides for dependency relations between **ptop:Agent**-s and **ptop:Object**-s.

The sub-classes of **ptop:Agent** are both PROTON Top module classes: **ptop:Person**, **ptop:Group**, **ptop:EthnicGroup**, and **ptop:Organization**. Those are described further in this document in separate sections (6.1.2, 6.1.3).

6.1.2 Person class

ptop:Person (a PROTON Top module class) is an agent (within the meaning of **ptop:Agent** in PROTON), which is an individual who is a human being (i.e. any living or extinct member of the family Hominidae).

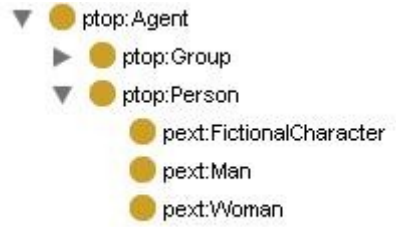

Figure 6: ptop:Person class hierarchy

Apart from the properties, inherited from the **ptop:Object** and **ptop:Agent** super-classes, **ptop:Person** has proprietary ones, as follows:

- **ptop:hasPosition** – this property relates a **ptop:Person** to a **ptop:JobPosition** (a sub-class of **ptop:Situation**: the situation of a person holding a position within an organization) – e.g. in the company he/she works for.
- **ptop:hasSocialPosition** – this property relates a **ptop:Person** to a **ptop:SocialPosition** (a sub-class of **ptop:Situation**: the situation of a person holding a social position within a society) – e.g. being Pope. Subproperties are: **pext:hasSocialPositionCleric**, **pext:hasSocialPositionCelebrity**, **pext:hasSocialPositionNobelty**, **pext:hasSocialPositionOutOfLaws**.
- **ptop:hasRelative** - this property relates a **ptop:Person** to another one, who he is a relative to and who is a relation to the former. This is a many-to-many, bi-directional relationship. This relation has a number of specializations, presented in Figure 7.
- **ptop:isBossOf** – this property relates a **ptop:Person** to another one, who he is an immediate boss or supervisor of. This is a many-to-many relationship, i.e. there can be more than one boss of a person, even co-temporally.
- **ptop:firstName**, **ptop:givenName**, and **ptop:lastName** – this datatype properties relates **ptop:Person-s** to their names.

- **pext:hasProfession** – this property relates a **ptop:Person** to a **pext:Profession** (a sub-class of the **pext:SocialAbstraction** class in the **ptop:Abstract** class hiearchy).
- **pext:birthDate** and **pext:deathDate** – this datatype properties relates **ptop:Person-s** to their birth date and the date of their death.
- **pext:birthPlace** and **pext:deathPlace** – this property relates a **ptop:Person** to a **ptop:Location** (a sub-class of the **ptop:Object** class) – the place where the person was born or where the person died.
- **pext:musicInstrument** – this property relates a **ptop:Person** to a **pext:MusicalInstrument** (a sub-class of the **ptop:Object** class) – e.g. Louis Armstrong played a trumpet.
- **pext:nationalityOf** – this property relates a **ptop:Person** to a **pext:Nationality** (a sub-class of the **ptop:Abstract** class).

The properties with prefix **pext:** are available only when PROTON Extend module is loaded.

Figure 7: Hierarchy of family relations

6.1.3 Group and Organization classes

ptop:Group is a group of agents, which is not necessarily organized in any way. This could be the group of people within a bus or the shareholders of a company. **ptop:Organization** is set as a sub-class of **ptop:Group** because of this very difference in the presence or the absence of organization of the agents in the group. **ptop:Organization** denotes a group, which is established in such a way that certain known relationships and obligations exist between the members, and/or between the organization and its members, and/or between the organization and `outsiders' (individuals or groups). **ptop:Organization** includes both informal and legally constituted organizations. Organizations can act as agents - to undertake projects, to enter into agreements, to own property, etc. Most organizations have names. Almost all have at least two members.

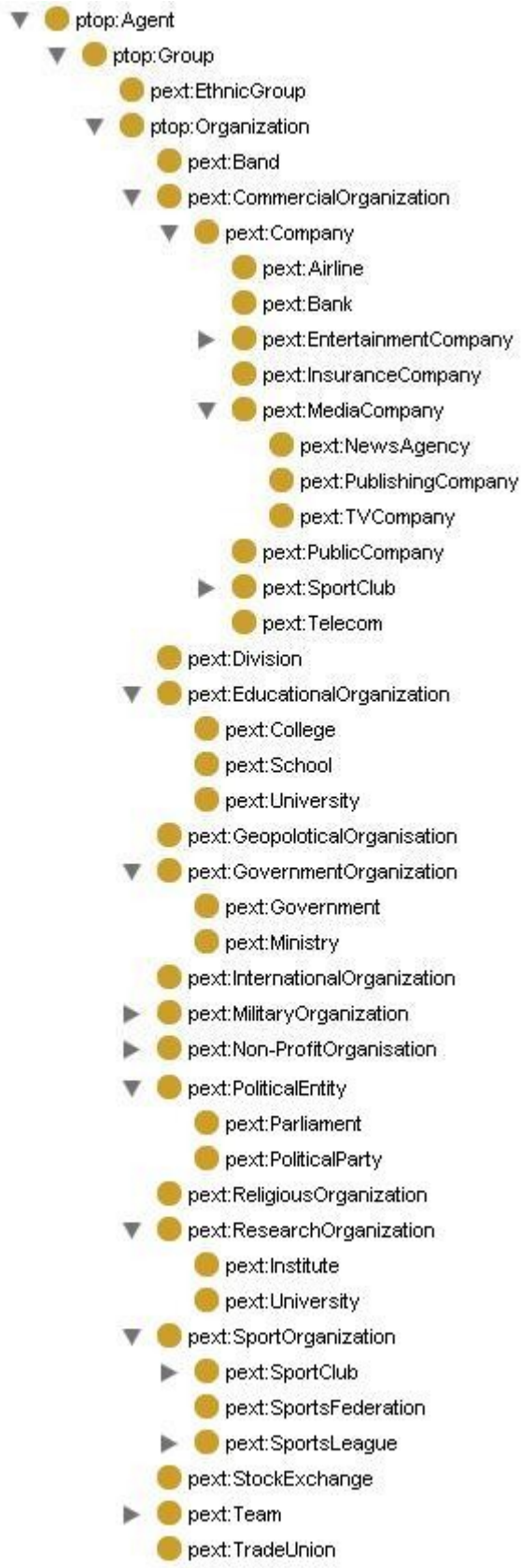

Figure 8: ptop:Group and **ptop:Organization class hierarchy**

Apart from the properties inherited from the **`ptop:Object`** super-classes, **`ptop:Group`** and **`ptop:Organization`** have proprietary ones, as follows:

- **`ptop:hasMember`** – this property relates a **`ptop:Group`** to **`ptop:Agent`**–s that are members of the group, e.g. ministers are members of the council of ministers.

- **ptop:establishedIn** – this property relates a **ptop:Organization** to a **ptop:Location** in which the organization is legally established, e.g. Ontotext is established in Bulgaria.
- **ptop:establishmentDate** – this data property relates a **ptop:Organization** to the date when it was established.
- **ptop:doingBusinessAs** – this data property relates a **ptop:Organization** to a name, used for marketing.
- **ptop:numberOfEmployees** – this data property relates a **ptop:Organization** to the number of its employees.
- **ptop:parentOrganizationOf** – this property relates a **ptop:Organization** to its dependent organization.
- **ptop:registeredIn** – this property relates a **ptop:Organization** to a **ptop:Location** in which the organization is registered.
- **ptop:subsidiaryOrganizationOf** – this property relates a subsidiary **ptop:Organization** to its main organization.
- **pext:productOf** – this property relates a **ptop:Organization** to its products (**pext:ProductModel**).
- **pext:industryOf** – this property relates a **ptop:Organization** to the industry sector in which the organization is active.
- **pext:organizationPosition** – this property relates a **ptop:Organization** to a position within the organization.

The properties with prefix **pext:** are available only when PROTON Extend module is loaded.

6.1.4 Location class

ptop:Location is a sub-class of **ptop:Object.** A location in PROTON is usually deemed a geographic location on the earth, however any sort of 3D regions also fit in this class. The **ptop:Location** class hiearchy of PROTON contains over 200 sub-classes, aligned with the GeoNames geographical database. For the sub-classes, the corresponding NIMA GNS designators (DSG) [18] are given.

[18] http://www.geonames.org/

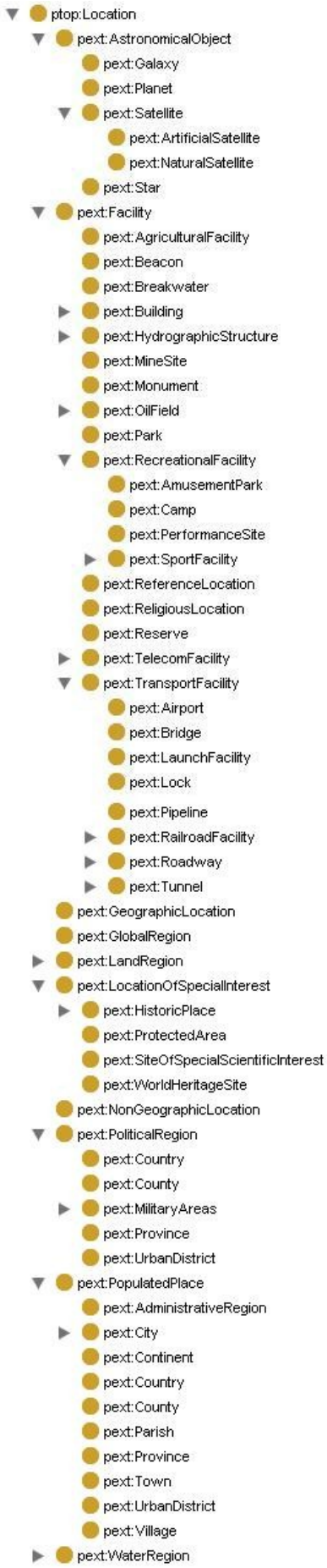

Figure 9: The Location class in both modules – Top and Extend

Because the Geographic features (Locations) form a large part of the entities of general importance, we developed the **ptop:Location** Top module class as a part of PROTON. In a way, it is a rather autonomous kind of a sub-ontology within the framework of PROTON. The purpose was to include the most important and frequently used types of Locations (which are specializations of **psys:Entity** through **ptop:Object**), including relations between them – such as **pext:hasCapital**, **ptop:subRegionOf** - a much more specific relation than the **ptop:partOf** one, inherited from the **psys:Entity** super-class), relations between **ptop:Location**-s and other **psys:Entity**-s (**Organization locatedIn Location**), and various attributes. The proprietary relations and attributes for the **Location** class are as follows:

- **ptop:nimaGNSDesignator** – the designator of the entity according to the NIMA GNS.
- **ptop:nimaGNSUniqueFeatureIndentifier** – the Unique Feature Identifier from the NIMA GNS. A number that uniquely identifies the location.
- **pext:hasUniversity** – a relation with the **University** sub-class of **Group**.
- **ptop:latitude** – in degrees, minutes, and seconds: no sign (+) = North; negative sign (-) = South.
- **ptop:longitude** – in degrees, minutes, and seconds: no sign (+) = East; negative sign (-) = West.
- **ptop:populationCount** – this data property relates a location with the number of inhabitants living in it.
- **ptop:subRegionOf** – the general part-of relation, which here designates a place between a whole and each of its parts. It has a number of specializations.

Also, among the properties that **ptop:Location** inherits from the **ptop:Object** super-class, the most important one is **ptop:locatedIn**. It is a transitive relation that is rather significant because it mainly ensures the relation between other **psys:Entity**-s and the **ptop:Location** class hiearchy. Further, the **ptop:subRegionOf** relation (described above) extends the "path" into a specialized 'part-of' type of a relation between different **ptop:Location**-s.

The **ptop:Location** entity denotes an area in 3D space, which includes geographic entities with physical boundaries, such as geographical areas and landmasses, bodies of water, geological formations and also politically defined areas (e.g. "U.S. Administered areas").

The classification hierarchy in the PROTON Extend module layer for **ptop:Location** (consisting of more than 300 sub-classes of **ptop:Location**) provides the following additional information:

- the exact type of a feature, e.g. to be able to recognize a geographic feature as an instance of **pext:CountryCapital** instead of just a **ptop:Location**.
- relations between a geographic feature and other entities (e.g. "Diego Garcia" is a **pext:MilitaryBase**, located somewhere in the Indian Ocean, and it is **ptop:subRegionOf** USA).
- the different names of a location ("Peking" and "Beijing" are two aliases for one location).
- the transitive **ptop:subRegionOf** relation allows one to search for Entities located in a continent (e.g. "Morgan Stanley" - **locatedIn** - "New York" – **subRegionOf** - "NY" - **subRegionOf** - "USA" – **subRegionOf** - "North America")

- **psys:trusted** vs. **psys:recognized** sources, linked by the **psys:generatedBy** property of a **ptop:Location** is an extra hint in disambiguation tasks. The class hierarchy is shown in Figure 9.

6.1.5 Statement class

ptop:Statement is a message that is stated or declared; a communication (oral or written) setting forth particulars or facts etc; "*according to his statement he was in London on that day*". Adapted from WordNet.

The **ptop:Statement** class has three proprietary properties, as follows:

- **ptop:statedBy**: this property best matches the **dc:creator** element (an entity primarily responsible for making the content of the resource; examples of a Creator include a person, an organization, or a service). Via the **ptop:statedBy** property, **ptop:InformationResource** can be linked to the **ptop:Agent** class.
- **ptop:validFrom, ptop:validUntil**: these properties define the range (period) of validity of the information resource. They align to a great extent to the **dc:value** element refinement, as defined in Dublin Core[19] (see subsection 6.1)

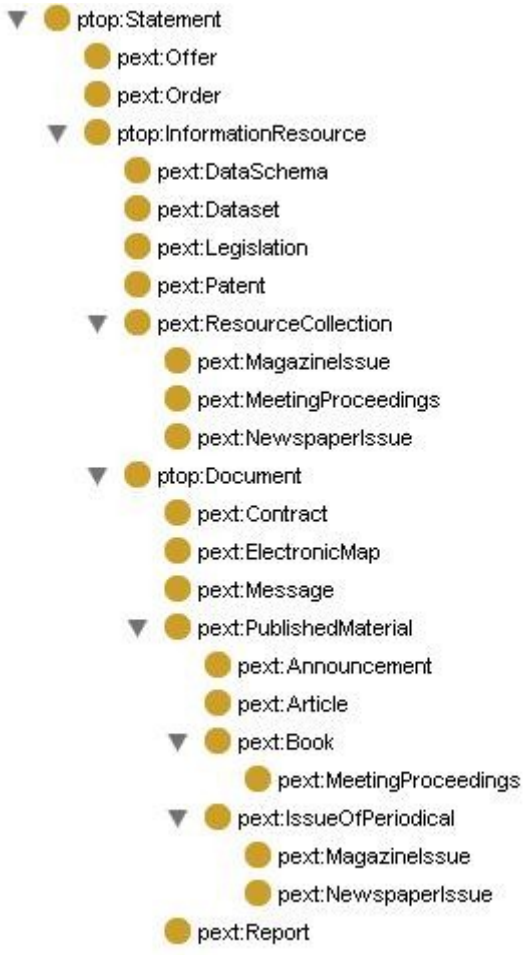

Figure 10: ptop:Statement, ptop:InformationResource, and ptop:Document class hierarchy

[19] dublincore.org

6.1.6 InformationResource and Document classes

ptop:InformationResource denotes an information resource with an identity, as defined in Dublin Core (DC2003ISO). It is a sub-class of **ptop:Statement**. Since **ptop:Statement** is described as "a message that is stated or declared; a communication (oral or written) setting forth particulars or facts etc.", **ptop:InformationResource** is considered any communication or message that is delivered or produced, taking into account the specific intention of its originator, and also the supposition (and anticipation) for a particular audience or counter-agent in the process of communication (i.e. passive or active feed-back). For instance, an **pext:Offer** differs from an **pext:Announcement** by the (non)obligatory aspect of an audience.

The **ptop:InformationResource** class hiearchy contains another class of the PROTON Top module - **ptop:Document**. This is the information content of any sort of a document; any tangible (material) aspects of a document are ignored; it is usually a document in free text with no formal structure or semantics. **ptop:Document** is a distinctive type of an information resource: what distinguishes the **ptop:Document** sub-class, as a specific type of an information resource, are the aspects of intention and "particularism" of a document, at least in the "general" connotations and understanding about "what a document is" and what cannot be described as a document (e.g. a database is not a document, it is rather a dataset). Descriptions of the proprietary properties and attributes of **ptop:Document** follow further in this section.

The **ptop:InformationResource** class hiearchy was designed with tight adherence to the Dublin Core (DC, section 6.1) metadata elements set. In this connection, it is worth mentioning that **ptop:InformationResource** is quite close in correspondence to what is considered **dc:metadata** in DC, of course, on the understanding that a high level of abstraction from the SEKT-specific meaning and usage of the term "metadata" is kept. The similarity is not in the senses of the two notions, but rather in the way they are described and defined "from the outside".

6.1.7 ProductModel class

ptop:ProductModel covers the general concept of a product model, says, Ford T. Apart from the properties inherited from **ptop:Object**, **ptop:ProductModel** has the proprietary one **ptop:producedBy** – a relation between the **ptop:ProductModel** and **ptop:Organization**/**ptop:Agent** that produces it. PROTON Extend module contains several sub-classes of product modules – car, airplane, media, etc.

6.1.8 Service class

ptop:Service denotes any sort of a service, ranging from scheduled flight or train service to weather forecast information/web service. Many services could be considered instances of **ptop:Agent**. Apart from the properties inherited from **ptop:Object**, **ptop:Service** has the proprietary one **ptop:operatedBy** - a relation between a **ptop:Service** and a **ptop:Agent** (usually an organization) that provides it.

6.1.9 CommercialOrganization class

pext:CommercialOrganization (a sub-class of **ptop:Organization** and a super-class of **pext:Company**) denotes an organization that buys or sells goods or services for a profit. Apart from the properties inherited from **ptop:Object**, **pext:CommercialOrganization** has several proprietary ones, as follows:

- **pext:fiscalNetIncome** - net income during the last fiscal year;
- **pext:fiscalSales** – sales during the last fiscal year;
- **pext:activeInSector** - denotes that an organization is active within a respective industry sector. This property relates **pext:ComercialOrganization** with

pext:IndustrySector (a sub-class of **pext:BusinessAbstraction**, sections 6.3.6 and 6.3.5);

- **pext:hasShareholder** - relates a particular organization to the agents that are members of this organization. This predicate indicates a 'generic' membership, although there may be specialized kinds of membership in one organization. Typically, membership eligibility is determined by the organization, and it is accepted with the agent's voluntary affiliation. In many cases, **Person**s who take **Position**s within an **Organization** are considered members of that organization, although this is not formally encoded here in any way.

6.2 Happening hierarchy

ptop:Happening is something that happens. It can be either dynamic – like in "drawing a circle" (the **ptop:Event** sub-class), or static – like in "being a president" or "sitting on a chair" (the **ptop:Situation** sub-class). In all cases, a happening has a certain temporal positioning (extent) – in the simplest case one, denoted by start and end points in time. Its proprietary relations and attributes are: **ptop:startTime** (the starting moment of a happening) and **ptop:endTime** (the end point of a happening).

Also, it is usually the case that some entities[20] take part in the happening – i.e. are involved in, or play a certain role in it. The general property relating a happening and a participant in it (participant in general sense) is **ptop:hasParticipatingEntity**. The entities could be active or passive participants, or part of context of the happening. There is a more specific property for relating of a happening and the agents that participate in it - **ptop: hasParticipant.** The roles played by the participants are instances of the class **ptop:Role** described below.

The **ptop:Happening** class is a super-class of the following classes: **ptop:Event** (see subsection 6.2.1), **ptop:Situation** (see subsection 6.2.2), **ptop:TimeInterval** (see subsection 6.2.3), **ptop:SocialPosition** (sub-class of **ptop:Situation**, subsection 6.2.4), **pext:Activity** (see subsection 6.2.3), **pext:RecurringEvent** (see subsection 6.2.3), Here is a list of direct sub-classes of **ptop:Happening** from PROTON Extend module: **pext:Activity** and **pext:RecurringEvent**.

[20] This is a quit unrestricted modelling; if we have to be more specific, only Objects and Abstract entities are expected to take part in happenings.

Figure 11: ptop:Happening class hierarchy in both modules – Top and Extend

6.2.1 Event class

ptop:Event denotes a dynamic happening, such as "running", or "a concert". **ptop:Event** is in opposition to **ptop:Situation** in terms of the dynamic vs. static nature of the **ptop:Happening**. **ptop:Event** inherits the properties of its super-class **ptop:Happening**. Its specializations include sub-classes like **pext:Accident**, **pext:Meeting** (see subsection 6.2.6), **pext:Project** (see subsection 6.2.7), **pext:SportEvent**, **pext:MilitaryConflict**, and **pext:ArtPerformance**.

6.2.2 Situation class

ptop:Situation denotes a static happening or situation, like "sitting on a chair" or "holding position". Typically, those are temporarily homogenuous, i.e. their nature is not expected to change within their duration. As a **ptop:Happening**, they use to be true for some periods of time and may or may not have a well-defined space extension. **ptop:Situation** is in opposition to **ptop:Event** in terms of the dynamic vs. static nature of the **ptop:Happening**. **ptop:Situation** inherits the properties of its super-class **ptop:Happening**. Its specializations include sub-classes like **ptop:SocialPosition** (see subsection 6.2.4) and **ptop:Role** (see subsection 6.2.5).

6.2.3 TimeInterval class, modelling of time

ptop:TimeInterval is a general time expression (TIMEEX) that refers to a particular period of time, an interval. Repeating periods (like *Spring* or *Christmas*) are not time intervals, while specific instances of theirs (like *the Spring of 1944*) are. In very special cases, a **ptop:TimeInterval** could collapse to a time point. However, in contrast to the **ptop:Abstract** time point (referring to some time during the day), it should be bound to a specific date, i.e. to represent a timestamp.

Figure 12: prop:TimeInterval class hierarchy

ptop:TimeInterval in PROTON is considered a specialization of **ptop:Happening**, because this is the place in PROTON for all entities that are largely defined through their temporal extent.

Abstract seasons, months, days of the week are represented as sub-classes of the **pext:TemporalAbstraction** in the **ptop:Abstract** class hiearchy of PROTON. Thus, the general notions of Tuesday and July appear as instances of classes in this hierarchy. It ought to be made clear that those do not have a temporal extent – they are just abstractions, symbols that can be used to refer to particular periods of time.

6.2.4 SocialPosition class and modelling

ptop:SocialPosition-s of people within a society are modelled as a special sort of a situation, i.e. a "static" happening. This matches their nature well, as seen in the following context: "*it happened that he was a CTO at XYZ Corp between 2001 and 2003.*" or "*he is a pope since 2005.*" Its proprietary relations and attributes are:

- **ptop:socialPositionHolder** – this property relates a position to the **ptop:Person** who holds it. There is an inverse relation to this one, named **ptop:hasSocialPosition**, defined for persons.
- **pext:hasSocialFunctionTitle** – this property relates a position to its title which is represented as an instance of a some sub-class of the class **pext:SocialFunction**.

Here is a list of its sub-classes: **ptop:JobPosition**, **pext:Cleric**, **pext:Celebrity**, **pext: Nobelty**, and **pext: OutOfLaws**.

The most developed class is **ptop:JobPosition**. Its proprietary relations and attributes are:

- **ptop:holder** – relates the position to the **ptop:Person** who holds it. There is an inverse relation to this one, named **hasPosition**, defined for persons.
- **ptop:withinOgranization** – the **ptop:Organization** where the position is situated.
- **ptop:heldFrom** and **ptop:heldTo** – two literal-ranged attributes, used to encode the start and end of a period, for which a particular person used to take a position, if known. These are specializations respectively of the **ptop:startTime** and **ptop:endTime** attributes of **ptop:Happening**.
- **pext:appointmentFor** – this property relates **ptop:JobPosition** to **pext:PositionOrganization**. **pext:PositionOrganization** is a situation in which there is a position within an organization regardless who holds it in a given period of time. **ptop:JobPosition** is related to a person who holds the position and the period in which it was held.
- **pext:hasTitle** – this property relates **ptop:JobPosition** to **pext:JobTitle**. **pext:JobTitle** is a social abstraction sub-class, which is the current way for classification of most job positions in PROTON. Many of the sub-classes of **pext:JobTitle** were reclassified as **pext:JobTitle** instances. They are **pext:BoardMember**, **pext:CEO**, **pext:Chairman**, **pext:Employee**, **pext:Executive**, **pext:Leader**, **pext:Manager**, **pext:MemberOfParliament**, **pext:Minister**, **pext:Premier**, and **pext:President**. This nomenclature can be extended depending on the application.

There is a more specialized support for official (e.g. government positions), through the **pext:OfficialPosition** sub-class with an **pext:officialPositionIn** property ranged **ptop:Location**. The latter provides a useful shortcut as seen on Figure 13. (the dashed arrows and nodes.)

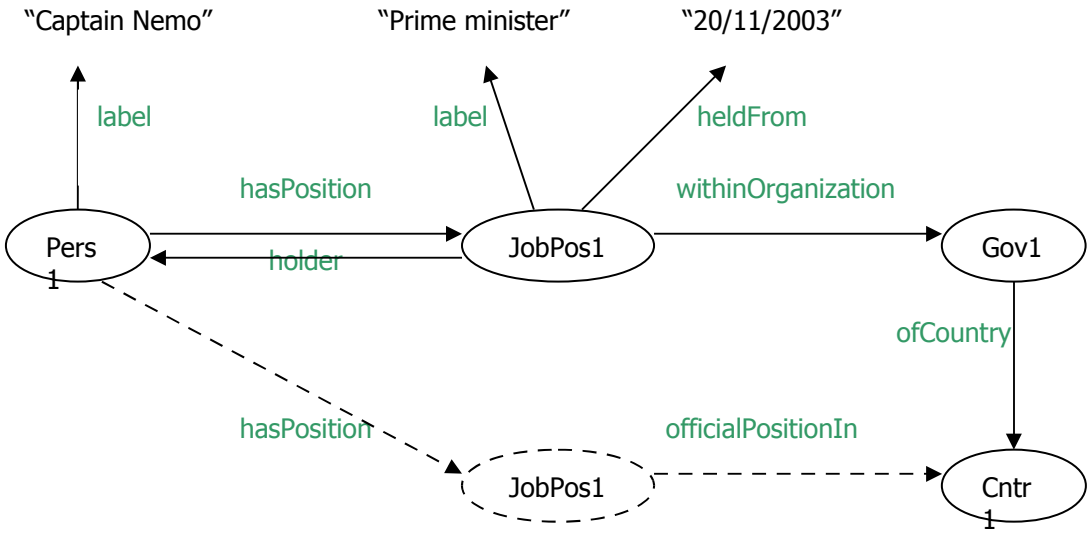

Figure 13: Modeling of (Official) JobPosition-s

This approach to modelling of **ptop:JobPosition** is adapted where it is necessary to the other sub-classes of **ptop:SocialPosition.**

6.2.5 Role class

A **ptop:Role** designates the role of an entity (usually an agent) within/for/during/effecting (intentionally or not) a particular happening. For instance, the role played by a coordinator of a project, or a defendant in a trial, or even an evidence in a trial (e.g. a material object that serves as an evidence, for example a knife a murder was committed with). Its proprietary relations and attributes are:

- **ptop:roleHolder** – relates the role to the **psys:Entity** that holds ("plays") it.
- **ptop:roleIn** – relates the role to the **ptop:Happening** that conditions and takes on the effect of the role.

As shown in the example in Figure 14., when sub-classes of **ptop:Role** are to be modelled (e.g. in a domain ontology), it should be taken into account that interdependencies between some roles are highly possible, so that transitive role-to-role relations might be necessary at some point.

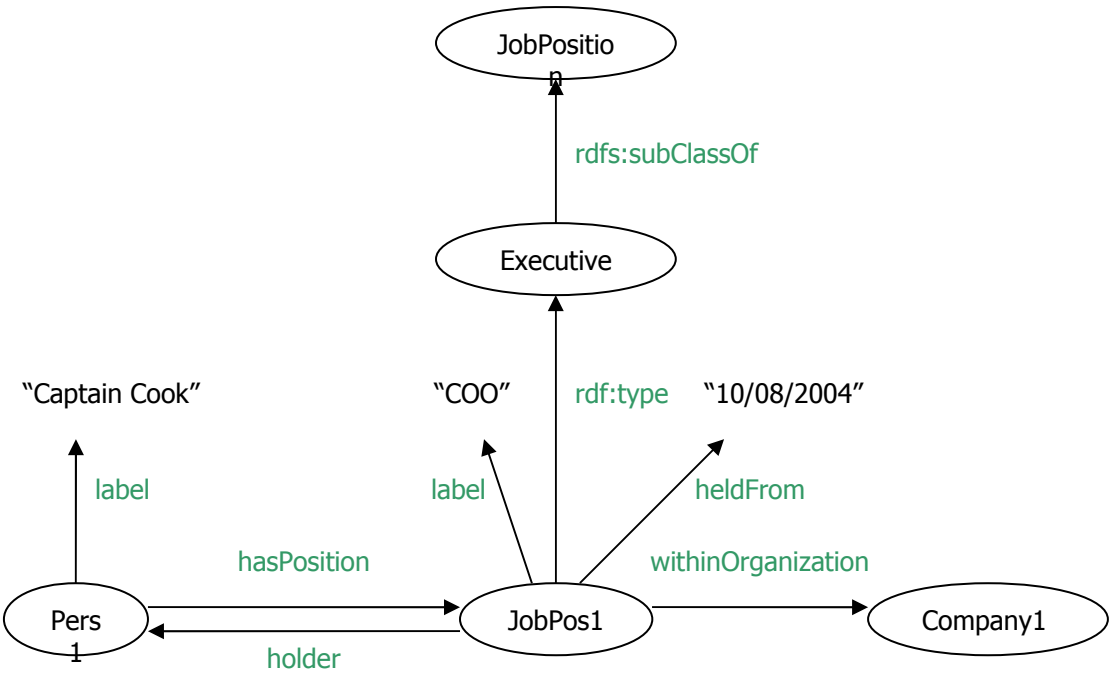

Figure 14: Modeling of the ptop:Role class

Moreover, by an axiom in PROTON, when the **ptop:roleHolder** of a role instance is an instance of a **ptop:Agent**, then this **ptop:Agent** is assumed to be **ptop:involvedIn** the respective **ptop:Happening**.

6.2.6 Meeting class

pext:Meeting (a sub-class of **ptop:Event** – subsection 6.2.1) denotes a formal or informal meeting, regarded as an event (i.e. it is specific in temporal and spatial aspects). It is worth noting that **pext:Meeting** here primarily implies the following senses of the word – i.e. a coming together of persons or things, or an assembly, gathering of people, especially to discuss or decide on matters. **pext:Meeting** has no properties of its own - it inherits the ones of its super-class.

6.2.7 Project class

pext:Project (a sub-class of **ptop:Event** – subsection 6.2.1) may denote a special unit of work, research, etc.; a proposal of something to be done, like a plan or a scheme; an organized undertaking; an extensive public undertaking, as in construction, conservation, etc. In any case, a **pext:Project** has a duration in time, i.e. it has its temporal connotations. **pext:Project** has no properties of its own - it inherits the ones of its super-class.

6.2.8 OfficialPosition class

pext:OfficialPosition (a sub-class of **ptop:JobPosition** - subsection 6.2.4) denotes a specific position that models the **Person->hasPosition->Location**, e.g. "president of USA", "mayor of NY". It is usually a shortcut to holding a position within the local government or other local administration, including military positions. Apart from the properties it inherits from its super-class, **ptop:JobPosition**, **pext:OfficialPosition** has the proprietary property **pext:officialPositionIn**, which relates it to the **ptop:Location** class (section 6.1.4).

6.2.9 Date class

pext:Date (a sub-class of **ptop:TimeInterval** – subsection 5.2.1) denotes a specific date, as 12th of April, 1961, as the time period (the 24 hours of the day). **pext:Date** inherits the properties of **ptop:TimeInterval**. Other sub-classes of **ptop:TimeInterval** are **pext:CalendarYear**, **pext:Month**, **pext:Quarter**, and **pext:Week**.

6.3 Abstract class hiearchy

ptop:Abstract denotes an abstraction: i.e. something, which neither happens nor exists, e.g. a number or a chemical compound. Those are usually some symbols invented to refer to general notions.

The **ptop:Abstract** class is a super-class of the following PROTON Top module classes: **ptop:Number** (see subsection 6.3.1), **ptop:ContactInformation** (see subsection 6.3.2), **ptop:Language** (see subsection 6.3.3), and **ptop:Topic** (see subsection 6.3.4). Here is a list of direct sub-classes of **ptop:Abstract** from PROTON Extend module: **pext:Award**, **pext:BusinessAbstraction**, **pext:Colour**, **pext:Genre**, **pext:Nationality**, **pext:NaturalPhenomenon**, **pext:SocialAbstraction**, and **pext:TemporalAbstraction**.

In some cases, when a class has specific, well-definable instances, which however may not be modelled as sub-classes, they are included in the knowledge base (e.g. some special instances of the **pext:Profession** class - artist, politician, religious person, scientist, sportsman – have been instaniated in the system part of the ontology).

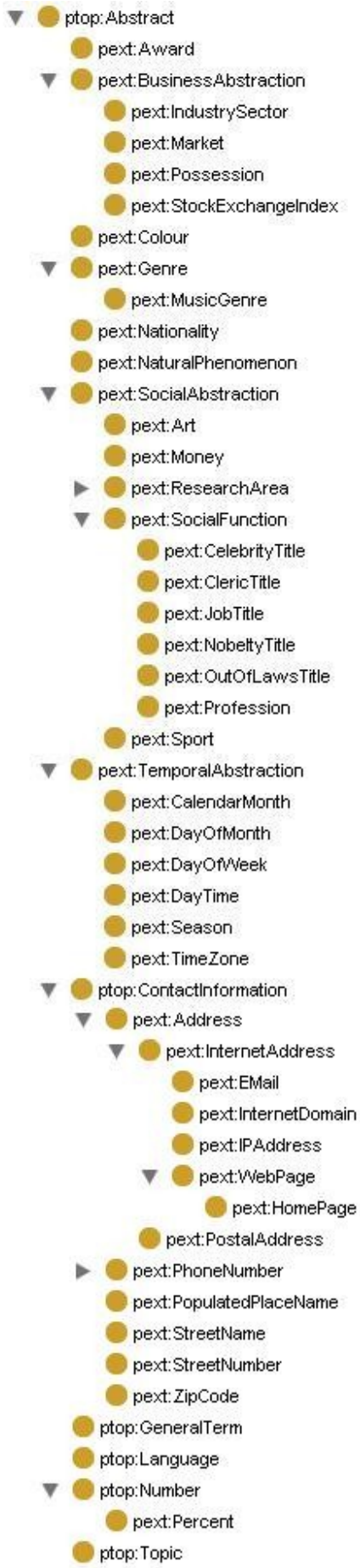

Figure 15: ptop:Abstract class hierarchy

6.3.1 Number class

ptop:Number is any given number, within the meaning that a number is: a concept of quantity derived from zero and units (“every number has a unique position in the sequence"); or a number is a numeral or string of numerals used for identification ("she refused to give them her Social Security number"); or a phone number, etc. A further specialization of **ptop:Number** is the **pext:Percent** sub-class, which denotes a specific percent value and thus it is a quite more specific kind of a number in terms of representation and meaning.

Figure 16: ptop:Number class

6.3.2 ContactInformation class

ptop:ContactInformation is any instance of a particular notation, used to make the contact with an individual or an organization possible. The class hierarchy of the different sorts of contact information is shown on the left-hand side of Figure 17. The hierarchy of properties is presented on its right-hand side.

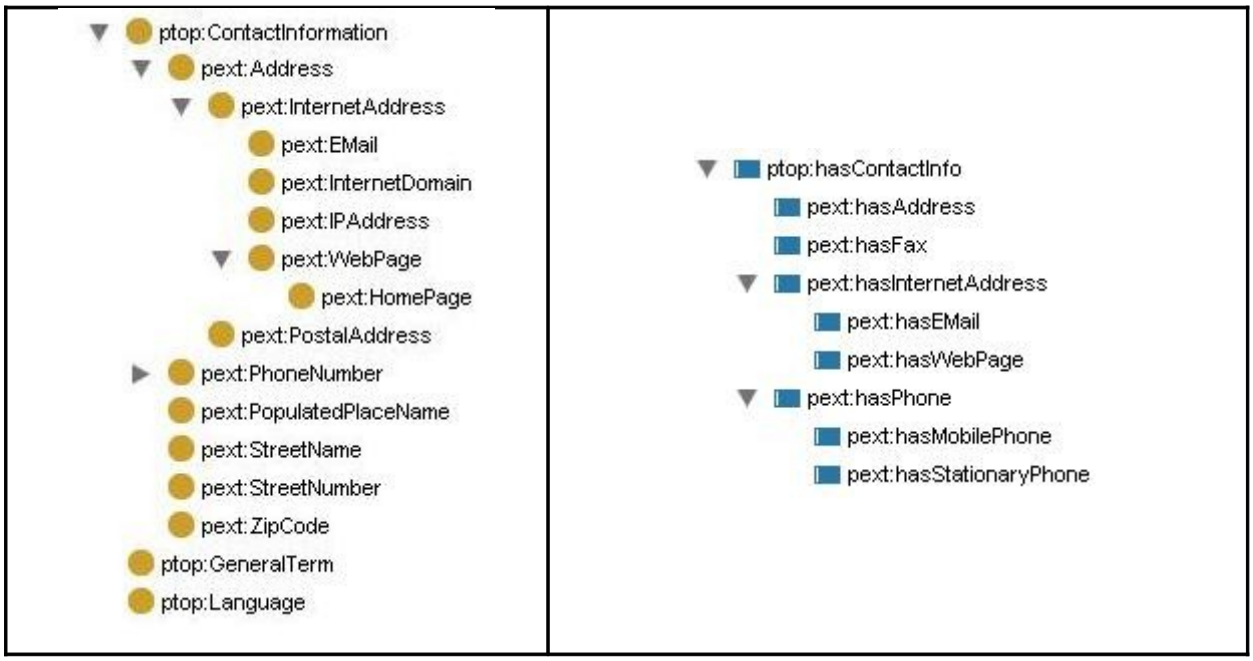

Figure 17: Modelling ptop:ContactInformation

6.3.3 Language class

ptop:Language denotes a spoken or written natural language – i.e. a human written or spoken language used by a community; opposed to e.g. a computer language (adapted from WordNet).

Along with the general properties that the **ptop:Language** class inherits from the **psys:Entity** super-class (defined in the PROTON System module), there is the following important property that concerns **ptop:Language** but is not proprietary to this class: **ptop:inLanguage** – it relates the **ptop:InformationResource** (see subsection 6.1.6) and **ptop:Language** top module classes. The domain of this property is the **ptop:InformationResource** class. It links an information resource to the language of the intellectual content of the resource.

6.3.4 Topic class

ptop:Topic is any sort of a topic or a theme, explicitly defined for classification purposes. As long as any other class or entity can play the role of a topic, the instances of this class are only those concepts that are defined to serve as topics. The topic class is the natural top-class for linkage of logically informal taxonomies. PROTON does not provide any **ptop:Topic** hierarchies as part of its Extend module layer. For more explanation see Deliverable SEKT D1.8.1.

6.3.5 BusinessAbstraction

pext:BusinessAbstraction (a sub-class of **ptop:Abstract**) denotes an abstract entity that is used in a business context – e.g. markets, industry sectors, brands, etc. Many products can also be seen as a business abstraction, but most of the products bear other important aspects, such as engineering and design.

6.3.6 Industry Sector Class and Modelling

Industry sectors are to be represented as instances of the **pext:IndustrySector** class, which is a sub-class of **pext:BusinessAbstraction**[21] (see subsection 6.3.5), which on its turn is a sub-class of **ptop:Abstract** (see subsection 6.3). Hierarchies of industry sectors can be specified with the **pext:subSectorOf** transitive relation (a specialization of **ptop:partOf**).

6.3.7 TemporalAbstraction

A **pext:TemporalAbstraction** (a sub-class of **Abstract**) denotes any sort of an abstraction that is used to refer to periods of time in general, i.e. periods of time, which are not specific or unique as a concrete date, for instance. Thus, the month of September is an instance of this class, while Sept 1989 is not (it is a specific **ptop:TimeInterval**, and thus not abstract).

Already hinted, there is a clear, though thin line of distinction between the two hierarchies in PROTON that deal with temporal aspects of world knowledge – **pext:TemporalAbstraction** and **ptop:TimeInterval** (see subsection 6.2.3). The key to this delineation lies in the specific or non-specific nature of the temporal notion classified.

6.3.8 GeneralTerm

A **ptop:GeneralTerm** (a sub-class of **ptop:Abstract**) is the "place" for the representation of a general concept with a well-defined (idiomatic) meaning, which can have a set of distinct lexical items (surface realizations) associated with it. Such examples are: F2F, I18N, P2P, B2B, VIP, ASAP, Semantic Web.

6.3.9 NaturalPhenomenon

pext:NaturalPhenomenon (a sub-class of **ptop:Abstract**) denotes natural phenomena such as a particular disease, the Gulfstream, or other similar natural abstractions.

Important: The particular events or objects that could instantiate an abstract natural phenomenon (i.e. a specific event of a sickness, caused by a disease) are not instances of this class.

6.3.10 SocialAbstraction

pext:SocialAbstraction (a sub-class of **ptop:Abstract**) denotes any sort of a general social phenomenon, such as particular sort of art or science.

---

[21] This is just a common super-class for any sort of an abstract business entity (such as Market). It has been introduced mostly for the sake of some better structuring of the taxonomy, in order to avoid an excessive number of direct sub-classes of protontop:Abstract. See section 6.3.5.