# Ontology Design Principles and Normalization Techniques in the Web

Xiaoshu Wang[1,a], Jonas S Almeida[2,b], and Arlindo L Oliveira[1,c]

[1]INESC-ID: Instituto de Engenharia de Sistemas e Computadores Investigação e Desenvolvimento em Lisboa, R. Alves Redol 9, 1000-029 Lisboa, Portugal
[2]Department of Biostatistics and Applied Mathematics, The University of Texas M.D. Anderson Cancer Center, 515 Holcombe Blvd, Box 0447, Houston, TX 77030
[a]xiao@kdbio.inesc-id.pt, [b]jalmeida@mdanderson.org, and [c]aml@inesc-id.pt

**Abstract.** The fundamental issue of knowledge sharing in the web is the sharability of the ontological constrains associated with the Uniform Resource Identifiers (URI). To maximize the expressiveness and robustness of an ontological system in the web, each ontology should be ideally designed for a confined conceptual domain and deployed with minimal dependencies upon others. Through a retrospective analysis of the existing design of BioPAX ontologies, we illustrated the often encountered problems in ontology design and deployment. We identified three design principles – minimal ontological commitment, granularity separation, and orthogonal domain – and two deployment techniques – intensionally normalized form (INF) and extensionally normalized form (ENF) – as the potential remedies for these issues.

**Keywords:** Semantic Web, Resource Description Framework (RDF), Ontology Web Language (OWL), Ontology, Uniform Resource Identifier (URI), BioPAX.

## 1. Introduction

As the word *ontology* is variously used, we should define our use first. *Ontology* is here defined to be the RDF graph retrieved from the Uniform Resource Identifier (URI) of an ontological term. Although the definition may seem overly restrictive, for it has limited both the ontology's formalism (to RDF) and its application (to the web), it is necessary to establish the context of this discussion: of all techniques related to ontology development, what is described in this article is applicable mostly, if not exclusively, to the ontologies deployed in the web.

As an RDF graph can be decomposed into a set of RDF triples (subject property object), an ontology is syntactically equivalent to a set of formal assertions, possibly made in a logic language, such as Ontology Web Language (OWL), and intended for modeling a particular aspect of reality. Common logics, however, concerns only the validity of inference among set of symbols but not the validity of what symbols represent. A logic language is, therefore, semantically neutral with regard to the knowledge of its modeled domain. To make itself useful in practice, each ontology must therefore commit its vocabularies to certain reality[1]. Take the following RDF statement as an example (See section 2 for the syntactical convention):

```
_:x biopax-1:NAME 'Myoglobin'.
```

In the absence of the ontological commitment of *biopax-1:NAME*, the above statement can be used by a logic language to model any relations between *_:x* and "Myoglobin", among which the full name, short name, synonym, or primary identifier of an external database, etc., are all valid choices. But with *biopax-1:NAME* being committed to "the preferred full name" as defined in its URI, the above statement should allude only to *_:x*'s naming relation to "Myoglobin".

Every ontological term, therefore, inherently carries two kinds of meanings (Fig. 1). The first kind is *extensional*. The extensional meaning is carried by the term itself because the meaning is implied in the term's referred extensional entity. The extensional meaning of *biopax-1:NAME*, for instance, is simply the "*full name*" of a subject entity. The second kind of meaning is *intensional*. Intensional meaning is commonly expressed as a logic theory, i.e., a set of logical constrains, of the term with respect to its relations to other terms in the underlying ontology. In the case of *biopax-1:NAME*, for example, its intensional meaning is the being of an *owl:DatatypeProperty* that can be used to associate a string-type data with one of the following entities: *entity*, *biosource, and datasource*. Using *biopax-1:NAME* to associate a *biopax:entity* with an integer value, for instance, would violate the intensional meaning of the *biopax-1:NAME*. Collectively, the extensional and intentional meanings of an ontological term build the term's conceptualization about an external reality[2].

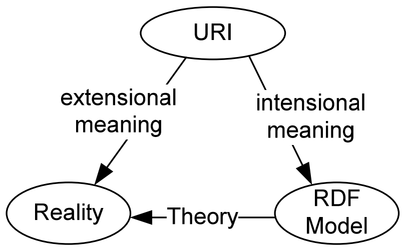

**Fig. 1.** The meaning of a URI.

Traditionally, an ontology is developed for the sharing of its intensional meanings; less care is taken to ensure the sharing of its extensional ones. This is understandable because, until the development of the web, the identity scope of ontological terms is typically restricted to individual files. When various conceptualizations are used across multiple ontologies, individual names must be either manually aligned or resort to external tool support like Ontolingua[3, 4]. The use of URI in the semantic web, however, should change this practice. Within an RDF document, the distinction between local and foreign identifiers becomes inconsequential. Hence, ontologies deployed in the web should be developed in a fashion that it can ensure the maximal sharing of not only its intensional meanings but also its extensional ones. The importance of sharing the latter must not be taken lightly. Because rarely will an extensional entity need – let along it hardly can – be completely identified by a set of logic constraints, every extensional entity is likely to be explained by various logical theories. Sharing ontology's extensional meanings, *viz.* URIs, enables those consistent theories to be freely combined in an engineer system. In addition, it allows those inconsistent theories to still coexist and to be communicated at human level. Hence, when an ontology is treated as a shared engineer artifact[4], sharing URIs helps to lower the cost of system integration; when it is used as a social agreement[5], it helps to retain the ontological commitment of a term as different theories about the term continue to evolve.

The purpose of this article, therefore, is to introduce a few design principles and engineer techniques that can be used in practice to improve the sharing and reuse of ontological URIs. To illustrate the benefit of these techniques, we used the ontologies developed at BioPAX (http://www.biopax.org) as our use cases. The reason that BioPAX ontologies are chosen is two-fold. First, BioPAX ontologies existed at three different levels, providing us a retrospective frame on the problem. A detailed analysis of these three BioPAX ontologies will allow us to glimpse at the complexity of ontology development, which, in turn, will help us to devise appropriate strategies to cope with them. Second, as the collaborative effort of numerous researchers from a number of institutes, the conceptualizations established in the BioPAX ontologies have become a valuable asset to the community. Hence, by analyzing the existing problems and finding solutions to avoid them in the future, we can further expand the applicability of this already valuable knowledge. A point that we want to stress here is that: it is easy – and understandably so – for people to take the provided problem analysis as a way to dismiss BioPAX ontologies. Thinking so, however, would misunderstand our intentions, which are to introduce ontology design principles and deployment techniques. We could easily build a set of trivial and imaginary ontologies to discuss the problem, but doing so would lessen the importance and urgency of these issues. Analyzing BioPAX, therefore, is aimed at offering constructive advices, as opposed to destructive criticisms, to the ongoing BioPAX development so that the resulting products could be more easily, efficiently, and broadly shared in the web.

## 2. Materials and Conventions

The three BioPAX ontologies used in this article are: Level-1 version 1.4, Level-2 version 1.0, and Level-3 version 0.9. Because of the different namespaces of the three levels of BioPAX ontologies, conceptualizations are compared in reference to the simple names of each URIs. In other words, if two BioPAX URIs have the same simple name, they will be assumed to denote the same extensional entity.

Qualified names as defined in [6] are used to shorten the URI notation. The URIs of all namespace prefixes used in this article except "biopax" are listed in Table 1. The prefix *biopax* is used to refer to a BioPAX concept in a general sense. It is worth noting that the namespace URI of the level-3 BioPAX ontology listed in Table 1 is not the official URI. Level-3 ontology was released informally through the BioPAX discussion mailing list. As the mailing list requires a registered account to access, we provided a copy at our own domain for reader's convenience.

All sample RDF statements will be written according to the syntax of Notation-3[7] with the namespace prefixes defined in Table 1. All RDF diagrams, such as Fig.

2, 3, and 5, are drawn with the graphical notation syntax of DLG$^2$ as defined in [8] as well as one of its extension defined in [9].

**Table 1.** Namespace Prefixes

| Prefix | URI |
|---|---|
| biopax-1 | http://www.biopax.org/release/biopax-level1.owl# |
| biopax-2 | http://www.biopax.org/release/biopax-level2.owl# |
| biopax-3 | http://dfdf.inesc-id.pt/ont/biopax-3# |
| owl | http://www.w3.org/2002/07/owl# |
| dc | http://purl.org/dc/elements/1.1/ |
| o3 | http://dfdf.inesc-id.pt/ont/o3# |
| [other] | http://dfdf.inesc-id.pt/ex/biopax/[other]# |

## 3. Analysis of BioPAX ontology

BioPAX ontologies gradually evolved over time. Level-1 ontology defined the usage of 76 terms, with additional 34 and 54 introduced at level-2 and 3, respectively. Each of the three ontologies, however, is governed by a unique namespace, making the data instances described at one level of ontology unable to interoperate with those at another. To make the problem easily understandable and presentable, let's take *biopax:NAME* as an example. Consider the following two statements.

```
_:x biopax-1:NAME 'Myoglobin'.
_:x biopax-2:NAME 'Myoglobin'.
```

From a machine's standpoint, the above two statements, in fact, entail two entirely different theories about the resource "_:x". To promote interoperability, a newly developed ontology, such as BioPAX level-2/3, should in principle reuse as much the conceptualizations defined in the existing ones, such as BioPAX level-1, as possible. Obviously, the problem can always be solved with a post-integration approach. For instance, the two versions of biopax:NAME can be easily aligned the following assertion.

```
biopax-1:NAME owl:sameAs biopax-2:NAME.
```

This integration approach, however, suffers two drawbacks. First, it is potentially expensive because, for every *k* URIs of the same conceptualization, *k(k-1)/2* statements must be made. As careful ontology engineering can easily avoid the problem, the integration approach should be taken as the last resort to integrate legacy semantic objects, such as those deployed in relational databases[10]. Second, the integration approach may not be possible when two conceptualizations logically contradict each other (see section 3.1.2).

The seemingly trivial problem encountered by the *NAME* conceptualization, in fact, elucidates a fundamental issue with regard to the knowledge engineering in the web. That is: how can the URI of an ontological term be maximally shared within the web? As ontological URIs denote conceptualizations, which can only be shared if they are logically *consistent* with each other, we should first investigate ontology's compatibility issue before evaluating the sharing capability of BioPAX ontologies. Strictly speaking, “consistency” should be used in place of “compatibility” to describe ontology’s sharing capability. However, as OWL has defined two properties – *owl:incompatibleWith* and *owl:backwardCompatibleWith* – to describe ontologies’ inter-relationship, it is in the best interest of the web community as a whole for us to align our terminologies with OWL.

## 3.1 Ontology Compatibility

Traditionally in computer science, product compatibility is defined with regard to their functional interface. A new version of software, for instance, is considered backward compatible if it can take the place of an older one in terms of fulfilling the existing functionalities. Ontology development is, however, different: it concerns data rather than functionalities; and it aims at sharing and reuse rather than replace. A new ontology is backward compatible if all data instances described in the new ontology are also valid data instances of existing ones.

Between ontology’s two kinds of meanings, the extensional meanings should not incur much, if any, ontology’s incompatibility. Extensional meaning is consumed by humans and their compatibilities are maintained through social agreement, which should in principle be very stable. The cause of ontology’s compatibility, therefore, should mainly come from the specification of its intensional meanings. As the intensional meanings of ontological terms are typically defined in logic languages, one ontological term is compatible with the other only if they are logically consistent. In the subsequent sections, we used a few examples from BioPAX ontologies to discuss the issue.

### 3.1.1 Compatible Use Case

Consider the definitions of *conversion* class in BioPAX ontologies. At level-1, *conversion* subclasses directly from *interaction* (Fig. 2a). But, at level-2, it does so indirectly from *physicalInteraction* – a class that was newly introduced into BioPAX at level-2 (See Fig. 2b). Because all instances of *biopax-2:conversion* would be valid instances of *biopax-1:conversion*, the level-2’s definition of *conversion* is compatible with that of the level-1.

The converse, however, is not true because a *biopax-1:conversion* is not

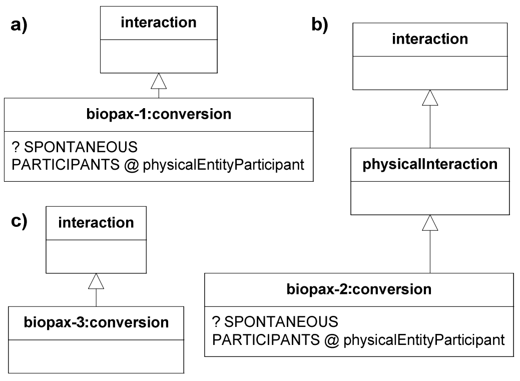

**Fig. 2.** BioPAX’s conversion class. (a) Level-1 definition (b) Level-2 definition (c) Level-3 definition.

necessarily a *physicalInteraction*, and therefore, a *biopax-2:conversion*. By the same reasoning, the *conversion* defined at level-3 (Fig. 2c) is not compatible with either *biopax-1:conversion* or *biopax-2:conversion*. The removal of the property constrains on SPONTANEOUS and PARTICIPANTS makes *biopax-3:conversion* a more subsuming class than its counterparts in level-1/2. For example, a *conversion* instance with two SPONTANEOUS properties would be a valid *conversion* at level-3 but an invalid one at level-1/2.

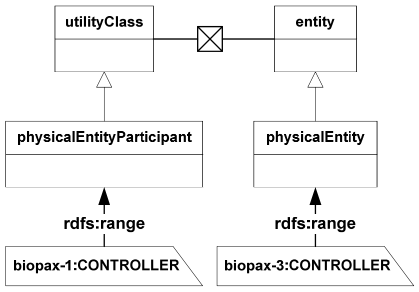

**Fig. 3.** CONTROLLER definitions in BioPAX.

Strictly speaking, an ontology term is either compatible with another one or it is not. But as the purpose of this article is to gain an insight into ontology development through the analysis of BioPAX ontologies, we designated the case similar to *biopax-3:conversion* as a *reverse compatible* case, in the sense that the compatibility is achieved in a reversed direction with reference to the progression of ontology development. The word "incompatibility", therefore, is reserved for those ontological terms whose conceptualizations that are not compatible in either direction.

### 3.1.2 Incompatible Use Case

The conceptualization of *CONTROLLER* property is a typical incompatible use case. At all three levels of BioPAX ontologies, *CONTROLLER*'s domain is defined to be *control*, but its range was changed at level-3 from *physicalEntityParticipant* to *physicalEntity* (Fig. 3). Because the latter two classes are respectively extended from two disjoint classes – *utilityClass* and *entity, biopax-3:CONTROLLER* is incompatible with either *biopax-1* or *biopax-2:CONTROLLER*.

It is worth noting that the concept of *physicalEntityParticipant* is left undefined at level-3. Owing to the informal status of the level-3 ontology that was used in this analysis, we are unsure whether the change is made by intention or by accident. In either case, nevertheless, the rationale behind the use case is valid because, in science, incompatible or contradicting theories about the same reality often coexist or emerge over time.

A milder case of incompatibility can be seen from the definition of *FEATURE-TYPE* property. At level-2, the property's domain is set to be *sequenceFeature*, but at level-3 it is changed to be *entityFeature*. Both *sequenceFeature* and *entityFeature* subclass from *utilityClass*, but their inter-relationship has yet to be defined in BioPAX. Logically, the two versions of *FEATURE-TYPE* are not inconsistent, but conceptually they reflect two different modeling approaches to the same entity. Sharing biopax-2:*FEATURE-TYPE* with biopax-3: *FEATURE-TYPE* will not improve data's interoperability because the same set of data instances would be interpreted by two completely different set of modeling primitives. In this article, this kind of definition was treated as an incompatible case because neither definition subsumes the other.

For the same reason, changing the property type from *owl:DatatypeProperty* to *owl:ObjectProperty* or *vice versa* is also considered an incompatible change, albeit its only consequence is the increase of inference complexity. A case in point is *biopax:KEQ*, which is an *owl:DatatypeProperty* at level-1 but an *owl:ObjectProperty* at level-2/3.

## 3.2 Compatibility of BioPAX Ontologies

Using the above defined criteria, we analyzed the inter-compatibility of ontological terms among three levels of BioPAX ontologies. The result is shown in Fig. 4. A detailed term-by-term analysis is provided at [11].

To simplify the analyzing process, two contributing factors – disjoint statement and term dependency – were disregarded because a full account of these factors would significantly increase the workload of the analysis without necessarily changing the delivered message.

The disjoint statement was disregarded due to its abundant use in BioPAX ontologies. A simple count of level-3 ontology, for example, revealed a total of 357 *owl:disjointWith* statements. Such abundance, together with the varying class hierarchy at different levels of BioPAX ontology, made the analysis of owl:disjoint statements quite a time consuming process. To reduce the workload, all disjoint statements were therefore excluded from the evaluation. Nevertheless, changing *owl:disjointWith* statement will have an effect on a term's compatibility. Adding a disjoint statement will make a class more logically restrictive, and removing one will make it more logically general, than its original form. A case in point is the *biopax-3:pathway*, which has an extra disjointed class – *referenceEntity* – compared with its counterparts at level-1/2.

The term's inter-dependency was disregarded owing to the difficulties of evaluation. Take the *modulation* class as an example. At all three levels of BioPAX ontologies, *modulation* is defined to be a subclass of *control* with

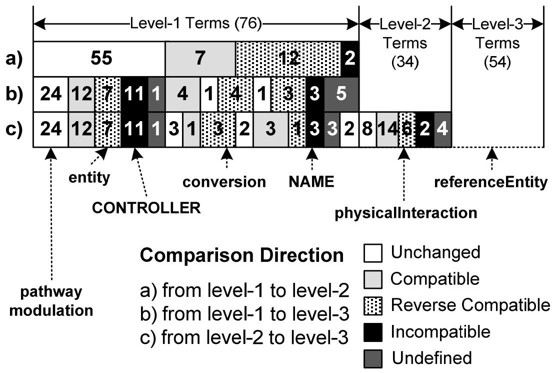

**Fig. 4.** Compatibility Analysis of BioPAX Ontologies. The compatibilities of the same term among three levels of ontologies are vertically aligned. The few example cases discussed in the text are illustrated. The numerical value inside each box shows the number of BioPAX terms with the similar compatibilities, which type is indicated by the boxes' filled pattern.

no more than one *CONTROLLER*, and with all CONTROLLED property coming from *catalysis*. But as the CONTROLLER's definition at level-3 logically contradicts those at level-1/2 (see last section), the *modulation's* dependency on the CONTROLLER property would make their definitions contradicting as well. In a logic system, a single contradiction would invalidate the entire theory. Even if we do not take this position during the evaluation, the interdependency of ontological terms will eventually lead us to the same conclusion. As seen in Fig. 3, the CONTROLLER's definition is ultimately related to *entity* and *utilityClass* – the very two top classes of BioPAX ontologies. Counting dependencies of BioPAX term will make *entity* and *utilityClass*, and therefore the entire ontology, a contradiction. In short, taking the full account of term dependency would prevent us from making more detailed and meaningful analysis. In the provided analysis, the compatibility was therefore evaluated solely based on their syntactic definitions; all dependency incurred changes were disregarded.

## 4. Methods to Improve Ontology's Sharing

The current practice of BioPAX ontology development is not optimal in terms of sharing and reusing existing conceptualizations. First, at each level of BioPAX ontologies, the same concept is renamed under a different namespace, making the sharing of their extensional meaning difficult. Second, the ontologies are presented in a monolithic fashion, making the sharing of their intensional meaning difficult as well.

Ideally, an ontology should be developed in an incremental fashion. Useful and relevant conceptualizations developed previously should be imported into the new ontology, where they can be logically refined with newly introduced conceptualizations. Take the conceptualization of *pathway* and *modulation* as an example. Their definitions (sans disjoint statements) remain unchanged throughout all three levels of BioPAX ontologies. In principle, both *pathway* and *modulation* need only to be defined once, e.g., at level-1, and then reused, e.g., at level-2/3, *via* a simple *owl:imports* statement. The same importing approach can also be applied to the compatible terms, such as *conversion*, whose level-1 definition can be imported into level-2, where additional subclass statements can be made to realign the class hierarchy in reference to the newly introduced class – *physicalInteraction*. Such an incremental approach allows both the URI and the logic definitions of a conceptualization be shared. Interoperability will be improved as data instances developed against the same ontology can be unambiguously understood on the same ground without additional engineering effort. Furthermore, it would also reduce the cost of ontology development and maintenance because the same statements no longer need to be redundantly defined.

But if ontologies are developed in a monolithic fashion, the incremental approach is unlikely to take place. Consider the development of BioPAX ontologies. Conceptualizations that would undergo different compatibility changes are physically bound in a single document (Fig. 4). The conceptualization of *modulation*, *pathway*, and *conversion* etc., for instance, were bundled up with that of *NAME* and

*CONTROLLER*. Because unlike *conversion*, which underwent a compatible change at level-2, the *biopax:NAME* made a reverse compatible change at level-2, suggesting that, if the conceptualization is to be shared, the import must be made from level-2 to level-1, but not *vice versa*. The *CONTROLLER*, on the other hand, took an incompatible change at level-3, suggesting that neither definition be imported into the other.

Since RDF does not yet support ontology modulization techniques, such as named graph[12] or C-OWL[13], statements in an RDF document cannot be selectively imported into another. To improve the sharing and reuse of existing conceptualizations, ontologies must, therefore, be carefully designed and deployed. First, large, monolithic ontologies should be broken up into sets of small and modular ones. Second, ontology's inter-dependency must be carefully structured in a manner that it avoids tight coupling. In the subsequent sections, we will introduce a few design principles and engineering techniques that may help in this regard.

## 4.1 Ontology Design Principles

### 4.1.1 Minimal Ontological Commitment

The principle of minimal ontological commitment (PMOC) was proposed by Gruber in [14]. Here we quote,

*An ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiate the ontology as needed. Since ontological commitment is based on consistent use of vocabulary, ontological commitment can be minimized by specifying the weakest theory (allowing the most models) and defining only those terms that are essential to the communication of knowledge consistent with that theory.*

Gruber proposed PMOC along with five other sound ontology design principles in the same article. Only PMOC is chosen here because it is the most pertinent to the ontology's sharing and reuse. Principles of similar essence have in fact been echoed in other research areas. For instance, *the principle of least effort* has been used to theorize the user behavior during information search, and *the rule of least power* has been proposed by W3C as the guidance for language design[15].

A case in point is BioPAX's definition of *NAME*. At level-1, the domain of *NAME* is defined to be the union of *entity*, *bioSource*, and *dataSource*. At level-2, a new class – *sequenceFeature* – is added to the domain, making *biopax-2:NAME* a reverse compatible definition of *biopax-1:NAME*. At level-3, however, the *NAME*'s domain is redefined to be the union of *bioSource, pathway, physicalEntity, and referenceEntity*, making it incompatible with either level-1 or 2's definition. However, as things that can have a *full name* – the NAME's extensional meaning – are not restricted to those entities defined in BioPAX, the conceptualizations of *NAME* have apparently over-committed its intended use. To follow the advice of PMOC, the domain of *NAME* should not be constrained at all. At most, if so desired, it can be constrained to the union of *entity* and *utilityClass* – the two top classes of BioPAX ontologies. Such a design would stabilize the *NAME*'s conceptualization, which, in turn, would allow the definition be easily shared by all three levels of BioPAX ontologies.

### 4.1.2 Granularity Separation

The PMOC, however, should not be taken in a rigid and extreme sense because, otherwise, there should be only top ontologies like Suggested Upper Merged Ontology (SUMO)[16] but nothing else. Most ontologies are developed to carry a specific application task, which should be used as the context for PMOC to be meaningfully applied. Take the conceptualizations of *conversion* as an example (See Fig. 2). Although only the level-3's definition satisfies PMOC, it is hard to fault the design of level-1 and 2's because the coarse semantic granularity of level-3's definition may not suit the need of the targeted application.

There is in general a tradeoff between an ontology's expressiveness and its shareability. A coarse grained ontology carrying only a few logical constraints is easier to be shared but less powerful to constrain an application's behavior. A fine grained ontology bearing a rich set of axioms, on the other hand, is more logically clear in directing specific application behaviors but less convenient and more expensive to be integrated.

Obviously, there should be all kinds of ontologies conceptualized at every grain of granularity. But the point that we want to stress here is: ontologies of different semantic granularities should be separately developed and deployed. This is, as we shall name it, *the principle of granularity separation (PGS)*. For instance, had the development of level-1 BioPAX followed the PGS, the general form of *conversion* definition as that of level-3 would have been separately developed at level-1, preventing it from confounding the sharing of other conceptualizations at level-3.

The PGS further implies that a fine grained ontology should be developed within the framework of a coarse grained top ontology. Starting from a top ontology is, in fact, almost always a good strategy [2, 17]. A top ontology usually carries less logical constraints; its conceptualizations are more general and stable, which make them easier to be shared as consensus among large communities of users. Furthermore, a top ontology usually contains the general structural information, which can help users to systematically identify and partition the targeted knowledge domain, against which detailed application ontologies can be developed and integrated.

### 4.1.3 Orthogonal Domain

Though not explicitly defined in the ontology field, orthogonal separation has been a well received concept in computer science. In both software[18] and database design[19], orthogonality principle has been used as a key guideline to improve system's expressive power and reusability. The very first architecture principle of the Web is, in fact, *the principle of orthogonal specifications* because it helps to increase the flexibility and robustness of the Web[20]. Here we propose *the principle of orthogonal domain (POD)* with regard to ontology development.

*Semantically orthogonal conceptualizations should be defined in separate ontologies.*

Consider, for example, the relationship between *entity* and *name*. A biological pathway entity is a *biopax:entity* regardless if it has a name or not. Conversely, whether an entity has a *name* or not bears no relations to its nature as a *biopax:entity* or not. Of course, developing the conceptualization of *NAME* in BioPAX can hardly be blamed because to label a *biopax:entity* does require a naming concept. But, deploying the conceptualizations of *NAME* and *entity* in the same ontology should be criticized because it hinders, if not prohibits, alternative conceptualizations from being applied. For instance, the conceptualization of *NAME* is, in fact, identical to that of *dc:title* defined in Dublin Core Metadata Initiative (DCMI)[21]. As the latter is a more thoroughly developed and shared community standard, *biopax:NAME* should eventually be replaced by *dc:title* to improve the overall data interoperability of BioPAX data. But, the physical binding of *entity* and *NAME* in the same document makes the replacement difficult to take place.

To follow the POD, the conceptualizations of *entity* and *NAME* should be separately specified in two different ontologies. There are two benefits of such a separation. First, separation reduces the sharing cost of each term, which in turn increases their shareabilities. In its current deployment form, for instance, *biopax:NAME* is unlikely to be used by someone, say a physicist, who has no interest in biology – not just because the *NAME*'s domain is so restricted, but more so because importing the *NAME*'s conceptualization would also import a few hundreds of assertions that are completely irrelevant to the physicist's tasks. Between increasing data interoperability and reducing computation cost, the physicist may be forced to choose the latter because a few importing of concepts similarly deployed as *biopax:NAME* would easily render his application into an unmanageable state. But if the conceptualization of *NAME* is separately deployed, the physicist would be more willingly to use it because doing so not only improves his data's interoperability but also reduces his development and maintenance cost.

Second, domain separation allows different conceptualizations to evolve independently without interfering with each other. In the field of knowledge engineering, there is an often encountered problem – the interaction problem, where "representing knowledge for the purpose of solving some problem is strongly affected by the nature of the problem and the inference strategy to be applied to the problem[22]". During the design of Chemical Markup Language (CML)[23], for instance, its designers have found "that many components with a 'chemical content' did not require chemical concepts for their implementation"[24]. Since one cannot, especially at the beginning of the Semantic Web, expect the presence of all kinds of ontologies available to his use, one must develop ontologies in both his familiar and his unfamiliar areas. Domain separation allows domain experts to divide their labors according to their area of expertise. A full scaled ontology can be developed in their familiar domains and makeshift solutions can be used to handle the areas that are out of their elements. More importantly, all these can be done without worrying that the latter's informality and less popularity may prevent the sharing and reuse of the former. In the next section, we introduce how such separation should be engineered in the semantic web.

## 4.2 Ontology Engineering

As conceptualizations are encoded and delivered in engineer artifacts, conceptual separations can only be realized if ontologies are engineered to be physically independent from each other. Consider the deployment of *entity* and *NAME* shown in Fig. 5a. At first glance, the two conceptualizations appeared to be separated because they are specified under two different namespaces "http://dfdf.inesc-id.pt/ex/biopax/a" and "http://dfdf.inesc-id.pt/ex/biopax/n", respectively. But a careful look will reveal that the separation is only half complete because *a:entity* is still tightly coupled with *n:NAME*, albeit not *vice versa*. A complete separation of the two conceptualizations should be deployed as shown in Fig. 5b, where *entity* and *NAME* are respectively specified at ontology "http://dfdf.inesc-id.pt/ex/biopax/*b*" and "http://dfdf.inesc-id.pt/ex/biopax/*n*". The two independent conceptualizations are collectively used at ontology "http://dfdf.inesc-id.pt/ex/biopax/*b1*".

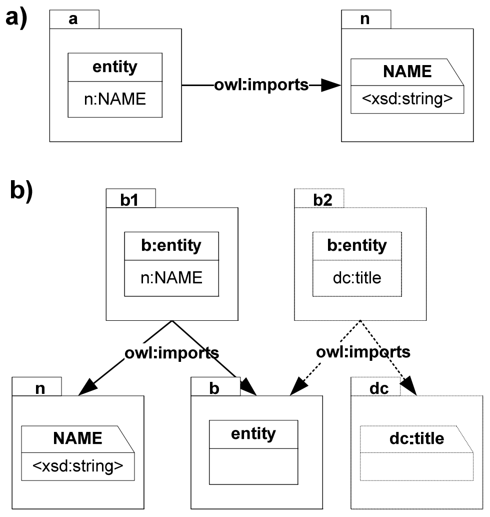

**Fig. 5.** Ontology Deployment Strategy. (a) and (b) shows two different deployment strategies. The dotted shapes show the potential changes in the future.

The advantage of a complete separation is that concepts defined in one ontology can be used independently of those in the other. This independence firstly reduces the sharing cost and, therefore, increases shareability. Secondly, it allows existing conceptualization to gracefully evolve over time without tampering the compatibility of the others. For instance, the *entity*'s conceptualizations in both "http://dfdf.inesc-id.pt/ex/biopax/*b1*" and "http://dfdf.inesc-id.pt/ex/biopax/*b2*" evolve from *b:entity*. But in ontology *b2*, *dc:title* is used in place of the *n:NAME* in '*b1*'. Owing to the complete independence between ontologies *b* and *n*, *b2* can coexist with *b1* so that the migration from one to another can be carried out gracefully. If, on the other hand, the *entity* is deployed as shown in Fig. 5a, the independent evolution of *entity* becomes impossible.

### 4.2.1 Ontology Classification

To facilitate the discussion on ontology's engineer issue, we defined an Ontology of ontologies (O3) [25] that classifies ontologies along few semantic dimensions. First, in O3, we made a distinction between ontologies that define a logic language, such as RDFS and OWL etc., from the rest, such as BioPAX and DCMI[21]. The former is defined as instance of *o3:Logic* and the latter of *o3:DomainOntology*. Second, we classified ontologies by their physical dependencies. If an ontology's conceptualizations are independent of the others, i.e., if the ontology does not import another *o3:DomainOntology*, it is defined to be an *o3:Local*. Otherwise, it is an *o3:Complex*. Third, we categorize ontologies according to the kinds of meanings that they carry. An ontology carrying only the extensional meanings of its terms is an *o3:Vocabulary*; one carrying only the intensional meanings is an *o3:Theory*; otherwise, it is an *o3:ConcreteOntology*. Different combinations of these basic ontology classes can lead to the definitions of *o3:LocalOntology*, *o3:ComplexOntology*, etc. An important class worthy of special mention is *o3:Profile*. An *o3:Profile* is an *o3:ComplexTheory* with characteristic RDF statements: all *o3:Profile*'s statements are made of concepts initially deployed elsewhere, meaning that the sole functionality of an *o3:Profile* is to make joint use of independent conceptualizations. Fig. 6 is a Venn diagram of the relationships among several ontology classes relevant to this discussion. More detailed information of those classes can be found at [25].

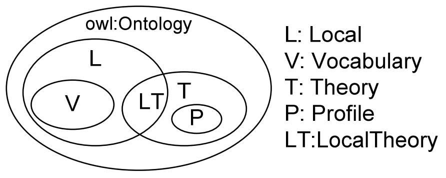

**Fig. 6.** Ontology classification.

### 4.2.2 Normalized Ontological System

With the conceptualizations established in O3, we can now define two ideal forms of ontology deployment.

- *Intensionally normalized form (INF)*: An ontological system is in intensionally normalized form if it consists of only *o3:Local* and *o3:Profile*.
- *Extensionally normalized form (ENF)*: An ontological system is in extensionally normalized form if it consists of only *o3:Vocabulary* and *o3:Theory*.

INF and ENF are not mutually exclusive. They can be conjunctively applied to an ontological system, resulting in a system that consists of only *o3:Vocabulary*, *o3:LocalTheory*, and *o3:Profile*.

By the above definition, the system deployed in Fig. 5a is an anomaly with regard to INF because ontology *a* is an *o3:Complex*. On the hand, the system deployed in Fig. 5b is in INF because ontologies *b*, *n* are *o3:Local* and *b1*, *b2* are *o3:Profile*. As we have discussed earlier, the advantage of an INF system is that ontologies' inter-dependency is carefully managed so that each ontology has the most shareability.

But deploying ontologies in INF is still insufficient to maintain the URI's stability under certain circumstances. A case in point is the conceptualization of *biopax:CONTROLLER* (See Fig. 3), whose two conceptualizations, if shared, leads to a logical contradiction. Thus, if the logical constraints of *CONTROLLER* are bound with the instantiation of its URI, two different URIs must be used to denote the two kinds of *CONTROLLER*. However, the two types of CONTROLLER differ only in their intensional meanings; their extensional meanings are still the same, implying that their URIs do not have to be different. To maintain the URI's stability, CONTROLLER can be first developed in an *o3:Vocabulary*, where its extensional meanings are first to be specified. This vocabulary URI can then be subsequently used in various *o3:Theory* to define its intensional meanings. In the history of science, a fundamental change in our conceptualization about an external entity was rarely, if ever, accompanied by a change in the words that are used to refer to the entity. For instance, Copernican revolution changed our conceptualization about the earth, but not the usage of the word "earth". The same principle, we think, should also be applied to ontological systems as well, and deploying ontologies in ENF will help in this regard. An *o3:Vocabulary* only establishes the identities of resources in the web but does not make any logical assertions about them. URIs defined in an *o3:Vocabulary*, therefore, can be shared in any logic theories, regardless if they contradict each other or not.

## 5. Conclusion

The web is an open, decentralized system, in which communication is carried out through sharing resources. Ontologies are no exception. They are web resources and they are to be shared. Good ontologies thus should be developed to have the maximal shareability. Of course, correct conceptualization necessitates an ontology's shareability. But other factors would also contribute. The first one is the size of an ontology. An ideal ontological system should not be comprised of a single, consistent, and comprehensive ontology. Putting aside the debate on whether such an ontology is attainable, we think that, even if it were, it may not be desirable. The size of knowledge base is always an important performance factor for an ontology driven application[26]. Using a few concepts from a galaxy-like ontology will not be a sensible approach under most circumstances. Most domain applications would have a well defined local task that rarely demands concepts beyond their domain knowledge. For instance, a BioPAX-driven program is unlikely to be interested in whether a *pathway* or a *conversion* is aligned to the "Physical" or "Abstract" as defined in SUMO. Forcing their alignment into a single ontology can only tax the program with unnecessary computation cost. This is, however, not to say that SUMO, or any other top ontology of the similar nature, is not useful. On the contrary, we believe that the opposite is true. A well defined top ontology always serves a valuable conceptual framework for guiding the design of domain ontologies. But, conceptually alignment does not have to be always realized in physical implementation. The broad spectrum of application needs demands that all kinds of ontologies be developed and be at every grains of semantic granularity. Moreover, each of them should exist independently while readily be assembled into a coherent system.

The second contributing factor to an ontology's shareability is its stability. As no useful engineer system can be developed to varying specifications and no agreement can be made to varying subjects, an ontology, once developed and deployed, must seldom, if ever, change. But things always change with time; ontologies are no exception. On one hand, scientific progress can change our conceptualizations about an extensional entity. On the other hand, technological advancement and social interaction can change the way that a particular problem is solved. The challenge, therefore, lies in how to find the balance between a system's stability and its adaptability. As shown in this article, the monolithic approach taken by the BioPAX ontologies cannot meet the challenge. In a monolithic system, change must take place in an all-or-none fashion. Because all conceptualizations are physically bound together, a partial change must be made in the original ontology or a completely newly developed one. Both approaches have serious drawbacks. The former risks the danger of breaking an existing application that depends on the previous conceptualizations; the latter impedes data interoperability by using different set of URIs for the same conceptualization.

"A new scientific truth", as Max Planck has judiciously stated in [27], "does not triumph by convincing its opponents and making them see the light, but rather because its opponents eventually die, and a new generation grows up that is familiar with it." Although an ontological system engineers – as opposed to discover – scientific truth, its use and sharing in the semantic web should nevertheless follow the same principle. A new ontology should not be put into use by replacing an older one. Rather, it should reuse or compete with the older one if reuse is not possible. The ultimate winner is chosen by the users rather than by the designers of an ontology. In this sense, the best ontologies can be defined to be the ontologies that are mostly shared, and the worst are those that are seldom linked. There is perhaps no such thing as the ideal ontology but only the ideal ontological system that is capable of fostering an ideal one. We hope that the few design principles and engineer techniques introduced in this article may offer help toward building such a system.

**Acknowledgement.** This work was developed while the first author was supported by a Portuguese FCT (Fundação para a Ciência e a Tecnologia, Portugal) Post-doctoral Fellowship grant with reference SFRH/BPD/34997/2007. The work is also partially supported by FCT supported PTDC program (project DynaMo - PTDC/EEA-ACR/69530/2006) and by the NHLBI Proteomics Initiative through contract N01-HV-28181 to the Medical University of South Carolina, PI. D. Knapp, specifically as refers to the work of its bioinformatics core (core C) and mathematically modeling project (project 7), as well as by its administrative center, separately funded by the same initiative to the same institution, PI. M.P. Schachte.

## References

1. Guarino, N., Carrara, M., Giaretta, P.: Formalizing Ontological Commitments. National Conference on Artificial Intelligence (AAAI-94). Morgan Kaufmann, Seattle, USA (1994)
2. Guarino, N.: Formal Ontology and Information Systems, in: Formal Ontology in Information Systems. IOS Press, Amsterdam, Netherlands (1998)
3. Farquhar, A., Fikes, R., Rice, J.: The Ontolingua Server: A Tool for Collaborative Ontology Construction. KSL-96-26. Knowledge Systems Laboratory, Department of Computer Science, Stanford University, Stanford, CA. (1996) http://ksi.cpsc.ucalgary.ca/KAW/KAW96/farquhar/farquhar.html
4. Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition **5** (1993) 199-220
5. Gruber, T.: Interview Tom Gruber. SIGSEMIS Bulletin **1** (2004) 4-9
6. Bray, T., Hollander, D., layman, A., Tobin, R.: Namespaces in XML 1.0 (Second Edition): W3C Recommendation. (2006) http://www.w3.org/TR/REC-xml-names/#ns-qualnames
7. Berners-Lee, T.: Notation 3. (1998) http://www.w3.org/DesignIssues/Notation3.html
8. Wang, X., Almeida, J.S.: DLG$^2$- A Graphical Presentation Language for RDF and OWL. (2005) http://www.charlestoncore.org/dlg2/
9. Wang, X., Oliveira, A.L.: DLG$^2$ Extensions for Property Constrains. (2007) http://dfdf.inesc-id.pt/dlg2

10. Kashyap, V., Sheth, A.: Semantic and Schematic Similarities between Database Objects: A Context based Approach. VLDB Journal **5** (1996)
11. Wang, X., Almeida, J.S., Oliveira, A.L.: Compatability Analysis of Terms in BioPAX Ontologies. (2007) http://dfdf.inesc-id.pt/tr/biopax-analysis
12. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. Proceedings of the 14th international conference on World Wide Web. ACM Press, Chiba, Japan (2005)
13. Bouquet, P.G., Fausto; van Harmelen, Frank; Serafini, Luciano; Stuckenschmidt, Heiner: Contextualizing Ontologies. Journal of Web Semantics **1** (2004)
14. Gruber, T.R.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing. Knowledge Systems Laboratory. Stanford University. (1993) file:///c:/refs/ontology/gruber93b.pdf
15. Berners-Lee, T., Mendelsohn, N.: The Rule of Least Power. (2006) http://www.w3.org/2001/tag/doc/leastPower.html
16. Niles, I., Pease, A.: Towards a Standard Upper Ontology. In: Welty, C., Smith, B. (eds.): Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), Ogunquit, Maine (2001) file:///c:/refs/ontology/methodology/niles_pease_01.pdf
17. Rector, A.: Modularisation of Domain Ontologies Implemented in Description Logics and related formalisms including OWL. Knowledge Capture. ACM, Sanibel Island, FL (2003) 121-128 file:///c:/refs/ontology/methodology/rector_03.pdf
18. Atkinson, M., Morrison, R.: Orthogonally persistent object systems The VLDB Journal **4** (1995) 319-402
19. Date, C.J., McGoveran, D.: A New Database Design Principle. Database Programming & Design **7** (1994) 46-53
20. W3C Technical Architecture Group: Architecture of the World Wide Web, Volume One. In: Jacobs, I., Walsh, N. (eds.): (2004) http://www.w3.org/TR/webarch/
21. DCMI: DCMI Metadata Terms. http://dublincore.org/documents/dcmi-terms/
22. Bylander, T., Chandrasekaran, B.: Generic tasks in knowledgebased reasoning: The right level of abstraction for knowledge acquisition. In: Gaines, B.R., Boose, J.H. (eds.): Knowledge Acquisition for Knowledge Based Systems. Academic Press, London (1988)
23. Murray-Rust, P., Rzepa, H.S.: Chemical markup, XML and the World-Wide Web. 2. Information objects and the CMLDOM. J Chem Inf Comput Sci **41** (2001) 1113-1123
24. Murray-Rust, P., Rzepa, H.S.: STMML. A markup language for scientific, technical and medical publishing. Data Science Journal **1** (2002) 1-65
25. Wang, X., Almeida, J.S., Oliveira, A.L.: Ontology of Ontologies. (2008) http://dfdf.inesc-id.pt/ont/o3
26. Heinsoh, J., Kudenko, D., Nebel, B., Profitlich, H.-J.: An Empirical Analysis of Terminological Representation System. Artificial Intelligence **68** (1994) 367-397
27. Planck, M.: Scientific Autobiography and Other Papers. Philosophical Library, New York (1949)