A Catalogue of Recurrent Pitfalls in Ontology Modelling

María Poveda, Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez
Ontology Engineering Group. Departamento de Inteligencia Artificial.
Facultad de Informática, Universidad Politécnica de Madrid.
Campus de Montegancedo s/n.
28660 Boadilla del Monte. Madrid. Spain
mpoveda@delicias.dia.fi.upm.es, {mcsuarez, asun}@fi.upm.es

Abstract. Modelling ontologies has become one of the main topics of research within the Ontology Engineering field because of the difficulties this area presents. These difficulties can involve the inclusion of anomalies or pitfalls in the modelling. In this regard, it is important both (a) to prevent the appearance of pitfalls in ontologies and (b) to correct the ontologies containing such common anomalies, with the aim of improving the ontology quality. In this paper we present a catalogue of recurrent pitfalls made by developers when modelling ontologies. Additionally, we provide some guidelines for helping developers to model ontologies free of pitfalls and to correct the ontologies that contain pitfalls.

Keywords: pitfalls, ontology modelling, ontology engineering

1 Introduction

Ontology modelling has become one of the main issues in the current ontology engineering research, since ontologies are becoming more and more popular among a wider range of users. Ontology modelling has been mainly performed using different knowledge modelling techniques (such as artificial intelligence modelling techniques based on frames and first-order logic, representation techniques based on description logics, etc.) [5]. Recently, a new modelling technique based on the use of Ontology Design Patterns (ODPs) [3] is also emerging.

However, independently of the technique used to develop an ontology, developers must tackle with a wide range of difficulties and handicaps when modelling ontologies. Most of the times, these difficulties imply the inclusion of anomalies in the ontology modelling [1, 2].

In this regard, it is worth mentioning that in [11] authors describe a set of common errors made by developers during ontology modelling. Moreover, in [6] it is shown a classification of errors identified when the consistency, the completeness, and the conciseness is evaluated in ontology taxonomies. Finally, in [9, 10] authors identify an initial set of common pitfalls as well as a classification of them.

Therefore, in this context it is important both (a) to prevent the appearance of pitfalls in ontologies and (b) to evaluate and correct the ontologies containing such common anomalies, with the goal of improving the ontology quality. With respect to the prevention of making pitfalls, in [8] authors provide several guidelines to avoid typical errors when modelling ontologies. Such guidelines help the ontology developer to prevent specifically errors related to the definition of classes, class hierarchies, and properties.

Our goal in this paper is to present a catalogue of recurrent pitfalls in ontology modelling that has been obtained as a result of an empirical analysis of 26 ontologies. In addition we aim to propose some guidelines to avoid the inclusion of pitfalls in ontology modelling as well as some help to correct common anomalies in ontologies.

The remainder of the paper is structured as follows: Section 2 presents the state of the art on errors when modelling ontologies. Section 3 describes the experiment carried out to obtain the ontologies that are the subject of our analysis, whose main goal was to find pitfalls already identified in the literature and new common pitfalls. Section 4 includes the catalogue of pitfalls together with their definitions. Section 5 shows the analysis of the pitfalls in the aforementioned set of ontologies; it includes the confirmation of recurrent existing pitfalls. Section 6 includes some guidelines to avoid and to correct a subset of the identified pitfalls. Finally, Section 7 includes the conclusions drawn and future lines of work.

2 State of the Art

The ontology modelling is a crucial activity in ontology engineering. However, it is well known that the ontology modelling involves several difficulties. As reported in [11], the development of ontologies using directly ontology languages, such as OWL DL, is not trivial. In [11] it is stated that for most people it is very difficult to understand the logical meaning and potential inferred statements of any DL formalism, including OWL DL. Such a paper [11] presents (a) the most common problems, errors, and misconceptions on understanding OWL DL as well as (b) tips on how to avoid such pitfalls in building OWL DL ontologies besides some help with the precise meaning of OWL-DL through questions and paraphrases. The common errors are mainly related to (a) the failure to make information explicit, (b) the mistaken use of universal and existential restrictions, (c) the open world reasoning, and (d) the effect of domain and range constraints.

With respect to the prevention of pitfalls, in [8] authors provide several guidelines to avoid typical errors when modelling ontologies. Such guidelines help the ontology developer to prevent errors related to the definition of classes, class hierarchies, and properties.

Other authors have mainly focused their efforts on the identification of errors that appear when modelling taxonomies in ontologies. In this context, the author of [6] relates a set of errors in ontologies with the dimensions of consistency, completeness, conciseness, expandability and sensitiveness. In addition, some examples of these types of errors when modelling taxonomies are presented in [6]. Besides, in [7] a methodology for validating taxonomical relationships based on the philosophical notions of essence, identity, and unity is presented.

3 Experiment on Developing Ontologies

In this section we present the experiment carried out on ontology development with the aim of analyzing the resultant ontologies. Section 3.1 presents the setting of the experiment, which was made up of two main parts: (1) ontology development by participants, described in Section 3.1.1, and (2) analysis of the ontologies, described in Section 3.1.2.

3.1 Experiment Setting

Through this section we present the setting of the experiment we have carried out. In Subsection 3.1.1, we describe the session setup for the experiment in which we have obtained a set of ontologies to be analyzed. Then, in Subsection 3.1.2, we present the approach followed during the analysis carried out over the aforementioned ontologies. The analysis allowed us to identify a set of anomalies that ontology developers, mainly beginners, often make.

3.1.1 Session Setup

We have divided the experiment into 3 separate sessions. Each session involved a different set of participants and was held in different periods of time. All the sessions were carried out involving participants with background in computer science and some experience in ontology engineering. Participants, who worked in groups of up to two, were distributed through of the following training courses:

S1. 22 participants attended the "Ontologies and the Semantic Web" course at the Master on Information Technologies (in the following IT Master 07-08) at the *Universidad Politécnica de Madrid (UPM)*, from October 2007 to February 2008.

S2. 10 participants attended the "Ontologies and the Semantic Web" course at the Master on Information Technologies (in the following IT Master 09-10) at UPM, from October 2009 to February 2010.

S3. 19 participants attended the "Ontologies and the Semantic Web" course at the Master of Research on Artificial Intelligence (in the following AI Master 09-10) at UPM, from October 2009 to February 2010.

In all the sessions the participants were taught in:

1. the theoretical foundations of ontologies
2. ontologies and terminologies
3. ontology languages, including RDF(S) and OWL
4. methodologies for building ontologies, specifically in the NeOn Methodology framework [12] and the methodological guidelines for specifying requirements, scheduling, reusing, reengineering, and mapping knowledge resources as well as modelling ontologies.

Additionally, all the sessions incorporated independent practical training in:

1. building a proof ontology in RDF(S) and OWL
2. writing the ontology requirements specification document (ORSD) [12, 13] in one of the domains proposed and shown in Table 1

3. reusing and reengineering non-ontological resources
4. reusing general ontologies
5. modelling ontologies

At the end of each session, participants should perform a global practical training, which consists on the development of an ontology in the domain selected when the requirements specification activity was performed. It is worth mentioning that the trainers reviewed, corrected, and improved all the ORSDs provided by the participants; and thus, participants used during the global practical training the corrected ORSD.

In Table 1 you can observe both the domains proposed by the trainers and the number of ontologies developed by the participants in each session with respect to the domains proposed. As a result of this phase, we obtained 26 ontologies: 11 ontologies in session S1 developed with Protégé, 5 ontologies in session S2 developed both with Protégé and NeOn Tookit, and 10 ontologies in session S3 built with Protégé and NeOn Toolkit.

Table 1. Domains proposed and number of ontologies by domains

Domain	**S1**	**S2**	**S3**	**Total**
Architecture	2			**2**
Art	2			**2**
Biological organisms			1	**1**
Community Services	4			**4**
Electronic devices		1		**1**
Food			2	**2**
Football		1	2	**3**
Geography	2			**2**
Judo			1	**1**
Movies		1		**1**
Music			1	**1**
Personality	1			**1**
Religion			1	**1**
Theatre			1	**1**
Traveling		1		**1**
Vehicles		1	1	**2**
Total	**11**	**5**	**10**	**26**

3.1.2 Ontology Analysis Approach

The focus of the ontology analysis performed with the 26 ontologies, obtained in the global practical training explained in Section 3.1.1, was on the following dimensions: structural, functional, and usability-profiling [4]. The analysis was carried out by manually inspect each ontology with respect to the following aspects:

- Modelling issues. This aspect refers to the structural dimension. In this sense we analyzed modelling decisions taken by participants paying particular attention to whether they could be improved.

- Coverage of the problem. This aspect is related to the functional dimension. In this sense we observed if the ontology represents all the features that were expected to find in it. This aspect is directly related to the requirements specified in the ORSD.
- Understandability and clarity. These aspects refer to the usability-profiling dimension. In this sense we observed if the ontology provides some information to facilitate its comprehensibility.

As a result of this analysis, we obtained a set of shortcoming in the ontologies that we identified as pitfalls. We want to note that we consider pitfalls as a clue to find possible mistakes that often appear when modelling ontologies.

The results of the analysis are shown in the next sections. Section 4 shows a set of pitfalls that include not only those discovered in this experiment but also the ones identified by [6, 8, 11]. Section 5 demonstrates that there are recurrent pitfalls that occur when developers build ontologies without knowing this set of pitfalls in advance.

4 Catalogue of Pitfalls

The manual inspection of the 26 ontologies implemented in OWL-DL obtained in the global practical training explained in Section 3.1.1 allows us to demonstrate that there are common pitfalls that occur when developers build ontologies knowing modelling techniques and ontological foundations but without knowing the set of pitfalls in advance. As a result of this empirical analysis, in this section we present the catalogue of 24 pitfalls[1].

P1. **Creating polysemous elements:** an ontology element, whose name has different meanings, is included in the ontology to represent more than one conceptual idea. For example, the class "Theatre" is used to represent both the artistic discipline and the place in which a play is performed.

P2. **Creating synonyms as classes:** several classes whose identifiers are synonyms are created and defined as equivalent. As an example we could define "Car", "Motorcar" and "Automobile" as equivalent classes. This pitfall is related to the guidelines in [8] which explain that synonyms for the same concept do not represent different classes.

P3. **Creating the relationship "is" instead of using "subclassOf", "instanceOf" or "sameIndividual":** a relationship "is" is created in the ontology instead of using OWL primitives for representing the subclass relationship ("subclassOf"), the membership to a class ("instanceOf"), or the equality between instances ("sameAs"). An example of this type of pitfall is to define the class "Actor" in the following way 'Actor $\equiv$ Person $\sqcap$ $\exists$interprets.Actuation $\sqcap$ $\exists$is.Man'. This pitfall is related to the guidelines for understanding the "is-a" relation provided in [8].

P4. **Creating unconnected ontology elements:** ontology elements (classes, relationships or attributes) are created with no relation to the rest of the

[1] http://delicias.dia.fi.upm.es/wiki/index.php/Catalogue_of_Pitfalls

ontology. An example of this type of pitfall is to create the relationship "memberOfTeam" and to miss the class representing teams; thus, the relationship created is isolated in the ontology.

P5. **Defining wrong inverse relationships:** two relationships are defined as inverse relations when actually they are not. For example, something is sold or something is bought; in this case, the relationships "isSoldIn" and "isBoughtIn" are not inverse.

P6. **Including cycles in the hierarchy** [6, 8]**:** a cycle between two classes in the hierarchy is included in the ontology, when it is not intended to have such classes as equivalent. That is, some class A has a subclass B and at the same time B is a superclass of A. An example of this type of pitfall is to create the class "Professor" as subclass of "Person", and the class "Person" as subclass of "Professor".

P7. **Merging different concepts in the same class:** a class whose identifier is referring to two or more different concepts is created. An example of this type of pitfall is to create the class "StyleAndPeriod" or "ProductOrService".

P8. **Missing annotations:** the ontology terms lack of annotations properties. It is worth mentioning that this type of properties improve the ontology understanding and usability from a user point of view.

P9. **Missing basic information:** needed information is not included in the ontology. Sometimes this pitfall is related with requirements in the ORSD [12, 13] that are not covered by the ontology. Other times it is related with details that could be added to the ontology in order to make it more complete without complicate it too much. An example of this type of pitfall is to create the relationship "startsIn" to represent that the routes have a starting point in a particular location; and to miss the relationship "endsIn" to show that a route has an end point.

P10. **Missing disjointness** [6, 8, 11]**:** the ontology lacks of disjoint axioms between classes or between properties that should be defined as disjoint. For example, we can create the classes "Odd" and "Even" (or the classes "Prime" and "Composite") without being disjoint; such representation is not correct based on the own definition of these types of numbers.

P11. **Missing domain or range in properties:** relationships and/or attributes without domain or range (or both) are included in the ontology. There are situations in which the relation is very general and the range can be the most general concept "Thing"; however, in other cases, the relations are more specific. An example of this type of pitfall is to create the relationship "hasWritten" in an ontology about art in which the relationship domain could be "Writer" and the relationship range could be "LiteraryWork". This pitfall is related to the common error when defining ranges and domains described in [11].

P12. **Missing equivalent properties:** when an ontology is imported into another, classes that are duplicated in both ontologies are normally defined as equivalent classes. However, the ontology developer misses the definition of equivalent properties in those cases of duplicated relationships and attributes. For example, the classes "CITY" and "City" in two different ontologies are defined as equivalent classes; however, relationships "hasMember" and "has-Member" in two different ontologies are not defined as equivalent relations.

P13. **Missing inverse relationships:** there are two relationships in the ontology that should be defined as inverse relations. For example, the case in which the ontology developer omits the inverse definition between the relations “hasLanguageCode” and “isCodeOf”.

P14. **Misusing “allValuesFrom”** [11]**:** this pitfall can appear in two different ways. In the first, the anomalie is to use the universal restriction (“allValuesFrom”) as the default qualifier instead of using the existential restriction (“someValuesFrom”). This means that the developer thinks that “allValuesFrom” implies “someValuesFrom”. In the second, the mistake is to include “allValuesFrom” to close off the possibility of further additions for a given property. An example of this type of pitfall is to define the class “Book” as ‘producedBy some Writer and uses only Paper’ and closing the possibility of adding “Ink” as element use in the writing.

P15. **Misusing “not some” and “some not”** [8]**:** to confuse the representation of “some not” and “not some”. An example of this type of pitfall is to define a vegetarian pizza as any pizza which both has *some* topping which is *not* meat and also has *some* topping which is *not* fish.

P16. **Misusing primitive and defined classes** [11]**:** to fail to make the definition ‘complete’ rather than ‘partial’ (or ‘necessary and sufficient’ rather than just ‘necessary). It is critical to understand that, in general, nothing will be inferred to be subsumed under a primitive class by the classifier. This pitfall implies that the developer does not understand the open world assumption.

P17. **Specializing too much a hierarchy:** the hierarchy in the ontology is specialized in such a way that the final leaves can not have instances. An example of this type of pitfall is to create the class “RatingOfRestaurants” and the classes “1fork”, “2forks”, and so on, as subclasses instead of as instances. This pitfall is related to the guidelines for distinguishing between a class and an instance provided in [8].

P18. **Specifying largely the domain or the range** [8, 11]**:** not to find a domain or a range general enough. An example of this type of pitfall is to restrict the domain of the relationship “isOfficialLanguage” to the class “City”, instead of allowing also the class “Country” or a more general concept such as “GeopoliticalObject”.

P19. **Swapping intersection and union:** the ranges and/or domains of the properties (relationships and attributes) are defined by intersecting several classes in cases in which the ranges and/or domains should be the union of such classes. An example of this type of pitfall is to create the relationship “takesPlaceIn” with domain “OlympicGames” and with range the intersection of the classes “City” and “Nation”. Another example can be to create the attribute “Name” for the classes “City” and “Drink” and to define its domain as the intersection of both classes. This pitfall is related to the common error that appears when defining ranges and domains described in [11] and also related to the guidelines for defining these elements provided in [8].

P20. **Swapping Label and Comment:** the contents of the Label and Comment annotation properties are swapped. An example of this type of pitfall is to include in the Label annotation of the class “Crossroads” the following sentence 'the place of intersection of two or more roads'; and to include in the Comment annotation the word 'Crossroads'.

P21. **Using a miscellaneous class:** to create in a hierarchy a class that contains the instances that do not belong to the sibling classes instead of classifying such instances as instances of the class in the upper level of the hierarchy. This class is normally named "Other" or "Miscellaneous". An example of this type of pitfall is to create the class "HydrographicalResource", and the subclasses "Stream", "Waterfall", etc., and also the subclass "OtherRiverElement".

P22. **Using different naming criteria in the ontology:** no naming convention is used in the identifiers of the ontology elements. For example, we can name a class by starting with upper case, e.g. "Ingredient", and its subclasses by starting with lower case, e.g. "animalorigin", "drink", etc. This pitfall is related to the guidelines for giving names to ontology elements provided in [8].

P23. **Using incorrectly ontology elements:** an ontology element (class, relationship or attribute) is used to model a part of the ontology that should be modelled with a different element. An example of this type of pitfall is to create the relationship "isEcological" between an instance of "Car" and the instance "Yes" or "No", instead of creating the attribute "isEcological" whose range is Boolean. This pitfall is related to the guidelines for distinguishing between a class and a property value provided by [8].

P24. **Using recursive definition:** an ontology element is used in its own definition. For example, it is used to create the relationship "hasFork" and to establish as its range the following 'the set of restaurants that have at least one value for the relationship "hasFork".

5 Pitfall Analysis: Recurrent Pitfalls

This section summarizes the set of pitfalls identified in the 26 ontologies developed by the 51 participants in the 3 experiment sessions presented in Section 3. Table 2 includes the occurrence of the 24 pitfalls previously identified in the 26 ontologies. During the analysis, we have observed that some pitfalls can appear almost in all the elements of the ontology, for example, the P8 "Missing annotations". In contrast, other pitfalls only appear in specific parts of the ontology for example, the P13 "Missing inverse relationships". Because of this, the numbers in Table 2 show how many ontologies contain the pitfall instead of how many times appears the pitfall along the 26 ontologies.

The recurrent pitfalls are signed by (*) in Table 2. Such pitfalls are mainly related to (a) create ontology elements disconnected from the rest of the ontology; (b) miss knowledge (annotations, basic information, disjoint axioms, domain or range in properties); (c) specialize too much a hierarchy; (d) swap knowledge representation elements (intersection and union, and label and comment); and (e) use different naming conventions in the ontology.

It is also worth mentioning those pitfalls that are less recurrent. Such pitfalls are mainly related to (a) create synonyms as classes; (b) create an ad-hoc relation named as “is”; and (c) use the same URI for different ontology elements. The latter pitfall is now avoided thanks to the new interface of Protégé 4.0.2.

Finally, we can mention two pitfalls that do not appear in any of the ontologies analyzed. These are including cycles in the hierarchy and misusing ‘not some’ and ‘some not’. Probably, these pitfalls do not appear in the ontologies analyzed because there were no requirements related to these aspects in the ORSDs used by the participants.

Table 2. Appearance of existing pitfalls

Pitfalls	**IT Master 07-08**	**IT Master 09-10**	**AI Master 09-10**	**Total**
P1. Creating polysemous elements	3	0	0	**3**
P2. Creating synonyms as classes	2	0	0	**2**
P3. Creating the relationship “is” instead of using “subclassOf”, “instanceOf” or “sameIndividual”	1	0	1	**2**
P4. Creating unconnected ontology elements (*)	4	2	5	**11**
P5. Defining wrong inverse relationships	2	0	1	**3**
P6. Including cycles in the hierarchy	0	0	0	**0**
P7. Merging different concepts in the same class	2	2	2	**6**
P8. Missing annotations (*)	7	5	8	**20**
P9. Missing basic information (*)	8	3	5	**16**
P10. Missing disjointness (*)	10	3	10	**23**
P11. Missing domain or range in properties (*)	8	4	13	**25**
P12. Missing equivalent properties	1	1	2	**4**
P13. Missing inverse relationships	4	1	1	**6**
P14. Misusing “allValuesFrom”	2	1	2	**5**
P15. Misusing “not some” and “some not”	0	0	0	**0**
P16. Misusing primitive and defined classes	4	0	4	**8**
P17. Specializing too much a hierarchy (*)	7	3	5	**15**
P18. Specifying largely the domain or the range	4	2	3	**9**
P19. Swapping intersection and union (*)	6	1	6	**13**
P20. Swapping Label and Comment (*)	9	0	4	**13**
P21. Using a miscellaneous class	2	0	1	**3**
P22. Using different naming criteria in the ontology (*)	6	3	6	**15**
P23. Using incorrectly ontology elements	2	2	4	**8**
P24. Using recursive definition	3	0	2	**5**
Total	**97**	**33**	**85**	

6 Guidelines to avoid Pitfalls

With the aim of developing ontologies without errors, it is important both (a) to prevent the appearance of pitfalls in ontologies and (b) to correct the ontologies containing such common anomalies, with the aim of improving their quality. In this paper we provide some guidelines for helping developers to model ontologies free of pitfalls and to correct the ontologies that contain pitfalls.

Our goal is to provide simple and direct guidelines that can be used both by ontology experts and naïve ontology developers. Therefore, our proposal consists of describing each pitfall identified in our catalogue (Section 4) with the following information in the so-called pitfall description card. This card follows the template shown in Table 3 and includes:

1. A description of the pitfall explaining possible inconveniences that it could entail and situations in which the pitfall could appear. This information appears in the "Description" slot.
2. Some guidelines to avoid the pitfall appearance when modelling ontologies are available in the "Recommendation" slot.
3. Some preliminary guidelines to correct a pitfall once it have appeared can be found in the "Example" slot, including its "Not recommended" and "Recommended" fields.
4. Additional information could be found in the "Comments" slot if necessary.

We are now focused on the guidelines to avoid pitfalls, but we also provide some preliminary ideas on how to correct a particular pitfall once it has appeared in an ontology. Due to the different nature of the pitfalls their corresponding preliminary guidelines provided can be described at different levels of granularity or abstraction.

Table 3. Pitfall description card template (based on [9])

Name	Name of the pitfall intended to avoid
Description	
This field includes a general description of the pitfall. It can also include the possible inconveniences that pitfall may imply and the different cases in which the pitfall can appear.	
Recommendation	
This field provides some guidelines or recommendations to avoid the pitfall.	
Example	
This field includes an example of the pitfall as well as some guides to correct it. In the "Not Recommended" field we can see a graphical representation of the example where the pitfall has been identified. In the "Recommended" field we can see a possible design for the problem proposed in the example; such a design tries to avoid the pitfall.	
Not Recommended	**Recommended**
Graphic representation: ontology editor screenshots or UML diagrams.	Graphic representation: ontology editor screenshots or UML diagrams.
Comments	
This optional field includes possible comments on the pitfall or on the associate recommendation.	

In this section, we describe a subset of the recurrent pitfalls identified in Section 4 following the template shown in Table 3. The pitfalls described in this section have been selected among the most recurrent ones. We will describe them in the following order, which corresponds to the alphabetical order followed in the catalogue (See Section 4): “P4. Creating unconnected ontology elements” (Table 4), “P9. Missing basic information (Table 5), “P10. Missing disjointness” (Table 6), “P11. Missing domain or range in properties” (Table 7), and. “P22. Using different naming criteria in the ontology” (Table 8).

Table 4. Pitfall “P4. Creating unconnected ontology elements”

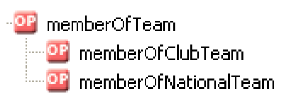
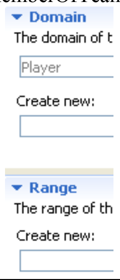
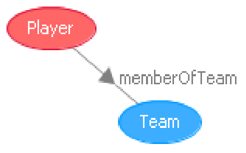
Name	P4. Creating unconnected ontology elements
Description	
The pitfall consists in defining a part of the ontology that has no relation with the rest. These unconnected parts can be classes, relationships or attributes.	
Recommendation	
Some possible recommendations to avoid this pitfall are: ▪ When a new class is created, to verify whether it could be: ○ domain or range of at least a relationship, which can have been created or can be created later on. ○ domain of at least an attribute, , which can have been created or can be created later on. ○ subclass or superclass of another class, which can have been created or can be created later on. ▪ When a new relationship is created, to verify whether both its domain and range already exist in the ontology or there are plans to create them. ▪ When a new attribute is created, to verify whether its domain already exists in the ontology or there are plans to create it.	
Example	
In the not recommended scenario, it is shown an ontology on football. In this case, we can observe that the relationship “memberOfTeam” has been created defining its domain but not its range. Also, it should be noted that there is not a class that represents teams in the ontology. In the recommended scenario, we have created the class “Team” and defined it as range of the relationship “memberOfTeam”.	
Not Recommended	**Recommended**
 (domain and range of “memberOfTeam”)  Domain The domain of t Player Create new: Range The range of th Create new:	
Comments	
This pitfall could be related to pitfall “P11. Missing domain or range in properties”.	

Table 5. Pitfall “P9. Missing basic information”

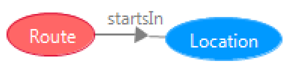
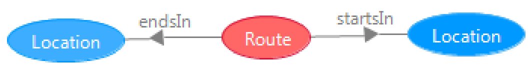
Name	P9. Missing basic information
Description	
This pitfall entails not representing part of the information needed that should be included in the ontology. Some of cases in which this pitfall can appear are: ▪ To create a relation and to miss its inverse relation (e.g., “hasDescendant” and “isDescendantOf”). ▪ To create a relation and to miss its complementary relation if necessary (e.g., “startsIn” and “endsIn”). ▪ To miss an explicit taxonomy when it is known. ▪ To miss the relation between classes and their instances.	
Recommendation	
It is recommended to apply the following actions to develop a more complete ontology: ▪ When a relationship is created, to check if any inverse relationship exists, and if so, then to add also that relationship and create them as inverse. In this way, we could take advantage of the reasoning power of the inverse relationships. ▪ When a property (relation or attribute) is created, to check if any complementary property exists, and if so, then to add also that property. ▪ When a class is included in the ontology, to check if it can be subclass or superclass of other existing class in the ontology. In this way, we could take advantage of the reasoning power of the inheritance due to make explicit taxonomical knowledge. ▪ When an instance is included in the ontology, to check if there are classes in the ontology to which the instance can belong. In this way, we could take advantage of the reasoning power of the inheritance.	
Example	
This example shows a particular situation that refers to the second case in the ‘description’ slot. In the non recommended scenario, we can observe that the relation “startsIn” is included in the ontology to represent that the routes have as starting point a location; however, the complementary relation (that is, “endsIn”) is not included in the ontology. In the recommended scenario, the ontology contains both the relation “startsIn” and the relation “endsIn” with its domain (“Route”) and its range (“Location”).	
Not Recommended	**Recommended**
	

Table 6. Pitfall "P10. Missing disjointness"

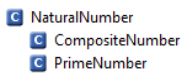
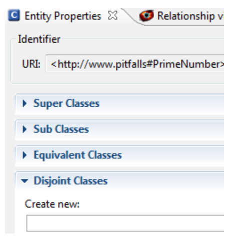
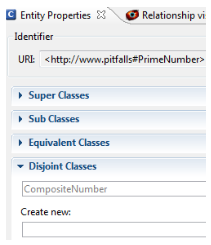
Name	P10. Missing disjointness
Description	
The pitfall consists in missing disjoint axioms between classes or properties that should be defined as disjoint.	
Recommendation	
It is recommended to apply the following action: ▪ For each two classes in the same level of granularity in a hierarchy, to check if it is possible that a particular instance belongs to both classes at the same time. If not, then such classes should be defined as disjoint. By defining classes that can not share instances as disjoints we could detect inconsistent definitions that would go unnoticed otherwise. Moreover, we obtain a more complete and better defined ontology.	
Example	
In the non recommended scenario, we can observe a hierarchy of three classes; the class "NaturalNumber" that has as subclasses "PrimeNumber" and "CompositeNumber". However, these subclasses have not been defined as disjoint. Therefore, the number "7" could be both prime and composite. In the recommended scenario, the subclasses "PrimeNumber" and "CompositeNumber" have been defined as disjoint. Thus, the fact that a natural number cannot be at the same time prime and composite is defined explicitly in the ontology.	
Not Recommended	**Recommended**
(properties of "PrimeNumber")   Entity Properties Relationship v Identifier URI: <http://www.pitfalls#PrimeNumber> Super Classes Sub Classes Equivalent Classes Disjoint Classes Create new:	(properties of "PrimeNumber")  Entity Properties Relationship vi Identifier URI: <http://www.pitfalls#PrimeNumber> Super Classes Sub Classes Equivalent Classes Disjoint Classes CompositeNumber Create new:

Table 7. Pitfall "P11. Missing domain or range in properties"

Name	P11. Missing domain or range in properties
Description	
This pitfall entails defining relationships and/or attributes without domain or range (or both) in the ontology.	
Recommendation	
It is recommended to specify, if possible, the domains and/or ranges in the properties and/or attributes. In this way, we could improve both the definitions of such properties and the ontology understandability. Also, by defining domains and ranges, we could take advantage of the reasoning power of those axioms. Additionally, annotations in the "comment" field should be added. These annotations should describe what it is intended to be represented with the property and/or attribute (for instance, if the property or attribute is part of an N-ary pattern).	
Example	

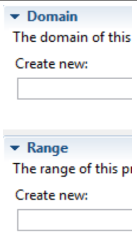
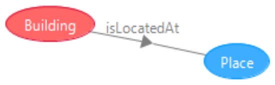
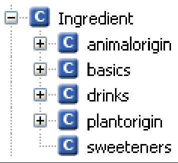
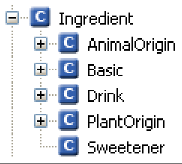
Example	
In the non recommended scenario, we can observe how the relationship "isLocatedAt" has been defined without specifying its domain nor its range; whereas, in the recommended scenario, the relationship "isLocatedAt" has been defined by specifying its domain and range ("Building" and "Place", respectively).	
Not Recommended	**Recommended**
(domain and range of "isLocatedAt")  Domain The domain of this Create new: Range The range of this pı Create new:	

Table 8. Pitfall "P22.Using different naming criteria in the ontology"

Name	P22. Using different naming criteria in the ontology
Description	
This pitfall consists in not following any naming convention when providing an identifier to the ontology elements.	
Recommendation	
A possible lightweight recommendation is to name the ontology elements following these rules: ▪ All words in a class identifier start with capital letter and follow with lowercase. ▪ The first word in properties (relationships and attributes) is written with lowercases and the other words start with capital letter and follow with lowercase. ▪ Individuals follow the same rule that properties unless they are name entities.	
Example	
In the non-recommended scenario, we can observe two criteria to name classes within an ontology on food. On the one hand, "Ingredient" stars with capital letters; on the other hand, its subclasses start with lowercases. Additionally, some class names are written with the "s" of the plural. In the recommended scenario we have renamed all subclasses of "Ingredient" following the recommendation proposed above for classes. Also, we have changed the plural identifiers into singular identifiers.	
Not Recommended	**Recommended**
 Ingredient animalorigin basics drinks plantorigin sweeteners	 Ingredient AnimalOrigin Basic Drink PlantOrigin Sweetener

7 Conclusions and Future Lines of Work

This paper presents a catalogue of 24 recurrent pitfalls in ontology modelling. Such a catalogue has been identified by manually analysing 26 ontologies in different domains (e.g. art, architecture, football, vehicles, etc.). Such ontologies have been developed by participants who have attended one semester course on ontologies and semantic web foundations. It is worth mentioning that the participants were taught in the theoretical foundations of ontologies, ontologies and terminologies, ontology languages (RDF(S) and OWL), and methodologies for building ontologies.

We have also described in the paper a subset of the most recurrent pitfalls. These descriptions are based on a template that includes (a) guidelines to avoid the pitfall when modelling ontologies and (b) examples of how to solve some pitfalls once they have appeared.

It is worth mentioning that as a result of the analysis performed we have identified some concurrences between pitfalls, which means that a particular pitfall might result in having simultaneously other pitfalls in the ontology. For example, let suppose an ontology that contains (1) the relation "isLocatedIn", whose domain is "Building" and whose range is "Place" and (2) the relation "isLocationOf' with no domain and no range. The omission of domain and range in the later relation refers to the abovementioned pitfall P11. This pitfall leads to make also pitfall P13; that is, to miss the inclusion of the inverse construct between these two relations ("isLocatedIn" and "isLocationOf"). Since one of our research aims is to provide ontology developers with guidelines to avoid common pitfalls in ontology modelling, we consider that having the knowledge about which pitfalls can be concurrent in an ontology would be useful for our research. Therefore, as a future line of work, we propose to analyse in depth the different possibilities of concurrences between the pitfalls identified in this paper. In this regard, it would be interesting to identify the groups of pitfalls that usually appear simultaneously so that, once the concurrence of a pitfall is identified, the other pitfalls that could appear could also be identified and thus avoided.

In the experiment reported in this paper, the participants did not know the set of pitfalls when they modelled their ontologies. Therefore, our next step will be performed experiments that involve two sets of participants: the first set will know the catalogue of 24 pitfalls described in this paper before develop the ontology and the second set will built the ontology without knowing the catalogue in advance. Both set of participants will develop the ontologies based on the same requirements with the aim of checking how useful is to know the list of pitfalls before develop an ontology.

Finally, as we have already mentioned, examples provided in the pitfalls description address only particular cases of how to solve a pitfall once it appears. Therefore, we are also working on collecting several cases of the same pitfall, on generalizing common points of all of them, and on dividing the pitfall in more specific cases, with the aim of providing the guidelines to correct the pitfall in all the situations.

Acknowledgments. This work has been partially supported by the Spanish projects *mIO!* (CENIT-2008-1019), *GeoBuddies* (TSI 2007-65677 C02), and *BuscaMedia* (CENIT 2009-1026).

References

1. Aguado De Cea, G., Gómez-Pérez, A., Montiel-Ponsoda, E., and Suárez-Figueroa, M.C. *Natural Language-Based Approach for Helping in the Reuse of Ontology Design Patterns*. In *Proceedings of the 16th International Conference on Knowledge Engineering (EKAW)*, pp. 32-47. (2008)
2. Blomqvist, E., *Gangemi, A., Presutti, V. Experiments on Pattern-based Ontology Design*. In *Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP),* pp. 41-48. (2009)
3. Clark, P., Thompson, J., Porter, B. W. *Knowledge Patterns*. In KR2000: Principles of Knowledge Representation and Reasoning. pp. 591-600. (2000).
4. Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann J. *Modelling Ontology Evaluation* and *Validation.* Proceedings of the 3rd European Semantic Web Conference (ESWC2006), number 4011 in LNCS, Budva. 2006
5. Gómez-Pérez, A., Fernández-López, M., Corcho, O. *Ontological Engineering*. November 2003. Springer Verlag. Advanced Information and Knowledge Processing series. ISBN 1-85233-551-3. (2003)
6. Gómez-Pérez, A. *Ontology Evaluation*. Handbook on Ontologies. Pages: 251-274. 2004. ISBN: 3-540-40834-7. S. Staab and R. Studer Editors. Springer. International Handbooks on Information Systems. (2004)
7. Guarino. N., Welty. C. *An overview of OntoClean.* In The Handbook on Ontologies, (S. Staab and R. Studer, eds.). Berlin: Springer-Verlag, 2009.
8. Noy, N.F., McGuinness. D. L. *Ontology development 101: A guide to creating your first ontology .Technical Report* SMI-2001-0880, Standford Medical Informatics. (2001)
9. Poveda, M., Suárez-Figueroa, M.C., Gómez-Pérez, A. *Common Pitfalls in Ontology Development.* In "Current Topics in Artificial Intelligence, CAEPIA 2009 Selected Papers". Springer LNAI 5988. Editors: P. Meseguer, L. Mandow, R. M. Gasca. Pp: 91-100 (2010)
10. Poveda, M., Suárez-Figueroa, M.C., Gómez-Pérez, A. *Ontology Analysis Based on Ontology Design Patterns*. WOP 2009 – Workshop on Ontology Patterns at the 8th International Semantic Web Conference (ISWC 2009). Proceedings of the WOP 2009. ISBN: ISSN 1613-0073. CEUR Workshop Proceedings. pp: 155-162. Washington, DC. 25 (2009)
11. Rector, A., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R.,; Wang, H., Wroe, C. *Owl pizzas: Practical experience of teaching owl-dl: Common errors and common patterns*. In Proc. of EKAW 2004, pp: 63–81. Springer. (2004)
12. Suárez-Figueroa, M.C. *PhD Thesis: NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse*. Spain. Universidad Politécnica de Madrid. June 2010.
13. Suárez-Figueroa, M.C., Gómez-Pérez, A., Villazón-Terrazas, B.. *How to write and use the Ontology Requirements Specification Document*. In Proceedings of the 8th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE 2009). On the Move to Meaningful Internet Systems: OTM 2009. (Robert Meersman, Tharam Dillon, and Pilar Herrero) (ISBN: 978-3-642-05150-0). LNCS 5871. Volume: Part II Pp: 966-982. Vilamoura, Algarve-Portugal. November 2009