OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation

María Poveda-Villalón, Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain

Asunción Gómez-Pérez, Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain

Mari Carmen Suárez-Figueroa, Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain

ABSTRACT

This paper presents two contributions to the field of Ontology Evaluation. First, a live catalogue of pitfalls that extends previous works on modeling errors with new pitfalls resulting from an empirical analysis of over 693 ontologies. Such a catalogue classifies pitfalls according to the Structural, Functional and Usability-Profiling dimensions. For each pitfall, we incorporate the value of its importance level (critical, important and minor) and the number of ontologies where each pitfall has been detected. Second, OOPS! (OntOlogy Pitfall Scanner!), a tool for detecting pitfalls in ontologies and targeted at newcomers and domain experts unfamiliar with description logics and ontology implementation languages. The tool operates independently of any ontology development platform and is available online. The evaluation of the system is provided both through a survey of users' satisfaction and worldwide usage statistics. In addition, the system is also compared with existing ontology evaluation tools in terms of coverage of pitfalls detected.

Keywords: *Ontology, Ontology Evaluation, Ontology Quality, Ontology Validation, Pitfalls*

INTRODUCTION

The Linked Data (LD) effort has become a catalyst for the realization of the vision of the Semantic Web originally proposed by Berners-Lee et al. (2001). In this scenario, a large amount of data, annotated by means of ontologies, is shared on the Web. Such ontologies enrich the published data with semantics and help their integration. In other cases, ontologies are used to model data automatically extracted from web sources, which can be noisy and contain errors. Therefore, ontologies not only must be published according to LD principles[1], but they also must be accurate and of high quality from a knowledge representation perspective in order to avoid inconsistencies or undesired inferences.

The correct application of ontology development methodologies (e.g., METHONTOLOGY (Fernández-López et al., 1999), On-To-Knowledge (Staab et al., 2001), DILIGENT (Pinto, Tempich, & Staab, 2004), or the NeOn

DOI: 10.4018/ijswis.2014040102

Methodology (Suárez-Figueroa et al., 2012)) benefits the quality of the ontology being built. However, such a quality is not totally guaranteed because ontologists face a wide range of difficulties and handicaps when modeling ontologies (Aguado de Cea et al., 2008; Blomqvist, Gangemi, & Presutti, 2009; Rector et al., 2004), and this fact may cause the appearance of anomalies in ontologies. Therefore, in any ontology development project it is vital to perform the ontology evaluation activity since this activity checks the technical quality of an ontology against a frame of reference.

In the last decades a huge amount of research and work on ontology evaluation has been conducted. Some of these attempts define a generic quality evaluation framework (Duque-Ramos et al., 2011; Gangemi et al., 2006; Gómez-Pérez, 2004; Guarino, & Welty, 2009; Strasunskas, & Tomassen, 2008); others propose evaluating an ontology depending on its final (re)use (Suárez-Figueroa, 2010); some others propose quality models based on features, criteria, and metrics (Burton-Jones et al, 2005); whereas others present methods for pattern-based evaluation (Djedidi, & Aufaure, 2010; Presutti et al., 2008).

As a consequence of the emergence of new methods and techniques, a few tools have been proposed. These tools ease the ontology diagnosis by reducing the human intervention. This is the case of XD-Analyzer[2], a plug-in for NeOn Toolkit and Ontocheck[3] (Schober et al., 2012), a plug-in for Protégé. The former checks some structural and architectural ontology features, whereas the latter focuses on metadata aspects. Moki[4] (Pammer, 2010), a wiki-based ontology editor, also provides some evaluation features. Finally, Radon (Ji et al., 2009) is a NeOn Toolkit plug-in that detects and handles logical inconsistencies in ontologies.

This paper presents two main contributions. The first contribution consists of a live and on-line catalogue of pitfalls[5] that extends previous works on modeling errors (Allemang, & Hendler, 2011; Gómez-Pérez, 2004; Noy, & McGuinness, 2001; Rector et al., 2004) identified in the ontology engineering field including some persistent problems of accessibility emerging in the Linked Data field (Archer, Goedertier, & Loutas, 2012; Heath, & Bizer, 2011; Hogan et al., 2010). The second contribution, OOPS! (OntOlogy Pitfall Scanner!) represents a tool for diagnosing (semi-) automatically OWL[6] ontologies. This system aims to help ontology developers to evaluate ontologies and is focused on newcomers and those not familiar with description logics and ontology implementation languages. OOPS! operates independently of any ontology development platform and is available online at http://www.oeg-upm.net/oops. It should be noted here that the repair of the ontology is out of the scope of OOPS!.

In this paper we first present the catalogue of pitfalls, including a compendium of pitfalls extracted from the literature review and from the manual analysis of ontologies. A classification of such pitfalls according to the Structural, Functional and Usability-Profiling dimensions proposed in Gangemi et al. (2006) is also provided. Then, for each pitfall, we incorporate its value of importance level (critical, important, and minor) because not all the pitfalls are equally relevant and important. Next, we explain the internal architecture of OOPS! and describe the pitfalls detection methods used within the system. After that, an empirical analysis of the proposed catalogue carried out on 693 ontologies is presented. Then, we present the evaluation of the system based both on a survey of users' satisfaction and on evidence of the real use of the tool worldwide. After that, we review related works about ontology evaluation tools. Finally we draw the conclusions and provide future lines of work.

COMMON PITFALLS IN ONTOLOGY DEVELOPMENT

One of the most common approaches for evaluating ontologies is to have a checklist of typical errors that other developers have made before. Thus the developer checks the ontology being built against such a list, detects the