# Aspect-Oriented Framework for Ontology Modularization

Mario J. Lorenzo

Nova Southeastern University

July 2019

Contact: mario@mjlorenzo.com or ml2159@nova.edu

**Abstract**

Ontologies traditionally have served the purpose of representing knowledge by defining the relevant concepts and relationships for a particular domain. Today, ontologies also serve as the foundation of knowledge for AI and Cognitive systems. These ontologies can quickly scale in size and complexity requiring additional system resources and reducing reusability. The technique of Ontology Modularization attempts to segment the ontology into smaller reusable modules that improve the reusability of the ontology.

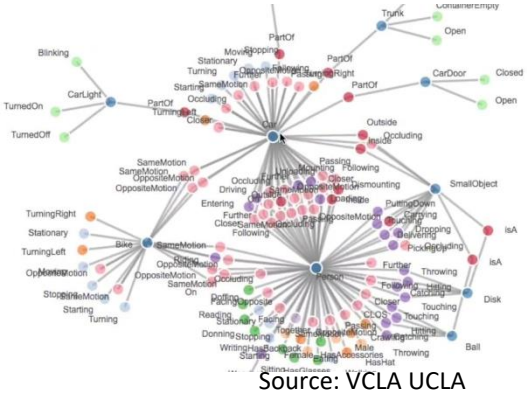
This paper proposes a new approach to modularizing an ontology using an Aspect-oriented design paradigm. Previous research produced complex meta-modeling methods and algorithms for module extraction that fail to improve reusability. Recent research has demonstrated the feasibility of an aspect-oriented approach through proof-of-concept.

This proposed research seeks to address the problems and limitations identified by previous researchers and demonstrated by defining Aspect support within OWL ontology language along with an Aspect weaving engine that improves modularity and reusability.

## Background: Ontologies, Concepts, and Relationships

An ontology defines the concepts and relationships for a particular domain or subject matter. Human subject matter experts who have a deep understanding for a given domain usually construct the ontology. The construction of the ontology produces a vocabulary of concepts and a set of relationships between pairs of concepts. The ontology can be represented as a graph where vertices represent individual concepts and edges represent relationships between concepts.

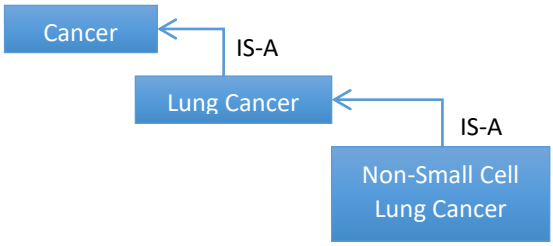
An ontology concept represents a well-defined idea within the domain. The concept may have multiple surface-forms or labels. This means that the same concept may occur in text documents with a different textual representation. For example, the concept of Cancer can occur in text as “malignant neoplasm”, “Carcinoma”, or simply as “cancer”. All of these surface-forms represent the same idea, the concept of “cancer”.



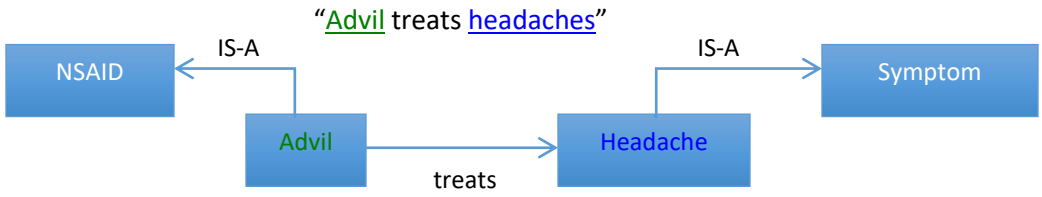
Source: VCLA UCLA

Another important aspect of the ontology is the relationships between concepts. An ontological relationship between concepts describes how two concepts are related to each other. These relationships can be taxonomic or semantic relationships. The taxonomic relationships define the hierarchy of concepts starting with generic or abstract concepts down to very specific or concrete subtypes of concepts.

For example, the concept “Non-Small Cell Lung Cancer” is a descendent or a subtype of the concept “Lung Cancer” and “Lung Cancer” is a descendent of “Cancer”. This kind of relationships is also known as subsumption or an “IS-A” relationship.




The ontology also defines other semantic relationships between concepts. These types of relationships may be very domain specific. For example, “treats” may be a semantic relationship that connects a treatment concept such as chemotherapy to a disease concept such as cancer. Here is another example:




## Application of Ontologies in AI Systems

Ontologies serve as a fundamental aspect to modern information extraction and cognitive systems. There has been an increased interest in these kinds of systems over the last decade due to the massive amount of unstructured documents available on the World Wide Web. Companies and research organizations are looking to extract additional value from their collection of data by

gaining semantic understanding of their text documents in order to support business processes and decision-making.

Ontologies also play an integral role in semantic search engines providing users with more meaningful search results and recommendations. This approach typically involves enriching the corpora of documents by identifying concepts and relationships from a related ontology. The search query at runtime undergoes a similar enrichment. Lastly, ontological matching of concepts between the search query and the documents can be used to produce semantically relevant documents.

## Ontology Modularization and State-of-the-Art

The construction of ontologies is a very complex and time-consuming task typically done by subject matter experts. This makes building ontologies a very expensive endeavor. A new discipline called Ontology engineering has surfaced to help bring a systemic process guided by modeling patterns and best practices. Much like Software Engineers, Ontology Engineers must work closely with subject matter experts in order to model an accurate and complete representation of the domain within the ontology representation. Unfortunately, most of the prominent ontologies in use across industries were developed prior to the formalization of the Ontology Engineering discipline, therefore resulting in large monolithic ontologies that cannot be easily decomposed into modules. This is in part due to the tight coupling and lack of cohesion that exists across the concept-relationships. The practice of modularization for ontology was borrowed from Software Engineering with the intent to improve modularity and therefore increase the reusability and applicability of an ontology to a wider set of domains and use cases. This provides significant

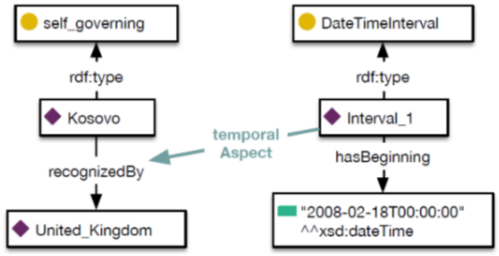
value by reducing duplicate ontology development efforts where significant overlap exists and therefore reducing project costs and duration.

## The Aspect-Oriented Paradigm

The Aspect-Oriented Programming (AOP) paradigm provides a method for encapsulating cross-cutting functionality that results from system decomposition occurring along the functional requirements dimension and therefore producing the scattering of code that addresses non-functional requirements throughout the system. This scattering ensues in dependencies that couple different functional aspects of the system and preventing modularity. An Aspect encapsulates the cross-cutting function away from the system using the principles of obliviousness and quantification to describe the points of concerns termed pointcuts (Kiczales 1997).

### An Ontology Modularization Example

The issues encountered in modular software design parallel the issues with Ontology Modularization. Ontologies are decomposed with a focus on a specific domain, but often require additional meta-information that cross-cuts the ontology. Examples of these cross-cutting concerns can include information that provides temporal context, such as the time-frame constraints for a relationship between concepts. The following example shows the country of Kosovo was recognized by the United Kingdom. The time-frame for this occurrence would ideally be encapsulated away from the core Countries ontology allowing for better reuse.




Source: (Schäfermeier 2014)

Another important example of a cross-cutting concern is the provenance of knowledge. This may include the institution and methodology that established a relationship or defined a new concept within the ontology. This sort of information is especially important within the healthcare domain where many institutions collaborate in contributing to open medical ontologies but differ in their standards for establishing medical facts (such as causal-relationships). As such, taking into account the provenance of a relationship between a disease and a treatment is important to a treatment recommendation system.

Another example for the need to capture these "non-functional" (non-domain) aspects of the ontology deal with understanding the temporal context for a given relationship. An example of this could be an ontology used within an industry that is regulated by a government entity. New regulations are introduced and repealed on a frequent basis and preserving when a regulatory obligation was introduced and when it expired is vital for understanding whether a business entity is and was under compliance at a given point-in-time.

These cross-cutting concerns in the ontology serve as an inhibitor to modularizing the ontology. Presently, an ontology engineer must use meta-modeling to represent these additional elements across concepts and relationships within the ontology. This introduces synthetic concepts and relationships not part of the domain, but required in order to satisfy the use case requirements (Schäfermeier 2018). Not only does this grow the size of the ontology but it also erodes its

reusability into other related domains or use cases. Compounding this problem are the restrictions imposed by some ontology languages, such as Owl, that are limited to binary relationships. This limitation prevents an ontology engineer from defining a ternary relationship for qualifying existing relationships with additional context. In the earlier example, the relationship "recognizedBy" between Kosovo and United Kingdom could be turned into a ternary relationship that links the concept of temporal occurrence. Although this approach does not resolve the modularity issues described, it would at least loosen the coupling between the concerns to a relationship. Instead, Owl would require additional concept class to be defined that includes the temporal property which in turn would be inherited by all the country concepts within the ontology. This produces very tight coupling and scattering of the temporal properties across the ontology.

## Problem Statement and Goal

Ontology modularization techniques today require complex metamodeling and expensive refactoring of existing ontologies. Improvements to the quality of modularity in turn improves the reusability of the ontology (d'Aquin et al 2007). Current modularization techniques fundamentally alter the structure of the original ontology by adding synthetic concepts in order to express cross-cutting concerns (Schäfermeier and Paschke 2014). This problem is further compounded by the ontology language constraints and lack of expressivity to model complex relationships (Doran et al 2008).

The goal of this research is to explore the applicability of the Aspect-Oriented Design paradigm to improving the quality of ontology modularity. The aim is to examine whether the use of "ontology aspects" as a method for encapsulating non-functional cross-cutting concerns

improves modularity of an ontology as compared to alternative modularization methods proposed by d'Aquin (2007) and Doran et al (2008, 2009). In completing this research, the successful outcome should demonstrate better modularity scores, measured with d'Aquin's (2007) ontology modularity metrics, using a newly proposed "ontology aspect" and weaving engine for the Owl ontology language. For additional background on Ontologies and the Modularization problem, see Appendix A & C.

## Significance, Relevance, and Brief Review of Literature

An ontology defines the concepts and relationships for a particular domain or subject matter. An ontology concept represents a well-defined idea within the domain. An ontological relationship between concepts describes how two concepts are related to each other (Maedche & Staab 2001). The recently formed discipline of Ontology Engineering focuses on bringing a systematic process guided by modeling patterns and best practices. Unfortunately, many prominent and prevalent ontologies, especially those in healthcare such as (Unified Medical Language System) UMLS, predate these best practices and are riddled with cross-cutting dependencies.

d'Aquin, Schlicht, Stuckenschmidt, and Sabou (2007) identify the problem of ontology modularization while experimenting with knowledge selection methods. They observed that large monolithic ontologies reduced reusability and increase maintenance efforts. d'Aquin et al describe the similarities between modularity problems encountered in Software Engineering and those of Ontology Engineering and propose algorithmic methods for modularizing an ontology. They identify the need for an evaluation criteria framework for quantitatively measuring the modularity quality of an ontology borrowing from the ideas put forth by Sant'Anna et al (2003) on the Software Metrics Suite. This work demonstrated that better modularity characteristics can be

achieved through a graph-transformation of the original ontology and a module extraction query method that produces a new sub-ontology. This approach, however, did not resolve the issue of reusability because it produces a new ontology, that requires dual maintenance, and moves the complexity from the ontology model into the module extraction query.

Doran, Palmisano, and Tamma (2008), also describe the problem of ontology modularity hindering reusability and propose a common framework for ontology extraction. Doran et al, address what they considered shortcuts in the ontology partitioning and module extraction methods proposed by d'Aquin. Doran approaches the problem from a reuse scenario and proposes a module selection querying framework and tooling called SOMET that performs selection, adaptation, and combination of the ontology using SPARQL graph query language. Their solution attempts to hide some of the complexity of module selection (querying) through a tool but do not provide a method for improving ontology model itself, instead it is able to produce a sub-ontology that includes a subset of the domain semantics with better modularity characteristics.

Schäfermeier and Paschke (2014, 2018) attribute the ontology modularization problem to dependencies caused by cross-cutting concerns that produce tight coupling and scattering of non-functional concepts and relationships throughout the ontology. Schäfermeier and Paschke identify the Aspect-Oriented paradigm as a potential solution for encapsulating these cross-cutting concerns into an ontology aspect. They cite several problems to address in future work including support for ontology languages that lack annotation over the ontology elements. Schäfermeier and Paschke also describe the need for a formal ontology pointcut language after attempting to use Doran's method leveraging the SPARQL query language, which resulted in unwieldy and unintuitive queries for complex pointcuts.

This proposed research will address the issues cited by Schäfermeier and Paschke through a new approach that features a weaving engine for the Owl ontology language and an aspect definition that features an expressive pointcut language for selecting ontological points of interests. This approach expects to demonstrate better modularity and reusability by providing ontology engineers with the Ontology Aspect construct used to encapsulate the cross-cutting concerns in the ontology. See appendix C for an illustrative example of an ontology modularization problem.

## Barriers and Issues

Improving the modularity of an existing monolithic ontology does not appear to have a universal approach (d'Aquin 2007), therefore all ontology modularization techniques are guided by requirement heuristics that can be expressed as a selection criteria used to segment the ontology into modules that produce a new ontology that produces only those relevant aspects represented by the domain requirements. Unfortunately, this approach produces a new ontology that must be maintained and kept in sync with the original ontology. The approach of ontology partitioning and module extraction methods proposed in literature by d'Aquin and Doran failed to address the underlying issue of reusability (Doran 2008).

Other approaches involve refactoring the ontology using meta-modeling to capture cross-cutting concerns. Unfortunately, this proves to be difficult due to limitations in the ontology language to model complex relationships such as ternary relationships. Instead, a workaround is used by adding additional properties and relationships to the concept class to represent these cross-cutting concerns. This however only reduces modularity and reuse and increases the complexity of the model. Ideally, these additional synthetic concepts and properties should be encapsulated

as an Aspect that can be dynamically weaved into the ontology depending on the use case (Schäfermeier 2018).

Defining an Aspect within the Owl ontology language has demonstrated feasibility but encountered several issues with the complexity of the pointcut syntax used and the reliance on annotations as a feature of the ontology language (Schäfermeier 2014). The approach did not actually perform weaving of the aspect but instead used annotations to expose points of interest for SPARQL queries used to perform selection (pointcut) over the ontology. Since Owl does not provide any facility to construct a sub-ontology based on a query, the Owl ontology is transformed to RDF and then using the RDF "construct" operation along with the SPARQL selection criteria representing the pointcut, a new ontology is produced.

Instead an ideal solution would provide an intuitive and expressive pointcut language capable of selection across concepts, relationships, and respective properties. In order to support ontology languages that do not feature annotations, a weaving engine is required to merge and produce an effective ontology with the desired aspects. This resulting ontology would be considered terminal (read-only) and not intended to be extended or modified. This ensures that changes occur through the defined Aspects or direct manipulation of the ontology model therefore increasing reusability reducing duplication and improving modularity through encapsulation of the cross-cutting concerns into aspects.

There does not currently appear to be a solution for targeting ontology axioms (i.e. assertions about the ontology) without employing second-order logic that would not be possible during the static-time pointcut selection. No solutions for this problem were found during literature

review. Instead, the proposed methods focus on Concepts, Relationships, and their properties (Schäfermeier 2018).

## Approach

This research proposes a new approach for addressing the Ontology Modularization and Reuse problems through the creation of an Aspect Ontology Framework for the Owl Ontology Language. The paradigm used by this new approach can be described as Aspect-Oriented Ontology Design (AOOD) or Aspect-Oriented Programming (AOP). See appendix B for additional background on AOP paradigm.

This proposed framework would include a specification defining the Aspect semantics within an ontology meta-model, defines an intuitive pointcut syntax for selection, an Owl Aspect weaving engine, and a tool allowing an ontology engineer (the user) an intuitive means for interacting with this framework. This tool would be incorporated into an existing ontology editor such as Protégé in the form of a plug-in.

The following describes the approach:

1. Define an Aspect ontology that will serve as a meta-model for describing Aspects within the Owl ontology language. This meta-model will feature an “Aspect” class that can be extended by a user-defined aspect. The approach would allow for separate smaller Owl ontologies to be defined and kept apart from the core ontology. These models would describe the cross-cutting concerns.
2. Define an Owl Aspect Pointcut Language that allows an ontology engineer to define the points-of-interest for an Aspect. This pointcut language should include the minimal expressiveness to perform selection allowing for targeting and restriction over class,

concept, relationship, and properties thereof. An evaluation of Description Logics (DL) languages (Krötzsch 2012) will be conducted to identify the feasibility of leveraging this specification as the underlying definition for the Aspect pointcut.

3. Define an Ontology Weaving engine and tool that will allow a domain expert to select from a collection of available Aspects of interest for an ontology reuse scenario. This engine should evaluate the pointcuts for each Aspect of interest and identify the ontological element targets across the ontology. These Ontological elements include concepts, relationships, and their properties. The engine will incrementally produce an internal "aspectized" version of the ontology by
    a. Inserting new concepts defined in the Aspect model
    b. Including the relationships definition into the "aspectized" ontology and traversing the ontology looking for pairs of concepts that satisfy the relationship.

    Once the engine completes the weaving process, it will produce the "aspectized" view of the ontology.
4. Define an Ontology Module extraction facility that can produce an "effective" ontology that includes only those domain relevant elements that address the reuse scenario.
    a. Using the proposed Aspect pointcut definition language, a module selection can occur based on the desired domain requirements.
    b. This module extraction facility can include extraction algorithms proposed d'Aquin and Doran in order to find the Least Common Ancestors for a vocabulary of concepts and relationships that the ontology engineer can choose from. Using this approach, the aspectized view of the ontology is traversed and an "effective" ontology module is produced that satisfy the selection algorithm criteria.

5. Measure the quality of the effective ontology produced by using d'Aquin's Ontology Modularization Evaluation Criteria Framework. This evaluation framework measures the characteristics of the ontology model including size (number of concepts and properties), redundancy, connectedness (number of relationships within and across modules), and distance between axioms in different modules. Using these metrics, a comparison between the "effective" ontology module produced by this new approach can be compared to the modules produced by d'Aquin(2007), Doran (2008), and Schäfermeier (2018).

   An exploration of additional quantitative metrics that can be included into the metrics suite that evaluates the semantic aspects of the ontology in addition to the structural aspects already covered by d'Aquins evaluation framework.

This proposed approach seeks to provide an improvement over Schäfermeier and Paschke's proof-of-concept because it formalizes a pointcut language, does not require transformation into RDF (an alternative ontology representation), and introduces an Ontology Weaving engine. This approach will apply the insights described by Schäfermeier and Paschke of leveraging an ontology meta-model to encapsulate aspects. The approach will also apply the insights from Doran et al (2008) of a common framework for ontology modularization and the idea of using a query language to perform selection over the ontology for module extraction and ontology segmentation. Lastly, the proposed approach will apply insights gained from the AspectJ compile-time weaving engine into the proposed Owl Aspect Weaving Engine.

The study format will follow a quantitative quasi-experimental approach. Validation of the proposed method will be conducted by leveraging an existing open Ontology to serve as the control during experimentation of different ontology modularization methods including the proposed

method. The only variable in the experiment will be the modularization methods under review. The results will be quantitatively evaluated using the established ontology modularity metrics by d'Aquin et al (2007).

This study will present the comparative results between the current modularization methods and the proposed methods. The study will additionally examine the strengths and weakness of each method and propose future work.

## Resources

Access to a large real world ontology used as a control to evaluate the various method is a requirement. This paper would use the ontologies provided by the National Library of Medicine called Snomed CT which covers a broad spectrum of Healthcare domain. Access to high-performance computing with 128gb of RAM memory. The Java Programming Language will be used to develop the executable parts of this study including the aspect weaving engine and tooling. The use of freely available Protégé Ontology Editor will be used to work with the ontologies.

# References

Schäfermeier, R., & Paschke, A. (2018). Aspect-Oriented Ontology Development. In *Synergies Between Knowledge Engineering and Software Engineering* (pp. 3-30). Springer, Cham.

Schäfermeier, R., & Paschke, A. (2014, September). Aspect-Oriented Ontologies: Dynamic Modularization Using Ontological Metamodeling. In *FOIS* (pp. 199-212).

Doran, P. (2009). *Ontology modularization: principles and practice* (Doctoral dissertation, University of Liverpool).

Doran, P., Palmisano, I., & Tamma, V. A. (2008). SOMET: Algorithm and Tool for SPARQL Based Ontology Module Extraction. *WoMO*, *348*.

d'Aquin, M., Schlicht, A., Stuckenschmidt, H., & Sabou, M. (2007, September). Ontology modularization for knowledge selection: Experiments and evaluations. In *International Conference on Database and Expert Systems Applications* (pp. 874-883). Springer, Berlin, Heidelberg.

Krötzsch, M., Simancik, F., & Horrocks, I. (2012). A description logic primer. *arXiv preprint arXiv:1201.4089*.

Sant'Anna, C., Garcia, A., Chavez, C., Lucena, C., & Von Staa, A. (2003, October). On the reuse and maintenance of aspect-oriented software: An assessment framework. In *Proceedings of Brazilian symposium on software engineering* (pp. 19-34).

Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J. M., & Irwin, J. (1997, June). Aspect-oriented programming. In *European conference on object-oriented programming* (pp. 220-242). Springer, Berlin, Heidelberg.

Maedche, A., & Staab, S. (2001). Ontology learning for the semantic web. IEEE Intelligent systems, 16(2), 72-79.