Software and Systems Modeling (2022) 21:1363–1387
https://doi.org/10.1007/s10270-021-00919-5

SPECIAL SECTION PAPER

Automated conceptual model clustering: a relator-centric approach

Giancarlo Guizzardi[1,2] · Tiago Prince Sales[1] · João Paulo A. Almeida[3] · Geert Poels[4]

Received: 18 March 2021 / Revised: 3 July 2021 / Accepted: 28 July 2021 / Published online: 15 September 2021

Abstract
In recent years, there has been a growing interest in the use of reference conceptual models to capture information about complex and sensitive business domains (e.g., finance, healthcare, space). These models play a fundamental role in different types of critical semantic interoperability tasks. Therefore, domain experts must be able to understand and reason with their content. In other words, these models need to be *cognitively tractable*. This paper contributes to this goal by proposing a model clustering technique that leverages on the rich semantics of ontology-driven conceptual models (ODCM). In particular, we propose a formal notion of *Relational Context* to guide the automated clusterization (or modular breakdown) of conceptual models. Such Relational Contexts capture all the information needed for understanding entities "qua players of roles" in the scope of an objectified (reified) relationship (relator). The paper also presents computational support for automating the identification of Relational Contexts and this modular breakdown procedure. Finally, we report the results of an empirical study assessing the cognitive effectiveness of this approach.

Keywords Ontology-driven conceptual modeling · Complexity management in conceptual modeling · Conceptual model clustering · OntoUML

1 Introduction

In recent years, there has been a growth in the use of reference conceptual models to capture information about complex and critical domains (e.g., [7,9]). As the complexity of these domains grows, often so does the size and complexity of the artifacts that represent them. Moreover, in sensitive domains (e.g., finance, healthcare), these models play a fundamental role in different types of critical semantic interoperability tasks including data integration [21], data interchange [9], and ontology-based data access [56]. Therefore, domain experts must be able to understand and accurately reason with the content of these models. However, the human capacity for processing unknown information is very limited, containing bottlenecks in visual short-term memory and causing problems to identify and hold stimuli [37]. For this reason, there is an evident need for developing adequate complexity management mechanisms for reference conceptual models.

One type of such complexity management mechanisms is conceptual model modularization or *Conceptual Model Clustering* (henceforth CMC) [1]. CMC is the process by which a model is fragmented into smaller interconnected parts [35], each of which can be more easily manipulated by a model user than the entire model. The greatest challenge in CMC is the process for *module extraction*, namely coming up with adequate criteria for dividing the model into modules that ease model understanding.

Traditionally, different techniques have been used for module extraction (e.g., [1,12,36]). However, almost the totality of these approaches addresses modularization in lan-

Communicated by Dominik Bork and Janis Grabis.

✉ Giancarlo Guizzardi
giancarlo.guizzardi@unibz.it; g.guizzardi@utwente.nl

Tiago Prince Sales
tiago.princesales@unibz.it

João Paulo A. Almeida
jpalmeida@ieee.org

Geert Poels
geert.poels@ugent.be

1 Conceptual and Cognitive Modeling Research Group (CORE), Free University of Bozen-Bolzano, Bolzano, Italy

2 Services & Cybersecurity Group, University of Twente, Enschede, The Netherlands

3 Ontology & Conceptual Modeling Research Group (NEMO), Federal University of Espírito Santo, Vitória, Brazil

4 UGent Business Informatics Research Group, Ghent University, Ghent, Belgium

guages that are *ontologically neutral* [25] such as UML, ER diagrams, or OWL.[1] While these languages may have a well-defined abstract syntax and a formal (logical) semantics, in general, they lack *ontological semantics*. Consequently, the modularization techniques developed for them rely on criteria that leverage almost exclusively on the syntactical properties of the models, typically, topological ones [51].

In contrast, ontology-driven conceptual modeling (ODCM) languages are systematically designed to conform to an underlying ontological theory. In particular, an ODCM language contains exactly the modeling primitives that are necessary to represent the ontological distinctions put forth by its underlying ontology. For example, as different ontological categories of types (e.g., Kinds, Mixins, Roles) play different roles with respect to their instances regarding issues such as classification (e.g., dynamic versus static) and identity, these distinctions should be explicitly represented by the language's constructs. ODCM approaches have enjoyed an increasing adoption by the Conceptual Modeling community as a number of independent results consistently show their benefits for improving the quality of conceptual models (e.g., [50]). An example of an ODCM language is OntoUML [25], whose primitives reflect the underlying UFO foundational ontology [25].

In this paper, we leverage the ontologically well-founded semantics of OntoUML to propose a formal approach for automated modularization in conceptual models. The proposed approach breaks down an OntoUML model into a number of *Relational Contexts*. Intuitively, Relational Contexts are modules that capture all the information needed for understanding entities *qua players of roles* in the scope of an objectified (reified) relationship (ontological speaking, the so-called *relators*).

As reported in [46], *Relators* and *Roles* are clearly the most used OntoUML constructs in conceptual models. This is unsurprising, given the strong adoption of OntoUML/UFO in business (organizational, social, and legal) domains [49], as well as the fact that in these realms the bulk of the domain knowledge is concentrated in relationships and roles. As argued in [26], specially in these realms, "we seldom interact with these entities *qua-themselves*, but we frequently conceive objects *qua-playing-certain-roles* in given 'contexts'... For example, most of our interactions with other human beings and, hence, our conceptualizations of these interactions are thought in terms of roles such as parent, employee, student, president, citizen, customer, etc. Analogously, when thinking about, for instance, cars, we think about them as means of transportation, insurable items, work-related resources, product offerings, etc. Moreover, we often conceive these 'contexts' as *relational* ones: marriages, employments, enrollments, and presidential mandates are themselves concrete 'object-like' entities that define a scope in which ordinary objects play complementary roles interacting with each other". This view is also defended by other authors such as [8], who go as far as to claim that "[r]oles are useful not only to model domains that include institutions and organizations. Rather, every object can be considered as an institution or an organization structured in roles". Also in social science and, in particular, in its sub-field called *Role Theory*, roles as patterns of *context-bound* behavior types are considered to be "one of the most important features of social life" [10].

The proposal advanced here is, thus, aimed at conceptual models in business (organizational, social, and legal) domains, which form the bulk of the Information Systems discipline. For models that are centered on taxonomic relations (e.g., product types, biological taxonomies), we recommend alternative complexity management techniques, in particular, the static *ontological views* as proposed in [17]. In fact, this paper can be seen as a companion to [17] and [27] in a general research program of defining ontology-driven complexity management theories, techniques, and tools. While in these two papers the focus is on *model recoding with ontology-design patterns*, and on *model abstraction*, respectively, here we propose the notion of relationship-centric conceptual model modularization (or clustering).

This paper is an extension of [30]. As in that original paper, we advance a formalization of the notion of *Relational Context* by leveraging on the theory of relators from UFO/OntoUML. We then use this notion to propose a strategy for relationship-centric modularization termed *Relator-Centric Clustering*. However, extending that work, we discuss here a full implementation of this strategy as a service part of an open-source software library for OntoUML models, as well as a plugin for the Visual Paradigm model engineering tool. Furthermore, in this paper, we report on the results of an empirical study conducted with expert conceptual modelers to evaluate the perceived cognitive effectiveness of our clustering strategy. In particular, in that experiment, our approach is compared with two other relevant approaches used in the literature of conceptual modeling. Finally, we present a much more elaborated analysis of related work: firstly, we review different strategies for complexity management of large symbolic models in the literature of traditional conceptual modeling, ontology engineering, and enterprise modeling; secondly, in the section discussing the experiment, we also provide an extensive analysis of these two competing approaches considered therein.

The remainder of the paper is organized as follows. Section 2 positions our work in reference to related efforts; Sect. 3 briefly presents the OntoUML language and some

[1] There is a long debate in philosophy regarding the ontological neutrality (or lack thereof) of formal languages. We simply mean here that they commit to a simple ontology of formal structures (e.g., that of set theory) in which sorts of types and relations are undifferentiated.

of the ontological notions underlying it; Sect. 4 presents the formal contributions of this paper. Firstly, it defines the notions of Ontological Views, Relational Contexts, and Modular Breakdown. This is done both formally, in terms of a precise definition of these notions, and intuitively by making use of a running example in the domain of Car Rental; in Sect. 5, we report on an implementation of this approach as a service and as a plug-in to a model-based OntoUML editor; Sect. 6 presents the empirical study, including: (i) the materials (description of the two alternative clustering approaches, as well as the alternative breakdowns of the aforementioned Car Rental model under these three competing approaches), (ii) the procedure used for conducting the experiment, (iii) the obtained results, (iv) a further analysis of qualitative responses, (v) the limitations of the experiment, and (vi) a discussion of the representativeness of the conceptual model that is used as a running example throughout the paper as well as in this experiment; finally, Sect. 7 presents our conclusions and outlines directions for future research.

2 Complexity management of conceptual models

The discipline of *complexity management of large conceptual models* (henceforth CM-CM) has been around for quite some time and has been represented in the literature by a series of different approaches and techniques. In fact, [51] claims that "one of the most challenging and long-standing goals in conceptual modeling... is to understand, comprehend, and work with very large conceptual schemas".

The challenge and importance of this discipline lie in the following. On the one hand, real information systems often have large and extremely complex conceptual models [51]. On the other hand, this complexity poses a serious additional challenge in the comprehension and, consequent, quality assurance of these models. For example, [44] reports on an empirical study conducted with a large and professionally constructed conceptual model.[2] In that study, the authors managed to show that the model contained 879 occurrences of error-prone structures (anti-patterns), 52.56% of which really introduced representation errors according to the creators of the model.

According to [51], the methods for CM-CM can be classified in three areas, namely *clustering*, *relevance*, and *summarization* methods. Clustering is about classifying the elements of a conceptual model into groups, or clusters, according to some criteria (e.g., a similarity function); relevance methods are about the application of ranking functions to the elements of a model in order to obtain ordered lists (i.e., a ranking) of model elements according to their perceived relevance in representing the domain at hand; finally, summarization is about producing a reduced version of a model consisting only of the elements that are judged to be of more relevance for representing the domain at hand. In clustering methods, the goal is to break down a model into fragments such that the sum of these fragments should be informationally equivalent to the whole (i.e., to the original model). In contrast, relevance and summarization methods (including *model abstraction*) aim to produce partial views of the original model at hand. In other words, while clustering methods have *lossless model transformations*, the latter classes of methods are based on *lossy transformations*.

Techniques for model clustering, relevance, and summarization have been developed for more than three decades in conceptual modeling, from earlier approaches such as [16] in the mid-1980s, to [1,12] in the second half of the 1990s, to [47,48] in the first decade of the 2000s, to recent approaches such as [11]. Recently, the problem of complexity management of large symbolic models has gained significant interest also in the areas of ontology engineering [2,6,13,14,20,33], enterprise models [34], and process models [53,55].

In enterprise modeling, in particular, in enterprise architecture modeling, this problem is always present given that, by their very nature, these models aim at putting together multiple layers (e.g., business, information system, and technology layers), multiple concerns (e.g., modeling the situation "as-is", the situation "to-be", and how to bridge the two), as well as multiple dimensions (e.g., a dimension of active entities, i.e., carriers of behavior; a dimension of behavior itself, i.e., occurrences of the behavior of these entities; a dimension of passive entities, i.e., entities that are created, terminated, changed, and manipulated by behavior). In this area, approaches like ArchiMate [34] define multiple *viewpoints*, which are archetypal "filters" over the complete model that select information over layers, concerns, and dimensions that are supposed to match particular aims (e.g., designing, deciding, informing), granularity (e.g., details, coherence, overview) and user roles (e.g., process manager, CEO, service designer, software developer, network administrator). However, unlike approaches for view extraction like [17], ArchiMate viewpoints are not designed to guarantee that the set of views is informationally equivalent to the original model.

In ontology engineering, this problem is also very salient, given that some of the models produced by the area are of significant size. An extreme case is the Foundational Model of Anatomy, which contains "75,000 classes and over 120,000 terms; over 2.1 million relationship instances from over 168

[2] We will refer to this model (henceforth termed the *MGIC* model [44]) in a few passages of this article. The MGIC model was developed by practitioners in the context of a Brazilian governmental project and for the domain of Ground Transportation Regulation. The model consists of 3,800 classes, 61 datatypes, 1,918 associations, 3,616 subtyping relations, 698 generalization sets, 865 attributes, i.e., navigable association ends.

relationship types".[3] However, even a foundational model such as SUMO contains 20,000 terms and 80,000 axioms.[4]

In this area, the approaches for complexity management vary considerably. A first concern there is how to separate these large theories in models that will form an *ontology network* (or ontology federation). For example, [14] propose the use of another model (in that case, an ArchiMate architectural model) to guide that process. Large ontologies can incorporate concerns from different *generality levels* ranging from domain-specific concerns (e.g., what kinds of elements are involved in a software service), to concerns that crosscut different domains (e.g., what is a service), to domain-independent ones (e.g., how to differentiate and characterize objects and events). Some approaches (e.g., [15]) use an understanding of ontologies at these different levels (i.e., foundational, core, and domain ontologies) as a strategy for model breakdown. Another example that takes a similar level-based approach for modularizing ontology networks is proposed in [41].

Most of the methods for complexity management of large ontologies, however, consider these models simply as logical theories and are oblivious to these distinctions about generality level. These methods are divided in approaches for model abstraction (known as *forgetting* [13]), clustering (called *ontology partitioning* [6,20]), as well as *ontology module extraction* [13]. A few approaches such as [13] propose to combine operations of module extraction and forgetting. Ontology partitioning is a lossless operation; module extraction and forgetting are lossy operations, both of which require from the user as input a set of model elements they are interested in (called a *seed signature* [13]). Module extraction is about producing a subset of the ontology for which answers for queries over the seed signature are the same as those obtained from the original model; forgetting is about eliminating from the ontology the terms provided in this list while preserving logical inferences involving the remaining elements [13].

None of the aforementioned approaches is fully automated. Approaches such as [14] assume the existence of additional supporting models (enterprise architecture models) and require an understanding of the mapping between these models and the model to be modularized; approaches such as [15] require a human interpretation of the non-consensual classification of ontologies in different generality levels. Finally, approaches for ontology module extraction and forgetting rely on the user input of seed signatures, which can be a problem in the case of large models. An exception in this sense is [2], which proposes an automated approach for recommending which ontology concepts can potentially be used as cluster seeds. This part of the approach is based on an algorithm that ranks the relevance of model elements (i.e., a relevance method). Finally, although module extraction is sometimes driven by the concern of identifying reusable fragments of the original model, this concern is itself strongly based on human heuristics and judgment [33].

A drawback that is common to the majority of existing methods in all these classes in the literature is that they are based on classic conceptual modeling notations (e.g., UML, ER) [51] and knowledge representation languages (e.g., OWL) and, as a consequence, they are constrained to rely almost exclusively on syntactic (mainly topological) or formal properties of the addressed models. These properties include *closeness* (a quantitative evaluation of the links among elements in the model) [19], *hierarchical distance* (length of the shortest relationship path between entities), *structural-connective distance* (elements are considered closer if they are neighbors in a mereological or subtyping structure), or *category distance* (elements are considered to be closer if one subtypes the other) [1]. For example, [12] proposes a (relevance) method based on the assumption that the number of attributes and relations characterizing an element in a model can be used as a (heuristic) measure of its relevance for that model. In the same spirit, [47,48] go as far as proposing PageRank-style algorithms to infer the relevance of elements in entity-relationship diagrams and RDF schemas (even ignoring the difference between association and subtyping relations). The problem with relying solely on these properties is that there is no guarantee that a model element satisfying some topological requirement (e.g., a node with more edges connected to it) by necessity represents the model's most important concepts. This is related to the work by [38,39], that while criticizing existing CM-CM methods, referred to it as *lack of cognitive justification*.

Variations of these ideas can also be found in more recent approaches such as the ones proposed in [2,6,11,20]. For example, [11] uses a genetic algorithm based on fitness functions that are topological in nature (e.g., the sum of relations within a model, the sum of relations across modules, number of modules, average number of elements per module, standard deviation of module size) [39]; [2] uses the number of relations (including subtyping, supertyping, and property relations) that a type is involved in to estimate its relevance, as well as indicators such as a notion of hierarchical similarity to calculate cohesion and coupling of candidate modules; [20] uses graph-partitioning heuristics for detecting *communities* (i.e., subset of nodes densely interconnected relatively to the rest of the graph); [6] also uses the number of direct relations between two nodes divided by the maximum number of global relationships shared by each of the two nodes with every other node in the graph to estimate the weight for their connection, and then uses these weights to generate

[3] http://sig.biostr.washington.edu/projects/fm/.

[4] http://www.ontologyportal.org/.

a breakdown that minimizes the weight of the connections crossing partitions.

The method proposed here is a type of clustering method. However, in contrast with all the aforementioned approaches, our proposal focuses on the *ontological semantics* [25] of the elements represented in a conceptual model. As previously discussed, the idea is to use a formal and ontological notion of *Relational Context* (see Sect. 4) as a clustering mechanism. In general strokes, a Relational Context is built in the following manner: starting from a focal reified relationship (*relator*), from there exploring the different relational types (typically *roles*) played by entities in the scope of that relationship, then finally navigating upwards the supertyping hierarchy of these relational types until reaching the fundamental types (*kinds*) that provide the essential properties and identity principles that characterize their instances.

This approach (detailed in Sect. 4) is only made possible because it is based on a non-classical CM language, namely the ODCM language OntoUML (briefly presented in Sect. 3). There are three CM–CM methods in the literature that are based on the same language, namely the approaches of (i) [17], (ii) [27], and (iii) [35,36]. The first method [17] is the one that is closer to the work presented here, since it is also a clustering method and, hence, a *lossless approach*. What is presented there is an approach for what the authors name *Model Recoding*. The method takes a conceptual model and produces a series of views constituted by ontological design patterns centered around general (as opposed to model specific) ontological constructs. So, for example, it groups all the *kinds* in the model in one view, all the *roles* played by instances of these kinds in a relational context in another view, etc. So, instead of breaking down the model into clusters that correspond to what one could intuitively call *sub-domains*, that approach breaks down the model in terms of general ontological categories. In contrast, the second [27] and third approaches [35,36] differ from the approach presented here since these are approaches for model summarization and hence *lossy approaches*. Finally, [35,36] also differs from our approach since it requires user input in selecting a set of entities in the model that are of particular relevance (a seed). Our approach, instead, is a fully automated one, which we argue is an important feature in methods dealing with large-scale models.

In summary, to the best of our knowledge, our approach is the only clustering method in the literature that is both fully automated and driven by ontological categories (ontological meta-types) of the elements represented in a model, as opposed to solely relying on topological properties of the graph, or on formal properties of the logical rendering of that model.

3 A Whirlwind introduction to UFO and OntoUML

OntoUML is a language whose meta-model has been designed to comply with the ontological distinctions and axiomatization of a theoretically well-grounded foundational ontology named UFO (Unified Foundational Ontology) [25,28,32]. UFO is an axiomatic formal theory based on contributions from Formal Ontology in Philosophy, Philosophical Logic, Cognitive Psychology, and Linguistics. OntoUML has been successfully employed in several industrial projects in different domains, such as petroleum and gas, digital journalism, complex digital media management, off-shore software engineering, telecommunications, retail product recommendation, and government [32]. A recent study shows that UFO is the second-most used foundational ontology in conceptual modeling and the one with the fastest adoption rate [49]. That study also shows that OntoUML is among the most used languages in ontology-driven conceptual modeling (together with UML, (E)ER, OWL, and BPMN). Moreover, empirical evidence shows that OntoUML significantly contributes to improving the quality of conceptual models without requiring an additional effort to producing them. For instance, the work of [50] reports on a modeling experiment conducted with 100 participants in two countries showing the advantages (in these respects) of OntoUML when compared to a classical conceptual modeling language (EER—Extended ER).

In the sequel, we briefly explain a selected subset of the ontological distinctions put forth by the Unified Foundational Ontology (UFO). We also show how these distinctions are represented by the modeling primitives of OntoUML (as a UML profile). For an in-depth discussion, philosophical justifications, formal characterization, and empirical support for these categories one should refer to [22,25].

Take a domain in reality restricted to endurants [25] (as opposed to events or occurrents). Central to this domain we will have object *Kinds*, i.e., the genuine fundamental types of objects that exist in this domain. We use the term "kind" in a strong technical sense, i.e., a type capturing essential properties of the things it classifies. In other words, the objects classified by that kind could not possibly exist without being of that specific kind.

Kinds tessellate the possible space of objects in that domain, i.e., all objects belong to **exactly one kind** and do so necessarily. Typical examples of kinds include 'Person', 'Organization', and 'Car' (see Fig. 1; stereotypes reflect the correspondence between the UML profile and UFO). We can, however, have other static subdivisions (or subtypes) of a kind. These are naturally termed *Subkinds*. As an example,

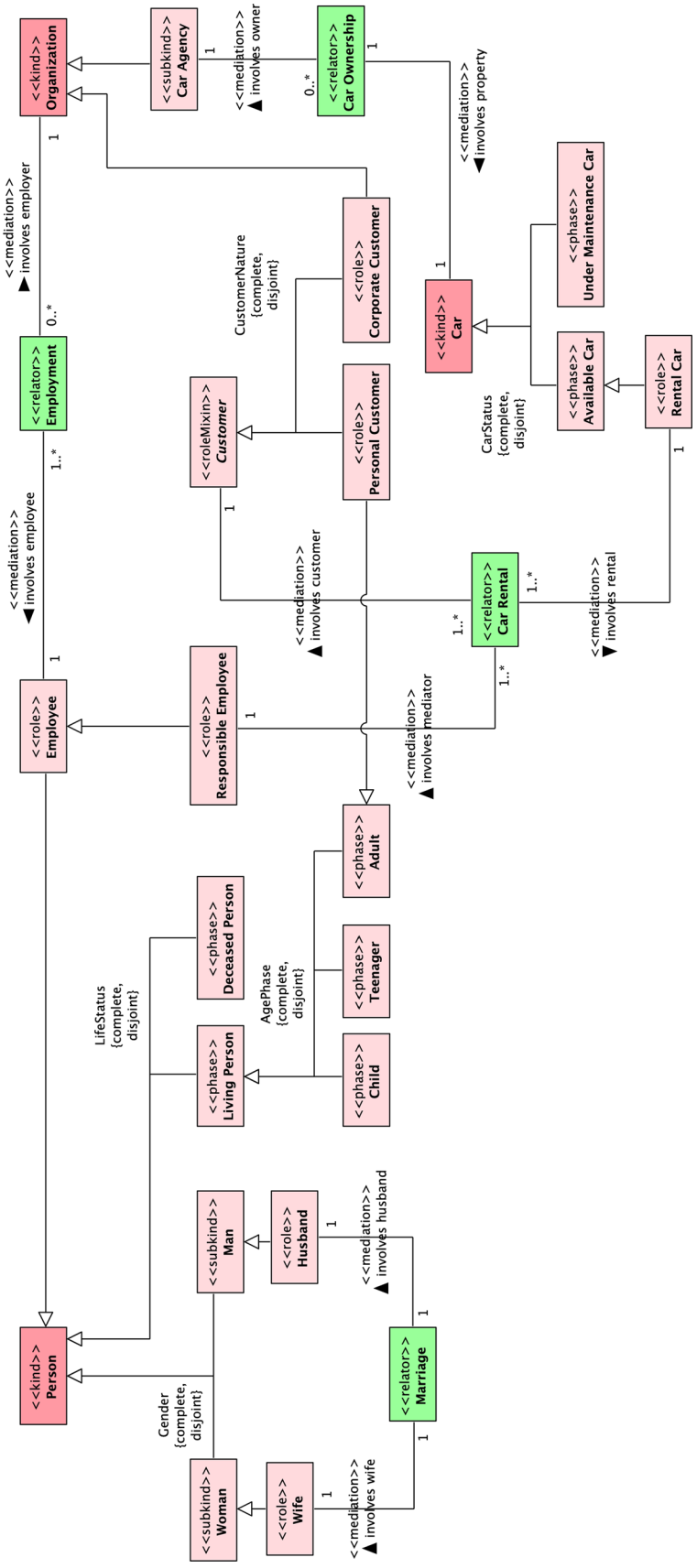

Fig. 1 A conceptual model in OntoUML in which relators are highlighted in green (color figure online)

the kind 'Person' can be specialized in the subkinds 'Man' and 'Woman' (in a conceptualization of gender that underlies many legacy information systems).

Object kinds and subkinds represent essential properties of objects (they are also termed rigid or static types [25,28]). We have, however, types that represent contingent or accidental properties of objects (termed anti-rigid types [25,28]). These include *Phases* (for example, in the way that 'being a living person' captures a cluster of contingent *intrinsic* properties of a person, or in the way that 'being a puppy' captures a cluster of contingent *intrinsic* properties of a dog) and *Roles* (for example, in the way that 'being a husband' captures a cluster of contingent *relational* properties of a man participating in a marriage, or that 'being a rental car' captures contingent *intrinsic* properties of a car participating in a car rental, see Fig. 1).

In other words, the difference between the contingent properties represented by a phase and a role is the following: phases represent properties that are intrinsic to entities (e.g., 'being a puppy' is being a dog that is in a particular developmental phase; 'being a living person' is being a person who has the intrinsic property of being alive; 'being an available car' is being a car that is functional and, hence, can be rented); roles, in contrast, represent properties that entities have in a relational context, i.e., contingent relational properties (e.g., 'being a husband' is to bear a number of commitments and claims towards a spouse in the scope of a marital relationship; 'being a student' is to bear a number of properties in the scope of an enrollment relationship with an educational institution).

Kinds, Subkinds, Phases, and Roles are categories of object *Sortals*. In the philosophical literature, a sortal is a type that provides a uniform principle of identity, persistence, and individuation for its instances [25]. To put it simply, a sortal is either a kind (e.g., 'Person') or a specialization of a kind (e.g., 'Student', 'Teenager', 'Woman'), i.e., it is either a type representing the essence of what things are or a sub-classification applied to the entities that "have that same type of essence".[5] Phases, but also typically subkinds, appear in OntoUML models forming (disjoint and complete, i.e., exhaustive) partitions following a *Dividing Principle* [54]. For example, in Fig. 1, we have the following phase partitions: the one including 'Living Person' and 'Deceased Person' (phase of Person' and according to a 'life status' dividing principle); the one including 'Child', 'Teenager' and 'Adult' (as phases of 'Living Person' and according to 'age phase'); the one including 'Available Car' and 'Under Maintenance Car' (as phases of 'Car' and according to a 'car status'). Since they are exclusively composed of phases, these are all dynamic partitions [54]. In this model, we also have a (static) subkind partition formed by the subkinds 'Man' and 'Woman', dividing 'Person' according to 'gender'.

Relators (or relationships in a particular technical sense [22,23]) represent clusters of relational properties that "hang together" by a nexus (provided by a relator kind). Moreover, relators (e.g., marriages, enrollments, presidential mandates, citizenships, but also car rentals, employments, and car ownerships, see Fig. 1) are full-fledged *Endurants*. In other words, entities that endure in time bearing their own essential and accidental properties and, hence, first-class entities that can change in a qualitative manner while maintaining their identity.[6]

As discussed in depth in [22,23], relators are the truthmakers of relational propositions, and relations (as classes of n-tuples) can be completely derived from relators [25,31]. For instance, it is 'the marriage' (as a complex relator composed of mutual commitments and claims) between 'John' and 'Mary' that makes true the proposition that "John is the husband of Mary". Relators are existentially dependent entities (e.g., the marriage between John and Mary can only exist if John and Mary exist) that bind together entities (their relata) by the so-called *mediation* relations—a particular type of existential dependence relation [25,31]. As discussed in depth in [22], like in the MERODE approach [45] (but here for ontological reasons), all domain relations in business models (the so-called material relations) can be represented exclusively by employing relators and these existential dependence relations (mediation).

Objects participate in relationships (relators) playing certain "roles" [25]. For instance, people play the role of a spouse in a marriage relationship; a person plays the role of president in a presidential mandate; a car plays the role of a rental car scope of a car rental, see Fig. 1. 'Spouse' and 'President' (but also typically student, teacher, pet) are examples of what we technically term a role in UFO, i.e., a relational contingent sortal (since these roles can only be played by entities of a unique given kind). There are, however, relational and contingent role-like types that can be played by entities of multiple kinds. An example is the role 'Customer' (which can be played by both people and organizations), see Fig. 1. We call these role-like types that classify entities of multiple kinds *RoleMixins* [25].

In general, types that represent properties shared by entities of multiple kinds are termed *Non-Sortals*. Non-Sortals

[5] In the model of Fig. 1, we use a color scheme in which object kinds are represented in red; other object types classifying instances of those kinds are represented in a lighter tone of that red; relators are represented in green.

[6] The model in Fig. 1 is a simplified model for this domain. A more detailed model could include cases of "relators mediating relators" (e.g., a car rental mediating a car ownership and an employment). The example avoids these to avoid deviating from the main focus of the discussion. Our formal definition of RCC (see Sect. 4.7), however, has no such a limitation, thus, addressing these cases that result in *nested contexts* (i.e., contexts including other contexts).

are always represented as *abstract types* as they can only be instantiated via instantiating sortal types, which typically specialize them [25,28]. For example, the only way an entity can be an instance of 'Customer' is either by being a 'Personal Customer'—in case this entity is a person, or a 'Corporate Customer'—in case it is an organization. Since all kinds are mutually disjoint, so are the types 'Personal Customer' and 'Corporate Customer', i.e., the RoleMixin 'Customer' is partitioned into these two sortals. In summary, all customers are either *of the kind* 'Person' or *of the kind* 'Organization', and the type 'Customer' represents (refactors) all the properties (including the relational ones) that are common to all customers of various kinds.

4 Views, relational contexts, and relator-centric clustering

In this section, we present a formal definition of our structure of ontological views, which are then used to formally define our notion of Relational Context (RC) and of Relator-Centric Clustering (RCC). Built over UFO's distinctions and for the OntoUML language, the approach presented here proposes rules to extract modules (clusters) from a conceptual model expressed in OntoUML.

4.1 Basic definitions

Let a Model M be a graph defined such that $M = \langle \Theta, \Sigma, \Phi \rangle$, where $\Theta = \{C_1..C_n\}$ is the (non-empty) set of concepts in the model M), $\Sigma = \{r_1..r_n\}$ is the set of directed relations in the model and $\Phi = \{gs_1..gs_n\}$ is the set of Generalization Sets in the model. Let CT (Concept Type), RT (Relation Type) and GST be domains of types such that CT = {SORTAL, NON-SORTAL, KIND, SUBKIND, PHASE, ROLE, ROLEMIXIN, RELATOR}, RT = {MEDIATION, SUBTYPING}, and GST = {PHASE-PARTITION, SUBKIND-PARTITION} . Now, let $<$ be partial order relation defined in CT in the following way to reflect the specializations in the taxonomy of types in UFO: KIND $<$ SORTAL, SUBKIND $<$ SORTAL, ROLE $<$ SORTAL, PHASE $<$ SORTAL, ROLEMIXIN $<$ NON-SORTAL. Finally, we define a number of auxiliary functions:

- $C(M)$ is a function that maps a model M to its associated set Θ;
- $R(M)$ is a function that maps a model M to its associated set Σ;
- $GS(M)$ is a function that maps a model M to its associated set Φ;
- $E\ HasType\ T$ is a relation connecting an element E to a type T in the following manner: if E is a concept, then $T \in CT$; if E is a relation then $T \in RT$, and if E is a generalization set, then $E \in GST$. We should also add that for any two types T and T' such that $T < T'$, if $E\ HasType\ T$ then $E\ HasType\ T'$;
- $t(r)$ is a function that maps a relation r to the target (destination) of that directed relation;
- $s(r)$ is the complementary function that maps a relation r to the source (origin) of that directed relation;
- $r \triangleright gs$ connects a relation r with a generalization set gs such that $r\ HasType$ SUBTYPING and: if $gs\ HasType$ PHASE-PARTITION then $s(r)\ HasType$ PHASE; if $gs\ HasType$ SUBKIND-PARTITION then $s(r)\ HasType$ SUBKIND. Moreover, for any two relations r_1 and r_2 such that $r_1 \triangleright gs$ and $r_2 \triangleright gs$, we have that $t(r_1) = t(r_2)$.

As expected, we have that for every model M and every relation such that $r \in R(M)$, we have that both $s(r) \in C(M)$ and $t(r) \in C(M)$. Moreover, every generalization set $gs \in GS(M)$ is such that all $r \triangleright gs$ implies that $r \in R(M)$.

For example, let M be the model depicted in Fig. 1. Then, $C(M)$ amounts to exactly the types represented there, while $R(M)$ includes all the mediation and UML subtyping relations. Finally, $GS(M)$ amounts to the generalization sets:

- *Gender*: a subkind partition comprising the subtyping relations connecting *Man* to *Person*, and *Woman* to *Person*;
- *AgePhase*: a phase partition on the developmental status of people, comprising the subtyping relations connecting *Child* to *Person*, *Teenager* to *Person*, and *Adult* to *Person*;
- *LifeStatus*: a phase partition comprising subtyping relations connecting *Living Person* to *Person*, and *Deceased Person* to *Person*;
- *OperationalStatus*: a phase partition comprising subtyping relations connecting *Available Car* to *Car*, and *Under Maintenance Car* to *Car*;
- *CustomerNature*: a phase partition comprising subtyping relations connecting *Personal Customer* to *Customer*, and *Corporate Customer* to *Customer*.

4.2 Direct subtyping and (indirect) subtyping

Let the functions $ST(C, C')$ (symbolizing that C is a direct subtype of C'), $ST*(C, C')$ (symbolizing that C is a subtype of C') and $IST*(C, C')$ (symbolizing that C is an improper subtype of C') be defined as follows:

- $ST(C, C')$ iff there is an r such that $r\ HasType$ SUBTYPING and $s(r) = C$ and $t(r) = C'$;
- $ST*(C, C')$ iff $ST(C, C')$ or there is a C'' such that $ST(C, C'')$ and $ST*(C'', C')$; and,
- $IST*(C, C')$ iff $ST*(C, C')$ or $C = C'$.

We also define the following auxiliary function:

- $K(C)$ mapping a sortal C to its unique supertyping KIND, i.e., we have that $K(C) = C'$ iff C' $HasType$ KIND and $IST*(C, C')$. (Notice that if C is a KIND, then $C = C'$.)

Again, using the model M of Fig. 1 as an example, we have that, for instance, $K(CarAgency) = Organization$ and $K(PersonalCustomer) = Person$.

4.3 View

Let M and M' be models as previously defined. It follows that M is a view of M' (symbolized as $V(M, M')$ iff:

- $C(M) \subseteq C(M')$ and
- $R(M) \subseteq R(M')$ and
- $GS(M) \subseteq GS(M')$.

Notice that, given our definition of a model, we have that all $r \in R(M)$ are such that $s(r) \in C(M)$ and $t(r) \in C(M)$, but also that for all $r \triangleright GS(M)$ we have that $r \in R(M)$. In other words, M is necessarily an original subgraph of M'.

The views we are ultimately interested in are the so-called Relational Contexts (RC), which will be defined in Sect. 4.6. Nevertheless, before we reach that, we need to establish two types of auxiliary views: Sortal Identity Paths and Non-Sortal Identity Paths. They are used later to support the definition of Relational Contexts.

4.4 Sortal identity path

We define that a view M is a Sortal Identity Path of M' based on a focus type c (symbolized as $SIP(M, M', c)$, where c $HasType$ SORTAL iff:

- $V(M, M')$ and
- $c' \in C(M)$ iff $(IST*(c, c')$ and $IST*(c', K(c))$ and
- $r \in R(M)$ iff r $HasType$ SUBTYPING and $s(r) \in C(M)$ and $t(r) \in C(M)$.

SIP is a generic parameterizable view definition that, given a sortal type c, provides a view that includes that type and all its supertypes (if any) until its corresponding kind is reached. Taking the model of Fig. 1 and picking, for instance, *Personal Customer* as focus type, the corresponding SIP would be constituted by the types that generalize *Personal Customer*, i.e., *Adult*, *Living Person*, and, finally, *Person*. Later, we use SIP to determine which supertypes should be included in a Relational Context, namely those that reveal the nature of the entities in the context.

4.5 Non-sortal identity paths

We define that the view M is a Non-Sortal Identity Paths of M' based on a focus type c (symbolized as $NSIP(M, M', c)$, where c $HasType$ NON-SORTAL iff:

- $V(M, M')$ and
- $c' \in C(M)$ iff $IST*(c', c)$ or (there is a c'' such that $IST*(c'', c)$ and $IST*(c'', c')$ and $IST*(c', K(c'')))$ and
- $r \in R(M)$ iff r $HasType$ SUBTYPING and $(s(r) \in C(M))$ and $(t(r) \in C(M)))$.

The intention of the $NSIP$ can be explained as follows. Take a non-sortal type c in the model M', this view should include: (i) c itself and all its non-sortal subtypes; (ii) the first sortal specializing c as well as the path from this sortal to the unique kind providing its *identity principle* [25]. Taking the model of Fig. 1 and picking, for instance, *Customer* as focus type, in the corresponding $NSIP$, we have, besides the rolemixin *Customer*, the sortals that immediately specialize it (the roles *Personal Customer* and *Corporate Customer*) as well as the supertypes of each of these sortals that are in the path between them and their kinds (*Person* and *Organization*, respectively, in this case).

4.6 Relational context

We define that M is a Relational Context of M' with focus on a relator type rel, where (rel $HasType$ RELATOR) (symbolized as $RC(M, M', rel)$) iff the following conditions are satisfied:

- $V(M, M')$;
- $c \in C(M)$ iff:
  - $c = rel$, or
  - there is a $r \in R(M)$ and $t(r) = c$, or
  - there is a view M'' and a $c' \in C(M)$ such that $(SIP(M'', M', c')$ or $NSIP(M'', M', c'))$ and $c \in C(M'')$, or
  - there is a $gs \in GS(M)$ and a $r \triangleright gs$ and $s(r) = c$, or
- $r \in R(M)$ iff:
  - $(r$ $HasType$ MEDIATION and $s(r) \in C(M))$ or
  - $(r$ $HasType$ SUBTYPING and $s(r) \in C(M))$ and $((t(r)$ $HasType$ RELATOR) or $t(r) \in C(M))$, or
  - there is a $gs \in GS(M)$ such that $r \triangleright gs$
- $gs \in GS(M)$ iff:
  - gs $HasType$ PHASE-PARTITION and there is an r such that $r \triangleright gs$ and $r \in R(M)$, or

– $gs\ HasType$ SUBKIND-PARTITION and for all r such that $r \triangleright gs$ then $r \in R(M)$.

Now, this definition can benefit from some unpacking. The Relational Context (RC) starts by (naturally) including the focal relator rel ($c = rel$). In addition, it includes all types that are connected by that relator via MEDIATION relations (henceforth, *mediated types*) (($r \in R(M)$ and $t(r) = c$) and ($r\ HasType$ MEDIATION and $s(r) \in C(M)$)). For example, if we take the relator *Car Rental* as focus, the corresponding RC would also include the types of entities that are bound by instances of *Car Rental* in that context, i.e., *Customer* and *Rental Car*.

Furthermore, this RC should include in this context all the types going from these mediated types to their respective kinds. The rationale here is that in order to understand the nature of the entities connected by instances of the relator at hand, one must understand what kinds of things those entities *essentially* are, i.e., what sort of *principle of identity* they obey. In case any of these mediated types c' is a sortal, then the RC will include all types in its SIP ($c' \in C(M)$ and there is and a view M'' such that $SIP(M'', M', c')$ and $c \in C(M'')$). So, in this example, for the sortal type *Rental Car*, it would include also the types *Available Car* and *Car*. In contrast, if any of the mediated types is a Non-Sortal, then the relational context will include all types in its $NSIP$ ($c' \in C(M)$ and there is a view M'' such that $NSIP(M'', M', c')$ and $c \in C(M'')$). The rationale here is analogous. However, since different instances of a non-sortal might take their identities from different kinds, in order to understand that context, we need to include all the information in the identity path between that non-sortal mediated type and the relevant kinds. For instance, for a *Car Rental* Relational Context, we need to understand the notion of *Customer* and, in order to understand this notion we have to understand the notions of *Personal Customer* and *Corporate Customer*. Finally, in order to understand the latter, we need to understand *Organizations*, and to understand the former, the notions of *Adult*, *Living Person* and *Person*. After all, instances of *Personal Customer* are adult living people.

Besides the types in SIP and $NSIP$ of mediated types, the Relational Context should also include all types that appear in phase partitions standing in the path between a mediated type and its identity supplier (i.e., its associated kind). The idea is that these types offer a contrast background that helps in the clarification of the semantics of the types in these paths. For example, in the Car Rental context, in order to understand that personal customers must be living adults, it is important to understand that they cannot be other alternatives of instances of *Person*, namely living children, living teenagers, as well as deceased person. In particular, given the anti-rigidity of these types (phases), all instances of living person can cease to be so, thus, becoming deceased people, in which case they can no longer play the role of *Personal Customer*. Formally, if one of the subtyping relations in a $(N)SIP$ is part of a phase partition, then that phase partition generalization set is included in the view ($gs\ HasType$ PHASE-PARTITION and there is an r such that $r \triangleright gs$ and $r \in R(M)$). Additionally, all other types that share the common supertype in that generalization set are also included in the view (there is a $gs \in GS(M)$ and a $r \triangleright gs$ and $s(r) = c$), and so are all these supertyping relations in that same generalization set ($r\ HasType$ SUBTYPING and $t(r) \in C(M)$ and (there is a gs such that $gs \in GS(M)$ and $r \triangleright gs$)). Notice that subkind partitions are only included (a posteriori) if all subtyping relations comprising it are already included in the view (e.g., *gender* in an RC with *Car Rental* as the focus).

Furthermore, we include in a relational context all subtyping relations involving two types included in that view ($r\ HasType$ SUBTYPING and $s(r) \in C(M)$ and $t(r) \in C(M)$). Finally, we include all supertypes of relators already included in the view ($r\ HasType$ SUBTYPING and $s(r) \in C(M)$ and $t(r)\ HasType$ RELATOR). This is because a subtype inherits all the properties of its supertypes, and thus to understand the context of a sub-relator we must understand the general notion (e.g., to understand 'foreign marriage' as a 'marriage' recognized abroad, we must understand 'marriage' as a relation binding spouses).

4.7 Relator-centric clustering

We are now in position to define the notion of a *Relator-Centric Clustering*:

> A **Relator-Centric Clustering** of a model M is a set of views symbolized as $RCC(M) = \{M_1..M_n\}$ such that for every $M_i \in RCC(M)$ there is a type rel such that $rel \in C(M)$ and $RC(M_i, M, rel)$.

Figure 2 depicts the application of this notion of RCC to the model of Fig. 1. Here we represent each Relational Context using UML packages and name these packages with the homonymous focal relator. As one can observe, the original model can be broken down into four contexts, namely: the *Car Rental*, the *Marriage*, the *Car Ownership*, and the *Employment* contexts. Each of these modules contains a view of the original model with all the information required to understand each of the contexts.

The *Car Rental* RC shows the roles (and role mixin) directly mediated by the *Car Rental* relator (*Responsible Employee*, *Rental Car*, *Customer*). The kinds involved are made explicit: *Person*, *Car* and *Organization* (when playing the role of *Corporate Customer*). Important business rules the model imposes on a *Car Rental* are revealed: only an *Adult* (a *Living Person*) can rent a car, and only a car that is in the *Available Car* phase can be rented. A similar observa-

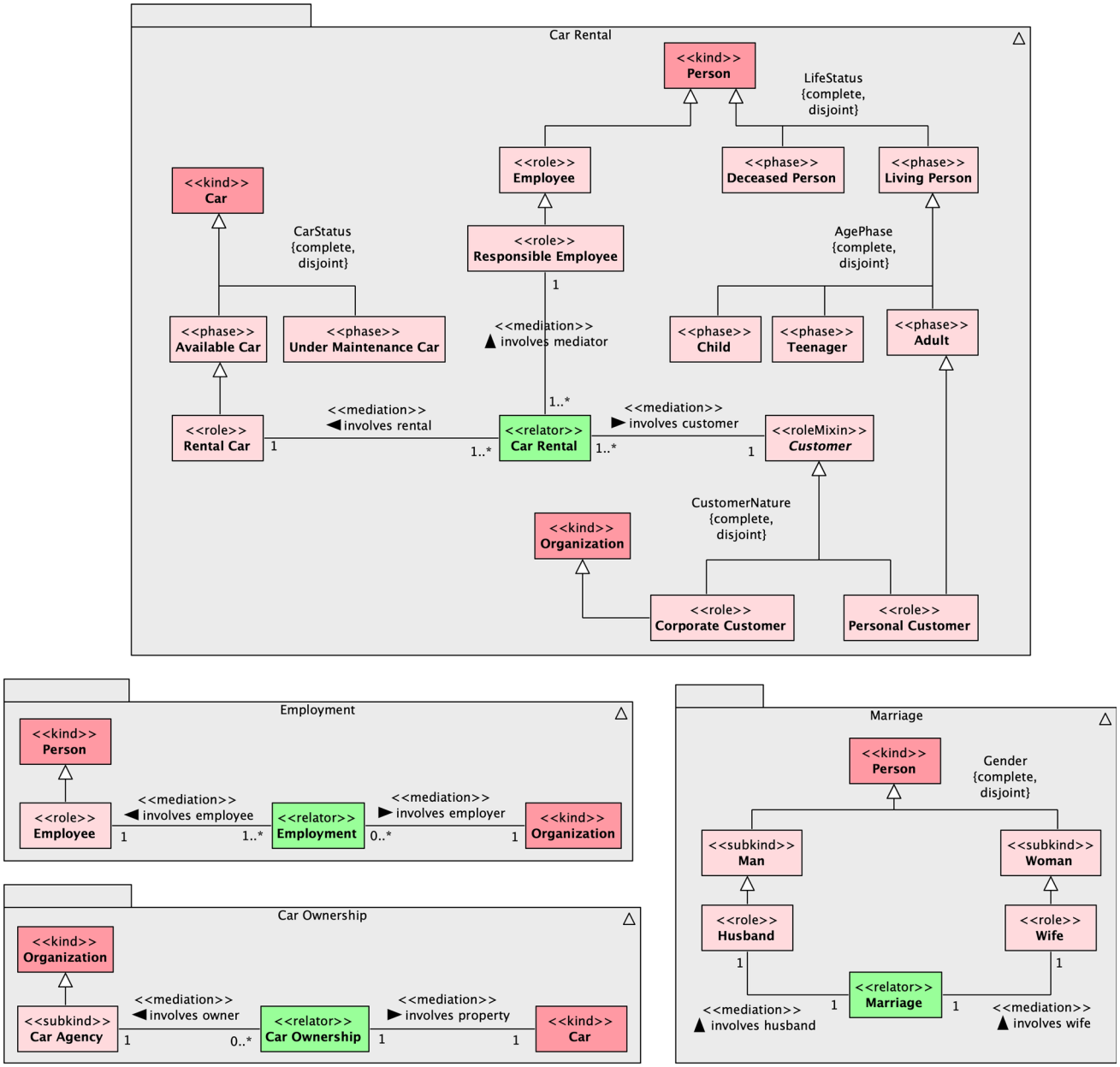

Fig. 2 An RCC for the model of Fig. 1 organized as (Onto)UML packages

tion can be made for the *Marriage* RC, as it reveals that the original model reflects a heteronormative setting and with *gender* in static classification. Finally, the *Car Ownership* and the *Employment* RCs are examples of simpler views, as the path from directly mediated entities to the involved kinds is short.

5 Tool support

We implemented our relator-centric clustering algorithm in JavaScript as a service within `ontouml-js`[7], an open-source software library for OntoUML. Maintained by the Conceptual and Cognitive Modeling Research Group (CORE),[8] `ontouml-js` provides an Application Programming Interface (API) for developers to programmatically create, navi-

[7] `ontouml-js` is distributed via NPM (https://www.npmjs.com/package/ontouml-js) and its source code is available at https://purl.org/krdb-core/ontouml-js.

[8] https://www.inf.unibz.it/krdb/core/.

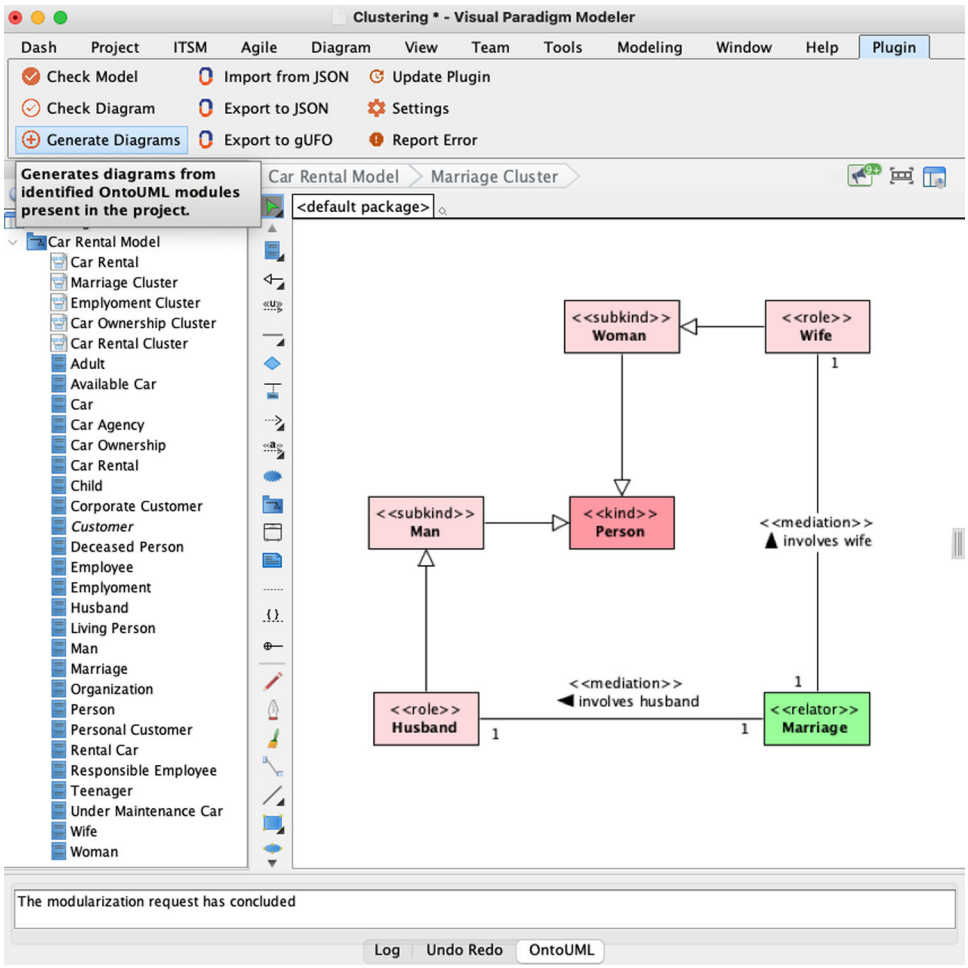

Fig. 3 A screenshot of the OntoUML plugin for Visual Paradigm showcasing a cluster generated using the relator-centric clustering algorithm

gate, and query OntoUML models, as well as a number of services that consume these models and can be explored in Computer-Aided Software Engineering (CASE) tools. The current version of the library provides the following model-based services:

- **Syntax verification**: a service that parses an OntoUML model and verifies its compliance with OntoUML's syntactical rules (see [28] for more details);
- **Transformation to OWL (ontouml2gufo)**: a parameterized service that generates OWL specifications compliant with gUFO [5], the reference implementation of the Unified Foundational Ontology (UFO) in the Web Ontology Language (OWL) [52];
- **Relator-centric clustering**: a service that implements the algorithm described in this paper. It receives an OntoUML model as an input and returns a set of diagrams, each of which contains one module produced by the algorithm.

To support OntoUML modelers and make the services implemented in `ontouml-js` available to them, CORE developed an open-source OntoUML plugin[9] for Visual Paradigm.[10] This plugin, which is implemented in Java, provides basic OntoUML modeling capabilities to the tool, such as the language's collection of stereotypes. It also acts as a bridge between Visual Paradigm and a web server that exposes `ontouml-js` services.[11] The communication between the plugin and the server is done by converting OntoUML models in Visual Paradigm into JSON objects that adhere to the structure defined by the `ontouml-schema`.[12] The `ontouml-schema` format is defined using JSON Schema [42], a vocabulary that allows one to annotate and automatically validate JSON documents.

[9] See source code at https://purl.org/krdb-core/ontouml-plugin.

[10] Visual Paradigm is a multi-platform desktop CASE tool that supports, among others, UML modeling, and whose functionalities can be extended via plugins written in Java. For more information, please refer to https://www.visual-paradigm.com/.

[11] See source code at https://purl.org/krdb-core/ontouml-server.

[12] See https://purl.org/krdb-core/ontouml-schema, an application-neutral reference format for serializing and exchanging OntoUML models and diagrams.

Figure 3 contains a screenshot of the OntoUML plugin for Visual Paradigm. The menu at the top includes buttons to access the functionalities of the plugin; the one to generate the relator-centric clustering is highlighted ("Generate Diagrams"). On the Model Explorer panel on the left, the classes composing our running example can be seen, along with the diagrams generated by the clustering algorithm (namely "Car Ownership Cluster", "Car Rental Cluster", "Employment Cluster", and "Marriage Cluster"). The selected diagram is shown in the center/right of the screen (" Marriage Cluster"). The diagrammatic disposition of the elements of this cluster in Fig. 3 differs from that of Fig. 2. That is because our relator-centric clustering algorithm only defines the elements that should be included in a cluster, not how they are to be laid out in a diagram. The diagram layout in Fig. 3 was automatically arranged by Visual Paradigm, while that of Fig. 2 was manually arranged.

6 Empirical evaluation

This section describes the empirical evaluation we have conducted to contrast the perceived effectiveness of the clustering obtained with the relator-centric approach with those clusterings obtained with ontologically-neutral (topological) approaches, more specifically those of Akoka and Comyn-Wattiau [1] and Castano et al. [12].

This empirical evaluation took the form of a study with conceptual modeling experts in which they were asked to rank the alternative modularizations of a model according to their preference. We first present the materials used in the study, including how we derived the alternative modularizations. Next, the empirical study procedure is described. Finally, we present the results and insights obtained from analyzing the rankings and their motivation, and discuss the study's limitations. We also provide a justification for the choice of conceptual model in the experiment as representative of the models in the application scope of our approach.

6.1 Materials

We have prepared the clusterization of the model presented in Fig. 1 according to the relator-centric approach and the other two approaches, which, not unlike ours, can be applied automatically. In the sequel, we present briefly the selected model break-down strategies and present the resulting diagrams that were shown to the subjects of the experiment.

6.1.1 The approach of Akoka and Comyn-Wattiau

Akoka and Comyn-Wattiau's [1] is a top-down clustering approach originally intended for EER diagrams, but which can straightforwardly be translated to be used in UML diagrams. It starts from a single cluster, and at each iteration, selects the model element that is considered the "farthest" from its current cluster. This element is moved to a separate cluster, and the model is then reorganized considering the newly formed cluster. Each element that is more distant to its own cluster than to another cluster is transferred to the cluster that is the closest to it. The improvement in the quality of the clustering is assessed in each iteration. The number of clusters is selected by observing when improvement from one iteration to the next drops.

This approach of Akoka and Comyn-Wattiau's can work with different distance functions. The ones proposed in the paper include: (i) visual distance; (ii) hierarchical distance; and (iii) cohesive distance.

The so-called *visual distance*, despite its name, does not take into account the actual positioning of elements in a diagram; instead, the distance between two elements is defined as the length of the shortest path between them, with each relationship or generalization in the model counting as a segment of length 1 (e.g., in Fig. 1, the distance between RentalCar and CarAgency is 4, along the path RentalCar–AvailableCar–Car–CarOwnership–CarAgency).

The *hierarchical distance*, in its turn, takes into account the cardinality of relationships. "The distance between entities linked by a 1:N relationship is equal to 1, whereas the distance is equal to 2 for M:N relationships. This is justified by the fact that any M:N relationship can always be translated into two different 1:N relationships." [1] Since we assume here models with reified relationships,[13] applying visual distance or hierarchical distance results in the same clusterization.

Finally, the so-called *cohesive distance* assigns different lengths to different kinds of segments. In particular, relationships are considered 10 to 100 times lengthier than generalizations. This favors the grouping of taxonomic structures, but performs very poorly in the presence of multiple inheritance (e.g., the path from Wife to Organization has length 80, while the path from Employment to Organization has length 1,000). Since multiple inheritance is not uncommon in conceptual models and is present in our test model, we have not employed the *cohesive distance* as it would result in poor clustering.

In summary, for the aforementioned reasons, we have opted for the so-called visual distance in applying Akoka and Comyn-Wattiau's method. The five resulting clusters are shown in Fig. 4. We have selected to emphasize the relator classes, as the notation employed in [1] suggests a double line for weak entities (and relators are relational weak entities [25]). Note that all diagrams, including our relator-centric approach, were prepared in plain UML (thus without the

[13] The ontological rules behind OntoUML yields models in which all domain relations are reified as relators [22].

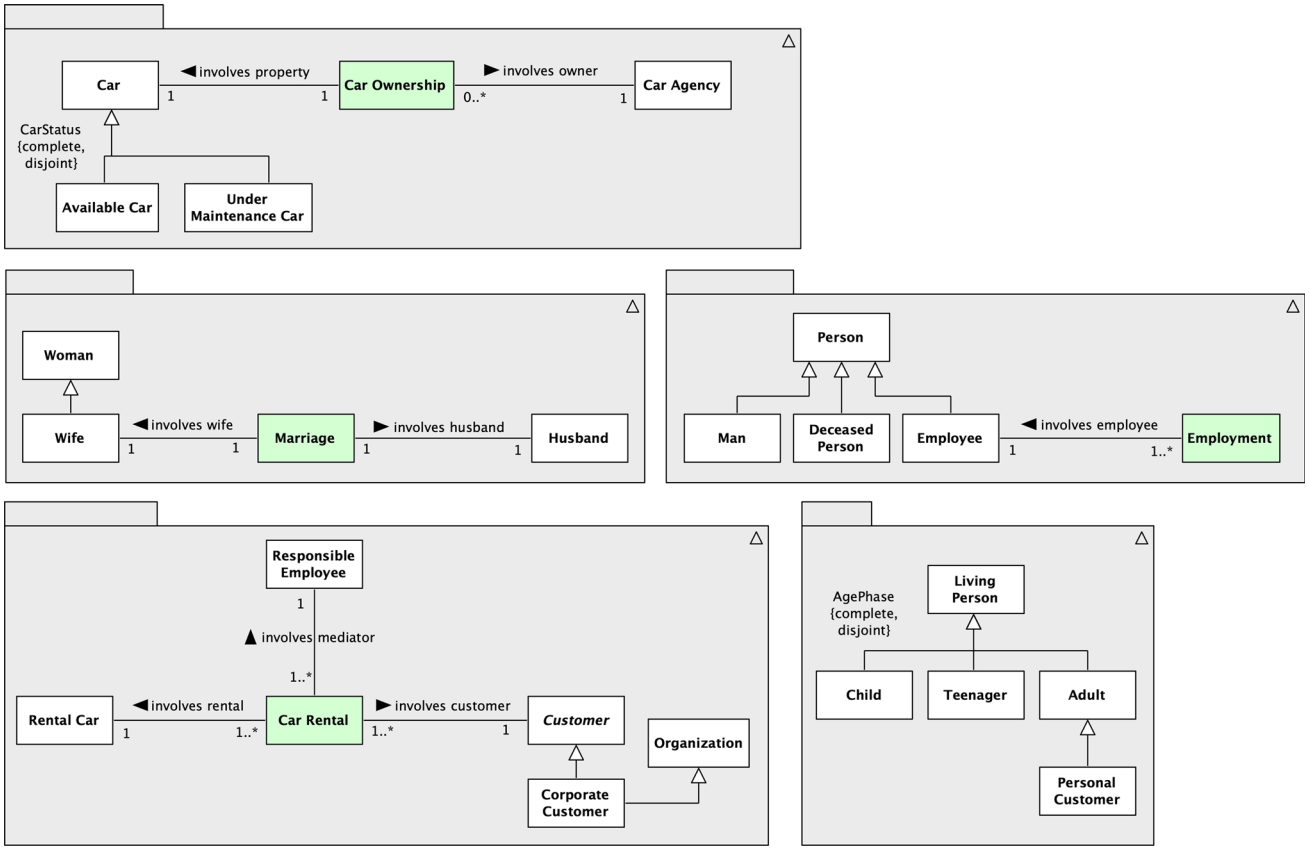

Fig. 4 A clusterization for the model of Fig. 1 following Akoka and Comyn-Wattiau's approach

OntoUML stereotypes) to allow the inclusion of UML users in general in the study, and to avoid introducing bias in the interpretation of the diagrams by OntoUML users. They are shown here in the same way as they were presented for the experiment subjects.

6.1.2 The approach of Castano et al.

Castano et al. [12] propose a number of conceptual schema analysis techniques, including a bottom-up clustering approach. Their objective is to obtain highly cohesive sub-schemas. They introduce the notions of *affinity* and *closeness* between model elements, and build up a definition of coupling between clusters in terms of these notions. Affinity is calculated with the help of a thesaurus (not specified in the work) and is based on the labels for entities. Synonyms, hypernyms/hyponyms and term relatedness are considered. Closeness considers the type and number of links among elements. A link strength is assigned to each type of link. A strength of 0.4 is assigned to relationships, a strength of 0.6 to generalizations, and a strength of 1.0 to attribute links (the latter are not considered in the present work).

As discussed in [12], the procedure for clustering consists in the merging of the closest pair of clusters. The procedure stops when the desired number of clusters is obtained. The authors suggest to use the number of "representative elements" as the number of clusters. They provide a procedure for the identification of "representative elements" by calculating an element's "relevance", summing the number of links weighted by strength. They suggest to select those ranked above average in terms of "relevance" as "representative elements". There are six of those in the test model: Person, LivingPerson, Car, Customer, Organization, and Employee. The authors parameterize their model with weights for affinity and closeness in the definition of coupling. The approach was tested by the authors in [12] with high precedence for closeness, which was given a 90% weight in the calculation of coupling. Because of the small effect of affinity and the lack of a standardized terminological resource to calculate it, we have opted to apply their approach by considering closeness alone in the calculation of coupling. The resulting clusterization can be seen in Fig. 5. Elements emphasized correspond to the "representative elements".

For completeness, Fig. 6 shows the relator-centric clusterization as presented to the subjects of the experiment.

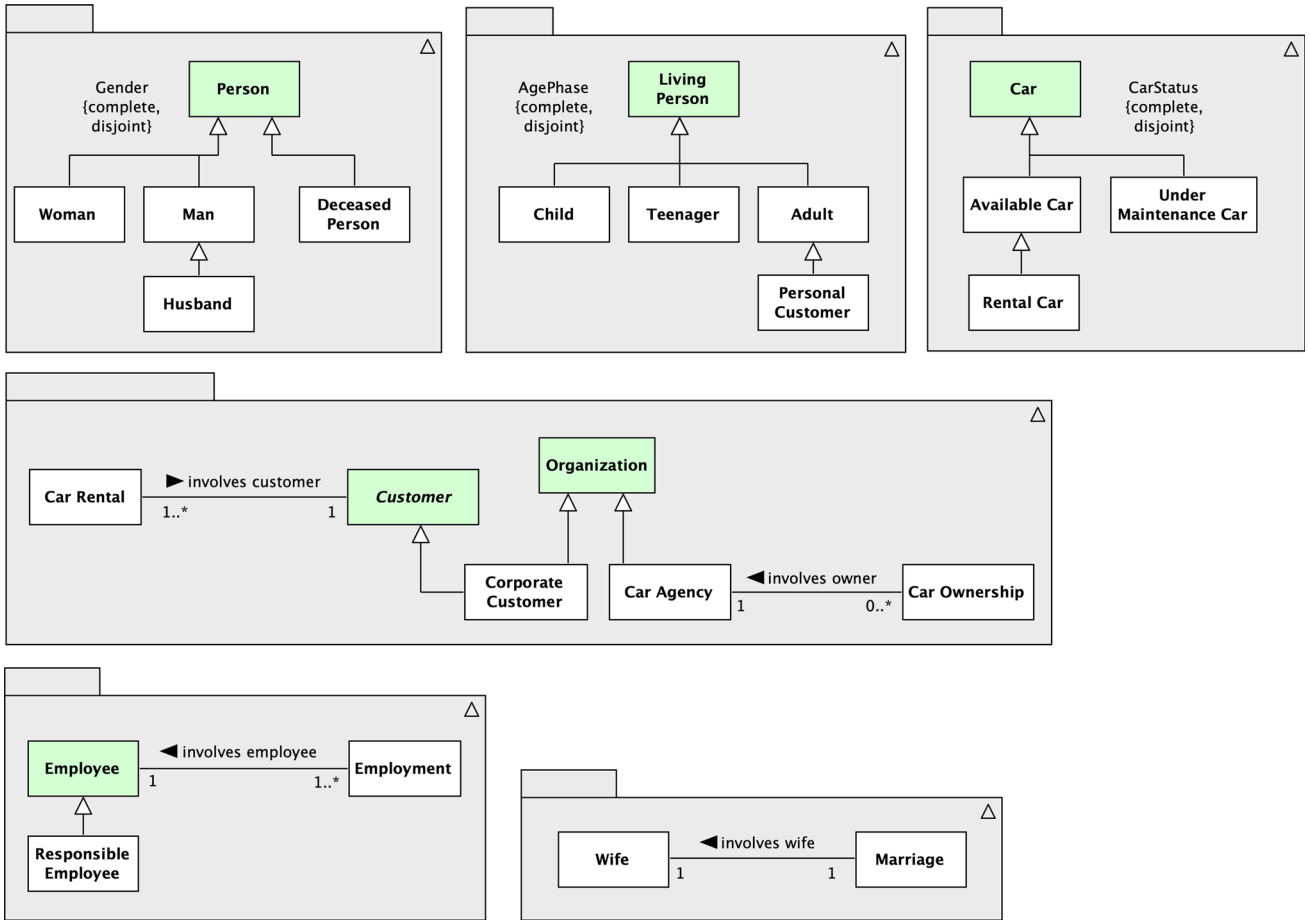

Fig. 5 A clusterization for the model of Fig. 1 following Castano et al.'s approach

6.2 Procedure

We sent out 160 invitations to participate in the empirical study. These invitations were sent to contacts in the personal networks of the researchers that were considered to have the required expertise in conceptual modeling to participate. We reckoned that subjects should have knowledge of UML (as the models were expressed in this language) and that they be used to working with conceptual models, assuming that sufficient exposure to conceptual models has made them aware of the need for complexity management. The invitations sent did not reveal the purpose of the study; they only mentioned that it was a study on complexity management of conceptual models and that modularization was investigated as a model management technique. It was by no means disclosed that one of the modularization alternatives was proposed by the researchers. The invitation made clear that participating included giving consent to participate, while emphasizing that the responses would be treated anonymously. In fact, there was no means for the researchers to identify who participated and to associate responses to the identities of the subjects.

The invitations contained a link to an online survey prepared with Google Forms. The survey first presented the subjects some general questions that could be used as controls for their level of expertise. Next, there was a description of the task requiring them to rank the three alternative modularizations according to their preference, taking into account the purpose of modularization as a model management technique. Upon this explanation, the complete model was shown, followed by the three alternative sets of model modules presented in the previous sub-section. The Google Forms survey referred to these sets of model modules with meaningless names, in order not to reveal the modularization strategies used to obtain them.

Each invitation sent out contained a link to one of three different versions of the survey, each of which with a different order of presentation of the three alternatives. This was made to rule out bias in the order of presentation. Subjects could always go back and forth to any of the alternative sets of modules and to the complete model, such that they could compare the alternatives and judge their effectiveness in managing the complexity of the model.

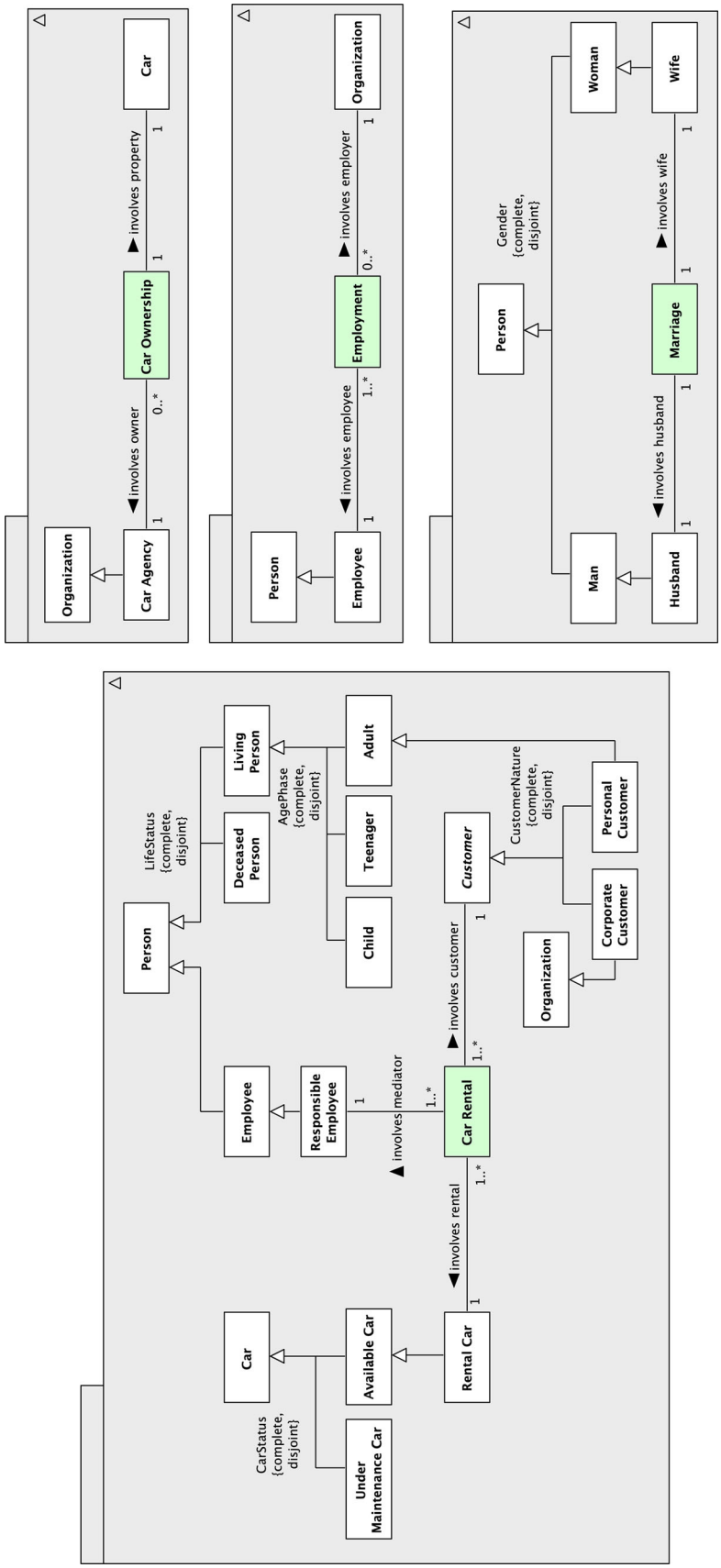

Fig. 6 The relator-centric clusterization for the model of Fig. 1

Table 1 Subjects' rankings of the three modularization alternatives, where 1 means the most preferred alternative, and 3 the least preferred

Approach\Ranking All experts—42 subjects	**1 (best)**	**2**	**3 (worst)**
Relator-centric	29	6	7
Akoka and Comyn-Wattiau	8	28	6
Castano et al.	5	8	29
Approach\Ranking OntoUML subgroup—28 subjects	**1 (best)**	**2**	**3 (worst)**
Relator-centric	19	4	5
Akoka and Comyn-Wattiau	6	19	3
Castano et al.	3	5	20
Approach\Ranking Non-OntoUML subgroup—14 subjects	**1 (best)**	**2**	**3 (worst)**
Relator-centric	10	2	2
Akoka and Comyn-Wattiau	2	9	3
Castano et al.	2	3	9

The survey ended with a response sheet where subjects could rank the three alternatives and optionally motivate their ranking. These qualitative data were collected to find out the reasoning of our subjects when ranking the alternatives. Finally, the response sheet presented the subjects with a five-point Likert scale for each alternative set of modules, in which subjects were asked to express their level of agreement with a statement that the modules were highly cohesive. The definition of cohesion presented to the subjects was the degree to which elements within a particular module can be said to belong together with stronger relations than with those in different modules. We included the explicit instruction not to change the ranking of the alternatives when indicating their level of agreement with the cohesiveness of the modules for each alternative, as we were interested in finding out whether the perceived cohesiveness of the different sets of modules was associated with the ranking of the alternatives, but did not want to impose this criterion upon our subjects. In other words, we reckoned that perceived cohesiveness could be part of the rationale used to rank the alternative sets of modules, but we did not want to rule out other kinds of reasoning that would be revealed by the motivation for the ranking provided by the subjects.

The complete survey can be obtained in supplementary material at http://purl.org/krdb-core/relator-centric-clustering.

6.3 Analysis of the ranking results

In total, we received 62 complete responses to our survey (i.e., a response rate of approximately 38%). From the 62 participants, we excluded the responses of those that indicated to belong to at least one of the following not mutually exclusive categories: (1) to be 'novices' in conceptual modeling (10 participants); (2) to have less than four-year experience with conceptual modeling (18 participants); (3) and to have no knowledge of UML (5 participants). This left us with 42 participants whose responses could be considered as reflecting expert opinions. Included in this group of 42 were 14 persons that declared their level of expertise in conceptual modeling as 'intermediate' rather than 'expert'. As these 14 participants had at least four years of experience in conceptual modeling and all but one indicated to have knowledge of at least two conceptual modeling languages, we decided that we could safely treat them as 'experts' for the purpose of our study.

The 42 participants that made through the selection are hereafter referred to as the subjects of our study. Because the experts were recruited from our personal networks, the majority of them held a position in academia (30 subjects), while the others indicated to have a position in industry or government (12 subjects). The most frequent academic position was that of professor (19 subjects), while positions held in industry or governmental organizations varied widely: business analyst / architect (3), software engineer (3), system analyst / architect (2), consultant (1), mid-level or senior manager (1), data scientist (1), other IT staff position (1).

Table 1 shows how the 42 subjects ranked the three modularization alternatives (i.e., relator-centric, Akoka and Comyn-Wattiau, and Castano et al.), where we also distinguish between subjects that have declared to know OntoUML (28) and those that did not (14). With this distinction, we wished to assess whether OntoUML experts could 'guess' (project) the underlying rationale for relator-centric modularization, even if the models did not show OntoUML stereo-

Table 2 Subjects' perceptions of module cohesion, where 0 = completely disagree, 1 = somewhat disagree, 2 = neither agree or disagree, 3 = somewhat agree, and 4 = completely agree

Approach\Perceived cohesiveness All experts—42 subjects	**0**	**1**	**2**	**3**	**4**
Relator-centric	3	6	3	13	17
Akoka and Comyn-Wattiau	1	8	16	10	7
Castano et al.	8	14	14	4	2
Approach\Perceived cohesiveness OntoUML subgroup—28 subjects	**0**	**1**	**2**	**3**	**4**
Relator-centric	2	5	2	6	13
Akoka and Comyn-Wattiau	0	5	10	7	6
Castano et al.	5	7	10	4	2
Approach\Perceived cohesiveness Non-OntoUML subgroup—14 subjects	**0**	**1**	**2**	**3**	**4**
Relator-centric	1	1	1	7	4
Akoka and Comyn-Wattiau	1	3	6	3	1
Castano et al.	3	7	4	0	0

types. We reckoned that this 'guessing' (projecting), which could surface in the motivation for the ranking provided, is not a problem per se; on the contrary as it would support the soundness of the logic of relator-centric clustering. However, if the results are consistent across the OntoUML and non-OntoUML subgroups, then we have some confidence that OntoUML (or UFO) knowledge does not play a role in appreciating the clarity of relator-centric modularization. In other words, although relator-centric modularization leverages ontological semantics, actual knowledge of those semantics is not needed to appreciate the effectiveness of this modularization strategy in managing the complexity of conceptual models.

The table shows a clear preference for Relator-centric modularization, for all experts and also for both subgroups. Regarding the other two modularization alternatives, the one by Akoka and Comyn-Wattiau seems to be preferred above the one by Castano et al., and also this observation seems to be consistent across both subgroups.

To assess whether the observed differences in ranking the modularization alternatives are also statistically significant, we ran tests using the SPSS software package. The Friedman nonparametric test for differences in the rankings of the three alternative modularizations indicated that these differences were significant considering all experts (Chi-square: 25.33; $p < 0.001$), only the experts with knowledge of OntoUML (Chi-square: 17.64; $p < 0.001$), and only the experts with no knowledge of OntoUML (Chi-square: 8.14; $p < 0.05$). Running the same test again for only relator-centric modularization and the Akoka and Comyn-Wattiau modularization (which comes closest to relator-centric modularization in the rankings), we observe that the difference in ranking is statistically significant considering all experts (Chi-square: 6.10; $p < 0.05$), close to being significant for the OntoUML subgroup (Chi-square: 3.57; $p = 0.059$), but not significant for the non-OntoUML group (Chi-square: 2.57; $p = 0.109$), which might be due to the small sample size. These results provide evidence that the relator-centric modularization presented in this paper is the preferred modularization strategy to manage the complexity of conceptual models. It also seems that knowledge of OntoUML does not have a considerable impact on this preference.

To verify whether the preferences in ranking the modularization alternatives were associated with the perceived cohesiveness of the modules (Table 2), we performed correlational analyses by calculating the Kendall rank correlation coefficient (i.e., Kendall's tau). For these analyses we assumed that the preference rank given to a particular modularization strategy (i.e., 1, 2 or 3) is a value on an ordinal scale of preference, which is only an approximation as we only asked subjects to rank alternatives and not to express their preferences on a scale. Given this approximation, the correlational analyses show that for relator-centric modularization and the Akoka and Comyn-Wattiau modularization, the correlation with perceived cohesiveness is statistically significant (relator-centric: Kendall's tau 0.336; $p < 0.05$/Akoka and Comyn-Wattiau: Kendall's tau 0.506; $p < 0.001$), while it is not for the Castano et al. modularization (Kendall's tau 0.261; $p = 0.058$). These results suggest that the perception of cohesiveness of the modules was taken into account by the experts when judging the different modularization strategies, at least for the relator-centric and Akoka and Comyn-Wattiau modularization strategies, but that it cannot explain fully the rankings. The analysis of other elements in the rationale of

the subjects for ranking the alternative modularizations is presented in the next subsection.

6.4 Analysis of the motivations for the rankings

Twenty-nine of the 42 subjects ranked relator-centric clustering first. When looking at the motivation they provide, we find many elements that support the logic of modularizing based on relational contexts, meaning a focus on modules that highlight reified relationships and the roles played by entities in such relationships. Illustrative examples include (note that relator-centric clustering is referred to as ALT-A, ALT-K and ALT-Y depending on the version of the survey received):

- *"Coherence of the unifying concept—relators"* (subject 4, who reported to have no knowledge of OntoUML);
- *"ALT-A is the best one. It is organized by the events*,[14] *what made me better understand the different model topics"* (subject 9);
- *"Association classes (although not represented as they should, imho*[15]*), should not be broken down by modularization. Inheritance hierarchies should also be kept together, whenever possible."* (subject 11, who reported to have no knowledge of OntoUML);
- *"This approach seems to be centered around the relators."* (subject 13);
- *"First, I identified what are the main concepts to be represented in the domain: rental agency, marriage, employment. Then I associate to each concept, what entities are requested to explain them. These sets are, for me, the best modules"* (subject 19);
- *"ALT-Y shows each reified relation in one place and more fully defines the relata."* (subject 20);
- *"ALT-Y modularization was the best since it presented all important core concepts related to Car Rental in one module, and additional relationships in other modules."* (subject 25);
- *"For me ALT-Y provides the best way of maintaining the relations between concepts. Concepts like marriage aren't broken up. And for example employee is clearly visible in the two modularizations and therefore the relations with other concepts are still clear."* (subject 30);
- *"Complete taxonomies, central concept (Car Rental) more contextualized"* (subject 34);
- *"Only ALT-Y shows the association between two objects that are involved in modeling the event. Also this diagram shows the 'role' of an object."* (subject 36, reported to have no knowledge of OntoUML);
- *"In my point of view, modules should capture a context (in a vague sense) that is auto-contained and has all the concepts, relations, and properties that are necessary for talking about that context. Also, it is necessary to preserve in each module the concepts that provide identity to the concepts that are relevant for that context."* (subject 42)

We also found some other rationales, for instance related to perceived cohesiveness (e.g., *"Based on ontology modules criteria: self-contained, loose coupling, and high cohesion"* (subject 12)) or noting the loss of information in the Akoka and Comyn-Wattiau and Castano et al. modularizations (e.g., *"There is loss of information from the original model to ALT-X and ALT-Z."* (subject 37)).

If we look into the motivations of the eight subjects that preferred the Akoka and Comyn-Wattiau solution above ours, we observe that some subjects noticed the imbalance in module size of relator-centric clustering (which is circumstantial because of the model used in the study), such as:

- *"The worst alternative was ALT-A, because the modularization was unbalanced in terms of the size of the modules."* (subject 3);
- *"I think that the "main" module of ALT-Y is very big. I would prefer if it was defined an extra module for the hierarchy of Person."* (subject 21);
- *"ALT-J is the one focusing the most on the interactions between actors in the business model. ALT-K could be even better, but the inclusion of whole specialization hierarchies for the involved roles hurts the goal of modularization."* (subject 59)

A similar reasoning was also provided by some of the five subjects that preferred the Castano et al. modularization, e.g., *"size of models"* (subject 50), and *"use smaller components"* (subject 44).

6.5 Limitations

There are a few limitations to our study. First, we did not compare relator-centric clustering to all possible other modularization strategies, whether published or conceivable. The idea was to compare it with a number of alternatives that first, were based on topological criteria rather than a

[14] As discussed in [23,29], relators and events form a duality that is often manifest in terms of *systematic polysemy* in language (e.g., "marriage as a relator bundling mutual rights and obligations of the spouses" versus "marriage as a process aggregating the actions of the spouses qua players of these roles"; "service as a contractual agreement" versus "service as a service delivery event").

[15] Association classes were introduced in UML to address one of the functions of relators (namely, association reification) but with a poorer semantics [31]. It is natural that UML users interpret relators as association classes and, in doing so, resist to what they interpret as "change in notation".

logic of ontological semantics, and second, received some attention (e.g., as to date, 188 and 54 citations in Google Scholar for the Castano et al. and Akoka and Comyn-Wattiau papers, respectively). Moreover, both approaches were published in important venues in conceptual modeling (ACM Transactions of Database Systems and Data & Knowledge Engineering, respectively). Finally, we wanted to compare our approaches with *clustering* approaches that were *lossless transformations* and that could be *fully automated*, thus excluding alternatives such as [17,36]. Our focus on just two alternatives for comparison is also because we reckoned that fatigue could occur in our study if subjects were asked to rank a larger number of alternatives (and motivate this ranking), which could have harmed the reliability of the data.

Second, when operationalizing the modularization strategies of the alternative approaches, some choices had to be made as explained in the Materials subsection. While each choice could have been different, we carefully deliberated our options to stay truthful to the logic behind the original modularization strategies, so we do not consider this factor as a threat to the validity of the results.

Third, although the response rate was good, the actual number of valid responses could have been higher than 42, which would have increased statistical power. Nevertheless, even with 42 responses, the differences in the rankings of the three strategies tested, including ours, were statistically significant.

Fourth, our focus was on collecting evidence of the perceived effectiveness of relator-centric clustering compared to the other approaches tested. We could have focused more on explaining why our clustering strategy is preferred above the presented alternatives to see whether the logic of relator-centric clustering is indeed the rationale followed by the experts. The qualitative responses regarding perceived cohesiveness and motivations for the ranking were quite insightful (see previous sub-sections); however, a more in-depth investigation could have employed a process tracing technique like a think-aloud protocol that required subjects to explicitly verbalize their rationale for the ranking. In the current times of physical contact restrictions due to the COVID-19 pandemic, such a verbal protocol analysis study would have been difficult to realize.

6.6 How representative is the conceptual model used?

The model we used as a running example throughout this article and employed in the reported experiment (Fig. 1) is representative of the class of models that this approach addresses, i.e., those in business (organizational, social, and legal) domains.

As presented in [43] and formalized in [57], OntoUML is what is termed a *pattern grammar*. This means that the actual primitives of the language are *ontology design patterns* and, hence, that the valid models of the language are created by iterating through instantiations of these patterns.

In this paper, we focus on the subset of OntoUML modeling primitives presented in Sect. 3 and formally accounted for in Sect. 4.1. We claim that this subset comprises the primitives that amount to the vast majority of modeling elements used in conceptual models in our domains of interest.[16]

In models restricted to our adopted primitives, the following patterns from [43] are of relevance: (i) the Subkind Pattern, variants 1 and 2; (ii) the Phase Pattern; (iii) the Role Pattern; (iv) the RoleMixin Pattern, variants 1 and 2; (v) the Relator Pattern variant 1; (vi) the Relationally Dependent Pattern variant 1. For example, the model of Fig. 1 is a combination of: one instance of (i) variant-1, one instance of (i) variant-2; three instances of (ii) (including an iterated application of the phase pattern); five instances of (iii) (including an iterated application of the role pattern); one instance of (iv), variant-2; four instances of (v), variant-1. The Relational Dependence Pattern (iv) is always a combination of a Relator Pattern connected either to an instance of the Role Pattern or of the RoleMixin Pattern. This is exactly what we have following the iterated combination of (iii), (iv) and (v) above. In summary, the model of Fig. 1 contains instances of all the patterns (i)–(vi) and only those patterns. The only example missing in that model is of a variant-1 of the RoleMixin Pattern (iv). In that variant, we can have chains of RoleMixins specializing the RoleMixin connected to a focal relator until we reach the level of sortal subtypes. Although, we did not include this case in our running conceptual model—in order to keep it within manageable dimensions for the paper and experiment—the reader should note that this case is also covered by the NSIP definition in Sect. 4.5.

The fact that OntoUML models are created by the iterated combination of these patterns also enables us to make *a priori* considerations regarding the expected dimensions of the clusters produced by our approach.

On the one hand, the smallest possible cluster we could get is (in theory) a cluster with just one relator and one single kind (e.g., a Marriage connected to two instances of the single type Person). However, given that relational properties are almost ubiquitously optional for kinds, and that OntoUML methodologically aims at eliminating optional properties from conceptual models [25], the smallest clusters to be found in practice are akin to those of *Employment* and *Car Ownership* depicted in Fig. 2.

[16] For example, in the large MGIC model previously mentioned, the class stereotypes considered in Sect. 4.1 account for circa 90% of the cases, and the meta-relations of subtyping and mediation account for more than 85% of the represented relations.

On the other hand, there are five ways in which clusters can grow:[17] (a) by having a long chain of mediation relations between relators; (b) by having large generalization sets with subkinds and phases (*subkind and phase partitions*); (c) by a having long chain of generalization (supertyping) relations between the sortal type directly mediated by a focal relator and their corresponding kind (i.e., the Sortal Identity Path (SIP), see definition 4.4); (d) by having a long chain of specialization (subtyping) relations between the non-sortal type mediated by a focal relator and their first sortal subtypes, as well as from these to their corresponding kinds (i.e., the Non-Sortal Identity Path (NSIP), see definition 4.5); (e) by having a long chain of specialization relations involving relator types. Let us analyze these case by case.

Regarding case (a), as we discuss in footnote 6, in this case, we would formally have nested clusters. For example, in a model in which we have an *Insurance* (relator) connecting a *Car Rental* and an *Insurance Agency*, we would have a cluster *Car Rental Insurance* including the *Car Rental Insurance* Relator, the *Insurance Agency* and the *Car Rental cluster* as a sub-cluster. In other words, that relational context would represent that a *Car Rental Insurance* is a relator connecting an *Insurance Agency* and a *Car Rental*, and to know more about what a *Car Rental* is and what else is involved in it, one would have to investigate that nested cluster. We argue that, at least formally, this mechanism contains the growth of clusters in this case by taking sub-clusters to cognitively count at each level as one model element.

Regarding case (b), according to our experience, in non-taxonomic models, subkind and phase partitions are small-sized. For example, in the large model discussed above from [44], the percentage of phases in the model is merely 2.9% of the classes, and the *number of phases divided by number of kinds* in the model is circa 0.25, while the *number of subkinds divided by number of kinds* in the model is less than 2.

Now, regarding cases (c) and (d), an analysis of ten OntoUML models in different domains reported in [24] considers among the analyzed variables the maximum height of the hierarchy (symbolized by h) in each of those models (i.e., maximum path size from a top-level class to a leaf class). According to the authors, the average h for that sample is 3. In particular, in the two largest models considered there that are clearly in the business domain—one about Normative Acts containing 59 classes, and one about IT corporate architectures containing 66 classes—the value of h is 3 in both cases. Notice that $h = 3$ is also what we have for the model of Fig. 1 in the present paper. Even in the largest model overall in that sample (231 classes), which is a model about Biodiversity, i.e., clearly a taxonomic model in which generalization/specialization paths tend to be longer, we have $h = 5$.

Finally, regarding case (e), according to our experience, large chains of specializing relator types are very uncommon in business domains. For example, when considering the ten models analyzed in [24], none of the models in business domains included specializations between relator types. In fact, in that entire sample, only two models included relator types specializing each other, namely the aforementioned taxonomic model in biodiversity, and a model representing a telecommunications ITU-T standard in the domain of optical networks (123 classes), an atypical model containing hierarchies of relator types representing different sorts of logical network connections. Even in these two models, for the former, the longest *relator type hierarchy path* is of length = 1; for the latter model, the longest path from leaf classes to the corresponding top-class stereotyped as <<relator>> is of length = 5.

7 Final considerations

In this paper, we propose a formal approach for conceptual model *clustering* by leveraging on the ontologically well-founded semantics of the modeling language OntoUML. As such, this is a contribution to a broader research program that aims at developing ontology-based complexity management techniques for conceptual models. Other techniques in this program include *model recoding with ontology design patterns* [17] and *model abstraction/summarization* [27].

In this particular contribution, we rely on the theory of relators underlying OntoUML to present a full formal account of the notions of *Relational Context* (RC) and *Relator-Centric Clustering* (RCC). An RCC is a model modular breakdown in terms of a number of adequate RCs. Each RC, in turn, captures all the information needed to understand the maximal scope of objects in the way they participate in certain relationships. The approach is formally characterized (*claim to formal precision*), and it is based on a well-founded ontological theory of relators (*claim to ontological adequacy*).

Additionally, we have reported on a fully implemented plug-in tool for a model-based OntoUML editor that automates this approach (*claim to practical realizability*). Following the formal characterization of this framework, the algorithm implementing it is *deterministic* (i.e., it generate the same RCC for a given model in every execution) and, in the worst possible case, the algorithm would execute a total of $(n_e - n_r) * n_r$ operations (where n_e is the number of elements in the model and n_r is the number of classes stereotyped as relators). So, even in the worst possible case, the algorithm is tractable (*claim to computational efficiency and scalability*). In practice, n_r is on average circa 6% of n_e

[17] This follows not only from the patterns that can be used to build a *Relational Context* but also, and more directly, from its very formal definition in Sect. 4.

(as observed by analyzing 54 OntoUML models in different domains in the OntoUML repository [44,46]), and RCs are often largely disjoint with minimal intersections only in the level of kinds. In other words, in practice, the algorithm will often execute approximately n_e steps as the different RCs tessellate the original model. We have tested, nonetheless, the performance and scalability of our implementation against the large MGIC model. MGIC is atypical in the sense that it has $n_r = 771$, i.e., the number classes stereotyped as relators there is significantly higher than the observed average, amounting to circa 20% of the model. However, even in the tests carried out with this model (on a MacBook Pro 2017, Intel Core i5 2.3GHz with 8GB of 2,133 MHz RAM), the corresponding modular breakdown was generated, on average, under 12 seconds. These are encouraging results. Computational efficiency and scalability are important features for algorithms used for complexity management of conceptual models, given that the value of having proper techniques for addressing this issue increases with the size and complexity of the models.

Finally, we have conducted an empirical study to evaluate one particular *claim to cognitive adequacy* of our strategy. Despite the limitations of that study as discussed in Sect. 6.5, we believe to have collected convincing empirical evidence of the effectiveness of relator-centric clustering as a model management technique for reference conceptual models in the business domain, as perceived by experts in conceptual modeling and relative to other approaches that do not leverage the ontological semantics of models. We also have indications that: (i) this *perceived effectiveness* is related to *perceived cohesiveness* of the resulting model clusters; (ii) the underlying rationale for the modularization was confirmed by the motivations for ranking relator-centric clustering first by a reasonable number of experts; (iii) the preference for relator-centric clustering does not depend on prior knowledge of ontological semantics. In other words, we believe to have shown that there is a certain 'naturalness' in the strategy used to modularize a conceptual model based on its relational contexts.

In the past years, there is a growing interest in the representation of reified events (occurrences) as first-class citizens in structural conceptual models with many benefits [3,4,29,40]. For example, in [4], some of us have proposed an approach for event reification in conceptual modeling in which events have their own properties, and can form taxonomic, partonomic, and temporal ordering structures. Moreover, objects participate in these events playing a number of 'processual roles' (e.g., the roles of victim and perpetrator in a crime). In a future work, and as an extension of the approach presented here, we intend to characterize contexts and clusters centered around this notion of events. As discussed in [23,29], there is always an indissoluble link between certain events and relators, namely, on one hand, these events constitute the "life" of a relator (i.e., they are composed of occurrences that are manifestations of the constituents of a relator, e.g., commitments, claims, powers, etc.). On the other hand, relators define a focus for events providing an ontological criteria for their individuation (i.e., for individuating which occurrences are part of a particular marriage, enrollment, employment, etc.). Given this strong bond between these two ontological categories, we have every reason to believe that the formal clustering strategy proposed here could be straightforwardly adapted to handle relational contexts centered around events.

The notion of Relational Context proposed here bears a resemblance also to the notion of *Frames* in C.J. Fillmore's Frame Semantics [18]. In fact, we first considered using the term *Ontological Frame* (or *Relational Frame*) for this notion. Frames, in that tradition, are patterns that describe situations, events or relationships and in which elements appear playing interconnected and mutually dependent (semantic) roles. However, unlike our approach, frames have the primary goal of providing a background structure for the interpretation of lexical terms. RCs, in contrast, have as primary goal ontological transparency, focusing on connecting the entities playing complementary roles in the scope of bundles of relational properties (relators) to their identity-providing kinds.

Acknowledgements We are grateful to Ricardo A. Falbo (*in memoriam*) for the spark that led to this investigation, and to Claudenir M. Fonseca for his contribution in incorporating our clustering algorithm in the OntoUML Plugin for Visual Paradigm. G. Guizzardi and T.P. Sales are funded by the NeXON (UNIBZ) and OntoScape (UNIBZ) projects. J.P. Almeida is funded by CAPES (Grant Number 23038.028816/2016-41) and CNPq (Grants Numbers 312123/2017-5 and 407235/2017-5).

References

1. Akoka, J., Comyn-Wattiau, I.: Entity-relationship and object-oriented model automatic clustering. Data Knowl. Eng. **20**(2), 87–117 (1996)
2. Algergawy, A., Babalou, S., Klan, F., König-Ries, B.: Ontology modularization with OAPT. J. Data Semant. **9**(2), 53–83 (2020)
3. Allen, G.N., March, S.T.: The effects of state-based and event-based data representation on user performance in query formulation tasks. MIS Q. **30**, 269–290 (2006)
4. Almeida, J.P.A., Falbo, R.A., Guizzardi, G.: Events as entities in ontology-driven conceptual modeling. In: Laender, A.H.F., Pernici, B., Lim, E.P., de Oliveira, J.P.M. (eds.) Conceptual Modeling. ER 2019, pp. 469–483. Springer, Cham (2019)
5. Almeida, J.P.A., Guizzardi, G., Falbo, R.A., Sales, T.P.: gUFO: a lightweight implementation of the Unified Foundational Ontology (UFO). http://purl.org/nemo/doc/gufo
6. Amato, F., De Santo, A., Moscato, V., Persia, F., Picariello, A., Poccia, S.R.: Partitioning of ontologies driven by a structure-based approach. In: Proceedings of the 2015 IEEE 9th International Conference on Semantic Computing (IEEE ICSC 2015), pp. 320–323. IEEE (2015)
7. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., et al.: Gene ontology: tool for the unification of biology. Nat. Genet. **25**(1), 25–29 (2000)

8. Baldoni, M., Boella, G., van der Torre, I.L.: Interaction between objects in powerJava. J. Object Technol. **6**(2), 7–12 (2003)
9. Bennett, M.: The financial industry business ontology: best practice for big data. J. Bank. Regul. **14**(3), 255–268 (2013)
10. Biddle, B.J.: Recent developments in role theory. Annu. Rev. Sociol. **12**(1), 67–92 (1986)
11. Bork, D., Garmendia, A., Wimmer, M.: Towards a multi-objective modularization approach for entity-relationship models. In: ER Forum, Demo and Posters 2020, pp. 45–58. CEUR-WS (2020)
12. Castano, S., De Antonellis, V., Fugini, M.G., Pernici, B.: Conceptual schema analysis: techniques and applications. ACM Trans. Database Syst. **23**(3), 286–333 (1998)
13. Chen, J., Alghamdi, G., Schmidt, R.A., Walther, D., Gao, Y.: Ontology extraction for large ontologies via modularity and forgetting. In: Proceedings of the 10th International Conference on Knowledge Capture, pp. 45–52 (2019)
14. Detoni, A.A., Miranda, G.M., Renault, L.D., Falbo, R.A., Almeida, J.P.A., Guizzardi, G., Barcellos, M.P.: Exploring the role of enterprise architecture models in the modularization of an ontology network: a case in the public security domain. In: 2017 IEEE 21st International Enterprise Distributed Object Computing Workshop (EDOCW), pp. 117–126. IEEE (2017)
15. El Ghosh, M., Naja, H., Abdulrab, H., Khalil, M.: Application of ontology modularization for building a criminal domain ontology. In: AI Approaches to the Complexity of Legal Systems, pp. 394–409. Springer (2015)
16. Feldman, P., Miller, D.: Entity model clustering: structuring a data model by abstraction. Comput. J. **29**(4), 348–360 (1986)
17. Figueiredo, G., Duchardt, A., Hedblom, M.M., Guizzardi, G.: Breaking into pieces: an ontological approach to conceptual model complexity management. In: 2018 12th International Conference on Research Challenges in Information Science (RCIS), pp. 1–10 (2018). https://doi.org/10.1109/rcis.2018.8406642
18. Fillmore, C.J.: Frame semantics. In: Geeraerts, D. (ed.) Cognitive linguistics: basic readings, pp. 373–400. Mouton de Gruyter, Berlin (2006). https://doi.org/10.1515/9783110199901.373
19. Francalanci, C., Pernici, B.: Abstraction levels for entity-relationship schemas. In: Proceedings of 13th ER, pp. 456–473. Springer (1994)
20. Garcia, A.C., Tiveron, L., Justel, C.M., Cavalcanti, M.C.: Applying graph partitioning techniques to modularize large ontologies. In: Joint V Seminar on Ontology Research in Brazil and VII International Workshop on Metamodels, Ontologies and Semantic Technologies. ONTOBRAS-MOST 2012, vol. 938, pp. 72–83. CEUR-WS (2012)
21. Gonçalves, B., Guizzardi, G., Pereira Filho, J.G.: Using an ECG reference ontology for semantic interoperability of ECG data. J. Biomed. Inform. **44**(1), 126–136 (2011)
22. Guarino, N., Guizzardi, G.: "We need to discuss the relationship": revisiting relationships as modeling constructs. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) Advanced Information Systems Engineering, pp. 279–294. Springer, Cham (2015)
23. Guarino, N., Guizzardi, G.: Relationships and events: towards a general theory of reification and truthmaking. In: Conference of the Italian Association for Artificial Intelligence, pp. 237–249. Springer (2016)
24. Guidoni, G.L., Almeida, J.P.A., Guizzardi, G.: Transformation of ontology-based conceptual models into relational schemas. In: International Conference on Conceptual Modeling, pp. 315–330. Springer (2020)
25. Guizzardi, G.: Ontological foundations for structural conceptual models. CTIT, Centre for Telematics and Information Technology (2005)
26. Guizzardi, G.: Objects and events in context. In: 11th International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT). Keynote Speech (2019)
27. Guizzardi, G., Figueiredo, G., Hedblom, M.M., Poels, G.: Ontology-based model abstraction. In: 2019 13th International Conference on Research Challenges in Information Science (RCIS), pp. 1–13 (2019). https://doi.org/10.1109/rcis.2019.8876971
28. Guizzardi, G., Fonseca, C.M., Almeida, J.P.A., Sales, T.P., Benevides, A.B., Porello, D.: Types and taxonomic structures in conceptual modeling: a novel ontological theory and engineering support. Data Knowl. Eng. **134**, 101,891 (2021)
29. Guizzardi, G., Guarino, N., Almeida, J.P.A.: Ontological considerations about the representation of events and endurants in business models. In: International Conference on Business Process Management, pp. 20–36. Springer (2016)
30. Guizzardi, G., Sales, T.P., Almeida, J.P.A., Poels, G.: Relational contexts and conceptual model clustering. In: IFIP Working Conference on The Practice of Enterprise Modeling, pp. 211–227. Springer (2020)
31. Guizzardi, G., Wagner, G.: What's in a relationship: an ontological analysis. In: International Conference on Conceptual Modeling, pp. 83–97. Springer (2008)
32. Guizzardi, G., Wagner, G., Almeida, J.P.A., Guizzardi, R.: Towards ontological foundations for conceptual modeling: the Unified Foundational Ontology (UFO) story. Appl. Ontol. **10**(3–4), 259–271 (2015). https://doi.org/10.3233/AO-150157
33. Hitzler, P., Shimizu, C.: Modular ontologies as a bridge between human conceptualization and data. In: International Conference on Conceptual Structures, pp. 3–6. Springer (2018)
34. Lankhorst, M., et al.: Viewpoints and visualisation. In: Enterprise Architecture at Work: Modelling, Communication and Analysis, pp. 171–214. Springer (2017)
35. Lozano, J., Carbonera, J., Abel, M., Pimenta, M.: Ontology view extraction: an approach based on ontological meta-properties. In: 2014 IEEE 26th International Conference on Tools with Artificial Intelligence, pp. 122–129 (2014). https://doi.org/10.1109/ictai.2014.28
36. Lozano, J., Carbonera, J.L., Abel, M.: A novel approach for extracting well-founded ontology views. In: Papini et al. (ed.) Joint Ontology Workshops 2015, vol. 1517. CEUR-WS (2015)
37. Moody, D.: The physics of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Trans. Softw. Eng. **35**(6), 756–779 (2009)
38. Moody, D.L., Flitman, A.: A methodology for clustering entity relationship models: a human information processing approach. In: Proceedings of 18th ER, pp. 114–130. Springer (1999)
39. Moody, D.L., Flitman, A.R.: A decomposition method for entity relationship models: a systems theoretic approach. In: Altmann, G., Lamp, J., Love, P., Mandal, P., Smith, R., Warren, M. (eds.) International Conference on Systems Thinking in Management. ICSTM2000, vol. 72 (2000)
40. Olivé, A., Raventós, R.: Modeling events as entities in object-oriented conceptual modeling languages. Data Knowl. Eng. **58**(3), 243–262 (2006)
41. Özacar, T., Öztürk, Ö., Ünalır, M.O.: ANEMONE: an environment for modular ontology development. Data Knowl. Eng. **70**(6), 504–526 (2011)
42. Pezoa, F., Reutter, J.L., Suarez, F., Ugarte, M., Vrgoč, D.: Foundations of JSON schema. In: Proceedings of the 25th International Conference on World Wide Web, pp. 263–273. International World Wide Web Conferences Steering Committee (2016). https://doi.org/10.1145/2872427.2883029
43. Ruy, F.B., Guizzardi, G., Falbo, R.A., Reginato, C.C., Santos, V.A.: From reference ontologies to ontology patterns and back. Data Knowl. Eng. **109**, 41–69 (2017)
44. Sales, T.P., Guizzardi, G.: Ontological anti-patterns: empirically uncovered error-prone structures in ontology-driven conceptual models. Data Knowl. Eng. **99**, 72–104 (2015)

45. Snoeck, M.: Enterprise Information Systems Engineering: The MERODE Approach. Springer, Berlin (2014)
46. Teixeira, M.: An ontology-based process for domain-specific visual language design. Federal University of Espirito Santo, Brazil/Ghent University, Belgium (2016)
47. Tzitzikas, Y., Hainaut, J.L.: How to tame a very large ER diagram (using link analysis and force-directed drawing algorithms). In: Delcambre, L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, O. (eds.) Conceptual Modeling. ER 2005, pp. 144–159. Springer, Berlin (2005)
48. Tzitzikas, Y., Kotzinos, D., Theoharis, Y.: On ranking RDF schema elements (and its application in visualization). J. Univers. Comput. Sci. **13**(12), 1854–1880 (2007)
49. Verdonck, M., Gailly, F.: Insights on the use and application of ontology and conceptual modeling languages in ontology-driven conceptual modeling. In: Proceedings of 35th ER (2016)
50. Verdonck, M., Gailly, F., Pergl, R., Guizzardi, G., Martins, B., Pastor, O.: Comparing traditional conceptual modeling with ontology-driven conceptual modeling?: An empirical study. Inf. Syst. **81**, 92–103 (2019). https://doi.org/10.1016/j.is.2018.11.009
51. Villegas Niño, A.: A filtering engine for large conceptual schemas. Universitat Politècnica de Catalunya (2013)
52. W3C: OWL 2 Web Ontology Language. Structural specification and functional-style syntax. W3C recommendation 11 December 2012 (2012). https://www.w3.org/TR/owl2-syntax/
53. Weber, B.: The impact of modularization on the understandability of declarative process models: a research model. In: Information Systems and Neuroscience, p. 133 (2020)
54. Wieringa, R., de Jonge, W., Spruit, P.: Using dynamic classes and role classes to model object migration. Theory Pract. Object Syst. **1**(1), 61–83 (1995)
55. Winter, M., Pryss, R., Probst, T., Baß, J., Reichert, M.: Measuring the cognitive complexity in the comprehension of modular process models. IEEE Trans. Cogn. Develop. Syst. (2020). https://doi.org/10.1109/TCDS.2020.3032730
56. Xiao, G., Calvanese, D., Kontchakov, R., Lembo, D., Poggi, A., Rosati, R., Zakharyaschev, M.: Ontology-based data access: a survey. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, pp. 5511–5519 (2018). https://doi.org/10.24963/ijcai.2018/777
57. Zambon, E., Guizzardi, G.: Formal definition of a general ontology pattern language using a graph grammar. In: 2017 Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE pp. 1–10 (2017)

Giancarlo Guizzardi Giancarlo Guizzardi is a Full Professor of Computer Science at the Free University of Bozen-Bolzano, Italy, where he leads the Conceptual and Cognitive Modeling Research Group (CORE). He is also a Full Professor of Software Science and Evolution at the University of Twente, The Netherlands. He has been active for more than two decades in the areas of Ontologies, Conceptual Modeling, and Information Systems Engineering. He is currently an associate editor for the Applied Ontology journal and for Data & Knowledge Engineering and a member of a number of international journal editorial boards. He is a member of the ER Steering Committee and of the Advisory Board of the International Association for Ontology and its Applications (IAOA).

Tiago Prince Sales Tiago Prince Sales is an Assistant Professor at the Conceptual and Cognitive Modelling Research Group (CORE) of the Faculty of Computer Science, Free University of Bozen-Bolzano, Italy. He received his PhD from the University of Trento, Italy, with a cum laude distinction and the additional qualification of doctor europaeus. His main research focus is on ontology-driven conceptual modeling. In particular, he works on the development and evolution of the Onto UML language, he investigates and develops engineering tools for conceptual modeling, such as model simulation and anti-patterns, and he develops well-founded reference ontologies for business and social domains, such as value, risk, competition, and trust. He has over ten years of experience in industrial and technology transfer projects in a wide range of domains, such as tourism, media asset management, public healthcare, transportation, and oil and gas.

João Paulo A. Almeida João Paulo A. Almeida is Associate Professor at the Federal University of Espírito Santo, Brazil, and Senior Member of the Ontology and Conceptual Modeling Research Group (NEMO). He holds a Ph.D. in Computer Science from the University of Twente, The Netherlands. For over a decade, he has been working on the application of ontologies in conceptual modeling, enterprise architecture and enterprise modeling. Particularly of interest to this project, he has a decade long experience in the development of Model-Based Engineering tools and was directly involved in the development of a number of computational tools for OntoUML. He is a senior member of the IEEE and the ACM. He serves as a member of the IEEE EDOC Conference Steering Committee and of the Advisory Board of the International Association for Ontology and its Applications (IAOA).

Geert Poels Geert Poels is currently a Senior Full Professor of business informatics with the Faculty of Economics and Business Administration, Ghent University, Ghent, Belgium, where he teaches intermediate and advanced courses on information systems, IT management, enterprise architecture, and service design. He also teaches in the master's degree in enterprise ICT architecture with the IC Institute Beerzel, Belgium. He supervises Ph.D. research on digital marketplaces, cybersecurity, and GDPR. His research relates to conceptual modeling (as research method) and enterprise modeling (as research domain), with a focus on business process architecture mapping, ArchiMate, value modeling, and NLP-based automated generation of conceptual models out of user requirements documents. As in academic service, he was a member of the development team of the COBIT 2019 framework for IT governance.