Chapter 1
Modular Knowledge Representation and Reasoning in the Semantic Web[*]

Luciano Serafini and Martin Homola

Abstract Construction of *modular ontologies* by combining different modules is becoming a necessity in ontology engineering in order to cope with the increasing complexity of the ontologies and the domains they represent. The modular ontology approach takes inspiration from software engineering, where modularization is a widely acknowledged feature. *Distributed reasoning* is the other side of the coin of modular ontologies: given an ontology comprising of a set of modules, it is desired to perform reasoning by combination of multiple reasoning processes performed locally on each of the modules. In the last ten years a number of approaches for combining logics has been developed in order to formalize modular ontologies. In this chapter we survey and compare the main formalisms for modular ontologies and distributed reasoning in the Semantic Web. We select four formalisms build on formal logical grounds of Description Logics: Distributed Description Logics, $\mathcal{E}$-connections, Package-based Description Logics, and Integrated Distributed Description Logics. We concentrate on expressivity and distinctive modeling features of each framework. We also discuss reasoning capabilities of each framework.

Luciano Serafini
Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
e-mail: serafini@fbk.eu

Martin Homola
Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
Comenius University, Fac. of Mathematics, Physics and Informatics, 84248 Bratislava, Slovakia
e-mail: homola@fmph.uniba.sk

[*] This is a preprint of a book chapter whose final version has been published in "Semantic Web Information Management: a Model-Based Perspective" © 2010 Springer. The original publication is available at: `http://www.springerlink.com/`. This version was downloaded from: `http://ii.fmph.uniba.sk/~homola/papers`.

1.1 Introduction

One of the opportunities opened by the Semantic Web is the possibility of accessing and reusing knowledge bases available via the Internet in the form of RDF/OWL ontologies. In many cases it is not necessary to build a new ontology from scratch, instead one is able to reuse and compose already existing ontologies, which describe some aspect of the world, by including them or selected parts of them in the newly build knowledge base[2]. This phenomenon is quite similar to modular software development known from software engineering, where software packages that implement certain algorithms are possibly included and reused in newly developed programs. However, differently from software, there is not a substantial agreement on what does it mean to integrate a set of ontology modules.

The past ten years have seen several proposals of logics which have the explicit goal to define a formal semantics of the integration of modular ontologies. Some of them are based on first order logic or modal logics, other combine dynamic with epistemic logics, etc. The most influential formalisms of modular ontologies for the Semantic Web where the one based on Description Logics (DL) [1]. We study four such formalisms in this chapter:

Distributed Description logics (DDL). A framework for combining DL ontologies by means of directed semantic mapping, which was originally introduced by Borgida & Serafini [9].

$\mathcal{E}$-connections. A framework for combining non-overlapping ontologies encoded in DL but possibly also other logics by special inter-ontology roles, originally introduced by Kutz et al. [29].

Package-based Description Logics (P-DL). A framework for distributed ontologies that enables import of ontology entities between the modules. P-DL was originally introduced by Bao & Honavar [7].

Integrated Distributed Description Logics (IDDL). A framework for aligning DL ontologies by means of bi-directional semantic mapping, originally introduced by Zimmerman [41].

The idea of modular ontologies opens a number of different issues, each of which deserves for a specific investigation. Even if reductions between the formalisms are known [28, 4, 14] and all of them are reducible into a regular monolithic DL ontology [10, 14, 8, 42], it has to be remarked that each of the formalisms is focused on different modeling scenarios and is suited for different operations that have been enabled by the introduction of modular ontologies. In this respect we identify the following four important operations associated with modular ontologies.

Ontology combination. Ontology combination is motivated by the combination of ontologies each of which describes a separated portion of the domain. The simple case is when two ontologies O_1 and O_2 describe two distinct dimensions

[2] As this chapter concentrates on ontologies as formally defined knowledge bases with logical semantics, we use the terms knowledge base and ontology interchangeably.

of a complex domain (e.g., one is a geographic ontology and another is an ontology about organization). In order to construct an ontology on a complex domain which encompasses both of these dimensions (e.g., territorial organization) it is necessary to define complex cross domain concepts that combine concepts of O_1 and O_2. The $\mathcal{E}$-connections framework supports this perspective.

Ontology mapping. Ontology mapping is motivated by the resolution of *semantic heterogeneity* between ontologies. A simple case is when two ontologies O_1 and O_2 represent in a heterogeneous way knowledge about the same domain or about partially overlapping domains. To combine the knowledge contained in O_1 and O_2 it is necessary to represent the *semantic mappings* between them (i.e., to indicate the relation between entities from O_1 and O_2). DDL and IDDL are formalisms that support this perspective.

Ontology import. Ontology import is motivated as a flexible mechanism for *partial ontology reuse*. For instance, in order to build an ontology O_2 it is possible to re-use some of the entities (namely concepts, relations and individuals) defined in an already developed ontology O_1. This is done by *importing* entities from one ontology into another, in a similar fashion as when some functionalities of a software package are imported in some other software package. P-DL is a formalism that supports this perspective.

Ontology partitioning. Ontology partitioning is motivated by the problem of dealing with large and very complex ontologies by decomposing them into modules. An example is when a large ontology O is partitioned in a set of ontologies $O_1, \dots O_n$ such that the combination of $O_1, \dots O_n$ is equivalent to the initial ontology O. Ontology partitioning has been investigated in connection with $\mathcal{E}$-connections [18]. This approach is different from the previous three in that it is more concerned with identifying parts of ontologies which are sufficiently unrelated, in contrast from the previous three focussing on describing the relation between the components.

In addition to the approaches that we compare in this chapter, which all craft distributed and modular ontology frameworks by introducing new constructs to the language and extending the standard DL semantics in order to combine the modules with these constructs, there is also another approach which addresses the problem of ontology modularity by identifying the requirements under which the modules are soundly combined simply by union. One such a requirement is *locality of knowledge* [13, 12]. Another one is the notion of *conservative extension* [22, 30]. In this chapter, however, we concentrate on distributed and modular ontology frameworks of the first kind. A reader interested in comparison of both kinds will find some discussion on this issue in the work of Cuenca Grau & Kutz [14].

We start by introducing the DL $\mathcal{SHOIQ}_\mathsf{b}$ whose sub-languages will serve as local languages of the modular ontology frameworks under comparison. In Sect. 1.3 we discuss on the abstract level the basic unified features shared by all of the logic-based modular ontology frameworks. We then continue by reviewing DDL, $\mathcal{E}$-connections, P-DL and IDDL in Sects. 1.4–1.7. We conclude in the final section.

1.2 Ontologies and Description Logics

Among the languages that are currently used to represent ontologies for the Semantic Web, the most important is OWL Web Ontology Language [31, 32]. The semantics of OWL is derived from DL[3], thus providing a well founded logical grounding for this language, tractable reasoning algorithms and practical implementations of reasoning engines. For this reason, most modular ontology frameworks supports integration of ontologies expressed in different DL languages. In this section, we briefly introduce syntax and semantics of the DL called $\mathcal{SHOIQ}_\mathrm{b}$ [25, 38] whose sub-languages will serve as the underlying logic for the modular ontology frameworks that will be described and compared later on.

1.2.1 Description Logic $\mathcal{SHOIQ}_\mathrm{b}$

Definition 1.1 (Syntax of DL $\mathcal{SHOIQ}_\mathrm{b}$). Let N_I, N_R and N_C be pairwise disjoint sets of individual names, atomic role names and atomic concept names in the respective order. $\mathcal{SHOIQ}_\mathrm{b}$-roles are defined inductively as the smallest set such that:

- each $R \in N_\mathrm{C}$ (atomic role) is a role;
- given an atomic role $R \in N_\mathrm{R}$, the expression R^- (inverse role) also is a role;
- given two roles R and S each of the expressions $\neg R$ (role complement), $R \sqcap S$ (role intersection) and $R \sqcup C$ (role union) is also a role if it is safe[4].

An RBox $\mathcal{R}$ is a collection of axioms, each of one of the two forms:

$$R \sqsubseteq S \qquad\qquad \mathrm{Trans}(R)$$

where R and S are roles. First form is called a role inclusion axiom (RIA) and second is called a transitivity assertion. Let $\mathrel{\underline{\boxed{\ast}}}$ be a transitive-reflexive closure of $\sqsubseteq$. Given an RBox $\mathcal{R}$, a role R is transitive if $\mathrm{Trans}(R) \in \mathcal{R}$; a role S is called simple if there is no transitive role R such that $R \mathrel{\underline{\boxed{\ast}}} S$. If $R \mathrel{\underline{\boxed{\ast}}} S$ then R is called a sub-role of S. $\mathcal{SHOIQ}_\mathrm{b}$-concepts are defined inductively as the smallest set such that:

- each $A \in N_\mathrm{C}$ (atomic concept) is a concept;
- given two concepts C, D and a role R the expressions $\neg C$ (complement), $C \sqcap D$ (intersection), $C \sqcup D$ (union), $\exists R.C$ (existential restriction) and $\forall R.C$ (value restriction) are also concepts;
- given a concept C, a simple role S and a natural number $n \geq 0$, the expressions $\geqslant n S.C$ and $\leqslant n S.C$ (qualified number restrictions) are also concepts,

[3] OWL actually provides a whole family of languages. Of these, most derive the semantics directly from some specific DL. One exception to this is the OWL Full language which is part of the OWL 1 standard [31]. This language relies on RDF semantics instead.

[4] A $\mathcal{SHOIQ}_\mathrm{b}$ role expression is safe if its disjunctive normal form contains at least one non-negated conjunct in every disjunct, please refer work of Tobies [38].

Construct	Condition	
$\neg C$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	
$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$\mathcal{A}$
$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$\mathcal{L}$
$\forall R.C$	$(\forall R.C)^{\mathcal{I}} = \{i \in \Delta^{\mathcal{I}} \mid (\forall j \in \Delta^{\mathcal{I}})\ (i,j) \in R^{\mathcal{I}} \implies j \in C^{\mathcal{I}}\}$	$\mathcal{C}$
$\exists R.C$	$(\exists R.C)^{\mathcal{I}} = \{i \in \Delta^{\mathcal{I}} \mid (\exists j \in \Delta^{\mathcal{I}})\ (i,j) \in R^{\mathcal{I}} \wedge j \in C^{\mathcal{I}}\}$	
$\{o\}$	$\{o\}^{\mathcal{I}} = \{o^{\mathcal{I}}\}$	$\mathcal{O}$
$\geqslant n S.C$	$(\geqslant n S.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \in \Delta^{\mathcal{I}} \mid (x,y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geq n\}$	$\mathcal{Q}$
$\leqslant n S.C$	$(\leqslant n S.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \in \Delta^{\mathcal{I}} \mid (x,y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leq n\}$	
R^{-}	$(x,y) \in R^{-\mathcal{I}} \iff (y,x) \in R^{\mathcal{I}}$	$\mathcal{I}$
$\neg R$	$(\neg R)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus R^{\mathcal{I}}$	
$R \sqcap S$	$(R \sqcap S)^{\mathcal{I}} = R^{\mathcal{I}} \cap S^{\mathcal{I}}$	b
$R \sqcup S$	$(R \sqcup S)^{\mathcal{I}} = R^{\mathcal{I}} \cup S^{\mathcal{I}}$	

Table 1.1 Semantic constraints on complex $\mathcal{SHOIQ}_{\mathrm{b}}$ concepts and roles.

- given some $o \in N_{\mathrm{I}}$, the expression $\{o\}$ (nominal) is also a concept.

A TBox $\mathcal{T}$ is a set of axioms called General Concept Inclusions (GCI), each of the form:

$$C \sqsubseteq D$$

where C and D are concepts. An ABox $\mathcal{A}$ is a set of axioms of the two possible forms, a concept assertion (on the left) and a role assertion (on the right):

$$C(a) \qquad\qquad R(a,b)$$

where $a,b \in N_{\mathrm{I}}$ are individuals, C is concept and R is role. A $\mathcal{SHOIQ}_{\mathrm{b}}$ knowledge base is a triple $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ formed by a TBox, an RBox and an ABox.

Definition 1.2 (Semantics of $\mathcal{SHOIQ}_{\mathrm{b}}$). An interpretation of a $\mathcal{SHOIQ}_{\mathrm{b}}$ knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ consisting of a non-empty set $\Delta^{\mathcal{I}}$ called the interpretation domain of $\mathcal{I}$ and of and interpretation function $\cdot^{\mathcal{I}}$ which assigns:

- $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, an element if the domain, to each individual $a \in N_{\mathrm{I}}$;
- $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, a subset of the domain, to each concept C that appears in $\mathcal{K}$;
- $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, a subset of the cross-product of the domain, to each role R that appears in $\mathcal{K}$.

In addition, for complex concepts and roles, the interpretation satisfies the constraints presented in Table 1.1.

Given an RBox $\mathcal{R}$, an interpretation $\mathcal{I}$ satisfies the role hierarchy of $\mathcal{R}$, if for each two roles R and S such that $R \sqsubseteq^{*} S$ we have $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$. $\mathcal{I}$ satisfies a transitivity axiom $\mathrm{Trans}(R) \in \mathcal{R}$ if $R^{\mathcal{I}}$ is a transitive relation (i.e., for each $x,y,z \in \Delta^{\mathcal{I}}$, if $\langle x,y \rangle \in R^{\mathcal{I}}$ and $\langle y,z \rangle \in R^{\mathcal{I}}$ then also $\langle x,z \rangle \in R^{\mathcal{I}}$). $\mathcal{I}$ satisfies $\mathcal{R}$ (denoted $\mathcal{I} \models \mathcal{R}$) if it satisfies the role hierarchy of $\mathcal{R}$ and each transitivity axiom of $\mathcal{R}$. Interpretation $\mathcal{I}$ satisfies a GCI axiom $C \sqsubseteq D \in \mathcal{T}$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$. $\mathcal{I}$ satisfies a TBox $\mathcal{T}$ (denoted $\mathcal{I} \models \mathcal{T}$) if it satisfies each GCI axiom of $\mathcal{T}$. $\mathcal{I}$ satisfies a concept assertion $C(a) \in \mathcal{A}$ if $a^\mathcal{I} \in C^\mathcal{I}$. $\mathcal{I}$ satisfies a role assertion $R(a,b) \in \mathcal{A}$ if $\langle a^\mathcal{I}, b^\mathcal{I} \rangle \in R^\mathcal{I}$. $\mathcal{I}$ satisfies an ABox $\mathcal{A}$ (denoted $\mathcal{I} \models \mathcal{A}$) if it satisfies each ABox assertion of $\mathcal{A}$. Interpretation I is a model of $\mathcal{SHOIQ}_\mathrm{b}$-knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ if $\mathcal{I} \models \mathcal{T}$, $\mathcal{I} \models \mathcal{R}$ and $\mathcal{I} \models \mathcal{A}$.

Basic reasoning tasks for Description Logics include concept satisfiability checking and subsumption entailment checking. These are formally defined as follows:

Definition 1.3 (Reasoning tasks in $\mathcal{SHOIQ}_\mathrm{b}$). Given a $\mathcal{SHOIQ}_\mathrm{b}$-knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ we say that:

- a concept, say C, is satisfiable with respect to $\mathcal{K}$, if there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that the set $C^\mathcal{I}$ is non-empty;
- a subsumption formula, say $\phi = C \sqsubseteq D$, is entailed by $\mathcal{K}$ (denoted $\mathcal{K} \models \phi$), if for each model $\mathcal{I}$ of $\mathcal{K}$ we have $C^\mathcal{I} \subseteq D^\mathcal{I}$.

Thanks to the well known reduction [1] in praxis we only need to deal with concept satisfiability checking.

Theorem 1.1. *Given a $\mathcal{SHOIQ}_\mathrm{b}$-knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ and a subsumption formula $\phi = C \sqsubseteq D$, ϕ is entailed by $\mathcal{K}$ if and only if $C \sqcap \neg D$ is unsatisfiable with respect to $\mathcal{K}$.*

1.2.2 Sub-languages of $\mathcal{SHOIQ}_\mathrm{b}$

We have introduced such a powerful language like $\mathcal{SHOIQ}_\mathrm{b}$ since we aim at comparing distributed ontology frameworks which all use some sub-language of $\mathcal{SHOIQ}_\mathrm{b}$ as the underlying logic. Let us briefly introduce selected sub-languages of $\mathcal{SHOIQ}_\mathrm{b}$ that are of particular interest for these in distributed ontology frameworks (see also 3rd column of Table 1.1 for naming conventions):

$\mathcal{ALC}$. A sub language of $\mathcal{SHOIQ}_\mathrm{b}$ that only allows for atomic roles (i.e., each $R \in N_\mathrm{R}$), it does not allow RBoxes at all, and from the concept constructors it only allows complement ($\neg$), intersection ($\sqcap$), union ($\sqcup$), existential restriction ($\exists$) and value restriction ($\forall$) in complex concepts. $\mathcal{ALC}$ is one of the most basic DL languages and it is supported by all of the modular ontology frameworks that we will review.

$\mathcal{SHIQ}$. Also known as $\mathcal{ALCQHI}_{\mathrm{R}+}$. It enriches $\mathcal{ALC}$ concept constructors with qualified number restrictions ($\leqslant$ and $\geqslant$), it adds inverse roles, role hierarchies and transitive roles. The basic variant of DDL is built on top of $\mathcal{SHIQ}$.

$\mathcal{SHOIQ}$. Enriches $\mathcal{SHIQ}$ with nominals. Its three sub-languages $\mathcal{SHIQ}$, $\mathcal{SHOQ}$ and $\mathcal{SHIO}$ are of particular interest for $\mathcal{E}$-connections.

$\mathcal{ALCQHI}_\mathrm{b}$. Enriches $\mathcal{ALC}$ with qualified number restrictions, role hierarchies and inverses, and in addition by complex roles derived by role complement ($\neg$), intersection ($\sqcap$) and union ($\sqcup$) constructors. DDL enriched with role mappings is build on top $\mathcal{ALCQHI}_\mathrm{b}$.

The only framework that covers full $\mathcal{SHOIQ}_\mathrm{b}$ (and even goes beyond) is IDDL. However since such a logic is intractable, reasoning must always happen in some sub-language here.

1.3 Reference Distributed Ontology Framework

While different ontology frameworks have many distinctive features that make each of them suitable for different applications, in general a unifying view on these frameworks is possible. In this section we describe the features that are common to all the formalisms of our interest in this chapter. These basic logical components, defined by all the formalisms, are as follows:

Set of modules. Sometimes called local ontologies or knowledge sources, they constitute the modules that have to be integrated. This set is represented by a family of ontologies $\{O_i\}_{i\in I}$ indexed by a set of indexes I. The set of components is finite, and it is fixed and constant (i.e., once specified, there are no means to add, remove or merge components by means of logic). Each index uniquely identifies an ontology O_i which is to be integrated. One important parameter of such a component is the language in which each O_i is expressed, so called local language. For the comparison in this chapter we assume that the local language is always some language from the family of Description Logics, however also approaches that allow integration of DL with other local logics are known [28].

Connection between the modules. This component represents the connection between the local modules within the framework. Each logic introduces different constructs to represents such a component. This is indeed the core aspect that differentiates the approaches. We distinguish two basic approaches: a *distributed* approach where the connections are expressed between pairs of local ontologies, and an *integrated* approach in which the mappings between local ontologies are represented in a centralized theory. Another important parameter, that should be taken into account in this respect, concerns the type of objects that are connected. Some approaches allow to link only concepts, while other approaches allow to link also relations and individuals.

Semantics. Each of the approaches provides a formal semantics of the logic as an extension (or restriction) of the DL semantics. We distinguish two different philosophies: *centralized semantics*, there is a unique domain of interpretation in which each ontology component is interpreted; *distributed semantics* local ontologies are interpreted in a “private” domain, and the integration is reached by imposing some compatibility constraints on the possible combinations of local semantics.

Axiomatization and reasoning support. All of the approaches provide a sound and complete reasoning algorithm which computes the consequences of the integration. These algorithms are usually based on an extension of the tableaux based reasoning techniques that are available for DL. For some of the approaches an axiomatization is known. Also here we have two philosophies: some approaches are limited and only consider the consequences of integration within the modules, while others are interested also in reasoning on the mapping component.

In the reminder of the chapter we review the four formalisms for representing modular ontologies cited in the introduction. For each of them we will describe the syntax with the associated expressive power, the semantics, and, if it is available, the axiomatization. Since our main interest lies in representational aspects, we focus mostly on modeling features of these frameworks which we demonstrate on examples.

1.4 Distributed Description Logics

The Distributed Description Logics framework was originally introduced by Borgida & Serafini [10] and subsequently developed by Serafini et al. [36, 35]. The basic DDL supports local ontologies expressed in $\mathcal{SHIQ}$ and mapping between concepts. This mapping allows for propagations of the concept subsumption hierarchy across the component ontology modules. Successively, Serafini & Tamilin [37] extended DDL with individual correspondences, which encode mapping between individuals of the component ABoxes. With this type of mapping is was possible to propagate not only TBox knowledge (i.e., concept subsumption) but also assertional knowledge stored in the ABoxes of the component modules. In a similar fashion Ghidini et al. [21] propose an axiomatization of mapping between roles that supports also the propagation of role-hierarchy between the modules. Homola [23] proposed a modification of the DDL semantics in order to improve the propagation of subsumption on complex concepts composed from concepts that are mapped between directly, and successively DDL formalism was modified by Homola & Serafini [24] in order to support the composition of mappings across a chain of ontologies. We limit our review in this chapter to the above listed contributions. However, it is worthwhile to notice that recently the formalism was further extended by Ghidini at al. [20] in order to allow so called heterogeneous mappings, which associate items of different syntactic type, for instance concepts with relations.

1.4.1 Formalization

DDL was build on top of local logics as expressive as $\mathcal{SHIQ}$ and $\mathcal{ALCQHI}_\mathrm{b}$. In this section we build DDL on top of the latter, in order to demonstrate also mapping between roles and modeling with complex roles.

Definition 1.4 (DDLs over $\mathcal{ALCQHI}_\text{b}$). Assume a non-empty index set I, a family of sets of concept names $N_\text{C} = \{N_{\text{C}i}\}_{i\in I}$ a family of sets of role names $N_\text{R} = \{N_{\text{R}i}\}_{i\in I}$ and a family of sets of individual names $\mathcal{ALCQHI}_\text{b}$-role build over $N_{\text{I}i}$.

1. A Distributed TBox is a family of T-Boxes $\mathfrak{T} = \{\mathcal{T}_i\}_{i\in I}$ such that $\mathcal{T}_i$ is a TBox in the language $\mathcal{ALCQHI}_\text{b}$ build over $N_{\text{C}i}$ and $N_{\text{R}i}$.
2. A distributed R-Box $\mathfrak{R} = \{\mathcal{R}_i\}_{i\in I}$ where each $\mathcal{R}_i$, is an RBox in the language $\mathcal{ALCQHI}_\text{b}$ build over $N_{\text{C}i}$ and $N_{\text{R}i}$.
3. To state that a certain axiom ϕ belongs to $\mathcal{T}_i$, $\mathcal{R}_i$ or $\mathcal{A}_i$ we write $i : \phi$
4. A distributed A-Box $\mathfrak{A} = \{\mathcal{A}_i\}_{i\in I}$ is a family of ABoxes, where each $\mathcal{A}_i$ is an ABox on the language $\mathcal{ALCQHI}_\text{b}$ build over $N_{\text{C}i}$, $N_{\text{R}i}$, and $N_{\text{I}i}$.,
5. A bridge rule from i to j is an expression of the form
$$i : X \xrightarrow{\sqsubseteq} j : Y \qquad i : X \xrightarrow{\sqsupseteq} j : Y \qquad i : a \longmapsto j : b$$
where X and Y are either two atomic concepts or two atomic roles, and a, b are two individuals, in the respective language. We also define the bridge rule $i : X \xrightarrow{\equiv} j : Y$ as a syntactic shorthand for the pair of bridge rules $i : X \xrightarrow{\sqsubseteq} j : Y$ and $i : X \xrightarrow{\sqsupseteq} j : Y$. A set of bridge rules $\mathfrak{B} = \{\mathfrak{B}_{ij}\}_{i\neq j\in I}$ is a family of sets of bridge rules $\mathfrak{B}_{ij}$ form i to j.
6. a DDL knowledge base over I is a quadruple $\langle \mathfrak{T}, \mathfrak{R}, \mathfrak{A}, \mathfrak{B} \rangle$ with all four components ranging over I.

Bridge rules do not represent semantic relations stated from an external *objective* point of view. As opposed to IDDL (see Sect. 1.7), there is no such global view in the DDL formalism. Instead, bridge rules from i to j express relations between the modules i and j viewed from the *subjective* point of view of the j-th ontology.

Intuitively, the into-bridge rule $i : X \xrightarrow{\sqsubseteq} j : Y$ states that, from the point of view of the module j, that the concept (respectively, the role) X in the module i is less general than its local concept (respectively, local role) Y. Similarly, the onto-bridge rule $i : X \xrightarrow{\sqsupseteq} j : Y$ expresses the fact that, from the viewpoint of the module j, X from the module i is more general than Y from the module j. Bridge rules from i to j provide the possibility of propagating into the local ontology j (under some approximation) the concepts of the foreign ontology ontology i. Note, that since bridge rules reflect a subjective point of view, bridge rules from j to i are not necessarily the inverse of the rules from i to j, and in fact there may be no rules in one or both of these directions.

Definition 1.5 (Semantics of DDL). A distributed interpretation of a distributed knowledge base over an index set I, is a pair $\mathfrak{I} = \left\langle \{\mathcal{I}_i\}_{i\in I}, \{r_{ij}\}_{i\neq j\in I} \right\rangle$ consisting of a set of local interpretations $\{\mathcal{I}_i\}_{i\in I}$ and various domain mappings. For each $i \in I$ the local interpretation $\mathcal{I}_i = \langle \Delta^{\mathcal{I}_i}, \cdot^{\mathcal{I}_i} \rangle$ consists of a domain $\Delta^{\mathcal{I}_i}$ and an interpretation function $\cdot^{\mathcal{I}_i}$. The domain is either non-empty, but also possibly empty (in such a case we call $\mathcal{I}_i$ a hole and denote it by $\mathcal{I}_i = \mathcal{I}^\varepsilon$).

Given I, $i, j \in I$, a distributed interpretation $\mathfrak{I}$ over I satisfies elements of a distributed knowledge base $\mathfrak{K}$ (denoted by $\mathfrak{I} \models_\varepsilon \cdot$) according to the following clauses:

1. $\mathfrak{I} \models_\varepsilon i : C \sqsubseteq D$ if $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}$.
2. $\mathfrak{I} \models_\varepsilon \mathcal{T}_i$ if $\mathfrak{I} \models_\varepsilon i : C \sqsubseteq D$ for each $i : C \sqsubseteq D \in \mathcal{T}_i$.
3. $\mathfrak{I} \models_\varepsilon \mathfrak{T}$ if $\mathfrak{I} \models_\varepsilon \mathcal{T}_i$ for each $i \in I$.
4. $\mathfrak{I} \models_\varepsilon i : R \sqsubseteq S$ if $R^{\mathcal{I}_i} \subseteq S^{\mathcal{I}_i}$.
5. $\mathfrak{I} \models_\varepsilon \mathcal{R}_i$ if $\mathfrak{I} \models_\varepsilon i : R \sqsubseteq S$ for each $i : R \sqsubseteq S \in \mathcal{R}_i$.
6. $\mathfrak{I} \models_\varepsilon \mathfrak{R}$ if $\mathfrak{I} \models_\varepsilon \mathcal{R}_i$ for each $i \in I$.
7. $\mathfrak{I} \models_\varepsilon i : C(a)$ if $a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}$.
8. $\mathfrak{I} \models_\varepsilon i : R(a,b)$ if $\langle a^{\mathcal{I}_i}, b^{\mathcal{I}_i} \rangle \in R^{\mathcal{I}_i}$.
9. $\mathfrak{I} \models_\varepsilon \mathcal{A}_i$ if $\mathfrak{I} \models_\varepsilon \phi$ for each $\phi \in \mathcal{A}_i$.
10. $\mathfrak{I} \models_\varepsilon \mathfrak{A}$ if $\mathfrak{I} \models_\varepsilon \mathcal{A}_i$ for each $i \in I$.
11. $\mathfrak{I} \models_\varepsilon i : X \xrightarrow{\sqsubseteq} j : Y$ if $r_{ij}\big(X^{\mathcal{I}_i}\big) \subseteq Y^{\mathcal{I}_j}$[5],
12. $\mathfrak{I} \models_\varepsilon i : X \xrightarrow{\sqsupseteq} j : Y$ if $r_{ij}\big(X^{\mathcal{I}_i}\big) \supseteq Y^{\mathcal{I}_j}$,
13. $\mathfrak{I} \models_\varepsilon i : a \longmapsto j : b$ if $b \in r_{ij}(a)$
14. $\mathfrak{I} \models_\varepsilon \mathfrak{B}$ if $\mathfrak{I} \models_\varepsilon \phi$ for all axioms $\phi \in \mathfrak{B}$.

The distributed interpretation $\mathfrak{I}$ is a (distributed) model of $\mathfrak{K}$ (denoted by $\mathfrak{I} \models_\varepsilon \mathfrak{K}$) if $\mathfrak{I} \models_\varepsilon \mathfrak{T}$, $\mathfrak{I} \models_\varepsilon \mathfrak{R}$, $\mathfrak{I} \models_\varepsilon \mathfrak{A}$ and $\mathfrak{I} \models_\varepsilon \mathfrak{B}$.

1.4.2 Reasoning in DDL

The two standard decision problems in DL, satisfiability of concepts and entailment of subsumption, play prominent rôle also in context of DDL. Formally, the decision problems are defined as follows.

Definition 1.6. Given a distributed knowledge base $\mathfrak{K}$, an i-local concept C is satisfiable with respect to $\mathfrak{K}$ if there exists a distributed model $\mathfrak{I}$ of $\mathfrak{K}$ such that $C^{\mathcal{I}_i} \neq \emptyset$.

Definition 1.7. Given a distributed knowledge base $\mathfrak{K}$ and two i-local concepts C and D, it is said that C is subsumed by D with respect to $\mathfrak{K}$ if $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}$ in every distributed model $\mathfrak{I}$ of $\mathfrak{K}$. We also sometimes say that the subsumption formula $i : C \sqsubseteq D$ is entailed by $\mathfrak{K}$ and denote this by $\mathfrak{K} \models_\varepsilon i : C \sqsubseteq D$.

In addition we also define entailment of ABox expressions.

Definition 1.8. Given a distributed knowledge base $\mathfrak{K}$, a concept expression $i : C(a)$ is entailed by $\mathfrak{K}$ (denoted by $\mathfrak{K} \models_\varepsilon i : C(a)$) if $a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}$ in every distributed model $\mathfrak{I}$ of $\mathfrak{K}$. A role expression $i : R(a,b)$ is entailed by $\mathfrak{K}$ (denoted by $\mathfrak{K} \models_\varepsilon i : R(a,b)$) if $\langle a^{\mathcal{I}_i}, b^{\mathcal{I}_i} \rangle \in R^{\mathcal{I}_i}$ in every distributed model $\mathfrak{I}$ of $\mathfrak{K}$.

DDL mappings may be thought of as inter-module axioms that constrain the interpretations of the ontologies they connect, or analogously, that allow to derive new knowledge via logical consequence. The axiomatization of DDL [21] is provided in terms of of *propagation rules* that allows to propagate subsumption statements

[5] The notation $r_{ij}(d)$ is used for the set $\{d' \mid \langle d, d' \rangle \in r_{ij}\}$ and $r_{ij}(D)$ denotes the set $\bigcup_{d \in D} r_{ij}(d)$.

across different ontology modules. We express propagations rules in the following form:

$$\frac{\begin{array}{l}\text{subsumptions in } i\\ \text{bridge rules from } i \text{ to } j\end{array}}{\text{subsumption in } j}$$

The above rule should be interpreted as: if the subsumptions specified in the premise are proved in the module i and $\mathfrak{B}_{ij}$ contains the bridge rules specified in the premise, then the subsumption specified in the conclusion is entailed in the module j. The following propagation rules completely axiomatize the logical consequence in distributed knowledge bases in case of DDL with homogeneous bridge rules between concepts and roles (i.e., disregarding individual correspondences):

$$\frac{\begin{array}{l} i : X \sqsubseteq \bigsqcup_{k=1}^{n} Z_k \\ i : X \xrightarrow{\sqsupseteq} j : Y \\ i : Z_k \xrightarrow{\sqsubseteq} j : W_k \quad \text{for } 1 \leq k \leq n \end{array}}{j : Y \sqsubseteq \bigsqcup_{k=1}^{n} W_k} \qquad \frac{\begin{array}{l} i : \exists P.(\neg \sqcup_{k=1}^{p} A_k) \sqsubseteq (\sqcup_{k=1}^{m} B_k) \\ i : P \xrightarrow{\sqsupseteq} j : R \\ i : A_k \xrightarrow{\sqsubseteq} j : C_k \quad \text{for } 1 \leq k \leq p \\ i : D_k \xrightarrow{\sqsubseteq} B_k : \quad \text{for } 1 \leq k \leq m \end{array}}{j : \exists R.(\neg \bigsqcup_{k=1}^{p} C_k) \sqsubseteq (\bigsqcup_{k=1}^{m} D_k)} \qquad (1.1)$$

In the first rule X, Y, Z_k, W_k are either all concepts or all roles. In the second rule A_h, B_h, C_k and D_k are concepts, and P and R are roles or inverse roles, and $p \geq 0$. Furthermore when $p = 0$, $\bigsqcup_{k=1}^{p} \phi_k$ is defined to be $\bot$. Note that the first rule allows to propagate the concept/role subsumption hierarchy while the second rule allow to propagate the domain and range restriction on atomic and complex roles.

A practical reasoner for Distributed Description Logics has been developed in the system called DRAGO.[6]

1.4.3 Modeling with DDL

The most basic modeling feature of DDL are bridge rules that allow concepts across ontologies to be associated (i.e., they are used to express semantic mapping between concepts).

Example 1.1. Let us build a small ontology that will track all kinds of professions in some IT Company and relations between this professions. In the TBox $\mathcal{T}_1$ we have among others the following axioms:

$$\begin{array}{rlrl} \mathsf{ITStaff} \sqsubseteq \mathsf{Employee} & & \mathsf{SupportStaff} \sqsubseteq \mathsf{Employee} \\ \mathsf{CateringStaff} \sqsubseteq \mathsf{SupportStaff} & & \mathsf{AdministrativeStaff} \sqsubseteq \mathsf{SupportStaff} \\ \mathsf{Cook} \sqsubseteq \mathsf{CateringStaff} & & \mathsf{Chef} \sqsubseteq \mathsf{Cook} \end{array}$$

[6] DRAGO is distributed reasoner specifically developed for DDL. It is based on the Pellet reasoner. It is available via its homepage `http://drago.itc.it/`.

Let us also have an individual $\mathsf{johnSmith}$ who serves as a chef in the catering division. We assert this in the ABox $\mathcal{A}_1$:

$$\mathsf{Chef}(\mathsf{johnSmith})$$

This simple modeling allows us to derive some entailed knowledge (for instance, $\mathsf{Cook}(\mathsf{johnSmith})$, $\mathsf{CateringStaff}(\mathsf{johnSmith})$, $\mathsf{Cook} \sqsubseteq \mathsf{Employee}$, etc.). Suppose that one day we run into a readily available ontology $\mathcal{T}_2$ based upon the Escoffier's brigade de cuisine system. The excerpt of the ontology $\mathcal{T}_2$ is as follows:

$$\mathsf{KitchenChef} \sqsubseteq \mathsf{Cook} \qquad \mathsf{Saucemaker} \sqsubseteq \mathsf{Cook}$$
$$\mathsf{PastryCook} \sqsubseteq \mathsf{Cook} \qquad \mathsf{Baker} \sqsubseteq \mathsf{PastryCook}$$

Suppose that we decide to reuse the knowledge of $\mathcal{T}_2$ within our organization. We will keep the ontology $\mathcal{T}_1$ cleaner and reuse knowledge of $\mathcal{T}_2$ whenever possible. First, we remove the two axioms $\mathsf{Cook} \sqsubseteq \mathsf{CateringStaff}$ and $\mathsf{Chef} \sqsubseteq \mathsf{Cook}$ from $\mathcal{T}_1$ because they are redundant in light of $\mathcal{T}_2$. We do keep however the concepts Cook and Cheff in $\mathcal{T}_1$ as these are positions that some people occupy in our company (e.g., $\mathsf{johnSmith}$). In order to integrate the ontologies, we add the following bridge rules:

$$2 : \mathsf{Cook} \xrightarrow{\sqsubseteq} 1 : \mathsf{CateringStaff} \qquad 2 : \mathsf{Cook} \xrightarrow{\equiv} 1 : \mathsf{Cook}$$
$$2 : \mathsf{KitchenChef} \xrightarrow{\sqsupseteq} 1 : \mathsf{Chef}$$

Thanks to the mapping expressed by the bridge rules, the two subsumptions that we have previously removed from the TBox $\mathcal{T}_1$ are now entailed (i.e., $\mathfrak{K} \models_\varepsilon 1 : \mathsf{Chef} \sqsubseteq \mathsf{Cook}$ and $\mathfrak{K} \models_\varepsilon 1 : \mathsf{Cook} \sqsubseteq \mathsf{CateringStaff}$). We are also equally able to derive $\mathfrak{K} \models_\varepsilon 1 : \mathsf{Cook}(\mathsf{johnSmith})$, $\mathfrak{K} \models_\varepsilon 1 : \mathsf{CateringStaff}(\mathsf{johnSmith})$ and $\mathfrak{K} \models_\varepsilon 1 : \mathsf{Cook} \sqsubseteq \mathsf{Employee}$, etc.

With individual correspondence axioms we are able to encode mapping between individuals. We illustrate this by an example.

Example 1.2. Let us extend the distributed ontology from Example 1.1. Suppose that the accounting department decides to create their own ontology where they model things in the company in way that is more practical for the accountants. In the local TBox $\mathcal{T}_3$ we have:

$$\mathsf{Accountant} \sqsubseteq \mathsf{Employee} \qquad \mathsf{Director} \sqsubseteq \mathsf{Employee}$$
$$\mathsf{SeniorExecutive} \sqsubseteq \mathsf{Employee} \qquad \mathsf{JuniorExecutive} \sqsubseteq \mathsf{Employee}$$
$$\mathsf{BasicStaff} \sqsubseteq \mathsf{Employee} \qquad \mathsf{SupportStaff} \sqsubseteq \mathsf{Employee}$$

This ontology then makes reuse of the knowledge already existing within the distributed knowledge base, employing the bridge rules: follows:

$$1 : \mathsf{CateringStaff} \xrightarrow{\sqsubseteq} 3 : \mathsf{SupportStaff}$$

$$1 : \mathsf{ITStaff} \xrightarrow{\sqsubseteq} 3 : \mathsf{BasicStaff}$$

In addition the accounting department keeps its own evidence of all employees in $\mathcal{A}_3$, using individuals with special nomenclature relying on employee numbers (i.e., individuals such as $\mathsf{e006B3F}, \mathsf{eF9DB15}, \mathsf{eE23A28}$, etc.). Using individual correspondence axioms, these individuals are matched with other representations of the same employee in the knowledge base, for instance:

$$1 : \mathsf{johnSmith} \longmapsto 3 : \mathsf{e006B3F}$$

Now, even without incorporating into $\mathcal{T}_3$ concepts such as Cook and Chef which are of little interest to company accountants and without explicit recording of all information about the precise working position of e006B3F in $\mathcal{T}_3$ and $\mathcal{A}_3$ we are able to derive $\mathfrak{T} \models_\varepsilon 3 : \mathsf{SupportStaff}(\mathsf{e006B3F})$.

In DDL it is also possible to bridge between roles. A practical modeling scenario that makes use of such bridge rules is showed below, by extending our running example.

Example 1.3. In order to improve work efficiency, the accounting department has planned a reorganization of offices. Under the assumption that people do more work when they feel fine in the office, the accounting department asked the employees to create a simple friend or a foe ontology $\mathcal{A}_4$. See, an excerpt from this ontology:

$$\mathsf{friendOf}(\mathsf{johnSmith}, \mathsf{robertKay}) \qquad \mathsf{friendOf}(\mathsf{robertKay}, \mathsf{johnSmith})$$

$$\mathsf{foeOf}(\mathsf{robertKay}, \mathsf{stanCoda}) \qquad \mathsf{foeOf}(\mathsf{stanCoda}, \mathsf{markHoffer})$$

The following bridge rules are used to transfer knowledge that is relevant into the ontology of the accounting department:

$$4 : \mathsf{friendOf} \xrightarrow{\sqsubseteq} 3 : \mathsf{goodOfficeMateOf} \qquad 4 : \mathsf{foeOf} \xrightarrow{\sqsubseteq} 3 : \mathsf{dislikes}$$

$$1 : \mathsf{officeMateOf} \xrightarrow{\sqsubseteq} 3 : \mathsf{currentOfficeMateOf}$$

We are especially interested in employees which do not feel comfortable sharing the office with their current office mates, hence we define a new role that will be called unhappyOfficeMateOf by an axiom in the RBox $\mathcal{R}_3$:

$$\mathsf{currentOfficeMateOf} \sqcap \mathsf{dislikes} \sqsubseteq \mathsf{unhappyOfficeMateOf}$$

Now, by using individual correspondence axioms, the relation between the individuals representing the same employee in different modules is encoded:

$$1 : \mathsf{johnSmith} \longmapsto 3 : \mathsf{e006B3F} \qquad 4 : \mathsf{johnSmith} \longmapsto 3 : \mathsf{e006B3F}$$

and so on for the other individuals. Making use of all this knowledge, we are finally able to derive in $\mathcal{T}_3$ who of the employees is located in the same office with an unfriendly co-worker and also we are able to determine which office mates would be mode suitable for such employees.

The distributed ontology representation framework of DDL allows to combine several ontologies by means of ontology mapping represented by bridge rules. Such an approach allows ontology engineers to reuse existing ontologies when building a new one. DDL also allows ontologies to be modeled in a modular way, keeping each module relatively small and maintainable, as part of the knowledge is imported from remote modules. Apart from other distributed ontology frameworks, $\mathcal{E}$-connections in particular, DDL is able to cope with heterogeneous environments such as for instance the envisioned Semantic Web, not requiring strict separation of modeling domains and being able to deal with inconsistency that may appear locally in such systems.

1.5 $\mathcal{E}$-connections

The motivation behind $\mathcal{E}$-connections, a framework originally introduced by Kutz et al. [28, 26], is the possibility of combining different logics each of which represents one aspect of a complex system. $\mathcal{E}$-connections were defined over so called Abstract Description Systems, a common generalization of Description Logics, modal logics, logics of time and space and some other logical formalisms, as introduced by Baader et al. [2]. Properties of $\mathcal{E}$-connections over Description Logics were studied by Kutz et al. [27] and Cuenca Grau et al. [16, 15]. $\mathcal{E}$-connections of OWL ontologies were also introduced by Cuenca Grau et al. [17].

A distinctive feature of $\mathcal{E}$-connections, different from other modular ontology frameworks presented in this chapter, is that each local ontology is supposed to model a portion of the domain that is complementary and non-overlapping with respect to the other local ontologies (e.g., the two domains of people and vehicles are non-overlapping). Hence, in $\mathcal{E}$-connections it is not possible to have a concept in some ontology module that has subconcepts or instances in some other ontology module. For further illustration of non-overlapping modeling domains see Example 1.4 below.

In $\mathcal{E}$-connections, the ontology modules are combined using so called *link properties*, in short also called *links*. Links are effectively inter-ontology roles, they are allowed in restrictions and thus they allow the local ontologies to be connected. In contrast with DDL and some other distributed ontology frameworks, links are not intended to represent semantic mappings between ontologies.

1.5.1 Formalization

The two most commonly explored flavours of $\mathcal{E}$-connections of Description Logics are $\mathcal{C}^{\varepsilon}_{\mathcal{HQ}}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ and $\mathcal{C}^{\varepsilon}_{\mathcal{HI}}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ [15, 16, 17]. These allow $\mathcal{SHIQ}$, $\mathcal{SHOQ}$, and $\mathcal{SHIO}$ local ontologies to be connected with links, the former allowing link hierarchies and qualified number restrictions on links, the latter link hierarchies and inverse links. It has been noted that more general frameworks such as $\mathcal{C}^{\varepsilon}_{\mathcal{HIQ}}(\mathcal{SHOIQ})$ and even $\mathcal{C}^{\varepsilon}_{\mathcal{HIQ}}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ lead to undesired behaviour such as projection of nominals from one local model to another [17, 28].

In addition, two extended frameworks named $\mathcal{C}^{\varepsilon}_{\mathcal{HQ}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ and $\mathcal{C}^{\varepsilon}_{\mathcal{HI}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ [34, 15] were also studied. These extend the framework with several tweaks useful from the modeling perspective: same property name is freely reusable as a role name (within a module) and as well as a link name (between the modules), transitive links are allowed and transitivity of roles and links is now controlled by so called general transitivity axiom which enables switching the transitivity on and off based on the context of modules within/between which the role/link is used. For sake of simplicity we introduce a slightly more general $\mathcal{C}^{\varepsilon}_{\mathcal{HIQ}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$, of which $\mathcal{C}^{\varepsilon}_{\mathcal{HQ}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ is the sub-language that does not allow inverse links and $\mathcal{C}^{\varepsilon}_{\mathcal{HI}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ is the sub-language that does not allow links in number restrictions.

Definition 1.9 (Syntax of $\mathcal{C}^{\varepsilon}_{\mathcal{HIQ}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$). Given a finite index set I, for $i \in I$ let m_i be a constant either equal to $\mathcal{SHIQ}$, $\mathcal{SHOQ}$ or $\mathcal{SHIO}$ that serves to identify the local language of a component knowledge base. For $i \in I$ let ${N_C}_i^{m_i}$ and ${N_I}_i^{m_i}$ be pairwise disjoint sets of concepts names and individual names respectively. For $i, j \in I$, i and j not necessarily distinct, let $\varepsilon_{ij}^{m_i}$ be sets of properties, not necessarily mutually disjoint, but disjoint w.r.t. ${N_C}_k^{m_k}$ and ${N_I}_k^{m_k}$ for any $k \in I$ and let ρ_{ij} be sets of ij-properties defined as follows:

- if $i = j$ and $m_i = \mathcal{SHOQ}$ then $\rho_{ij} = \varepsilon_{ij}^{m_i}$;
- otherwise $\rho_{ij} = \varepsilon_{ij}^{m_i} \cup \{P^- \mid P \in \varepsilon_{ji}^{m_j}\}$;

For $P_1, P_2 \in \rho_{ij}$ an ij-property axiom is an assertion of the form $P_1 \sqsubseteq P_2$. A general transitivity axiom is of the form $\mathrm{Trans}(P; (i_1,i_2),(i_2,i_3)\dots,(i_{n-1},i_n))$ provided that for each $k \in \{1,\dots,p\}$ we have $P \in \varepsilon_{i_k j_k}^{m_{i_k}}$.An ij-property box $\mathcal{R}_{ij}$ is a finite set of ij-property axioms. The combined property box $\mathcal{R}$ contains all the property boxes for each $i, j \in I$ (not necessarily distinct) and also all transitivity axioms. Let us denote by $\overset{*}{\sqsubseteq}$ the transitive-reflexive closure on $\sqsubseteq$. A property P is said to be transitive in (i, j), if $\mathrm{Trans}(P; (i_1,i_2),\dots,(i_{n-1},i_n)) \in \mathcal{R}$ such that $i_k = i$ and $i_{k+1} = j$, for some $1 \le k < n$. An ij-property P is called simple if there is no S that is transitive in (i, j) and $S \overset{*}{\sqsubseteq} P$.

Given some $i \in I$ the set of i-concepts is defined inductively, as the smallest set such that:

- each atomic concept $A \in {N_C}_i^{m_i}$ and two special symbols $\top_i$ and $\bot_i$ are i-concepts;

- given i-concepts C,D a j-concept Z and $P \in \rho_{ij}$, also $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists P^{(j)}.Z$, and $\forall P^{(j)}.Z$ are i-concepts;
- if $m_i \in \{\mathcal{SHOQ},\mathcal{SHIO}\}$ and $o \in N_{\mathrm{I}i}$ then $\{o\}$ is an i-concept;
- given a j-concept Z, a natural number n and a simple property $S \in \rho_{ij}$ such that if $i = j$ then $m_i \in \{\mathcal{SHIQ},\mathcal{SHOQ}\}$, also $\geqslant n S^{(j)}.Z$ and $\leqslant n S^{(j)}.Z$ are i-concepts[7].

Given two individuals $a,b \in N_{\mathrm{I}i}$, an i-concept C and some role $R \in \rho_{ii}$, the expression $C(a)$ is called an i-local concept assertion and the expression $R(a,b)$ is called an i-local role assertion. In addition, given two individuals $a \in N_{\mathrm{I}i}$ and $b \in N_{\mathrm{I}j}$ such that $i \neq j$, and some link property $E \in \rho_{ij}$, an object assertion is an axiom of the form:

$$a \cdot E \cdot b$$

A combined TBox is a tuple $\mathcal{K} = \{\mathcal{K}_i\}_{i\in I}$ where each $\mathcal{K}_i$ is a set of i-local GCI each of the form $C \sqsubseteq D$, where C and D are i-concepts. A combined ABox $\mathcal{A} = \{\mathcal{A}_i\}_{i\in I} \cup \mathcal{A}_{\mathcal{E}}$ is a set containing local ABoxes[8] $\mathcal{A}_i$, each comprising of a finite number of i-local concept and role assertions, and a finite number of object assertions $\phi \in \mathcal{A}_{\mathcal{E}}$. A combined knowledge base $\Sigma = \langle \mathcal{K},\mathcal{R},\mathcal{A}\rangle$ is composed of a combined TBox, a combined property box, and a combined ABox, all of them defined over same index set I.

Semantics of $\mathcal{E}$-connections employs so called combined interpretations which are similar to distributed interpretations of DDL. A combined interpretation $\mathcal{I}$ consists of local domains and interpretation functions. Local domains $\Delta^{\mathcal{I}_i}$ are pairwise disjoint and unlike in DDL they are strictly non-empty. There is no distinctive domain relation as in DDL, instead there are special interpretation functions for links which assign to each link $E \in \varepsilon_{ij}$ a subset of $\Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$ thus effectively turning them into a form of inter-ontology properties.

Definition 1.10 (Semantics of $\mathcal{C}^{\varepsilon}_{\mathcal{HIQ}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$). Let us assume a combined knowledge base $\Sigma = \langle \mathcal{K},\mathcal{R},\mathcal{A}\rangle$ with some index set I. A combined interpretation is a triple $\mathcal{I} = \langle \{\Delta^{\mathcal{I}_i}\}_{i\in I}, \{\cdot^{\mathcal{I}_i}\}_{i\in I}, \{\cdot^{\mathcal{I}_{ij}}\}_{i,j\in I}\rangle$, where for each $i \in I$, $\Delta^{\mathcal{I}_i} \neq \emptyset$ and for each $i,j \in I$ such that $i \neq j$ we have $\Delta^{\mathcal{I}_i} \cap \Delta^{\mathcal{I}_j} = \emptyset$. Interpretation functions $\cdot^{\mathcal{I}_i}$ provide denotation for i-concepts and interpretation functions $\cdot^{\mathcal{I}_{ij}}$ are employed for sake of denotation of ij-properties.

Each ij-property $P \in \rho_{ij}$ is interpreted by $P^{\mathcal{I}_{ij}}$ a subset of $\Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$. If $P \in \rho_{ij}$ is an inverse, say $P = Q^-$, $Q \in \rho_{ji}$, then $P^{\mathcal{I}_{ij}} = \{(x,y) \in \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j} \mid (y,x) \in Q^{\mathcal{I}_{ji}}\}$. A

[7] Note that even if the local language of the i-th module is $\mathcal{SHIO}$ (i.e., $m_i = \mathcal{SHIO}$) for some index $i \in I$, number restrictions may still appear in i-concepts as long as they are specified on link properties [34, 15].

[8] While local ABoxes are not included in $\mathcal{C}^{\varepsilon}_{\mathcal{HQ}}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$, $\mathcal{C}^{\varepsilon}_{\mathcal{HI}}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$, $\mathcal{C}^{\varepsilon}_{\mathcal{HQ}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$, $\mathcal{C}^{\varepsilon}_{\mathcal{HI}+}(\mathcal{SHIQ},\mathcal{SHOQ},\mathcal{SHIO})$ $\mathcal{E}$-connections as defined by Cuenca Grau et al. [15, 16, 17] nor as given by Parsia & Cenca Grau [34], they are present in the previous work of Kutz et al. [27, 28]. Since they do not constitute any problem semantically we add them for sake of comparison. Note also that even if the inter-module object assertions are possibly expressed using the same syntax as role assertions, we favour a syntax similar to the one of Cuenca Grau & Kutz [14] in order to make a clear distinction here.

combined interpretation $\mathcal{I}$ satisfies an ij-property axiom $P_1 \sqsubseteq P_2$ if ${P_1}^{\mathcal{I}_{ij}} \subseteq {P_2}^{\mathcal{I}_{ij}}$. It satisfies a transitivity axiom $\mathrm{Trans}(P;(i_1,i_2),\ldots,(i_{n-1},i_n))$ if both of the following conditions are true:[9]

1. for each $1 \leq k < n$, $P^{\mathcal{I}_{i_k i_{k+1}}}$ is transitive relation;
2. for each $1 \leq k < h < n$, $P^{\mathcal{I}_{i_k i_h}} = P^{\mathcal{I}_{i_k i_{k+1}}} \circ \cdots \circ P^{\mathcal{I}_{i_h i_{h+1}}}$.

A combined interpretation satisfies an ij-property box $\mathcal{R}_{ij}$ if it satisfies all ij-property axioms of $\mathcal{R}_{ij}$ and it satisfies a combined property box $\mathcal{R}$ if it satisfies each ij-property box and each transitivity axiom contained in $\mathcal{R}$.

Each i-concept C is interpreted by some subset $C^{\mathcal{I}_i}$ of $\Delta^{\mathcal{I}_i}$. For special symbols we have $\top_i = \Delta^{\mathcal{I}_i}$ and $\bot_i = \emptyset$. In addition, denotation of complex i-concepts must satisfy the constraints as given in Table 1.2. Combined interpretation $\mathcal{I}$ satisfies an i-local GCI $C \sqsubseteq D$ (denoted by $\mathcal{I} \models C \sqsubseteq D$) if $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}$; it satisfies a local TBox $\mathcal{K}_i$ (denoted $\mathcal{I} \models \mathcal{K}_i$) if it satisfies each i-local GCI thereof; and is satisfies a combined TBox $\mathcal{K}$ over I (denoted $\mathcal{I} \models \mathcal{K}$) if it satisfies $\mathcal{K}_i$ for each $i \in I$.

Construct	Condition
$\neg C$	$(\neg C)^{\mathcal{I}_i} = \Delta^{\mathcal{I}_i} \setminus C^{\mathcal{I}_i}$
$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}_i} = C^{\mathcal{I}_i} \cap D^{\mathcal{I}_i}$
$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}_i} = C^{\mathcal{I}_i} \cup D^{\mathcal{I}_i}$
$\{o\}$	$\{o\}^{\mathcal{I}_i} = \{o^{\mathcal{I}_i}\}$
$\forall P.Z$	$(\forall P.Z)^{\mathcal{I}_i} = \{x \in \Delta^{\mathcal{I}_i} \mid (\forall y \in \Delta^{\mathcal{I}_j})\ (x,y) \in P^{\mathcal{I}_{ij}} \implies y \in Z^{\mathcal{I}_j}\}$
$\exists P.Z$	$(\exists P.Z)^{\mathcal{I}_i} = \{x \in \Delta^{\mathcal{I}_i} \mid (\exists y \in \Delta^{\mathcal{I}_j})\ (x,y) \in P^{\mathcal{I}_{ij}} \wedge y \in Z^{\mathcal{I}_j}\}$
$\geqslant n S.Z$	$(\geqslant n S.Z)^{\mathcal{I}_i} = \{x \in \Delta^{\mathcal{I}_i} \mid \sharp\{y \in \Delta^{\mathcal{I}_j} \mid (x,y) \in S^{\mathcal{I}_{ij}}\} \geq n\}$
$\leqslant n S.Z$	$(\leqslant n S.Z)^{\mathcal{I}_i} = \{x \in \Delta^{\mathcal{I}_i} \mid \sharp\{y \in \Delta^{\mathcal{I}_j} \mid (x,y) \in S^{\mathcal{I}_{ij}}\} \leq n\}$

Table 1.2 Semantic constraints on complex i-concepts in $\mathcal{E}$-connections. C, D are i-concepts Z is an j-concept, P, S are either roles or links, S is simple.

A combined interpretation $\mathcal{I}$ satisfies an i-local concept assertion $C(a)$ (denoted $\mathcal{I} \models C(a)$), if $a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}$; it satisfies an i-local role assertion $R(a,b)$ (denoted $\mathcal{I} \models R(a,b)$), if $\langle a^{\mathcal{I}_i}, b^{\mathcal{I}_i}\rangle \in R^{\mathcal{I}_{ii}}$; $\mathcal{I}$ satisfies an object assertion $a \cdot E \cdot b$ (denoted $\mathcal{I} \models a \cdot E \cdot b$), $a \in N_{\mathrm{I}i}$, $b \in N_{\mathrm{I}j}$, $E \in \rho_{ij}$, if $\langle a^{\mathcal{I}_i}, b^{\mathcal{I}_j}\rangle \in E^{\mathcal{I}_{ij}}$. Putting this together, $\mathcal{I}$ satisfies a combined ABox $\mathcal{A}$ (denoted $\mathcal{I} \models \mathcal{A}$) if it satisfied each ABox assertion of each $\mathcal{A}_i \in \mathcal{A}$ and as well each object assertion contained in $\mathcal{A}$.

Finally, a combined interpretation $\mathcal{I}$ over some index set I is a model of a combined knowledge base $\Sigma = \langle \mathcal{K}, \mathcal{R}, \mathcal{A}\rangle$ defined over I (denoted by $\mathcal{I} \models \Sigma$) if $\mathcal{I}$ satisfies all three $\mathcal{K}$, $\mathcal{R}$ and $\mathcal{A}$.

[9] This definition of the semantics of general transitivity axioms differs from that originally given by Cuenca Grau et al. [15] and Parsia & Cuenca Grau [34], however, we believe that this new definition actually suits the intuition behind the general transitivity axiom as it is explained in these works.

1.5.2 Reasoning in $\mathcal{E}$-connections

A key reasoning task for $\mathcal{E}$-connections is the satisfiability of *i*-concepts with respect to a combined knowledge base. Other reasoning tasks such as entailment of subsumption between *i*-concepts are reducible.

Definition 1.11 (Reasoning tasks for $\mathcal{E}$-connections). Given a combined knowledge base Σ over some index set I and $i \in I$, an *i*-concept C is satisfiable with respect to Σ if there exists a combined interpretation $\mathcal{I}$ of Σ that is a model of Σ such that $C^{\mathcal{I}_i} \neq \emptyset$. Given two *i*-concepts C and D, it is said that Σ entails the subsumption $C \sqsubseteq D$ (denoted $\mathcal{I} \models C \sqsubseteq D$) if in each model $\mathcal{I}$ of Σ we have $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}$.

As for regular DL, the entailment of subsumption decision problem is reducible into satisfiability [28, 15].

Theorem 1.2. *Given a combined knowledge base Σ over some index set I, $i \in I$ and two i-concepts C and D, the subsumption formula $C \sqsubseteq D$ is entailed by Σ if and only if the complex i-concept $C \sqcap \neg D$ is unsatisfiable with respect to Σ.*

Other classic decision problem that have been considered for $\mathcal{E}$-connections are the problem of entailment of ABox knowledge [28].

Definition 1.12 (Entailment of ABox formulae). A combined knowledge base Σ entails an *i*-local expression $C(a)$, if in each model $\mathcal{I}$ of Σ we have $a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}$. Σ entails an *i*-local expression $R(a,b)$, if in each model $\mathcal{I}$ of Σ we have $\langle a^{\mathcal{I}_i}, b^{\mathcal{I}_i} \rangle \in R^{\mathcal{I}_{ii}}$.

Tableaux reasoning algorithms for the $\mathcal{E}$-connections languages $\mathcal{C}^{\mathcal{E}}_{\mathcal{HQ}}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$, $\mathcal{C}^{\mathcal{E}}_{\mathcal{HI}}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$, $\mathcal{C}^{\mathcal{E}}_{\mathcal{HQ}+}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$, and $\mathcal{C}^{\mathcal{E}}_{\mathcal{HI}+}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$ were introduced by Cuenca Grau et al. [15] and Parsia & Cuenca Grau [34]. For each of these algorithms, the complexity of deciding the satisfiability of an *i*-concept C with respect to a combined knowledge base Σ is 2NExpTime in the size of C and Σ in the worst case. The Pellet reasoner[10] supports $\mathcal{C}^{\mathcal{E}}_{\mathcal{HQ}}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$ and $\mathcal{C}^{\mathcal{E}}_{\mathcal{HI}}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$ [17, 15]; Pellet's extension towards $\mathcal{C}^{\mathcal{E}}_{\mathcal{HQ}+}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$ and also $\mathcal{C}^{\mathcal{E}}_{\mathcal{HO}+}(\mathcal{SHIQ}, \mathcal{SHOQ}, \mathcal{SHIO})$ is planned [34].

1.5.3 Modeling with $\mathcal{E}$-connections

Since DDL and $\mathcal{E}$-connections take a different modelling perspective, the examples that we present bellow in this section do not intend to remodel precisely the scenario of Sect. 1.4. Instead they aim to demonstrate the practical modeling features offer to us by $\mathcal{E}$-connections.

[10] Pellet OWL DL reasoner is an open-source competitive DL reasoner currently available. Pellet is available though its homepage `http://clarkparsia.com/pellet/`.

Example 1.4. Let us build a combined knowledge base $\mathcal{K}$ about people and their occupation in a modular way. Let us split the modeling domain into three local ontologies:

- People, that contains individuals such as johnSmith, rickDeckard, etc., and concepts such as Man, Woman, Adult, Parent, etc.;
- Organizations, with individuals such as googleInc, operaSA, teatroAllaScala, etc., and concepts such as Enterprise, Theatre, ITCompany, etc.;
- Locations, that contains concepts such as Oslo, Milan, MountainView, Norway, Italy, EU, USA, etc.

We are allowed to model this way, since people clearly are disjoint from institutions and hence no individual will ever be an instance of some concept from People and some other concept from Organizations. The same holds for Locations.

However, in order to introduce yet another local ontology Professions that will contain concepts such as Researcher, Chef or Pilot, we ought to be cautious. We need to think twice about how do we really want to use these concepts: given an individual that represents a person, say johnSmith that comes from the ontology People, do we want this individual to possibly become an instance of one of the concepts in Professions? This kind of modeling is not possible in $\mathcal{E}$-connections. In fact, as showed below, we will be able to keep professions in a separate local ontology, but under certain restrictions.

Let us continue Example 1.4 as a running example in order to illustrate the usage of links.

Example 1.5. Assume the Organizations module contains the following knowledge:

$$\mathsf{ITCompany}(\mathsf{operaSA})$$
$$\mathsf{ITCompany}(\mathsf{googleInc})$$
$$\mathsf{ITCompany} \sqsubseteq \mathsf{Enterprise}$$

As people and institutions are separated between two distinct modules, we will need the link worksAt to indicate employment. It is important to specify the source and the destination ontology for links; the source module for the worksAt link is People and its destination module is Organizations. We also add another link locatedIn between Organizations and Locations in order to indicate where the institutions are located. We do so by introducing the following object assertions:

$$\begin{array}{cc} \mathsf{johnSmith}\cdot\mathsf{worksAt}\cdot\mathsf{googleInc} & \mathsf{googleInc}\cdot\mathsf{locatedIn}\cdot\mathsf{MountainView} \\ \mathsf{rickDeckard}\cdot\mathsf{worksAt}\cdot\mathsf{operaSA} & \mathsf{operaSA}\cdot\mathsf{locatedIn}\cdot\mathsf{Oslo} \end{array}$$

In addition, we employ links in restrictions in order to derive new knowledge within People in form of complex concepts, for instance by including the following GCI axiom:

$$\exists \mathsf{worksAt.Enterprise} \sqsubseteq \mathsf{IndustryEmployee}$$

Using the $\mathcal{E}$-connections semantics, one is able to derive that both johnSmith and rickDeckard are in fact industry employees.

As mentioned above, $\mathcal{E}$-connections require strict separation of modeling domains between modules. Hence if we want to introduce a new local ontology Professions that will contain concepts such as Chef or Pilot this comes at some cost. Particularly, we will not be able to assert in the ABox of People that some its individuals belongs to any of the concepts of Professions. Also, we will not be able to relate directly concepts such as Adult of People with say Policeman of Professions by means of subsumption. In part these issues are overcomed by using links instead of class membership, as we show in the following example. While inter-ontology class membership is ruled by this approach, it possibly yields satisfactory modeling.

Example 1.6. Let us now include a new local ontology Professions containing concepts such as Chef, Pilot, Researcher, Policeman, etc. In order to express that some people belong to certain profession we use the link hasProfession between People and Professions:

$$\begin{gathered}\mathsf{johnSmith} \cdot \mathsf{hasProfession} \cdot \mathsf{Chef} \\ \mathsf{rickDeckard} \cdot \mathsf{hasProfession} \cdot \mathsf{Researcher}\end{gathered}$$

The two more expressive $\mathcal{E}$-connections frameworks $\mathcal{C}^{\mathcal{E}}_{\mathcal{HQ}+}(\mathcal{SHIQ},\mathcal{SHIQ},\mathcal{SHOQ})$ and $\mathcal{C}^{\mathcal{E}}_{\mathcal{HI}+}(\mathcal{SHIQ},\mathcal{SHIQ},\mathcal{SHOQ})$ also include the general transitivity axiom [15, 34]. Taking advantage of the fact that names are freely allowed to be reused for role and link properties, this allows very for flexible transitivity control. In order to demonstrate this feature, we borrow an example that originally appears in these works.

Example 1.7. Let us add some more knowledge into our running example. First, we add to Organizations (indexed by 2) further structure for the Google company:

$$\begin{gathered}\mathsf{partOf}(\mathsf{gCateringDpt}, \mathsf{gEmployeeServicesDiv}) \\ \mathsf{partOf}(\mathsf{gEmployeeServicesDiv}, \mathsf{googleInc})\end{gathered}$$

In addition, for whatever reason we decide to also include knowledge about human body parts in the component ontology. In accordance, we add the following axiom into People (indexed by 1):

$$\mathsf{patOf}(\mathsf{finger47}, \mathsf{johnSmith})$$

Finally, we are also free to use the property patOf as a link, and so we add the following object assertion between these two component ontologies:

$$\mathsf{johnSmith}\cdot\mathsf{partOf}\cdot\mathsf{gCateringDpt}$$

With previous flavours of $\mathcal{E}$-connections we would need to use a separate role name for each component ontology an for each linkbox between each two component modules. In this case we would be forced to use three names.

However, the real utility of this feature only comes in combination with general transitivity axioms. Before these axioms were introduced, $\mathcal{E}$-connections only included the possibility to declare transitive roles inside each component RBox. This would allow us to derive $\mathcal{K} \models \mathsf{patOf}(\mathsf{gCateringDpt},\mathsf{googleInc})$ in our case, if we have previously declared the role partOf transitive in the Organizations component, but not anything more. General transitivity axioms allow us to assert transitivity on the combined relation resulting from the composition of all partOf properties, whether roles or links, e.g., by the following axiom:

$$\mathrm{Trans}(\mathsf{partOf};(1,1),(1,2),(2,2))$$

By including this axiom in $\mathcal{K}$ we derive that $\mathcal{K} \models \mathsf{johnSmith}\cdot\mathsf{partOf}\cdot\mathsf{googleInc}$ which is intuitively justified but as well $\mathcal{K} \models \mathsf{finger47}\cdot\mathsf{partOf}\cdot\mathsf{googleInc}$ which may not be justified for each modeling scenario. However, the transitivity axiom is versatile enough, and we easily rule out the second consequence, if undesired, by replacing the above transitivity axiom by:

$$\mathrm{Trans}(\mathsf{partOf};(1,2),(2,2))$$

$\mathcal{E}$-connections allow to connect several ontologies under the assumption that their modeling domains are separated. Knowledge from one local ontology is reused in the other ontologies thanks to use of link properties. Such a framework is of demand and is easily applicable if one is about to build a new an ontology in a modular way, and from the very beginning clearly perceives how to split the modeling domain into modules. Highly developed reasoning algorithms and readily available reasoning engines make $\mathcal{E}$-connections also available for practical use. A tool that allows to automatically partition a monolithic ontology is also available [18]. On the other hand, the separate domains assumption is a serious obstacle if one wishes to combine several already existing ontologies. In such a case we consider the assumption of separate domains too strict, as with fair probability the modeling domains covered by these existing ontologies overlap to some extent.

1.6 Package-based Description Logics

Package-based Description Logics (P-DL) was originally introduced by Bao & Honavar und/er the name Package-extended Ontologies [7]. Later the framework was developed by Bao et al. [6, 4, 5, 8]. Within this framework, a modular ontology is composed of several packages. Each ontological entity (namely, each individual, concept and role) is assigned its home package – the package it primarily belongs to. Given a package P_i, besides for its home terms, also other terms are free to appear in P_i, even if their home package is different from P_i. These terms are said to be *foreign terms* in P_i, and they are said to be *imported into* P_i.

Similarly to the formalisms presented above, in the semantics of P-DL each packages is interpreted in a local model. The denotation of foreign symbols is related to the denotation of these symbols in their home package. P-DL also include several other modularisation features inspired by modular software engineering [6]. Package nesting is possible, by virtue of which packages are organized into a hierarchy. The package hierarchy in a P-DL ontology constitutes an organizational structure, in contrast to the semantic structure imposed by ontology axioms and importing. In addition P-DL includes scope limitation modifiers which allow ontology engineers to limit the visibility of certain terms from the perspective of packages other than their home package. Most typical examples of these are the three predefined modifiers public, protected and private, however P-DL allows users to introduce their own modifiers.

1.6.1 Formalization

We now proceed by introducing the P-DL framework formally. The review in this section is based on the most recent publication of Bao et al. to date [8]. Here, P-DL are built over the local language $\mathcal{SHOIQ}$ resulting into the P-DL language $\mathcal{SHOIQP}$.

Definition 1.13 (Syntax of $\mathcal{SHOIQP}$ PD-L). A package based ontology is any $\mathcal{SHOIQ}$ ontology $\mathcal{P}$, which partitions into a finite set of packages $\{P_i\}_{i\in I}$, where I is some finite index set, and such that for every term (namely, concept, role, and individual symbol) t of the alphabet of $\mathcal{P}$, there is a unique package, called the *home package* of t.

Given a package-based ontology $\mathcal{P} = \{P_i\}_{i\in I}$, we will use the following terminology:

- the set of home terms of a package $P_i \in \mathcal{P}$ is denoted by Δ_{S_i};
- the index of the home package of a term t is denoted by $\text{home}(t)$;
- if t occurs in P_i then t is a *local term* in P_i if $\text{home}(t) = i$, otherwise it is said to be a *foreign term* in P_i;
- If t is a foreign term in P_j, and $\text{home}(t) = i$, then we write $i \xrightarrow{t} j$;
- $i \rightarrow j$ means that $i \xrightarrow{t} j$ for some t;
- $i \xrightarrow{*} j$ if $i = i_1 \rightarrow i_2 \dots i_{n-1} \rightarrow i_n = j$ for some $i_1,\dots,i_n \in I$;
- $P_j^* = P_j \cup \{P_i | i \xrightarrow{*} j\}$.

When a package P_j imports another package P_i, it imports also the domain of P_i. In order to denote in P_j the domain of an imported package P_i, the language of P-DL introduces the concept symbol $\top_i$. Within the local model of P_j, the denotation of $\top_i$ represents the domain of P_i (it is an image of this domain). P-DL uses also introduces so called *contextualized negation* $\neg_i$, which is a concept constructor applicable within the package P_i and the packages that import P_i. When it occurs in P_i, $\neg_i$ stands for normal concept complement. However, if $\neg_i C$ occurs in P_j is the complement with respect to the domain of P_i as imported into P_j (i.e., within P_j we have $\neg_i C \equiv \top_i \sqcap \neg_j C$).

The semantics of P-DL is also called semantics of importing by Bao et al. [4] due to its ability to import concepts from one ontology to another. In this sense it is similar to the approach of Pan et al. [33] who also investigate semantic imports in distributed ontologies. The P-DL semantics is reminiscent of the DDL semantics in that sense that it also relies on a domain relation which projects interpretation of terms from one package to another. Unlike to the original DDL semantics, strict constraints are placed on the domain relation in P-DL. It is worthwhile to note that some later works on DDL experiment with incorporating some of these restrictions into the DDL framework [24].

Definition 1.14 (Semantics of $\mathcal{SHOIQP}$ P-DL). Given a package-based ontology $\mathcal{P} = \{P_i\}_{i\in I}$, a distributed interpretation of $\mathcal{P}$ is a pair $\mathcal{I} = \langle\{\mathcal{I}_i\}_{i\in I}, \{r_{ij}\}_{i \xrightarrow{*} j}\rangle$ such that each $\mathcal{I}_i = \langle\Delta^{\mathcal{I}_i}, \cdot^{\mathcal{I}_i}\rangle$ is an interpretation of the local package P_i and each $r_{ij} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$ is a domain relation between $\Delta^{\mathcal{I}_i}$ and $\Delta^{\mathcal{I}_j}$. A distributed interpretation $\mathcal{I}$ is a model of $\{P_i\}_{i\in I}$ if the following conditions hold:

1. there is at least one $i \in I$ such that $\Delta^{\mathcal{I}_i} \neq \emptyset$;
2. $\mathcal{I}_i \models P_i$;
3. r_{ij} is an injective partial function, and r_{ii} is the identity function;
4. If $i \xrightarrow{*} j$ and $j \xrightarrow{*} k$, then $r_{ik} = r_{ij} \circ r_{jk}$;
5. if $i \xrightarrow{t} j$, then $r_{ij}(t^{\mathcal{I}_i}) = t^{\mathcal{I}_j}$;
6. if $i \xrightarrow{R} j$, then if $(x,y) \in R^{\mathcal{I}_i}$ and $r_{ij}(x)$ is defined then, also $r_{ij}(y)$ is defined.

In a nutshell, the P-DL semantics is characterized as follows: if two terms t_1 and t_2 appear within some package P_i and they are related by means of axioms in P_i this relation is propagated to any other package P_j where both they also appear, no matter whether one of P_i, P_j is their home package, or they are imported into both P_i and P_j and their home package P_k is different from both P_i and P_k.

1.6.2 Reasoning in P-DL

The three main reasoning tasks for P-DL are consistency of knowledge bases, concept satisfiability and concept subsumption entailment with respect to a knowledge base. In addition, these three decision problems are always defined with respect to a so called witness package P_w of the knowledge base $\mathcal{P}$. When answering the decision problems we only care about the importing closure P_w^* of the witness package and we neglect the rest of the knowledge base. Given two different packages P_1 and P_2, the answer to these decision problems may be different when witnessed by P_1 as it is when witnessed by P_2.

Definition 1.15 (Reasoning tasks for P-DL). A package-based ontology $\mathcal{P}$ is consistent as witnessed by a package P_w of $\mathcal{P}$, if there exists a model $\mathcal{I}$ of P_w^* such that $\Delta^{\mathcal{I}_w} \neq \emptyset$. A concept C is satisfiable as witnessed by a package P_w of $\mathcal{P}$, if there exists a model $\mathcal{I}$ of P_w^* such that $C^{\mathcal{I}_w} \neq \emptyset$. A subsumption formula $C \sqsubseteq D$ is valid as witnessed by a package P_w of $\mathcal{P}$ (denoted $\mathcal{P} \models C \sqsubseteq_w D$), if for every model $\mathcal{I}$ of P_w^* we have $C^{\mathcal{I}_w} \subseteq D^{\mathcal{I}_w}$.

While not formally introduced in by Bao et al. [8] we analogously define also ABox formulae entailment, as follows.

Definition 1.16 (ABox reasoning for P-DL). A formula $C(a)$ is valid as witnessed by a package P_w of $\mathcal{P}$, if for every model $\mathcal{I}$ of P_w^* we have $a^{\mathcal{I}_w} \in C^{\mathcal{I}_w}$. A formula $R(a,b)$ is valid as witnessed by a package P_w of $\mathcal{P}$, if for every model $\mathcal{I}$ of P_w^* we have $\langle a^{\mathcal{I}_w}, b^{\mathcal{I}_w}\rangle \in R^{\mathcal{I}_w}$.

A distributed reasoning algorithm for the P-DL language $\mathcal{ALCP}_\mathcal{C}$, a sub-language of $\mathcal{SHOIQP}$ which is build on top the local DL $\mathcal{ALC}$ and only allows for importing of concepts, was given by Bao et al. [5]. To our best knowledge, no implementation is known. The decidability and the computational complexity of reasoning for the full expressive P-DL language $\mathcal{SHOIQP}$ are shown via a reduction of a package-based ontology into a standard DL knowledge base. $\mathcal{SHOIQP}$ is decidable and the computational complexity for the decision problems is NExpTime-complete [8].

1.6.3 Modeling with P-DL

The $\mathcal{E}$-connections users will find P-DL familiar. In fact, one is able to model in P-DL in a very similar fashion. In order to illustrate this, we remodel Example 1.4 in P-DL.

Example 1.8. Let us build a package based ontology $\{P_1, P_2, P_3\}$ where P_1 will contain knowledge about people, P_2 about organizations, and P_3 about locations. Let us fix the home terms for these packages as follows: $\Delta_{S_1} = \{\mathsf{Person}, \mathsf{Man}, \mathsf{Woman}, \mathsf{Child}, \mathsf{Adult}, \mathsf{hasChild}, \mathsf{IndustryEmployee}, \mathsf{worksAt}, \mathsf{johnSmith}, \mathsf{rickDeckard}\}$, $\Delta_{S_2} = \{\mathsf{ITCompany}, \mathsf{Enterprise}, \mathsf{Theatre}, \mathsf{googleInc}, \mathsf{operaSA}\}$, and $\Delta_{S_3} = \{\mathsf{Oslo}, \mathsf{Milan}, \mathsf{MountainView}, \mathsf{Norway}, \mathsf{Italy}, \mathsf{EU}, \mathsf{USA}\}$.

It shows that we are able to mimic the conceptual modeling that we have done in case of $\mathcal{E}$-connections. In this respect we put into P_1 the following axioms:

$$\begin{aligned} \mathsf{Child} &\sqsubseteq \neg\mathsf{Adult} \\ \mathsf{Woman} &\sqsubseteq \neg\mathsf{Man} \\ \mathsf{Parent} &\sqsubseteq \exists\mathsf{hasChild}.\mathsf{Person} \\ \exists\mathsf{worksAt}.\mathsf{Enterprise} &\sqsubseteq \mathsf{IndustryEmployee} \end{aligned}$$

Since worksAt now turns into a regular role with its home package being P_1, role assertions are used to express where our P_1-individuals work. In accordance we add the following axioms into P_1:

$$\mathsf{worksAt}(\mathsf{johnSmith},\mathsf{googleInc})$$
$$\mathsf{worksAt}(\mathsf{rickDeckard},\mathsf{operaSA})$$

In the TBox of P_2 we put:

$$\mathsf{ITCompany} \sqsubseteq \mathsf{Enterprise}$$

And in the P_2's ABox we put

$$\begin{array}{ll} \mathsf{ITCompany}(\mathsf{operaSA}) & \mathsf{locatedIn}(\mathsf{operaSA},\mathsf{Oslo}) \\ \mathsf{ITCompany}(\mathsf{googleInc}) & \mathsf{locatedIn}(\mathsf{googleInc},\mathsf{MountainView}) \end{array}$$

Notice that in the last axiom of P_1 we have build the 1-concept $\exists\mathsf{worksAt}.\mathsf{Enterprise}$ in a very similar fashion as we would do in $\mathcal{E}$-connections. The only foreign term that comes into play here is Enterprise. The fact that worksAt in not a link but a 1-role here is only a minor difference from $\mathcal{E}$-connections.

The semantics of P-DL allows us to derive that both individuals johnSmith and rickDeckard are instances of the concept IndustryEmployee. This does not follow from P_1 directly, distributed reasoning is required.

P-DL offers more freedom in modeling however. Let us remodel Example 1.8 in a more intuitive fashion. First of all it is not necessary to keep the concept IndustryEmployee between the home terms of P_1. We are free to put into P_4 that will deal with professions and still we will be able to have 1-individuals (representing persons) as its instances. Thus also, if we want to assert that johnSmith is a Chef we are now free to do it using a concept assertion, we do not need a separate role hasProfession. In addition, the role worksAt connects persons which belong to P_1 and organizations which belong to P_2, however if we intuitively see it as a part of the modeling domain of P_2, we are free to put it there. The remodeled example is as follows.

Example 1.9. Consider $\{P_1,P_2,P_3\}$ such that $\Delta_{S_1} = \{\mathsf{Person},\mathsf{Man},\mathsf{Woman},\mathsf{Child},\mathsf{Adult},\mathsf{hasChild}\}$, $\Delta_{S_2} = \{\mathsf{ITCompany},\mathsf{Enterprise},\mathsf{Theatre},\mathsf{worksAt}\}$ and Δ_{S_3} is same as in Example 1.8. In addition there is now a new package P_4 with home terms $\Delta_{S_4} = \{\mathsf{Chef},\mathsf{Pilot},\mathsf{Researcher},\ \mathsf{Policeman},\mathsf{IndustryEmployee}\}$

We will keep in P_1 The part of the knowledge that deals with the home terms thereof:

$$\begin{aligned} \mathsf{Child} &\sqsubseteq \neg\mathsf{Adult} \\ \mathsf{Woman} &\sqsubseteq \neg\mathsf{Man} \\ \mathsf{Parent} &\sqsubseteq \exists\mathsf{hasChild}.\mathsf{Person} \end{aligned}$$

In the ABox of P_1 we put the following assertions:

$$\mathsf{worksAt}(\mathsf{johnSmith}, \mathsf{googleInc}) \qquad \mathsf{Chef}(\mathsf{johnSmith})$$
$$\mathsf{worksAt}(\mathsf{rickDeckard}, \mathsf{operaSA}) \qquad \mathsf{Researcher}(\mathsf{rickDeckard})$$

The package P_2 is without any change:

$$\mathsf{ITCompany} \sqsubseteq \mathsf{Enterprise}$$

And in the ABox:

$$\mathsf{ITCompany}(\mathsf{operaSA}) \qquad \mathsf{locatedIn}(\mathsf{operaSA}, \mathsf{Oslo})$$
$$\mathsf{ITCompany}(\mathsf{Google}) \qquad \mathsf{locatedIn}(\mathsf{googleInc}, \mathsf{MountainView})$$

The last axiom of P_1 is now moved into P_4 as it has to do with definition of the concept IndustryEmployee which now belongs to P_4. This package contains one new axiom in addition:

$$\exists \mathsf{worksAt}.\mathsf{Enterprise} \sqsubseteq \mathsf{IndustryEmployee}$$
$$\mathsf{Policeman} \sqsubseteq \mathsf{Adult}$$

Even if both worksAt and Enterprise are 2-terms, the concept $\exists\mathsf{worksAt}.\mathsf{Enterprise}$ is allowed on the left hand side of GCI in P_4. The second axiom of P_4 directly uses the 1-concept Adult on its right hand side. Even if we have decided to split the modeling domain into the domain of persons in P_1 and the domain of professions in P_4, we were able to relate the concepts Policeman and Adult by a GCI even if each of them belongs to a different home package. Recall from our conclusion from Sect. 1.5, that this is not possible in $\mathcal{E}$-connections.

As we have seen, the prominent feature of P-DL is term importing which allows us to freely reuse in any local ontology also terms defined in some other part of the system together with the knowledge associated with these terms. This makes P-DL an attractive formalism for modelling distributed ontologies. The reasoning algorithm is, however, known only for a rather limited case of package-based $\mathcal{ALC}$ with only concept imports allowed. Also no implementation of a P-DL reasoner is available, a serious obstacle towards its practical use.

1.7 Integrated Distributed Description Logics

Integrated Distributed Description Logics (IDDL) was introduced by Zimmerman [41] with the main motivation of overcoming some of the limitations of the already existing formalisms for ontology mapping. IDDL was particularly designed to support reasoning about ontology mappings and mapping composition. The basic intuition of IDDL is to represent mappings in a separate logical language, on which it is possible to define a calculus, that allows to determine logical consequence between mappings. A second motivation for the introduction of IDDL was the fact that DDL,

$\mathcal{E}$-connections and P-DL considers ontology mappings as a method linking the items of a source ontology (concepts, roles and individuals) with the items of a target ontology, and they are expressed from the perspective of the target ontology. In the vision of IDDL, mappings are semantic relations between items of different ontologies, stated form an external perspective and so they do not distinguish between a source and a target ontology. This vision is much more similar to peer-to-peer information integration approach introduced by Ullman [39] and further formalized by Calvanese et al. [11], in which mappings are represented as implication formulas of two data sources.

1.7.1 Formalization

Theoretically IDDL is build on top of an underlying DL that employs basic $\mathcal{ALC}$ features, plus qualified number restrictions, nominals, role complement, union, intersection and composition, inverse roles and the transitive-reflexive closure role constructor [41, 42] (i.e., this language extends $\mathcal{SHOIQ}_\mathrm{b}$ introduced in Sect. 1.2 with composition of roles and role transitive-reflexive closure constructors). This language is known to be undecidable [3]. It has been showed, however, that reasoning in IDDL is decidable as long as each local KB uses a sub-language which is decidable [42].

Definition 1.17 (Syntax of IDDL). An IDDL knowledge base, called distributed system (DS), is a pair $\langle \mathbf{O}, \mathbf{A} \rangle$ where $\mathbf{O} = \{O_i\}_{i \in I}$ is a set of DL ontologies and $\mathbf{A} = \{A_{ij}\}_{i,j \in I}$ is a family of alignments, each A_{ij} consisting of correspondence axioms of the following six possible forms:

$$\begin{array}{ll} i : C \overset{\sqsubseteq}{\longleftrightarrow} j : D & \qquad i : R \overset{\sqsubseteq}{\longleftrightarrow} j : S \\ i : C \overset{\perp}{\longleftrightarrow} j : D & \qquad i : R \overset{\perp}{\longleftrightarrow} j : S \\ i : a \overset{\in}{\longleftrightarrow} j : C & \qquad i : a \overset{=}{\longleftrightarrow} j : b \end{array}$$

where C, D are concepts, R, S, are roles, a, b are individuals, and $k : \phi$ means that the expression ϕ belongs to the local ontology O_i and conforms to its local language. From left to right from top to bottom the above correspondence axioms are called cross-ontology concept subsumption, cross-ontology role subsumption, cross-ontology concept disjointness, cross-ontology role disjointness, cross-ontology membership and cross-ontology identity.

In order to simplify the notation we add a syntactic shorthand $i : X \overset{\equiv}{\longleftrightarrow} j : Y$, representing the pair of cross-ontology subsumptions $i : X \overset{\sqsubseteq}{\longleftrightarrow} j : Y$, $j : Y \overset{\sqsubseteq}{\longleftrightarrow} i : X$, where X, Y are either both concepts or both roles.

Definition 1.18 (Semantics of IDDL). Given a distributed system $S = \langle \mathbf{O}, \mathbf{A} \rangle$ over an index set I, a distributed interpretation of S is a pair $\langle \mathbf{I}, \varepsilon \rangle$ where $\mathbf{I} = \{\mathcal{I}_i\}_{i \in I}$ is a family of local interpretations such that each $\mathcal{I}_i = \langle \Delta^{\mathcal{I}_i}, \cdot^{\mathcal{I}} \rangle$ is an interpretation of $\mathcal{O}_i$ in its own language with a non-empty local domain $\Delta^{\mathcal{I}_i}$, and the equalizing function $\varepsilon = \{\varepsilon_i\}_{i\in I}$ is a family of functions $\varepsilon_i : \Delta^{\mathcal{I}_i} \to \Delta_\varepsilon$ that map all elements of each local domain $\Delta^{\mathcal{I}_i}$ into a single global domain Δ_ε[11].

A distributed interpretation $\mathcal{I} = \langle \mathbf{I}, \varepsilon \rangle$ satisfies a local axiom $i : \phi \in O_i$ (denoted by $\mathcal{I} \models_{\mathrm{d}} i : \phi$) if ϕ is satisfied by $\mathcal{I}_i$ in according to its local DL language. A local ontology $\mathcal{O}_i$ is satisfied by $\mathcal{I}$ (denoted $\mathcal{I} \models_{\mathrm{d}} O_i$) if each its axiom $i : \phi \in O_i$ is satisfied by $\mathcal{I}$. A correspondence axiom ψ is satisfied by $\mathcal{I}$ depending on its type as follows:

$$
\begin{array}{lcl}
\mathcal{I} \models_{\mathrm{d}} i : C \xleftrightarrow{\sqsubseteq} j : D & \text{if} & \varepsilon_i\left(C^{\mathcal{I}_i}\right) \subseteq \varepsilon_j\left(D^{\mathcal{I}_j}\right) \\
\mathcal{I} \models_{\mathrm{d}} i : R \xleftrightarrow{\sqsubseteq} j : S & \text{if} & \varepsilon_i\left(R^{\mathcal{I}_i}\right) \subseteq \varepsilon_j\left(S^{\mathcal{I}_j}\right) \\
\mathcal{I} \models_{\mathrm{d}} i : C \xleftrightarrow{\perp} j : D & \text{if} & \varepsilon_i\left(C^{\mathcal{I}_i}\right) \cap \varepsilon_j\left(D^{\mathcal{I}_j}\right) = \emptyset \\
\mathcal{I} \models_{\mathrm{d}} i : R \xleftrightarrow{\perp} j : R & \text{if} & \varepsilon_i\left(R^{\mathcal{I}_i}\right) \cap \varepsilon_j\left(S^{\mathcal{I}_j}\right) = \emptyset \\
\mathcal{I} \models_{\mathrm{d}} i : a \xleftrightarrow{\in} j : C & \text{if} & \varepsilon_i\left(a^{\mathcal{I}_i}\right) \in \varepsilon_j\left(D^{\mathcal{I}_j}\right) \\
\mathcal{I} \models_{\mathrm{d}} i : a \xleftrightarrow{=} j : b & \text{if} & \varepsilon_i\left(a^{\mathcal{I}_i}\right) = \varepsilon_j\left(b^{\mathcal{I}_j}\right)
\end{array}
$$

An alignment A_{ij} is satisfied by $\mathcal{I}$ (denoted $\mathcal{I} \models_{\mathrm{d}} A_{ij}$) if for each correspondence axiom $\psi \in A_{ij}$ we have $\mathcal{I} \models_{\mathrm{d}} \psi$.

Given an index set I, a distributed system $S = \langle \mathbf{O}, \mathbf{A} \rangle$ and a distributed interpretation $\mathcal{I} = \langle \mathbf{I}, \varepsilon \rangle$ both over I, we say that $\mathcal{I}$ is a model of S (denoted by $\mathcal{I} \models_{\mathrm{d}} S$) if for each $i \in I$ we have $\mathcal{I} \models_{\mathrm{d}} O_i$ and for each pair $i, j \in I$ we have $\mathcal{I} \models_{\mathrm{d}} A_{ij}$.

1.7.2 *Reasoning in IDDL*

The main reasoning task described for IDDL is consistency checking for distributed systems. Since the definition of a consistent distributed system is not given by Zimmerman & Duc [41, 42], we assume the usual definition, which is as follows.

Definition 1.19 (DS consistency in IDDL). A distributed system S is consistent if there exists a distributed interpretation $\mathcal{I}$ that is a model of S.

The other reasoning tasks for IDDL are local formula entailment and also correspondence formula entailment.

Definition 1.20 (Other reasoning tasks in IDDL). A formula ϕ (either a local formula or an correspondence formula) is entailed by S (denoted $S \models_{\mathrm{d}} \phi$) if for each distributed interpretation $\mathcal{I}$ that is a model of S it holds that $\mathcal{I} \models_{\mathrm{d}} \phi$.

[11] Each of ε_i is a function, hence it certainly assigns to each element $x \in \Delta^{\mathcal{I}_i}$ some element $y \in \Delta_\varepsilon$. As the local domain $\Delta^{\mathcal{I}_i}$ is required to be nonempty for each $i \in I$, it follows that the global domain Δ_ε is also nonempty.

All the other reasoning tasks are reducible into distributed system consistency checking [42]. A sound and complete algorithm for consistency checking of IDDL distributed systems with alignments limited to cross-ontology concept subsumption, disjointness and role subsumption was given by Zimmerman & Duc [42]. This reasoning algorithm is based on a reduction of a distributed system into a set of independent ontologies which are all consistent if and only if the original distributed system is consistent. Worst-case complexity of this procedure is 3ExpTime [42].

An attempt to provide an axiomatization of the effects of the correspondences in terms of propagation rules, similar to the rules presented for DDL, is presented in [41]. The completeness of these rules is still an open problem.

1.7.3 Modeling with IDDL

IDDL is particularly intended for reasoning about ontology mapping. In IDDL the mapping is not primarily viewed as a mean of knowledge transfer between ontologies, instead it is viewed as an integrating element that forms the global semantics of a distributed system on top of the local semantics of each of the components thereof. As such, the mapping is bi-directional – a unique feature among the frameworks part of this review. This poses some implications on the way how one models with IDDL. In order to demonstrate this we remodel in part of the scenario created in Examples 1.1–1.3.

Example 1.10. In our example company, three ontologies are used. The first one is O_1, the ontology of employees. An excerpt:

$$\begin{array}{rr} \mathsf{ITStaff} \sqsubseteq \mathsf{Employee} & \mathsf{SupportStaff} \sqsubseteq \mathsf{Employee} \\ \mathsf{CateringStaff} \sqsubseteq \mathsf{SupportStaff} & \mathsf{AdministrativeStaff} \sqsubseteq \mathsf{SupportStaff} \\ \mathsf{Cook} \sqsubseteq \mathsf{CateringStaff} & \mathsf{Chef} \sqsubseteq \mathsf{Cook} \end{array}$$

There is also O_2, the brigade de cuisine ontology of kitchen jobs. An excerpt:

$$\begin{array}{rr} \mathsf{KitchenChef} \sqsubseteq \mathsf{Cook} & \mathsf{Saucemaker} \sqsubseteq \mathsf{Cook} \\ \mathsf{PastryCook} \sqsubseteq \mathsf{Cook} & \mathsf{Baker} \sqsubseteq \mathsf{PastryCook} \end{array}$$

And in addition the accounting department prefers to model things according to their own taste, so there is O_3, the accounting ontology. An excerpt:

$$\begin{array}{rr} \mathsf{Accountant} \sqsubseteq \mathsf{Employee} & \mathsf{Director} \sqsubseteq \mathsf{Employee} \\ \mathsf{SeniorExecutive} \sqsubseteq \mathsf{Employee} & \mathsf{JuniorExecutive} \sqsubseteq \mathsf{Employee} \\ \mathsf{BasicStaff} \sqsubseteq \mathsf{Employee} & \mathsf{SupportStaff} \sqsubseteq \mathsf{Employee} \end{array}$$

With IDDL we are able to integrate these three ontologies into a DS $S = \langle \mathbf{O} = \{O_1, O_2, O_3\}, \mathbf{A} \rangle$ using the alignment $\mathbf{A}$ as follows:

$$2 : \mathsf{KitchenChef} \overset{\sqsubseteq}{\longleftrightarrow} 1 : \mathsf{Chef} \qquad 2 : \mathsf{Cook} \overset{\equiv}{\longleftrightarrow} 1 : \mathsf{Cook}$$

$$3 : \mathsf{Employee} \overset{\equiv}{\longleftrightarrow} 1 : \mathsf{Employee} \qquad 3 : \mathsf{Accountant} \overset{\sqsubseteq}{\longleftrightarrow} 1 : \mathsf{AdministrativeStaff}$$

$$1 : \mathsf{ITStaff} \overset{\sqsubseteq}{\longleftrightarrow} 3 : \mathsf{BasicStaff} \qquad 1 : \mathsf{SupportStaff} \overset{\equiv}{\longleftrightarrow} 3 : \mathsf{SupportStaff}$$

In the DS S we now derive for instance $S \models_{\mathrm{d}} 2 : \mathsf{Baker} \overset{\sqsubseteq}{\longleftrightarrow} 1 : \mathsf{CateringStaff}$ and as well $S \models_{\mathrm{d}} 2 : \mathsf{Baker} \overset{\sqsubseteq}{\longleftrightarrow} 3 : \mathsf{SupportStaff}$. In fact, the entailed relation $S \models_{\mathrm{d}} i : X \overset{\sqsubseteq}{\longleftrightarrow} j : Y$ provides us with a global view, i.e., ontological organization of all concepts regardless of what ontology they originally belonged to.

Besides for mapping between concepts, IDDL allows to map between roles, in a very analogous fashion. We move on and demonstrate working with individuals in IDDL in the following example.

Example 1.11. After asserting in O_1 the following axiom:

$$\mathsf{Cook}(\mathsf{johnSmith})$$

and asserting the following correspondence:

$$1 : \mathsf{johnSmith} \overset{=}{\longleftrightarrow} 3 : \mathsf{e006B3F}$$

we are immediately able to derive as a consequence the cross-ontology membership $S \models_{\mathrm{d}} 3 : \mathsf{e006B3F} \overset{\in}{\longleftrightarrow} 1 : \mathsf{CateringStaff}$. In addition, if we want to further indicate the specialization of this employee (a 1-individual), it is not necessary to copy the desired target concept from O_2 to O_1, instead we use the cross-ontology membership correspondence as an axiom. We add:

$$1 : \mathsf{johnSmith} \overset{\in}{\longleftrightarrow} 3 : \mathsf{Baker}$$

Sometimes it is handy to indicate the disjointness of concepts (also roles) across ontologies. IDDL provides us with means to do it.

Example 1.12. Since the company does not specialize in catering services but in IT instead, we add the following cross-ontology disjointness:

$$1 : \mathsf{CateringStaff} \overset{\perp}{\longleftrightarrow} 3 : \mathsf{Director}$$

We are immediately able to derive both $S \models_{\mathrm{d}} 1 : \mathsf{johnSmith} \overset{\in}{\longleftrightarrow} 3 : \neg\mathsf{Director}$ and $S \models_{\mathrm{d}} 3 : \mathsf{e006B3F} \overset{\in}{\longleftrightarrow} 3 : \neg\mathsf{Director}$ which are consequences on the global semantics.

It also sometimes happens in IDDL that the global semantics resulting from the mapping influences the local semantics of a particular component ontology. For instance, in our example the entailed consequence $S \models_{\mathrm{d}} 3 : \mathsf{e006B3F} \overset{\in}{\longleftrightarrow} 3 : \neg\mathsf{Director}$ causes that within $\mathcal{O}_3$ we derive $s \models_{\mathrm{d}} 3 : \neg\mathsf{Director}(\mathsf{e006B3F})$.

As we have seen, IDDL is especially suitable for ontology integration, representing mapping and reasoning about mapping. In contrast, the other approaches concentrate on reuse of knowledge, for instance, when building a new ontology one may reuse some ontologies that already exist. DDL, $\mathcal{E}$-connections and P-DL are more suitable for this task than IDDL because of two reasons; first, in IDDL the mapping is bi-directional, and hence if ontology O_1 wishes to reuse ontology O_2 also O_2 is affected by O_1; and second, IDDL creates a global view which is rather centralized and not distributed.

A typical application scenario for IDDL is for instance reasoning about mapping computed by one of the ontology matching algorithms [19]. These algorithms typically produce bi-directional mapping and the main reasoning task here would be determining the consistency of the resulting integrated ontology.

1.8 Conclusion

This chapter gives a comparative presentation of the most important logical representation frameworks for modular ontologies. Above all we have concentrated on expressivity and modeling with these formalisms: by which practical features the modular ontology development is supported in each of the formalisms and how one is able to use these features in order to model ontologies in a modular fashion. We summarize our observation as follows.

- Distributed Description Logics allow for distributed ontologies in which the component modules are connected by directional semantic mapping. With this mapping, concepts, roles and individuals from distinct components are semantically associated. This allows for reuse of knowledge between the component ontologies. This reuse is directed, that is, if ontology O_1 reuses ontology O_2, the latter is not affected at all. DDL is able to cope with partially overlapping modeling domains and as well with inconsistency that appears locally in one of the component modules.
- $\mathcal{E}$-connections allow to combine several component ontology modules under the assumption that the local modeling domains are strictly disjoint. In $\mathcal{E}$-connections, one connects ontologies with links, practically inter-ontology roles. This framework is especially applicable when building a complex ontology in a modular fashion, and from the beginning it is understood how the local modeling domains should be separated. $\mathcal{E}$-connections also provide an elaborate transitivity control for combinations of local and inter-ontology roles.
- Package-based Description Logic allow to combine several ontology modules by semantic importing of ontology entities, that is, concepts, roles and individuals. Semantically, P-DL overcomes some of the limitations of the other approaches and improves their expressivity.
- Integrated Distributed Description Logics allow for integration of several ontologies with bi-directional semantic mapping, while maintaining both the local view (of each component ontology) and as well the global view (the result of

integration). While the previous approaches aim at ontology combination and reuse, IDDL aims more at ontology integration. Typical application for IDDL is reasoning about automatically computed ontology mapping.

All four frameworks feature distributed reasoning algorithms, however, only for DDL and $\mathcal{E}$-connections the reasoning support is available in form of practical implementations (DDL reasoner DRAGO and $\mathcal{E}$-connections support in the reasoner Pellet). To our best knowledge, no practical implementations of reasoners for the other two frameworks is known, which leaves them, for the time being, at the level of theoretical proposals without practical applicability.

This chapter is not the first attempt for comparison of modular ontology representation frameworks. Wang et al. propose a first qualitative comparison of the different approaches [40]. More formal approach to comparison of the expressivity of the frameworks also exits. First formal comparison between expressive power of DDL and $\mathcal{E}$-connections was carried out by Kutz et al. [28]; Cuenca Grau & Kutz [14] compare DDL, $\mathcal{E}$-connections and P-DL, and in addition they compare these three approaches with the approach that combines ontologies by simple union under specific conditions as discussed in Sect. 1.1; and Bao et al. [4] claim that DDL with mapping on concepts and $\mathcal{E}$-connections are strictly less expressive than Package-based Description Logics. Also, all four approaches are reducible into regular Description Logics [10, 14, 8, 42].

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press (2003)
2. Baader, F., Lutz, C., Sturm, H., Wolter, F.: Fusions of description logics and abstract description systems. J. Artif. Intell. Res. (JAIR) **16**, 1–58 (2002)
3. Baader, F., Sattler, U.: Number restrictions on complex roles in description logics. In: Procs. of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96). Morgan Kaufmann (1996)
4. Bao, J., Caragea, D., Honavar, V.: On the semantics of linking and importing in modular ontologies. In: Procs. of ISWC 2006, *LNCS*, vol. 4273. Springer (2006)
5. Bao, J., Caragea, D., Honavar, V.G.: A tableau-based federated reasoning algorithm for modular ontologies. In: Procs. of IEEE/WIC/ACM International Conference on Web Intelligence (WI'06). IEEE Press (2006)
6. Bao, J., Caragea, D., Honavar, V.G.: Towards collaborative environments for ontology construction and sharing. In: Procs. of the 2006 International Symposium on Collaborative Technologies and Systems. Las Vegas, Nevada, USA (2006)
7. Bao, J., Honavar, V.: Ontology language extensions to support collaborative ontology building. In: 3rd International Semantic Web Conference (ISWC2004) (2004)
8. Bao, J., Voutsadakis, G., Slutzki, G., Honavar, V.: Package-based description logics. In: Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization, *LNCS*, vol. 5445, pp. 349–371. Springer (2009)
9. Borgida, A., Serafini, L.: Distributed description logics. In: I. Horrocks, S. Tessaris (eds.) Proceedings of the 2002 Intl. Workshop on Description Logics (DL2002). CEUR-WS, Toulouse (2002)

10. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. Journal of Data Semantics **1**, 153–184 (2003)
11. Calvanese, D., Giacomo, G.D., Lenzerini, M., Rosati, R.: Logical foundations of peer-to-peer data integration. In: PODS, pp. 241–251 (2004)
12. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: A logical framework for modularity of ontologies. In: IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007, pp. 298–303 (2007)
13. Cuenca Grau, B., Horrocks, I., Kutz, O., Sattler, U.: Will my ontologies fit together? In: Procs. of the 2006 International Workshop on Description Logics (DL'06), pp. 175–182 (2006)
14. Cuenca Grau, B., Kutz, O.: Modular ontology languages revisited. In: SWeCKa 2007: Proc. of the IJCAI-2007 Workshop on Semantic Web for Collaborative Knowledge Acquisition , Hyderabad, India, January 7, 2007 (2007)
15. Cuenca Grau, B., Parsia, B., Sirin, E.: Tableaux algorithms for $\mathcal{E}$-connections of description logics. Tech. rep., University of Maryland Institute for Advanced Computer Studies (2004)
16. Cuenca Grau, B., Parsia, B., Sirin, E.: Working with multiple ontologies on the semantic web. In: Procs. of ISWC2004, *LNCS*, vol. 3298. Springer (2004)
17. Cuenca Grau, B., Parsia, B., Sirin, E.: Combining OWL ontologies using $\mathcal{E}$-connections. Journal of Web Semantics **4**(1), 40–59 (2006)
18. Cuenca Grau, B., Parsia, B., Sirin, E., Kalyanpur, A.: Modularity of web ontologies. In: Proc. of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006), pp. 198–209. AAAI Press (2006)
19. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer (2007)
20. Ghidini, C., Serafini, L., Tessaris, S.: On relating heterogeneous elements from different ontologies. In: Procs. of the Sixth International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT'07), *LNCS*, vol. 4635. Springer (2007)
21. Ghidini, C., Serafini, L., Tessaris, S.: Complexity of reasoning with expressive ontology mappings. In: FOIS, pp. 151–163 (2008)
22. Ghilardi, S., Lutz, C., Wolter, F.: Did I damage my ontology? A case for conservative extensions in description logics. In: Procs. of the Tenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2006), pp. 187–197. AAAI Press (2006)
23. Homola, M.: Distributed description logics revisited. In: Procs. of the 20th International Workshop on Description Logics (DL-2007), *CEUR-WS*, vol. 250 (2007)
24. Homola, M., Serafini, L.: Towards distributed tableaux reasoning procedure for DDL with increased subsumption propagation between remote ontologies. In: Advances in Ontologies, Procs. of Knowledge Representation Ontology Workshop (KROW 2008), *CRPIT*, vol. 90. Australian Computer Society (2008)
25. Horrocks, I., Sattler, U.: A tableaux decision procedure for $\mathcal{SHOIQ}$. In: Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05), pp. 448–453 (2005)
26. Kutz, O.: $\mathcal{E}$-connections and logics of distance. Ph.D. thesis, University of Liverpool, UK (2004)
27. Kutz, O., Lutz, C., Wolter, F., Zakharyaschev, M.: $\mathcal{E}$-connections of description logics. In: Procs. of the 2003 International Workshop on Description Logics (DL2003), *CEUR-WS*, vol. 81 (2003)
28. Kutz, O., Lutz, C., Wolter, F., Zakharyaschev, M.: $\mathcal{E}$-connections of abstract description systems. Artificial Intelligence **156**(1), 1–73 (2004)
29. Kutz, O., Wolter, F., Zakharyaschev, M.: Connecting abstract description systems. In: Proceedings of the 8th Conference on Principles of Knowledge Representation and Reasoning (KR 2002), pp. 215–226. Morgan Kaufmann (2002)
30. Lutz, C., Walther, D., Wolter, F.: Conservative extensions in expressive description logics. In: IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007, pp. 453–458 (2007)
31. McGuinness, D.L., van Harmelen, F. (eds.): OWL Web Ontology Language Overview. A W3C Recommendation. Available online, at `http://www.w3.org/TR/owl-features/`

32. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C. (eds.): OWL 2 Web Ontology Language Profiles. A W3C Candidate Recommendation. Available online, at `http://www.w3.org/TR/owl2-profiles/`
33. Pan, J.Z., Serafini, L., Zhao, Y.: Semantic import: An approach for partial ontology reuse. In: P. Haase, V. Honavar, O. Kutz, Y. Sure, A. Tamilin (eds.) Procs. of the 1st International Workshop on Modular Ontologies (WoMo-06), *CEUR WS*, vol. 232. Athens, Georgia, USA (2006)
34. Parsia, B., Cuenca Grau, B.: Generalized link properties for expressive $\mathcal{E}$-connections of description logics. In: Proc. of the Twentieth National Conference on Artificial Intelligence (AAAI-2005). AAAI, Pittsburgh, Pennsylvania, USA (2005)
35. Serafini, L., Borgida, A., Tamilin, A.: Aspects of distributed and modular ontology reasoning. In: Procs. of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), pp. 570–575 (2005)
36. Serafini, L., Tamilin, A.: Local tableaux for reasoning in distributed description logics. In: Procs. of the 2004 International Workshop on Description Logics (DL2004), *CEUR-WS*, vol. 104 (2004)
37. Serafini, L., Tamilin, A.: Distributed instance retrieval in heterogeneous ontologies. In: Proc. of the 2nd Italian Semantic Web Workshop Semantic Web Applications and Perspectives (SWAP'05). Trento, Italy (2006)
38. Tobies, S.: Complexity results and practical algorithms for logics in knowledge representation. Ph.D. thesis, RWTH Aachen, Germany (2001)
39. Ullman, J.D.: Information integration using logical views. In: F.N. Afrati, P.G. Kolaitis (eds.) Database Theory - ICDT '97, 6th International Conference, Delphi, Greece, January 8-10, 1997, Proceedings, *Lecture Notes in Computer Science*, vol. 1186, pp. 19–40. Springer (1997)
40. Wang, Y., Bao, J., Haase, P., Qi, G.: Evaluating formalisms for modular ontologies in distributed information systems. In: Web Reasoning and Rule Systems, First International Conference, RR 2007, Proceedings, *Lecture Notes in Computer Science*, vol. 4524, pp. 178–193. Springer Berlin / Heidelberg (2007)
41. Zimmermann, A.: Integrated distributed description logics. In: Procs. of 20th International Workshop on Description Logics (DL-2007), *CEUR WS*, vol. 250 (2007)
42. Zimmermann, A., Le Duc, C.: Reasoning on a Network of Aligned Ontologies. In: D. Calvanese, G. Lausen (eds.) Web Reasoning and Rule Systems, Second International Conference, RR 2008, Karlsruhe, Germany, October/November 2008, Proceedings, *Lecture Notes in Computer Science*, vol. 5341, pp. 43–57. Springer (2008)