Overall Progress:

Bachelor's Thesis

Student Consulting Organizations

A Domain Ontology

Felix Förtsch
3708438

01.04.2020

Leipzig University

Abstract

This work develops a domain ontology for Student Consulting Organizations (SCOs). The model declares the domain knowledge and defines its vocabulary. It contains the information necessary to establish or run such an organization in a university context. Additionally it allows for optimization in existing organizations and contributes to cooperation between SCOs by organizing the existing knowledge. It maximizes the use of vocabularies, relations, and classes from established ontologies like Friend of a Friend (FOAF), Financial Industry Business Ontology (FIBO), General Formal Ontology (GFO), and GIST (GIST) to link the domain knowledge into a bigger context. The main resource of the developed ontology are SCOs from Germany, but the concepts can be transferred and made applicable in a wider area.

Formatting

- Hyperlinks are embedded and clickable in the PDF. They are marked with an arrow and a light blue border: ↪Hyperlink
- Everything related to the ontology implementation, such as references to classes or relations, is written as `typewriter text`.
- Relations are written in camelCase: `subclassOf`
- Classes are bold, capitalized, and use Snake_Case: **`Awesome_Class`**
- Name spaces may be added to a class for clarification; they are separated by a colon: **`namespace:Class`**

Contents

1. Introduction

1.1. Motivation

SCO[1] are student-run consulting businesses, that focus on teaching their members essentials business and life skills exceeding the theoretical knowledge from university. They are very similar to small to medium consulting businesses, but are run and organized—most of the time exclusively—by students. And even though the concept is not universally know, these kind of organizations exists worldwide and have a history dating back to at least 1967[2]. Germany has two different umbrella organizations for SCOs with more than 60 member organizations.[3]

But as far as we know, there hasn't yet been any effort to collect and compose the existing domain knowledge of German SCOs in a publicly available and usable form. We consider this an important task, since it is a contribution to prevent knowledge loss that is inherent in the dynamics of these organizations: the majority of the staff are students and thus their consulting career is inherently linked to their university career:

1. The career is time-bound to the duration of the education. A bachelor's degree in Germany averages 7,5-7,6 semesters and a master's degree 4,2-4,5 semesters, which adds up to a total of 11,7-12,1 semesters or ca. six years. [3] This frames the available time for the transfer of the domain knowledge.[4]
2. The career is in parallel to the curriculum. From our experience, freshmen that decide to join student organizations typically do so at the beginning of their second or third semester, after they got acclimated with the workload of their university classes. Since students usually participate in parallel to their education—and the focus is typically on the education—, they have to manage their time accordingly, which reduce time spent with the SCO. Furthermore, students may have other (e. g. personal) interests that compete with the same time budget.

The reasons above reduce the available time for knowledge transfer and persistence and make these problems harder. Many SCOs have worked on and developed solutions to help with this problem. Some of them are informal, some formal in nature. For example: One particular organization, ↪Hanseatic Consulting (HC), used process methodology to document a lot of their knowledge.

However, the majority of available domain documents are highly individualized and miss the necessary level of abstraction to make them directly applicable to other SCOs. But even though

[1] Also known as ↪Junior Enterprises (JEs) in some parts of the world.

[2] The founding of ↪Junior ESSEC in France.

[3] ↪Bundesverband Deutscher Studentischer Unternehmensberatungen (BDSU), ↪Junior Consultant Network (JCNetwork)

[4] There sometimes are also PhD students, but they can be considered outliers and are atypical.

every SCO is organized slightly differently than the next, uses different vocabulary and each has their individual culture, they all share the idea of teaching consulting and project work to their members. Since they aim for the same goal, they are very similar at their core.

Therefore we try to contribute a more general model in the form of an ontology that tries to combine the domain knowledge, vocabulary, and common concepts.

1.2. Goal and Scope of the Work

The goal of this work is the description of an abstract SCO. It extracts the available implicit expert knowledge, links it with related work, and transforms it into explicit knowledge by using an ontology as its vehicle. It defines common classes and relations required to describe such an organization using domain vocabulary. Additionally it provides terminology explanations, background knowledge, and links into other ontologies where it is sensible.

1.3. Deliverables

The output of this work are two documents:

- This thesis as a documentation and explanation of the ontology development process including but not limited to: methodology, background information, decisions in regards to the ontology, etc.
- The ontology document as a representation of the domain knowledge.

1.4. Out of Scope

1. This work is not a thesaurus and not a documentation about a specific Student Consulting Organization.
2. No diagram for individual orgs
3. The ontology will not include the individual project process, since projects differ vastly between each other and more general ontologies and frameworks for projects already exists.

1.5. Outlook

The main motivation of this work is documenting the domain knowledge and making it available to interested parties, such as the umbrella organizations, SCOs, or students. Furthermore: Creating a computer-readable ontology with this goal in mind can help advance the idea of SCOs, for example by enabling software projects.

One particular use case in the intersection between knowledge management and software projects, is the creation of a tool that helps with founding new SCOs at universities where no SCO currently exists. Creating an organization without guidance is a daunting task; having a repository available, that structures and describes the elemental components of such an organization, can be a great help.

2. Ontologies

> "Knowledge Representation is the field of Artificial Intelligence that focuses on the design of formalisms that are both epistemologically and computationally adequate for expressing knowledge about a particular domain." [2, p. XV, Preface]

This work develops one specific domain ontology.

Ontology development is a complex subject. To avoid ambiguity it is important to define how the terms domain and ontology are used.

Additionally this section touches on ontology and ontology-engineering research to reflect on some general challenges that occur during ontology development.

There are numerous definitions of the term ontology available in literature [8, p. 4, section 1.1.2.1] and there is no perfectly unified understanding of the term [5].

2.1. Definitions

2.1.1. Ontology

Ontologies are a way of organizing knowledge. They make it possible to structure a domain in a way, that it can be used in a technical project

"In computer science, an ontology is a conceptual model specified using some ontology language; this idea was succinctly captured by Gruber in his definition of an ontology as "an explicit specification of a conceptual- isation" [2]

2.1.2. Domain

2.1.3. Classes

2.2. Types of Ontologies

2.2.1. Upper Ontology

(GFO)

2.2.2. Domain Ontology

Ontologies as Domain Models -> Something special?!

2.2.3. Content Ontology

`http://ontologydesignpatterns.org/wiki/Category:ContentOP`

2.3. The Open World Assumption

2.4. The Unique Name Assumption

2.5. The Connection to other Ontologies

2.6. Ontology Representation

Model-theoretical Languages, Graph-Based Systems, Frame-Based Systems, Hybrid Systems (see GFO document)

Format: `http://www.ksl.stanford.edu/knowledge-sharing/kif/`

An ontology allows the domain knowledge to grow and to be flexible. As already stated above, the core concepts of different SCOs are very similar. However, different SCOs may use different vocabulary to describe the same thing, object or process. This creates a requirement for a knowledge system: it has to be extensible and mutable.

-> OWL

3. Student Consulting Organizations: Meta Discussion

3.1. Methodology for the Development of the Ontology

The primary goal of this work is the creation of a particular domain ontology. To achieve this goal, we start with the methodology that is proposed in the documentation [10] of the ontology editor ↪***Protégé*** – built and maintained by ontology researchers of ***Stanford University***. [9]

It involves the following steps:

(1) Determine the domain and scope of the ontology,
(2) consider reusing existing ontologies,
(3) enumerate important terms in the ontology,
(4) define the classes and the class hierarchy,
(5) define the properties of classes-slots,
(6) define the facets of the slots, and
(7) create instances.

It is important to note, that even though these steps look like they should be performed sequentially, this is not the case. Instead, the ontology starts out as a draft and is refined during development [10, Section 3, Introduction], following the iterative approach, that is common for ontology development. [15, p. 158, section 1.5.1] This quickly becomes apparent during the process of answering the suggested ***Competency Questions*** to (1) determine the domain and scope of the ontology [10, Section 3, Step 1] and taking into account (2) existing ontologies. And this also is true for steps (3) to (6). Therefore the steps are grouped together to make the overall structure of this work easier to follow.

The phases of the methodology are discussed in more detail in the following two sections and group the proposed steps as follows:

1. Steps 1 and 2 are performed during the ***Research Phase***.
2. Steps 3 to 6 during the ***Analysis and Synthesis Phase***.

The last step, (7) the creation of instances, is omitted in this work. It is only really relevant if the ontology is used to describe one specific SCO. [**CN**] However, this ontology is operating on a higher level of abstraction, trying to describe a more general case.

3.1.1. Research Phase

To our understanding, the main goal of the first part of the methodology is the creation of a foundation for the ontology. It should have a clearly defined scope and its limits should be set. Additionally the recommended reuse of other ontologies helps creating a web of linked knowledge and reduces the amount of duplicate work.

To find a starting point for data collection and identify existing ontologies, we take an intuitive first look at SCOs and their driving factor:

The Idea of Student Consulting Organizations
Selecting a career is a very difficult and important choice in a young persons life. University education is closely linked to this choice and entering a specific field often requires a specific degree (e. g. to become a lawyer, a student has to pass the bar exam).

Most universities know this and have set up dedicated offices to offer career advice to their students. They not only help picking a fitting course of studies at the beginning of a university career, but also help the students to aim for a fitting job.

Doing an internship with a company working in the field the student is interested in, is a widespread recommendation. [**CN**] It allows for a glimpse into the profession as well as gathering work experience.

SCOs offer an option to investigate a career in business consulting, as well as learning the associated skills and getting paid in the process. They offer the students a way to learn about concept like project based work—the modus operandi of consulting companies—, e. g. project planning and management, as well as structuring and presentation of information.

Consulting is a growing [**CN**] and very diverse [**CN**] field of work. Since consulting can be applied to any field of business, it is often used as a stepping stone into a career.

Observing this intuitive perspective, we can see, that SCOs are connected to other knowledge domains in various ways: They are a type of social organization and thus are driven by people and processes. Organizations and in extension their processes have actors with responsibilities ([**RACI**]). This is a hint that the concept of roles has to be a part of the ontology. SCOs can be generally considered a form of business and therefore business aspects have to be taken into account. The fact that they do consulting work, creates a connection into the domain of (business) consulting and the domain of projects, since consulting work is project based.

This intuitive approach generates a the starting point for the research:

- Previously developed ontologies in related domains, e. g. consulting, project management, educational organizations.
- Available domain knowledge, e. g. process documentation of HC and Campus Inform (CI).[1]
- Personal expert domain knowledge and peer-review by other SCO members.

Furthermore it implies some more general research topics:

- Implications of other general, upper-level-, and top-level- ontologies, e. g. GFO, Basic Formal Ontology (BFO), GIST, .

[1] Two SCOs the author has worked with the longest.

- Theory of description logic and ontologies, e. g. modeling of roles and processes.

The results of the Research Phase influence all parts of this work. However, some links are evident: The identified Related Work is discussed in section 3.2. The implications of higher-level ontologies and the classification can be found in section 3.3. The solutions to challenges of modeling the domain are discussed in sections 3.4 and 3.5.

Defining the scope of the ontology is the formal step that concludes the Research Phase. This work accomplishes this by answering the Competency Questions. Since the questions can be considered a part of the ontology, they and their corresponding answers can be found as part of the ontology in section 4.1.

3.1.2. Analysis and Synthesis Phase

The majority of this work happens during the Analysis and Synthesis Phase. Its goal is the review, interpretation, and structuring of the collected data; ultimately generating an ontology in the target format: OWL.

Based on the Protégé-methodology, the first two steps of this phase are: (3) the creation of an enumeration of terms that are important for the domain. And (4) the translation of the terms into the backbone of every ontology: the class hierarchy. Both are rooted in the results of the Research Phase and further supplemented by expert knowledge.

At the core of this process is the conversion of available implicit knowledge into explicit knowledge. This task is generally not trivial, because the class hierarchy is a construct that already has an important relation built in: `subclassOf`. This means that sub-classing already gives meaning to the terms in the hierarchy. It is therefore important to only introduce a sub-class relationship, if it is correct for the representation of the domain. This makes it mandatory to think about the connection between different terms.

To help with this thought process, we introduce a creative step: We start with a brainstorming to create a domain vocabulary collection in the form of a word cloud. This word cloud can then be represented by a graph, using the terms as vertices and display association between terms (e. g. connected ideas or concepts) with edges. We try to use existing vocabulary as much as possible, to prepare the links into other domains that will be done in the later stages of development. This word cloud helps to create a starting point for the more rigorous class hierarchy.

Starting out with the list of terms creates a first-draft/skeleton class hierarchy containing high-level classes and trivial sub-classes (e. g. high-level class **Process** and all the identified processes as trivial sub-classes). Next is the organization and

3.2. Related Work

Ontologies are knowledge representers SCO have overlap in two directions: project management and consulting PM is a very wide topic that basically has unlimited amount of detail -> needs abstraction Part of PM are in itself complex topics: time, problem analysis, ...

3.3. Classification of the Ontology

vocab vs ontology

dcterms[2]

3.3.1. Relevant Top-Level-Ontologies

- BFO - DOLCE - GFO - GIST - BPMN [11]

GFO: process, roles and time

3.3.2. Relevant Upper-Domain-Ontologies

- OWL-S - SUMO

FIBO FOAF is close to schema, link to dublin core: ”dct:Agent Dublin Core's notion of Agent is much like FOAF's; Dublin Core says ”A resource that acts or has the power to act.”, we say ”things that do stuff”. As nobody has provided a counter-example of something fitting one definition but not the other, we say here that foaf:Agent stands in an 'equivalent class' relationship to dct:Agent (and vice-versa).” [4, External Vocabulary References] Description of a Project (DOAP) https://github.com/ewilderj/doap Schema.org Ontology (Schema): not really ideal, but useful for general concepts like Person or Organization

3.4. General Aspects of Ontology Development

3.4.1. Keeping Things Simple

Polysemy Paper [1] Keep It Stupid Simple (KISS)

keep it as simple as possible (e. g. *contract* and *contract document* can be considered two distinct things, but this distinction is not important for the domain knowledge – maybe add a relation ”has document”?)

Example: A contract is a document that captures a business agreement. The word ”contract” can refer to the immaterial agreement between the parties, but it can also refer to the document itself. Depending on the use case of the ontology it might be useful to separate these two things.

However, in this ontology the goal is to keep it a simple as possible, since the potential users of this ontology are not necessarily experts.

[2] `dcterms` is used in the ***FOAF*** rdf file, `dct` is used in the ***FOAF*** documentation.

3.4.2. Content Completeness Problem

`https://en.wikipedia.org/wiki/Content_completeness_problem` As is true for any domain ontology [**CN**], the content completeness problem exists for this ontology as well.

active/passive content completeness

bewusst weggelassen:

Consulting Topics The main goal of consulting companies is in their name: consulting. They are a source of expertise and knowledge and can be employed as an option to solve a difficult problem at hand. The problem space of consulting companies is vast; examples are: Digitization, Human Resources, Knowledge Management, Market Research, Marketing, Coporate Strategy, etc. These topics are obviously part of the consulting domain. However, they are deliberately omitted, since their exploration would exceed the scope of the work.

IT and Communication Systems IT systems are an essential part of modern business and there are companies where these systems are integral to everything (e. g. AI companies). However, in the context of a consulting company they are mainly used to support, supplement, and optimize the already existing processes. Hence, a model of an IT system would not contribute in a meaningful way to the ontology.

3.4.3. Time

Implement time abstract -> only needed for processes before/after no absolute time

BFO GFO have complex implementations of time that are not needed in this ontology.

3.5. Domain Specific Aspects of Ontology Development

Next to the general aspects of ontology development are domain specific considerations. These can include domain specific ideas and concepts that have to be modeled to represent the domain correctly. This work discusses three of these aspects more extensively, that have a high impact on the whole model: Context switching between the organizational and project context, the model for social constructs, and processes.

3.5.1. Context Switches

We use the term *context switching* in this work to describe the fact that some concepts influence instances of certain classes: They impose their context on the class. Even though individual instances are not in scope of this work, the implications of different contexts has to be taken into consideration while modeling the domain.

Organizational Context The first context is the Organizational Context (OC) and it is straight forward: The SCO itself is the primary and default context of the domain. It applies to all classes until otherwise stated. It is the context for the majority of classes. It is especially relevant for the internal formal structure of the organization. For example: It separates individuals or organizations that are involved with the SCO in any form from those that are not involved with it.

Project Context The second important context is the Project Context (PC). It is a sub-context of the OC, since projects are considered a part of the SCO

Each project on its own has its own context with associated roles

An SCO can be thought of as a central hub for multiple projects. It exists to generate the projects

E. g. a `Person` can be part of a project and part of the organizational structure at the same time.

E. g. a `Person`, can exist in different contexts at the same time.

The first two contexts are rooted in the nature of SCOs. On the one hand, they are organizations and as such have their internal structures, hierarchies, business ranks, etc. On the other hand they exists to provide project opportunities.

Org-wide vs specific This pattern is not limited to a particular branch of the ontology.

3.5.2. Social Constructs

One of the first things to consider in an ontology where social dynamics play a big role, are human beings, their grouping and their roles in social contexts. Since this ontology describes a social construct and the whole domain is driven by processes that involve people, it requires an adequate class representation. Furthermore the context switches described in section 3.5.1 can also apply here: Individuals can act in different capacities, e. g. as a SCO member or -non-member, as part of a project, as a customer, etc. Aggregation of these actors can occur in different degrees of formalization, e. g. informal meeting of SCO members as friends, a project team meeting, an official meeting of the member council, etc. This variability has to be taken into account for an accurate representation.

Since this is not a domain specific phenomenon, it is sensible to use this observation and consider how existing and related ontologies (see section 3.2) represent these cases.

General Implementation

Starting with the general model of human beings, FOAF is a very common choice when thinking about representing social structures. It is a well established ontology and referenced multiple times as backbone for social concepts. Its implementation and description are relatively basic: The anchor is the top-level class **`foaf:Agent`**[3], which is referred to as the class of “things

[3] `foaf:Agent rdfs:comment: "An agent (eg. person, group, software or physical artifact)."`

that do stuff". It is connected to the name space of the Dublin Core Metadata Terms (DCMT) via `equivalentTo` `dcterms:Agent`. It is sub-classed by **foaf:Group**[4], **foaf:Organization**[5], **foaf:Person**[6], **Person**[7], and **schema:Person**[8]. **Person** and **schema:Person** are defined as `equivalentTo` **foaf:Person**.[9] **foaf:Person** and **foaf:Organization** are disjoint. **foaf:Group** aggregates any type of **foaf:Agent**. DOAP reuses exactly the same classes as FOAF. It also has the same links to Schema and DCMT.

Schema implements **schema:Person**[10] and **schema:Organization**[11]. **schema:Person** is considered `equivalentTo` **foaf:Person**. This establishes a two-way link between FOAF and Schema. **schema:Organization** is sub-classed to accommodate for specialized forms of organizations that are relevant for the use cases schema was developed for, e. g. **schema:Airline**, **schema:NGO**. A collection class like **foaf:Group** does not exists explicitly, but a **schema:Person** as well as a **schema:Organization** can be a `memberOf` an **Organization**.

FIBO uses very similarly or identically named classes with a more complex description. The root class is called **fibo:AutonomousAgent**[12], sub-classed by **fibo:Person**[13], representing individual humans. Like in FOAF, this class is disjoint with **fibo:Organization**[14]. **fibo:Group**[15] exists as a sub-class of **fibo:Collection**[16] and is described as collection of **fibo:AutonomousAgent**.

GIST offers the three classes **gist:Person**[17], **gist:Group**[18], and **gist:Organization**[19] as its implementation of the social structure. However, the classes are organized very differently in the hierarchy and use the `subclassOf` relation more extensively compared to e. g. FOAF: To fully extract all information about the class **gist:Person**, its whole class path has to be taken into account. A **gist:Person** is `subclassOf` **gist:LivingThing**[20], which in turn is `subclassOf`

[4] `foaf:Group rdfs:comment: "A class of agents."`

[5] `foaof:Organization rdfs:comment: "An organization."`

[6] `foaof:Person rdfs:comment: "A person."`

[7] Note: The ontology doesn't offer any description.

[8] See footnote 7.

[9] The link to Schema was added in the last update in 2014.

[10] `schema:Person rdfs:comment "A person (alive, dead, undead, or fictional)."`

[11] `schema:Organization rdfs:comment: "An organization such as a school, NGO, corporation, club, etc. "`

[12] `fibo-fnd-aap-agt:AutonomousAgent skos:definition: "An agent is an autonomous individual that can adapt to and interact with its environment."`

[13] `fibo-fnd-aap-ppl:Person skos:definition: "a person; any member of the species homo sapiens"`

[14] `fibo-fnd-org-org:Organization skos:definition: "a unique framework of authority within which a person or persons act, or are designated to act, towards some purpose, such as to meet a need or pursue collective goals on a continuing basis"`

[15] `fibo-fnd-org-fm:Group skos:definition: "a collection of autonomous entities"`

[16] `fibo-fnd-arr-arr:Collection skos:definition: "a grouping of some variable number of things (may be zero) that have some shared significance"`

[17] `gist:Person rdfs:comment: "NEGATIVE EXAMPLE: fictional characters."`

[18] `gist:Group rdfs:comment: "A collection of People. The group may or may not be an Organization. Many organizations consist of groups of people, but that is not a defining characteristic."`

[19] `gist:Organization rdfs:comment: 1. "A generic organization that can be formal or informal, legal or non-legal. It can have members, or not.", 2. "EXAMPLES: Legal entities like companies; non-legal entities like clubs, committees, or departments.", 3. "NOTE: There are a plethora of different kinds of organizations that differ along many facets, including members, structure, purpose, legal vs. non-legal, etc."`

[20] `gist:LivingThing rdfs:comment: 1. "EXAMPLES: A cat, a mushroom, a tree.", 2. "NEGATIVE EXAMPLES: fictional life forms such as Unicorns or Mickey Mouse.", 3. "NOTE: In the open world, you must assume that it might have since died.", 4. "Something that is now, or at some point in time was, alive and growing."`

gist:PhysicalIdentifiableItem[21]; and both parent classes are carrying additional properties. Similarly **gist:Group** is `subclassOf` **gist:Collection**[22] with the limitation of every **gist:Group** `hasMember some` **gist:Person**.

BFO and GFO don't offer any directly usable implementation for this specific problem, since they operate on a different level of abstraction.

When looking at the related work, we make the following observations:

1. The modeling of human beings is concrete and intuitive: It operates on a low level of abstraction. Concepts that are in use by the layperson, e. g. **Person** and **Organization**, are commonly used in reviewed ontologies; except for the top-level ontologies that operate on a much higher level of abstraction and are therefore not concerned with the concreteness of modeling human beings.

2. The class **Agent** represents actors of an action. Sub-classing it gives the model freedom to express what exactly acts: It can be an **Agent**, a **Person**, a **Group**, or an **Organization**. This flexibility makes the models powerful. They can describe general (e. g. somebody) or other agents (e. g. robots) directly via **Agent**, but can also be used more specifically via a sub-class (e. g. a **Person**). Additionally, a group is a collection class that is also `subclassOf` **Agent** and hence can become an **Agent**—an actor—itself.

As shown above, the classes **Agent**, **Person**, **Organization**, and **Group** are common in the class hierarchies of the related ontologies. Therefore this ontology will use these classes. However, the different ontologies also use different ways of defining classes. Ranging from the very direct and simple way of FOAF, to the very intricate way of GIST. Since this ontology is trying to be as intuitive to use as possible, the more simple approach from FOAF is adapted.[23]

After deciding on these basic building blocks, they can be extended according to the additional domain specifications. For example, a **Person** might need further differentiation based on SCO membership status, on business rank for internal organization and career progress, or on organizational roles. But there are other concepts that can involve a **Person**—e. g. "*being a customer* of the SCO"—that could just as easily involve an **Organization**. This observation points to the requirement of a more general approach for modeling these cases: Roles.

[21] `gist:PhysicalIdentifiableItem rdfs:comment: 1. "EXAMPLES: a computer, a book.", 2. "NEGATIVE EXAMPLE: A discontinuous thing like a manufacturing line cannot reasonably have an RFID attached to it, even though its parts are not the same kind of thing as the whole.", 3. "NOTE: You could, at least in principle, put an RFID tag on members of this class. Physical things are made of something. E.g., statues are made of bronze.", 4. "NOTE: In practice, this always means that the parts are not the same kind of thing as the whole."`

[22] `gist:Group rdfs:comment: 1. "Any identifiable grouping of instances. For instance, a jury is a collection of people.", 2. "EXAMPLES: A jury is a group of people, a financial ledger is a collection of transaction entries; a route is an (ordered) collection of segments."`

[23] This decision is a direct application of the KISS principle (see section 3.4.1). Having more information in an ontology can obviously be useful for a very detailed model of a domain. However, its size can be kept smaller and the complexity lower by omitting information (e. g. certain relations or attributes) that can be inferred from linked ontologies when necessary. For example: FIBO and GIST offer attributes for a **Person**; e. g. in FIBO a **Person** `hasDateOfBirth exactly 1` **Date**, in GIST a **Person** is `offspringOf` another **Person** and need to have a **name** `xsd:string`. These attributes can be extracted on demand, by following the `equivalentTo` relation.

Roles

As already shown by *Loebe*, the concepts and ideas about roles have been heavily discussed in the ontology community and literature. The concept is not trivial and very fundamental. Using roles as part of an ontology allows for a flexible and powerful model. The concept can be applied in various ways: It can be used in the intuitive way regarding social roles, e. g. thinking about a human being playing the role of a patient and another the role of a doctor. But it is also possible to think about numbers and relationship between them in the form of abstract roles. [6, p. 130–131]

Since SCOs are a social construct and are defined by the people of the organization, the modeled roles are from the *social* role type. Furthermore, the contexts describes in section 3.5.1 also have implications for the roles that exists in the domain.

Membership To identify the roles that are necessary in the domain model, we first look at the OC. Within it, the most basic property is membership: Either being part of the organization and thus being a `Member` or not participating and a `Non-Member`. `Members` of the SCO can play different roles in the OC. `Non-Members` don't play roles within the organization, but can play external roles. For example: The role of a `Customer`. The distinction is important for this ontology, since the status is typically used in the internal organizational procedures. For example: Someone might be required to be a proper member to be allowed to vote in the Member Assembly, to be part of a project, or to become part of the Executive Board.

It is important to note, that within SCOs the `Member` role can only be played by human beings. For this model this means a restriction to `Person`. The members are the defining group that fills all the organizational functions, works on the projects, and participates in the majority of processes. Even though membership does not have to be limited like this in general—there are many examples where organizations are members of other organizations—it is limited in this domain. Since `Member` are necessarily **always** human beings, we introduce the role as `subclassOf Person`.

The `Non-Member` role, however, is not limited to only `Person`. In fact, everything and everyone that is not a `Member` is by definition a `Non-Member`. Therefore it is sufficient to model the class `Member` and omit `Non-Member`.

In this duality exists one domain edge case, where the membership is not clear cut when projecting on the real world: Alumni. In common usage, almuni are a group of people that have been affiliated with an organization in the past, but are typically not members of that organization anymore. A good example are university alumni: the group of people that graduated from a specific university. Depending on the internal structure of the particular SCO alumni can be either `Members` or `Non-Members`. To account for this, we introduce `Alumnus` as an external role that has the restriction to be only played by `Person`.

Business Ranks The second property a `Person` can have in the OC is the business rank. Examples from the business world are: Associate, Senior Associate, Consultant, Partner, etc. Similar to regular businesses, SCOs also organize these ranks around their career process: A person receives the lowest available rank at the begin of their career. During the time with the organization a person is awarded higher ranks based on some organizational system (e. g. a merit-based system), until the highest rank is reached or the person leaves the organization.

The exact terms for the ranks, their meaning, gradation, and organizational implications are highly specific to the SCO instance. We therefore introduce a rough and extensible skeleton representation:

1. **`Trainee`**: The entry level rank, without a formal membership.
2. **`Junior Consultant`**: The first rank after someone acquires an official, formal membership.
3. **`Consultant`**: The rank that implies someone has reached the required amount of knowledge and experience to fulfill all organizational functions.
4. **`Senior Consultant`**: The ultimate rank of the organization that can be reached after gathering a substantial amount of knowledge, experience, and organizational social status.

`Ranks` can only be attained by **`Members`** and therefore both are directly related. Since a **`Member`** can only hold exactly one **`Rank`** and the **`Rank`** further specifies the **`Member`** in the OC, we introduce the ranks as `subclassOf` **`Member`**. This is similar to the Schema approach that sub-classes **`Organization`** to be more specific about the type of organization: We are sub-classing **`Member`** to be more specific about it.

Multiple Contexts The strength of the role concept is its flexibility. A player can play multiple roles at the same time.

An example for this is the CEO of the organization. A CEO has defined responsibilities and has to fulfill certain tasks. With SCOs typically any member can become CEO by being elected. The elected member still retains the aforementioned rank. On the other hand there are roles people take on within the organization. For SCOs these roles can be divided into two distinct groups based on their context: project roles and organizational roles.

For example, a **Consultant** (rank within the organization) can work as a **Project Member** (role in the project), while being the CEO (role in the organization) of the organization.

> **Note:** The model says nothing about social status and political power that typically come with ranks and roles within an organization (e. g. a person that holds a rank or role for a long time may still have organizational power after stepping down: ↪Éminence grise).

Degree of Formalization

A more abstract concept that has to be considered is contract eligibility. In this work, this term represent the concept of being able to enter into a contract. The underlying discussion is fueled by the question

This is relevant for this ontology, because the **`Contract`**—and especially its sub-class **`Project_Contract`**—is a central document for every **`Project`**.

In the German jurisdiction for example, any natural person at or above the age of 18 has contractual freedom. Additionally there exists the concept of the legal person that is created to allow organizations to enter contracts in a similar way as a natural person.

SCOs themselves are an aggregation of individuals working towards a common goal. Their clients can be individuals or businesses, which, by German law, are represented by individuals ("Geschäftsführung"[24]).

If the SCO is a registered association, a *formal* membership to participate might even be required by law under certain circumstances.

Conclusion: Human Beings in this Work

3.5.3. Processes

Processes are a helpful concept when describing organizations: They are created to achieve a goal and its processes are the steps needed to reach that goal. [16, p. 5, Definition 1.1] In theory, every organization can be decomposed to a sequence of single activities, which, when executed correctly and in the correct order, terminate in reaching the goal of the organization.

Since processes are a commonly used concept in the business world, it is not surprising, that many different methods and frameworks for modeling them have been developed. Their output often are visual representations of all workflows that make up an organization. Combining process models with goals and measurements makes them a powerful tool for optimization and quality control. For example, ISO 9001 is an industry standard that uses a process approach as the foundation of measuring quality. [14] Because process documentation contains a lot of data about organizations, it is a valuable source for ontology development.

Widely known representations and methods include: Flowcharts, Business Process Modeling and Notation (BPMN), Event-Driven Process Chain (EPC), Unified Modeling Language (UML) Activity Diagrams, and Object Process Methodology (OPM)[25]. There are also contributions rooted in ontology research, such as the BPMN ontology (an OWL ontology for the BPMN notation) [11], the Process Specification Language (PSL)[26], and processes concepts as part of GFO or BFO.

Implementation in Related Ontologies

When compared to the rather practical and direct implementation of social structures discussed in section 3.5.2, processes are a more abstract concept. The impact of abstraction levels clearly shows when analyzing related ontologies. For example: While FOAF is a good source when discussing its niche—the modeling of connection between human beings—it does not require an implementation of a process concept. The closest possible link between these two knowledge domains is the class `foaf:Project`[27], which can be viewed as a procedural concept. However, it doesn't offer any additional reusable detail.

A similar observation can be made for Schema. Its primary purpose is adding semantic meaning to the internet: "Schema.org is a collaborative, community activity with a mission to create,

[24] § 6 Nr. 2 S. 1 Gesetz betreffend die Gesellschaften mit beschränkter Haftung (GmbHG), §§ 76, 77 Aktiengesetz (AktG)

[25] Standardized as ISO 19450.

[26] Developed by the National Institute of Standards and Technology (NIST) and standardized as ISO 18629.

[27] `foaf:Project rdfs:comment: "A project (a collective endeavour of some kind)."`

maintain, and promote schemas for structured data on the Internet, on web pages, in email messages, and beyond.” [13] Hence, it is not surprising, that it doesn’t implement a detailed process representation.

On the other hand, the two related top-level ontologies, BFO and GFO, deal with time (see section 3.4.3) on a very high level and also implement process concepts: BFO uses the class **`bfo:Occurent`**[28] as entry point for its process concepts. It is sub-classed by **`bfo:Process`**[29], **`bfo:Process_Boundary`**[30], **`bfo:Spatiotemporal_Region`**[31], and **`bfo:Temporal_Region`**[32]. Looking closer at the given examples[33] for **`bfo:Process`** emphasizes the classes high-level nature.

GFO uses the class **`gfo:Processual_Structure`**[34] as its entry point. It is sub-classed by **`gfo:Occurent`**[35] and **`gfo:Processes`**[36].

DOAP

FIBO

GIST uses the root class **`Event`**[37] to deal with time-related constructs such as processes.

bfp gfo contain valuable information, that can be modified and adapted for the process implementation of the SCO domain.

Structure of the Class Hierarchy

Processes need special attention when implementing them in a domain ontology, since their nature is quite different from other classes that represent physical, e. g. **`Document`**, or intuitive concepts, e. g. **`Person`**. As mentioned in section 3.1.2, the built in `subclassOf` relation of the class

[28] `bfo:Occurent elucidation: "An occurrent is an entity that unfolds itself in time or it is the instantaneous boundary of such an entity (for example a beginning or an ending) or it is a temporal or spatiotemporal region which such an entity occupies_temporal_region or occupies_spatiotemporal_region. (axiom label in BFO2 Reference: [077-002])"`

[29] `bfo:Process definition: "p is a process = Def. p is an occurrent that has temporal proper parts and for some time t, p s-depends_on some material entity at t. (axiom label in BFO2 Reference: [083-003])"`

[30] `bfo:Process_Boundary definition: "p is a process boundary =Def. p is a temporal part of a process and p has no proper temporal parts. (axiom label in BFO2 Reference: [084-001])"`

[31] `bfo:Spatiotemporal_Region elucidation: "A spatiotemporal region is an occurrent entity that is part of spacetime. (axiom label in BFO2 Reference: [095-001])"`

[32] `bfo:Temporal_Region elucidation: "A temporal region is an occurrent entity that is part of time as defined relative to some reference frame. (axiom label in BFO2 Reference: [100-001])"`

[33] `bfo:Process example of usage: 1. a process of cell-division, 2. a beating of the heart, 3. a process of meiosis, 4. a process of sleeping, 5. the course of a disease, 6. the flight of a bird, 7. the life of an organism, 8. your process of aging.`

[34] `gfo:Processual_Structure dc:description: "The category of processual structures centers around the more intuitive notion of processes. It captures processes themselves and occurrents, i.e., primarily structures of several other kinds that can be derived from processes."`

[35] `gfo:Occurent dc:description: "The category of occurrents comprises several categories that can be derived from processes."`

[36] `gfo:Processes dc:description: "Processes are directly in time, they develop over and unfold in time. Processes have characteristics which cannot be captured by a collection of time boundaries. In particular, processes exhibit internal coherence."`

[37] `gist:Event rdfs:comment: "Something happening over some period of time, often characterized as some kind of activity being carried out by some person, organization, or software application."`

hierarchy already carries semantic meaning, that is generally not applicable to processes. For example: a `Delivery_Process` may involve a `Food_Preperation_Process` as a procedural step. However, it is easy to see and understand that a `Food_Preperation_Process` is *not* `subclassOf` a `Delivery_Process` and therefore should not inherit its properties.

To model processes correctly, one could consider introducing a class like `*_Process_Part` (in the given example: `Delivery_Process_Part`) and use it to collect and connect sub-processes to their parent process. However, this results in many additional *helper* classes in the class hierarchy, since every level of sub-processes requires another `*_Process_Part` class. This makes the class hierarchy harder to read and understand, since the process structure is encoded in these helper classes.

Another solution is the use of a root `Process` class to collect all processes and the relation `isProcessPartOf`[38] to connect a sub-process to its parent process. This results in a completely flat structure of the class hierarchy: every process is directly `subclassOf Process`, independent from the level of abstraction.

Ordering of Processes

Another aspect that has to be discussed is the ordering of processes.

Processes are a concept that heavily relies on abstraction. The right level of abstraction depends on the use case.

1. It is hard to create a complete process diagram/describe a complete process
2. A successful process model relies on the correct
3. Besides from the problem of completeness, if you go on the lowest level of abstraction ordering the steps become easier (true?!)
4. Trying to create a generally applicable domain ontology brings up an interesting questions in regards to process concepts: what is the correct level of abstraction and is it possible to bring certain processes in the correct order
5. It is therefore necessary to look at every process and its parts – and discuss if 1) correct level of abstraction 2) can it be ordered
6. A parent process aggregates child process
7. When discussing the lowest level of a process Depending on the level of abstraction,
8. Indepentendt from the class hierarchy problem, another topic: The procedural information is encoded in the relation.
9. Processes are a sequence of action -> but not always
10. there should be a next_process relation, but it's not straight forward, since there is not necessarily a strict ordering.
11. introduce a non strict ordering?

Discreet Events and Liquid Processes

1. In addition to that a distinction has to be made between *discreet* events and *liquid* processes. [12, p. 447]

[38] Inverse: `hasProcessPart`.

2. Processes are an immaterial concept that is strongly connected to relations of relative time, such as `before` and `after`.

Legal Requirements in the Processes

SCOs are organizations in the social context; German law applies to them, like it applies to every other German organization. It influences their structure and processes. However, discussing the impact of the law onto internal workings of an organizations would go far out of scope of this work. Therefore the ontology omits a detailed description of the legal obligations, but references them abstractly where it is necessary.

For example: German law requires every company to pay taxes on their earnings. Depending on the SCO and the way projects are handled, this influences the process that is concerned with taxation. To develop a perfectly correct model, a very detailed discussion of specific processes would be required; this is out of scope. However, interacting with the tax authorities and learning about and filing the correct paper work is an important part of the learning experience for student consultants. It is therefore important for the ontology. To address this fact, but keep the ontology focused, it is condensed into the class *Project Taxation Process* as part of the *Project Process*.

Implementation of Processes in this Work

They intuitively have a start, duration, and end.

Since the focus of the ontology is on simplicity, we decide to use a single class **`Process`** as root for all processes in conjunction with the `isProcessPartOf` relation. This method utilizes the core concepts of ontologies, classes and relations, and avoids encoding extra information in unconventional ways. To compensate for the resulting un-intuitive flat class hierarchy, we add diagrams to describe the processes and their relationships graphically (see section A.2.2).

The primary goal of an SCO is teaching students project work. They reach this goal by training their members and offering them opportunities to work on real-world projects. Looking at this from a high-level process perspective, this can be boiled down to distinct steps that have to be performed by the organization:

- Members have to be recruited.
- Members have to be taught the necessary skills, to be able to work on projects.
- Projects have to be acquired.
- Projects have to be worked on by members.

Additionally these steps have various amount of support processes that help facilitate them, e. g. technical and legal support.

Conflating this intuitive view results in two complex processes that are commonly know in the business world [**CN**] and are also present in the available process documentation [**HCPD**]:

1. A Human Resource Process (HRP), that focuses on the recruitment, training, and generally enabling of human beings (or in the case of SCOs: Members).
2. A Project Process (PP), that documents the way an organization handles projects from start to finish.

Again, both processes are influenced by the context switches discussed in section 3.5.1. On the one hand, they can be viewed as an individual instance for one of the main protagonists of the process. On the other, they can viewed as the process of the organization. Example:

- The main protagonist of the HRP is one individual student. This individual student is following one instance of the HRP; this instance does not have to be identical with the instance of a second individual student, nor with the planned process of the organization. Both individuals might do different educational courses, hold different business ranks within the organization, or might be at different points in time of their career.
- The SCO itself has a HRP. It structures important aspects of the organization such as the career path. It describes the complete path from recruitment of a new member to offboarding at the end of the membership. Most importantly this process describes the plan on an abstract level and knowingly omits parts of the real world process that are not important to the organization.

4. Student Consulting Organizations: The Ontology

4.1. Scope of the Domain

4.1.1. What is the domain that the ontology will cover?

SCOs are a form of consulting firms. They can be compared to small consulting businesses, but are staffed – most of the time exclusively – by students. In other countries, e. g. in France and Brazil, they are also referred to as *Junior Enterprises* (JE). In Germany they are usually a registered association (German: *Verein*) and/or a group associated with a specific university (German: *Hochschulgruppe*). They aim to teach students about consulting as a profession by providing a platform that educates and trains students in the craft and provides them with the organizational means to work on consulting projects.

The domain is a specialization of the a classical consulting firm. It differs especially in terms of professionalization, since companies are focused on profit using education as the means, whereas SCOs focus on the educational aspect and on providing experience, while having profit as secondary goal.

4.1.2. For what we are going to use the ontology?

This ontology is a contribution to the knowledge management of SCOs. It can be used to learn or teach about the domain. It can also be used as a starting point for projects that require a model of the domain.

4.1.3. For what types of questions the information in the ontology should provide answers?

The ontology serves as an abstract description of the SCO domain. It defines all classes and relations that are typically present in this type of organization. Therefore it can answer questions like:

- What processes exist and are required in an SCO?
- What roles exist and have to be filled in an SCO?

4.1.4. Who will use and maintain the ontology?

The users of this ontology are the leadership of SCOs in Germany as well as the leadership of the SCO umbrella organizations. The release version coincides with the finalization and grading of this work. If the ontology sees use by the target group, it will be maintained by the author. Access will be publicly provided on a GitHub repository. It is considered a living document, hence not necessarily complete until otherwise stated. Contributions and forks will be possible via the GitHub interface.

4.2. Classes

4.2.1. Agent

Group

Organization

Person

Trainee

Junior Consultant

Consultant

Senior Consultant

4.2.2. Document

4.2.3. Processes

Human Resource Process

Project Process

Support Processes

4.2.4. Projects

4.3. Relations

Syntactic decision: is/has relations

Agent

- All members except trainees and almunus can be corporate officers ◇
- Non-members can't become corporate officers ◇
- Members can play project team roles ◇
- Every agent can play the customer role in a project ◇
- Every member goes through his individual HRP
- An organizational rank has tbd requirements
- Customer, Team, Contract, etc are **part of** a project
- Organizations can only **play** the customer **role** in a project ◇
- Organization can only play external roles ◇

Processes

- All processes have a **next_process** ◇
- **previous_process** should be inferred?!
-

before/after:

- FIBO: relates to -> precedes/succeeds?
- plays role

5. Conclusion

6. Further Research

A. Appendix

A.1. Term Enumeration

A.2. Diagrams

A.2.1. General Diagrams

Figure A.1.: Ranks

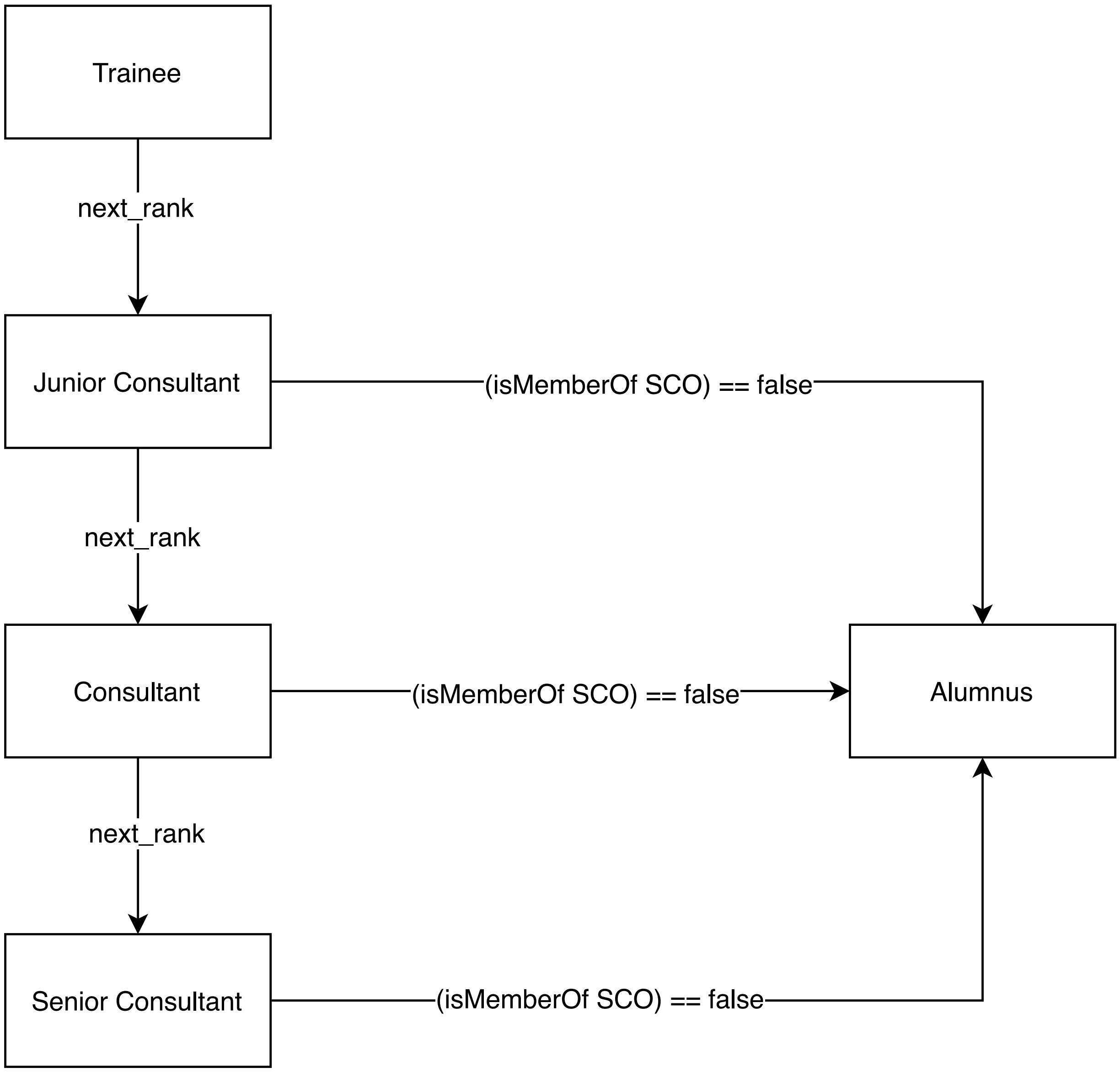
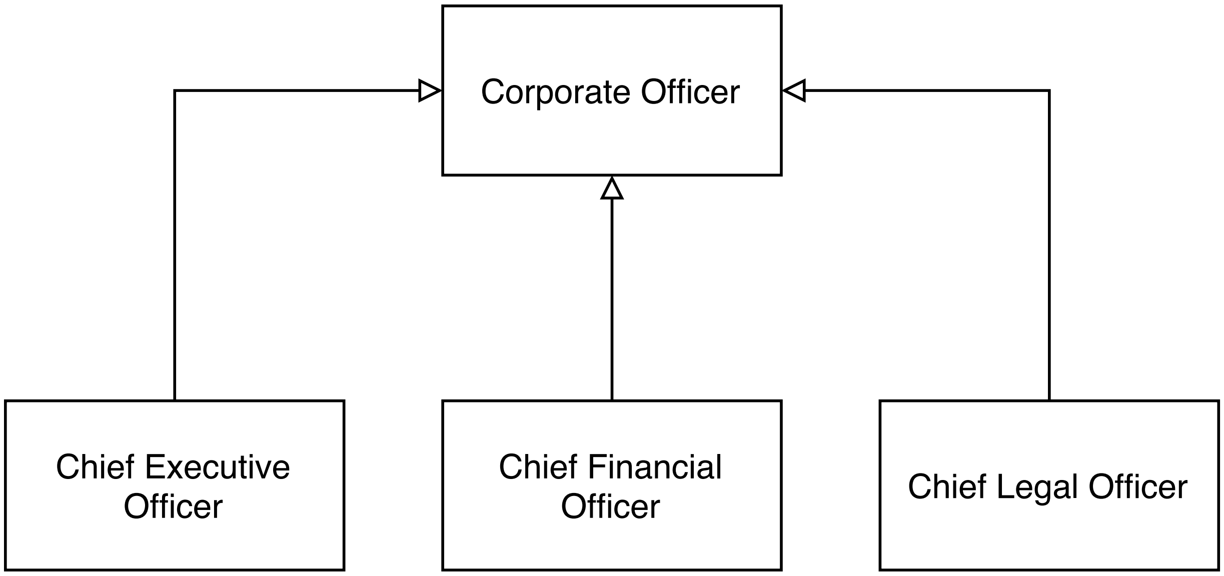


Figure A.2.: Corporate Officers



A.2.2. Process Diagrams

Figure A.3.: Project Process

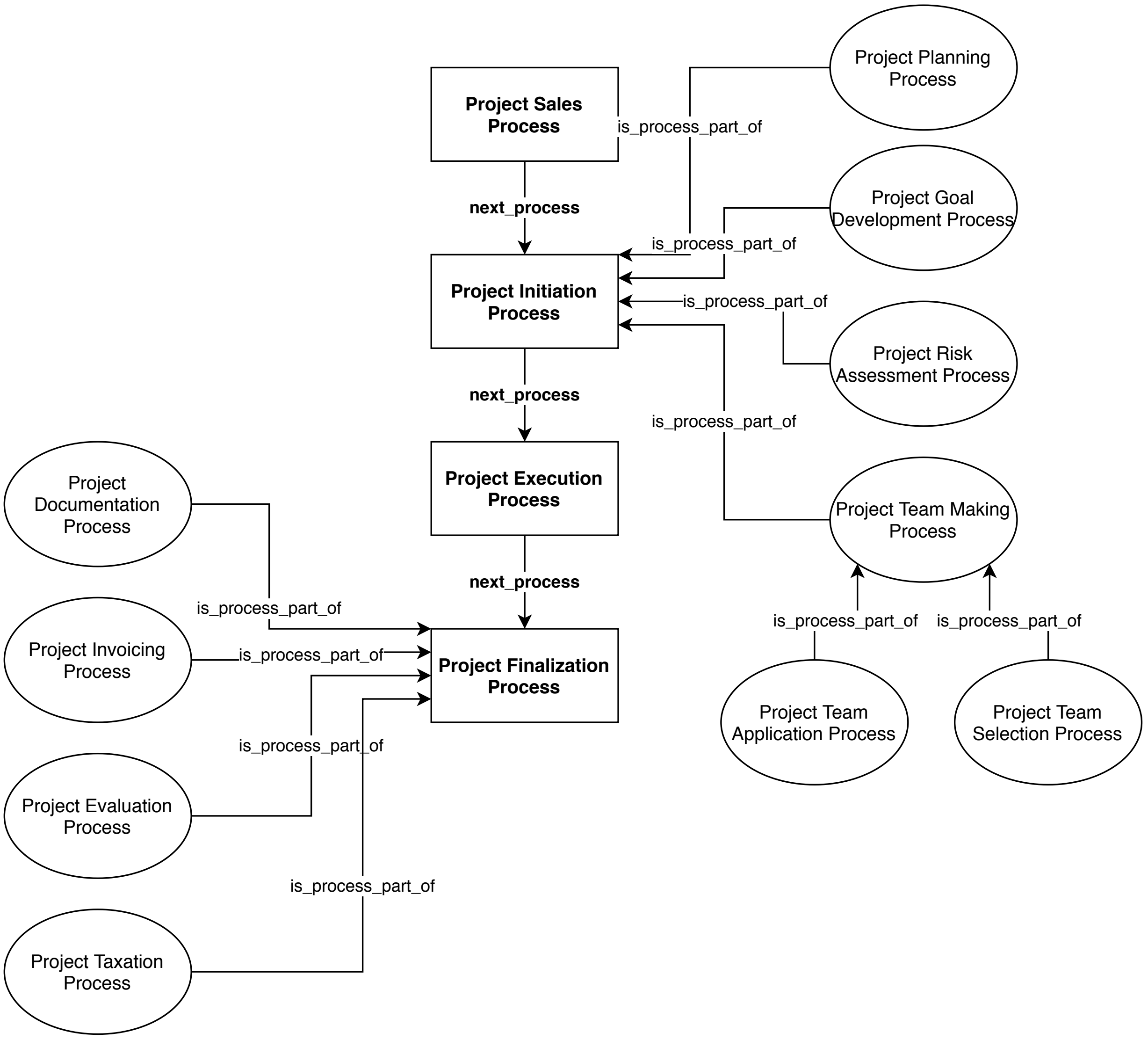
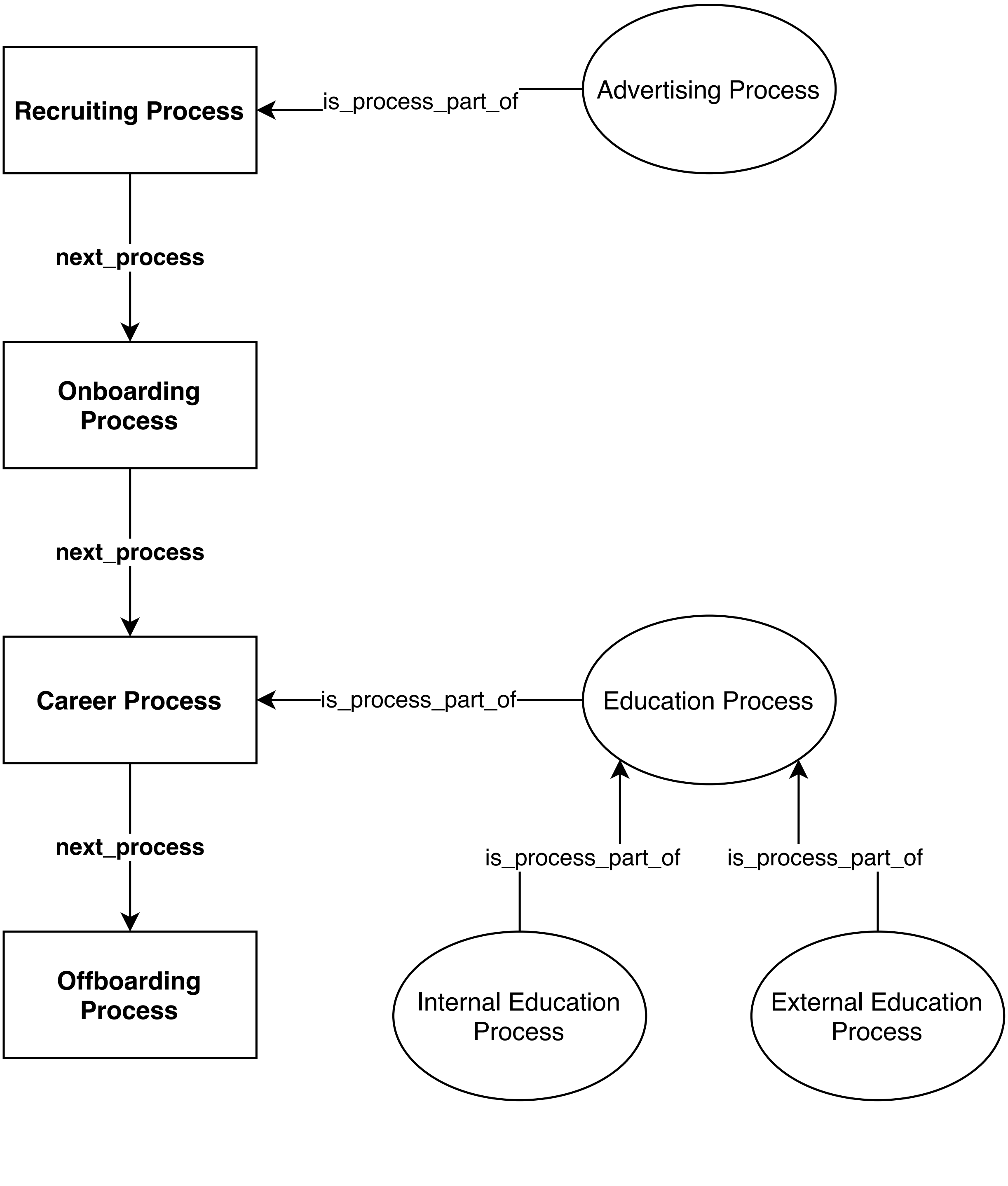

Project Planning Process
Project Sales Process
is_process_part_of
next_process
Project Goal Development Process
is_process_part_of
Project Initiation Process
is_process_part_of
Project Risk Assessment Process
next_process
is_process_part_of
Project Documentation Process
Project Execution Process
Project Team Making Process
next_process
is_process_part_of
is_process_part_of
is_process_part_of
Project Invoicing Process
is_process_part_of
Project Finalization Process
Project Team Application Process
Project Team Selection Process
is_process_part_of
Project Evaluation Process
is_process_part_of
Project Taxation Process

Figure A.4.: Human Resources Process



A.3. Glossary

AktG Aktiengesetz. 15

BDSU Bundesverband Deutscher Studentischer Unternehmensberatungen. 1

BFO Basic Formal Ontology. 6, 9, 12, 15, 16, 33

BPMN Business Process Modeling and Notation. 15, 33

CI Campus Inform. 6

DCMT Dublin Core Metadata Terms. 11

DOAP Description of a Project. 8, 11, 16, 33

EPC Event-Driven Process Chain. 15

FIBO Financial Industry Business Ontology. i, 8, 11, 12, 16, 33

FOAF Friend of a Friend. i, 8, 10, 11, 12, 15, 33

GFO General Formal Ontology. i, 6, 9, 12, 15, 16, 33

GIST GIST. i, 6, 11, 12, 16, 33

GmbHG Gesetz betreffend die Gesellschaften mit beschränkter Haftung. 15

HC Hanseatic Consulting. 1, 6

HRP Human Resource Process. 18

JCNetwork Junior Consultant Network. 1

JE Junior Enterprise. 1

KISS Keep It Stupid Simple. 8, 12

NIST National Institute of Standards and Technology. 15

OC Organizational Context. 10, 13, 14

OPM Object Process Methodology. 15

PC Project Context. 10

PP Project Process. 18

PSL Process Specification Language. 15

Schema Schema.org Ontology. 8, 11, 14, 15, 33

SCO Student Consulting Organization. i, 1, 2, 4, 5, 6, 7, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21

UML Unified Modeling Language. 15

A.4. Bibliography

[1] Alexandra Arapinis and Laure Vieu. “A plea for complex categories in ontologies”. In: *Applied Ontology* 10.3-4 (2015), pp. 285–296.

[2] Franz Baader et al. *Introduction to Description Logic.* Cambridge University Press, 2017. ISBN: 978-0-521-87625-4.

[3] Statistisches Bundesamt. *Zusammengefasste Abschlussprüfungen mit erstem und weiteren Abschluss sowie Gesamtstudienzeit (2016-2018).* Oct. 2019. URL: `https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Bildung-Forschung-Kultur/Hochschulen/Tabellen/bestandenepruefungen-studiendauer.html` (visited on 2020-03-01).

[4] Libby Miller Dan Brickley. *FOAF Vocabulary Specification 0.99.* English. Jan. 2014. URL: `http://xmlns.com/foaf/spec/` (visited on 2020-02-27).

[5] Wolfgang Hesse and Hermann Engesser. “Ontologie”. In: *Informatik-Spektrum* 37.4 (May 2014), pp. 281–282. ISSN: 1432-122X. DOI: `10.1007/s00287-014-0808-2`. URL: `http://dx.doi.org/10.1007/s00287-014-0808-2`.

[6] Frank Loebe. “Abstract vs. social roles–Towards a general theoretical account of roles”. In: *Applied Ontology* 2.2 (2007), pp. 127–158.

[7] Frank Loebe. “An analysis of roles: Towards ontology-based modelling”. In: *Onto-Med Report* 6 (2003).

[8] Frank Loebe. “Ontological Semantics: An Attempt at Foundations of Ontology Representation”. PhD thesis. Universität Leipzig, Mar. 2015.

[9] Mark A Musen. “The protégé project: a look back and a look forward”. In: *AI matters* 1.4 (2015), pp. 4–12.

[10] Natalya F. Noy and Deborah L. McGuinness. *Ontology Development 101: A Guide to Creating Your First Ontology.* URL: `https://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html` (visited on 2020-02-20).

[11] Marco Rospocher, Chiara Ghidini, and Luciano Serafini. “An ontology for the Business Process Modelling Notation”. In: *Formal Ontology in Information Systems - Proceedings of the Eighth International Conference, FOIS2014, September, 22-25, 2014, Rio de Janeiro, Brazil.* Ed. by Pawel Garbacz and Oliver Kutz. Vol. 267. IOS Press, 2014, pp. 133–146. DOI: `10.3233/978-1-61499-438-1-133`. URL: `http://dx.doi.org/10.3233/978-1-61499-438-1-133`.

[12] Stuart Russell. *Artificial intelligence : a modern approach.* 3th ed. New Jersey: Pearson, 2010. ISBN: 978-0-136-04259-4.

[13] Schema.org. *About Schema.org.* URL: `https://schema.org/docs/about.html` (visited on 2020-03-16).

[14] International Organization for Standardization. *The Process Approach in ISO 9001:2015 (ISO/TC 176/SC 2/N1289)*. (Visited on 2020-03-10).

[15] Heiner Stuckenschmidt. *Ontologien: Konzepte, Technologien und Anwendungen*. Springer, 2010. ISBN: 978-3-642-05403-7.

[16] Mathias Weske. *Business Process Management : Concepts, Languages, Architectures*. Springer, 2019. ISBN: 978-3-662-59432-2.

A.5. Ontology Import Links

This work lists different ontologies in the related work section. To import them into the Protégé editor, the following links can be used:

BFO: `http://purl.obolibrary.org/obo/bfo/2.0/bfo.owl`

BPMN: `https://dkm-static.fbk.eu/resources/ontologies/BPMN/BPMN_2.0_ontology.owl`

DOAP: `http://usefulinc.com/ns/doap`

FIBO: `https://spec.edmcouncil.org/fibo/ontology/master/2019Q4.1/LoadFIBOProd.rdf`

FOAF: `http://xmlns.com/foaf/spec/index.rdf`

GFO: `http://www.onto-med.de/ontologies/gfo-basic.owl`

GIST: `https://ontologies.semanticarts.com/o/gistCore9.0.0.owl`

Schema: `http://schema.org/version/latest/schema.rdf`

A.6. Acknowledgments

This work was conducted using the Protégé resource, which is supported by grant GM10331601 from the National Institute of General Medical Sciences of the United States National Institutes of Health.