An Ontology of Instruction 1.0

Contributors:
COGAN SHIMIZU — Kansas State University
PASCAL HITZLER — Kansas State University
AARON EBERHART — Kansas State University
QUINN HIRT — Wright State University
CHRISTOPHER STEVENS — Air Force Research Laboratory
CHRISTOPHER W. MYERS — Air Force Research Laboratory
BENJI MARUYAMA — Air Force Research Laboratory
COLIN KUPITZ — Oak Ridge Institute for Science and Education
DARIO SALVUCCI — Drexel University

Document Date: December 11, 2020

This material is based upon work supported by the Air Force Office of Scientific Research under award numbers FA9550-18-1-0386 and FA9550-18-1-0371.

Contents

List of Figures

1 Overview

We are presenting the ontology which drives the data gathering and integration done as part of the project, *Towards Undifferentiated Cognitive Agents: Determining Gaps in Comprehension*,[1] funded by the Air Force Office of Scientific Research under award number FA9550-18-1-0386. It is a collaborative effort with the projects *Toward Undifferentiated Cognitive Agents for Diverse Specializations* (Air Force Research Laboratory, PIs Chris Myers (RH) and Benji Maruyama (RX)) and *Toward Undifferentiated Cognitive Agents: Translating Instructions to Knowledge* (Drexel University, PI: Dario Salvucci).

Development of the ontology was a collaborative effort and was carried out using the principles laid out in, e.g., [Shimizu et al., 2020]. The modeling team included domain experts, data experts, software developers, and ontology engineers.

The ontology has, in particular, be developed as a *modular* ontology [Shimizu et al., 2020, Hitzler and Shimizu, 2018] based on ontology design patterns [Hitzler et al., 2016].[2] This means, in a nutshell, that we first identified key terms relating to the data content and expert perspectives on the domain to be modeled, and then developed ontology modules for these terms. The resulting modules, which were informed by corresponding ontology design patterns, are listed and discussed in Chapter 2. The Uagent Ontology, assembled from these modules, is then persented in Chapter 3.

For background regarding Semantic Web standards, in particular the Web Ontology Language OWL, including its relation to description logics, we refer the reader to [Hitzler et al., 2012, Hitzler et al., 2010].

[1] See `https://daselab.cs.ksu.edu/projects/afosr-cogagents`.

[2] See `https://daselab.cs.ksu.edu/content/modular-ontology-engineering-portal` for pointers to further resources on the approach.

2 Modules

We list the individual modules of the ontology, together with their axioms and explanations thereof. Each axiom is listed only once (for now), i.e. some axioms pertaining to a module may be found in the axiom set listed for an earlier listed module. Schema diagrams are provided throughout, but the reader should keep in mind that while schema diagrams are very useful for understanding an ontology [Karima et al., 2017], they are also inherently ambiguous.

Primer on Ontology Axioms

Logical axioms are presented (mostly) in description logic notation, which can be directly translated into the Web Ontology Language OWL [Hitzler et al., 2010]. We use description logic notation because it is, in the end, easier for humans to read than any of the other serializations.[1]

Logical axioms serve many purposes in ontology modeling and engineering [Hitzler and Krisnadhi, 2016]; in our context, the primary reason why we choose a strong axiomatization is to disambiguate the ontology.

Almost all axioms which are part of the Enslaved Ontology are of the straightforward and local types. We will now describe these types in more detail, as it will make it much easier to understand the axiomatization of the Enslaved Ontology.

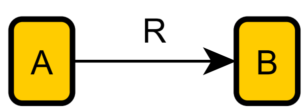

Figure 2.1: Generic node-edge-node schema diagram for explaining systematic axiomatization.

There is a systematic way to look at each node-edge-node triple in a schema diagram in order to decide on some of the axioms which should be added: Given a node-edge-node triple with nodes A and B and edge R from A to B, as depicted in Figure 2.1, we check all of the following axioms whether they should be included.[2] We list them in natural language, see Figure 2.2 for the formal versions in description logic notation, and Figure 2.3 for the same in Manchester syntax, where we also list our names for these axioms.

1. A is a subClass of B.
2. A and B are disjoint.
3. The domain of R is A.
4. For every B which has an inverse R-filler, this inverse R-filler is in A. In other words, the domain of R, scoped by B, is A.

[1]Preliminary results supporting this claim can be found in [Shimizu, 2017].

[2]The OWLAx Protégé plug-in [Sarker et al., 2016] provides a convenient interface for adding these axioms.

1. $A \sqsubseteq B$
2. $A \sqcap B \sqsubseteq \bot$
3. $\exists R.\top \sqsubseteq A$
4. $\exists R.B \sqsubseteq A$
5. $\top \sqsubseteq \forall R.B$
6. $A \sqsubseteq \forall R.B$
7. $A \sqsubseteq R.B$
8. $B \sqsubseteq R^{-}.A$
9. $\top \sqsubseteq \leq 1R.\top$
10. $\top \sqsubseteq \leq 1R.B$
11. $A \sqsubseteq \leq 1R.\top$
12. $A \sqsubseteq \leq 1R.B$
13. $\top \sqsubseteq \leq 1R^{-}.\top$
14. $\top \sqsubseteq \leq 1R^{-}.A$
15. $B \sqsubseteq \leq 1R^{-}.\top$
16. $B \sqsubseteq \leq 1R^{-}.A$
17. $A \sqsubseteq \geq 0R.B$

Figure 2.2: Most common axioms which could be produced from a single edge R between nodes A and B in a schema diagram: description logic notation.

5. The range of R is B.
6. For every A which has an R-filler, this R-filler is in B. In other words, the range of R, scoped by A, is B.
7. For every A there has to be an R-filler in B.
8. For every B there has to be an inverse R-filler in A.
9. R is functional.
10. R has at most one filler in B.
11. For every A there is at most one R-filler.
12. For every A there is at most one R-filler in B.
13. R is inverse functional.
14. R has at most one inverse filler in A.
15. For every B there is at most one inverse R-filler.
16. For every B there is at most one inverse R-filler in A.
17. An A may have an R-filler in B.

Domain and range axoims are items 2–5 in this list. Items 6 and 7 are extistential axioms. Items 8–15 are about variants of functionality and inverse functionality. All axiom types except disjointness and those utilizing inverses also apply to datatype properties.

Structural tautologies are, indeed, tautologies, i.e., they do not carry any formal logical content. However as argued in [Hitzler and Krisnadhi, 2016] they can help humans to understand the ontology, by indicating *possible* relationships, i.e., relationships intended by the modeler which, however, cannot be cast into non-tautological axioms.

Explanations Regarding Schema Diagrams

We utilize schema diagrams to visualize the ontology. In our experience, simple diagrams work best for this purpose. The reader needs to bear in mind, though, that these diagrams are ambiguous and incomplete visualizations of the ontology (or module), as the actual ontology (or module) is constituted by the set of axioms provided.

We use the following visuals in our diagrams:

rectangular box with solid frame and orange fill: a class

rectangual box with dashed frame and blue fill: a module, which is described in more detail elsewhere in the document

rectangular box with dashed frame and purple fill: a set of URIs constituting a controlled vocabulary

oval with solid frame and yellow fill: a data type

1. A SubClassOf B (subClass)
2. A DisjointWith B (disjointness)
3. R some `owl:Thing` SubClassOf A (domain)
4. R some B SubClassOf A (scoped domain)
5. `owl:Thing` SubClassOf R only B (range)
6. A SubClassOf R only B (scoped range)
7. A SubClassOf R some B (existential)
8. B SubClassOf inverse R some A (inverse existential)
9. `owl:Thing` SubClassOf R max 1 `owl:Thing` (functionality)
10. `owl:Thing` SubClassOf R max 1 B (qualified functionality)
11. A SubClassOf R max 1 `owl:Thing` (scoped functionality)
12. A SubClassOf R max 1 B (qualified scoped functionality)
13. `owl:Thing` SubClassOf inverse R max 1 `owl:Thing` (inverse functionality)
14. `owl:Thing` SubClassOf inverse R max 1 A (inverse qualified functionality)
15. B SubClassOf inverse R max 1 `owl:Thing` (inverse scoped functionality)
16. B SubClassOf inverse R max 1 A (inverse qualified scoped functionality)
17. A SubClassOf R min 0 B (structural tautology)

Figure 2.3: Most common axioms which could be produced from a single edge R between nodes A and B in a schema diagram: Manchester syntax.

arrow with white head and no label: a subClass relationship
arrow with solid tip and label: a relationship (or property) other than a subClass relationship

2.1 Module Overview

The following are the modules which together constitute the Uagent Ontology. Each of them will be presented in detail further below, though in different sequence. The Domain Ontology for Instruction captures instructions for a specific cognitive agent task called ISR-MATB. Currently, the ontology supports the memory of a cognitive agent by adding structure to its knowledge and providing new varieties of query-like recall.

Action
Affordance
Instruction
ISR-MATBExperiment
Item
SituationDescription

2.2 Action

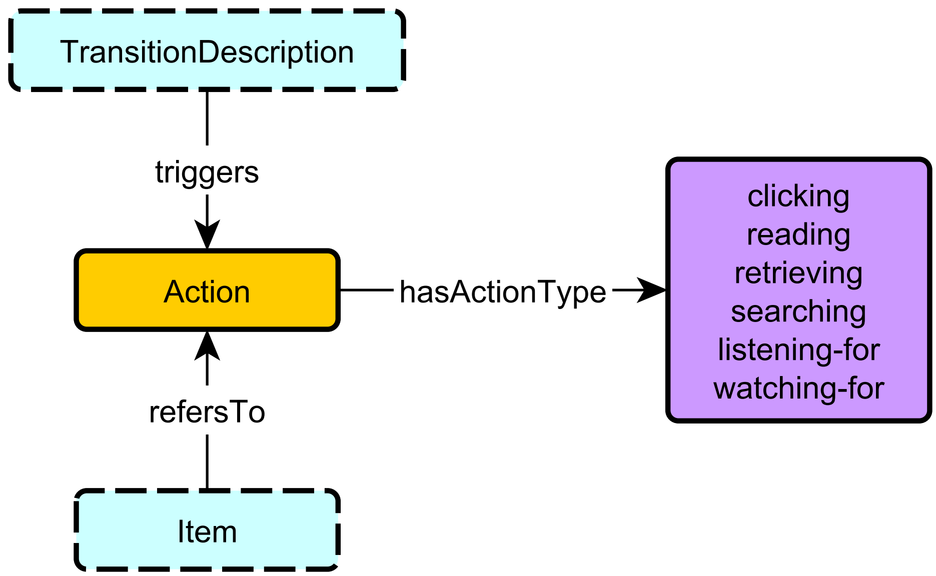

Figure 2.4: Schema Diagram for the Action module.

Axioms:

$$\text{Action} \sqsubseteq {=}1\text{ofType}.\text{ActionType} \tag{1}$$

Explanation of axioms above:

1. Exact cardinality. An Action has exactly one ActionType.

2.3 Affordance

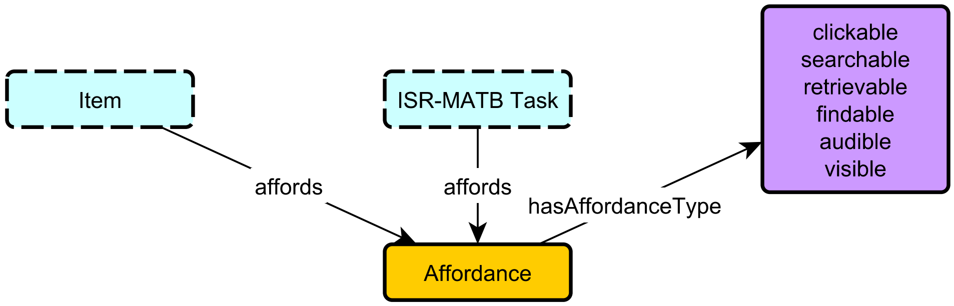

Figure 2.5: Schema Diagram for the Affordance module.

Axioms:

$$\mathsf{Affordance} \sqsubseteq = 1\ \mathsf{ofType}.\mathsf{AffordanceType} \tag{1}$$

Explanation of axioms above:

1. Exact cardinality. An Affordance has exactly one AffordanceType.

2.4 Instruction

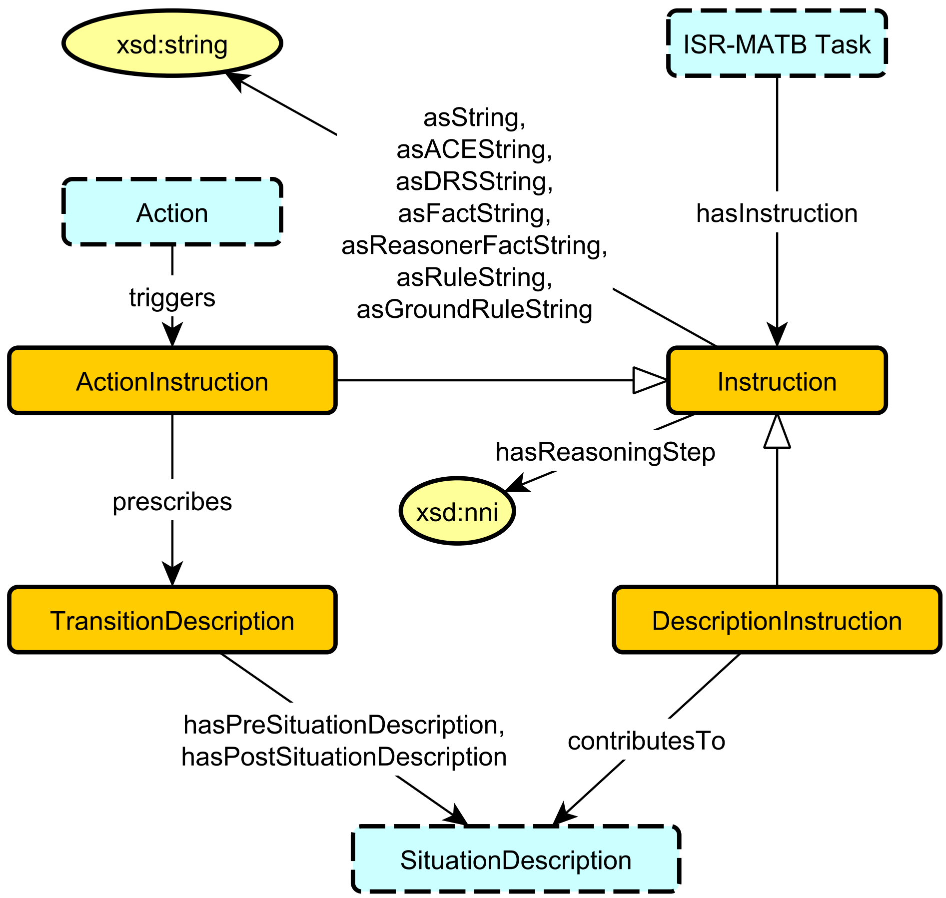

Figure 2.6: Schema Diagram for the Instruction module.

Axioms:

$$\mathsf{ActionInstruction} \sqsubseteq \mathsf{Instruction} \tag{1}$$

$$\mathsf{ActionInstruction} \sqsubseteq \forall \mathsf{prescribes}.\mathsf{TransitionDescription} \tag{2}$$

$$\top \sqsubseteq \forall \mathsf{asString}.\mathsf{xsd{:}string} \tag{3}$$

$$\mathsf{Instruction} \sqsubseteq \geq 0\ \mathsf{asString}.\mathsf{xsd{:}string} \tag{4}$$

$$\mathsf{DescriptionInstruction} \sqsubseteq \mathsf{Instruction} \tag{5}$$

$$\mathsf{DescriptionInstruction} \sqsubseteq \forall \mathsf{contributesTo}.\mathsf{SituationDescription} \tag{6}$$

$$\mathsf{TransitionDescription} \sqsubseteq \forall \mathsf{hasPreSituationDescription}.\mathsf{SituationDescription} \tag{7}$$

$$\mathsf{TransitionDescription} \sqsubseteq \forall \mathsf{hasPostSituationDescription}.\mathsf{SituationDescription} \tag{8}$$

Explanation of axioms above:

1. Subclass. Every ActionInstruction is an Instruction.
2. Scoped Range. The range of prescribes is TransitionDescription, scoped by ActionInstruction.
3. Range. The range of asString is xsd:string.
4. Structural Tautology. An Instruction may have a string representation.
5. Subclass. Every DescriptionInstruction is an Instruction.
6. Scoped Range. The range of contributesTo is SituationDescription, scoped by DescriptionInstruction.
7. Scoped Range. The range of hasPreSituationDescription is SituationDescription, scoped by TransitionInstruction.
8. Scoped Range. The range of hasPostSituationDescription is SituationDescription, scoped by TransitionInstruction.

Remarks

1. In the interest of space, we note that for each of the properties with label as*String, as well as hasReasoningStep are all functional and existential, meaning that they have exactly one value.
2. Reasoning steps non-negative integers. There must be one that starts with 0, the starting Instructions.

2.5 ISR-MATBExperiment

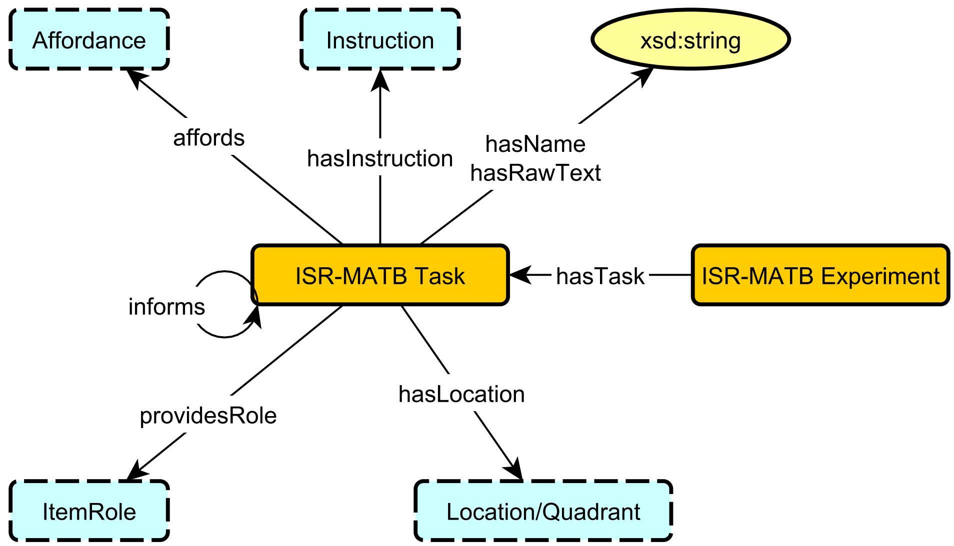

Figure 2.7: Schema Diagram for the ISR-MATBExperiment module.

Axioms:

$$\top \sqsubseteq \forall \text{affords.Affordance} \quad (1)$$

$$\text{ISR-MATBTask} \sqsubseteq \geq 1 \text{ hasInstruction.Instruction} \quad (2)$$

$$\text{ISR-MATBExperiment} \sqsubseteq \leq 4 \text{ hasTask.ISR-MATBTask} \quad (3)$$

$$\top \sqsubseteq \forall \text{hasLocation.Location} \quad (4)$$

$$\top \sqsubseteq \forall \text{hasName.xsd:string} \quad (5)$$

$$\text{ISR-MATBTask} \sqsubseteq =1 \text{ hasName.xsd:string} \quad (6)$$

$$\text{ISR-MATBTask} \sqsubseteq \forall \text{providesRole.ItemRole} \quad (7)$$

$$\text{ISR-MATBTask} \sqsubseteq \forall \text{informs.ISR-MATBTask} \quad (8)$$

Explanation of axioms above:

1. Range. The range of affords is Affordance.
2. Minimum Cardinality. An ISR-MATBTask has at least one Instruction.
3. Maximum Cardinality. An ISR-MATBExperiment consists of at most four ISR-MATBTasks.
4. Range. The range of hasLocation is Location.
5. Range. The range of hasName is xsd:string.
6. Scoped Range. The range of providesRole is ItemRole when the domain is ISR-MATBTask.
7. Scoped Range. The range of informs is ISR-MATBTask when the domain is ISR-MATBTask.

Notes

1. Should there be an existential for affords?
2. What is the difference between Location and Quadrant?
3. hasName should probably not point to a string.
4. The providesRole axiomatization is, at best, incomplete.

2.6 ItemRole

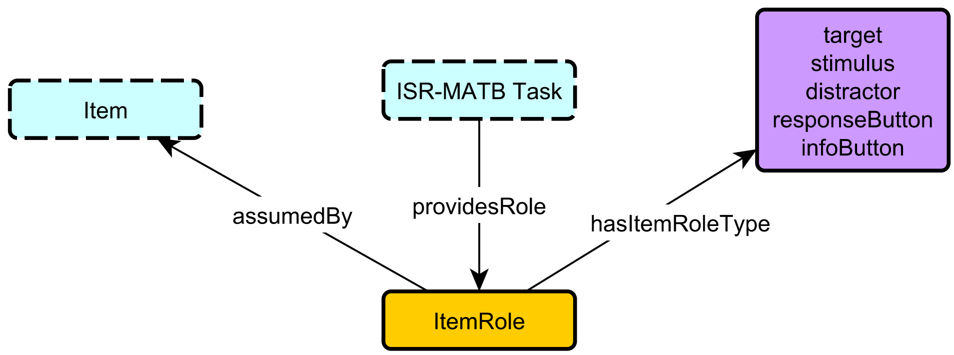

Figure 2.8: Schema Diagram for the ItemRole module.

Axioms:

$$\text{ISR-MATBTask} \sqsubseteq \forall \text{providesRole.ItemRole} \quad (1)$$
$$\top \sqsubseteq \forall \text{hasItemRoleType.ItemRoleType} \quad (2)$$
$$\text{ItemRole} \sqsubseteq \forall \text{assumedBy.Item} \quad (3)$$
$$\text{ItemRole} \sqsubseteq \exists \text{assumedBy.Item} \quad (4)$$

Explanation of axioms above:

1. Scoped Range. The range of providesRole is ItemRole when the domain is ISR-MATBTask.
2. Range. The range of hasItemRoleType is ItemRoleType.
3. Scoped Range. ItemRoles are assumedBy Items.
4. Existential. Every ItemRole is assumedBy an Item.

Notes

1. The providesRole axiomatization is, at best, incomplete.
2. Is an ItemRole always assumed by exactly one Item? I assume so.

2.7 SituationDescription

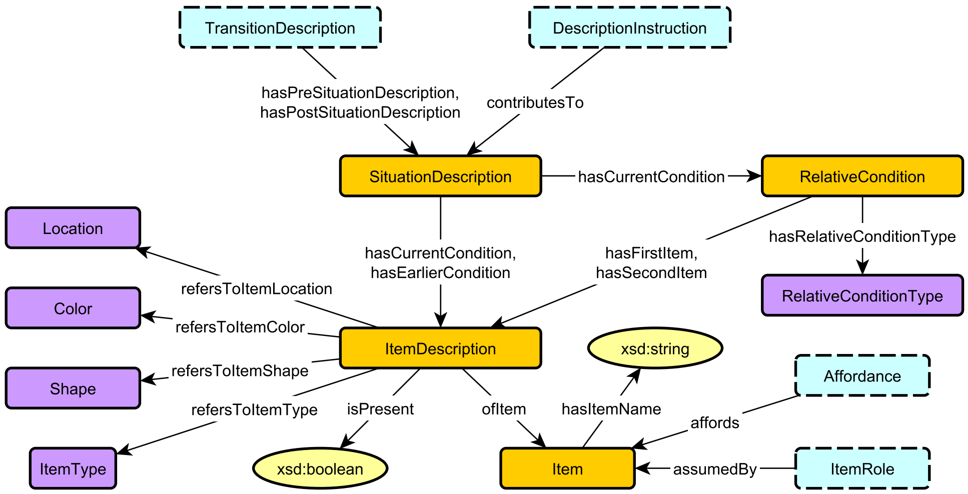

Figure 2.9: Schema Diagram for the SituationDescription module.

Axioms:

$$\mathsf{SituationDescription} \sqsubseteq \forall \mathsf{hasCurrentCondition}.(\mathsf{RelativeCondition} \sqcup \mathsf{ItemDescription}) \tag{1}$$

$$\mathsf{SituationDescription} \sqsubseteq \forall \mathsf{hasEarlierCondition}.\mathsf{ItemDescription} \tag{2}$$

$$\top \sqsubseteq \forall \mathsf{hasRelativeConditionType}.\mathsf{RelativeConditionType} \tag{3}$$

$$\mathsf{RelativeCondition} \sqsubseteq \forall \mathsf{hasFirstItem}.\mathsf{ItemDescription} \tag{4}$$

$$\mathsf{RelativeCondition} \sqsubseteq \forall \mathsf{hasSecondItem}.\mathsf{ItemDescription} \tag{5}$$

$$\mathsf{ItemDescription} \sqsubseteq \forall \mathsf{ofItem}.\mathsf{Item} \tag{6}$$

$$\mathsf{ItemDescription} \sqsubseteq {=}1\ \mathsf{isPresent}.\mathsf{xsd{:}boolean} \tag{7}$$

$$\top \sqsubseteq \forall \mathsf{refersToItemLocation}.\mathsf{LocationType} \tag{8}$$

$$\top \sqsubseteq \forall \mathsf{refersToItemColor}.\mathsf{ColorType} \tag{9}$$

$$\top \sqsubseteq \forall \mathsf{refersToShapeType}.\mathsf{ShapeType} \tag{10}$$

$$\top \sqsubseteq \forall \mathsf{refersToItemType}.\mathsf{ItemType} \tag{11}$$

$$\mathsf{ItemDescription} \sqsubseteq {\geq}0\ \mathsf{refersToItemLocation}.\mathsf{LocationType} \tag{12}$$

$$\mathsf{ItemDescription} \sqsubseteq {\geq}0\ \mathsf{refersToItemColor}.\mathsf{ColorType} \tag{13}$$

$$\mathsf{ItemDescription} \sqsubseteq {\geq}0\ \mathsf{refersToItemShape}.\mathsf{ShapeType} \tag{14}$$

$$\mathsf{ItemDescription} \sqsubseteq {\geq}0\ \mathsf{refersToItemType}.\mathsf{ItemType} \tag{15}$$

$$\top \sqsubseteq \forall \mathsf{hasItemName}.\mathsf{xsd{:}string} \tag{16}$$

$$\exists \mathsf{hasItemName}.\top \sqsubseteq \mathsf{Item} \tag{17}$$

Explanation of axioms above:

1. Scoped Range. The range of hasCurrentCondition is a RelativeCondition or ItemDescription when the domain is SituationDescription.
2. Scoped Range. The range of hasEarlierCondition is ItemDescription when the domain is SituationDescription.
3. Range. The range of hasRelativeConditionType is RelativeConditionType.
4. Scoped Range. The range of hasFirstItem is ItemDescription when the domain is RelativeCondition.
5. Scoped Range. The range of hasSecondItem is ItemDescription when the domain is RelativeCondition.
6. Scoped Range. The range of ofItem is Item when the domain is ItemDescription.
7. Scoped Range. An ItemDescription has exactly one boolean flag indicating whether or not it is present.
8. Range. The range of refersToItemLocation is LocationType.
9. Range. The range of refersToItemColor is ColorType.
10. Range. The range of refersToItemShape is ShapeType.
11. Range. The range of refersToItemType is ItemType.
12. Structural Tautology. An ItemDescription may refer to a LocationType.
13. Structural Tautology. An ItemDescription may refer to a ColorType.
14. Structural Tautology. An ItemDescription may refer to a ShapeType.
15. Structural Tautology. An ItemDescription may refer to an ItemType.
16. Range. The range of hasItemName is xsd:string.
17. Domain Restriction. The domain of hasItemName is restricted to Items.

3 Putting Things Together

This ontology, the Domain Ontology for Instruction, is constituted by the union of the modules described previously, plus meta-level annotations using the Ontology Design Pattern Representation Language (OPLa) [Hitzler et al., 2017, Shimizu et al., 2018].

We consider the controlled vocabularies to be separate from the actual ontology. One advantage of using controlled vocabularies as indicated in this document is, that they provide a seamless capability for expansion of the ontology, by adding further vocabulary items. Sometimes, however, it is the case that there are specific interactions between items in the controlled vocabulary and axioms.

Figure 3.1 shows a schema diagram for the combined ontology. Please recall that all our schema diagrams are necessarily ambiguous and incomplete – while they help to understand and use the ontology, it is the set of logical axioms which actually constitutes the ontology.

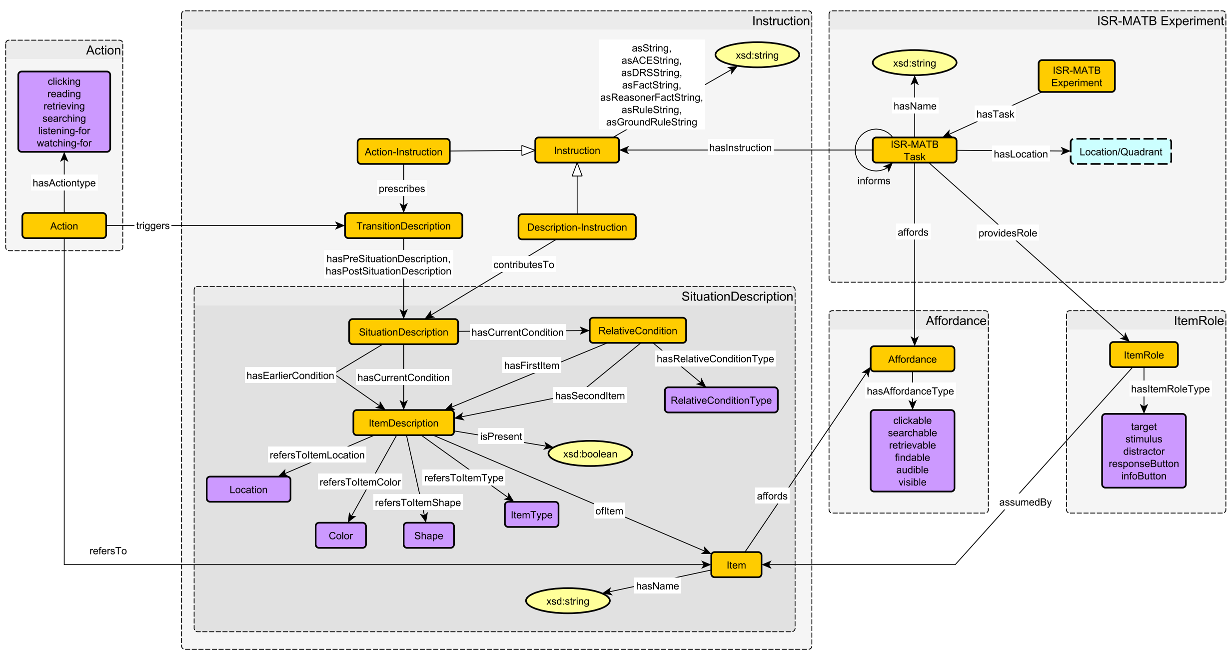

Figure 3.1: Schema Diagram for the combined ontology.

3.1 Axioms

All axioms belonging to the separate modules are part of the overall ontology. In addition, all pairs of classes which are not declared or inferred to be in a subclass relationship, are declared to be disjoint.

Bibliography

[Hitzler et al., 2016] Hitzler, P., Gangemi, A., Janowicz, K., Krisnadhi, A., and Presutti, V., editors (2016). *Ontology Engineering with Ontology Design Patterns - Foundations and Applications*, volume 25 of *Studies on the Semantic Web*. IOS Press.

[Hitzler et al., 2017] Hitzler, P., Gangemi, A., Janowicz, K., Krisnadhi, A. A., and Presutti, V. (2017). Towards a simple but useful ontology design pattern representation language. In Blomqvist, E., Corcho, Ó., Horridge, M., Carral, D., and Hoekstra, R., editors, *Proceedings of the 8th Workshop on Ontology Design and Patterns (WOP 2017) co-located with the 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 21, 2017.*, volume 2043 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Hitzler and Krisnadhi, 2016] Hitzler, P. and Krisnadhi, A. (2016). On the roles of logical axiomatizations for ontologies. In Hitzler, P., Gangemi, A., Janowicz, K., Krisnadhi, A., and Presutti, V., editors, *Ontology Engineering with Ontology Design Patterns - Foundations and Applications*, volume 25 of *Studies on the Semantic Web*, pages 73–80. IOS Press.

[Hitzler et al., 2012] Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P., and Rudolph, S., editors (11 December 2012). *OWL 2 Web Ontology Language: Primer (Second Edition)*. W3C Recommendation. Available at http://www.w3.org/TR/owl2-primer/.

[Hitzler et al., 2010] Hitzler, P., Krötzsch, M., and Rudolph, S. (2010). *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC.

[Hitzler and Shimizu, 2018] Hitzler, P. and Shimizu, C. (2018). Modular ontologies as a bridge between human conceptualization and data. In Chapman, P., Endres, D., and Pernelle, N., editors, *Graph-Based Representation and Reasoning – 23rd International Conference on Conceptual Structures, ICCS 2018, Edinburgh, UK, June 20-22, 2018, Proceedings*, volume 10872 of *Lecture Notes in Computer Science*, pages 3–6. Springer.

[Karima et al., 2017] Karima, N., Hammar, K., and Hitzler, P. (2017). How to document ontology design patterns. In Hammar, K., Hitlzer, P., Lawrynowicz, A., Krisnadhi, A., Nuzzolese, A., and Solanki, M., editors, *Advances in Ontology Design and Patterns*, volume 32 of *Studies on the Semantic Web*, pages 15–28. IOS Press / AKA Verlag.

[Sarker et al., 2016] Sarker, M. K., Krisnadhi, A. A., and Hitzler, P. (2016). OWLAx: A Protégé plugin to support ontology axiomatization through diagramming. In Kawamura, T. and Paulheim, H., editors, *Proceedings of the ISWC 2016 Posters & Demonstrations Track co-located with 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 19, 2016.*, volume 1690 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Shimizu, 2017] Shimizu, C. (2017). Rendering OWL in LaTeX for improved readability: Extensions to the OWLAPI. Master's thesis, Department of Computer Science and Engineering, Wright State University, Dayton, Ohio.

[Shimizu et al., 2018] Shimizu, C., Hirt, Q., and Hitzler, P. (2018). A Protégé plug-in for annotating OWL ontologies with OPLa. In Gangemi, A., Gentile, A. L., Nuzzolese, A. G., Rudolph, S., Maleshkova, M., Paulheim, H., Pan, J. Z., and Alam, M., editors, *The Semantic Web: ESWC*

2018 Satellite Events – ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers, volume 11155 of *Lecture Notes in Computer Science*, pages 23–27. Springer.

[Shimizu et al., 2020] Shimizu, C., Krisnadhi, A., and Hitzler, P. (2020). Modular Ontology Modeling: a tutorial. In Cota, G., Daquino, M., and Pozzato, G. L., editors, *Applications and Practices in Ontology Design, Extraction, and Reasoning*, Studies on the Semantic Web. IOS Press. To appear.