HHAI 2024: Hybrid Human AI Systems for the Social Good
F. Lorig et al. (Eds.)

doi:10.3233/FAIA240187

EASY-AI: sEmantic And compoSable glYphs for representing AI systems

Alexis ELLIS [a] Brandon DAVE [a] , Hugh SALEHI [a] , Subhashini GANAPATHY [a] and Cogan SHIMIZU [a]
[a] *Wright State University*

Abstract. Despite the rapid integration of artificial intelligence (AI) into various research domains and the lives of everyday people, challenges with communicating and understanding these AI systems arise. The lack of a consistent method of communication highlights the need for a transdisciplinary approach to explain the inner workings of AI systems in a cohesive and accessible manner. We thus propose an ontological visual framework using semantically-enhanced, symbols, providing a symbolic language for conveying the structure, purpose, and characteristics of AI systems. The framework encompasses a generalizable glyph set of various AI system components, ensuring both common and obscure architectures can be represented. In this paper, we present the underlying logical formalisms that dictate the behavior of this visual framework as a means to significantly enhance the comprehensibility and understandability of AI system behaviors.

Keywords. Human-Computer Interaction, Semantics, Ontology, Symbology

1. Introduction

As artificial intelligence (AI) systems surge into not only the spotlight of major research interest in different domains but also its rapid integration into our daily lives, so too does the need to understand, explain, and interpret exactly what an AI system is doing. As the complexities of these systems increase with the state of the art[14], the finer details of their functionality may become locked behind opaque systems. It would behoove this area of research to create a standardized means of communicating the structure of AI systems comprehensively to users of various backgrounds. One approach is to develop a visual framework that utilizes symbols to convey the complexities of AI systems. Providing a visual framework of AI systems using symbols has many benefits since symbols have been used by humans since our earliest recorded history [6]. Presently, the use of symbols, more notably known as emojis, are used in ways that convey entire sentences, feelings, or thoughts [8]. Moreover, symbols have the potential to convey information cross-culturally [5]. Therefore, we have built upon the state of the art and created a visual framework that aims to communicate and represent AI systems, this framework is comprised of *sEmantic And compoSable glYphs* (EASY-AI). EASY-AI conveys a visual of the various constraints that govern components of an AI system in an intuitive and easily comprehensible way for the user. EASY-AI will provide a means for more effective communication and better comprehension of AI systems with varying complexities. EASY-AI aims to *ease* not only the tensions between AI and its integration into society but also the lack of understanding that surrounds it by providing a clear and concise way of communication thus increasing human-computer interaction for the future.

2. The EASY-AI framework

This section introduces the breakdown of our tool EASY-AI from transformational ontology design pattern(s) taken from the Boxology format introduced by [15] to an underlying formalized ontology. We demonstrate EASY-AI through 3 distinct layers: **1** The visual framework layer corresponds directly to the structures indicated in Boxology. **2** The classification layer explains the 1-to-1 transformation from the top visual layer into concepts for ontological manipulation. **3** EASY-AI's framework will be presented in the final layer. We represent this 3-layer perspective to demonstrate how EASY-AI was constructed and how this tool works at a fundamental level. We end our description with a use case to demonstrate EASY-AI in a real-world application.

2.1. The Boxology

The Boxology patterns from [15] provide the visual foundation to simplistically break down the inner workings of an AI system, but lack any formalization to dictate how the Boxology patterns interact with one another. These patterns are considered our first layer, providing a user-friendly, surface-level glimpse of what an algorithm might be doing internally. However, the idea is to provide a visual guide to AI Systems with an in-depth look at the processes that take place within these patterns. The core takeaway from Boxology is that each symbol represents a component of an AI system. The application of EASY-AI will take a specific interest in the core building patterns, also referred to as the *elementary patterns*. Since EASY-AI is an extension of Boxology we will be adopting the taxonomy to keep the information consistent. A brief description of the fundamental taxonomy is as follows: **Instances:** are represented by rectangles. **Processes:** are represented by ovals. **Models:** are represented by hexagons. **Actors:** are represented by triangles. **Solid arrows:** indicating the sequential order of the flow of the input. The taxonomy outlined above represents the extent of our discussion on Boxology research. However, should you be interested, the paper is accessible in our reference section. EASY-AI takes another more in-depth look at the intricacies that lie under these patterns without losing the intuition the Boxology visually provides to the user.

2.1.1. Ontological Transformation

The EASY-AI framework implements an ontology which is the representation of concepts and interlinked relationships with specifications defining how these connections are allowed to functionally operate together. This second layer is included as an auxiliary layer for demonstrating how the transition from Boxology to ontology is being conducted to eliminate the potential of another "black box phenomenon." The ontological transformation is a 1-to-1 translation of the patterns into conceptualized classes, where the dashed red lines visually represent this 1-to-1 translation. In our example, using patterns 1a and 1b depicted in Figure 1, though both have different defined inputs, at the EASY-AI layer these features are generally the same, reiterating the importance of keeping the Boxology patterns as a visual component for a quick and comprehensible first assessment of an AI system before diving deeper into its intricacies.

2.2. EASY-AI

The final layer from Figure 1, represented in the previous section, is where we reveal our in-depth observations of the underlying components that the Boxology visual patterns are conveying. This section will delve into the specific concepts and roles that make up the rules that govern EASY-AI.

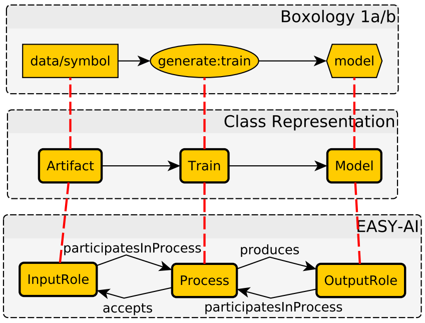


Figure 1.: Graphical representation of the transformation process from Boxology to ontology.

2.2.1. Formalization

The EASY-AI framework is seeded with a core pattern from modular ontology [10], specifically the DataTransformation pattern [13]. The full formalization of the EASY-AI framework is graphically depicted in Figure 3a. In this schema diagram, we represent conceptual classes as gold rectangles, while blue dashed rectangles represent a class described externally from EASY-AI [13]. Filled, directed arrows represent binary relations between classes and open arrows represent subsumption. For visual convenience, the graphical representation includes grey boxes to group similar entities; however, the grey boxes have no ontological meaning.

As the name suggests, the DataTransformation pattern is an established representation of the transformation of sourced data. We implement Process as an abstraction of DataTransformation concepts. From the pattern, the notion of Roles, which describes the part that a particular concept plays in some transformation activity, can be established and assigned to concepts that are accepted and produced by a Process. Thus, OutputRole represents the role for concepts produced by Process. However, Boxology has functional usages within multi-step tasks that require the *product* of a Process to frequent the notion of playing the part of InputRole for the next Process. EASY-AI incorporates the notion of Roles to be assigned to Boxology *instances*.

2.2.2. Axioms

The EASY-AI framework provides both a schema to describe concept connections via relationships and also implements axioms, or *constraints*, to describe the shape of how concepts and relationships adhere. EASY-AI incorporates 8 axioms, as depicted in Figure 2, which is a condensed list as provided by [11]. The axioms constrain the shapes to which a relationship R is applied between the concepts A and B. **Axiom 1** represents a subsumption relationship to describe that properties of *Concept A* are inherited from the parenting *Concept B*. **Axiom 2** asserts *scoped domain* of R, scoped by A, is B. **Axiom 3** asserts *scoped range* of R, scoped by A, is B. **Axiom 4** asserts that for every A, there has to be a relationship R with B. **Axiom 5** asserts that for every B, there has to be an inverse

1. $A \sqsubseteq B$	4. $A \sqsubseteq \exists R.B$	7. $B \sqsubseteq \leq 1R^{-}.A$
2. $\exists R.B \sqsubseteq A$	5. $B \sqsubseteq \exists R^{-}.A$	8. $A \sqsubseteq \geq 0R.B$
3. $A \sqsubseteq \forall R.B$	6. $A \sqsubseteq \leq 1R.B$	

Figure 2. A and B represent a conceptual class. R represents a relationship that is applied to connect the conceptual classes. $\sqsubseteq$ indicates a 'SubClassOf' relationship. $\exists$ represents 'Existentiality'.

relationship of R with A. **Axiom 6** asserts that for every A, there is at most one relationship R with B. **Axiom 7** asserts that for every B, there is at most one inverse relationship of R with A. **Axiom 8** asserts that an A may have a relationship R with B. This axiom is all encompassing for the whole ontology, as it is explicit to represent the possibility of existence.

These are the broad constraints that preside over the data transformation axioms that we have adopted. Boxology *instances* with their relationships are described within the EASY-AI framework in Sections 2.3-2.5. Boxology *processes* with their relationships are described within Sections 2.6-2.13.

2.3. Actors

The concept of Actor can be defined as an entity, whether human or otherwise (i.e., software), which engages in actions or behaviors within a system or environment.
Actor-performsInputRole-InputRole: An Actor can only ever perform as an InputRole to a Process. This is constrained by Axioms 3, 4, 6, and 7.
Actor-participatesInProcess-Engineer: An Actor is always present with the Process abstracted to Engineer. This is constrained by Axioms 4, 5, and 7.

2.4. Artifacts

EASY-AI unifies the concept of Data and Symbol to be implemented with the notion of Artifacts, as the abstracted concepts have shared relationships between other concepts. As with [2], the distinction of Data and Symbols, as defined in [1], are still followed for EASY-AI. Data can be represented similar to other AI systems, which include numbers, text, tensors, and streams (representing a sequence of data). Symbols are distinct from Data as Symbols represent labels, relationships between concepts, and records reflected by Data.
Artifact-participatesInProcess-(Transform or Train or Deduction): Artifacts can participate in either Processes of Transform, Train, and Deduction. The relationship of participatesInProcess with concepts of Model and Train or Deduction are not constrained by any axioms.
Artifact-performsInputRole-InputRole: An Artifact can perform as an InputRole to a Process. This is constrained by Axioms 3, 4, 6, and 7.
Artifact-performsOutputRole-OutputRole: An Artifact can perform as an OutputRole to a Process. This is constrained by Axioms 3, 4, 6, and 7.
(Data and Symbol) SubClassOf-Artifact: Data and Symbol inherit from Artifact. This is represented by Axiom 1.

2.5. Models

In the same nature as Artifacts, a top-level description is used to describe Models. As described in [2], models are descriptions of entities and their relationships.
Model-participatesInProcess-(Train or Deduction): A Model is a participant of either Processes of Train or Deduction. The relationship of participatesInProcess with concepts of Model and Train or Deduction are not constrained by any axioms.
Model-performsInputRole-InputRole: An Model can only ever perform as an InputRole to a Process. This can be constrained with Axioms 3, 4, 6, and 7.
Model-performsOutputRole-OutputRole: An Model can only ever perform as an OutputRole to a Process. This can be constrained with Axioms 3, 4, 6, and 7.
SemanticModel-SubClassOf-Model: Models can be abstracted for distinction; such as, the representation of SemanticModels, implemented with Axiom 1. SemanticModels represent the meaning of symbols identified by their concepts, and relationships [14].

2.6. DataTransformation

EASY-AI inherits the DataTransformation pattern, as previously stated in Section 2.2.1. The axioms that govern these are:
DataTransformation-providesRole-(InputRole or OutputRole): providesRole is adopted by the DataTransformation pattern. This relationship can connect to either an InputRole or OutputRole concept, which will be later assigned to a respective Boxology *instance*. As described in [13], the Roles allow for the distinction of sourced data to retain its representation after an applied process or transformation, as these tend to be destructive to the originating sourced data. The axioms applicable to this relationship are: Axioms 2, 4, 5, 6, and 7.
DataTransformation-implements-Process: A DataTransformation can be implemented with a particular Process. The relationship of implements between the concepts of DataTransformation and Process can be applied with Axioms 2, 3, 5, and 7.

2.7. Process

As defined in [15], Processes represent a series of actions or operations to an end.
Process-SubClassOf-DataTransformation: Process inherits from DataTransformation. This is represented by Axiom 1.

2.8. Generate

Generate acts as an abstraction of a Process. The axioms that govern these are:
Generate-SubClassOf-Process: Generate inherits from Process. This is represented by Axiom 1.

2.9. Engineer

Engineer only occurs when a Process involves an Actor. The axioms that govern these are:
Engineer-SubClassOf-Generate: Engineer inherits from Generate. This is represented by Axiom 1.
Engineer-accepts-Actor: The Process of Engineer can only accept Actors. This is constrained by Axiom 2, 3, 4, 5, 6, and 7.

2.10. Train

Generate acts as an abstraction of a Process. The axioms that govern these are:
Train-SubClassOf-Generate: Train inherits from Generate. This is represented by Axiom 1.
Train-accepts-Artifact: Processes of Train can only accept Artifacts. This is constrained by Axioms 3, 4, and 6.
Train-produces-Model: Processes of Train can only produce Models. This is constrained by Axioms 2, 4, and 6.

2.11. Infer

Infer acts as an abstraction of a Process. This process typically results in a conclusion or opinion based on known facts [15]. The axioms that govern these are:
Infer-SubClassOf-Process: Infer inherits from Process. This is represented by Axiom 1.
Induction-SubClassOf-Infer: Induction inherits from Infer. This is represented by Axiom 1.
Deduction-SubClassOf-Infer: Deduction inherits from Infer. This is represented by Axiom 1.
Deduction-accepts-(Artifact and Model): Processes of Deduction only occurs when the InputRoles are a joint combination of Artifact and any Model.

This is constrained by Axioms 2, 3, 4, 5, 6, and 7.
Deduction-produces-Symbol or Model: Processes of Deduction can either produce Symbols or Models. The relationship of produces with concepts of Deduction and Symbol or Model are not constrained by any axioms.

2.12. Transform

Transform acts as an abstraction of a Process. The axioms that govern these are:
Transform-SubClassOf-Process: Transform inherits from Process. This is represented by Axiom 1.
Transform-produces-Data: Processes of Transform can only produce Data. This is constrained by Axioms 2, 3, 4, 5, 6, and 7.

2.13. Embed

Embed is a particular kind of Transform process if, and only if, the Transform process accepts Symbols and SemanticModels and produces Data.
Embed-SubClassOf-Transform: Embed inherits from Transform. This is represented by Axiom 1.
Embed-(accepts-Symbol and SemanticModel$\cup$produces-Data) : Processes of Embed only occurs when the InputRoles are a joint combination of Symbol and SemanticModel and the OutputRole is performed by Data. This is constrained by Axioms 2, 3, 4, 5, 6, and 7.
Embed-produces-Data: Processes of Embed only provides an OutputRole to Data. This is constrained by Axioms 2, 3, 4, 5, 6, and 7.

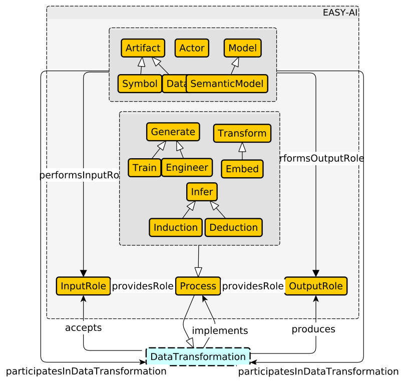
2.14. Availability and Licenses

The ontology documentation, which includes the serialization of the ontology into Turtle syntax and each schema diagram for every pattern in [15], is provided online via a GitHub repository.[1] These resources are licensed permissibly under the MIT license.

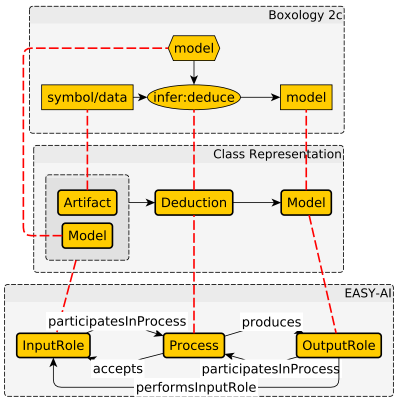
3. Use-Case Scenario

To represent EASY-AI's capabilities and real-world application, we have applied our framework to a Roomba use-case explored in [3]. This Roomba utilizes a Reinforcement Learning (RL) core with 2 attached sensors allowing navigation throughout an office space. The EASY-AI framework for this use-case can be seen in Figure 4. To simplify the construction of this Roomba in EASY-AI's framework, by not reiterating the 1-to-1 translation, we have presented our rendition of this use-case with color-coded labels: **Blue boxes** represent the ontological representation of Boxology *instances* as Input, as described in Section 2.1.1. **Purple boxes** correspond to the ontology representation for Output. **Yellow boxes** correspond to the ontology translation of Boxology *processes* as Process. These colors act as a guide as this is the first introduction to EASY-AI and its application from theory to the real world. The Roomba use-case puzzles together the patterns to represent the base use-case; however, the modularity of EASY-AI allows us to piece the reinforcement learning aspect into the robotic devices' process providing a clear and detailed formalization of the internal workings of the Roomba under the Boxology patterns. This use-case serves as a working example of how we can pair the Boxology patterns as an acting top layer of observation for a user, fulfilling the visual aspect of representing an AI system, while having a second layer of EASY-AI that ontologically displays the intricacies and relationships more descriptively but staying easily tied to the visuals of the Boxology patterns in a real-world application.

[1]`https://github.com/kastle-lab/easy-ai`



(a) Graphical representation of EASY-AI ontology.



(b) Graphical representation of transforming Boxology elementary pattern 2c to an ontology design pattern.

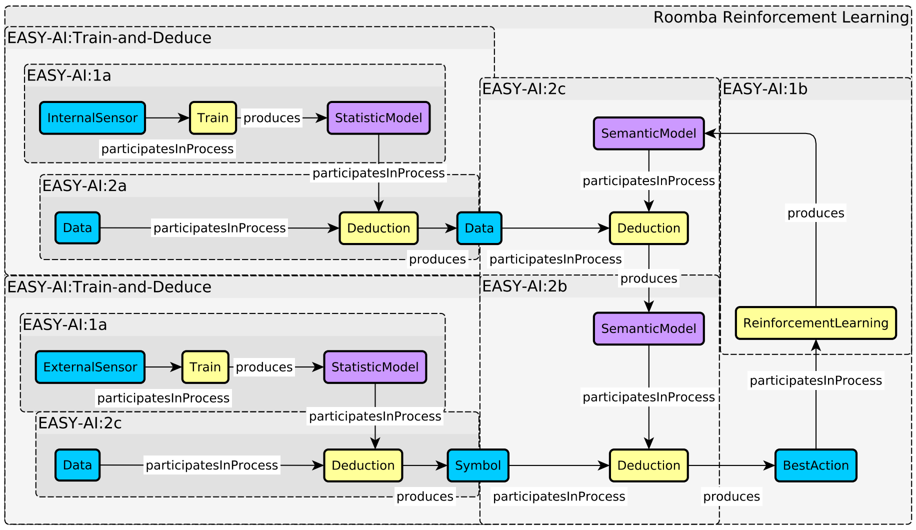


Figure 4. Graphical representation of implementing reinforcement learning into an AI-enhanced Roomba.

4. Conclusion

The EASY-AI framework provides an underlying formalization for the graphical structures from the already established visual Boxology. The goal of EASY-AI is to provide an in-depth inspection of the concepts and rules that govern the individual aspects we see in the Boxology patterns. This in-depth inspection will allow for a more established comprehension of AI systems and more effective communication between not only expert users in collaboration but also everyday users. EASY-AI will provide the grounds for a healthy synergy in human-machine interaction.

Future Work There are several foreseeable next steps with this framework. We anticipate the connection of additional ontological resources, including the provenance ontology [9] and computational observation and environment patterns [12,4] for significantly improved descriptions of AI software systems. furthermore, we now must also connect the actual glpyhs to underlying formalisms herein described. Finally, we will leverage EASY-AI building tools similar to Orange [7], where the logical constraints can accelerate auto-complete-like functionality when constructing, analyzing, and visualizing AI systems.

References

[1] I. S. N. Berkeley. What the <0.70, 1.17, 0.99, 1.07> is a Symbol? *Minds and Machines*, 18(1):93–105, Mar. 2008.

[2] F. V. Harmelen, Department of Computer Science, Vrije Universiteit Amsterdam, Netherlands, A. T. Teije, and Department of Computer Science, Vrije Universiteit Amsterdam, Netherlands. A Boxology of Design Patterns forHybrid Learningand Reasoning Systems. *Journal of Web Engineering*, 18(1):97–124, 2019.

[3] T. D. Hawkes and T. J. Bihl. Symbols to represent AI systems. In *NAECON 2021 - IEEE National Aerospace and Electronics Conference*, pages 61–68, Dayton, OH, USA, Aug. 2021. IEEE.

[4] D. Huo, J. Nabrzyski, and C. F. V. II. An ontology design pattern towards preservation of computational experiments. In C. Keßler, J. Zhao, M. van Erp, T. Kauppinen, J. van Ossenbruggen, and W. R. van Hage, editors, *Proceedings of the 5th Workshop on Linked Science 2015 - Best Practices and the Road Ahead (LISC 2015) co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, Pennsylvania, USA, October 12, 2015*, volume 1572 of *CEUR Workshop Proceedings*, pages 15–18. CEUR-WS.org, 2015.

[5] C.-W. Joy Lo, H.-W. Yien, and I.-P. Chen. How Universal Are Universal Symbols? An Estimation of Cross-Cultural Adoption of Universal Healthcare Symbols. *HERD: Health Environments Research & Design Journal*, 9(3):116–134, Apr. 2016. Publisher: SAGE Publications Inc.

[6] C. Mühlenbeck and T. Jacobsen. On the origin of visual symbols. *Journal of Comparative Psychology*, 134(4):435–452, 2020. Place: US Publisher: American Psychological Association.

[7] I. P. Popchev and D. Orozova. Algorithms for machine learning with orange system. *Int. J. Online Biomed. Eng.*, 19(4):109–123, 2023.

[8] M. A. Riordan. Emojis as Tools for Emotion Work: Communicating Affect in Text Messages. *Journal of Language and Social Psychology*, 36(5):549–567, Oct. 2017. Publisher: SAGE Publications Inc.

[9] S. Sahoo, D. McGuinness, and T. Lebo. PROV-o: The PROV ontology. W3C recommendation, W3C, Apr. 2013. http://www.w3.org/TR/2013/REC-prov-o-20130430/.

[10] C. Shimizu, Q. Hirt, and P. Hitzler. MODL: A modular ontology design library. In K. Janowicz, A. A. Krisnadhi, M. Poveda-Villalón, K. Hammar, and C. Shimizu, editors, *Proceedings of the 10th Workshop on Ontology Design and Patterns (WOP 2019) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 27, 2019*, volume 2459 of *CEUR Workshop Proceedings*, pages 47–58. CEUR-WS.org, 2019.

[11] C. Shimizu, P. Hitzler, Q. Hirt, A. Shiell, S. Gonzalez, C. Foley, D. Rehberger, E. Watrall, W. Hawthorne, D. Tarr, R. Carty, and J. Mixter. The enslaved ontology 1.0: People of the historic slave trade. Technical report, Michigan State University, East Lansing, Michigan, April 2019.

[12] C. Shimizu, P. Hitzler, and C. F. V. II. A pattern for modeling computational observations. In V. Svátek, V. A. Carriero, M. Poveda-Villalón, C. Kindermann, and L. Zhou, editors, *Proceedings of the 13th Workshop on Ontology Design and Patterns (WOP 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022), Online, October 24, 2022*, volume 3352 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[13] C. Shimizu, R. McGranaghan, A. Eberhart, and A. C. Kellerman. Towards a modular ontology for space weather research, 2020.

[14] A. Teije and F. Harmelen. Chapter 3. Architectural Patterns for Neuro-Symbolic AI. July 2023.

[15] M. van Bekkum, M. de Boer, F. van Harmelen, A. Meyer-Vitali, and A. t. Teije. Modular design patterns for hybrid learning and reasoning systems: a taxonomy, patterns and use cases. *Applied Intelligence*, 51(9):6528–6546, 2021. Publisher: Springer.