

Datish: A universal conceptual modeling language to model anything by anyone

Roman Lukyanenko¹, Binny M. Samuel², Jeffrey Parsons³, Veda C. Storey⁴ and Oscar Pastor⁵

¹ University of Virginia, 140 Hospital Dr, Charlottesville, 22903, United States

² University of Cincinnati, 2920 Woodside Drive, Cincinnati, OH 45219, United States

³ Memorial University of Newfoundland, P.O. Box 4200, 230 Elizabeth Avenue St. John's, NL A1C 5S7 Canada

⁴ Georgia State University, P.O. Box 3965, Atlanta, GA 30302 Country, United States

⁵ Universidad Politécnica de Valencia, Camí de Vera, s/n, 46022 València, Spain

Abstract

This paper presents the architecture of a universal conceptual data modeling language, Datish, with the aim of enabling anyone to model anything. Although there are many conceptual modeling languages, there was no language that could model a wide range of domains and at the same time be used by diverse audiences, including the general public. In this paper, we present the motivation, theoretical foundations, and the architecture of such language, Datish. We then illustrate its use in a real-world scenario. We conclude by discussing promising avenues for future conceptual modeling research using Datish.

Keywords

Conceptual modeling, Datish, universal conceptual modeling, lightweight conceptual modeling

1. Introduction

Conceptual models are formal or semi-formal representations that support information technology (IT) development and use. The conceptual modeling community commonly assumes that “there is no universal [conceptual modeling] approach and no universal language” *because of the wide variation of modeling needs* [1, p. 2]. The broadening of modeling to wider audiences and increased need to simultaneously model different systems and domains, motivates us to rethink this assumption.

Recently universal conceptual modeling – inclusive modeling of anything by anyone – has attracted increased attention. Notably, principles of universal conceptual modeling have been proposed [2] and a vision for inclusive conceptual modeling has been formulated [3]. Consistent with these ideas, this paper (1) *considers* the opportunities and challenges of a universal conceptual modeling language, and (2) *proposes* the architecture of a new conceptual data modeling language *Datish* (as in *English* or *Spanish*).

The explicit aim of Datish is to be usable in any situation and by anyone. A conceptual *data* model (*data model* for short), as a type of conceptual model, is a representation of form and structure of a domain to facilitate data collection, storage, retrieval, and interpretation [4], [5]. Popular graphical data models, such as ER diagrams or UML class diagrams, are particularly valuable for database design and for understanding the relevant things in a domain [6]–[8]. Despite the many data modeling languages created since the 1970s, none are at the same time general-purpose and usable by anyone. All have limitations on the types of rules they can model and/or target more seasoned modelers. With Datish, we explicitly seek to attain both.

Datish addresses several challenges noted by researchers and practitioners [2], [6], [8]–[10]. There is a growing need for conceptual models of a variety of systems by an ever-expanding

ER2023: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium, November 06-09, 2023, Lisbon, Portugal

✉ romanl@virginia.edu (R. Lukyanenko)



© 2023 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).



CEUR Workshop Proceedings (CEUR-WS.org)

cohort of modelers (e.g., end users, novices, users with disabilities). The new Datish language can also offer additional support in more novel systems development scenarios cases (e.g., when building social media or AI systems or selecting a NoSQL database technology). Having a common language should promote communication and mutual understanding among different stakeholders and facilitate an integrated use of IT and data.

The aim of Datish is to be accessible to as many users as possible, while permitting modeling of any domain (albeit at a *medium-to-high level of abstraction*). Datish can be used on its own or with existing (more specialized) conceptual modeling languages when more nuanced rules are needed. Datish can also be extended to permit more granular modeling for specialized uses. The remainder of this paper provides our motivation for a universal conceptual modeling language generally, presents the architecture of Datish, and illustrates its use.

2. Background: Motivation and review of relevant work

There is a wide array of benefits of universal conceptual modeling, given the broad range of possible applications and users [2], [11]. We briefly summarize the key arguments that are relevant for our aims (for details, see [2], [3], [11]).

Modeling for everyone. An important societal trend is involvement of ordinary people, such as non-IT employees or members of the public, in data modeling tasks. Research following this trend suggests traditional modeling approaches struggle to support non-IT experts [9]. Existing languages are quite complex and often difficult to comprehend [12], [13]. It would be useful to filter out non-core elements in a complex language, as the presence of more advanced features (e.g., participation cardinality) has been shown to impede domain understanding, especially by modeling novices [9].

Modeling emerging systems and domains. Practitioners struggle to use conceptual modeling to support emerging applications such as artificial intelligence applications, social media, NoSQL, and data lakes [14]–[16]. Many popular approaches to development, such as Agile or DevOps, routinely forgo formal specification [17]. Informal and lightweight modeling can better support these practices [18].

Broader systems and data integration. Many modern technologies are used together and integrated (e.g., AI-based ad recommendations on a social media platform powered by a highly scalable distributed database storage and computational technology). A language capable of modeling different domains at the same time could be especially valuable for systems that integrate data across an organization.

These challenges can be potentially addressed with a more flexible and universally accessible conceptual modeling language. We believe such universal modeling language is feasible for several reasons. First, similar precedents for a universal language exist in the field of computing, such as binary and assembly code. In databases, SQL has been a universal language for querying databases that has remained dominant even after the emergence of NoSQL. Second, a modeling language manifests principles of communication, language, and design [2]. There is considerable commonality in how humans represent reality, thus making it possible to model diverse, complex, and emerging systems and domains for various people.

A universal language model should enable anyone to model anything. Existing approaches to conceptual modeling struggle to satisfy these requirements at the same time. Several established conceptual data modeling languages are used for a wide range of applications, e.g., the ER model and UML class diagrams are de facto standards for relational database design and software engineering, respectively. Extensions of these languages make them more expressive (e.g., extended ER model, [19]) and their semantics more precise (e.g., OntoUML, [20]) and accessible (e.g., ConML, [21]). Building on these, other wide-applicability efforts include enterprise modeling languages, such as Domain Modelling [22], ArchiMate [23] or SysML [24].

Despite their wide applicability, established languages do not meet our requirements for a universal language. First, they have known limitations for some modeling tasks. Notably, a common feature of these languages is requiring or encouraging objects to be members of

predefined entity types or classes (i.e., inherent classification [25]). This makes it especially challenging to use these languages to support evolving requirements and heterogeneous data in artificial intelligence, analytics, social media, and NoSQL contexts [15], [26]–[28].

Second, these languages are geared toward a technical audience (cf. ConML, [21]), and not intended for complete novices and members of the public. Some even presuppose highly advanced technical skills (e.g., [22]). Similarly, most of these languages are quite complex, making their learning and use curve steep for non-expert modelers.

Flexible language efforts include frameworks, such as RDF, Petri nets, graph models and their extensions (e.g., HERAKLIT [29]). While applicable to many applications, these approaches do not consider broad audiences and cater to seasoned developers. Also, it is questionable whether the extreme flexibility of, for example, RDF results in excessive construct overload (e.g., when the same element can be used for modeling individuals and categories).

Efforts to make conceptual modeling more accessible to less technical users include ConML [21] and FlexiSketch [30], which enable modeling to have new uses and users. However, they have notable restrictions: the discussed inherent classification assumption of ConML and the visual forms focus of FlexiSketch. They also cater to modelers with some experience. For example, the “target users of FlexiSketch are: (i) software or systems engineers who create sketches during a system development project and (ii) requirements engineers (business analysts) who use sketches to create and communicate requirements” [30, p. 1513].

3. Datish grammar: Constructs and the rule

We develop Datish as an explicitly general-purpose language with the specific aim to be usable for anyone, including domain novices and experts. It is medium-agnostic and thus, can be realized in many forms (e.g., visual, sound, text), hence, better supporting users with sensory impairments (e.g., blind). To achieve these properties of the language, we implement the Principles of Universal Conceptual Modeling (UCM) [2]. The five principles of flexibility, accessibility, minimalism, primitivism, and modularity provide the foundation for the development of the Datish, and our strategy is to consider all principles together.

Based on the principles of modularity, minimalism, and primitivism, Datish has *Formative* and *Structural Modules*. The Formative Module shapes the form of the language and is comprised of two constructs - *object* and *description* - and a single rule. This is the simplest possible architecture for a language, and a strong adherence to the minimalism principle: any grammar being a conceptual system must have at least two logically connected components [31]. While it is possible to have valid Datish models using only the constructs from the Formative Module, for practical purposes, constructs from the Structural Module will likely be necessary. These constructs include *individuals*, *categories*, and *relationships*.

Construct 1: Object. The core construct of Datish is object. This choice was the outcome of a broad search across diverse interdisciplinary literature and simultaneous consideration of different UCM principles.

General ontology is a branch of philosophy that studies what exists in reality. Ontological theories have generally pursued two major approaches: substance and process. In substance ontology, the principal unit of reality is some atomic unit or substance [32], [33]. In process ontologies, events are the fundamental elements of reality, either on par with substances or superseding substances [34]–[36].

By far the most common philosophical approach is the ontology of substance [34], which holds that the *basic element of being* is what is referred to as entity, object, thing, or substance. Some consider these terms synonymous. Halpin [37, p. 4], for example, defines object as “any individual thing of interest.” Others suggest there are nuanced differences; for example, Bunge [38, p. 294] defines thing as “an object other than construct [i.e., product of thought, such as idea or concept].”

Conceptual modeling widely uses *object* as a key modeling construct. Object has been a core construct in conceptual modeling languages (e.g., UML, ORM and OASIS) [37], [39], [40]. The concept is important in systems development more broadly (as in object-oriented analysis and

object-oriented programming). Object is a foundational notion in a number of ontologies, such as UFO [41], [42], BFO [43], Object-oriented ontology [44], systems ontology [45], [46].

We now synthesize the substance view of object with the process philosophy. Although Datish is a modeling language for structure and form (rather than dynamics and flow), support for process modeling is desirable, as processes are intrinsically intertwined with substances. This is guided by the flexibility, accessibility and ubiquity principles. Process is a common way to model reality. The more Datish can support this modeling, the more flexibility and universality is attained.

An approach we take follows two philosophical positions. First, there are philosophies that claim that an absolute philosophical primitive is necessary to have a unified view of reality. This motivation is especially consistent with the requirements for Datish. An embodiment of this approach is a recent revision of Bunge's ontology [46]. In his later writings, Bunge proposed an "absolute primitive" notion of "object," Namely, "[w]hatever can exist, be thought about, talked about, or acted upon" [38, p. 199].

Second, while Bunge does not address the relationship between process and substance in the unified notion of object, several promising attempts have been made to show how, by modeling objects, one can also capture domain dynamics in a more process-faithful manner. The Unified Foundational Ontology (UFO), for example, proposes to consider events to be entities, such as "act of music composition", "marriage event" [47], [48]. This view is also consistent with Object-oriented ontology [44], which considers events such as "fight" or "leaves turning green" as objects. Hence, object-events can have properties [44]. This position is compatible with the modeling practice of reified relationships [48], [49] and some graph databases (e.g., Neo4J). Based on the principles of flexibility and commonality, Datish objects can be used to represent events.

Common among both process and substance ontologies is the belief that things, objects, or entities can be uniquely identified; that is, being recognized as unique and different from other objects. No two objects are the same [33]. By synthesizing the above perspectives on the nature of objects we provide the following definition:

Definition 1 - Object: *An object is anything that can exist, be thought about, talked about, or acted upon, and be distinguished from other objects.*

This general approach permits an object to be anything- a concrete entity, such as a chair, an idea of a perfect capitalist market, a process of creating a work of art, an association between two people, as well as a group of celestial bodies under the label "planet." Furthermore, in different domains, especially in science, the ambiguity of the object notion is desirable. For example, in biology, it is not clear whether some objects (e.g., species) are individuals or categories. Hence, object is a suitable foundational construct for the language like Datish.

Construct 2: Description. To be useful, an object in a model needs to carry some information. This is needed for the user of the model to understand what object is being represented and from what perspective. An object, therefore, needs to be expressed with some additional information. We call this a *description*.

Definition 2 - Description: Description is a statement that communicates some relevant aspect(s) of an object by any linguistic or extra-linguistic means.

A description does not require any specific form (since Datish is agnostic of the medium). Instead, we suggest for Datish to have *patterns of object descriptions*. These patterns are templates that optionally "specialize" the description construct of Datish based on modeling needs. Below, we suggest several such patterns.

Identifier. As Definition 1 suggests, *identity* is an essential element of the object construct. Identity can be realized as unique identifiers, which can be local or global in scope [50]. These can be names, numbers, or textual descriptions. They can also take more advanced forms, such as hashes, QR-codes, URLs, RFIDs, or digital signatures. Note, by itself, an identifier only permits distinguishing among objects. Ideally, an identifier should point to additional description (e.g., a URL to the contents of a webpage).

Attributes. Objects are commonly understood as bundles or collections of properties or *attributes*. For example, attributes are often understood as characteristics or features of an object

used to identify or categorize it (e.g., color, shape, and size) [51]. Commonly, objects can be described using a list of their applicable attributes.

Text. Objects can be described using more complex linguistic structures. These permit the depiction of complex relationships among the attributes. For example, “the underdog election winner (the object) challenged all the norms of political behavior” is a complex association between the attributes of the object and the beliefs and norms set in the broader community. To understand an object, it is sometimes necessary to position it within a broader context, so even richer descriptions are necessary. For example, *sikuaq* is a particular type of thin ice in the Inuktitut language. To understand its nature, a richer textual description of the culture of the Inuit in Canada is necessary. Importantly, it would be challenging to convey the richness, subtlety, and nuances of some objects by using a list of attributes alone.

Multimedia. Objects can be described by unwritten means. These include images, videos, sounds, smells, and other sources of sensory experience. Indeed, research in conceptual modeling has already begun considering multimedia forms, such as symbols, pictures and even virtual reality [52], [53].

Additional patterns of the description may exist. For example, an extension to Datish could use attribute templates (e.g., type-date, number, text, possible default value, nulls allowance, changeable aspect, constant or variable, derived nature) for an application where the types and properties of attributes are important.

In sum, to describe objects flexibly and inclusively, the conventional approach of using a list of attributes is not suitable for all possible situations. Hence, we adopt a novel use of *description* in conceptual modeling to include alternative forms a user may wish to use. The objective of the description is to convey something important about the particular object, especially if it enables distinguishing between one object and another. Finally, as an object by itself does not convey meaning, we introduce the only rule of Datish:

Datish Rule. Object description: All objects must have a description.

Taken together, the two constructs and the rule described above constitute the minimal elements of Datish grammar. It is conceivable to create Datish models consisting only of objects along with their descriptions. Such minimalism may be useful in cases where very little is known about the nature of these objects, or when showing their relationships may not be necessary.

A given object may have multiple descriptions (which is especially valuable if these descriptions are coming from multiple users or are taken from multiple perspectives). A description may not be an accurate, complete, or true way of communicating information about an object. For this reason, objects cannot be reduced to their descriptions. Descriptions are mental constructs (in themselves, mental objects) created by modelers to communicate something of value about other objects of interest.

Finally, a description of an object in one model can be modeled as an object in another model (which also implies it will need its own description). This can be particularly useful for capturing the provenance or metadata about the original description of the object.

We assume every representation in Datish represents either an object or a description of an object. As data modeling deals with the form and structure of the domain, Datish provides additional constructs for structuring objects. Specifically, all additional constructs in Datish are types of objects (which also implies they must abide by the Rule; they need to have a description).

Construct 3: Individual. A fundamental approach for capturing the structure of a domain is in terms of individual or groups of individuals. Most existing conceptual modeling languages draw a distinction between particulars and universals, also known as individuals and categories, or instances and classes, respectively. At the same time, conceptual modeling traditionally focused on representing universals or categories, under the “assumption of inherent classification” [25], [54]. The traditional approach to modeling underrepresents the essential role of individuals in reality and for modeling reality [15], [27]. In many philosophies, the world is made of unique individuals, or things (atomic or complex). For example, according to Bunge things (e.g., specific planets, birds, trees, atoms) are the primary constituents of reality [33]. In contrast, some categories are secondary, in that humans use the categories to group existing things with common attributes (i.e., category of planet, bird, tree, atom).

Individuals matter even in cases where categories may come before an instance. Hence when dealing with social reality where categories are typically created before individual objects, once instances of the social categories are created they can take on their own existence [55]. For example, in the late 1970s the conceptual modeling community decided to create a new category: “Conference on Conceptual Modeling” [56]. Initially, it was merely a mental idea that lacked specific members. Yet, since 1979, the conferences, which became known as ER, have been held annually [56]. Each such conference is itself an object, with unique properties (descriptions), somewhat different from those of other conferences.

Reasoning with individual objects is important for human cognition. While people experience continuous sensory input (e.g., light falling on retina, sound waves), they invariably transform their sense data into distinct mental objects [57]. Instances are important for everyday naïve thinking about reality. In day-to-day life, the level that is naturally accessible to humans is that of “middle-sized” objects - those that can be “picked out using unaided human sensory capacities” [58, p. 1], such as trees, animals, or rocks [10], [15].

Leveraging these ontological and cognitive benefits, representation of individuals is widespread in mathematics, logic, and computing. In logic, including predicate calculus and extensional logic, individuals can be directly modeled [59], [60]. Datish has a direct support for representing individuals, irrespective of whether they are members of predefined categories.

Definition 3 - Individual: Any specific, singular object, abstract or concrete.
Construct 4: Category. Categories (or concepts) organize individual objects into groups based on similarity of their features or common patterns of use. In the ontological view that the world is made of substantial (concrete, material, physical) individuals, a category is a non-essential and observer-dependent construct [33]. However, conceptual modeling is a social activity, performed mainly by human beings [61]. Humans create categories to group concrete objects in useful ways. In many scenarios, humans think more in terms of categories than individual objects - “humans are compulsive classifiers” [62], [63]. Classification is central to human perception, memory, reasoning, problem-solving and communication [64], [65]. Furthermore, in social domains, categories are typically created before any individual members of the categories exist [50], [55], [61], [66]. Categorization is a vital cognitive mechanism to manage the infinite diversity of stimuli in the real world. Categories capture the similarity among the objects, thereby allowing these objects to be conceptualized and treated in a similar manner. For example, it may be more efficient to refer to all objects having certain common attributes as “birds”, thereby eliminating redundant descriptions of their shared attributes.

The use of categories also permits “completely” representing domains [15], in that they capture generalizations and abstractions over the “infinitely” diverse individuals. Having a “complete” specification is an important outcome of conceptual modeling [67], [68]. Similarly, categories can group not only individual objects, but other categories, forming hierarchies (e.g., robins, birds, animals). It is impossible to convey domain boundaries and attain the desired level of domain completeness and structuring with individual objects alone. We, thus, propose category as a Datish construct:

Definition 4 - Category: A collection of individual objects or other categories having common characteristics.

Construct 5: Relationship. The principle of minimalism dictates having only essential constructs in Datish. However, a modeler might need to create relationships between multiple individual objects and categories. Furthermore, there can be more than one object in a Datish model, so it is useful to show how these objects are related. Finally, elaborating object relationships is essential for meaningful domain comprehension and learning since it organizes knowledge into coherent structures and integrates new information with prior knowledge [69].

Relationships are ontological primitives. Both in process and substance ontology, reality is based on interactions. This suggests that *interaction* is a fundamental way to describe the connections among objects. Not all objects interact or interact *directly*. Our inclusive definition of objects permits mental thoughts, concepts, and events and processes to be modeled as objects. None of these objects interact with each other, or other objects with mass, in the same manner as two physical objects (e.g., billiard balls) do. We assume concepts, and ideas (e.g., number “3” or

“Anna Karenina”) do not themselves change since they, themselves, do not possess energy. They change when humans (and other sentient beings) think and communicate about these objects [55], [70], and affect physical objects, via, for example, linguistic declarations, such as commands and requests, or *speech acts* [55], [71], [72].

Similarly, categories and other objects are related to one another in some way. For example, one concept can be a subtype of another (e.g., bird is a subtype of animal). These concepts are conceptually (or logically) linked via a “type of” relationship. In other words, categories can be related to categories, as well as to individuals.

Hence, a notion applicable to all objects is a broad concept of relationship, which includes both physical interaction and conceptual linkages among concepts. We thus define a relationship as follows.

Definition 5 – Relationship: A representation of a physical interaction or conceptual connection among one or more objects.

We can now clarify some important implementation choices when using Datish. First, all constructs in a Datish model are objects and must have a description. However, this rule is agnostic of the exact nature of the description. In the simplest case, giving a category name is sufficient to satisfy this rule. The length, form and presentation of the description may vary depending on the purpose of the modeling.

Datish views category differently than most existing conceptual modeling languages. Nearly all conceptual modeling languages manifest the principle of “inherent classification,” whereby individual objects are represented as members of categories (or, classes, entity types) [7], [73], [74]. In its strong form, the principle states that “specific things in the domain of interest (entities, objects, etc.) can be referred to only as instances of classes (variously referred to as entity types, categories, kinds)” [25, p. 229]. The best-known example is an entity-relationship model [75]. A weaker form of this principle is that an object may not be constrained to have only the attributes of its category (i.e., it may have additional, unique attributes), but the category remains the most common way of modeling domains (e.g., ArchiMate).

In Datish, category serves two objectives, consistent with the cognitive benefits of classification and their role in creation of social reality. First, a category is cognitive tool, a convenient shortcut that eliminates the need for lengthy and repetitive descriptions. Second, a category is an object in a Datish model that can be used when no members of category yet exist. Hence, categories in Datish are much more flexible, but also inclusive of their use in traditional conceptual modeling. To enable this flexibility, in Datish it is possible to model categories first, and insist on objects always being members of categories. Furthermore, objects do not have to be members of any given category or can be members of multiple categories. For example, the same object may be simultaneously a member of the category student as well as employee. These variations become modeling choices based on the semantics of a domain. We additionally recommend documenting this choice (e.g., via a simple explanatory note, leveraging the flexibility of the description construct).

Each approach – objects being dependent upon or independent of categories - can be beneficial in different scenarios. When domain rules are well-established, and objects share strong similarity with one another, it may be advantageous to treat them as members of predefined categories. In contrast, in domains characterized by high heterogeneity, it is prudent to model objects independently or somewhat independently of the categories to which they may belong.

Furthermore, categories need not be homogeneous. This is consistent with both the flexibility and ubiquity principles. Natural categories are often heterogeneous [76]. For example, the category “bird” has very diverse members, including birds that do not fly. Similarly, some categories are best described by their exemplars. The category “game” is famous for lacking specific necessary and sufficient conditions (as famously argued by the philosopher Ludwig Wittgenstein). Having a flexible and rich apparatus for describing categories is therefore essential to ensure the nature of categories is well communicated when necessary. Additionally, we provide several suggested approaches for visually depicting homogeneous versus heterogeneous categories below. Emphasizing the way Datish views categories, we introduce two implementation notes:

Datish Note 1 - Object-category: An object may be a member of one or more categories or exist independently of any categories.

Datish Note 2 - Category-object: A category may or may not have defined members (objects); objects-members of category may or may not have the same description (e.g., share all attributes).

Having only five constructs in the modeling toolkit, along with a single rule, offers a great deal of flexibility. At the same time, this grammar enables the modeling of a wide variety of scenarios, domains, and systems.

The descriptions attached to all objects can be simple or complex, depending on the purpose of the modeling. The description need not be next to the object and can be in a separate section of a diagram (as common in architectural blueprints) or another representation [77], to permit longer descriptions.

The specific ways to arrange or express these constructs may vary. It is feasible that some organizations' analysts may customize their own styles or introduce situational or systematic constraints upon the usage of these constructs [78]. For example, some projects may insist on always modeling objects independent of categories; other projects may stipulate that every object must have a unique identifier attached. The flexibility of Datish explicitly enables and encourages these local choices. Extensions upon the core of Datish are also encouraged, especially when the language is used by more technical teams. In such cases additional rules and abstractions familiar to these users (e.g., cardinality) may be introduced.

4. Illustrative application

Datish does not insist on a particular way to graphically depict its constructs. Indeed, we encourage the exploration of different approaches, as well as different presentation mediums (not only paper, but multimedia and interactive virtual environments). To enable exploration of Datish, we suggest a visual notation for Datish based on a two-dimensional format (further discussed below). Figure 1 shows the symbols we deem consistent with the principles of universal conceptual modeling, with some rationale for why the symbols were chosen.



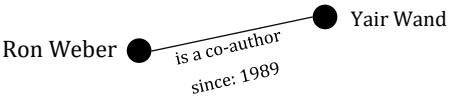
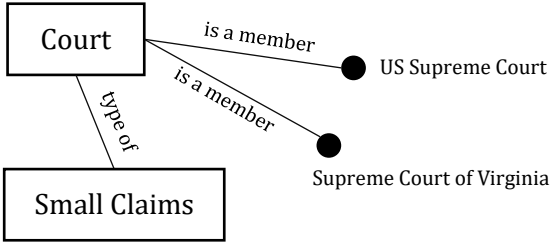
Description	Graphical representation	
Object shown as cloud (universal symbol) with a simple description		
Individual Object shown as filled circle (universal symbol) with name, and attributes	<ul style="list-style-type: none"> ● ID: X56Z5 ● Barack Obama Bird Blue beak ● Gulf of Mexico Oil Spill Crane-like 	
Individual Object shown using multimedia (e.g., video, image) either in the diagram or in the object description section	<ul style="list-style-type: none"> ● ID: X56Z5 	 <p style="font-size: small; margin-top: 5px;">Object: X56Z5</p>
Relationship shown as lines or arrows (both universal shapes)		
Homogeneous and heterogeneous categories shown as square (a universal shape) heterogeneous categories can be shown as squares with distinct members (or can be described in a textual narrative)		

Figure 1: Illustrative visual representations of object and description

Depending on the modeling objectives, description of objects, categories and relationships can be represented with greater or lesser formality and greater or lesser consistency. For example, if Datish is used to model relational databases, descriptions can be lists of attributes, and the unique identifier is unnecessary. If such lists are short, attributes are best shown together with the objects. In contrast, when modeling heterogeneous data to be ingested by a data lake, descriptions may be shown in a separate description section as lists of attributes, representative multimedia, or narratives.

We illustrate the use of Datish in a real-world restaurant modeling case. We model these requirements with Datish (Figure 2). The restaurant domain is suitable for this illustration because it is simple enough to understand, yet as we show, to model the real-world complexity of this domain, we need a flexible language such as Datish to capture this complexity.

Dante Lake is the owner of a restaurant chain called Deats (all names in the case are pseudonyms). Deats serves menu items following recipes with specially sourced local ingredients. Among the employees, Dante employs cooks in restaurants, and utilizes a third-party delivery service, YourDoor, to bring items to customers' homes.

Dante is a data-driven decision-maker and relies on data to make employee performance, food quality, and menu selection decisions. Dante uses several databases to manage the restaurants, including a data lake. These databases store data about employees, items, recipes, ingredients, and orders. Dante uses IoT sensors and their data streams (e.g., log files) to ensure health code compliance on ingredient storage and preparation, including tracking temperature and humidity. Dante also uses data from YourDoor, which provides data via an API in JSON format to its clients. Last, Dante monitors social media opinions of restaurants and uses data from the popular review site Yum, which also utilizes JSON to make data available.

There are several locations of Deats, captured by the homogenous category Restaurant that has a standardized description. Dante also hires employees to work at the restaurants but wishes to record idiosyncratic information about them. The Employee heterogeneous category reflects this. Notice how Stefano is an individual object and one of a kind. Stefano is the head chef for all restaurants, but also an employee who supervises cooks and invents items. Stefano is one of the key reasons for the success of the restaurant, as he, together with his wife, Gina, scout local organic farms and develops the award-winning unique recipes that the customers love so much. Gina is not an employee of a restaurant but has an influence over its operations and occasionally leaves digital traces in her opinions about the meals and modification suggestions.

At the restaurants, cooks prepare items for orders using ingredients that should comply with safety. Safety is a heterogeneous category as different ingredients have varying safety requirements that are based on local and regional public health policies for each Deats location, furthermore special one-off safety rules also exist. Safety includes parameters such as Temperature and Humidity but might also specify other, idiosyncratic requirements. Customers can place delivery or Dine-in orders. Delivery orders are handled by a delivery service (currently YourDoor) that delivers the orders to customers. Customers' descriptions might be tied to Reviews provided by Yum or they may remain independent. We also note the social network of the customers (made of social media connections, friends, family, online influences). It is not modeled in detail, as its complexity exceeds the scope of modeling, but showing it indicates an important source of the influence on customer's behavior and perceptions of meals and service.

This illustration in a very simple domain shows that modeling the real world is complex, nuanced, and messy. At times, individual people are the linchpins of operation, and their oversized role must be captured in the model. In addition, some suspected influences (such as social networks) are based on crude assumptions, to be evaluated with more data later. Some categories in the domain are uniform, whereas others are heterogeneous and focal individual instances of these objects are important to represent. Datish is capable of capturing and representing these nuances.

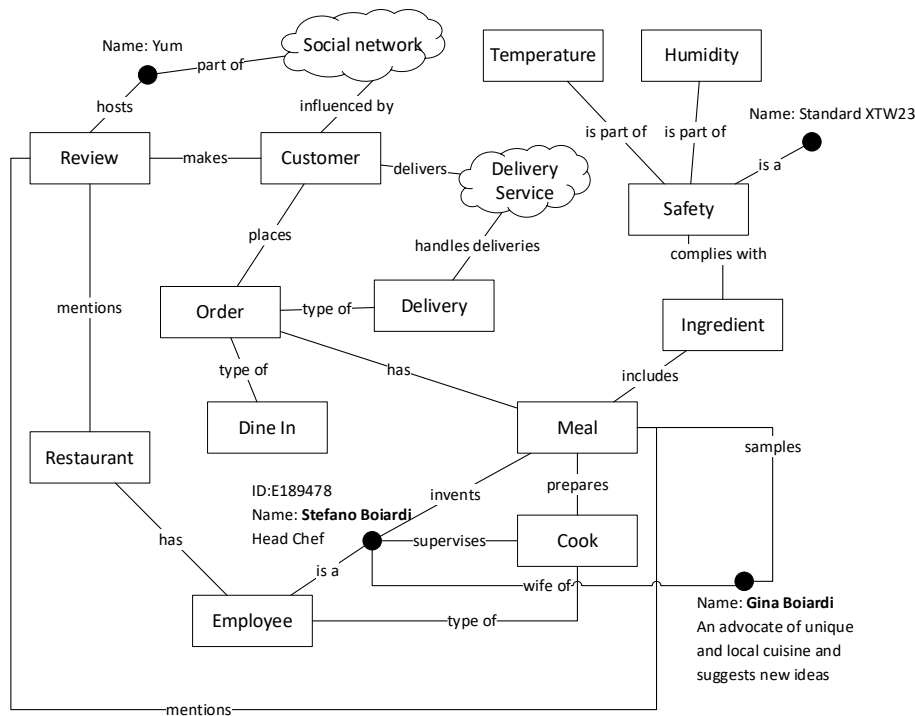


Figure 2: Datish script of the restaurant scenario

5. Discussion and outlook for the future

We argue for the need of a universal language for conceptual modeling and propose Datish to be such language. Datish is based on a set of principles derived from multiple disciplines and is intended to be useful for modeling “anything, anytime by anybody,” reflecting the need for new types of users to be engaged in conceptual modeling and flexible language use. A sample visual representation was presented to illustrate how Datish can be used.

By design, Datish is intended to be lightweight and accessible, yet also expressive. With Datish users can model many rules, supported by the flexible and modular language design. It is even possible to model more advanced rules (e.g., cardinality as descriptions of relationships). However, in its basic form Datish only insists on the constructs and the rule that the general audiences can understand.

Although we presented a possible visual notation for Datish, it is not intended to be final or definitive. Just as Datish constructs are explicitly grounded in the theoretical principles of universal conceptual modeling, similar attention is required to develop Datish visual notations. These need to be grounded in relevant foundations of visual notation development (e.g., Moody [79]), and be rigorously empirically evaluated [80], including by considering different design alternatives [81].

Furthermore, visual is only one of the modalities. Datish is medium-agnostic and future efforts are needed to develop additional ways of representing Datish (e.g., sound, text). This is especially important for supporting users with impairments. As with visual notations, additional modalities require rigorous development and evaluation. Future research could help to refine and develop the visual, auditory, and other modality symbols of Datish as well as to apply it to a large set of modeling applications in various domains. Future work can also consider using artificial intelligence to generate and parse Datish scripts. For example, advanced natural language and computer vision techniques could be applied to descriptions to parse the semantics of Datish automatically.

Another important area for future research is the development of *methods* for using Datish. Datish is inherently flexible, so future research can provide more explicit rules for how to use the language in specific modeling scenarios. One opportunity is to use Datish for lightweight and

informal modeling in Agile and DevOps settings. Alternatively, in cases where requirements are stable, well understood and agreed upon, Datish may be used in a traditional way (e.g., to support relational database design by focusing on category, relationship and description using attributes).

Finally, Datish might become helpful in the recognition of the importance of conceptual modeling in any application domain or content. The ultimate success of the Datish project may not be in the language itself, but in fostering the dialog within the conceptual modeling community on universal and inclusive modeling.

References

- [1] B. Thalheim, "Towards a theory of conceptual modelling.," *J. Univers. Comput. Sci.*, vol. 16, no. 20, Art. no. 20, 2010.
- [2] R. Lukyanenko, J. Parsons, V. C. Storey, B. M. Samuel, and O. Pastor, "Principles of universal conceptual modeling," in *EMMSAD 2023*, Saragosa, Spain: Springer, 2023, pp. 1–15.
- [3] R. Lukyanenko, D. Bork, V. C. Storey, J. Parsons, and O. Pastor, "Inclusive Conceptual Modeling: Diversity, Equity, Involvement, and Belonging in Conceptual Modeling," in *ER Forum 2023*, Lisbon, Portugal: Springer, 2023, pp. 1–4.
- [4] C. E. H. Chua, M. Indulska, R. Lukyanenko, W. Maass, and V. C. Storey, "Data Management," *MISQ Quarterly Online*, pp. 1–10, 2022.
- [5] V. C. Storey, R. Lukyanenko, and A. Castellanos, "Conceptual Modeling: Topics, Themes, and Technology Trends," *ACM Computing Surveys*, vol. 55, no. 14s, pp. 1–38, 2023.
- [6] H. C. Mayr and B. Thalheim, "The triptych of conceptual modeling," *Software and Systems Modeling*, pp. 1–18, 2020.
- [7] J. Mylopoulos, "Information modeling in the time of the revolution," *Information Systems*, vol. 23, no. 3–4, pp. 127–155, 1998.
- [8] J. Recker, R. Lukyanenko, M. A. Sabegh, B. M. Samuel, and A. Castellanos, "From Representation to Mediation: A New Agenda for Conceptual Modeling Research in A Digital World," *MIS Quarterly*, vol. 45, no. 1, pp. 269–300, 2021.
- [9] M. Hvalshagen, R. Lukyanenko, and B. M. Samuel, "Empowering Users with Narratives: Examining The Efficacy Of Narratives For Understanding Data-Oriented Conceptual Models," *Information Systems Research*, pp. 1–38, 2023.
- [10] A. Castellanos, M. Tremblay, R. Lukyanenko, and B. M. Samuel, "Basic Classes in Conceptual Modeling: Theory and Practical Guidelines," *Journal of the Association for Information Systems*, vol. 21, no. 4, pp. 1001–1044, 2020.
- [11] S. Al-Fedaghi, "In Pursuit of Unification of Conceptual Models: Sets as Machines," *arXiv preprint arXiv:2306.13833*, 2023.
- [12] I. Compagnucci, F. Corradini, F. Fornari, and B. Re, "Trends on the Usage of BPMN 2.0 from Publicly Available Repositories," presented at the International Conference on Business Informatics Research, Springer, 2021, pp. 84–99.
- [13] M. zur Muehlen and J. Recker, "How much language is enough? Theoretical and practical use of the business process modeling notation," in *Seminal Contributions to Information Systems Engineering*, Springer, 2013, pp. 429–443.
- [14] D. Bork, "Conceptual Modeling and Artificial Intelligence: Challenges and Opportunities for Enterprise Engineering," in *Enterprise Engineering Working Conference*, Springer, 2022, pp. 3–9.
- [15] R. Lukyanenko, J. Parsons, and B. M. Samuel, "Representing Instances: The Case for Reengineering Conceptual Modeling Grammars," *European Journal of Information Systems*, vol. 28, no. 1, pp. 68–90, 2019.
- [16] S. Nalchigar and E. Yu, "Conceptual modeling for business analytics: a framework and potential benefits," in *2017 IEEE 19th Conference on Business Informatics (CBI)*, IEEE, 2017, pp. 369–378.
- [17] M. Schwartz, *War and peace and IT: business leadership, technology, and success in the digital age*. New York NY: IT Revolution, 2019.

- [18] L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A survey of DevOps concepts and challenges," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–35, 2019.
- [19] T. J. Teorey, D. Yang, and J. P. Fry, "A logical design methodology for relational databases using the extended entity-relationship model," *ACM Computing Surveys*, vol. 18, no. 2, pp. 197–222, 1986.
- [20] G. Guizzardi, C. M. Fonseca, A. B. Benevides, J. P. A. Almeida, D. Porello, and T. P. Sales, "Endurant types in ontology-driven conceptual modeling: Towards OntoUML 2.0," presented at the International conference on conceptual modeling, Springer, 2018, pp. 136–150.
- [21] C. Gonzalez-Perez, "How Ontologies Can Help in Software Engineering," in *International Summer School on Generative and Transformational Techniques in Software Engineering*, Springer, 2015, pp. 26–44.
- [22] D. Bjørner, *Domain Science and Engineering: A Foundation for Software Development*. Springer Nature, 2021.
- [23] C. L. Azevedo, M.-E. Iacob, J. P. A. Almeida, M. van Sinderen, L. F. Pires, and G. Guizzardi, "Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate," *Information systems*, vol. 54, pp. 235–262, 2015.
- [24] L. Lima *et al.*, "An integrated semantics for reasoning about SysML design models using refinement," *Software & Systems Modeling*, vol. 16, no. 3, pp. 875–902, 2017.
- [25] J. Parsons and Y. Wand, "Emancipating Instances from the Tyranny of Classes in Information Modeling," *ACM Transactions on Database Systems*, vol. 25, no. 2, pp. 228–268, 2000.
- [26] P. Atzeni, C. S. Jensen, G. Orsi, S. Ram, L. Tanca, and R. Torlone, "The relational model is dead, SQL is dead, and I don't feel so good myself," *ACM SIGMOD Record*, vol. 42, no. 1, pp. 64–68, 2013.
- [27] O. Eriksson, P. Johannesson, and M. Bergholtz, "The case for classes and instances—a response to representing instances: the case for reengineering conceptual modelling grammars," *European Journal of Information Systems*, vol. 28, no. 6, pp. 681–693, 2019.
- [28] R. Lukyanenko and J. Parsons, "Beyond Micro-Tasks: Research Opportunities in Observational Crowdsourcing," *Journal of Database Management (JDM)*, vol. 29, no. 1, pp. 1–22, 2018.
- [29] P. Fettke and W. Reising, "Systems Mining with Heraklit: The Next Step," in *BPM 2022 Forum*, Münster, Germany: Springer, 2022, pp. 89–104.
- [30] D. Wüest, N. Seyff, and M. Glinz, "FlexiSketch: a lightweight sketching and metamodeling approach for end-users," *Software & Systems Modeling*, vol. 18, pp. 1513–1541, 2019.
- [31] M. A. Bunge, "Systems everywhere," in *Cybernetics and applied systems*, London England: CRC Press, 2018, pp. 23–41.
- [32] J. Benovsky, "The bundle theory and the substratum theory: deadly enemies or twin brothers?," *Philosophical Studies*, vol. 141, no. 2, pp. 175–190, 2008.
- [33] M. A. Bunge, *Treatise on basic philosophy: Ontology I: the furniture of the world*. Boston, MA: Reidel, 1977.
- [34] J. Dupré, "A process ontology for biology," *The Philosophers' Magazine*, no. 67, pp. 81–88, 2014.
- [35] H. Herre, "General Formal Ontology (GFO): A foundational ontology for conceptual modelling," in *Theory and Applications of Ontology: Computer Applications*, Springer, 2010, pp. 297–345.
- [36] A. N. Whitehead, *Process and Reality*. London England: Free Press, 2010.
- [37] T. Halpin, *Object-role modeling fundamentals: a practical guide to data modeling with ORM*. Technics Publications, 2015.
- [38] M. A. Bunge, *Philosophical dictionary*. Amherst, NY: Prometheus Books, 2003.
- [39] I. Jacobson, G. Booch, and J. Rumbaugh, *The unified software development process*, vol. 1. Reading MA: Addison-Wesley, 1999.
- [40] O. P. Lopez, F. Hayes, and S. Bear, "Oasis: An object-oriented specification language," in *International Conference on Advanced Information Systems Engineering*, Springer, 1992, pp. 348–363.

- [41] G. Guizzardi, *Ontological foundations for structural conceptual models*. Enschede, The Netherlands: Telematics Instituut Fundamental Research Series, 2005.
- [42] G. Guizzardi, "Ontological meta-properties of derived object types," presented at the International Conference on Advanced Information Systems Engineering, Springer, 2012, pp. 318–333.
- [43] R. Arp, B. Smith, and A. D. Spear, *Building Ontologies with Basic Formal Ontology*. in The MIT Press. Cambridge, MA: MIT Press, 2015. [Online]. Available: <https://books.google.com/books?id=AUXQCgAAQBAJ>
- [44] G. Harman, *Object-oriented ontology: A new theory of everything*. London England: Penguin UK, 2018.
- [45] R. Lukyanenko, V. C. Storey, and O. Pastor, "System: A Core Conceptual Modeling Construct for Capturing Complexity," *Data & Knowledge Engineering*, vol. 141, pp. 1–29, 2022.
- [46] R. Lukyanenko, V. C. Storey, and O. Pastor, "Foundations of information technology based on Bunge's systemist philosophy of reality," *Software and Systems Modeling*, vol. 20, no. 1, pp. 921–938, 2021.
- [47] J. P. A. Almeida, R. A. Falbo, and G. Guizzardi, "Events as entities in ontology-driven conceptual modeling," in *International Conference on Conceptual Modeling*, Springer, 2019, pp. 469–483.
- [48] G. Guizzardi, G. Wagner, J. P. A. Almeida, and R. S. Guizzardi, "Towards ontological foundations for conceptual modeling: the unified foundational ontology (UFO) story," *Applied ontology*, vol. 10, no. 3–4, pp. 259–271, 2015.
- [49] S. Hitchman, "An interpretive study of how practitioners use entity-relationship modelling in a ternary relationship situation," *Communications of the Association for Information Systems*, vol. 11, no. 1, p. 26, 2003.
- [50] O. Eriksson and P. J. Agerfalk, "Rethinking the Meaning of Identifiers in Information Infrastructures," *Journal of the Association for Information Systems*, vol. 11, no. 8, pp. 433–454, 2010.
- [51] E. Rosch, C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyesbraem, "Basic Objects in Natural Categories," *Cognitive Psychology*, vol. 8, no. 3, pp. 382–439, 1976.
- [52] K. Masri, D. Parker, and A. Gemino, "Using iconic graphics in entity-relationship diagrams: the impact on understanding," *Journal of Database Management (JDM)*, vol. 19, no. 3, pp. 22–41, 2008.
- [53] F. Muff and H.-G. Fill, "Initial Concepts for Augmented and Virtual Reality-based Enterprise Modeling*," in *ER Demos and Posters 2021 co-located with 40th International Conference on Conceptual Modeling (ER 2021)*, 2021.
- [54] V. Ramesh, J. Parsons, and G. Browne, "What is the role of cognition in conceptual modeling? A report on the First Workshop on Cognition and Conceptual Modeling," *Conceptual Modeling*, pp. 272–280, 1999.
- [55] J. R. Searle, *The construction of social reality*. Simon and Schuster, 1995.
- [56] P. Chen, "Entity-relationship modeling: historical events, future trends, and lessons learned," *Software pioneers: contributions to software engineering*, pp. 296–310, 2002.
- [57] S. R. Harnad, *Categorical Perception: The Groundwork of Cognition*. Cambridge, MA: Cambridge University Press, 1990.
- [58] J. Foster, "Ontologies without Metaphysics: Latour, Harman and the Philosophy of Things," *Analecta Hermeneutica*, no. 3, 2011.
- [59] A. Pap, "Disposition concepts and extensional logic," in *Dispositions*, 1978, pp. 27–54.
- [60] W. V. Quine, "Intensions revisited," *Midwest Studies in Philosophy*, vol. 2, no. 1, pp. 5–11, 1977.
- [61] S. March and G. Allen, "Toward a social ontology for conceptual modeling," in *11th Symposium on Research in Systems Analysis and Design*, Vancouver, Canada, 2012, pp. 57–62.
- [62] B. Berlin, D. E. Breedlove, and P. H. Raven, "General Principles of Classification and Nomenclature in Folk Biology," *American Anthropologist*, vol. 75, no. 1, pp. 214–242, 1973.
- [63] J. H. Langdon, "Background: Evolutionary Classification and Fossil Dating," in *Human Evolution: Bones, Cultures, and Genes*, Springer, 2023, pp. 31–49.

- [64] D. D. Kahneman, "The reviewing of object files: Object-specific integration of information," *Cognitive psychology*, vol. 24, no. 2, pp. 175–219, 1992.
- [65] G. Murphy, *The big book of concepts*. Cambridge, MA: MIT Press, 2004.
- [66] O. Eriksson and P. J. Agerfalk, "Speaking things into existence: ontological foundations of identity representation and management," *ISJ*, vol. 35, no. 1, pp. 1–30, 2021.
- [67] R. Clarke, A. Burton-Jones, and R. Weber, "On the Ontological Quality and Logical Quality of Conceptual-Modeling Grammars: The Need for a Dual Perspective," *Information Systems Research*, vol. 27, no. 2, pp. 365–382, 2016.
- [68] D. L. Parnas, "A technique for software module specification with examples," *Communications of the ACM*, vol. 15, no. 5, pp. 330–336, May 1972.
- [69] S. Kalyuga, "Knowledge elaboration: A cognitive load perspective," *Learning and Instruction*, vol. 19, no. 5, pp. 402–410, 2009.
- [70] M. A. Bunge, *Chasing reality: strife over realism*. University of Toronto Press, 2006.
- [71] J. L. Austin, J. O. Urmson, and M. Sbisà, *How to Do Things with Words*. in William James lectures. Clarendon Press, 1975.
- [72] Y. Wand, D. E. Monarchi, J. Parsons, and C. C. Woo, "Theoretical foundations for conceptual modelling in information systems development," *Decision Support Systems*, vol. 15, no. 4, pp. 285–304, 1995.
- [73] J. Peckham and F. Maryanski, "Semantic data models," *ACM Computing Surveys*, vol. 20, no. 3, pp. 153–189, 1988.
- [74] J. M. Smith and D. C. P. Smith, "Database abstractions: aggregation and generalization," *ACM Transactions on Database Systems*, vol. 2, no. 2, pp. 105–133, 1977.
- [75] P. Chen, "The entity-relationship model - toward a unified view of data," *ACM Transactions on Database Systems*, vol. 1, no. 1, pp. 9–36, 1976.
- [76] R. Lukyanenko and B. M. Samuel, "Are all Classes Created Equal? Increasing Precision of Conceptual Modeling Grammars," *ACM Transactions on Management Information Systems (TMIS)*, vol. 40, no. 2, pp. 1–25, Forthcoming 2017.
- [77] M. Jabbari, J. Recker, P. Green, and K. Werder, "How do Individuals Understand Multiple Conceptual Modeling Scripts?," *J AIS*, vol. 23, no. 4, pp. 1037–1070, 2022.
- [78] B. M. Samuel, L. Watkins, A. Ehle, and V. Khatri, "Customizing the Representation Capabilities of Process Models: Understanding the Effects of Perceived Modeling Impediments," *Software Engineering, IEEE Transactions on*, vol. 41, no. 1, pp. 19–39, 2015.
- [79] D. L. Moody, "The 'physics' of notations: toward a scientific basis for constructing visual notations in software engineering," *Software Engineering, IEEE Transactions on*, vol. 35, no. 6, pp. 756–779, 2009.
- [80] A. Burton-Jones, Y. Wand, and R. Weber, "Guidelines for Empirical Evaluations of Conceptual Modeling Grammars," *Journal of the Association for Information Systems*, vol. 10, no. 6, pp. 495–532, 2009.
- [81] R. Lukyanenko, J. Parsons, and B. M. Samuel, "Artifact Sampling: Using Multiple Information Technology Artifacts to Increase Research Rigor," in *HICSS 2018*, Big Island, Hawaii, 2018, pp. 1–12.