Tawny-SBOL: Using ontologies to design and constrain genetic circuits

Goksel Misirli [1][*] and Phillip Lord [2]

[1] School of Computing and Mathematics, Keele University, UK
[2] School of Computing Science, Newcastle University, UK

ABSTRACT

Synthetic biology is a data-driven engineering discipline and designing novel genetic circuits often requires utilizing existing information where possible. Semantic Web technologies, and particularly ontologies, are important to formalize knowledge for computational design processes and to facilitate data interoperability. The Synthetic Biology Open Language has already emerged as a data standard and is based on RDF/XML. This language is ideal to represent information as graphs in which nodes and edges are defined using multiple properties. Terms from ontologies and controlled vocabularies are used to indicate the meaning of these multiple properties. Semantic representation of these nodes and edges would simplify both the representation of information and the querying of underlying information. Here, we present Tawny-SBOL as a domain specific language and a framework to address these issues. Tawny-SBOL is a proof-of-concept project, based on the Tawny-OWL ontology library, to specify genetic circuit designs. Users can query and potentially constrain these designs. As a result, designs can be evolved based on predefined requirements. Due to the native Clojure language support, users can extend Tawny-SBOL programmatically and work interactively.

1 INTRODUCTION

The Synthetic Biology Open Language (SBOL) (Bartley *et al.*, 2015) has been developed to computationally exchange information about genetic circuits. Using this language, complex genetic circuits can be defined in terms of constituting simpler components such as DNA, proteins and small molecules. Designs can be hierarchical, formed of many sub designs, and the querying of the underlying information becomes challenging due to the complexity of relationships between different components. Each component may have additional properties such as the intended biological role, its molecular composition and so on.

These details are encoded using RDF and it can be difficult to construct SBOL documents manually. Although, there are discussions to adopt the Turtle format in the future, RDF/XML is currently adopted and the utilisation of existing Semantic Web tooling is particularly valuable. There are already ongoing developments to create SBOL APIs which are available in Java, C, Python and JavaScript languages. Although these APIs are necessary to create SBOL documents, detailed knowledge about the SBOL data model and how each SBOL entity is related to others is required. These APIs can be used by experienced programmers who are expert in using the programming language for their chosen API and these programmers usually follow the development of SBOL closely.

Ideally, biologists should use tools built upon these APIs. Clearly, interacting with different tools takes time and effort to learn. Simplified textual representation is another way of sketching genetic circuit designs and improving them later on. Moreover, decoupling the connection between APIs and complex data structures may further facilitate the development of useful tools.

ShortBOL (Pocock *et al.*, 2016) has particularly been developed as a shorthand language to produce complex SBOL documents more easily. It is a human-readable textual language and allows defining design components and their composition. The mechanism behind ShortBOL is *template expansion*, in which templates can be hierarchical and each template adds additional graph attributes until fully-serialized SBOL RDF graphs are created. For example, in SBOL to represent a promoter component, it needs to be declared as a `ComponentDefinition` with the `sbol:type` of `biopax:DnaRegion`, and the `sbol:role` of `SO:0000167` (The Sequence Ontology promoter term). In Shorthand, a promoter is already defined as a template, and the SBOL Shorthand compiler utilises this template to inject required RDF triples.

In this work, we present Tawny-SBOL, based on the promising development of ShortBOL, and utilise ontologies to provide the meaning of design entities through subsumption. This domain specific language (DSL) based ontological representation allows executing simple semantic queries that can be quite complex when represented as a graph query. Moreover, as opposed to creating static documents that can be exchanged between researchers, our aim is to provide an interactive design environment, where users can create semantic constraints and queries, and designs can evolve over time.

2 THE SBOL ONTOLOGY

Standardised SBOL terms to describe the SBOL data model already exist. However, these terms are part of a controlled vocabulary, which is embedded in the SBOL specification documents using free text. In order to utilise ontological representation of SBOL documents, we created the SBOL ontology using Tawny-OWL (Lord, 2013) programmatically (Figure 1). We defined classes for SBOL entities that are represented as RDF resources. Some of the SBOL entities are not serialised but act as interfaces to group others. In this work, super classes have been defined to represent these interface entities. Moreover, SBOL specific terms that are only referenced to uniquely identify features of SBOL entities have been represented as classes. These include classes to indicate `Access`, `Direction` and `Refinement` types in SBOL.

3 TAWNY-SBOL

Tawny-SBOL provides a simple DSL to create SBOL data. It is implemented using Clojure and therefore inherits the properties of

---

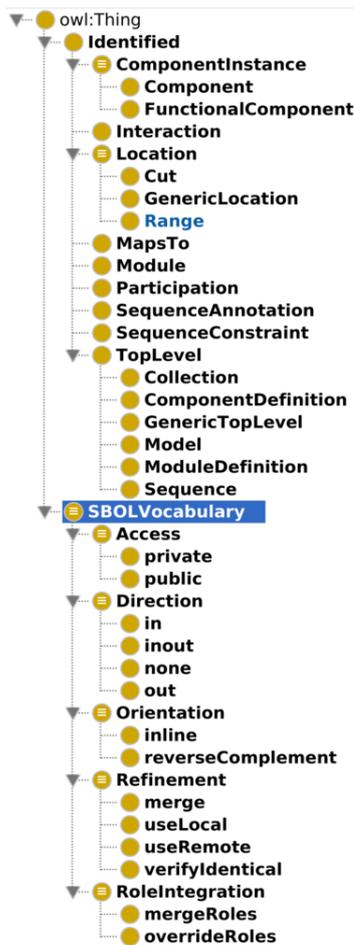[*] To whom correspondence should be addressed: g.misirli@keele.ac.uk

Fig. 1. OWL classes representing various SBOL entities.

```
(sboldocument "http://virtualparts.org/v2#" "v2")
...

(cds "lacI"
    {name "lacI",
     description "lacI coding sequence",
     designedBy "..."
     }
)
...

(design "lacI_expression prom1 1..40:+ rbs1 41..50
    :+ lacI 51..800:+ term1 801..850:+")

(design "prom1 lac1 1..10:+ lac2 30..40:+")

(save "lacI_expression")
```

Fig. 2. Partial information about the `lacI_expression` genetic circuit design, formed of a promoter, a RBS, a CDS and a terminator. The numeric range is used to provide the location information, + and − signs indicate the DNA strand. Only the representation of the CDS component is included here. The promoter component is further annotated with the use of two LacI binding sites using the second design command.

```
ComponentDefinition and (role some SO:0000167)

ComponentDefinition and ((component some lac1) or
    (component some lac1Parent))
```

Fig. 3. Using ontological queries to extract information about biological components and genetic circuits.

this language, such as using parentheses to create a block of SBOL data or to perform specific actions such as saving the results and so on. A specific Tawny-SBOL keyword is used to indicate the type of a simple biological component such as promoter, coding sequence, ribosome binding site, terminator and so on. The complex designs formed of simple components are represented using the `design` command. This command takes a parameter specified according to a grammar (Figure 2).

The resulting files not only include SBOL specific information, but also additional classes that facilitate executing semantic reasoners. These classes are injected by Tawny-SBOL. Currently, queries can be written using the OWL syntax and can directly be executed using the Tawny-OWL framework. For example, the first query in Figure 3 lists promoter resources, which are represented using SBOL's `ComponentDefinition` entity and has the role of `SO:0000167` term. In the second query, all the parents of the lac1 component is queried. The query in this case is an ontology class named `lac1Parent` and is used to recursively find all the uses of the child component in parent designs. In the future, we will further simplify the querying process by introducing SBOL specific commands in Tawny-SBOL.

4 CONCLUSION

Ontologies can be extremely useful to capture domain knowledge and to execute logical queries in synthetic biology (Misirli *et al.*, 2016). Tawny-SBOL has been developed to exploit these features for the ontological representation of genetic circuit designs. Here, we introduced the SBOL ontology together with a human readable textual DSL for SBOL. This DSL is based on Tawny-OWL and the Clojure programming language, providing users an extensible and interactive environment to add new design information when it is available, to query design information, and to create logical constrains. As the design-build-test cycle of engineering biological systems can take several iterations and can be achieved in long timescales, this constraint based approach will help to achieve desired systems and also to evolve designs in a controlled manner.

REFERENCES

Bartley, B., Beal, J., Clancy, K., Misirli, G., Roehner, N., Oberortner, E., Pocock, M., Bissell, M., Madsen, C., Nguyen, T., Zhang, Z., Gennari, J. H., Myers, C., Wipat, A., and Sauro, H. (2015). Synthetic Biology Open Language (SBOL) Version 2.0.0. *Journal of Integrative Bioinformatics*, **12**(2), 272.

Lord, P. (2013). The semantic web takes wing: Programming ontologies with tawny-owl. *arXiv preprint arXiv:1303.0213*.

Misirli, G., Hallinan, J., Pocock, M., Lord, P., McLaughlin, J. A., Sauro, H., and Wipat, A. (2016). Data integration and mining for synthetic biology design. *ACS Synthetic Biolology*, **5**(10), 1086–1097.

Pocock, M., Taylor, C., McLaughlin, J., Misirli, G., and Wipat, A. (2016). Shortbol: A shorthand for sbol. In *8th International Workshop on Bio-Design Automation*.