

Semantic representation of genetic circuit designs

Dr GÖKSEL MISIRLI

School of Computing and Mathematics, Keele University

&

ANGEL GONI-MORENO, JAMES MCLAUGHLIN, ANIL WIPAT AND PHILLIP LORD

Harmony 2018



Syntax vs semantics

```
<?xml version="1.0" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dcterms="http://purl.org/dc/terms/" xmlns:prov="http://www.w3.org/ns/prov#"
xmlns:sbol="http://sbols.org/v2#">
  <sbol:ComponentDefinition rdf:about="http://www.partsregistry.org/BBs_J23119">
    <sbol:type rdf:resource="http://www.biopax.org/release/biopax-level3.owl#DnaRegion"/>
    <sbol:role rdf:resource="http://identifiers.org/so/SO:0000167"/>
    <sbol:sequence rdf:resource="http://www.myapp.com/mydna/seq"/>
    <prov:wasDerivedFrom rdf:resource="http://parts.igem.org/Part:BBa_C0012"/>
    <dcterms:title>J23119</dcterms:title>
    <dcterms:description>Constitutive promoter</dcterms:description>
  </sbol:ComponentDefinition>
</rdf:RDF>
```

uri= **partsRegistry:BBa_J23119**
name= "J23119"
description= "Constitutive promoter"
type= **biopax:DnaRegion**
role= **so:0000167**

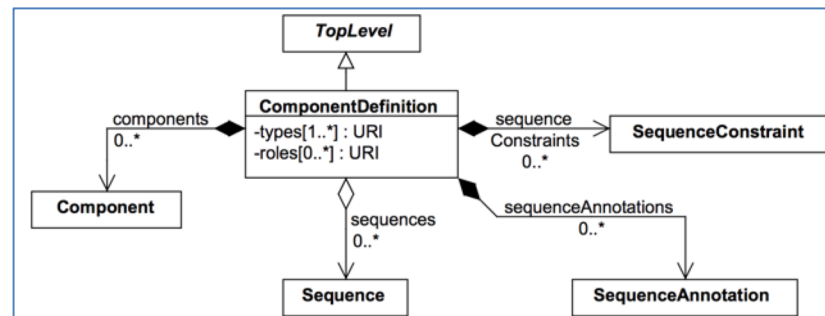
uri= partsRegistry:BBa_J23119_seq
elements= "ttga...ca"
encoding= IUPAC:DNA
sequence

SBOL semantics

- The SBOL **syntax** is described in free text in a PDF document
 - Inline UML diagrams are used to explain the text to humans
 - Partial serialisation examples are provided
- Validation rules are in an appendix section
- Querying of the SBOL data can be very complex
 - Representation of genetic components and designs is verbose
 - Hierarchical designs are especially difficult to query
- External ontologies and controlled vocabularies are utilised to provide the **meaning** of different terms
- How to formally define XML data that actually represent graphs?
 - XML Schema is not ideal for RDF/XML

7.7 ComponentDefinition

The **ComponentDefinition** class represents the structural entities with designed sequences used to represent any other entity that is part of a design.

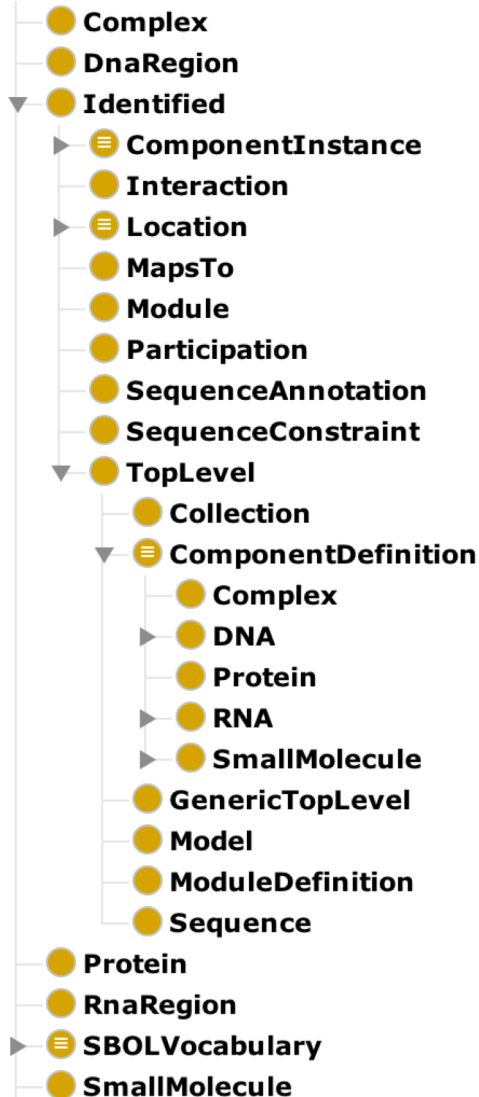
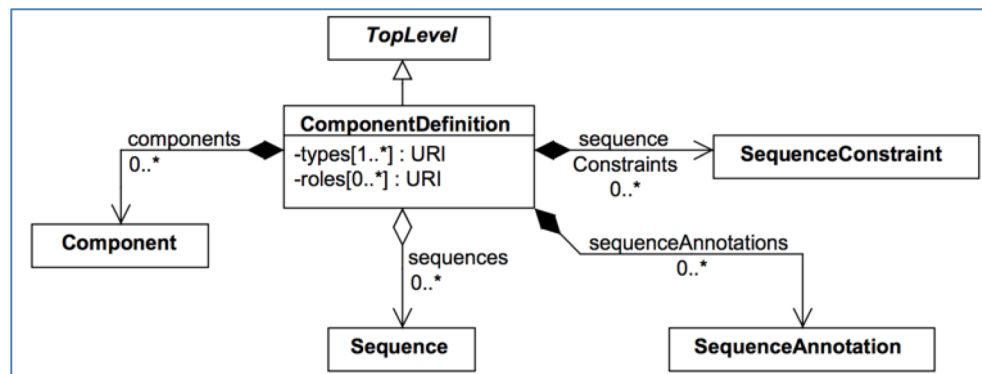


Ontological representation of the SBOL data model – SBOL-OWL

- How about defining the specification semantically and formally?
- SBOL-OWL → The SBOL Ontology
 - Used SBOL terms to create **classes**
 - **Additional classes** to represent the meaning of design specific components
 - Formally captured **relationships** between the classes
 - **Inverse** relationships were also incorporated

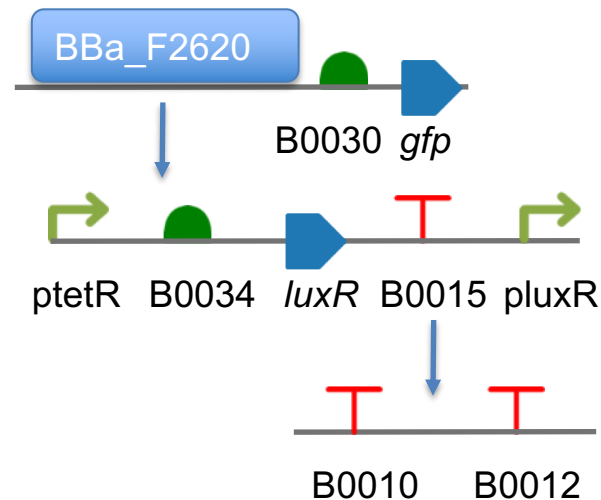
7.7 ComponentDefinition

The **ComponentDefinition** class represents the structural entities with designed sequences used to represent any other entity that is part of a design.

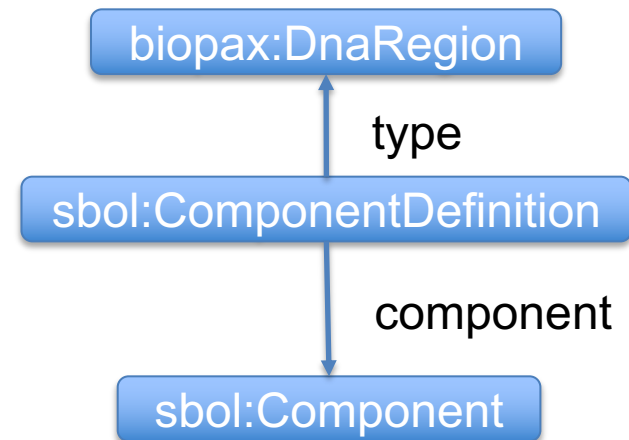


Logical design axioms

BBa_S03839



Return a list of **ComponentDefinitions** that are of **type DnaRegion** and that have **some Components**



ComponentDefinition
 and (type **some** DnaRegion)
 and (component **some** Component)

DNA class definition

DNA and (component **some** Component)

Query results

Instances (3 of 3)

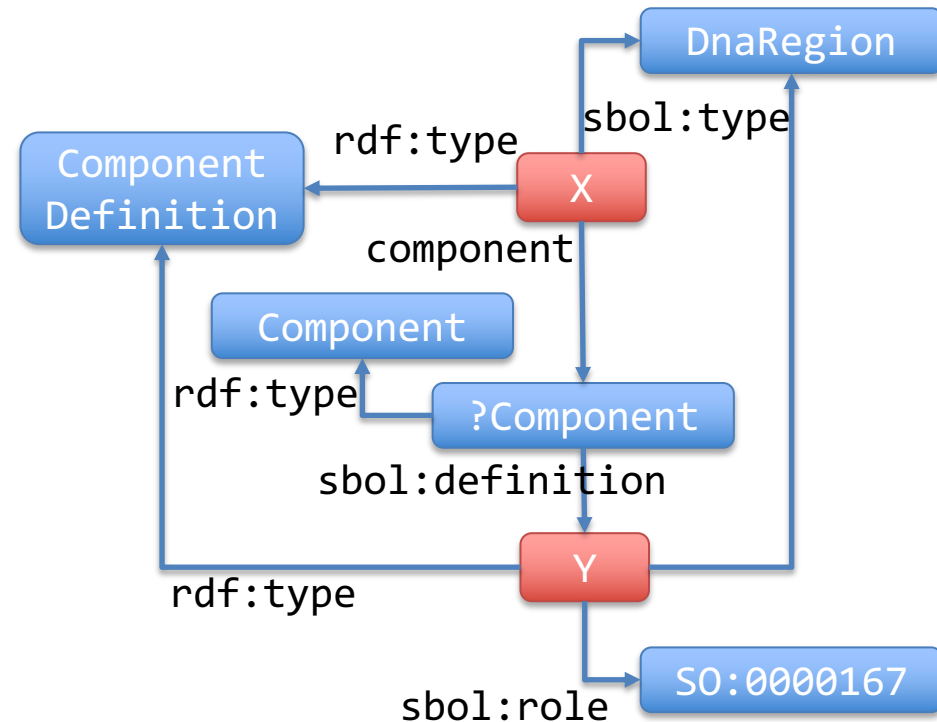
- ◆ B0015
- ◆ BBa_F2620
- ◆ BBa_S03839

Semantic querying of genetic circuit designs

Return **ComponentDefinitions** that have **some Components**, which are **Promoters** and are of type **DnaRegion**

```
PREFIX sbol: <http://sbols.org/v2> .  
PREFIX biopax: <...> .  
SELECT ?X  
WHERE {  
  ?X a ComponentDefinition ;  
    sbol:component ?Component ;  
    sbol:type biopax:DnaRegion .  
  ?Component a sbol:Component ;  
    sbol:definition ?SubComponentDefinition .  
  ?Y a ?ComponentDefinition ;  
    sbol:type biopax:DnaRegion ;  
    sbol:role so:S0:000167 .  
}
```

SPARQL query



Description: Promoter

Equivalent To +

● role **some** SO:0000167

SubClass Of +

● DNA

DL query

DNA **and** (component **some** (definition **some** Promoter))

Summary

- SBOL-OWL is ontological representation of the SBOL data model
- Provides machine accessible description of SBOL entities
- Enables computational validation of designs using verification rules
- Automated comparison of incremental SBOL specifications
- Annotating genetic circuit designs with rich semantics → Ability to use logical axioms for reasoning over design information

Thanks



Please let us know if you would like to contribute and be involved!

- International Conference on Biomedical Ontology 17, September 2017, Newcastle upon Tyne
 - Tawny-SBOL: Using ontologies to design and constrain genetic circuits
- COMBINE 2017, October, Italy
- UK Ontology Network, April 2018, Keele



Dr Phil Lord



Prof. Anil Wipat

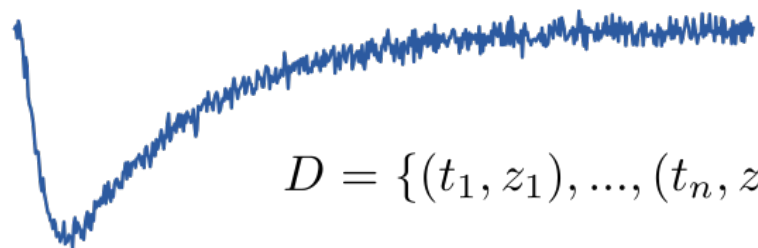


Dr Ángel Goñi Moreno

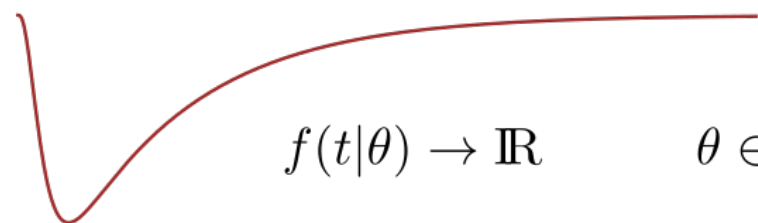


James McLaughlin

Probabilistic Inference on Noisy Time-Series



$$D = \{(t_1, z_1), \dots, (t_n, z_n)\}$$



$$f(t|\theta) \rightarrow \mathbb{R} \quad \theta \in \mathbb{R}^d$$

```
class MyModel(pints.Model):  
    def dimension(self):  
        return 20  
    def simulate(self, parameters, times):  
        # This is where you:  
        # - Write a simple Python method  
        # - Or call super fast C/C++ code  
        # - As long as you can  
        return simulated_values
```

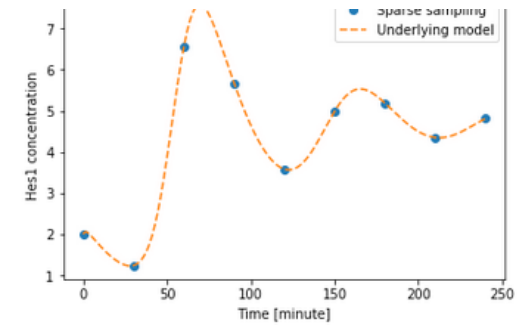
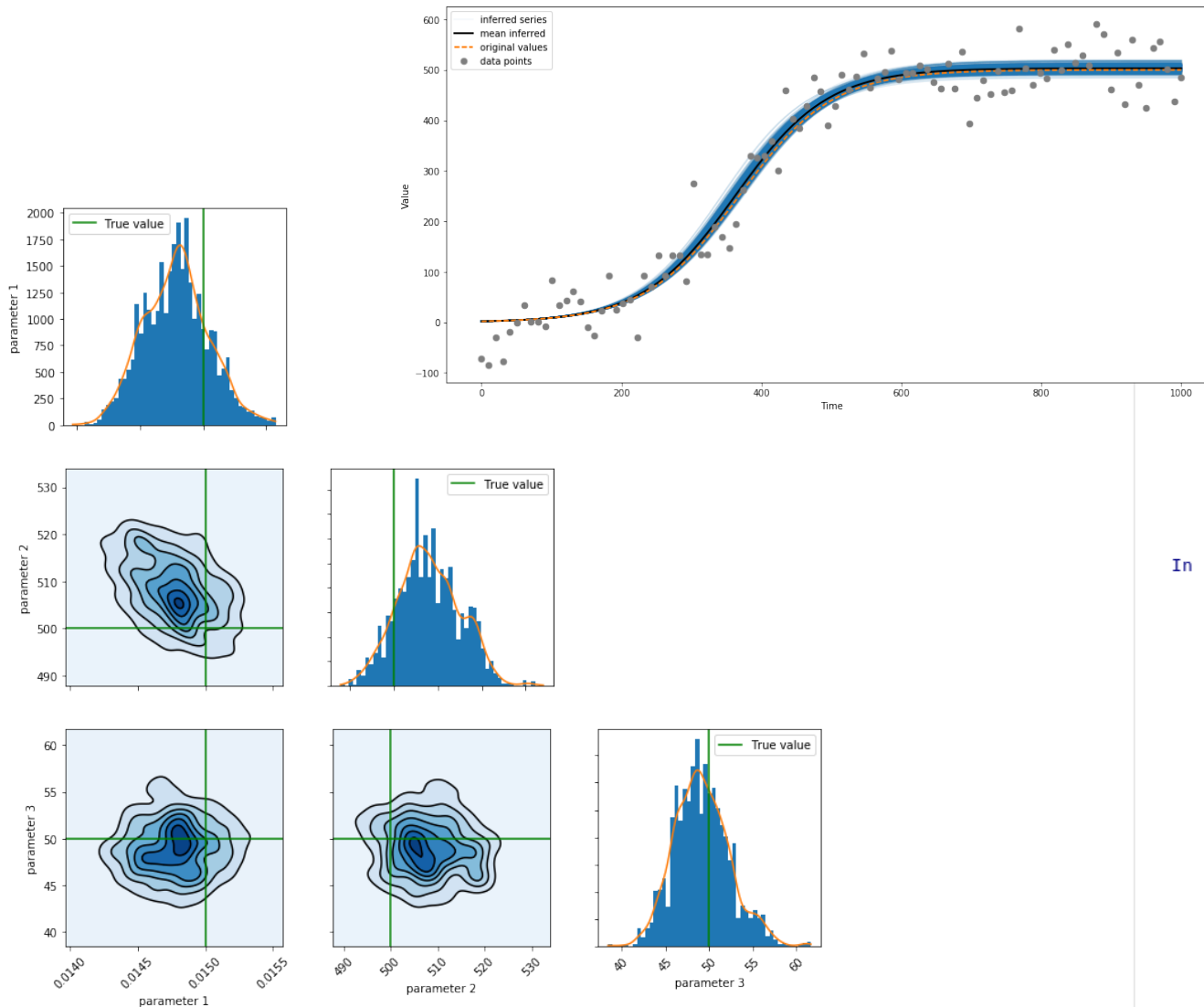
```
problem = SingleSeriesProblem(  
    model, times, measured_values)  
error_measure = SumOfSquares(problem)
```

```
initial_point = [1, 2, 3]  
optimisation = Optimisation(  
    error_measure, initial_point,  
    method=pints.XNES)
```

```
best_parameters = optimisation.run()
```



- Multiple optimisation / Bayesian inference methods
- Built-in plotting methods, lots of examples
- Tested, well documented, and free



In this set-up, we only see one of the state variables representing the Hes1 concentration. To see the full dynamics of the whole system (i.e. by inspecting all state variables). This can be done by inspecting the 'all_states' provided in this model.

```
In [3]: all_states = model.simulate_all_states(parameters, smooth_t:
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlabel('m')
ax.set_ylabel('p1')
ax.set_zlabel('p2')
plt.plot(all_states[:, 0], all_states[:, 1], all_states[:, 2])
plt.show()
```

