

Synthetic Biology Open Language (SBOL)

Version 1.0.9

Michal Galdzicki, Mandy L. Wilson, Cesar A. Rodriguez, Laura Adam, Aaron Adler, J. Christopher Anderson, Jacob Beal, Deepak Chandran, Douglas Densmore, Omri A. Drory, Drew Endy, John H. Gennari, Raik Grünberg, Timothy S. Ham, Allan Kuchinsky, Matthew W. Lux, Curtis Madsen, Goksel Misirli, Chris J. Myers, Josh Natarajan, Carlos Olguin, Jean Peccoud, Hector Plahar, Matthew R. Pocock, Nicholas Roehner, Trevor F. Smith, Guy-Bart Stan, Alan Villalobos, Anil Wipat, and Herbert M. Sauro

14 January 2012

1. Purpose

In this BioBricks Foundation Request for Comments (BBF RFC), we specify the Synthetic Biology Open Language (SBOL) Version 1.1.0 to enable the electronic exchange of information describing DNA components used in synthetic biology. We define:

1. the **vocabulary**, a set of preferred terms and
2. the **core data model**, a common computational representation.

2. Relation to other BBF RFCs

BBF RFC 84 REPLACES BBF RFC 31 and UPDATES BBF RFC 30 as it proposes a standard conforming to BBF RFC 30. BBF RFC 84 REPLACES BBF RFC 16.

3. Copyright Notice

Copyright (C) The BioBricks Foundation (2011). All Rights Reserved.

4. Acknowledgments

This document is the result of discussions between participants in the Synthetic Biology Open Language Workshops held in Blacksburg, Virginia on January 7-10, 2011, and San Diego, California on June 8, 2011. The ongoing work was finalized through email exchanges on the SBOL Developers mailing list through September 30, 2011. We are indebted to the following individuals for their contribution in these discussions: Laura Adam (Virginia Bioinformatics Institute), Aaron Adler (BBN Technologies), J. Christopher Anderson (Dept of Bioengineering, University of California Berkeley), Jacob Beal (BBN Technologies), Matthieu Bultelle (Bioengineering, Imperial College London), Kevin Clancy (Life Technologies), Kendall G. Clark (Clark & Parsia, LLC.), Douglas Densmore (Electrical and Computer Engineering, Boston University), Omri Drory (Genome Compiler), Drew Endy (BIOFAB and Dept of Bioengineering, Stanford University), John H. Gennari (Biomedical and Health Informatics, University of Washington), Raik Gruenberg (EMBL-CRG Systems Biology program, CRG), Jennifer Hallinan (School of Computing Science, Newcastle University), Timothy Ham (Joint BioEnergy Institute), Allan Kuchinsky (Agilent Technologies), Matthew W. Lux (Virginia Bioinformatics Institute), Curtis Madsen (School of Computing, University of Utah), Akshay Maheshwari (UCSD), Barry Moore (Human Genetics, University of Utah), Chris J. Myers (Electrical and Computer Engineering, University of Utah), Carlos Olguin (Autodesk Research), Jean Peccoud (Virginia Bioinformatics Institute), Hector Plahar (Joint BioEnergy Institute), Matthew Pocock (School of Computing Science, Newcastle University), Cesar A. Rodriguez (BIOFAB), Nicholas Roehner (Bioengineering, University of Utah), Vincent Rouilly (Biozentrum, University of Basel), Trevor F. Smith (Agilent Technologies), Guy-Bart Stan (Bioengineering, Centre for Synthetic Biology and Innovation, Imperial College London), Vinod Tek (Bioengineering, Imperial College London), Alan Villalobos (DNA 2.0, Inc.), Mandy Wilson (Virginia Bioinformatics Institute), Chris Winstead (Electrical and Computer Engineering Utah State University), Anil Wipat (School of Computing Science, Newcastle University), and Fusun Yaman Sirin (BBN Technologies).

5. Table of Contents

- 1. Purpose 1
- 2. Relation to other BBF RFCs 1
- 3. Copyright Notice 1
- 4. Acknowledgments 2
- 5. Table of Contents 3
- 6. Motivation 4
- 7. Introduction to SBOL 5
- 8. Description of SBOL..... 9
 - 8.1 Overview of SBOL..... 9
 - 8.2 Conventions..... 9
 - 8.2.1 SBOL versions and releases..... 10
 - 8.3. SBOL vocabulary 10
 - 8.3.1 Core 11
 - 8.4 Definition of the SBOL Core Data Model 12
 - 8.5 SBOL Core Model Classes..... 13
 - 8.5.1 DnaComponent:..... 13
 - 8.5.2 DnaSequence: 14
 - 8.5.3 SequenceAnnotation:..... 15
 - 8.5.4 Collection:..... 17
- 9. Examples 19
 - 9.1 Annotated Composite *DnaComponent* 19
 - 9.2 Multi-Tiered Annotated *DnaComponent* 21
 - 9.3 Partially Realized Design Template..... 21
 - 9.4 Collection 22
- 10. Serialization..... 22
- 11. Best Practices 22
- 12. Authors’ Contact Information 23
- 13. References 24
- 14. Appendix..... 25

6. Motivation

Synthetic biology is an engineering discipline where biological components, usually in the form of segments of DNA, are assembled to form devices and systems with more complex functions. A number of software tools have been developed to help synthetic biologists to design, optimize, validate and share these DNA systems, but the lack of a defined information standard for synthetic biology makes it extremely difficult to combine the appropriate applications into a refined systems process. To move the synthetic biology field towards best practices in engineering, synthetic biologists need software that can unambiguously interpret and exchange information about DNA components.

Computer Exchange Format

The lack of a standard exchange format means that synthetic biology information access and transfer is limited to manual efforts such as copy-and-paste and *ad hoc* scripts. Not only are these prohibitively lengthy approaches to data transfer, they can also be error-prone, either due to changes in the underlying architectures of the data sources or simple human error. Establishing a standard exchange format would not only save time, but would also help reduce the errors of manual transfer.

A standard exchange format would also provide a greater range of tools available to synthetic biologists. Although there is a wide variety of software tools out there, the difficulty inherent to designing interfaces between them sometimes makes it more effective for software developers to write their own applications rather than take advantage of something already out there; a standard exchange format would alleviate their need to develop interfaces or duplicate software, which in turn would free them up to develop new tools. Furthermore, if a standard format enabled programmatic access to public information resources (Galdzicki 2011), such as the Registry of Standard Biological Parts (<http://partsregistry.org>) and the BIOFAB Electronic Datasheets (<http://biofab.org/data>), software developers would be able to take advantage of these repositories directly within their applications. For example, CAD and modeling tools, such as TinkerCell (<http://tinkercell.com/>) and iBioSim (<http://www.async.ece.utah.edu/iBioSim/>), would be able to retrieve components for new designs. A gene network design created by tools such as the Proto Biocompiler (Beal 2011) could be further refined by Eugene into collections of physical implementations (Bilitchenko 2011).

In addition to improving the ability to share data across applications, a standard format would make it easier for synthetic biologists to exchange data with their collaborators at other sites. For example, synthetic biologist researchers could use software such as GD-ICE (<http://code.google.com/p/gd-ice>) and Clotho (<http://clothocad.org>) to integrate their own data from local laboratory repositories with their collaborators' designs and publicly available data. A synthetic biologist who designed new DNA constructs with a software tool such as GenoCAD (<http://www.genocad.org>) could send them to a collaborator who would then review and edit them using Gene Designer (<https://www.dna20.com/tools/genedesigner.php>).

The definition of a standard for electronic information exchange would also help the refinement of standards for the DNA components themselves, through an iterative

process of feedback to synthetic biology research groups concerned with standardization.

In summary, a standard exchange format would encourage reuse of existing DNA components; it would reduce error caused by manual or ad hoc data exchange; it would aid in collaboration between researchers; and it would save time which could then be used for advancing research and the development of new software tools.

Electronic exchange of synthetic biology information in a common format and using a common vocabulary will encourage the creation of interoperable software to support the information needs of synthetic biologists. Software developers will be able to write fewer data converters and offer access to a larger number of data sources. Finally, compatibility with Semantic Web information technology will serve as leverage for the software developed by this broad community, as it will facilitate reuse of previously generated knowledge across independent research efforts.

To establish such an information standard, we have proposed the Synthetic Biology Open Language (SBOL) as a launching point for a community development effort. As software tools are adapt to progress in the synthetic biology field, we expect SBOL to evolve to meet the needs of synthetic biology researchers and engineers.

7. Introduction to SBOL

The first version of the Synthetic Biology Open Language (SBOL) is defined to meet the information exchange needs of synthetic biologists. SBOL is composed of:

- The **core data model**, to structure the information used to describe DNA designs,
- and the **vocabulary**, which defines the terminology used in the data model.

To encourage expansion of this standard's capabilities, the guiding principle is openness. Therefore SBOL allows and expects extensions to version 1.0.0, proposed by the community. This collaboration in defining the common information exchange framework is driven by the community of researchers participating in the Synthetic Biology Data Exchange Group (<http://sbolstandard.org/>).

The goal of this specification is to define the terminology and relationships used to describe DNA designs. In order to provide a shared understanding between engineers seeking to exchange DNA designs, SBOL provides a common definition of the concepts needed. As much as possible, we attempt to make explicit the meaning of all terminology and data structures.

Scope

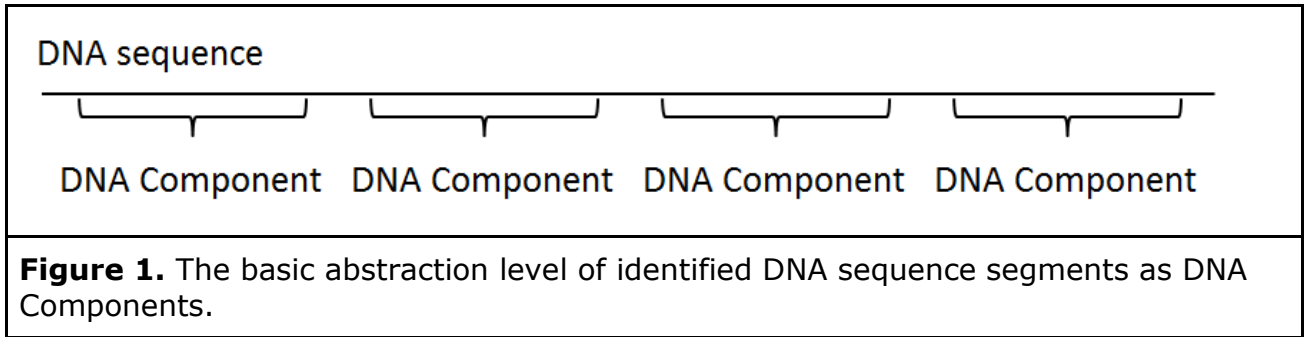
Version 1.0.0 of the SBOL core data model is limited to the description of discrete segments of DNA, called DNA components. To remove ambiguity when specifying the design of synthetic DNA, the information about DNA components is structured. DNA components described using the SBOL core data model may have an associated DNA sequence, or may be left as abstract descriptions. This flexibility allows for the description of DNA component designs which have not yet been realized, as well as those which are specific implementations of that design.

This version does not, however, provide a mechanism to represent the biological complexity of complete cellular systems beyond the DNA level. Additional biological knowledge needed to engineer aspects of complete biological systems will be modeled by future SBOL extensions. Furthermore, extensions of SBOL may add the ability to describe DNA components before and after a process, such as assembly, evolution, or implementation of a design *in silico*. Existing tools, such as GenoCAD, Eugene, and TASBE already offer solutions for bridging the “before and after”, so they can provide a basis for future specifications.

Abstraction Level

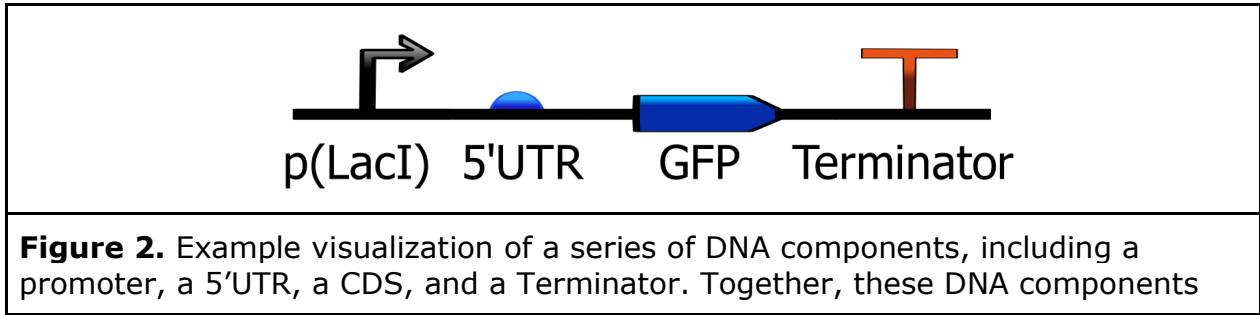
Within SBOL, we consider DNA regions as elements of design for DNA circuits (Savageau 2001), analogous to electrical circuits (Kaern 2003). This conceptualization of DNA segments as an element of design is a level of abstraction used to form the basis of engineering synthetic biological systems (Endy 2005). This level of abstraction (Figure 1) has been shown to be useful in the practice of forward engineering of biological systems (Nandagopal 2011). Therefore, we define these elements as DNA Components (Figure 1) in the SBOL vocabulary, and represent them as computational objects in the SBOL core data model.

DNA Components form the basic objects used in design, assembly, testing, and analysis. For example, DNA components can be hypothesized to have a biological function, deemed necessary for DNA assembly processes, or serve the synthetic biologist as landmarks in analysis.



Examples of Simple DNA Components

An example of the design of an expression cassette is shown in Figure 2; it illustrates DNA components along a DNA sequence. The symbols used represent their role in gene expression.



constitute the design of the expression cassette which expresses GFP. The iconography used represents the types of these DNA components (see SBOL:Visual for more on this extension <http://www.sbolstandard.org/initiatives/sbol-visual>). Individual icons are labeled with an informal name for the specified DNA component.

An example DNA construct which fulfills the design specified in Figure 2 is the BioBrick™ Part BBa_J04430 (<http://partsregistry.org>) (Figure 3). This example illustrates the representation of a DNA construct as a DNA component with annotations.

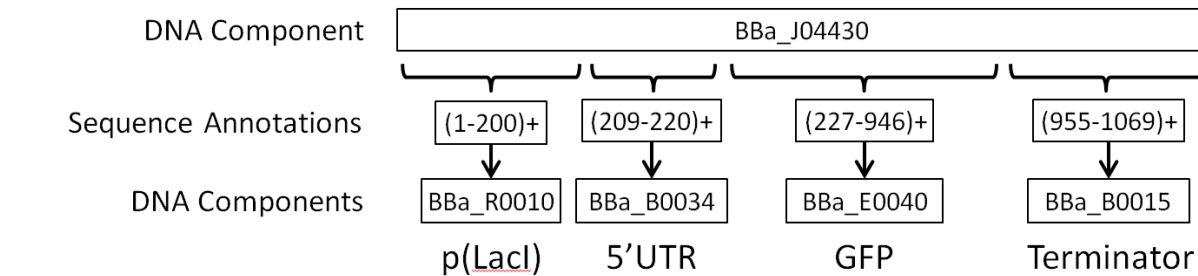
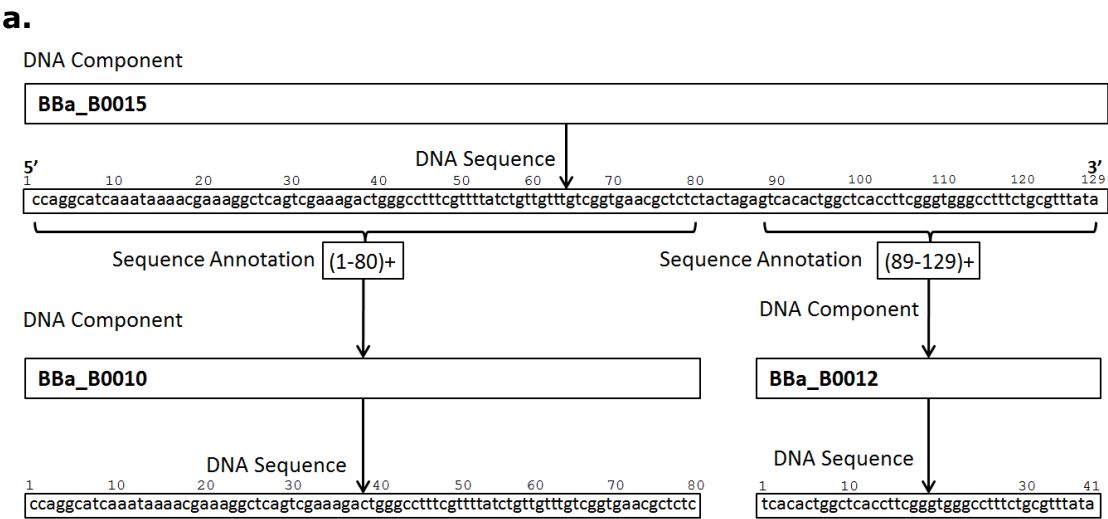


Figure 3. A diagram of BioBrick™ BBa_J04430 represented by SBOL objects. Sequence annotations of BBa_J04430 are used to describe the location of DNA components that are found within its sequence. These annotations are DNA components which correspond to the design specified in Figure 2.

Each DNA component can be further described with additional information (Figure 4).



DNA Component

display ID:	BBa_B0015
name:	B0015
description:	double terminator (B0010-B0012)
type:	http://identifiers.org/obo.so/SO:0000167
uri:	http://partsregistry.org/Part:BBa_B0015

b.

Figure 4. BBa_B0015, a sub-component of BBa_J04430, **(a)** with DNA sequence and sequence annotations. **(b)** To describe BBa_B0015 in more detail a set of fields for a human readable ID, name, and text description is defined. Structured information fields will enable basic retrieval capabilities ie. *type* using the Sequence Ontology (Eilbeck 2005, Mungall 2010); and *uri* as a unique identifier.

8. Description of SBOL

SBOL version 1.0, focuses on the representation of DNA components as the SBOL:Core:model. SBOL:Vocabulary defines the key terms used in the core model.

8.1 Overview of SBOL

SBOL Vocabulary

In version 1.0.0, SBOL:Vocabulary defines the core concepts used by SBOL in the structured description of synthetic DNA designs. SBOL:Core terms are *DnaComponent*, *DnaSequence*, and *SequenceAnnotation*. To provide user-defined groupings of *DnaComponents*, SBOL:Core also defines a *Collection*. Additional term definitions such as those needed to classify *DnaComponents* by type can be obtained from the Sequence Ontology (Eilbeck 2005, Mungall 2010). For example, a promoter region, coding sequence, and transcriptional terminator are all defined by the Sequence Ontology. Currently, terminology outside of the scope of the Sequence Ontology is being considered either for submission as new term requests to its curators, or for possible implementation as SBOL:Vocabulary extensions; examples of terminology under consideration include SBOL:Gene Expression and SBOL:DNA Construction (see the Appendix for a list of these provisional terms and their definitions).

SBOL Core Data Model

To enable electronic exchange of information, SBOL:Core:model defines the data model describing the DNA sequence and groupings of components used for biological engineering. The model specifies the object and data properties associated with instances of the concepts defined by SBOL:Core terms. SBOL:Core:model represents a consensus of the minimal information needed to describe DNA sequences used in synthetic biological designs.

8.2 Conventions

The project to define SBOL is comprised of constitutive parts. The convention for naming them takes the form `SBOL:Specification Part Name[:subspecification]`; for example, SBOL:Core:model is used to denote the core data model part of the SBOL as it uses the portion of SBOL:Vocabulary called SBOL:Core and specifies the data model to structure it.

SBOL:Vocabulary terms are italicized in the remainder of this document to distinguish them. SBOL:Core:model Class names are written in upper *CamelCase*, starting with an upper case letter. For example, *DnaComponent* is the class or type of object that represents a 'DNA Component'. Field or property names defined within classes start with lower cases letters and follow lower *camelCase*. For example, *bioStart* is a field that specifies the position of the first base pair of a *SequenceAnnotation*.

Instances of SBOL:Core:model classes are written as abbreviations.

Abbreviation key:

DC_∅ – a *DnaComponent* w/o type, w/o sequence
DC_t – a *DnaComponent* w/ type
DC_s – a *DnaComponent* w/ sequence
DC_{st} – a *DnaComponent* w/ sequence and type
SA_{posN} – a *SequenceAnnotation* w/ position coordinates [N-ordinal notation only]
SA_{rdN} – a *SequenceAnnotation* w/ relative position (precedes)
SA_∅ – a *SequenceAnnotation* w/ position unspecified
Col – a *Collection*

The terms "MUST", "REQUIRED", "SHALL", "MUST NOT", "SHALL NOT", "SHOULD", "RECOMMENDED", "SHOULD NOT", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BBF RFC 0].

The following URI namespaces are defined as prefix: "namespace":

sbol: "http://sbols.org/v1#"
xsd: "http://www.w3.org/2001/XMLSchema#"

8.2.1 SBOL versions and releases

SBOL follows a version strategy inspired by Semantic Versioning (<http://semver.org>) and simplified for SBOL as a specification of a standard in contrast to a public API.

SBOL version numbers MUST take the form X.Y.Z where X, Y, and Z are integers. X is the major version, Y is the minor version, and Z is a patch version. Each element MUST increase numerically. For instance: 1.9.0 < 1.10.0 < 1.11.0.

Prior to version 1.0.0 (specified by this document) a less formal version system was in place. After the ratification of this document as SBOL version 1.0.0, the following strategy MUST be followed.

- Major versions (X) correspond to releases of SBOL, the submission of the specification document to the BioBrick Foundation as a Request For Comment (BBF RFC).
- Minor versions (Y) correspond to revisions to the specification as approved by the SBOL Forum, as defined by the SBOL governance document (<http://www.sbolstandard.org/sbol-governance>), and confirmed by the voting process.
- Patch versions (Z) correspond to draft revisions made by the SBOL Editors during the specification process.

The SBOL definitions that follow this section conform to these conventions.

8.3. SBOL vocabulary

The SBOL:Vocabulary defines terms used in SBOL. Below we define terms for the Core. Additional terms, such as those related to gene expression and DNA construction, are being considered as extensions in collaboration with the Sequence Ontology project.

8.3.1 Core

The SBOL:Core terms are defined to be used as concepts common to descriptions of DNA sequences in synthetic biology.

DNA Component	<p>A DNA component represents a segment of DNA that serves to abstract the DNA sequence as an individual object, which can then be manipulated, combined, and reused in engineering new biological systems.</p> <p>SBOL name: <i>DnaComponent</i> SBOL URI: http://sbols.org/v1#DnaComponent</p> <p>Nominal definition: [Standard] Biological Part (Endy 2005, Shetty 2008), BioBrick™ Part, DNA Part</p>
DNA Sequence	<p>The DNA sequence is a contiguous sequence of nucleotides. The sequence is a fundamental information object for synthetic biology and is needed to reuse components, replicate synthetic biology work, and to assemble new synthetic biological systems. Therefore, both experimental work and theoretical sequence composition research depend heavily on the exact base pair sequence specification associated with <i>DnaComponents</i>.</p> <p>SBOL name: <i>DnaSequence</i> SBOL URI: http://sbols.org/v1#DnaSequence</p> <p>Nominal definition: The deoxyribonucleic acid sequence, primary structure of DNA</p>
Sequence Annotation	<p>The sequence annotation is the position and direction of a notable sub-sequence found within the <i>DnaComponent</i> being described. Annotations provide the link which describes the DNA sequence of a component in terms of other components, <i>subComponents</i>. When a DNA component is abstract, <i>SequenceAnnotations</i> specify relative positions between <i>subComponents</i>.</p> <p>SBOL name: <i>SequenceAnnotation</i> SBOL URI: http://sbols.org/v1#SequenceAnnotation</p> <p>Nominal definition: position, location, order [of <i>subComponents</i>]</p>
Collection	<p>A collection is an organizational container, a group of <i>DnaComponents</i>. For example, a set of restriction enzyme recognition sites, such as the components commonly used for BBF RFC 10 BioBricks™ could be grouped. A <i>Collection</i> might contain DNA components used in a specific project, lab, or custom grouping specified by the user.</p> <p>SBOL name: <i>Collection</i> SBOL URI: http://sbols.org/v1#Collection</p> <p>Nominal definition: Set, Collection, Bag of Parts, Library</p>

8.4 Definition of the SBOL Core Data Model

This section defines the structure of SBOL:Core:model. In Figure 5, the UML class diagram notation is used to describe the SBOL:Core:model classes, their properties, and the main associations between classes. Section 9 provides complete examples encoded in SBOL. There are four classes in the data model, *DnaComponent*, *DnaSequence*, *SequenceAnnotation*, and *Collection*, which correspond to the four concepts needed to unambiguously describe the DNA design of synthetic biological systems. Each instance of a *DnaComponent* class refers to an actual or planned DNA component. When using SBOL to describe information about DNA components, an instance of the *DnaComponent* class MUST be created. The *DnaComponent* instance MAY specify an associated *DnaSequence* instance that it pertains to, and MAY be described using *SequenceAnnotation* instances to specify the position of subcomponents (*DnaComponent*). *Collection* instances MAY have associated *DnaComponent* instances. These concepts are illustrated in Figure 5.

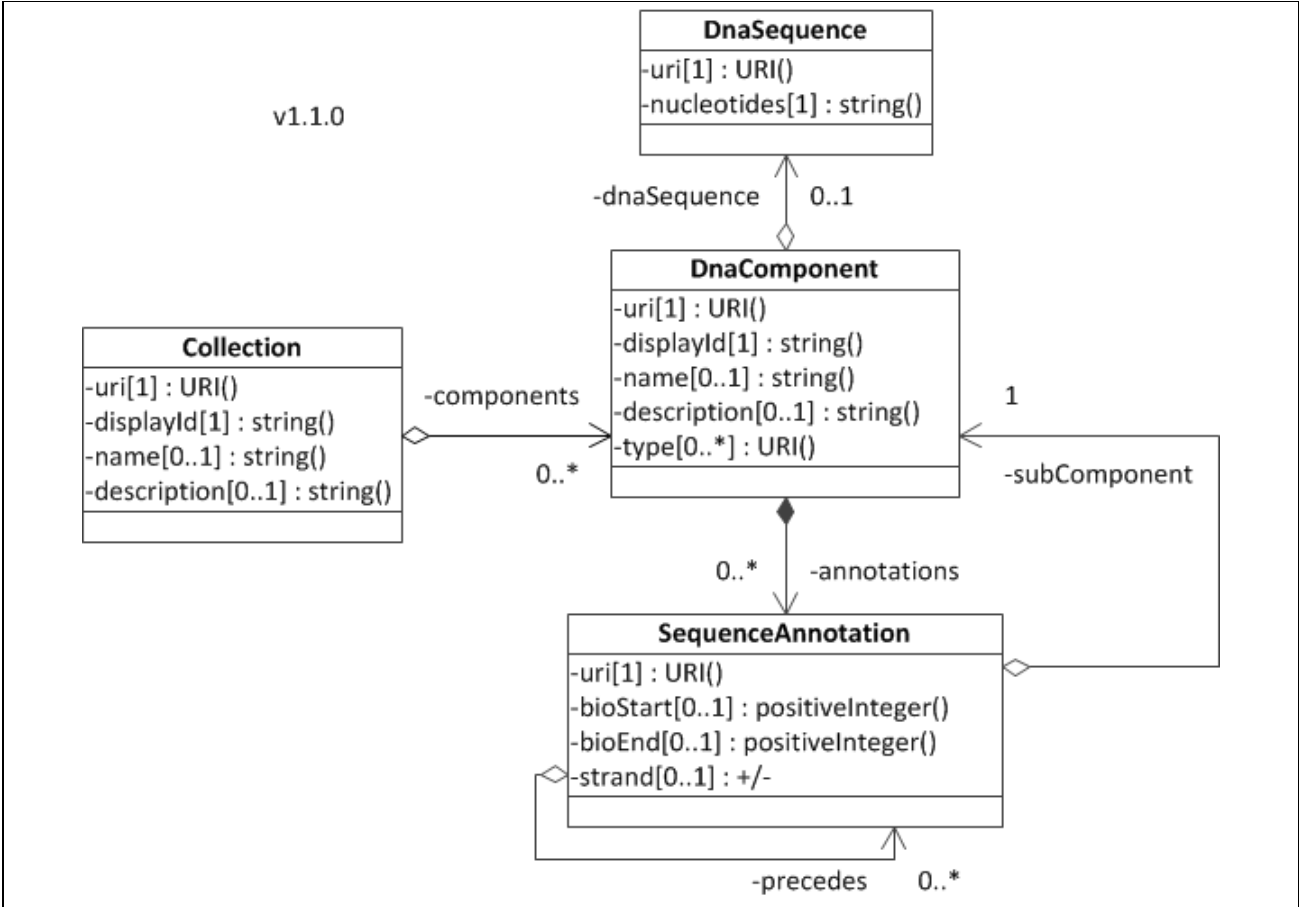


Figure 5. The SBOL core data model is specified using a UML 2.0 diagram (OMG 2005). Classes (rectangles) are named at the top and connected by associations (arrows). Each association is labeled with its role name, and has a range type and a plurality, such as “exactly zero or one *dnaSequence*” [0..1] or “one and only one *subComponent*” [1]. These can be interpreted as Sets of objects which are instances of the Class specified. An arrowhead indicates that the association can be traversed in that direction. Diamonds classify the association; open-faced diamonds are shared aggregation, meaning the object at the end of the arrow can exist independently of the source object, and filled diamonds indicate composite aggregation, or a part-whole relationship, which means that a part instance must be included in at most one whole and cannot exist independently. Data properties for objects of each class are listed in a separate compartment below, with the cardinality and corresponding data types specified.

8.5 SBOL Core Model Classes

We define each class individually and specify requirements for their intended use in the SBOL:Core data model.

8.5.1 DnaComponent:

Objects representing a DNA component MUST be instances of the class *sbol:DnaComponent*, defined as:

Class: <i>DnaComponent</i>
Context: Instances of the <i>DnaComponent</i> class represent segments of DNA as defined by <i>sbol:DnaComponent</i> . The <i>DnaComponent</i> 's DNA sequence can be annotated using <i>SequenceAnnotation</i> instances, positionally defined descriptors of the sequence which specify additional <i>DnaComponent</i> instances as <i>subComponents</i> . A <i>DnaComponent</i> MAY specify one <i>DnaSequence</i> instance it abstracts. <i>DnaComponent</i> instances MAY be grouped into <i>Collections</i> .

A *DnaComponent* instance MUST have the following REQUIRED data properties.

Data properties:
<i>uri</i> : (required) One and only one field of type URI (IETF RFC 2396). This property uniquely identifies the instance, and is intended to be used whenever a reference to the instance is needed, such as when referring to a <i>DnaComponent</i> stored on a server from another location. This URI MAY be a fully qualified URI (e.g. http://sbols.org/data#P0123), which then MUST remain constant forever, or this URI MAY consist of a relative identifier only (e.g. #P0123), which then MUST be unique within the enclosing context (e.g. <i>Collection</i> , an XML document). This form MAY be used for the informal exchange of free-floating temporary SBOL instances. For any other purpose it is RECOMMENDED that the relative identifier can be resolved against the enclosing context (e.g. the <i>uri</i> of a <i>Collection</i> or the fixed address of an XML document) into a fully qualified URI which then, again, MUST remain constant forever.
<i>displayId</i> : (required) One and only one field of type <i>xsd:string</i> starting with a letter or underscore followed by only alphanumeric and underscore characters. The <i>displayId</i> is a human readable identifier for display to users. For example, users could use this identifier in combination with the namespace of the source as an unambiguous reference to the DNA construct.

OPTIONALLY a *DnaComponent* instance MAY have the following RECOMMENDED data and object properties.

Object properties:

dnaSequence: (optional)

Zero or one value of type *DnaSequence*. This property specifies the DNA sequence which this *DnaComponent* object represents. See also: *DnaSequence*.

annotations: (optional)

Zero or more values of type *SequenceAnnotation*. This property links to *SequenceAnnotation* instances, each of which specifies the position and direction of a *DnaComponent* that describes a *subComponent* of this DNA component.

Data properties:

name: (optional)

Zero or one value of type *xsd:string*. The *name* of the DNA component is a human-readable string providing the most recognizable identifier used to refer to this *DnaComponent*. It often confers meaning of what the component is in biological contexts to a human user. A *name* may be ambiguous, in that multiple, distinct *DnaComponents* may share the same *name*. For example, acronyms are sometimes used (eg. pLac-O1) which may have more than one instantiation in terms of exact DNA sequence composition. As these names are intended for human consumption, they SHOULD be kept short and meaningful, as may be done by using an acronym, or re-using names that have commonly been used in literature.

description: (optional)

Zero or one value of type *xsd:string*. The *description* is a free-text field that contains text such as a title or longer free-text-based description for users. This text is used to clarify what the *DnaComponent* is to potential users (eg. engineered Lac promoter, repressible by LacI). The description could be lengthy, so it is the responsibility of the user application to format and allow for arbitrary length.

type: (optional)

Zero or more values of type URI (IETF RFC 2396) referencing the Sequence Ontology (see Appendix for commonly used terms). These provide a defined terminology of types of *DnaComponents*. This vocabulary may be extended, (use "Request a Term" <http://sequenceontology.org>).

8.5.2 DnaSequence:

Objects representing a DNA Sequence MUST be instances of the class *sbol:DnaSequence*, defined as:

Class: *DnaSequence*

Context:

Instances of the *DnaSequence* class contain the actual DNA sequence string. This specifies the sequence of nucleotides that comprise the *DnaComponent* being described.

For SBOL *DnaSequence*, the base pairs MUST be represented by a sequence of lowercase characters corresponding to the 5' to 3' order of nucleotides in the DNA

segment described, eg. "actg". The string value MUST conform to the restrictions listed below:

- a. The DNA sequence will use the IUPAC ambiguity recommendation. (See http://www.genomatix.de/online_help/help/sequence_formats.html)
- b. Blank lines, spaces, or other symbols must not be included in the sequence text.
- c. The sequence text must be in ASCII or UTF-8 encoding. For the alphabets used, the two are identical.

A *DnaSequence* instance MUST have the following REQUIRED data properties.

Data properties:

- uri*: (required)
One and only one field of type URI (IETF RFC 2396). This property uniquely identifies the instance, and is intended to be used whenever a reference to the instance is needed. See also *DnaComponent.uri*
- nucleotides*: (required)
One and only one value of type *xsd:string*. See requirements for value of string in the class definition.

8.5.3 SequenceAnnotation:

Objects representing a Sequence Annotation MUST be instances of the class *sbol:SequenceAnnotation*, defined as:

Class: *SequenceAnnotation*

Context:

Individual instances of the *SequenceAnnotation* class provide the position and direction of *subComponents* (*DnaComponents*) that are found within the annotated *DnaComponent*. Location CAN be specified by the *bioStart* and *bioEnd* positions of the *subComponent*, along with the DNA sequence. Alternatively, the partial order of *SequenceAnnotations* along a *DnaComponent* can be specified by indicating the *precedes* relationship to other *SequenceAnnotations*. As a convention, numerical coordinates in this class use position 1 (not 0) to indicate the initial base pair of a DNA sequence. This convention is followed by the broader Molecular Biology community, especially in the relevant literature. The direction of the *subComponent* is specified by the *strand* [+/-]. Sequences used are assumed, by convention, to be specified 5' to 3', therefore the + *strand* is 5' to 3' and the - *strand* is 3' to 5'.

A *SequenceAnnotation* instance MUST specify a *subComponent* of type *DnaComponent*. *SequenceAnnotations* MUST belong to exactly 1 *DnaComponent*.

A *SequenceAnnotation* instance MUST have the following REQUIRED object properties.

Object properties:
<i>subComponent</i> : (required) One and only one value of type <i>DnaComponent</i> . This property specifies the DNA sequence feature being annotated on the <i>DnaComponent</i> 's sequence. The <i>DnaComponent</i> value serves to indicate information about the subsequence at the position specified by the <i>SequenceAnnotation</i> 's location data properties or the relative position object property.
Data properties:
<i>uri</i> : (required) One and only one field of type URI (IETF RFC 2396). This property uniquely identifies the instance, and is intended to be used whenever a reference to the instance is needed. see also <i>DnaComponent.uri</i>

A *SequenceAnnotation* instance MAY have one of the following Location Data or Relative Position Object property groups.

Location Data Group

Data properties:
<i>bioStart</i> : (optional) Zero or one value of type <i>xsd:positiveInteger</i> . Positive integer coordinate of the position of the first base of the <i>subComponent</i> on the <i>DnaComponent</i> . <i>bioStart</i> coordinate is relative to the parent sequence.
<i>bioEnd</i> : (optional) Zero or one value of type <i>xsd:positiveInteger</i> . Positive integer coordinate of the position of the last base of the <i>subComponent</i> on the <i>DnaComponent</i> . <i>bioEnd</i> coordinate is relative to the parent sequence.
<i>strand</i> : (optional) Zero or one value of type <i>xsd:string</i> . Strand orientation + or - of the <i>subComponent</i> relative to the DNA sequence of the <i>DnaComponent</i> being annotated. <i>DnaSequence</i> is by convention assumed 5' to 3', therefore the "+" strand is 5' to 3' and the "-" strand is 3' to 5'.

Relative Position Object Group

Object property:
<i>precedes</i> : (optional) Zero or more values of type <i>SequenceAnnotation</i> . The <i>precedes</i> relation specifies the relative order of <i>SequenceAnnotations</i> for a given <i>DnaComponent</i> . It is a constraint on the order of <i>subComponents</i> when there is not enough information to specify exact positions. <i>Precedes</i> indicates the intended location by specifying that a <i>SequenceAnnotation</i> is to come before another when <i>DnaSequence</i> information becomes available. For example, you may want to say the promoter <i>SequenceAnnotation</i> <i>precedes</i> the CDS <i>SequenceAnnotation</i> , which <i>precedes</i> the

terminator *SequenceAnnotation*. This ordering gives us the position, relative to other *SequenceAnnotations* (which can have a location or a relative position (using *precedes*)). During a validation process, the set of *precedes* relations on the *SequenceAnnotation* are required to be linearized to a sequence.

Well-formed constraint: Location Data

The Location Data fields of *SequenceAnnotation* are *bioStart*, *bioEnd*. They must either both be present or absent. It is a well-formedness violation for one to be present while the other is absent.

Well-formed constraint: Relative Position

The Relative Position field of *SequenceAnnotation* is *precedes*. A relative position is valid if one of the following is true: *precedes* with a value of type *SequenceAnnotation*; or *precedes* with a value of `null`, indicating a terminal *SequenceAnnotation*.

Logical consistency of Location Data and Relative Position

Given *sa1:SequenceAnnotation* and *sa2:SequenceAnnotation*, where *sa1 precedes sa2*, they are logically consistent if:

- absent data: *sa1.bioEnd* or *sa2.bioStart* are not provided; or
- consistent data: both *sa1.bioEnd* and *sa2.bioStart* are provided and *sa1.bioEnd* is strictly greater than *sa2.bioStart*

Logical consistency of the subComponent’s DnaSequence value

If present, the *DnaSequence* value of the *DnaComponent* referenced by *subComponent* in a *SequenceAnnotation* with Location Data MUST be the exact sequence found in the interval specified by the Location Data.

8.5.4 Collection:

Instances representing a Collection MUST be instances of the class *sbol:Collection*, defined as:

Class: <i>Collection</i>
Context: Individual instances of the <i>Collection</i> class represent an organizational container which helps users and developers conceptualize a set of <i>DnaComponents</i> as a group. Any combination of these instances CAN be added to a <i>Collection</i> instance, annotated with a <i>displayID</i> , <i>name</i> , and <i>description</i> and be published or exchanged directly. For example, a set of restriction enzyme recognition sites, such as the components commonly used for BBF RFC 10 BioBricks™, could be placed into a single <i>Collection</i> . A <i>Collection</i> might contain DNA components used in a specific project, lab, or custom grouping specified by the user. Arbitrary groupings and new <i>Collection</i> instances SHOULD NOT be created and named when the groupings are not defined, but also Collections SHOULD NOT be created whenever an arbitrary set is possible. Instances should only be used to represent a grouping that is useful to a user.

A *Collection* instance MUST have the following REQUIRED data properties.

Data Properties:
<i>uri</i> : (required) One and only one field of type URI (IETF RFC 2396). This property uniquely identifies the instance, and is intended to be used whenever a reference to the instance is needed. see also <i>DnaComponent.uri</i>
<i>displayId</i> : (required) One and only one value of type xsd:string starting with a letter or underscore followed by only alphanumeric and underscore characters. The displayID is a human-readable identifier for display to users. For example, users could use this identifier in combination with the namespace of the source as an unambiguous reference to the DNA construct.

A *Collection* instance CAN have the following OPTIONAL object and data properties.

Object properties:
<i>components</i> : (optional) Zero or more instances of type <i>DnaComponent</i> which are members of this <i>Collection</i> and represent DNA segments for engineering biological systems. For example, standard biological parts, BioBricks, pBAD, B0015, BioBrick Scar, Insertion Element, or any other DNA segment of interest as a building block of biological systems.
Data Properties:
<i>name</i> : (optional) Zero or one value of type xsd:string. The common name of the Collection is the most recognizable identifier used to refer to this <i>Collection</i> . It SHOULD confer what is contained in the <i>Collection</i> . It is often ambiguous (eg. Mike's Arabidopsis Project A; Parts from Sleight, et al. (2010) J.Bioeng; BBF RFC 10 DNA Components; My Bookmarked Parts).
<i>description</i> : (optional) Zero or one value of type xsd:string. Descriptions are a free-text field, therefore they SHOULD contain human-readable text describing the <i>Collection</i> for users to interpret what this <i>Collection</i> 'is'. This text SHOULD focus on an informative statement about the reason for grouping the <i>Collection</i> members. The result should allow users to interpret the reason for inclusion of members in this <i>Collection</i> (eg "Collecting parts which could be used to build honey production directly into mouse-ear cress"; "T9002 and I7101 variants from Sleight 2010, designs aim to improve stability over evolutionary time"; "Components useful when working with BBF RFC 10").

9. Examples

Sharing information about a variety of DnaComponents using the SBOL allows unambiguous specification of their DNA sequence-based descriptions. This section presents examples illustrative of different use cases as defined by SBOL:Core:model.

9.1 Annotated Composite DnaComponent

The first example is the SBOL:Core:model for the BioBrick™ BBa_I0462. The BBa_I0462 *DnaComponent* codes for the LuxR protein when inserted downstream of a promoter (Figure 6). Information comes from the Registry of Standard Biological Parts (<http://partsregistry.org>) to describe this canonical composite BioBrick™ part.

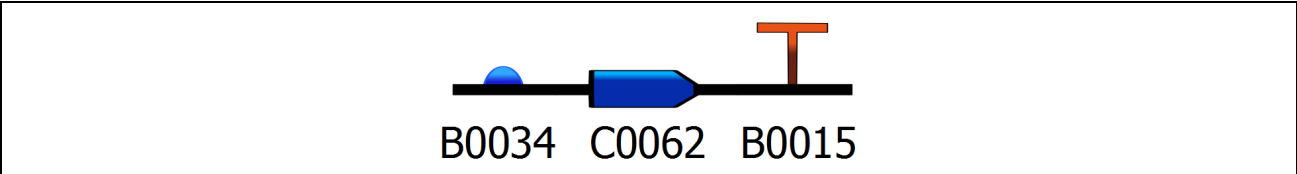
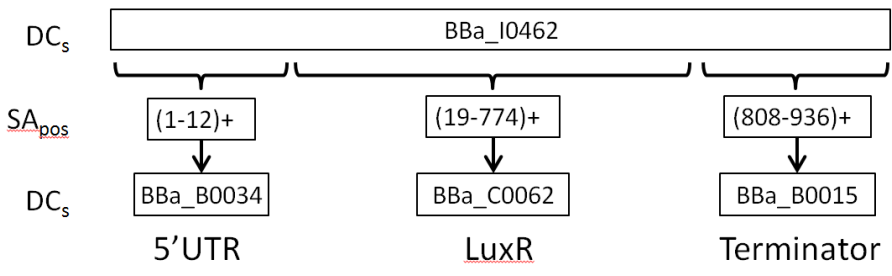


Figure 6. An example of a simple DNA design, BBa_I0462 (http://partsregistry.org/Part:BBa_I0462) drawn using SBOL:Visual symbols in TinkerCell (<http://tinkercell.com>) and composed of BBa_B0034, BBa_C0062, BBa_B0015 DnaComponents. The icons are labelled with a shorthand notation of the *displayName* from <http://partsregistry.org>.

In Figure 7a the BioBrick™ part BBa_I0462, a *DnaComponent*, is depicted with annotations of three *DnaComponents*: a ribosome binding site (BBa_B0034), the coding sequence for LuxR (BBa_C0062), and a double terminator BBa_B0015. In Figure 7b, the same *DnaComponent* is described using pseudocode as an example of SBOL:Core:model as text.

a.



b.

```
DnaComponent [  
  uri: http://partsregistry.org/Part:BBa_I0462  
  displayName: BBa_I0462  
  name: I0462  
  description: LuxR protein generator  
  annotations:[  
  
    SequenceAnnotation [  
      uri: http://sbols.org/anot#1234567  
      bioStart: 1  
      bioEnd: 12  
      strand: +  
      subComponent:[
```

```

    DnaComponent [
      uri: http://partsregistry.org/Part:BBa_B0034
      displayId: BBa_B0034
      name: B0034
      type: ribosome_entry_site
    ]
  ]
]

SequenceAnnotation [
  uri: http://sbols.org/annot#2345678
  bioStart: 19
  bioEnd: 774
  strand +
  subComponent:[

    DnaComponent [
      uri: http://partsregistry.org/Part:BBa_C0062
      displayId: BBa_C0062
      name: luxR
      type: CDS
    ]
  ]
]

SequenceAnnotation [
  uri: http://sbols.org/data#3456789
  bioStart: 808
  bioEnd: 936
  strand +
  subComponent:[

    DnaComponent [
      uri: http://partsregistry.org/Part:BBa_B0015
      displayId: BBa_B0015
      name: B0015
      type: terminator
    ]
  ]
]

DnaSequence [
  uri: http://sbols.org/seq#d23749adb3a7e0e2f09168cb7267a6113b238973
  nucleotides:
aaagaggagaaatactagatgaaaaacataaatgccgacgacacatacagaataattaataaaaattaaagcttgtagaagcaataatga
tattaatcaatgcttatctgatatgactaaaatggtacattgtgaatattatttactcgcgatcatttatcctcattctatggttaaat
ctgataatttcaatcctagataattaccctaataaaatggaggcaatattatgatgacgctaatttaataaaaatatgatcctatagtagat
tattctaactccaatcattcaccaattaattggaatataatttgaaaacaatgctgtaaataaaaaatctccaaatgtaattaaagaagc
gaaaacatcagggtcttatcactgggttttagtttcctattcatacggctaacaatggcttcggaatgcttagttttgcacattcagaaa
aagacaactatatagatagtttattttttacatgcgtgtatgaacataaccattaattgttccttctcctagttgataattatcgaaaaata
aatatagcaaataataaatcaaacacgatttaacccaaaagagaaaaagaatgttttagcgtgggcatgcgaaggaaaaagctcttgga
tatttcaaaaatattaggttgagtgagcgtactgtcactttccatttaaccaatgcgcaaatgaaactcaatacaacaaaccgctgcc
aaagtatttctaaagcaattttaacaggagcaattgattgccatactttaaaaattaataaacactgatagtgctagtgtagatcacta
ctagagccaggcatcaaataaaacgaaaggctcagtcgaaagactgggcctttctgtttatctgttgtttgtcgggtgaacgctctctac
tagagtcacactgggtcaccttcgggtgggcctttctgcgtttata
]
]

```

Figure 7. Annotated Composite *DnaComponent* **(a)** A diagram of the SBOL instances used to describe BBa_I0462. The gaps shown between the sequence annotations are unannotated segments of DNA. **(b)** Pseudocode is used to demonstrate the use of core data model structure and data fields in a complete example of a *DnaComponent*.

9.2 Multi-Tiered Annotated *DnaComponent*

The next example depicts the composition of BBa_I0462 in terms of each of its *subComponents* (Figure 8).

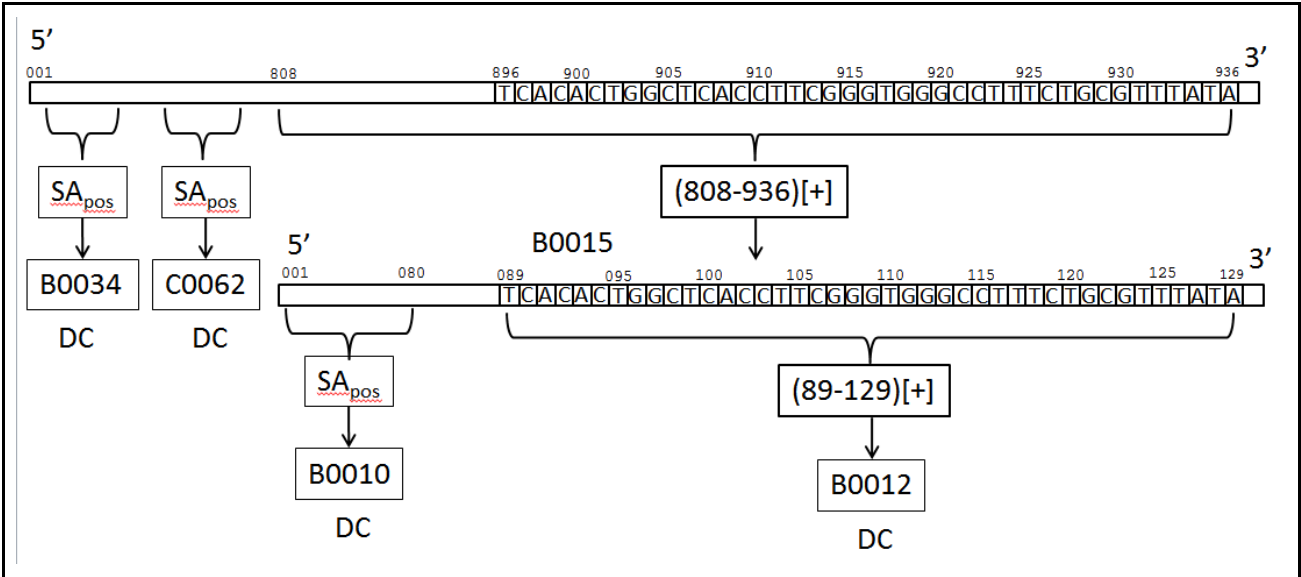


Figure 8. An expanded instance of BBa_I0462, which demonstrates key features of SBOL:Core:model. In this instance, the BBa_B0015 component of BBa_I0462 from the examples above is composed of two elements itself, BBa_B0010 and BBa_B0012. The letters of the DNA sequence in the top *DnaComponents* is omitted, so only the sequence corresponding to BBa_B0012 is shown.

9.3 Partially Realized Design Template

This example illustrates the partial specification of designs in terms of *DnaComponent* layout constraints. Figure 9 demonstrates the use of *SequenceAnnotations* with a Relative Position to specify the order of *DnaComponents* within a planned *DnaComponent*.

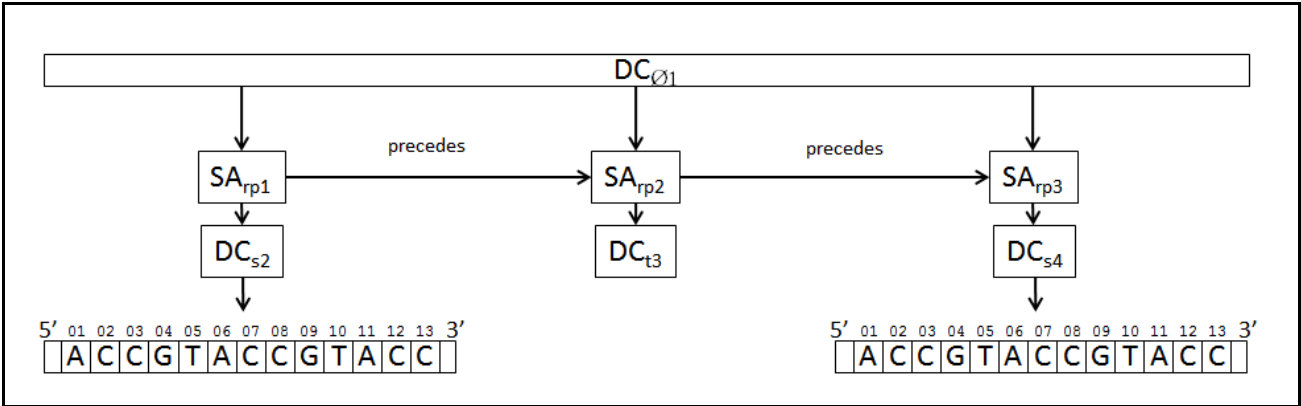
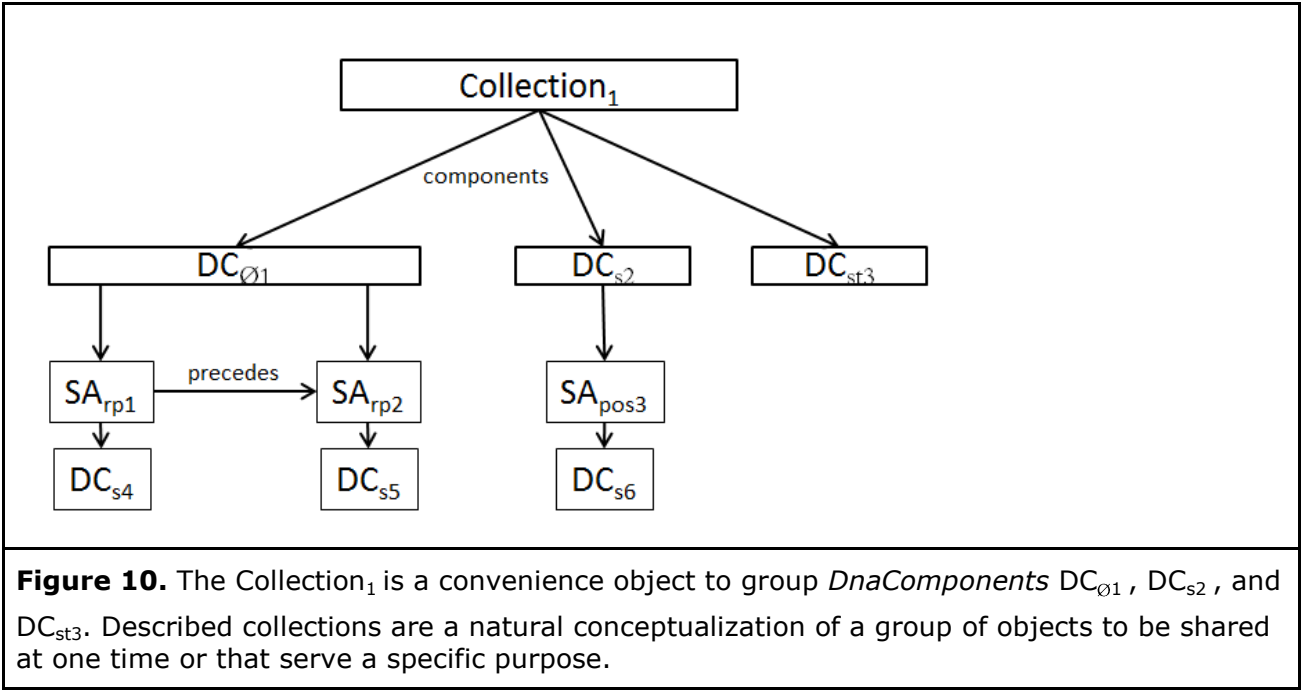


Figure 9. The design template for *DnaComponent* DC₀₁ specifies that at least three *DnaComponents* must be present in this design. Their ordering is constrained, DC_{s2} precedes DC_{t3} and DC_{t3} precedes DC_{s4}. In this template the DC_{s2} and DC_{s4} already have a *DnaSequence* specified, however DC_{t3} does not, instead it specifies a type which it must be constrained to. Therefore, the DC_{t3} component can be filled in to match the type constraint later.

9.4 Collection

To provide an organizational container for multiple *DnaComponent* instances, we provide the Collection class. The example in Figure 10 shows a Collection with multiple *DnaComponents* grouped together and ready to be shared between software applications.



10. Serialization

Examples of serialization are maintained on the web, and will be updated as the libSBOL reference implementation (<http://github.com/synbiodex>) is finalized. (<http://www.sbolstandard.org/initiatives/serialization>)

11. Best Practices

For SBOL version 1.1.0, best practices are being maintained in a dynamic document on the web, and will be updated as the use of SBOL increases (<http://www.sbolstandard.org/initiatives/best-practices>)

In future versions, Best Practices and Validation Criteria will be included in the specification.

12. Authors' Contact Information

Michal Galdzicki mgaldzic@uw.edu (SBOL Editor)
Mandy L. Wilson mandywil@vbi.vt.edu (SBOL Editor)
Cesar A. Rodriguez cesarr@berkeley.edu (SBOL Editor)
Laura Adam ladam@vbi.vt.edu
Aaron Adler aadler@gmail.com
J. Christopher Anderson jcanderson@berkeley.edu
Jacob Beal jakebeal@bbn.com
Deepak Chandran deepakc@uw.edu
Douglas Densmore dougd@bu.edu
Omri A. Drory omri@genomecompiler.com
Drew Endy endy@stanford.edu
John H. Gennari gennari@uw.edu
Raik Grünberg raik.gruenberg@crg.es
Timothy S. Ham tsham@lbl.gov
Allan Kuchinsky allan_kuchinsky@agilent.com
Matthew W. Lux mlux@vbi.vt.edu
Curtis Madsen curtis.madsen@utah.edu
Goksel Misirli goksel.misirli@ncl.ac.uk
Chris J. Myers myers@ece.utah.edu
Jean Peccoud jpeccoud@vbi.vt.edu
Hector Plahar hplahar.jbei@gmail.com
Matthew R. Pocock matthew.pocock@ncl.ac.uk
Nicholas Roehner n.roehner@utah.edu
Trevor F. Smith trevorfsmith@gmail.com
Guy-Bart Stan g.stan@imperial.ac.uk
Alan Villalobos avillalobos@dna20.com
Anil Wipat neilwipat@googlemail.com
Herbert M. Sauro hsauro@uw.edu (SBOL Chair)

13. References

- Beal, J., Lu, T., & Weiss, R. (2011). Automatic Compilation from High-Level Biologically-Oriented Programming Language to Genetic Regulatory Networks. *PLoS ONE*, 6(8), e22490. Public Library of Science.
<http://dx.plos.org/10.1371/journal.pone.0022490>
- Bilitchenko, L., Liu, A., Cheung, S., Weeding, E., Xia, B., Leguia, M., Anderson, J. C., et al. (2011). Eugene--a domain specific language for specifying and constraining synthetic biological parts, devices, and systems. *PloS one*, 6(4), e18882.
<http://dx.plos.org/10.1371/journal.pone.0018882>
- Eilbeck, K., Lewis, S. E., Mungall, C. J., Yandell, M., Stein, L., Durbin, R., & Ashburner, M. (2005). The Sequence Ontology: a tool for the unification of genome annotations. *Genome biology*, 6(5), R44. doi:10.1186/gb-2005-6-5-r44
- Endy, D. (2005). Foundations for engineering biology. *Nature*, 438(7067), 449-53. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/16306983>
- Galdzicki, M., Rodriguez, C., Chandran, D., Sauro, H. M., & Gennari, J. H. (2011). Standard Biological Parts Knowledgebase. (C. Schönbach, Ed.) *PLoS ONE*, 6(2), e17005. Public Library of Science. doi:10.1371/journal.pone.0017005
- Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., et al. (2003). The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4), 524.
<http://bioinformatics.oxfordjournals.org/cgi/content/abstract/19/4/524>
- Kaern, M., Blake, W. J., & Collins, J. J. (2003). The engineering of gene regulatory networks. *Annual review of biomedical engineering*, 5, 179-206. doi:10.1146/annurev.bioeng.5.040202.121553
- Mungall, C. J., Batchelor, C., & Eilbeck, K. (2010). Evolution of the Sequence Ontology terms and relationships. *Journal of Biomedical Informatics*. doi:10.1016/j.jbi.2010.03.002
- Nandagopal, N., & Elowitz, M. B. (2011). Synthetic Biology: Integrated Gene Circuits. *Science*, 333(6047), 1244-1248. Retrieved from <http://www.sciencemag.org/content/333/6047/1244.abstract>
- Savageau, M. A. (2001). Design principles for elementary gene circuits: Elements, methods, and examples. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 11, 142.
- Sleight, S. C., Bartley, B. A, Lieviant, J. A, & Sauro, H. M. (2010). Designing and engineering evolutionary robust genetic circuits. *Journal of biological engineering*, 4(1), 12. doi:10.1186/1754-1611-4-12

14. Appendix

The Sequence Ontology is used in SBOL to specify the types of *DnaComponents*. Below are some samples of these types which are commonly used in synthetic biology designs. For example, the value of `dc.type` MUST be a valid SO URI such as those listed below. Additional Sequence Ontology types can be found at the website (<http://www.sequenceontology.org/>) the namespace used for SO URIs is "`http://purl.obolibrary.org/obo/`" these URIs are maintained as Persistent Uniform Resource Locators (PURLs) by the OBO Foundry (<http://obolibrary.org/about.shtml>) and can be used to retrieve additional information about the term.

Promoter	A regulatory_region composed of the TSS(s) and binding sites for TF_complexes of the basal transcription machinery. SO URI: http://purl.obolibrary.org/obo/SO_0000167 SO Name: promoter
Operator	A regulatory element of an operon to which activators or repressors bind, thereby effecting translation of genes in that operon. SO URI: http://purl.obolibrary.org/obo/SO_0000057 SO Name: operator
CDS	A contiguous sequence which begins with, and includes, a start codon, and ends with, and includes, a stop codon. SO URI: http://purl.obolibrary.org/obo/SO_0000316 SO Name: cds
5' UTR	A region at the 5' end of a mature transcript (preceding the initiation codon) that is not translated into a protein. SO URI: http://purl.obolibrary.org/obo/SO_0000204 SO Name: five_prime_utr
Terminator	The sequence of DNA located either at the end of the transcript that causes RNA polymerase to terminate transcription. SO URI: http://purl.obolibrary.org/obo/SO_0000141 SO Name: terminator
Insulator	A transcriptional cis regulatory region that, when located between a CM and a gene's promoter, prevents the CRM from modulating that genes expression. SO URI: http://purl.obolibrary.org/obo/SO_0000627 SO Name: insulator
Origin of Replication	The origin of replication; starting site for duplication of a nucleic acid molecule to give two identical copies. SO URI: http://purl.obolibrary.org/obo/SO_0000296 SO Name: ori
Primer Binding Site	Non-covalent primer binding site for initiation of replication, transcription, or reverse transcription.

	SO URI: http://purl.obolibrary.org/obo/SO_0005850 SO Name: primer_binding_site
Restriction Enzyme Recognition Site	Represents a region of a DNA molecule which is a nucleotide region (usually a palindrome) that is recognized by a restriction enzyme. SO URI: http://purl.obolibrary.org/obo/SO_0001687 SO Name: restriction_enzyme_recognition_site