

Data model for biopart datasheets

ISSN 2398-6182

Received on 27th September 2017

Revised on 17th November 2017

Accepted on 7th December 2017

doi: 10.1049/enb.2017.0020

www.ietdl.org

Iñaki Sainz de Murieta, Matthieu Bultelle, Richard I. Kitney ✉

Department of Bioengineering, Imperial College London, London SW7 2AZ, UK

✉ E-mail: r.kitney@imperial.ac.uk

Abstract: This study introduces a new data model, based on the DICOM-SB (see glossary of terms for definition of acronyms) standard for synthetic biology, that is capable of describing/incorporating the data, metadata and ancillary information from detailed characterisation experiments – to present DNA components (bioparts) in datasheets. The data model offers a standardised mechanism to associate bioparts with data and information about component performance – in a particular biological context (or a range of contexts, e.g. chassis). The data model includes the raw, experimental data for each characterisation run, and the protocol details needed to reliably reproduce the experiment. In addition, it provides metrics (e.g. relative promoter units, synthesis/growth rates etc.) that constitute the main content of a biopart datasheet. The data model has been developed to directly link to DICOM-SB, but also to be compatible with existing data standards, e.g. SBOL and SBML. It has been implemented within the latest version of the API that enables access to the SynBIS information system. The work should contribute significantly to the current standardisation effort in synthetic biology. The standard data model for datasheets is seen as a necessary step towards effective interoperability between part repositories, and between repositories and BioCAD applications.

Nomenclature

Glossary of terms

AHL	N-acyl homoserine lactones in gram-negative bacteria	MG1655	E. coli strain K12
BioCAD	computer aided design tools for biology	MOPS Medium	growth culture medium used in synthetic biology
BSI	British Standards Institute	NIST	US National Institute for Standards and Technology
CAD	Computer Aided Design	ORI	origin of replication of a plasmid
Cell Designer	software package for designing gene circuits	OWL	web ontology language
DICOM	Digital Image Communication in Medicine, an international standard for biomedicine	PAMDB	Plant Associated and Environmental Microbes Database
DICOM-SB	working title for a new part of the DICOM standard specifically designed for synthetic biology	RestFUL	type of web services using the JAVA language
GeneBank	online repository of gene sequences supporting the GeneBank data format	RPU	relative promoter units, used as a relative measure of promoter performance.
GFP	Green Florescent Protein	SBML	Systems Biology Mark-up Language. A mark-up language, similar to HTML, specifically designed for systems biology
HTML	Hypertext Mark-up Language, used for web browsing and web-based applications	SBOL	Synthetic Biology Open Language; an open standard for the representation of <i>in silico</i> biological designs
iBioSim	software package for simulating gene circuits	SBOL Stack	platform for storing, publishing and sharing synthetic biology designs
ICE/JBEI ICE	web-based registry platform for managing synthetic biological parts	SBOL API	application programming interface for SBOL
iGEM	international Genetically Engineered Machine competition – a major international annual student competition in synthetic biology	SEVA	Standard European Vector Architecture for the de-/re-construction of biological entities
iGEM Registry of Parts	registry of BioParts associated with the iGEM competition	SNOMED	systematically organised computer processable collection of medical terms
libSBOLj	Java library associated with the SBOL standard	SynBIS	web-based synthetic biology information system designed by the Kitney Lab at Imperial College
J23101	constitutive promoter part of the Anderson Dataset, often called the Kelly Promoter, and often used as a ‘standard’ promoter	TTL	Transistor Transistor Logic; a type of technology used in integrated electronic circuits
		UML	Unified Modelling Language; a general-purpose, developmental, modelling language in the field of software engineering

1 Introduction

'Synthetic biology is the design and engineering of biologically based parts, novel devices and systems as well as the redesign of existing, natural biological systems' [1]. Born out of the convergence of advances in engineering, biology, computer science and chemistry, it is a young discipline that aims to make biology easier to engineer, thus enabling the rapid development and production of new biological products.

One of the main drivers for the development of synthetic biology has been the application of the engineering principles of modularity, characterisation and standardisation to the engineering of biology [2]. It is indeed thanks to the judicious deployment of these concepts that in other branches of engineering (e.g. electrical engineering or aeronautics) complex systems can be produced from the combination of standardised components. The approach may look like a needless limitation of the space of possible designs, but it is a pragmatic acknowledgement that: first, the space of imaginable designs will always exceed our ability to implement them; and second, it is a waste to build too many solutions to the same problem. In addition, the interfacing of the parts is frequently non-trivial, time consuming and optimal composition rules are often unknown. Importantly, these apparent limitations are offset by the fundamental power of the approach: every device does not need to be designed and built from scratch. Existing devices (either originating from nature or designed, possibly by other groups in different locations) can be reused and altered with other standard parts. Consequently, interfacing parts becomes simpler. Finally, newly-constructed parts and devices can then be characterised and added to the repository of available parts and devices than can be reused for other projects.

Modularity and standardisation also facilitate the division of labour (the specialisation of cooperating individuals who perform specific tasks and roles). In the case of synthetic biology, this is already happening. As projects become more complex, larger teams of specialists (ranging from engineers to computer scientists and molecular biologists) are needed to bring them to successful completion. Furthermore, some of the technologies have matured to such an extent that viable commercial companies can be built around them and it is beneficial for portions of a project to be entrusted to these companies. For instance, it is now common to outsource the DNA synthesis to commercial companies such as Twist [3] or DNA 2.0 [4] (and for final assembly to take place in-house using an ever wider range of techniques [5–13]) – while companies such as Transcriptic [14] offer their clients the design experiments in the cloud, and then run them on their automated platforms.

Despite all the signs that synthetic biology is embracing engineering concepts and developing its own specific set of methods and good practices, the capacity to routinely build or modify new devices and pathways from standardised components remains difficult, if not elusive. One of the main stumbling blocks is the absence of large, online open-source repositories of fully characterised parts [15]. The situation could, however, change rapidly in the future due to advances in several areas – the most notable being in automation and the development of comprehensive parts repositories.

1.1 Automation

Although liquid handling automation had been identified as an important technology for synthetic biology [16], there were only a few reports on the application of automation to other laboratory processes until 2014, with the notable exception of the work carried out in the Alon Lab on GFP-based characterisation assays [17, 18]. This is despite clear evidence that the use of automation

in the execution of synthetic biology experiments can lead to significant improvements in throughput of repetitive processes [19]. This situation is now changing with the opening of DNA foundries around the world [20, 21]; the increased disclosures by companies like Amyris [22] or Gingko [23]; and the development of automated platforms (such as the microfluidics platform by Linshiz *et al.* [24]).

1.2 Part repositories

The first wave of repositories sought to store basic information on parts (sequence, hierarchical structure and other useful information). The iGEM Registry of Standard Biological Parts [25] is the oldest, largest (it is regularly expanded thanks to the compulsory contributions of teams taking part in the annual iGEM competition) and best-known part repository. Plasmid repositories such as Addgene [26] and the Standard European Vector Architecture (SEVA) repository [27] have focused on modular design and quality control. The Virtual Parts repository [28] has sought to associate biological parts (mainly from *B. subtilis*) with their associated biochemical reactions (obtained from the literature). The second wave of repositories including the SBOL Stack [29] and the Joint BioEnergy Institute's Inventory of Composable Elements (ICE) [30], comprise agile platforms installed locally. They allow synthetic biologists to manage their own data and information about biological parts (and designs). Recently, a third wave of repositories has begun to appear – seeking to create an explicit link between parts and characterisation data (and models derived from them). At Imperial College London, SynBIS [31] has been developed to facilitate the dissemination of detailed characterisation data for bioparts, and, for example, Huynh and Tagkopoulos [32] created the PAMDB repository of parts by integrating data from more than a 100 publications.

An indication that the computer-assisted biodesign is getting close was provided by Nielsen *et al.* [33] when they introduced Cello. They describe it as the first programming language for living cells. Cello applies electronic design automation principles to the problem of biodesign and builds on top of a dedicated repository of gates, sensors and actuators.

In this paper, we address one of the important problems associated with the development of really effective repositories of characterised parts, i.e. the development of a data model for biopart datasheets. In addition, we discuss how to aggregate and structure characterisation data and information so they can be stored and shared effectively on parts repositories.

1.3 Motivation: datasheets in synthetic biology

In other fields of engineering (e.g. electrical and mechanical), component behaviour is routinely quantitatively and qualitatively described in standardised, comprehensive datasheets. Datasheets offer, in a compact form [34], a representation of a series of context-dependent, input–output relationships, tolerances, requirements – as well as details of the relationship between the component and other systems. Such properties make datasheets an appealing framework to describe, encode and visualise the behaviour of parts and devices in synthetic biology.

Whenever standardised components are developed (to be ultimately assembled into larger, more complex systems), their functional behaviour must be documented in a clear, consistent and, as far as possible, unambiguous manner. For example, in computer science code is expected to be styled and annotated [For instance, Python's Pep 008 [35] and PEP 257 [36] contain the guidelines for the style guide and the dosctrings (documentation lines) conventions for the language.].

The first basic datasheets were introduced in synthetic biology in 2008 by Canton *et al.* [37] and Arkin [38]. They presented the first datasheet for a cell–cell communication receiver, named BBa_F2620. In it they not only discussed the specifics of the characterisation of BBa_F2620, but, also, showed to a

non-engineering audience what the design and use of datasheets means in practice. Their datasheets mixed qualitative information (structure description, summary of its function) with quantitative measurements such as

- Static performance response curve: $\text{output} = f([\text{AHL}])$.
- Dynamic response for a given AHL concentration.
- Input compatibility: response curve for inducers similar to AHL.
- Reliability: evolutionary reliability of the receiver.

In the same issue of *Nature* [38] Arkin expanded the discussion. He made the case that datasheets provide a suitable framework to describe the behaviour of part and devices – and presented a minimal list of properties that should feature in a datasheet. In addition, and importantly, he described some of the limitations of the datasheet approach. Using three examples (a cell–cell communication module a la BBa_F2620, a DNA-binding protein domain and a therapeutic bacterium) to discuss the metrics that could be used to capture the behaviour of a component, Arkin showed how variable the biologically-relevant metrics can be (and how much they change with the type of component). Further, while some of the metrics were unambiguous (e.g. the transfer function of steady-state and dynamic induction curves of the output promoter by different homoserine lactone inputs in different cells), others were not as specific (e.g. effect of induction on cellular growth rate, the survival in different hosts etc.). Finally, Arkin discussed the wisdom of using models (such as the Hill function) to describe component behaviour, noting the potential price in accuracy (There is always the potential danger of models being too simple, but they provide a means of avoiding ambiguous behaviour.).

Despite the initial burst of enthusiasm, few datasheets followed the BBa_F2620 proof of concept. The next attempt was described by Lee *et al.* [39]. They used datasheets to describe the properties of a family of modular plasmids, designed to be compatible with the BglBrick standard of part assembly [9]. Noting that ‘the standardisation of bioparts and their assembly is one of the core ideas of synthetic biology’, these authors treated the swappable parts of their design (ORI, resistance, expression module) as variables with potential effects on the plasmid behaviour and characterised accordingly.

Both user cases presented by Canton *et al.* [37] and Lee *et al.* [39], as well as the guidelines devised by Arkin [38], show how datasheets can be used to document the behaviour of a part, device or system. Tools have subsequently been built to help with the creation of similar datasheets. Examples are: OWL [40], (that integrates with existing registries such as JBEI ICE); and tools such as Pigeon [41] and Raven [42]; that give users a flexible way to describe component behaviour (consistent with the variety of parts and possible associated metrics) – while standardising the output (e.g. HTML page with standard typesetting, and/or a PDF file). However, these approaches only provide a partial insight into the use of datasheets in synthetic biology, as they only focus on human-readable datasheets and their content. More recently, the limitations of such a narrow focus have become apparent as synthetic biology develops into an engineering discipline. Some of these limitations will now be discussed.

First, the authors cited in the previous paragraph describe a typical user case involving a biologist browsing through a catalogue of parts and checking their datasheets one by one to find a part that meets some design specifications (This is very much like browsing the TTL [43] catalogue of electronic components.). Such a user case has become increasingly inappropriate. Relying on humans to parse the content of the datasheets is an approach that cannot scale. This is because, unlike other forms of engineering, where the number of similar components is limited, synthetic biology is bound to rely on large libraries of similar components (Such components may vary from each other by as little as a few base pairs, if they were generated with techniques such as error-prone PCR [44–46] – or, conversely, endowed with similar

characteristics, but based on very different biochemical processes [47–50].).

Second, datasheets display only a portion of the data gathered during its generation (This is of course by design: the metrics they contain offer an incomplete, but adequate, picture of the behaviour of a part/device/system.). Such metrics are the outcome of the processing of raw data. In addition, the experimental conditions and protocols are often not included in their entirety. The omission of so much information from a datasheet has important consequences – the most important being that the results they contain cannot be verified or duplicated. In addition, the data associated with the datasheet cannot be used for other purposes (for instance to calibrate a cell level model [51, 52], or to compute other metrics).

Finally, the data remains locked in the datasheets. Datasheets should not be seen as end points, but, rather, as intermediaries that hold valuable information to be used in other projects (CAD, systems biology etc.). Although some of the data can be removed from a human-readable html page or a pdf, the data contained in the image plots cannot be used directly. For instance, simulation software such as TinkerCell [53], CellDesigner [54] and iBioSim [55], requires some human intervention or some ad-hoc conversion scripts to ‘fill in the gaps’ (with an associated error rate).

For all the reasons cited above, the concept of a datasheet for use in synthetic biology, must, therefore, be expanded to include a machine-readable version (henceforth referred to as an ‘electronic datasheet’ or ‘e-datasheet’) endowed with its own versatile data model.

2 Results

2.1 Electronic datasheets in synthetic biology

In this section, we present the foundations of the datasheet data model (its most important entities will be detailed in following sections). It is important to note that whether datasheets are a suitable framework to describe the behaviour of a part, device or system is beyond the scope of this paper. Unlike in electronics, where a component interfaces with a handful of other components, a part, device or system in synthetic biology is always liable to interact with existing pathways (for instance, crosstalk or competition for resources). It is obvious that a datasheet cannot comprehensively capture these interactions, even for a given biological context. Nevertheless, an electronic datasheet can provide useful, conveniently structured information. It can therefore be used to inform the design of more complicated systems or investigate some of their properties; providing it performs the following three functions.

2.2 Datasheet design strategy

(i) An electronic datasheet should build on the work of Arkin [38], Canton *et al.* [37] and Lee *et al.* [39], so it can be easily serialised into a human-readable datasheet (see Fig. 1, as an example of a high-level template). This means that an electronic datasheet must be able to encode the type of biologically-relevant metrics discussed by Arkin as a means to describe the behaviour of a component. However, in addition, it must provide background information and metadata on the characterised part/device/system and on the experimental context.

The ‘Identity’ section includes high level, background information on the part/system/device that is characterised (name, origin, sequence etc.), as well as qualitative information on its inputs/outputs and known crosstalk. Quantitative results are represented by a range of transfer functions (referred to as ‘metrics’). Contextual information on the experiments typically includes description of the constructs used, and the experimental protocol(s) that were run.

(ii) An electronic datasheet should contain or link to all the information collected and generated as part of the characterisation

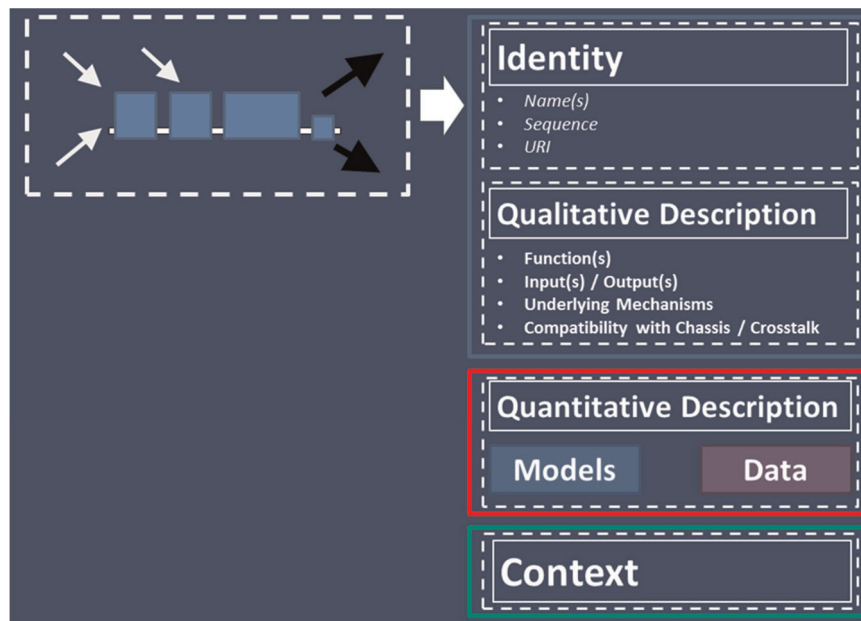


Fig. 1 High-level template for a human-readable datasheet of a biopart/device/system

process, including all the intermediary information. The most natural way to do this is to take advantage of the characterisation workflow (see Fig. 2) to structure the data and link the information to the stages of the datasheet generation. Seen from a process point of view, an electronic datasheet simply becomes the ordered record of this workflow.

It is important to note that in practice datasheet generation is not a linear process. Datasheets comprise several elements, each of which are generated/assembled in different ways – for example, the construction of the various genetic constructs and subsequent transformations.

Electronic datasheets must be able to record these workflows. The metrics listed in the datasheet are of particular interest. A metric is the product of an analysis protocol (with associated algorithms and human interventions – see Fig. 3) applied to a subset of the experimental data. For instance, with the Kelly characterisation protocol [56], the relative promoter unit (RPU) is calculated as follows:

- (1) In the exponential phase, calculate the ratio of corrected fluorescence over corrected OD for the promoter of interest.
- (2) Repeat the calculation with data for the reference promoter (J23101).
- (3) The RPU is the ratio of the quantity derived from (1) divided by that derived from step (2).

The calculation requires datasets issued from four experiments: a positive experiment, a reference experiment (with J23101),

a negative experiment (no promoter) and a blank experiment. For the first three experiments both OD and GFP fluorescence are needed; for the blank only the OD is required.

(iii) An electronic datasheet should be structure and formatted in such a way that existing resources and software can easily retrieve and further process the data and information. In practice this means using existing data standards and good practices whenever possible. The DICOM-SB data standard for raw data is of particular interest for this work. DICOM-SB [57] is a new standard based on the Digital Imaging and Communications in Medicine (DICOM) standard [58] (an important standard in biomedicine). DICOM-SB enables the efficient capture and exchange of experimental data, metadata and protocol information. This is because of its modular, extensible data model (specifically developed for synthetic biology); and its compatibility with other standards. DICOM-SB also has properties that are directly inherited from the original DICOM standard. These include the capacity to use binary encoding to optimise data storage for large amounts of data and a set of services orientated towards the automatic exchange of data and information between modalities and repositories (As far as the authors are aware, the DICOM-SB data model is also the only published data model for an experiment in synthetic biology.).

In order to understand the compatibility of our data model with the SBOL (the Synthetic Biology Open Language),

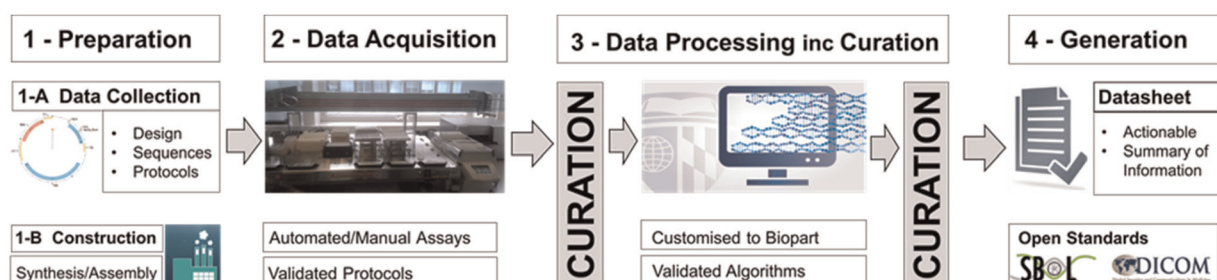


Fig. 2 Typical characterisation pipeline. It comprises in four successive steps. (1) Preparation: when all genetic constructs are built, cloned into designated plasmid vectors and transformed into the hosts of interest. (2) Data acquisition: the capture of several runs of the characterisation experiment. (3) Data processing: when experimental data are processed to calculate the metrics that describe the behaviour of the part. (4) Datasheet generation

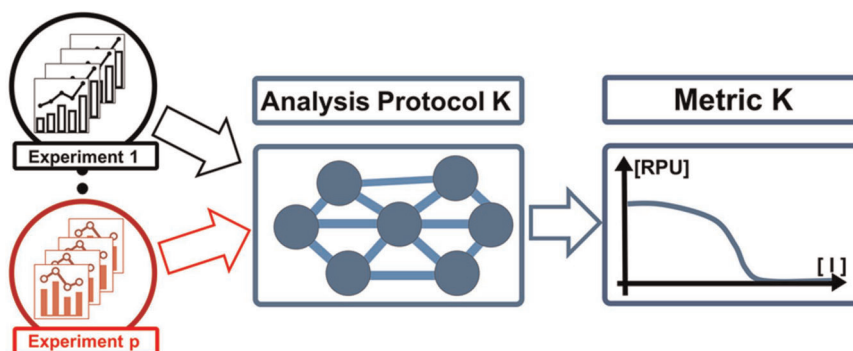


Fig. 3 Metric generation. Metrics are generated by applying a set of computations to the raw data arising from a set of different experiments. It follows that each metric should be linked to the raw data, the analysis protocol and what intermediary files are available

we now provide an introduction to some of the salient points. SBOL is a synthetic-biology-focused standard that builds on the GenBank [59] standard for naturally occurring, annotated, sequences. In this context, SBOL captures the hierarchy and modularity of designs in synthetic biology by allowing fully hierarchical annotation of DNA components within DNA components [60]. A companion notation system – SBOL visual [61] – provides a standardised way to describe genetic designs, making their sharing easier. SBOL (now in version 2.1) is supported by many bioinformatics and molecular cloning design tools [55, 62, 63]. SBOL's original version was limited in its scope, as it could only represent DNA components and their hierarchical composition by means of sequence annotations. A revision to the core model [60, 64] has been presented to support a wider range of components (with or without a sequence), including RNA components, protein components, small molecules and molecular complexes. Perhaps more importantly, SBOL 2.0 (and 2.1) are reorganised around the idea of module, functions and ports – notions that are familiar to designers of electrical circuits.

2.3 Notations and exemplar

The work presented in the subsequent sections of the paper uses our data model to enhance the SBOL functionality by providing a compatible datasheet extension that is able to annotate DNA components with information related to their performance. In order to facilitate this, we are using UML [65] to describe the entities, properties and relations between entities in the data

model (see Figs. 4–10) – a common approach to describing data models.

The following UML notations apply to all UML figures:

- Boxes represent classes. Details of class properties have been moved to supplementary tables for better readability.
- Edges represent unidirectional associations between classes (direction going from the cross to the arrowhead).
- White diamonds implement aggregation associations, meaning that the child class (pointed by the arrowhead) can exist without the parent (pointed by the white diamond).
- Black diamonds implement composition associations, meaning that the child class (pointed by the arrowhead) cannot exist without the parent (pointed by the black diamond).
- Numbers indicate multiplicity: 1 – the parent must have one child; 0..1 – the parent can have none or one child; 0..* – the parent can have none, one or multiple children; 1..* – the parent can have one or multiple children.
- Edges with a white triangle encode inheritance associations, meaning that the child inherits all the properties and associations from the parent (pointed by the white triangle).

To illustrate the meaning of the various entities involved, we will use a very simple characterisation process as exemplar – the characterisation of a constitutive promoter in *E. coli* using GFP as reporter (see supplementary information for a detailed description of the protocol and constructs, as run at the Centre for Synthetic Biology and Innovation at Imperial College).

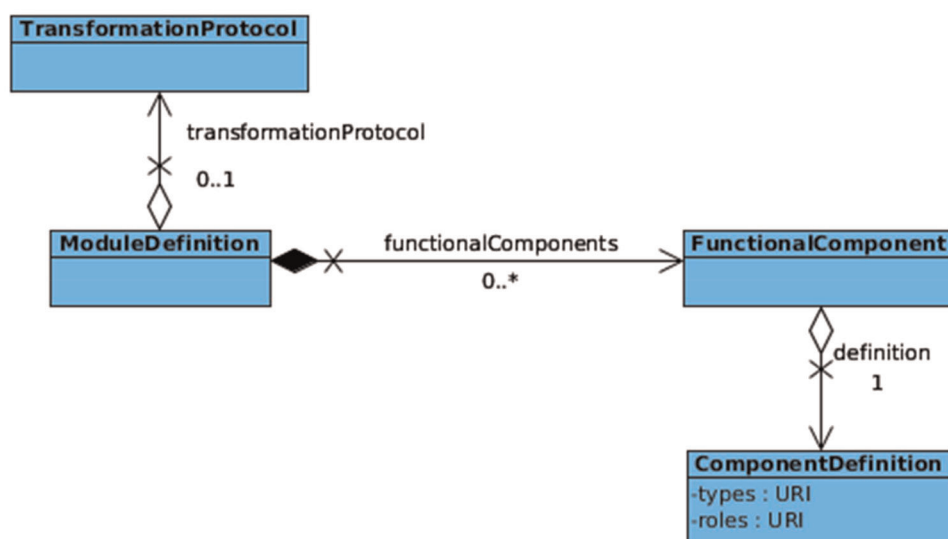


Fig. 4 UML diagram describing transformations

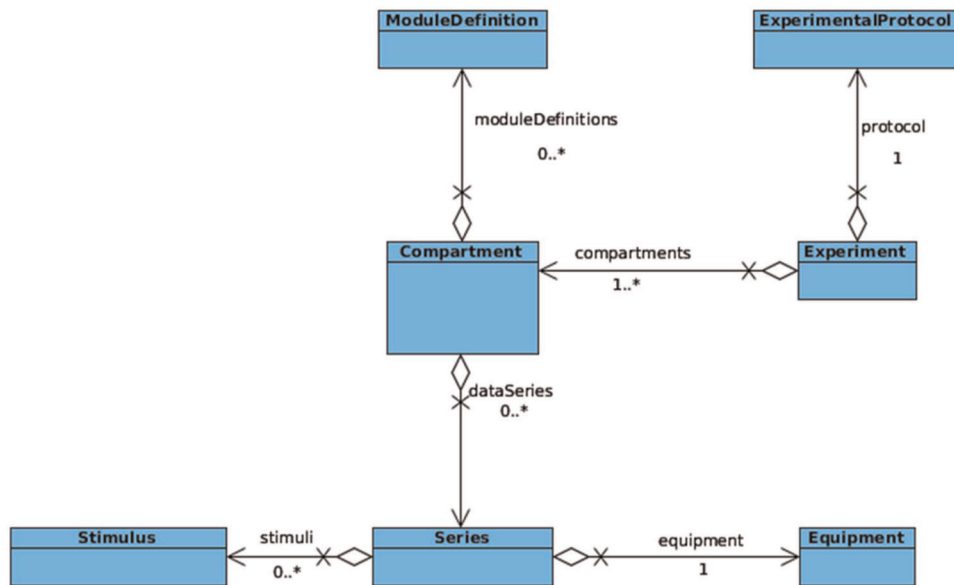


Fig. 5 UML diagram describing experiments

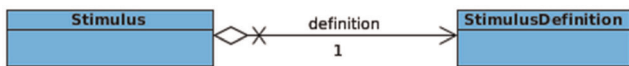


Fig. 6 UML diagram describing stimuli

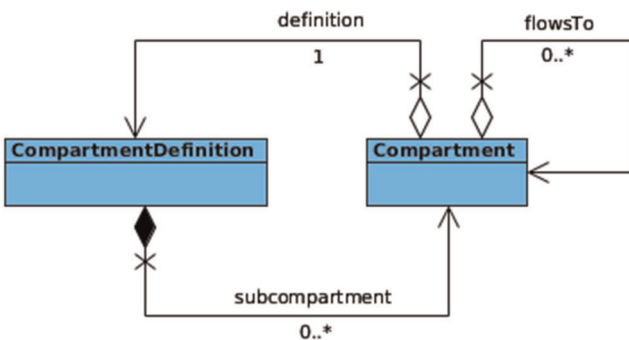


Fig. 7 UML diagram describing compartments

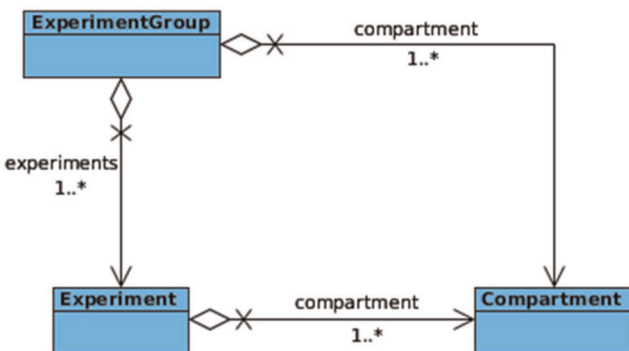


Fig. 8 UML diagram describing experiment groups

The most important elements of the characterisation process are:

- *Genetic context*: For the positive experiment, *E. coli* MG1655 has been transformed with a Kanamycin-resistant plasmid (p15a ORI, average copy number 15). The characterisation construct includes

a GFP gene driven by a constitutive promoter (the promoter to be characterised). For the negative experiment, the promoter driving GFP has been removed, while the GFP gene is driven by the reference promoter J23101 for the reference experiment. Blank experiments only contain the growth medium (MOPS).

- *Experimental protocol*: Characterisation takes place on an automated robotic platform. After an initial overnight culture, the samples are plated onto a 96-well plate in fresh MOPS medium (OD is controlled to achieve a target OD). The outgrowth phase lasts 90 min and is followed by a final dilution step to achieve different target OD. The subsequent assay phase is 6 h long. During the assay phase, plate reader data are acquired every 15 min; flow cytometry data after 3 and 6 h.

2.4 Defining transformations

In the context of characterisation, the objective of a synthetic biology experiment is the transformation of a host organism with a set of biopart constructs (following a transformation protocol, whose behaviour within the host is to be determined). Our model describes this event using the following classes (see Fig. 4 and Supplementary Table 1 for details of class properties):

- *Component definition*: From the SBOL standard, it describes the basic features to be tracked for each biopart under analysis. In this scenario, this class is used to describe the different DNA plasmids to be used in the transformation. This facilitates a detailed annotation of its DNA sequence and the representation of recursive biopart structures. In our example, one of the components will be the whole plasmid, others will be the characterisation construct inserted into the plasmid, and the promoters driving GFP (the promoter of interest for the positive, J23101 for the reference, no promoter for the negative).
- *Module definition*: The module definition in SBOL represents a grouping of structural and functional entities in a biological design. We are using this class to represent host cells that may be genetically modified (transformed) by DNA components before they are used in an experiment. In this context, the module definition class is used to represent this type of ‘cell design’ or transformation, as a combination of one host organism and a list of DNA components.

- The host details, such as name, species, strain or origin, enhance this module definition class as attribute annotations. Cell free

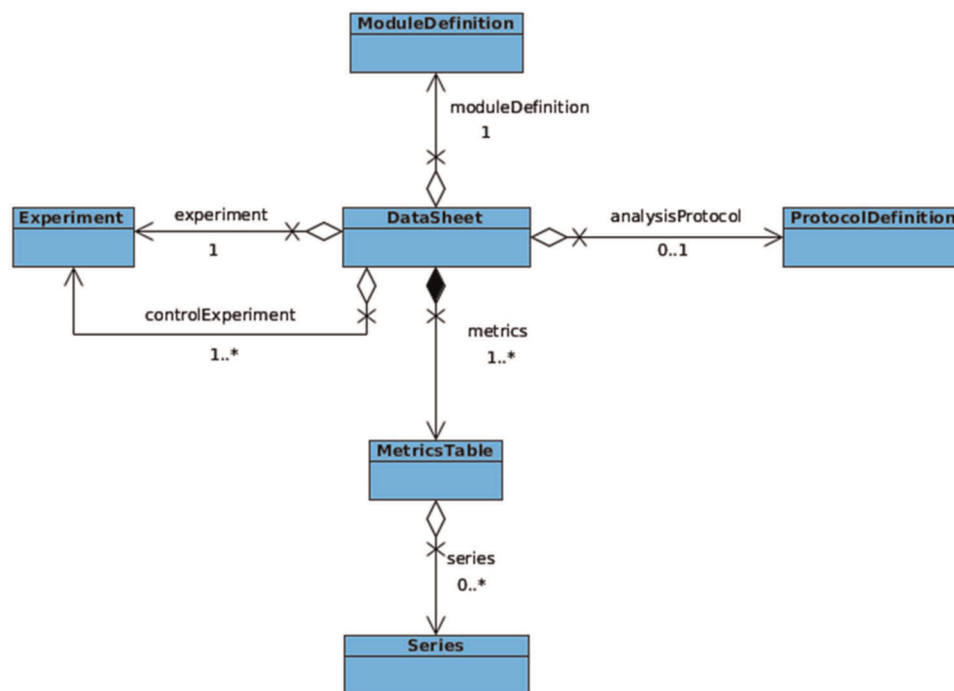


Fig. 9 UML diagram describing datasheets

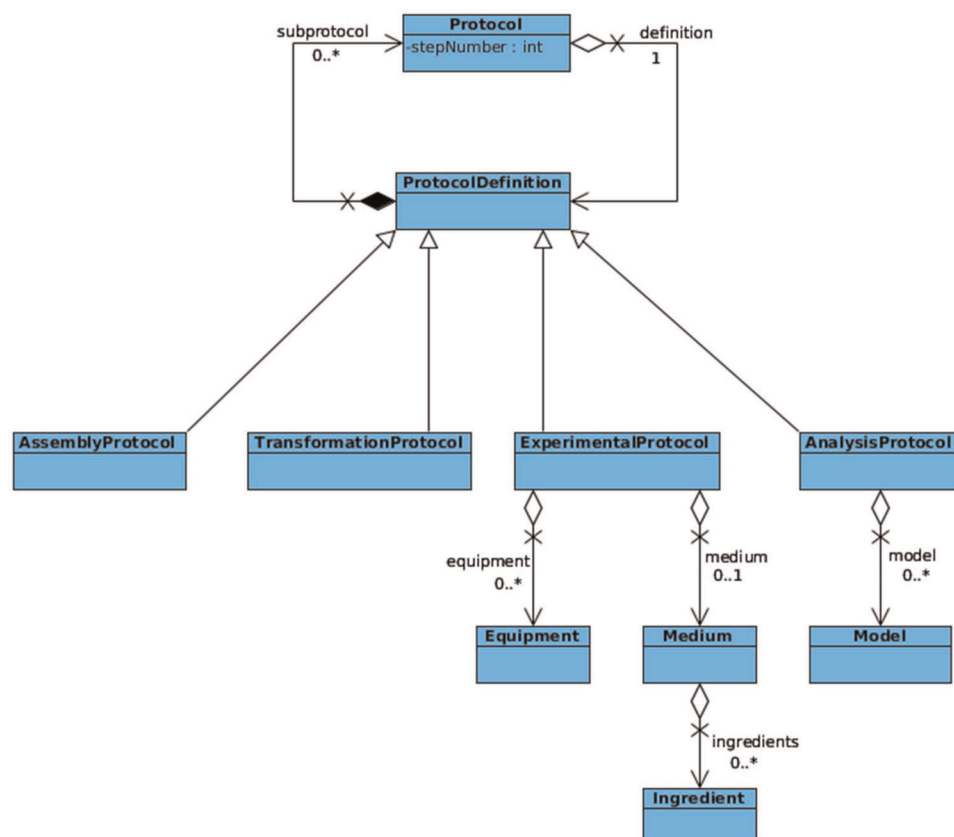


Fig. 10 UML diagram describing protocols

systems are also represented by this class, using a special host type. In our example, the host is always *E. coli* MG1655.

o The list of component definitions encoding the bioparts under analysis are linked to the module definition through functional component instances. This list may be empty, meaning that the

host would be used untransformed in the experiment (typically to be used as the control).

• *Protocol definition*: Optionally, a transformation can also include details about the transformation protocol in the laboratory, or the

assembly protocol used to build each of the DNA components inserted in the host.

2.5 Defining experiments

The objective of an experiment is to perform all the procedures required to analyse the change of behaviour that the integrated set of components produces in the host. Our model describes these events using the following classes (see Fig. 5 and Supplementary Table 2 for details of class properties):

- **Experiment:** An experiment comprises a set of procedures that are repeated on different compartments over time. Each repeat of a specific procedure in a compartment, performed with dedicated equipment constitutes a series (as in DICOM-SB).
- **Experimental protocol:** Each experiment should adhere to an experimental protocol, defining the details of how transformations are distributed in different compartments and how different series of measurements are taken from the compartments.
- **Compartment:** The transformations (encoded as module definitions) described above can be grouped in different compartments, according to the cell interactions that need to be tested. Thus, the compartment class can be seen as a container (e.g. a vessel or a cuvette), where an experiment is performed. When working with automated platforms, it is common to use plates that arrange their wells as a matrix of rows and columns, where each well is assigned different series. In such a scenario each well is represented by a compartment that enables tracking the series location. The term 'compartment' was chosen to be consistent with the modelling standard SBML (where it is defined as a bounded space in which the species are located).
- **Series:** Each series references the raw data generated by the equipment after a particular run. In synthetic biology the raw data are often organised as a list of data arrays. Here an array represents each of the different magnitudes/channels measured by the experimental equipment (e.g. time, temperature, fluorescence intensity, optical density etc.) within its corresponding values. The series class is defined by a timestamp. This logs the following: execution date and time, a list of channel descriptors (e.g. 'optical density', 'green fluorescence', a list of channel types with the corresponding channel data types (e.g. 'integer', 'float', 'short'), and a URI pointing to a dataset that contains all the data related to the series. In SynBIS, all the series referencing referring to raw data, measured from a compartment for an experiment, point to a DICOM-SB object.
- **Equipment:** This class identifies the specific equipment performing the measurements, whether it is microscopy, flow cytometry and so on. Supplementary Table 2 summarises the equipment properties tracked by SynBIS for plate, the reader and flow cytometer (Please note, these properties are not part of the basic data model, but, rather, an example of the kind of information that can be serialised by this class.).
- **Stimulus** (When a particular series and characterisation experiment requires the interaction with external stimuli.): This class represents the addition of external components (e.g. small molecules) or variations in the experimental conditions (e.g. temperature changes) during the course of the series. More details are given in the next section.

2.6 Defining stimuli

More details on the definition of stimuli are given below (see Fig. 6 and Supplementary Table 3):

- **Stimulus definition:** This describes the basic features defining a stimulus. The property 'type' determines whether the stimulus describes an environmental condition (e.g. temperature) or a chemical species (e.g. AHL). The property 'amountType' determines whether the amounts should be considered absolute

(e.g. temperature, total concentration) or relative (e.g. temperature increase, concentration increase). The property 'amountUnit' shows the stimulus' unit of measure. Finally, when the stimulus is a chemical species, the field 'component' contains a URI to an SBOL standard component definition object, describing the characteristics of the species.

- **Stimulus:** This instantiates a particular stimulus definition for a data series (to represent an abstraction by a concrete instance, i.e. from class to object). The property 'timeStamp' presents the time point when the stimulus is to be introduced in the system, while 'amount' shows the stimulus quantity effected at that time point.

2.7 Defining compartments

More details on the definition of compartments can be found below (see Fig. 7 and Supplementary Table 4):

- A *compartment* can be connected to other compartments – allowing its content to flow from one to another. This is defined by the property 'flowsTo' (which defines a channel that flows the channel content towards another compartment). The specification of each compartment is provided through the property 'definition' by the compartment definition class.
- A *compartment definition* is described by its 'coordinates' (e.g. if part of a plate, A2), 'type' (e.g. 'well in a 96 plate'), 'brand' and 'model'. Each definition can be extended by linking it to a list of sub-compartments, using the property 'compartments'; this allows, for example, a plate to be defined as a compartment whose sub-compartments are its wells.

2.8 Defining experiment groups

When working with automated platforms, it is common to use plates that arrange wells corresponding to different experiments – so they can all be executed in parallel. Consequently, our model is able to track groups of experiments performed in parallel (see Fig. 8 and Supplementary Table 5). An experimental group must contain at least one experiment and may be linked to different compartment definitions – each containing the details of the plate used to perform all the parallel experiments.

2.9 Defining datasheets

Once the details (data and metadata) of the characterisation experiment have been obtained and an analysis protocol undertaken, it is then possible to generate datasheets (see Fig. 9 and Supplementary Table 6). In our approach, a datasheet involves the results and ancillary information that comprises a qualitative and quantitative study (the characterisation experiment), followed by a particular analysis protocol (see the classes component, transformation and protocol definition in Fig. 9).

It is important that the input data for the experiments are represented in the datasheet – so that the user community can verify the validity of the analysis, or even be able to run a different analysis with the same raw data. There are two types of experiments presented in our datasheets:

- (i) The experiment studying the transformation and component subject of the datasheet, referenced by the property 'experiment'.
- (ii) The control experiments (negative/medium/empty wells) used to normalise the data from the main experiment, referenced by the property 'controlExperiment'.

The analysis protocol produces a set of processed data series, calculated from the raw data series linked to the input experiments (main and controls) after normalisation, analysis and curation. The set of processed data series is used to

generate tables of metrics that constitute the main content of the datasheet:

- *Series*: The series generated as output of an analysis protocol for a datasheet (hence for the processed series) does not differ significantly from those addressing raw data in an experiment. In both cases series are related to a piece of equipment and can also be related with a set of stimuli. Normally, raw data series point to DICOM-SB objects to encode data (improving the storage efficiency of potentially big files) while processed series normally point to text-based data representations such as JSON or CSV – since they are normally much smaller in size.
- *Metrics table*: Data arising from several series and series groups can be used to generate different metrics (e.g. RPU), doubling time and growth rate) as properties of the metrics table. The metrics may change, depending on the type of biopart under analysis, equipment used and so on.

2.10 Defining protocols

Our datasheet model reserves some classes to define the different types of protocols that may be specified within a particular datasheet and characterisation experiment. The main classes here are protocol and protocol definition. These are designed using a similar motif to component and component definition (see top of Fig. 10):

- *Protocol definition*: This class provides a container that summarises the main properties to be tracked within a protocol. When the protocol comprises multiple steps, the property ‘subprotocol’ can be used to reference each step instantiated in a protocol class.
- *Protocol*: This represents an instance of a protocol step. The property ‘stepNumber’ determines the position of the step in a sequence. The specification of a protocol step is described in the field ‘definition’, linking it to another protocol definition class.

Depending on the step in the characterisation experiment that is to be tracked, different types of protocols may be of interest (see bottom of Fig. 10):

- *Assembly protocol*: When defining a transformation, the DNA components may be annotated with this class to track details about the assembly protocol used.
- *Transformation protocol*: This class may be used when defining a transformation to annotate the tracking details of the transformation protocol used.
- *Analysis protocol*: This may be used to track details about the analysis protocol followed to generate the datasheets. It may be annotated with URIs to the model class (SBOL standard), which serves as a placeholder for an external computational model.
- *Experimental protocol*: This should be used to track all the details corresponding to the experimental protocol used during a characterisation experiment. The property ‘equipment’ may be used to annotate the different pieces of equipment (through the class equipment) used in a protocol. The experimental medium can also be annotated through the property ‘medium’ and the class medium (which can also be annotated with the list of its components through the property ‘ingredients’ and the class ingredient).

2.11 SynBIS API

SynBIS is a web-based synthetic biology information system developed in the Kitney Lab at Imperial College. Readers interested in the complete list of subprotocols and fields that are tracked by SynBIS in our experimental protocol can check Supplementary Tables 7–9 (The details in Supplementary Tables 8 and 9 are not part of our proposed datasheet model. They have been included as an example of the kind of information that can

be serialised using our model.). The data model presented in this paper has been used to standardise the software interface to SynBIS. SynBIS is a repository built to disseminate the results of its biopart characterisation pipelines and characterisation protocols at other centres. SynBIS’ web interface [31] allows users to search its registry by attributes such as part type, name, sequence and its input–output function. Since the influence of the context (chassis, medium, plasmid, reporter, and assay protocol and experiments settings) on parts behaviour is relatively poorly understood, SynBIS hosts multiple datasheets (one per context) for the same part.

Taking the example of a constitutive promoter (J23108), the link <http://synbis.bg.ic.ac.uk/synbisapi02/datasheet/J23108> takes the user directly to the entire machine-readable dataset. However, alternatively, a part (here J23108) can be navigated to by the following steps via the datasheet route: (i) to go <http://synbis.bg.ic.ac.uk> (which takes the user to the homepage; click, proceed to SynBIS; then click ‘Constitutive Promoter’; then search SynBIS and find J23108. The user is then in the homepage for J23108. Clicking ‘Download’ takes the user to either the pdf of the datasheet or the full machine-readable SBOL version of the original data. It is a strategic question whether to view a particular part from the starting point of the raw data or from a datasheet. SynBIS currently starts from the datasheet but has a straightforward route to the original data.

2.12 Implementing the SynBIS API

SynBIS datasheets comprise several sections: a summary page (see Fig. 11, left) that contains the biopart description and main results; a description of the experimental context (see Fig. 11, right), including constructs; transformation, instrument settings and so on. There are also modality-specific sections, detailing the characterisation results based on the data modality used (plate reader data and flow cytometry in most cases).

The advantages of our data model are that (i) it provides not only multi-page datasheets, but also access to all the raw data from the characterisation experiments; and (ii) the datasheets encompass data, metadata and ancillary information. To this end, the data model, designed under DICOM-SB, is compatible with SBOL. Consequently, all the properties and classes presented here not belonging to the DICOM-SB standard have been designed, either as annotations and nested annotations on existing SBOL classes, or annotating a generic top level class for each of the new classes requiring to be serialised at the top level of an SBOL document. All the new property and class names of our model are linked to a new namespace with prefix ‘synbis’ and URL ‘<http://synbis.bg.ic.ac.uk>’.

The server-side logic of SynBIS was originally implemented in Java. It was therefore relatively straightforward to implement a set of RESTful web services that took advantage of the libSBOLj library developed by the SBOL community. Currently there are two types of functionalities available in the API:

- *Direct datasheet invocation*: A GET service (GET requests are read-only http requests used to retrieve resource representation/information) that allows the retrieval of any datasheet providing its “displayId” property is shown at the end of the URL. See examples below:

<http://synbis.bg.ic.ac.uk/synbisapi02/datasheet/<displayId>>
<http://synbis.bg.ic.ac.uk/synbisapi02/datasheet/J23108>

- *Datasheet search*: A GET service that implements a search of datasheets matching the following list of input parameters:

- lowrpu: lower range of RPU, e.g. 0.2.
- highrpu: lower range of RPU, e.g. 2.8.
- partsequence: DNA sequence of the part, e.g. ‘gctagc’.
- rnaspecies: RNAP species, e.g. ‘RNAP’.
- rnapsigmafactor: RNAP sigma factor, e.g. ‘RNAP’.
- species: host species, e.g. ‘*E. coli*’.
- partname: name of the biopart in the database, e.g. ‘J23101’.

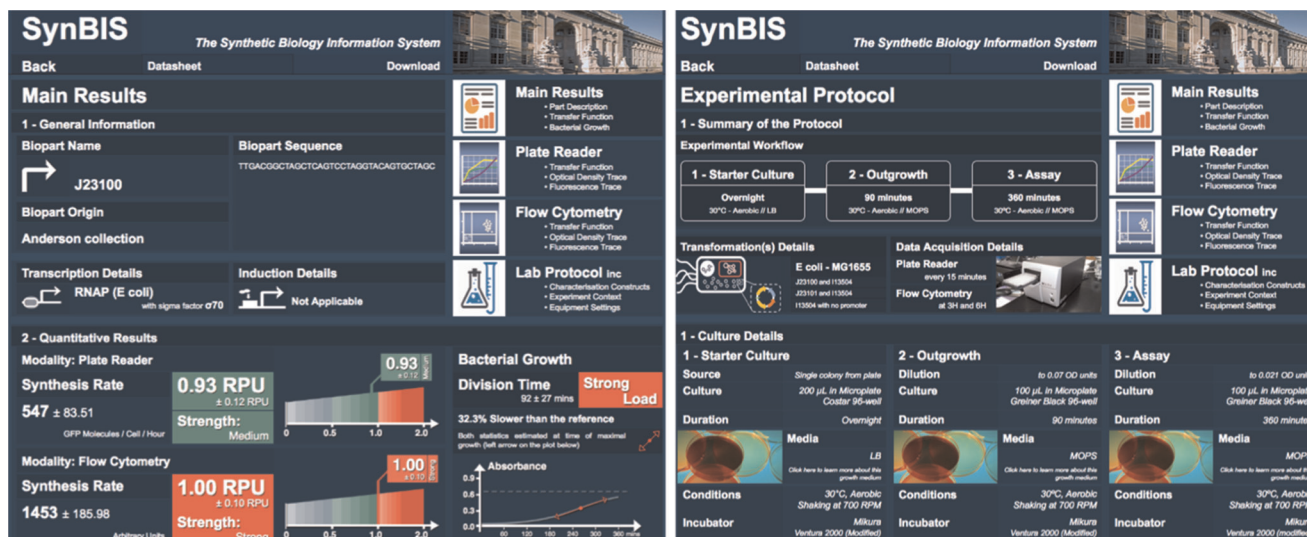


Fig. 11 Sections of the datasheet for the constitutive promoter J23100 (left, main results; right, description of the assay protocol)

See the examples below:

<http://synbis.bg.ic.ac.uk/synbisapi02/search/datasheet?<paramID1>=<value1>&<paramID2>=<value2>&...>

<http://synbis.bg.ic.ac.uk/synbisapi02/search/datasheet?lowrpu=0.2&highrpu=2.8>

The response to these calls can be either an empty SBOL document (if no results found by any of the services), or an SBOL document serialising one or more datasheets. Readers are invited to use a web-browser to test the API with different parameters and observe the results.

3 Discussion and conclusions

The development and publication of a working version of the DICOM-SB standard was the first milestone towards the standardisation of characterisation experiments for bioparts [57] – providing a unified format to encode the raw data arising from data acquisition. Continuing with the aim of standardising our characterisation pipeline and SynBIS, the next logical step was to develop a standard data model for the representation and dissemination of biopart datasheets, which is the subject of this paper.

The choice of SBOL as the language to encode our model was as a result of careful consideration. Many synthetic biology software tools already rely on SBOL as their standard representation to exchange genetic constructs [66]. Therefore, a datasheet extension was considered to be a natural way to enhance SBOL encoded bioparts with extra qualitative information relating to their performance.

The existence of a research community actively working on SBOL development has also been a key aspect of our work. As an example, the development of the SynBIS API encoding the data model uses the libSBOLj Java library [67] to produce SBOL serialisations. In the next version of SynBIS we are planning to use sboljs [68] within our SynBIS front-end so that it can directly access the data from our new SBOL API without parsing.

The data model presented in this paper is based on practical requirements (the extension of previous works support for common workflows and openness). These requirements were also carefully considered. As a result, the following conditions apply:

- E-datasheets and visualisation of its content become uncoupled. Human readable datasheets (in pdf or more interactive in html) can be directly generated from the content of the electronic version; multiple templates can also be applied, depending on the case and users.

- E-datasheets are not bound to a single application (since they link to all the data collected during a characterisation exercise and order them logically). For instance, the raw data can be used for systems biology exercises – such as the development of a minimal cell model [51, 52] or an analysis of cell burden [69]. CAD software can use the extracted parameters, as is, or reanalyse the data according to the models they use. As for the experimental protocols (and associated metadata), they can be reused for other projects in other setups and locations, as they have provenance.

- E-datasheets help foster the adoption of good practice. Allowing access to all the data enables other users to verify the analysis and test their own methods against that used originally. Similarly, this approach – including a comprehensive description of the experimental setup and the protocols – encourages good practice (in terms of reproducibility) and enables comparisons at different geographical locations (e.g. at different laboratories). Also, by forcing the collection of so much data, it is hoped that characterisation teams will start analysing their data, in order to improve their processes.

Using the data model to develop the APIs for our repository has proved to be a valuable practical test for the usefulness of SynBIS with real data. The approach has also provided valuable insights into what areas to improve next – namely, the experimental and analysis protocols.

As part of the current API, we provide a static view of the main properties of the experimental protocol – followed by the characterisation pipeline. However, it is clear that the data currently returned to SynBIS are not comprehensive enough (as a resource) to reproduce a characterisation experiment without additional information (this is currently provided via a link to a human-readable text document). Our objective for the next version is, therefore, to upgrade to a more comprehensive protocol description to enable full reproducibility. This will include not only static properties, but, also, the flow processes (as proposed in Gupta *et al.* [70]) and the ability to build on Antha [71] and Autoprotocol [72]. Once this is achieved, the next step will be to include assembly and transformation protocol details in the SynBIS datasheets.

We are currently building on the work described in this paper to develop a standard specification for analysis protocols. This will facilitate the input of standard experimental data (e.g. encoded as DICOM-SB), together with a standard description of the experimental protocol – to output a unique version of a datasheet – with full reproducibility and platform independence. The fact that SBOL already deals with the concept of modelling (used in SBML) and that the datasheet enhancement will bring the concept

of compartment (also key in SBML), may well facilitate the design of such a data model as an SBML enhancement – building on the work presented by Roehner and Myers [73].

Finally, it is important to note that a significant amount of work remains, in terms of the data model for datasheets. There is a need for further work on the standardisation of content. It can be argued that with developments on the data model front, it is now even more important that guidelines and good practices are developed and adopted within the synthetic biology community. This is particularly important with the increasing use of foundries and the industrial translation of synthetic biology projects.

Clearly, an important aspect of a datasheet is the encoding of biologically relevant information. In the current data model, the information resulting from a data processing step is encoded in a metric (A metric can be as simple as a number, e.g. an RPU, or an array – for a one-dimensional transfer function.). As shown in Arkin's discussion [38], there is a clear need for the standardisation of metrics. The list of metrics that can be used to describe the behaviour of a biological component is, typically, extensive, and varies greatly depending on the type of component. In addition, some metrics are ambiguous. Such an effort is beyond the scope of this paper and, in our view, should be led by a panel of experts – and agreed by the rest of the synthetic biology community. However, it is important that this should be done with machine readability in mind. Again, in our view, the list of metrics should be maintained and supported by a large organisation such as NIST [74] and/or the BSI [75]. Each metric should have a unique identification number and be unambiguously encoded. One example of this is SNOMED [76], in medicine. For example, a metric would have at least one input (with the option of more), and an output. The input and output would need to be a real number. Units would be defined (with an option for arbitrary units).

Another area where guidelines should be developed is the description of the experimental context. Many variables in an experiment and the protocols are known to influence the measured outputs. For example, the carbon source, the media, temperature, strain [36], the reporter gene and plasmid [77]. These are all known to impact the transcript synthesis rate for a specific promoter – e.g. by affecting the growth rate [18, 78] or the activity of the polymerase. In an ideal world everything that is likely to have an influence on the output should be captured and reported; however, this is currently unrealistic. Consequently, guidelines should feature a realistic list of metrics (e.g. several if a grading system is to be introduced). For example, a datasheet should describe:

- the constructs used,
- the assembly methods used to build them,
- the chassis is and integration methods,
- the experimental conditions, including the medium and assay protocol,
- all the data acquisition details, including instrument settings.

Most of this information would be captured in one of the commonly used data formats. Where this is not possible (e.g. protocol description or assembly record, currently), the current datasheet model offers a way to encode the information. The SynBIS API is already being used by other institutions; for example, the University of Cambridge (in their SynBioMine search engine) and the National University of Singapore (Professor Poh's Lab). We plan to continue expanding the use of the API and establishing further collaborations.

4 Methods

The work presented in this paper has been developed using two software environments:

- *Visual Paradigm* [79] has been used to model the UML description of our data model. All the UML diagrams presented in the paper have been produced by this tool.

- The SynBIS API implementing our data model has been developed as a RESTful Web Service under Java EE 7 [80]. The generation of the code has been supported by the libSBOLj Java library [67].

5 Associated content

A detailed description of UML classes presented in the figures is available free of charge via the Internet at <http://pubs.acs.org>. Several examples of datasheet serialisations using the data model presented by this paper can be downloaded using the SynBIS API. Documentation, tools and updates about the model will be made available through the SynBIS website [31].

6 Acknowledgment

The authors acknowledge the invaluable feedback provided by Anil Wipat and Göksel Misirli from the University of Newcastle. The authors also acknowledge the support provided for synthetic biology research by the Engineering and Physical Science Research Council [EP/J02175X/1] and the European Commission funded 7th Framework Program [FP7-KBBE 289326].

7 References

- Clarke, L., Adams, J., Sutton, P., *et al.*: 'A synthetic biology roadmap for the UK' (Research Councils UK, 2012), pp. 1–35
- Kitney, R., Calvert, J., Challis, R., *et al.*: 'Synthetic biology: scope, applications and implications' (The Royal Academy of Engineering, 2009)
- Twist Bioscience. <https://www.twistbioscience.com> (accessed 12 October 2016)
- DNA2.0. <https://www.dna20.com> (accessed 12 October 2016)
- Knight, T.: 'Idempotent Vector Design for Standard Assembly of Biobricks', 2003
- Shetty, R.P., Endy, D., Knight, T.F.: 'Engineering BioBrick vectors from BioBrick parts', *J. Biol. Eng.*, 2008, **2**, p. 5
- Engler, C., Kandzia, R., Marillonnet, S.: 'A one pot, one step, precision cloning method with high throughput capability', *PLoS One*, 2008, **3**, p. e3647
- Engler, C., Marillonnet, S.: 'Generation of families of construct variants using golden gate shuffling', in Lu, C., Browse, J., Wallis, J.G. (Eds.): 'cDNA libraries' (Humana Press, 2011), pp. 167–181
- Anderson, J.C., Dueber, J.E., Leguia, M., *et al.*: 'Bglbricks: a flexible standard for biological part assembly', *J. Biol. Eng.*, 2010, **4**, p. 1
- Gibson, D.G.: 'Chapter fifteen – enzymatic assembly of overlapping DNA fragments', in Voigt, C. (Eds.): 'Methods in enzymology' (Academic Press, 2011), pp. 349–361
- Gibson, D.G., Young, L., Chuang, R.-Y., *et al.*: 'Enzymatic assembly of DNA molecules up to several hundred kilobases', *Nat. Methods*, 2009, **6**, pp. 343–345
- Trubitsyna, M., Michlewski, G., Cai, Y., *et al.*: 'Paperclip: rapid multi-part DNA assembly from existing libraries', *Nucleic Acids Res.*, 2014, **42**, p. e154
- Casini, A., MacDonald, J.T., Jonghe, J.D., *et al.*: 'One-pot DNA construction for synthetic biology: the modular overlap-directed assembly with linkers (MODAL) strategy', *Nucleic Acids Res.*, 2014, **42**, pp. e7–e7
- Transcript: The Robotic Cloud Laboratory
- Kitney, R., Freemont, P.: 'Synthetic biology – the state of play', *FEBS Lett.*, 2012, **586**, pp. 2029–2036
- Kahl, L.J., Endy, D.: 'A survey of enabling technologies in synthetic biology', *J. Biol. Eng.*, 2013, **7**, p. 13
- Kaplan, S., Bren, A., Zaslaver, A., *et al.*: 'Diverse two-dimensional input functions control bacterial sugar genes', *Mol. Cell*, 2008, **29**, pp. 786–792
- Zaslaver, A., Kaplan, S., Bren, A., *et al.*: 'Invariant distribution of promoter activities in *Escherichia coli*', *PLoS Comput. Biol.*, 2009, **5**, p. e1000545
- Gardner, T.: 'A swan in the making', *Science*, 2014, **345**, pp. 855–855
- Eisenstein, M.: 'Living factories of the future', *Nature*, 2016, **531**, pp. 401–403
- Chambers, S., Kitney, R., Freemont, P.: 'The foundry: the DNA synthesis and construction foundry at imperial college', *Biochem. Soc. Trans.*, 2016, **44**, pp. 687–688
- Cherry, J.: 'Amyris synthetic biology: purposeful, predictable, profitable'. 37th Symp. on Biotechnology for Fuels and Chemicals, 2015
- Shetty, R.: 'Developing plant-inspired cultured aromas using a foundry for organism engineering'. 2016 SIMB Annual Meeting and Exhibition, 2016
- Linshiz, G., Jensen, E., Stawski, N., *et al.*: 'End-to-end automated microfluidic platform for synthetic biology: from design to functional analysis', *J. Biol. Eng.*, 2016, **10**, p. 3
- Registry of Standard Biological Parts. <http://parts.igem.org> (accessed 29 October 2015)
- Herscovitch, M., Perkins, E., Baltus, A., *et al.*: 'Addgene provides an open forum for plasmid sharing', *Nat. Biotechnol.*, 2012, **30**, pp. 316–317
- Martinez-Garcia, E., Aparicio, T., Goñi-Moreno, A., *et al.*: 'SEVA 2.0: an update of the standard european vector architecture for de/re-construction of bacterial functionalities', *Nucleic Acids Res.*, 2015, **43**, pp. D1183–D1189
- Virtual Parts Repository. <http://sbol.ncl.ac.uk:8081> (accessed 29 October 2015)
- The SBOL Stack. <https://github.com/ICO2S/sbolstack> (accessed 29 January 2017)

- 30 Ham, T.S., Dmytriv, Z., Plahar, H., *et al.*: 'Design, implementation and practice of JBEL-ICE: an open source biological part registry platform and tools', *Nucleic Acids Res.*, 2012, **40**, p. e141
- 31 Synthetic Biology Information System (SynBIS). <http://synbis.bg.ic.ac.uk> (accessed 29 October 2015). On-line repository of biopart-datasheets
- 32 Huynh, L., Tagkopoulos, I.: 'A parts database with consensus parameter estimation for synthetic circuit design', *ACS Synth. Biol.*, 2016, **5**, pp. 1412–1420
- 33 Nielsen, A.A.K., Der, B.S., Shin, J., *et al.*: 'Genetic circuit design automation', *Science*, 2016, **352**, DOI: 10.1126/science.aac7341
- 34 Cardinale, S., Joachimiak, M.P., Arkin, A.P.: 'Effects of genetic variation on the *E. coli* host-circuit interface', *Cell Rep.*, 2013, **4**, pp. 231–237
- 35 Goodger, D., van Rossum, G.: 'PEP 257– Docstring Conventions', 2001. <https://www.python.org/dev/peps/pep-0257> (accessed 22 February 2017)
- 36 van Rossum, G., Warsaw, B., Coghlan, N.: 'PEP 8–style guide for python code', 2001. <http://www.python.org/dev/peps/pep-0008> (accessed 22 February 2017)
- 37 Canton, B., Labno, A., Endy, D.: 'Refinement and standardization of synthetic biological parts and devices', *Nat. Biotechnol.*, 2008, **26**, pp. 787–793
- 38 Arkin, A.: 'Setting the standard in synthetic biology', *Nat. Biotechnol.*, 2008, **26**, pp. 771–774
- 39 Lee, T.S., Krupa, R.A., Zhang, F., *et al.*: 'Bglbrick vectors and datasheets: a synthetic biology platform for gene expression', *J. Biol. Eng.*, 2011, **5**, p. 12
- 40 Appleton, E., Tao, J., Wheatley, F.C., *et al.*: 'OWL: electronic datasheet generator', *ACS Synth. Biol.*, 2014, **3**, pp. 966–968
- 41 Bhatia, S., Densmore, D.: 'Pigeon: a design visualizer for synthetic biology', *ACS Synth. Biol.*, 2013, **2**, pp. 348–350
- 42 Appleton, E., Tao, J., Haddock, T., *et al.*: 'Interactive assembly algorithms for molecular cloning', *Nat. Methods*, 2014, **11**, pp. 657–662
- 43 Instruments, T.: 'The TTL logic data book' (Texas Instruments, 1988)
- 44 Cheng, A.A., Lu, T.K.: 'Synthetic biology: an emerging engineering discipline', *Annu. Rev. Biomed. Eng.*, 2012, **14**, pp. 155–178
- 45 Isaacs, F.J., Dwyer, D.J., Ding, C., *et al.*: 'Engineered riboregulators enable post-transcriptional control of gene expression', *Nat. Biotechnol.*, 2004, **22**, pp. 841–847
- 46 Win, M.N., Liang, J.C., Smolke, C.D.: 'Frameworks for programming biological function through RNA parts and devices', *Chem. Biol.*, 2009, **16**, pp. 298–310
- 47 Anderson, J.C., Voigt, C.A., Arkin, A.P.: 'Environmental signal integration by a modular AND gate', *Mol. Syst. Biol.*, 2007, **3**, p. 133
- 48 Guet, C.C., Elowitz, M.B., Hsing, W., *et al.*: 'Combinatorial synthesis of genetic networks', *Science*, 2002, **296**, pp. 1466–1470
- 49 Rinaudo, K., Bleris, L., Maddamsetti, R., *et al.*: 'A universal RNAi-based logic evaluator that operates in mammalian cells', *Nat. Biotechnol.*, 2007, **25**, pp. 795–801
- 50 Win, M.N., Smolke, C.D.: 'Higher-Order cellular information processing with synthetic RNA devices', *Science*, 2008, **322**, pp. 456–460
- 51 Weiße, A.: 'Quantifying host-circuit interactions with a mechanistic chassis model'. Synthetic Biology: Engineering, Evolution, and Design Conf. 2015, SEED 2015, 2015, pp. 40–49
- 52 Weiße, A.Y., Oyarzún, D.A., Danos, V., *et al.*: 'Mechanistic links between cellular trade-offs, gene expression, and growth'. *Proc. Natl. Acad. Sci.*, 2015, **112**, pp. E1038–E1047
- 53 Chandran, D., Bergmann, F.T., Sauro, H.M.: 'Tinkercell: modular CAD tool for synthetic biology', *J. Biol. Eng.*, 2009, **3**, p. 19
- 54 Funahashi, A., Matsuoka, Y., Jouraku, A., *et al.*: 'CellDesigner 3.5: a versatile modeling tool for biochemical networks'. *Proc. IEEE*, 2008, **96**, pp. 1254–1265
- 55 Myers, C.J., Barker, N., Jones, K., *et al.*: 'Ibosisim: a tool for the analysis and design of genetic circuits', *Bioinformatics*, 2009, **25**, pp. 2848–2849
- 56 Kelly, J.R., Rubin, A.J., Davis, J.H., *et al.*: 'Measuring the activity of BioBrick promoters using an in vivo reference standard', *J. Biol. Eng.*, 2009, **3**, p. 4
- 57 Sainz de Murieta, I., Bultelle, M., Kitney, R.I.: 'Toward the first data acquisition standard in synthetic biology', *ACS Synth. Biol.*, 2016, **5**, pp. 817–826
- 58 NEMA PS3/ISO 12052: Digital imaging and communications in medicine (DICOM) standard, National Electrical Manufacturers Association, Rosslyn, VA, USA
- 59 Benson, D.A., Cavanaugh, M., Clark, K., *et al.*: 'Genbank. *Nucleic Acids Res.*, 2013, **41**, pp. D36–D42
- 60 Bartley, B., Beal, J., Clancy, K., *et al.*: 'Synthetic biology open language (SBOL) version 2.0.0', *J. Integr. Bioinform.*, 2015, **12**, p. 27
- 61 Quinn, J., Beal, J., Bhatia, S., *et al.*: 'Synthetic Biology Open Language Visual (SBOL Visual), version 1.0.0, 2013
- 62 Benchling. <https://benchling.com> (accessed 15 March 2017)
- 63 Hillson, N.: 'J5 DNA assembly design automation'. in (Valla, S., Lale, R. (Eds.): 'DNA cloning and assembly methods' (Humana Press, 2014), pp. 245–269
- 64 Beal, J., Cox, R.S., Grünberg, R., *et al.*: 'Synthetic biology open language (SBOL) version 2.1.0', *J. Integr. Bioinform.*, 2016, **13**, p. 291
- 65 Fowler, M.: 'UML distilled: a brief guide to the standard object modeling language' (Addison-Wesley Professional, 2004)
- 66 Galdzicki, M., Clancy, K.P., Oberortner, E., *et al.*: 'The synthetic biology open language (SBOL) provides a community standard for communicating designs in synthetic biology', *Nat. Biotechnol.*, 2014, **32**, pp. 545–550
- 67 Zhang, Z., Nguyen, T., Roehner, N., *et al.*: 'libSBOLj 2.0: a java library to support SBOL 2.0', *IEEE Life Sci. Lett.*, 2015, **1**, pp. 34–37
- 68 A Javascript library for SBOL: sboljs. <http://sbolstandard.org/software/libsbol/sboljs> (accessed 11 October 2016)
- 69 Ceroni, F., Algar, R., Stan, G.-B., *et al.*: 'Quantifying cellular capacity identifies gene expression designs with reduced burden', *Nat. Methods*, 2015, **12**, pp. 415–418
- 70 Gupta, V., Irimia, J., Pau, I., *et al.*: 'Bioblocks: programming protocols in biology made easier', *ACS Synth. Biol.*, 2017, **6**, (7), pp. 1230–1232
- 71 Antha. <https://www.antha-lang.org/about/index.html> (accessed 15 October 2016)
- 72 Autoprotocol. <http://autoprotocol.org> (accessed 15 October 2016)
- 73 Roehner, N., Myers, C.J.: 'A methodology to annotate systems biology markup language models with the synthetic biology open language', *ACS Synth. Biol.*, 2014, **3**, pp. 57–66
- 74 NIST|National Institute of Standards and Technology. NIST, <https://www.nist.gov/> (accessed September 2017)
- 75 BSI Group | Standards, Training, Testing, Assessment and Certification, BSI, <https://www.bsigroup.com/en-GB/> (accessed September 2017)
- 76 Spackman, K.A., Campbell, K.E., Côté, R.A.: 'SNOMED RT: a reference terminology for health care'. *Proc. AMIA Annual Fall Symp.*, 1997, pp. 640–644
- 77 Kittleston, J.T., Cheung, S., Anderson, J.C.: 'Rapid optimization of gene dosage in *E. coli* using DIAL strains', *J. Biol. Eng.*, 2011, **5**, p. 10
- 78 Klumpp, S., Zhang, Z., Hwa, T.: 'Growth rate-dependent global effects on gene expression in bacteria', *Cell*, 2009, **139**, pp. 1366–1375
- 79 Visual Paradigm. <https://www.visual-paradigm.com> (accessed 12 October 2016)
- 80 The Java™ API for RESTful Web Services. <https://jcp.org/en/jsr/detail?id=311> (accessed 29 October 2015)