Constructing synthetic biology workflows in the cloud

Göksel Misirli1, Curtis Madsen2, İnaki Sainz de Murieta2, Matthieu Bultellet2, Keith Flanagan1, Matthew Pocock4, Jennifer Hallinan5, James Alastair McLaughlin1, Justin Clark-Casey6, Mike Lyne6, Gos Micklem6, Guy-Bart Stan3, Richard Kitney3, Anil Wipat1

1School of Computing Science, Newcastle University, Newcastle upon Tyne, UK
2Electrical & Computer Engineering Department, Boston University, Boston, USA
3Department of Bioengineering, Imperial College London, London, UK
4Turing Ate My Hamster Ltd, Tyne and Wear, UK
5School of Biological Sciences, Macquarie University, Sydney, Australia
6Department of Genetics, Cambridge Systems Biology Centre, University of Cambridge, Cambridge, UK

E-mail: anil.wipat@ncl.ac.uk

Abstract: The synthetic biology design process has traditionally been heavily dependent upon manual searching, acquisition and integration of existing biological data. A large amount of such data is already available from Internet-based resources, but data exchange between these resources is often undertaken manually. Automating the communication between different resources can be done by the generation of computational workflows to achieve complex tasks that cannot be carried out easily or efficiently by a single resource. Computational workflows involve the passage of data from one resource, or process, to another in a distributed computing environment. In a typical bioinformatics workflow, the predefined order in which processes are invoked in a synchronous fashion and are described in a workflow definition document. However, in synthetic biology the diversity of resources and manufacturing tasks required favour a more flexible model for process execution. Here, the authors present the Protocol for Linking External Nodes (POLEN), a Cloud-based system that facilitates synthetic biology design workflows that operate asynchronously. Messages are used to notify POLEN resources of events in real time, and to log historical events such as the availability of new data, enabling networks of cooperation. POLEN can be used to coordinate the integration of different synthetic biology resources, to ensure consistency of information across distributed repositories through added support for data standards, and ultimately to facilitate the synthetic biology life cycle for designing and implementing biological systems.

1 Introduction

To date, the different steps in the design of an engineered biological system are largely carried out manually by a human expert. Whilst automation is becoming more commonplace for building these designs, as the complexity of biological designs increases, automation is becoming increasingly valuable for producing for the designs themselves.

The resources required for the synthetic biology design process are increasingly using the Internet for communication [1, 2]. Communication between these resources can be carried out using workflows which can then coordinate tasks across geographic, industrial or scientific institutions allowing complex interactions between academics, commercial providers, and experimental facilities. Workflows can therefore be used for the automation of biological system design and implementation, through the integration of different tools and datasets which operate together to carry out a particular task that would be too complex for a single tool in isolation.

The development of workflow technology is an ongoing research effort in bioinformatics where stand-alone bioinformatics tools are often brought together to achieve complex goals. Tools such as Taverna [3] provide a platform to link up different tools. Online registries such as MyExperiment [4] can be used to identify tools that can directly be incorporated into a workflow. In addition, workflow engines based on the Business Process Execution Language (BPEL) are used in a variety of applications [5]. However, the development of workflows in synthetic biology is challenging, involving numerous steps and requiring data exchange between a larger range of tools and data sources with different aims and formats, often involving researchers with different areas of expertise, frequently in different locations [reviewed in 6, 7]. Moreover, the need to carry out wet experimental work can lead to longer delays in workflow completion where they are included as processes. There is also a variety of design tools and repositories of parts which would be desirable to include in synthetic biology workflows [8–13].

Workflows systems for synthetic biology have been described previously [6]. In these systems, participating tools are typically tightly coupled and may require synchronous communication from upstream processes to achieve a particular goal. In a design automation workflow, Beal et al. [14] developed a platform to generate biological networks from high-level specifications. The execution of several tools was orchestrated in a controlled manner. One of the first promising and reusable workflow platforms in synthetic biology is Cloths [15]. Cloths can be configured to work with different types of tools. Data exchange between a client tool and a Cloths server can be asynchronous, allowing tools to continue working locally, whilst waiting for data. Cloths comes with an object model, and handles the retrieval and persistence of data from remote repositories. However, this approach requires that users initially develop custom data schemas and converters in order for Clotho to access remote data.

Communication between diverse hardware and software tools requires the exchange of data. To allow data exchange in a workflow, computational tools need to be enabled with
a programmatic interface that allows programmatic access. This interface allows data to be automatically passed from one tool or resource to another. A defined programmatic interface allows diverse software systems to connect directly to a resource, enabling their use in conjunction without human intervention. REST-based Web services are particularly popular for providing programmatic access, since they are simple, lightweight and easy to access computationally [16]. When Web services output data in standard formats the Web services can act as the endpoints of databases providing uniform access to data. Often different processes in a workflow will require different data formats and types. One of the limitations of many workflows is the need to provide a data conversion process to convert between data formats to allow processes to receive data in a format they can work with. However, data format conversion can result in data loss and requires considerable development time.

Much of this data conversion effort could be avoided with availability of a common data format. For synthetic biology the Synthetic Biology Open Language (SBOL) is very useful in this respect. SBOL has been developed to exchange information about biological designs and their components [17–19]. This language provides a standardised format for designs, which is ideal for passing data between processes in a workflow. Application-specific data can be embedded in SBOL documents in the form of annotations, enabling the exchange of information that is not captured explicitly by the SBOL data model. Other standards that are useful for synthetic biologists include DICOM-SB, which adds features such as raw measurement data [20, 21], the Systems Biology Markup Language (SBML) [22], CellML [23] and Kappa [24], both of which allow the easy representation and exchange of dynamic models of biological parts.

In addition to the exchange of data, the order in which the different resources and tools are invoked in a workflow also needs to be carefully managed, since it is this order that will ultimately determine the overall task. Many business workflows, and bioinformatics tools such as Taverna [3] utilise specific languages, such as SCUFL2, for specifying the order of execution of processes in a workflow. Whilst this allows carefully defined workflows to be developed for specific tasks, new workflows need to be specified as new tasks emerge. Moreover, these systems are often synchronous, requiring other processes to put their execution on hold whilst waiting for the completion of a previous task in the workflow. For synthetic biology, with a plethora of different tools and resources that need to be linked for design, implementation and testing, often between academia and industry, a more asynchronous model of process execution is desirable.

Here, we present the Protocol for Linking External Nodes (POLEN) system for coordinating workflows of synthetic biology databases and tools, utilising synthetic biology standards and Web services. This system is Cloud-based, and uses a push–pull, client–server architecture. A single server manages and distributes messages from registered clients to all clients in the system, and operates in a secure, sandboxed, environment. Each client is associated with an Internet-based synthetic biology resource. The messages are used to coordinate behaviour within the set of clients to orchestrate the order in which processes are carried out. Messages also carry details of where data can be sourced by an interested resource in the workflow (Fig. 1). The system offers several advantages over existing workflow systems as it is tailored for synthetic biology. The server maintains a log of all messages that have been received and a client that is newly added to the system can catch up with the history of workflow execution. The messaging passing nature allows asynchronous operation, tolerating delays whilst laboratory experiments are carried out. The client system is freely available and easily incorporated into synthetic biology software applications. The client provides the necessary interface to allow the resource providers application to send and receive messages. Also included in the client are data conversion tools that allow applications using the POLEN client to convert their existing data formats into SBOL, to allow data to be passed in a standard format between resources in the workflow. We demonstrate here how POLEN can help to automate the integration of synthetic biology resources into workflows, promote standardisation in the exchange of data, and ultimately contribute to the automation of the synthetic biology design process across physical, geographic and institutional boundaries.

### 2 Automated and distributed workflows in synthetic biology

POLEN automates data flows by implementing a subscribable, distributed message-based system that propagates events across the Internet. For instance, data from one database can be automatically stored in a different resource elsewhere in a different format, or can be used to initiate a set of tasks in another resource without human interaction. CAD or design automation tools tracking updates from the databases can access newly available data in appropriate standardised formats and remain consistent and synchronised despite being distributed.

POLEN facilitates the construction of computational workflows by broadcasting messages about newly available data. Some POLEN-enabled resources act as data providers and publish messages to notify other databases, CAD or design automation tools. Typically a Uniform Resource Identifier (URI) is embedded in a POLEN message to indicate the endpoint from which the data that the message relates to, may be sourced. Messages in the POLEN system are simple objects. Each message has a type and includes a URI. These URIs commonly point to repositories from whence data can be accessed using a transfer protocol such as HTTP. Data access and conversion are the responsibility of registered repositories and tools, using the information provided in the messages. The client provides conversion utilities which allow GenBank and FASTA files to be passed as SBOL.

Messages pertaining to specific topics can be published or received. An example of some of the topics available in POLEN

![Fig. 1](https://example.com/fig1.png)  
**Fig. 1** Overview of the POLEN system. POLEN allows for the notification of repositories and design tools using light-weight messages that point to actual data in standard formats. These messages are also associated with particular topics, such as Part and Model
are Part, Datasheet, Model, Provenance and Repository; however, this list can be extended by the user to enable complex message-based workflows. When new parts become available in a repository, other repositories are notified by the publication of a POLEN message with the Part topic. New characterisation data are published using the Datasheet topic, while models of biological parts are published using the Model topic. The Provenance topic is used to indicate additional information about parts, such as where the data originated and whether new data about experiments exist. Finally, the Repository topic is used to register repositories with the POLEN system. Ideally, messages can be published or retrieved by repositories and can only be retrieved by external tools. Messages can include additional fields, such as names, descriptions and a list of properties stored in JSON objects. These properties can be used to store application-specific data.

2.1 Model-driven design of biological systems using POLEN

Here, we describe an example of the use of POLEN. In this scenario, POLEN was used to create and parameterise models of biological parts. This example illustrates how POLEN can be used to help the application of model-driven design for engineered biological systems.

Models of engineered biological systems are typically structured and parameterised through the analysis of experimental data. In synthetic biology applications smaller models can be built for a variety of component parts, e.g. promoters, and then the models composed to produce a full model of the final device or system. One limitation of this approach is the lack of good experimental data for many parts. Therefore, when parts are characterised or re-characterised, it is valuable to analyse the resulting experimental data and incorporate parameters derived from these new measurements into new or existing part models. Ultimately, this process can enhance the reliability of computer simulations for systems.

Part datasheet repository also detects the new Part message and triggers a request for promoter characterisation data. If the part is already characterised, SynBIS notifies the VPR using a new Datasheet message. If the part is not characterised, SynBIS starts an internal characterisation workflow, which runs independently of the POLEN system. SynBIS submits a new message when the characterisation data are available. This message includes the URI of another SBOL document, including the genetic description of the part. This document is annotated with a URI pointing to a datasheet document and another URI pointing to experimental raw data in the form of a DICOM-SB document. The new Datasheet message is detected by the VPR, which then fetches the datasheet using the datasheet URI and creates a dynamic model of the promoter. As a result, the VPR publishes a new Model message for other tools that listen for such messages. In this scenario, all of the resources already use

Fig. 2 Depiction of the Flowers project POLEN network. POLEN allows the connection of geographically distributed repositories and tools asynchronously using a message-based notification system. In this illustration of a recent use case, three different repositories relevant to synthetic biology design are connected through the POLEN system. External tools can also poll messages to use the content from these repositories.
SBOL as the standard data format for making data available. However, if the POLEN enabled resource does not support SBOL format it can take advantage of the conversion utilities that are built into the API to convert FASTA and GenBank formats to SBOL and make the SBOL data available by specifying the URI that points to the SBOL output endpoint of the client.

Using this POLEN workflow, models for a number of constitutive promoters were created and stored in the VPR. For example, the mdoG promoter derived from the Escherichia coli MG1655 strain was parameterised using the characterisation data. Steady-state values of the reporter protein, plasmid copy number, and cell division and dilution times were used as parameters when calculating the strength of this promoter, and to create the associated SBML model.

As demonstrated in this particular use case, POLEN helps coordinate communication between different repositories. Experimental values, once available, are turned into models, or are used to re-parameterise existing models that can, in turn, be used by design tools to facilitate the construction of more predictable designs. Facilities are also provided to allow data standards such as SBOL to be used for moving data between software in a workflow.

3 Discussion

The POLEN system is a real time, light-weight, and asynchronous workflow platform. POLEN facilitates the creation of complex design workflows by coordinating communication between different repositories and tools, which may move and transform data in an automated fashion. The system also offers facilities to aid users to adopt standards such as SBOL through the provision of standards converters. Currently, FASTA and GenBank to SBOL are supported but we envisage this repertoire will be expanded to include conversion of other emerging standards such as DICOM-SB and SBML as the client is developed. Compared to existing solutions, the aim of this system is to decouple the execution of tasks in different institutions and hence to facilitate the creation of distributed workflows.

This system is already an important part of the Flowers consortium. The consortium is a collaboration between several universities, and has the aim of creating an infrastructure for synthetic biology in the UK. POLEN acts as a secure hub to connect resources developed by different consortium members.

In order to take advantage of POLEN, it is crucial that repositories provide computational access to data in standard formats. A recent version of SBOL allows hierarchical designs of biological systems and the capture of rich sets of biological constraints, such as cis and trans interactions between genetic components. SBOL also allows application specific data to be embedded in genetic descriptions. These developments permit the development of more complex communication scenarios using POLEN. By incorporating SBOL conversion utilities into the POLEN client the system moves towards a situation where a common interface for all resources in a workflow may be envisaged, mediated by the use of a common standard data format.

Federated querying of biological parts is becoming increasingly important as the number of part repositories increases. One repository used to store and fetch data about parts described using SBOL is SynBioHub [13]. Examples include data from the Registry of Standard Biological Parts [11] and SyBioOntKB [26] including Bacillus subtilis specific design data. SynBioHub supports the integration of data from instances installed in different locations. Standard SPARQL queries find and integrate additional data about parts. In the future SynBioHub will be integrated into the POLEN network to help automate the federated querying of part repositories and to provide a generic storage solution for SBOL data in workflows.

Currently, POLEN is only used to schedule data flow in workflows and to aid in data conversion. However, the system is based on the powerful cloud-based tool, Microbase [27]. In the future, notifications could also be used to trigger computer-intensive tasks using Cloud resources such as Amazon Web Services, allowing POLEN registered resources to take advantage of more computational power. Interested parties could then register and be notified once these potentially long-running tasks are completed.

Also, whilst we have set up a single, central POLEN server, it is possible that other consortiums may also wish to setup concurrent, isolated systems. This should be entirely possible given the security features of the system which allows POLEN networks to operate in isolation.

In summary, POLEN is a simple integration platform for data repositories and tools in synthetic biology. The POLEN system facilitates the development of workflows for the design of complex genetic circuits using a message-based system. This system facilitates the implementation of tasks that cannot be achieved by any of the participating resources alone. It also provides access
to Cloud resources, providing the potential to increase the efficiency of design, implementation and characterisation in synthetic biology.

4 POLEN infrastructure

4.1 Cloud infrastructure

POLEN is based on Microbase [27], a Cloud-based platform that can distribute computational access to multiple computers, which can be geographically distributed. Microbase is optimised to utilise Cloud resources when creating computational workflows, especially to parallelise computer-intensive tasks. Moreover, it can be customised to receive and process application-specific inputs, using extensions called Microbase responders. POLEN utilises Microbase by providing its own responder for POLEN specific requests, taking advantage of this distributed system. This POLEN responder uses a notification-based system to push and pull messages, and provides a REST-based Web interface for computational access to these messages. Although only the message storage and retrieval features of Microbase are used in this initial version of POLEN, we aim to extend POLEN in the future to take advantage of the Microbase Cloud platform for distributed processing of tasks that are not feasible using a single machine, for example to provide fast and reliable simulation of models for the synthetic biology community.

4.2 Computational access

The POLEN system uses REST-based Web services to publish and query the published messages. Each message topic is handled by a different endpoint. Data are also exposed using RSS feeds to track recent updates. A Java client API (see Section 4.3) that handles the communication between POLEN and clients is also available. This client also provides access to data converters that produce SBOL. POLEN is secured by an access token system. Tokens are provisioned manually by the service administrators, and then sent by clients using a custom HTTP header with each request. This prevents unauthorised users from reading or publishing messages to the network.

4.3 Availability and usage

The POLEN client system is available at https://polen.ico2s.org. The client API was developed using the Java programming language, and is freely available for download under the Apache License, Version 2.0 at https://github.com/ICO2S/polen. In short, in order to POLEN-enable a software resource a developer can incorporate the client code in their server as a jar package and build the system into their Java code as a library. The resource provider contacts the host of the POLEN server (The Flowers project in this case) and receives the security keys necessary to access the POLEN server and its network of resources. Currently, there is only a single POLEN server, but since each POLEN network is secured and isolated, multiple POLEN systems could be envisaged. The POLEN server system is available on request from the authors. The documentation for the client API and the Web service interface, is available from the URL above.

5 Acknowledgments

G. Misirli, I. S. M., M. B., J.C-C. and M. L. supported by the Engineering and Physical Sciences Research Council (EPSRC) grant EP/J02175X/1; A. W., J. H., G. Micklem, G. B. S. and R. K., in part, supported by the EPSRC grant EP/J02175X/1 and and EP/N031962/1; C. M. supported by the EPSRC grant EP/K539083/1. G. B. S. supported, in part, by the EPSRC grant EP/M002187/1. J.A.M. supported by FUJIFILM DioSynth Biotechnologies.

6 References

1. MacDonald, J.T., Barnes, C., Kitney, R.I., et al.: ‘Computational design approaches and tools for synthetic biology’, Integr. Biol., 2011, 3, (2), pp. 97–108
2. Chandran, D., Copeland, W.B., Sleight, S.C., et al.: ‘Mathematical modeling and synthetic biology’, Drug Discov. Today. Dis. Models, 2008, 5, (4), pp. 299–309
3. Misseri, P., Soiland-Reyes, S., Owen, S., et al.: ‘Taverna, released’, 2010 Proc. 22nd Int. Conf. Scientific and Statistical Database Management, SSDM 2010, Heidelberg, Germany, 30 June–2 July, 2010, pp. 471–481
4. Goble, C.A., Bhagat, J., Alekseev, S., et al.: ‘myExperiment: a repository and social network for the sharing of bioinformatics workflows’, Nucleic Acids Res., 2010, 38, (suppl 2), pp. W677–W682
5. Dalman, T., Wiechert, W., Noh, K.: ‘A scientific workflow framework for 13c metabolic flux analysis’, J. Biotechnol., 2016, 232, pp. 12–24
6. Marchioiu, M.A., Stelling, J.: ‘Computational design tools for synthetic biology’, Curr. Opin. Biotechnol., 2009, 20, (4), pp. 479–485
7. Myers, C.: ‘Platforms for genetic design automation’, Methods Microbiol., 2013, 46, pp. 177–202
8. Li, C., Donizelli, M., Rodriguez, N., et al.: ‘Biomodels database: an enhanced, curated and annotated resource for published quantitative kinetic models’, BMC Syst. Biol., 2010, 4, (1), p. 92
9. Ham, T.S., Dmytriv, Z., Dhawan, H., et al.: ‘Design, implementation and practice of JBEI-ICE: an open source biological part registry platform and tools’, Nucleic Acids Res., 2012, 40, (18), p. e1614
10. Galdzicki, M., Rodrigues, G., Chandran, D., et al.: ‘Standard biological parts knowledgebase’, PLoS One, 2011, 6, (2), e17005
11. Pecoud, J., Blauvelt, M.F., Cai, Y., et al.: ‘Targeted development of registries of biological parts’, PLoS One, 2008, 3, (7), p. e2671
12. Misirli, G., Hallinan, J.S., Wipat, A.: ‘Composable modular models for synthetic biology’, ACM J. Emerg. Technol. Comput. Syst., 2014, 11, (3), pp. 22:1–22:19
13. Madsen, C., McLaughlin, J.A., Misirli, G., et al.: ‘The SBOL stack: a platform for storing, publishing, and sharing synthetic biology designs’, ACS Syst. Biol., 2016, 5, (6), pp. 487–497
14. Beal, J., Weiss, R., Denomme, D., et al.: ‘An end-to-end workflow for engineering of biological networks from high-level specifications’, ACS Syst. Biol., 2012, 1, (8), pp. 317–331
15. Oberortner, E., Denomme, D., Anderson, J.C.: ‘An interactive pattern story on designing the architecture of clotho’. Proc. of the 19th Conf. Pattern Languages of Programs, 2012, pp. 21:1–21:15
16. Pautasso, C., Zimmermann, O., Leymann, F.: ‘Resful web services vs. big’ web services: architectural decision’. Proc. of the 17th Int. Conf. on World Wide Web, 2008, pp. 805–814
17. Galdzicki, M., Wilson, M.L., Rodriguez, C.A., et al.: ‘Synthetic biology open language (SBOL) version 1.1’. (Massachusetts Institute of Technology, 2012).
18. Bartley, B., Beal, J., Clancy, K., et al.: ‘Synthetic biology open language (SBOL) version 2.0’. J. Integr. Bioinform., 2015, 12, (2), doi: 10.2390/biecoll-jib-2015-272
19. Roehner, N., Beal, J., Clancy, K., et al.: ‘Sharing structure and function in biological design with SBOL 1.0’, ACS Syst. Biol., 2016, 5, (6), pp. 498–506
20. de Murieta, I.S., Bultelle, M., Kitney, R.I.: ‘Towards the first data acquisition standard in synthetic biology’, ACS Syst. Biol., 2016, 5, (8), pp. 817–826
21. Kitney, R., Freemont, P.: ‘Synthetic biology – the state of play’, FEBS Lett., 2012, 586, (15), pp. 2029–2036
22. Hucka, M., Finney, A., Sauro, H.M., et al.: ‘The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models’, Bioinformatics, 2003, 19, (4), pp. 524–531
23. Lloyd, C.M., Halstead, M.D.B., Nielsen, P.F.: ‘CellML: its future, present and past’, Prog. Biophys. Mol. Biol., 2004, 85, (2–3), pp. 433–450
24. Danos, V., Laneve, C.: ‘Formal molecular biology’, Theor. Comput. Sci., 2004, 325, (1), pp. 69–110
25. Bultelle, M.A., Samz de Murieta, I., Kitney, R.I.: ‘Introducing SynHIS – the synthetic biology information system’. 2015 Synthetic Biology: Engineering, Evolution & Design Conf., Boston, USA, 2015
26. Misirli, G., Hallinan, J., Pocock, M., et al.: ‘Data integration and mining for synthetic biology design’, ACS Syst. Biol., 2016, 6, (10), pp. 1086–1097
27. Flanagan, K., Nakajima, S., et al.: ‘Microbase: a generic framework for computationally intensive bioinformatics workflows in the cloud’, J. Integr. Bioinform., 2012, 9, (2), p. 212