
Copyright:

This is an open access article published under a Creative Commons Attribution (CC-BY) License, which permits unrestricted use, distribution and reproduction in any medium, provided the author and source are cited.

DOI link to article:

10.1021/acssynbio.6b00286

Date deposited:

26/07/2017
A Visual Language for Protein Design

Robert Sidney Cox, III,*†‡ James Alastair McLaughlin,*‡ Raik Grünberg,*§ Jacob Beal,*∥ Anil Wipat,*‡ and Herbert M. Sauro,*⊥

1Material Science Institute, University of Oregon, Eugene, Oregon 97403, United States
2School of Computing Science, Newcastle University, Newcastle upon Tyne NE1 7RU, U.K.
3Computational Bioscience Research Center, King Abdullah University for Science and Technology, Thuwal 23955, Saudi Arabia
4Raytheon BBN Technologies, Cambridge, Massachusetts 02138, United States
5Department of Bioengineering, University of Washington, Seattle, Washington 98105, United States

Supporting Information

ABSTRACT: As protein engineering becomes more sophisticated, practitioners increasingly need to share diagrams for communicating protein designs. To this end, we present a draft visual language, Protein Language, that describes the high-level architecture of an engineered protein with easy-to-draw glyphs, intended to be compatible with other biological diagram languages such as SBOL Visual and SBGN. Protein Language consists of glyphs for representing important features (e.g., globular domains, recognition and localization sequences, sites of covalent modification, cleavage and catalysis), rules for composing these glyphs to represent complex architectures, and rules constraining the scaling and styling of diagrams. To support Protein Language we have implemented an extensible web-based software diagram tool, Protein Designer, that uses Protein Language in a “drag and drop” interface for visualization and computer-aided-design of engineered proteins, as well as conversion of annotated protein sequences to Protein Language diagrams and figure export. Protein Designer can be accessed at http://biocad.ncl.ac.uk/protein-designer/.

KEYWORDS: synthetic biology, visualization, Synthetic Biology Open Language, genetic circuits, protein engineering

Protein engineering is one of the oldest disciplines of molecular biotechnology, with a rich history of engineering by mutation and fusion of genes coding for functional protein sequences. As more sophisticated and model-driven methods have become available, practitioners need to communicate increasingly complex designs. In other disciplines, such as electrical engineering or architecture and mechanical engineering, standard visual symbols and diagram languages allow engineers to more easily comprehend designs, avoid mistakes, build software tools, etc. No standard visual language has been proposed, however, for the depiction of design features within individual engineered proteins. We address this by presenting a draft visual language for protein design, Protein Language.

Protein Language is specifically intended to aid protein design and not to describe all existing knowledge of protein biology. This approach is in keeping with other visual languages in engineering disciplines: for example, electronics diagrams do not aim to capture the full range of electromagnetic phenomena and architectural diagrams do not aim to describe the full physics of built structures. Accordingly, we have created glyphs focused on a subset of design elements intended to cover many of the most common changes that protein engineers make to manipulate protein function, expression and production. Protein Language has been simultaneously developed with the aim of compatibility with other standards in biological engineering, including the Systems Biology Graphical Notation (SBGN) and the Synthetic Biology Open Language Visual (SBOLv). Thus, Protein Language makes use of design standards from other fields to produce a distinct and clear visual style, while remaining largely compatible with related efforts.

Protein Language provides users with a wide range of expressive capabilities, which can improve communication of protein designs with rapidly drawn, easy to interpret, high-quality technical diagrams. To support use and adoption of Protein Language, we have also implemented a web-based software tool, Protein Designer, that provides an accessible interface for using these symbols to construct diagrams. We plan for Protein Language and its symbols to be adjusted and further refined through the experience of practitioners and an open community standardization process.

RESULTS

Protein Language and Glyph Set. At present, there are 12 glyphs defined for Protein Language: four region glyphs and
eight site glyphs. These glyphs have been chosen to be compatible with existing literature where possible, plus a number of novel symbols intended to be clear, easy to draw, and easy to distinguish. All 12 glyphs are shown in Figure 1 and described in detail in Supporting Appendix A. These glyphs are intended to serve as general categories for design rather than formal ontological definitions. For example, a region containing several transmembrane domains could be represented as several membrane glyphs, as a single structured region glyph, or omitted altogether with the omitted protein region glyph, depending on what a practitioner wishes to communicate regarding that sequence. Together, the 12 glyphs can generate a wide range of conceivable protein designs.

A Protein Language diagram is built around a straight line, a common literature representation of an amino acid chain. Other significant features of the protein’s structure and function are represented by eight site glyphs, representing features from one to 30 amino acids in length. The catalytic glyph represents an enzyme active site or binding pocket. The binding glyph is used to represent protein binding to various ligands including protein, DNA, and small molecules. The cleavage glyph covers proteolytic sites, and the similar degradation glyph includes recognition sites for processive protein degradation machinery and systems such as ubiquitination. Protein modifications by covalent attachment of small molecules are represented by the covalent glyph, covering post-translational modifications such as phosphorylation—a focus of intense research in the proteomic literature (e.g., refs 10, 12). Two localization glyphs allow for the description of C-terminal, N-terminal, or internal sequences for protein transport, allowing protein designs to specify cellular location. Finally, the biochemical tag glyph includes sites for protein purification, crystallization, and other chemical handles. The eight site glyphs thus describe enzyme active sites and locations where a protein is post-translationally modified, cleaved, degraded, bound, transported, or biochemically manipulated.

Protein Designer. Protein Designer is a web-based software tool for creating and manipulating Protein Language diagrams, available at http://biocad.ncl.ac.uk/protein-designer/. A screenshot is shown in Figure 2. (Note: At present, Protein Designer requires the Google Chrome or Chromium desktop browser.) The user can create a protein backbone (arbitrary region) by right clicking on the blank canvas. An unlimited number of resizable backbone lines are supported. The sidebar allows the user to select a glyph from the glyph set, which can then be placed on the canvas or attached to a protein backbone. The structured protein region glyph, in turn, has its own backbone attachment points for adding site glyphs to the top or bottom. Once completed, designs can be exported, using the button located in the top right, into Scalable Vector Graphic (SVG) images. The SVG can also be converted to PDF by the browser’s print dialogue, and either form imported into compatible illustration or presentation software. Protein
Designer’s simple interface allows fast layout of designs using Protein Language.

Protein Designer uses a modular system of drawing rules to render SVG. New glyphs can be defined as geometrical rules using the SVG commands for path drawing: moveto, lineto, and closepath (ref 13 Section 8.3). This allows users the option of contributing new glyphs to the language as SVG geometry definitions, which can be incorporated into the Protein Designer code. The architecture of Protein Designer allows new glyphs and sets of glyphs to be added easily, which we hope will facilitate the development of a standard visual protein language.

**Example A: Protease Sensor.** Figure 3 shows a Protein Language diagram representing a protease-based sensor.14 This protein device consists of regions encoding two colors of fluorescent proteins with a disordered region between them. Inside the disordered region is a protein cleavage site. This sensor exhibits fluorescent resonance energy transfer (FRET) between the two fluorescent protein domains, which is abolished when the protein is cleaved. The FRET signal is enhanced through a noncovalent binding: an intramolecular noncovalent interaction with a Proline-rich peptide motif located close to the protein’s C-terminus. These two binding sites are shown by the noncovalent binding glyph (“b”). A second cleavage site at the C-terminus allows for the cleavage of a biochemical purification tag (hexa histidine, “H”).

**Example B: Light-Inducible Protein Membrane Localization.** Figure 4 shows a Protein Language diagram representing light-inducible protein membrane localization. The immediate next steps we envision for this effort, however, focus on refinement of Protein Language and its integration with existing standards and communities. In particular, we aim to integrate Protein Language with the Systems Biology Graphical Notation (SBGN)5 and Synthetic Biology Open Language Visual (SBOLv)6,7 standards, both of which are free and open standards supported by diverse international communities and part of the COMputational Modeling in Biology NETwork (COMBINE) federated standards collection. Together, SBOLv and SBGN enable canonical depictions of functional pathways, structural features of DNA, and biochemical interactions, but presently neither has a means of depicting the substructure of a protein—a complementary capability provided by Protein Language. Moreover, efforts already underway in both of these communities will facilitate integration with Protein Language: SBGN is being enhanced to support diagram elements that show the substructure of general, reversible system for regulated recruitment in eukaryotes.

**Example C: Inducible Artificial Transcription Factor.** Figure 5 shows a Protein Language diagram representing an inducible artificial transcription factor.16 The estrogen receptor region is used to add an inducible response to an artificial transcription factor. This design brings together three protein regions: the N-terminus encodes a DNA binding domain (“d”, a zinc finger DNA recognition region binding to a specific 9 base-pair DNA sequence); the middle region contains the estrogen receptor, which controls nuclear localization of the entire protein with an inducible response to the hormone beta-estradiol; the C-terminus encodes the activation domain VP16 (“a”), which recruits polymerase to activate a eukaryotic promoter. The nuclear localization is modulated by a retained nuclear localization signal “N” and a retained nuclear export signal “X” where “N” is blocked when bound to Hsp90 and unblocked when bound to estrogen.

**DISCUSSION**

Visual depictions have always been an important tool in the design of biological systems. We have presented the first diagram language for constructing visualizations specifically for purposes of protein engineering. Rather than focusing on protein structure, as in protein ribbon diagrams,17 Protein Language operates at a higher level of functional abstraction. This abstraction to the modular aspects of protein design reflects the increasing sophistication of protein engineering models, allowing the communication between practitioners to focus on the primary functional characteristics of a design and leaving the specific details of its realization to be examined as necessary. As protein engineering capabilities improve, we expect that abstract design diagrams will become increasingly important. Concurrently, as protein engineering capabilities improve, we expect that Protein Language will expand to cover a large range of routinely engineered features.

The immediate next steps we envision for this effort, however, focus on refinement of Protein Language and its integration with existing standards and communities. In particular, we aim to integrate Protein Language with the Systems Biology Graphical Notation (SBGN)5 and Synthetic Biology Open Language Visual (SBOLv)6,7 standards, both of which are free and open standards supported by diverse international communities and part of the COMputational Modeling in Biology NETwork (COMBINE) federated standards collection. Together, SBOLv and SBGN enable canonical depictions of functional pathways, structural features of DNA, and biochemical interactions, but presently neither has a means of depicting the substructure of a protein—a complementary capability provided by Protein Language. Moreover, efforts already underway in both of these communities will facilitate integration with Protein Language: SBGN is being enhanced to support diagram elements that show the substructure of...
The authors thank Steven Schkolne for consultation on glyph design and Matthew Pocock and Christopher Voigt for helpful discussions. J.A.M. is supported by FUJIFILM DioSynth Technologies. A.W. is supported by the Engineering and Physical Sciences Research Council grant EP/J02175X/1 and EP/N031962/1. R.S.C. is supported in part by US DoD grant FA5009-16-P-0041 and National Science Foundation grant DBI-1355909. J.S.B. is supported in part by the National Science Foundation Expeditions in Computing Program Award DBI-1355909. J.S.B. is supported in part by the National Science Foundation Expeditions in Computing Program Award DBI-1355909.

Corresponding Authors
E-mail: jakebeal@ieee.org.
E-mail: anil.wipat@ncl.ac.uk.
E-mail: uw.hsauro@gmail.com.

ORCID
Robert Sidney Cox III: 0000-0002-2009-6236

Notes
The authors declare no competing financial interest.

ACKNOWLEDGMENTS

The authors thank Steven Schkolne for consultation on glyph design and Matthew Pocock and Christopher Voigt for helpful discussions. J.A.M. is supported by FUJIFILM DioSynth Technologies. A.W. is supported by the Engineering and Physical Sciences Research Council grant EP/J02175X/1 and EP/N031962/1. R.S.C. is supported in part by US DoD grant FA5009-16-P-0041 and National Science Foundation grant DBI-1355909. J.S.B. is supported in part by the National Science Foundation Expeditions in Computing Program Award DBI-1355909 as part of the Living Computing Project. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

REFERENCES
