

A Visual Language for Protein Design

Robert Sidney Cox III,^{*,‡} James Alastair McLaughlin,^{*,§} Raik Grünberg,^{*,||} Jacob
Beal,^{*,⊥} Anil Wipat,^{*,§} and Herbert M Sauro^{*,¶}

*Material Science Institute, University of Oregon, USA, School of Computing Science,
Newcastle University, UK, Computational Bioscience Research Center, King Abdullah
University for Science and Technology, KSA, Raytheon BBN Technologies, USA, and
Department of Bioengineering, Seattle, USA*

E-mail: sidney@dna.caltech.edu; j.a.mclaughlin@newcastle.ac.uk; raik.gruenberg@gmail.com;
jakebeal@ieee.org; anil.wipat@ncl.ac.uk; uw.hsauro@gmail.com

Running header

A Visual Language for Protein Design

*To whom correspondence should be addressed

‡University of Oregon

§Newcastle University

||KAUST

⊥Raytheon BBN Technologies

¶University of Washington

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 a few easy-to-draw glyphs, intended to be compatible with other biological diagram languages such as SBOL 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

Introduction

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 (1, 2) or architecture and mechanical engineering (3, 4), standard visual symbols

and diagram languages allow engineers to more easily comprehend designs, avoid mistakes, build software tools, etc. No standard visual language has previously existed, 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) (5) and the Synthetic Biology Open Language Visual (SBOLv) (6, 7). 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 twelve 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 twelve glyphs are shown in Figure 1 and described in detail in 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 twelve 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 are then represented by “region” glyphs ordered along this backbone and “site” glyphs that are ordered along a region. The backbone line represents an arbitrary protein region, with unspecified structural properties. Unstructured and linker regions are normally shown in this way as backbone line. A rectangle with rounded edges describes a structured protein region, such as a protein domain, consistent with typical conventions from the literature on protein domains (e.g., (8, 9)). The width of the rectangle may be scaled to indicate relative region size. Membrane regions are shown with a zig-zag line, inspired by several literature illustrations (10, 11). This membrane glyph can be used on either the backbone or the structured region glyph. We also include a dotted line to describe a region that is present in the protein but omitted from the diagram. These four region glyph types describe variably sized protein regions, are consistent with previous literature descriptions, and allow the user to highlight the basic structure of globular domains, disordered regions, and membrane regions.

Smaller significant features of an protein's structure and function are represented by eight site glyphs, typically representing features from one to thirty amino acids in size. 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., (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, binded, 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 Fig. 2.¹ The user can create a protein backbone (unspecified 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 (13)) images. The SVG can also be converted to PDF by the browser's print dialog, and either form imported into compatible illustration or

¹Note: At present, Protein Designer requires the Google Chrome or Chromium desktop browser.

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* ((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 presented in (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 non-covalent binding: an intramolecular “helper interaction.” Other features include synthetic linker sequences and a biochemical purification tag.

Example B: Light-inducible protein membrane localization

Figure 4 shows a Protein Language diagram representing light-inducible protein membrane localization presented in (15). This engineered system consists of two separate protein backbones that can be brought together via a light-induced conformational change that reversibly controls protein binding. Two fluorescent reporter domains (mCitrine and mCherry) are used to image the localization of each protein to the cell plasma membrane, where one of the proteins is anchored by a membrane region. The system can be used as a general, reversible system for regulated recruitment to the plasma membrane in eukaryotes.

Example C: Inducible artificial transcription factor

Figure 5 shows a Protein Language diagram representing an inducible artificial transcription factor presented in (16). The estrogen receptor region is used to add inducible response to an artificial transcription factor. This design incorporates three structured protein regions: a DNA binding domain, the estrogen receptor, and a eukaryotic activation domain. Each domain's function is described by site glyphs for binding and localization.

Discussion

Visual depictions have always been an important tool in the design of biological systems. In this paper, 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 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 only if necessary. As protein engineering capabilities improve, we expect that such 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 sub-structure 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 sub-structure of chemical species using other visual languages, and SBOLv is being enhanced to support the standardized depiction of non-nucleic-acid components. As there are a number of minor differences in how Protein Language is currently formulated and the rules of these standards, integration will involve a number of refinements and adjustments. Given the positive reception that Protein Language has received in initial community discussions, however, we have confidence that it will ultimately form the basis for a broadly accepted, community-supported open standard that helps to effectively integrate engineered proteins into the design of biological systems.

Acknowledgement

The authors thank Steven Schkolne for consultation on glyph design and Matthew Pocock 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 FA5209-16-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 #1522074 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.

Supporting Information Available

Supporting Information 1: Glyph images in SVG format; Supporting Information 2: Glyph images in PNG format; Supporting Information 3: Appendices: Details on Protein Glyphs

and Drawing Rules This material is available free of charge via the Internet at <http://pubs.acs.org/>.

References

1. IEEE, IEEE Graphic Symbols for Logic Functions (Includes IEEE Std 91A-1991 Supplement, and IEEE Std 91-1984). IEEE Std. 91a-1991, 1991.
2. IEEE, IEEE Standard American National Standard Canadian Standard Graphic Symbols for Electrical and Electronics Diagrams (Including Reference Designation Letters). IEEE Std. 315-1975 (Reaffirmed 1993), 1993.
3. Schley, M., Buday, R., Sanders, K., and Smith, D. *AIA CAD layer guidelines*; The American Institute of Architects Press: Washington, DC, 1997.
4. British Standards Institution, Collaborative production of architectural, engineering and construction information. BS 1192:2007, 2007.
5. Le Novere, N., Hucka, M., Mi, H., Moodie, S., Schreiber, F., Sorokin, A., Demir, E., Wegner, K., Aladjem, M. I., and Wimalaratne, S. M. (2009) The systems biology graphical notation. *Nature biotechnology* 27, 735–741.
6. Quinn, J. Y. et al. (2015) SBOL Visual: A Graphical Language for Genetic Designs. *PLoS Biol.* 13, e1002310.
7. Quinn, J., Beal, J., Bhatia, S., Cai, P., Chen, J., Clancy, K., Hillson, N., Galdzicki, M., Maheshwari, A., Pocock, M., Rodriguez, C., Stan, G.-B., and Endy, D. *Synthetic Biology Open Language Visual (SBOL Visual), version 1.0.0*; 2013.
8. Chen, C., Nott, T. J., Jin, J., and Pawson, T. (2011) Deciphering arginine methylation: Tudor tells the tale. *Nat. Rev. Mol. Cell Biol.* 12, 629–642.

9. Lai, A., Sato, P. M., and Peisajovich, S. G. (2015) Evolution of synthetic signaling scaffolds by recombination of modular protein domains. *ACS Synth. Biol.* *4*, 714–722.
10. Choudhary, C., and Mann, M. (2010) Decoding signalling networks by mass spectrometry-based proteomics. *Nat. Rev. Mol. Cell Biol.* *11*, 427–439.
11. Lim, W. A. (2010) Designing customized cell signalling circuits. *Nat. Rev. Mol. Cell Biol.* *11*, 393–403.
12. Whitaker, W. R., Davis, S. A., Arkin, A. P., and Dueber, J. E. (2012) Engineering robust control of two-component system phosphotransfer using modular scaffolds. *Proc. Natl. Acad. Sci. U. S. A.* *109*, 18090–18095.
13. Ferraiolo, J., Jun, F., and Jackson, D. *Scalable Vector Graphics (SVG) 1.0 Specification*; iuniverse, 2000.
14. Grünberg, R., Burnier, J. V., Ferrar, T., Beltran-Sastre, V., Stricher, F., van der Sloot, A. M., Garcia-Olivas, R., Mallabiabarrena, A., Sanjuan, X., Zimmermann, T., and Serrano, L. (2013) Engineering of weak helper interactions for high-efficiency FRET probes. *Nat. Methods* *10*, 1021–1027.
15. Levskaya, A., Weiner, O. D., Lim, W. A., and Voigt, C. A. (2009) Spatiotemporal control of cell signalling using a light-switchable protein interaction. *Nature* *461*, 997–1001.
16. McIsaac, R. S., Oakes, B. L., Wang, X., Dummit, K. A., Botstein, D., and Noyes, M. B. (2013) Synthetic gene expression perturbation systems with rapid, tunable, single-gene specificity in yeast. *Nucleic Acids Res.* *41*, e57.
17. Richardson, J. S. (1985) Schematic drawings of protein structures. *Methods in enzymology* *115*, 359–380.

Graphical TOC Entry

unspecified		catalytic		localization (cleaved)	
omitted		binding		localization (retained)	
membrane		cleavage		degradation	
structured		covalent		biochemical tag	

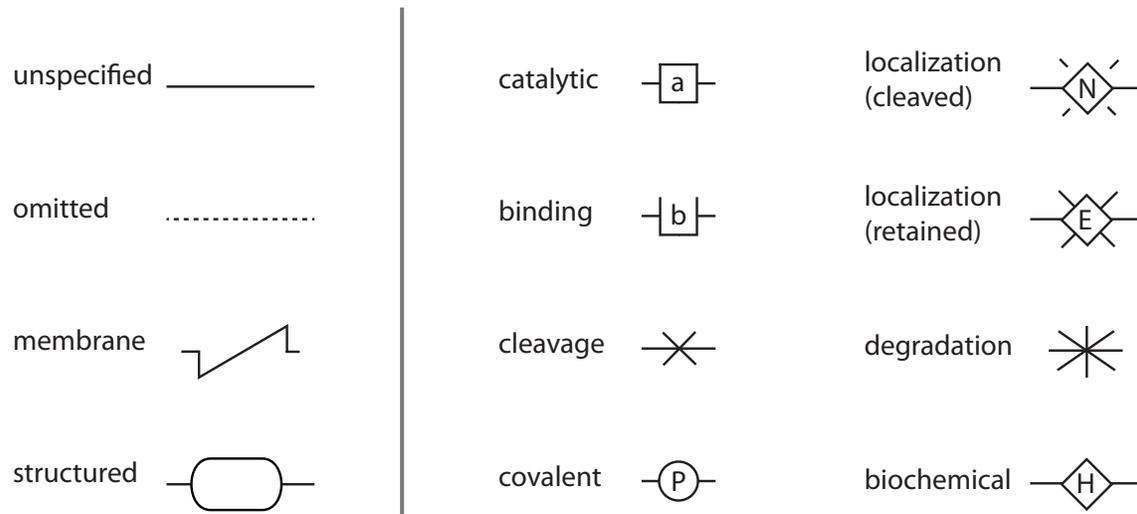


Figure 1: Glyphs defined for Protein Language: four protein region glyphs and eight protein site glyphs.

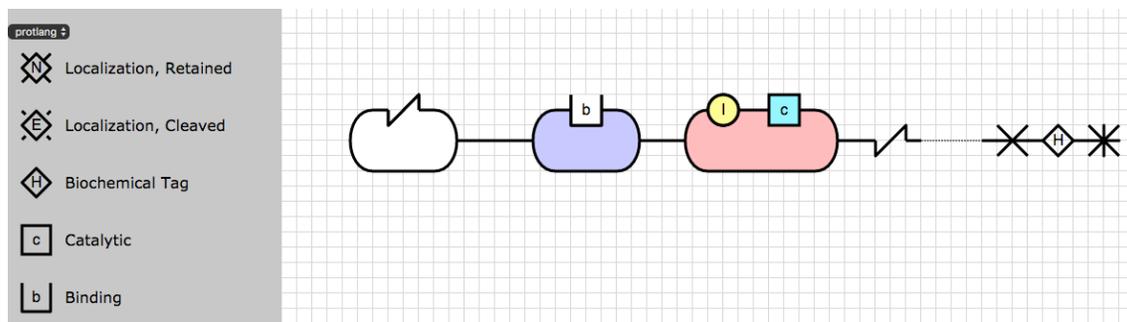


Figure 2: Protein Designer is a web-based software tool for construction of Protein Language diagrams using scalable vector graphics.

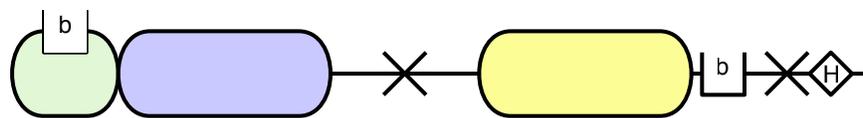


Figure 3: Diagram example: protease FRET sensor combining fluorescent reporter domains (mTurquoise and mCitrine) connected with a disordered region containing a protein cleavage site for caspase 3 protease, which is active at the onset of apoptosis. mTurquoise is shown in blue and mCitrine in yellow. At the N-terminus, the domain containing the peptide binding site $WW\Delta C$ contains a non-covalent contact point to the Proline-rich peptide *wp2* placed at the C-terminus of mCitrine (yellow). These two binding sites are shown by the non-covalent binding glyph ('b'). A second cleavage site at the C-terminus allows for the cleavage of a biochemical purification tag (hexahistidine, 'H').

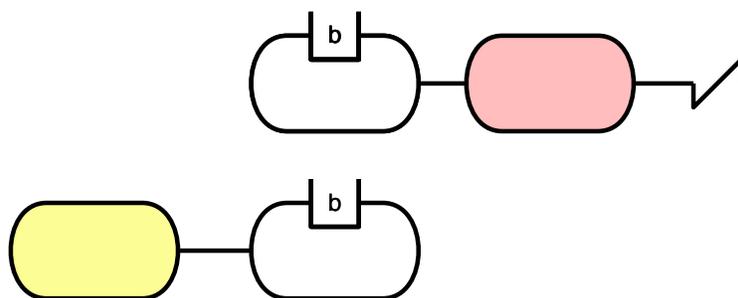


Figure 4: Diagram example: the light-inducible PIF domain is used to create a reporter system for programmed localization of proteins to the plasma membrane. The protein binding domain ('b') is modulated by light reversibly when exposed to red (650 nm) or infra-red (750 nm) light. Each protein backbone also contains a distinct fluorescent reporter protein, yfp (yellow) or rfp (red).

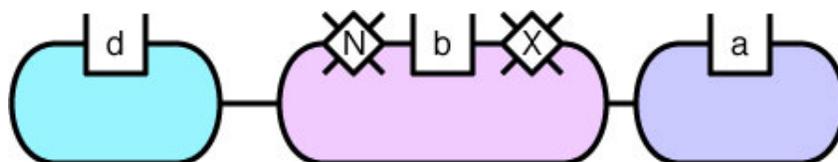


Figure 5: Diagram example: an estrogen receptor region is used to add an inducible response to an artificial transcription factor. This design brings together three protein regions: the N-terminus encodes a DNA binding domain ('d,' a zinc finger DNA recognition region binding to a specific 9 base-pair 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.