Embedding machine-readable proteins interactions data in scientific articles for easy access and retrieval

Paolo Tieri[1,2§], Alberto Termanini[2§], Piero Fariselli[3] and Claudio Franceschi[1,2]

[1]Department of Experimental Pathology, University of Bologna, Via San Giacomo 12, 40126 Bologna, Italy.

[2]"L. Galvani" Interdepartmental Center, University of Bologna, Via San Giacomo 12, 40126 Bologna, Italy.

[3]Biocomputing group, Department of Experimental and Evolutionistic Biology, via San Giacomo 9/2, 40126 Bologna, Italy.

[§]These authors equally contributed to this work.

ABSTRACT

Summary: Extraction of protein-protein interactions data from scientific literature remains a hard, time- and resource-consuming task. This task would be greatly simplified by embedding in the source, i.e. research articles, a standardized, synthetic, machine-readable codification for protein-protein interactions data description, to make the identification and the retrieval of such very valuable information easier, faster, and more reliable than now.

We shortly discuss how this information can be easily encoded and embedded in research papers with the collaboration of authors and scientific publishers, and propose an online demonstrative tool that shows how to help and allow authors for the easy and fast conversion of such valuable biological data into an embeddable, accessible, computer-readable codification.

Availability: www.eypid.org

Contact: p.tieri@unibo.it

INTRODUCTION

The reconstruction of biological systems for computational analyses relies on the existence of data describing components as well as their interactions and relationships (Kersey and Apweiler 2006, Viswanathan et al. 2008). It is well established that understanding the essence of protein interactions is a key factor for the development of systems biology as well as, in perspective, of novel therapeutics (Ewing et al 2007, Viswanathan et al. 2008).

On the other hand, it is also well recognized the difficulty of collecting such kind of information through the many diverse sources available, i.e. scientific literature, databases, and other resource aggregators and tools. Effective information mining from these disparate knowledge repositories and sources poses an actual challenge (Krallinger and Valencia 2005, Kersey and Apweiler 2006, Ewing et al 2007).

Scientific literature explicitly contains protein-protein interactions data, that are evidently presented in the unstructured format of human natural language. Many successful efforts are spent to develop efficient natural language processing (NLP) tools, text-mining algorithms and databases that extract, and possibly store, this information directly from the research literature. Such attempts represent an answer to the critical need to capture and synthetize these results in machine-readable formats, thus allowing for fast retrieval and computational analysis of large datasets. However, seizing relevant information and populating these databases largely requires a manual and/or automatic, resource-consuming process of reading, interpreting and extracting interaction relationships (Temkin and Gilder 2003, Krallinger and Valencia 2005). These efforts also have led to the development of a new field devoted to text-mining and information extraction for molecular biology ('BioNLP', Krallinger and Valencia 2005, Malik et al 2006). In this view, the advancement of *ad hoc* algorithms yielded significant results, even if the reliability of such tools is not yet complete. A gene mention finding evaluation showed balanced precision and recall scores over 80% (Yeh et al 2005). Combined use of different text-mining algorithms for protein-protein interaction data yielded a precision score above 81% (Malik et al 2006). Nonetheless, reconstructing the binary relationship it is far more difficult (Leitner and Valencia 2008). Text-mining is a way to cope with the increasing amount of free textual data. In perspective, another complementary way can be represented by the implementation of biology-specific semantic web methodologies (Berners-Lee and Hendler 2001). Semantic web is an extension of the WWW in which the semantics of information is defined to be understandable to machines (see http://www.w3.org/2001/sw/). This alternative lies in the same strand of the perspective of electronic annotated information (EAI) as recently proposed and reported by several authors (Leitner and Valencia 2008, Ceol et al 2008, Gerstein et al 2007).

EMBEDDING CODE IN THE SOURCE

A possible and feasible solution to data mining issues relies in embedding at the *primary source level*, i.e. directly linked to the reference article, a *standardized, synthetic, machine-readable code* for protein-protein interactions data. In this way the reconstruction of biological systems can be obtained by parsing 'formal languages' instead of natural ones, making the identification and the retrieval of such very valuable information easier, faster, and much more reliable (Aho et al. 2006). Practically, this implies "asking people to make some extra effort, in repayment for which they get major new functionality" (Berners-Lee and Hendler 2001). In this

case, the extra effort would consist in the curation by authors of the "translation" of their own interaction data into a machine-readable code by means of user-friendly online conversion tools. We would suggest that scientific publishers ask directly to authors of research papers for the codification of their own interaction data, to be embedded for example in dedicated online spaces provided in the online paper submission and editing systems, as for example it is already done today for *supplementary data* and *materials* sections (Leitner and Valencia 2008), or in already existing databases. To this scope, it will be necessary to provide the authors with specific software tools for the easy and friendly conversion of their data into the adopted general format. A further future improvement would be a central repository (or a distributed system exploiting DAS queries, www.biodas.org) that automatically updates itself using the supplementary non-ambiguous information.

The advantages of such approach could become noticeable not only in terms of accessing and retrieving the coded information, but also in terms of the care on, and the interpretation of the research data that will be curated directly by the authors themselves.

METHODS

In the Supplementary Materials section and on the website www.eypid.org we describe a simplified and abridged logical model for protein-protein interaction description. An online and purely demonstrative tool –the EYPID web converter- is provided in the website, in which the user inputs protein interactions data and produces as output machine-readable 'extensible markup language' -XML- code lines. In the interface we specify just some of the information fields that could be defined, described and coded for proteins interactions data, exactly as done for instance in other well-known structured diagram editors for drawing gene-regulatory and biochemical networks. Please refer to online Supplementary Materials and to www.eypid.org for further information.

CONCLUSION

There is a stringent need to code protein-protein interactions in a machine-readable format to avoid waste of time and uncertainties, difficulties and typical issues encountered in the retrieval of such data and in the reconstruction of comprehensive interaction maps and pathways based on interaction data. Standard machine-readable formats available at the primary source would greatly help and facilitate such tasks, avoiding at the same time problems and errors deriving from the possible misinterpretation of text-mining algorithms, and reducing the need and the work of human curators (Leitner and Valencia 2008).

The descriptive standards necessary to make embedding and retrieval systems work have partly been developed in the framework of the semantic web approaches and technologies (Berners-Lee and Hendler 2001; Kersey and Apweiler 2006).

We show an online purely demonstrative tool (www.eypid.org/translator.htm), with a simple intuitive interface, that produces an exemplar XML-based code once the proteins interaction data have been input. The output can easily be copied and pasted into a dedicated space in an online submission system provided on the scientific publisher website, or elsewhere, for a relatively easy automated information retrieval.

However, for an effective adoption and to become of a real benefit to the scientific community, a key factor is the endorsement, support and help of scientific publishers. Without this element any kind of efforts in this direction would be very difficult or even impossible.

ACKNOWLEDGEMENTS

We wish to thank Ilaria Cornia for the website layout, Rita Casadio and Gianluca Tasco for useful comments and discussion.

REFERENCES

Aho,A., Ullman,J.D., (2006) *Compilers: Principles, Techniques, and Tools*, Addison Wesley; 2 edition.

Berners-Lee,T., Hendler,J., (2001) Publishing on the semantic web. *Nature* 410, 1023–1024.

Ceol,A.,Chatr-Aryamontri,A.,Licata,L.,Cesareni,G., (2008) Linking entries in protein interaction database to structured text: The FEBS Letters experiment. *FEBS Letters* 582, 1171-1177.

Ewing,R.M., Chu,P., Elisma,F., Li,H., Taylor,P., Climie,S., et al. (2007), Large-scale mapping of human protein–protein interactions by mass spectrometry, *Molecular Systems Biology,* 3, 89.

Gerstein,M., Seringhaus,M., Fields,S., (2007) Structured digital abstract makes text mining easy. *Nature* 447, 142.

Leitner,F., Valencia,A., (2008) A text-mining perspective on the requirements for electronically annotated abstracts. *FEBS Lett.*, 582, 1178-1181.

Kersey,P., Apweiler,R., (2006) Linking publication, gene and protein data. *Nat Cell Biol.* 8, 1183-1189

Krallinger,M., Valencia,A., (2005) Text-mining and information-retrieval services for molecular biology. *Genome Biol.,* 6, 224.

Malik,R., Franke,L., Siebes,A., (2006) Combination of text-mining algorithms increases the performance, *Bioinformatics*, 22, 2151-2517.

Temkin,J.M., Gilder,M.R., (2003) Extraction of protein interaction information from unstructured text using a context-free grammar *Bioinformatics,* 19, 2046-2053.

Viswanathan,G.A., Seto,J., Patil,S., Nudelman,G., Sealfon,S.C., (2008) Getting started in biological pathway construction and analysis. *PLoS Comput Biol.* 4, e16.

Yeh,A., Morgan,A., Colosimo,M., Hirschman,L., (2005) BioCreAtIvE task 1A: gene mention finding evaluation. *BMC Bioinformatics,* 6, S2.

EYPID, Embed Your Protein Interaction Data

Supplementary materials

An user-friendly web converter tool

A possible and feasible solution to data mining issues relies in **embedding at the primary source level, i.e. directly in research papers, a standardized, synthetic, machine-readable code for protein-protein interactions data**. In this way the reconstruction of the biological systems can be obtained by **parsing 'formal language' instead of natural ones**, making the identification and the retrieval of such very valuable information easier, faster, and much more reliable.

Practically, this implies *"asking people to make some extra effort, in repayment for which they get major new functionality"* (Berners-Lee and Hendler 2001). In this case, the extra effort would consist in the curation by authors of the 'translation' of their own interaction data into a machine-readable code by means of user-friendly online translation tools.

We would suggest that scientific publishers ask directly to authors of research papers for the codification of their own interaction data, to be embedded for example in dedicated spaces provided in the online paper submission and editing systems, as it is already done today for supplementary materials sections.

To this scope, it will be necessary to **provide the authors with specific software tools for the easy and friendly translation of their data into the adopted general format**. A further future improvement would be a central repository (or a distributed system exploiting DAS queries) that automatically updates itself using the supplementary non- ambiguous information.

The advantages of such approach could become noticeable not only in terms of accessing and retrieving the coded information, but also in terms of the care on, and the interpretation of the research data that will be curated directly by the authors themselves.

The EYPID web converter is an **online and purely demonstrative tool** in which the user inputs protein interactions data and produces as output machine-readable 'extended markup language' -XML- code lines. We indicate just some of the information that could be defined, described and coded for protein interactions, exactly as done for instance in other well-known structured diagram editors for drawing gene-regulatory and biochemical networks, such as CellDesigner (http://celldesigner.org/).

To the extent of biological networks reconstruction, the work of well renowned and international alliances to standardize graphical notation is well known and accepted among scientists of the field. Just to cite some among the most renowned, implementations of the Systems Biology Graphical Notation (SBGN) and the Edinburgh Pathway Notation (EPN) enable graphical representations of protein interactions and pathways.

Both SBGN and EPN use a logical state-transition representation to describe protein interactions and biological pathways, and fulfil the objective of providing a graphical notation that is useable by biologists and serves as the basis for computational model development.

The information coded in a standard XML format can be easily mapped to the Systems Biology Markup Language (SBML) or in another highly portable world-standard XML-based codification, able to carry the logical content of the biological description and data.

In this framework, it will be greatly facilitated the implementation of tools that, starting from the XML data of the protein-protein interaction stored in the repositories, will give to the user the ability to build large pathways in an easy way.

The proposed and abridged logical model of protein-protein interactions

Here we describe a possible protein-protein interaction model using the Unified Modeling Language (UML).

UML is a standard language for modelling software or non-software systems and has been proven successful in the modelling of large and complex systems. UML uses graphical notations to express the design of the projects. We describe our model of interaction with an UML **Class Diagram** (Fig. 1) and we show an example wit an UML **Object Diagram** (Fig. 2).

Classes and objects are concepts derived from the Object Oriented Programming (OOP), a programming paradigm that uses "objects" and their interactions to design applications and computer programs.

Class (from Wikipedia) defines the abstract characteristics of a thing (object), including the thing's characteristics (its attributes, fields or properties) and the thing's behaviours (the things it can do, or methods, operations or features). One

might say that a class is a blueprint or factory that describes the nature of something. For example, the class Dog would consist of traits shared by all dogs, such as breed and fur colour (characteristics), and the ability to bark and sit (behaviours). Classes provide modularity and structure in an object-oriented computer program. A class should typically be recognizable to a non-programmer familiar with the problem domain, meaning that the characteristics of the class should make sense in context. Also, the code for a class should be relatively self-contained (generally using encapsulation). Collectively, the properties and methods defined by a class are called members.

Object (from Wikipedia) is a pattern (exemplar) of a class. The class of Dog defines all possible dogs by listing the characteristics and behaviours they can have; the object Lassie is one particular dog, with particular versions of the characteristics. A Dog has fur; Lassie has brown-and-white fur.

Class Diagram of the model

In UML, a Class Diagram gives an overview of the system by showing its classes and the relationships among them. The relationships between classes are shown as connecting links. Class diagrams are static diagrams because they display the elements that are interacting, but not what is happening when the interactions occur. Class names begin with a capital letter.

In our model (**Fig. 1**), the interaction between two proteins is described by specifying the "**state**" of the two proteins **before** and **after** the interaction. With "**state**" of a protein we mean all the information that can describe the protein properties, **in particular those that are significant for the interaction and for the protein functions**. For example, if a protein will change its cellular localization after the interaction, this information should be stated.

We choose to define the classes **Protein**, **Residue**, **Interaction**, **DataReferences,** and the relationships among them.

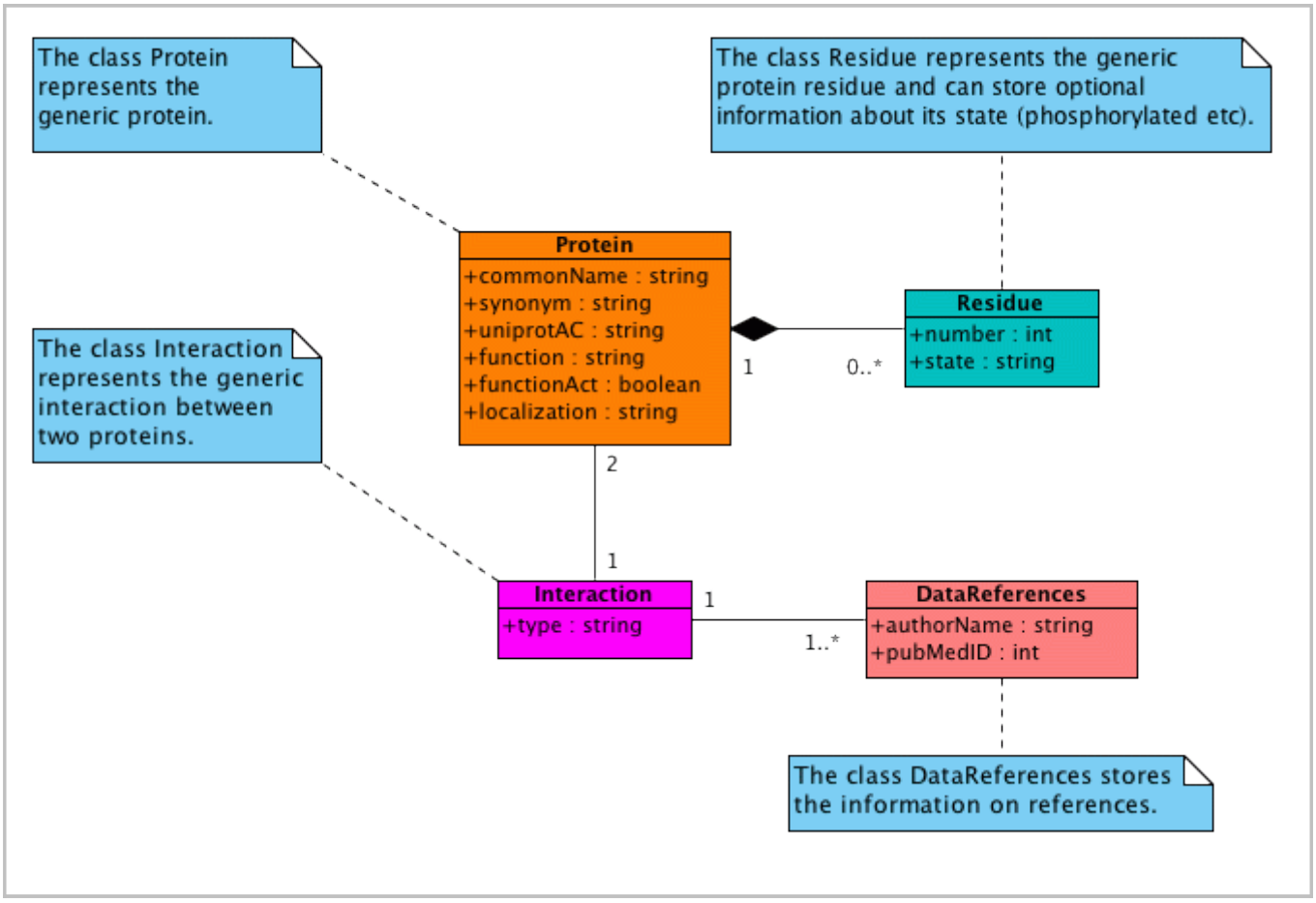

Fig. 1 - Class diagram of the protein-protein interaction model. Further explanation in the tables below.

Class *Protein*

The class Protein represents the generic protein. Note that between the class Protein and the class Residue there is a '**composition**' relationship. In UML, a composition relationship is used when an object is made up of other objects and the whole and parts have coincident lifetimes.

The multiplicity of the relationship is 1 for Protein and (0..*) for Residue. This means that there can be zero, one or more Residue for each Protein. In fact, we don't need to store the information of all the residues of the protein because we would like to define the interaction between two proteins and not the exact residue sequence of the proteins.

Thus, for example, if we know that the interaction takes place only if a certain residue is phosphorylated, we should state only the information about this residue and its status (phosphorylated).

List of attributes:

Name	Type	Multiplicity	Description
commonName	string	1	The common name of the protein.
synonym	string	0..*	Other names of the protein.
uniprotAC	string	1..*	The Accession Number (AC) from the UniProt Database.
function	string	0..*	A function of the protein.
functionAct	boolean	0..*	True if the function described in the field 'function' is active, false otherwise.
localization	string	1..*	One or more protein cellular localizaition.

Table 1 – Description and attributes relative to the Class *Protein*

Class *Residue*

The class Residue represents the generic residue of the protein. As we described in Table 1, there is a **composition** relationship between Residue and Protein.

List of attributes:

Name	Type	Multiplicity	Description
number	numeric	0..*	The number of the residue of the protein.
state	char	1..*	The state of the residue (for example 'P' if phosphorylated, 'U' if ubiquitinated etc.).

Table 2 - Description and attributes relative to the Class *Residue*

Class *Interaction*			
The class Interaction represents the generic interaction between two proteins. Note that between the class Protein and the class Interaction there is an '**association**' relationship. In UML, an association relationship is used when a class must be necessarily associated to another class. The multiplicity of the relationship is 1 for Interaction and 2 for Protein. This indicates that for every object of type Interaction there must be two objects of type Protein. This is because we want to model the interaction that takes place between two proteins. **List of attributes:**			
Name	**Type**	**Multiplicity**	**Description**
type	string	1	The type of the interaction (phosphorylation etc.).

Table 3 - Description and attributes relative to the Class *Interaction*

Class *DataReferences*			
The class DataReferences store the information on references. **List of attributes:**			
Name	**Type**	**Multiplicity**	**Description**
authorName	string	1..*	The names of the authors of the experiment that describe the interaction data.
pubMedID	string	1	The Pub Med ID (PMID) of the paper in which the interaction is described.

Table 4 - Description and attributes relative to the Class *DataReferences*

An example of interaction

We show an example of interaction in the Object Diagram in **Fig. 2**. In UML, a pictorial representation of the instances of classes (i.e. objects) and the relationships between them is called "Object Diagram." It looks similar to a Class Diagram and uses similar notations to denote relationships. The object names are separated from the class names by a "**:**" and are underlined.

In the example we consider as interaction the case of the phosphorylation of the residue Ser177 of a protein "A" by the protein "B", that leads to the activation for a given function of the protein B.

It is important to say that **we store the 'state' of both proteins before and after the interaction**. In the example, for the protein A we have an indicator "copy" of the object 'A' because the interaction doesn't change any property of the protein A, but we have a change of state indicator "become" for the objects 'B' and 'Ser177' because the properties of these objects changes after the interaction.

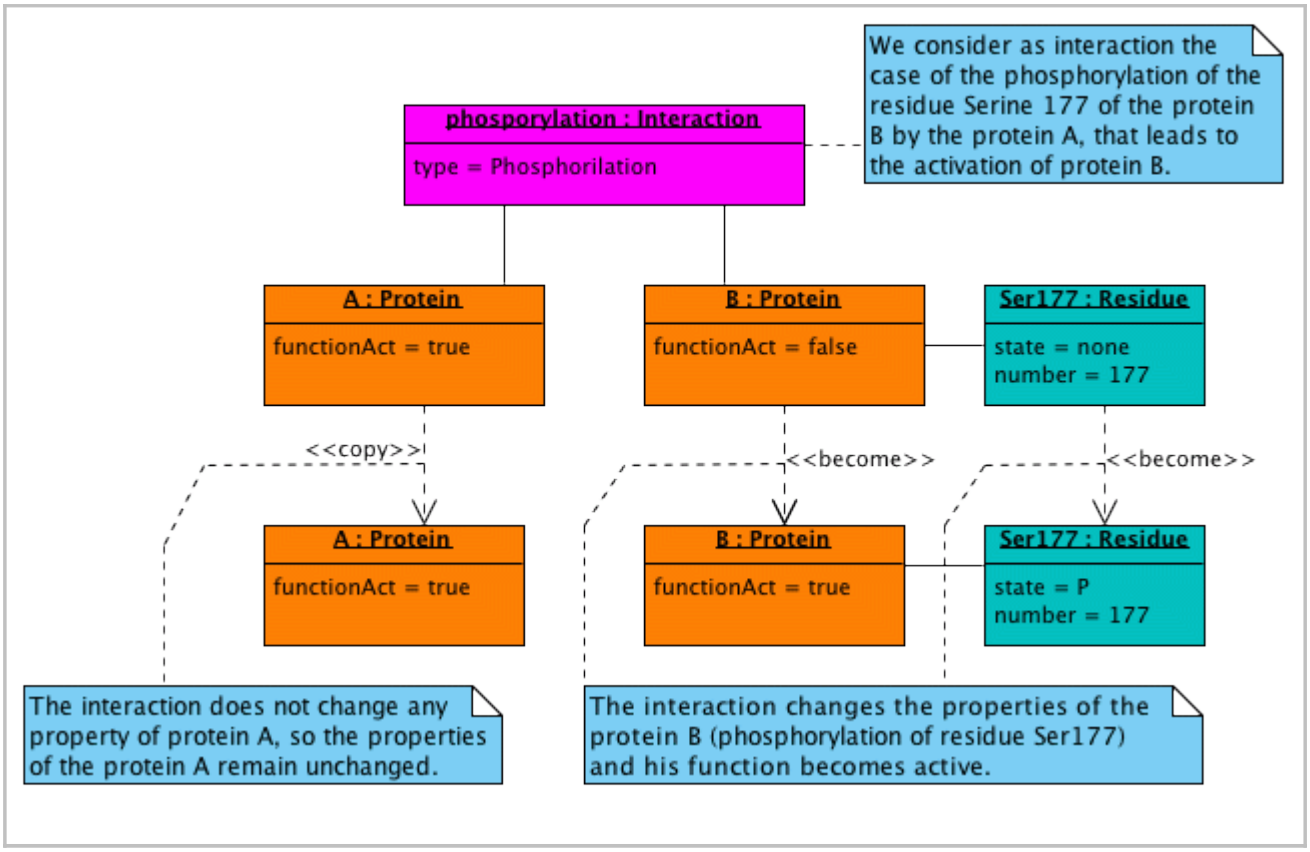

Fig. 2 - Object Diagram of the example of a phosphorylation