Systems Biology Graphical Notation: Entity Relationship Level 1

Draft of April 2, 2009

Disclaimer: This is a working draft of the SBGN Entity Relationship Level 1 specification. It is not a normative document.

To discuss any aspect of SBGN, please send your messages to the mailing list sbgn-discuss@sbgn.org. *To get subscribed to the mailing list or to contact us directly, please write to* sbgn-team@sbgn.org.

Preface

Acknowledgements

Notes on typographical conventions

The concept represented by a glyph is written using a normal font, while a *glyph* means the SBGN visual representation of the concept. Note on the color code: The glyphs that have been thorougly discussed, and are considered frozen, are represented in blue. The glyphs that have been thorougly discussed, but are still posing problems are represented in green. The glyphs that have been proposed but for which in-depth discussion is yet to come are represented in red.

Contents

Chapter 1

What is the Systems Biology Graphical Notation?

The goal of the **S**ystems **B**iology **G**raphical **N**otation (SBGN) is to standardize the graphical/visual representation of essential biochemical and cellular processes studied in systems biology. SBGN defines a comprehensive set of symbols with precise semantics, together with detailed syntactic rules defining their use. It also describes the manner in which such graphical information should be interpreted.

Standardizing graphical notations for describing biological interactions is an important step towards the efficient and accurate transmission of biological knowledge between different communities. Traditionally, diagrams representing interactions among genes and molecules have been drawn in an informal manner, using simple unconstrained shapes and edges such as arrows. Until the development of SBGN, no standard agreed-upon convention existed defining exactly how to draw such diagrams in a way that helps readers interpret them consistently, correctly, and unambiguously. By standardizing the visual notation, SBGN can serve as a bridge between different communities such as computational and experimental biologists, and even more broadly in education, publishing, and more.

For SBGN to be successful, it must satisfy a majority of technical and practical needs, and must be embraced by the community of researchers in biology. With regards to the technical and practical aspects, a successful visual language must meet at least the following goals:

1. Allow the representation of diverse biological objects and interactions;
2. Be semantically and visually unambiguous;
3. Allow implementation in software that can aid the drawing and verification of diagrams;
4. Have semantics that are sufficiently well defined that software tools can convert graphical models into formal models, suitable for analysis if not for simulation;
5. Be unrestricted in use and distribution, so that the entire community can freely use the notation without encumbrance or fear of intellectual property infractions.

This document defines the *Entity Relationship* visual language of SBGN. As explained more fully in Section 1.2, Entity Relationship diagrams are one of three views of a model offered by SBGN. It is the product of many hours of discussion and development by many individuals and groups. In the following sections, we describe the background, motivations, and context of Entity Relationship diagrams.

1.1 History of SBGN development

Although problems surrounding the representation of biological pathways has been discussed for a long time, see for instance [1], the effort to create a well-defined visual notation was pioneered

by Kurt Kohn with his Molecular Interaction Map (MIM), a notation defining symbols and syntax to describe the interactions of molecules [2]. MIM is essentially a variation of the entity-relationship diagrams [3]. Kohn's work was followed by numerous other attempts to define both alternative notations for diagramming cellular processes (e.g., the work of Pirson and colleagues [4], BioD [5], Patika [6, 7], and others), as well as extensions of Kohn's notation (e.g., the Diagrammatic Cell Language of Maimon and Browning [8]).

Kitano originated the idea of having multiple views of the *same* model. This addresses two problems: no single view can satisfy the needs of all users, and a given view can only represent a subset of the semantics necessary to express biological knowledge. Kitano proposed the development of process diagrams, entity-relationship diagrams, timing charts (to describe temporal changes in a system), and abstract flow charts [9]. The Process Diagram notation was the first to be fully defined using a well-delineated set of symbols and syntax [10]. It led to a desire to establish a unified standard for graphical representation of biochemical entities, and from this arose the current SBGN effort. Separately and roughly concurrently, other groups designed similar notations, for example the Edinburgh Pathway Notation [11] or Patika [6, 7]. All of these efforts began to attract attention as more emphasis in biological research was placed on networks of interactions and not just characterization of individual entities.

In 2005, thanks to funding from the Japanese agency *The New Energy and Industrial Technology Development Organization* (NEDO, `http://www.nedo.go.jp/`), Kitano initiated the Systems Biology Graphical Notation (SBGN) project as a community effort. The first SBGN workshop was held in February 2006 in Tokyo, with over 30 participants from major organizations interested in this effort. From the in-depth discussions held during that meeting emerged a set of decisions that are the basis of the current SBGN specification. These decisions are:

- SBGN should be made up of two different visual grammars, describing Entity Relationship and Process Diagram diagrams (called *State Transition* diagrams at the time). See Section 1.2.
- In order to promote wide acceptance, the initial version(s) of SBGN should stick to at most a few dozens symbols that non-specialists could easily learn.

The second SBGN workshop was held in October, 2006, in Yokohama, Japan. This meeting featured the first technical discussions about which symbols to include in SBGN Level 1, as well as discussions about the syntax, semantics, and layout of graphs. A follow-up technical meeting was held in March, 2007, in Heidelberg, Germany; the participants of that meeting fleshed out most of the design of SBGN. The third SBGN workshop, held in Long Beach in October, 2007, was dedicated to reaching agreement on the final outstanding issues of notation and syntax. The participants of that meeting collectively realized that a third language would be necessary: the Activity Flow diagrams. The specification for the Process Diagram language was finalized and largely completed during a follow-up technical meeting held in Okinawa, Japan, in January, 2008. At this last meeting, attendees also held the first in-depth discussions about the syntax of the Entity Relationship language.

The specification for SBGN Process Diagram Level 1 was publicly released on August 23$^{\text{rd}}$ 2008 during the ICSB in Göteborg [12].

SBGN workshops are an opportunity for public discussions about SBGN, allowing interested persons to learn more about SBGN and help identify needs and issues. More meetings are expected to be held in the future, long after this specification document has been issued.

1.2 The three languages of SBGN

Readers may well wonder, why are there *three* languages in SBGN? The reason is that this approach solves a problem that was found insurmountable any other way: attempting to include all relevant facets of a biological system in a single diagram causes the diagram to become hopelessly complicated and incomprehensible to human readers.

The three different notations in SBGN correspond to three different *views* of the same model. These views are representations of different classes of information, as follows:

1. *Process Diagram*: the causal sequences of molecular processes and their results
2. *Entity Relationship*: the interactions between entities irrespective of sequence
3. *Activity Flow*: the flux of information going from one entity to another

In the Process Diagram view, each node in the diagram represents a given *state* of a species, and therefore a given species may appear multiple times in the same diagram if it represents the same entity in different states. Conversely, in the Entity Relationship view, a given species appears only once in a diagram. Process Diagrams are suitable for following the temporal aspects of interactions, and are easy to understand. The drawback of the Process Diagram, however, is that because the same entity appears multiple times in one diagram, it is difficult to understand which interactions actually exist for the entity. Conversely, Entity Relationship diagrams are suitable for understanding relationships involving each molecule, but the temporal course of events is difficult or impossible to follow because Entity Relationship diagrams do not describe the sequence of events.

Process Diagrams can quickly become very complex. Moreover, when diagramming a biochemical network, one often wants to ignore the biochemical basis underlying the action of one entity on the activity of another. A common desire is to represent only the flow of activity between nodes, without representing the transitions in the states of the nodes. This is the motivation for the creation of the Activity Flow view. Activity Flow diagrams permit the use of *modulation*, *stimulation* and *inhibition* and allow them to point to State/Entity nodes rather than process nodes. The Activity Flow view is thus a hybrid between Process Diagram and Entity Relationship diagrams. It is particularly convenient for representing the effect of perturbations, whether genetic or environmental in nature.

A recurring argument in SBGN development is that these these three types of diagrams should be merged into one. Unfortunately, each view has such different meanings that merging them would compromise the robustness of the representation and destroy the mathematical integrity of the notation system. While having three different notations makes the overall system more complex, much of the complexity and increase in burden on learning is mitigated by reusing most of the same symbols in all three notations. It is primarily the syntax and semantics that change between the different views, reflecting fundamental differences in the underlying mathematics of what is being described.

1.3 SBGN levels

It was clear at the outset of SBGN development that it would be impossible to design a perfect and complete notation right from the beginning. Apart from the prescience this would require (which, sadly, none of the authors possess), it also would likely require a vast language that most newcomers would shun as being too complex. Thus, the SBGN community followed an idea used in the development of the Systems Biology Markup Language (SBML; [13]): stratify language development into levels.

A *level* of SBGN represents a set of features deemed to fit together cohesively, constituting a usable set of functionality that the user community agrees is sufficient for a reasonable set of tasks and goals. Capabilities and features that cannot be agreed upon and are judged insufficiently critical to require inclusion in a given level, are postponed to a higher level. In this way, SBGN development is envisioned to proceed in stages, with each higher SBGN level adding richness compared to the levels below it.

1.4 Developments, discussions, and notifications of updates

The SBGN website (http://sbgn.org) is a portal for all things related to SBGN. It provides a web forum interface to the SBGN discussion list (sbgn-discuss@sbgn.org) and information about how anyone may subscribe to it. The easiest and best way to get involved in SBGN discussions is to join the mailing list and participate.

Face-to-face meetings of the SBGN community are announced on the website as well as the mailing list. Although no set schedule currently exists for workshops and other meetings, we envision holding at least one public workshop per year. As with other similar efforts, the workshops are likely to be held as satellite workshops of larger conferences, enabling attendees to use their international travel time and money more efficiently.

Notifications of updates to the SBGN specification are also broadcast on the mailing list and announced on the SBGN website.

Chapter 2

Entity Relationship Glyphs

This chapter provides a catalog of the graphical symbols available for representing entities in Entity Relationship diagrams. There are different classes of glyphs corresponding to different classes of entities, predicates, controls and operators.

In Chapter 3 beginning on page 22, we describe the rules for combining these glyphs into a legal SBGN Entity Relationship, and in Chapter 4 beginning on page 25, we describe requirements and guidelines for the way that diagrams are visually organized.

2.1 Overview

To set the stage for what follows in this chapter, we first give a brief overview of some of the concepts in the Entity Relationship notation with the help of an example shown in Figure 2.1 on the next page. This example will be re-use throughout the description of the graphical symbols (glyphs) used by SBGN Entity Relationship Level 1 (with a few additions when the concepts are missing in the example)

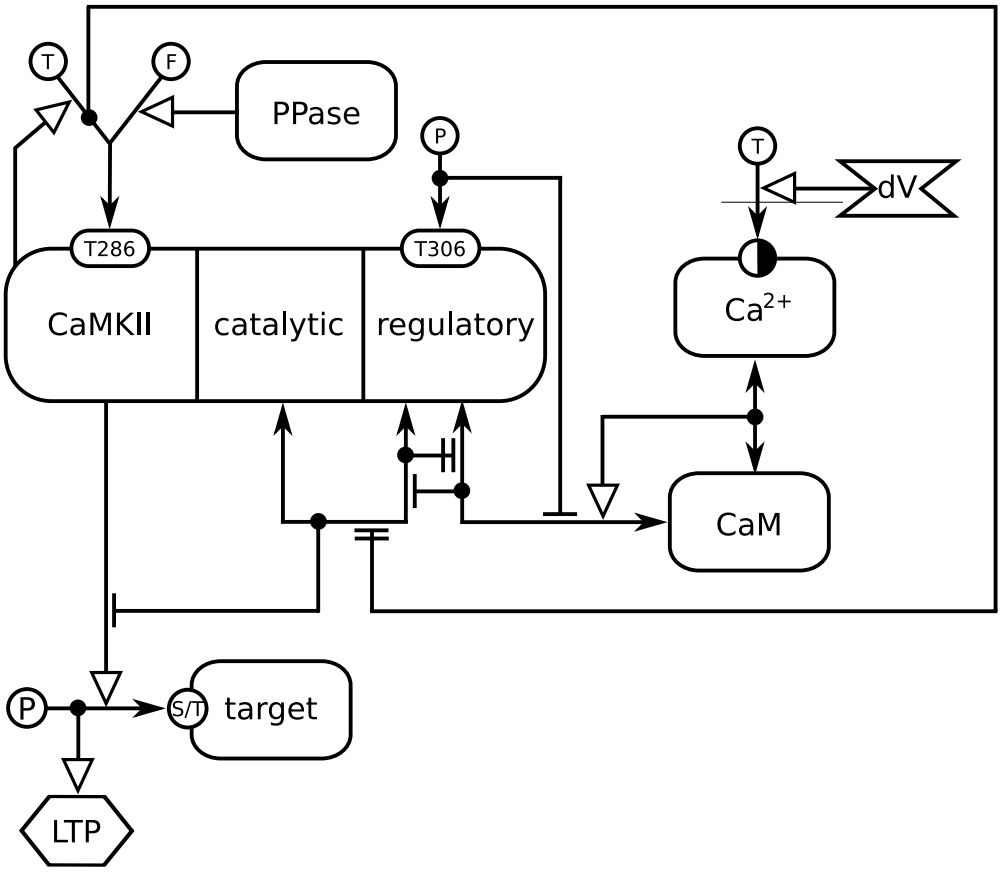

Figure 2.1: *This example of a Entity Relationship diagram depicts the effect of a depolarisation (dV) on the intracellular calcium, that binds to Calmodulin (CaM), that itself binds to the calcium/calmoduline kinase II (CaMKII). The binding of calmodulin inhibits the interaction between the catalytic and regulatory domains, thus relieving the inhibition on the kinase activity. The phosphorylation of the targets finally leads to the Long Term Potentiation (LTP) of the synapses. In addition, the diagram shows the effect of phosphorylation on threonine 286, that makes the enzyme constitutively active, and on threonine 306, that renders the kinase insensitive to calmodulin.*

The essence of the Entity Relationship is ... It shows how different entities in the system ... interact ...

In the example of Figure 2.1, ...

All nodes in Entity Relationship...

2.2 Controlled vocabularies used in SBGN Entity Relationship Level 1

Some glyphs in SBGN Entity Relationship diagrams can contain particular kinds of textual annotations conveying information relevant to the purpose of the glyph. These annotations are carried by *units of information* (Section 2.4.1) or *state variable values* (Section 2.4.2).

The text that appears as the unit of information decorating an entity must be prefixed with a controlled vocabulary term indicating the type of information being expressed. The prefixes are mandatory. Without the use of controlled vocabulary prefixes, it would be necessary to have different glyphs to indicate different classes of information; this would lead to an explosion in the number of symbols needed.

In the rest of this section, we describe the controlled vocabularies (CVs) used in SBGN Entity Relationship Level 1. In each case, some CV terms are predefined by SBGN, but unless otherwise noted, *they are not the only terms permitted*. Authors may use other CV values not listed here, but in such cases, they should explain the terms' meanings in a figure legend or other text accompanying the diagram.

2.2.1 Entity material types

The material type of an *Entity* indicates its chemical structure. A list of common material types is shown in Figure 2.2, but others are possible. The values are to be taken from the Systems Biology Ontology (`http://www.ebi.ac.uk/sbo/`), specifically from the branch having identifier `SBO:0000240` (*material entity*). The labels are defined by SBGN Entity Relationship Level 1.

Name	Label	SBO term
Non-macromolecular ion	`mt:ion`	SBO:0000327
Non-macromolecular radical	`mt:rad`	SBO:0000328
Ribonucleic acid	`mt:rna`	SBO:0000250
Deoxribonucleic acid	`mt:dna`	SBO:0000251
Protein	`mt:prot`	SBO:0000297
Polysaccharide	`mt:psac`	SBO:0000249

Figure 2.2: *A sample of values from the* material types *controlled vocabulary (Section 2.2.1).*

The material types are in contrast to the *conceptual types* (see below). The distinction is that material types are about physical composition, while conceptual types are about functions. For example, a strand of RNA is a physical artifact, but its use as messenger RNA is a function.

2.2.2 Entity conceptual types

An *entity*'s *conceptual type* indicates its function within the context of a given Entity Relationship. A list of common conceptual types is shown in Figure 2.3, but others are possible. The values are to be taken from the Systems Biology Ontology (`http://www.ebi.ac.uk/sbo/`), specifically from the branch having identifier `SBO:0000241` (*functional entity*). The labels are defined by SBGN Entity Relationship Level 1.

Name	Label	SBO term
Gene	`ct:gene`	SBO:0000243
Transcription start site	`ct:tss`	SBO:0000329
Gene coding region	`ct:coding`	SBO:0000335
Gene regulatory region	`ct:grr`	SBO:0000369
Messenger RNA	`ct:mRNA`	SBO:0000278

Figure 2.3: *A sample of values from the* conceptual types *vocabulary (Section 2.2.2).*

2.2.3 Macromolecule covalent modifications

A common reason for the introduction of state variables on an entity is to allow access to the configuration of possible covalent modification sites on that entity. For instance, a macromolecule may have one or more sites where a phosphate group many be attached; this change in the site's configuration (i.e., being either phosphorylated or not) may factor into whether, and how, the entity can participate in different processes. Being able to describe such modifications in a consistent fashion is the motivation for the existence of SBGN's covalent modifications controlled vocabulary.

Figure 2.4 on the next page lists a number of common types of covalent modifications. The most common values are defined by the Systems Biology Ontology in the branch having identifier `SBO:0000210` (*addition* under *events→reaction→biochemical reaction→conversion→addition*).

The labels shown in Figure 2.4 are defined by SBGN Entity Relationship Level 1; for all other kinds of modifications not listed here, the author of a Entity Relationship diagram must create a new label (and should also describe the meaning of the label in a legend or text accompanying the diagram).

Name	Label	SBO term
Acetylation	`Ac`	SBO:0000215
Glycosylation	`G`	SBO:0000217
Hydroxylation	`OH`	SBO:0000233
Methylation	`Me`	SBO:0000214
Myristoylation	`My`	SBO:0000219
Palmytoylation	`Pa`	SBO:0000218
Phosphorylation	`P`	SBO:0000216
Prenylation	`Pr`	SBO:0000221
Protonation	`H`	SBO:0000212
Sulfation	`S`	SBO:0000220
Ubiquitination	`Ub`	SBO:0000224

Figure 2.4: *A sample of values from the* covalent modifications *vocabulary (Section 2.2.3).*

2.2.4 Miscellaneous terms

SBGN Entity Relationship Level 1 requires several reserved characters. A special unit of information usable on interactions describe the number of identical interactors involved. Note that the value is a unitary number, and not (for example) a range. There is no provision in SBGN Process Diagram Level 1 for specifying a range in this context because it leads to problems of entity identifiability. Other reserved characters are used in state variable assignments to represent truth or falsehood.

Name	Label	SBO term
cardinality	`N:#`	`SBO:0000364`
true	`T`	`SBO:new`
false	`F`	`SBO:new`

Table 2.1: *Miscellaneous controlled terms. For the cardinality, # stands for the number; for example, "`N:5`".*

2.3 Interactor nodes

SBGN Entity Relationship Level 1 contains three glyphs representing classes of interactors: *entity*, *perturbing agent* and *outcome*. Entities can optionally carry auxiliary units such as *units of information* and *state variables*.

2.3.1 Glyph: *Entity*

SBGN Entity Relationship Level 1 defines only one glyph for all entities, whether physical entity, such as protein, a nucleic acid, metabolite or functional entity such as a gene. Indeed the exact nature of entities does not impact the rules of interactions within a diagram. The nature of

a particular entity may then be clarified using its label and decorations, as will become clear below.

SBO Term:

SBO:0000245 ! entity

Container:

An entity is represented by a rectangular container with rounded corners, as illustrated in Figure 2.5.

Label:

An *entity* is identified by a label placed in an unbordered box containing a string of characters. The characters can be distributed on several lines to improve readability, although this is not mandatory. The label box must be attached to the center of the container if the *entity* does not contain domains. If the *entity* contains *domains*, the label box must be attached to the center of the container representing the whole entity (Section **??**). The label may spill outside of the container.

Auxiliary items:

An *entity* might carry state variables that can add information about its state (Section 2.4.2). A state variable is represented by a "stadium", that is a rectangle capped with two hemi-circles, with the long axis of this stadium placed on the border of the *entity*'s container, as illustrated in Figure 2.5. The label of the state variable (which can precise the type of characteristic represented by the state variable, residue type, residue number etc.) is written within the state variable's container. Particular *state variables* are the existence (Section 2.4.2) and the location (Section 2.4.2).

An *entity* can carry one or several *units of information* (Section 2.4.1). Particular *units of information* are available for describing the material type (Section 2.2.1) and the conceptual type (Section 2.2.2) of a macromolecule. The center of the bounding box of a *unit of information* is located on the mid-line of the border of the macromolecule.

An *entity* can carry one of several *domains*. *Domains* are represented by labelled subsections of the *entity*. The leftward section represents the whole entity.

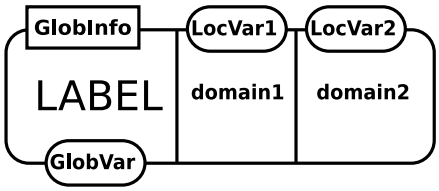

Figure 2.5: *The Entity Relationship glyph for* entity*, showing the location of domains, units of information, global and local state variables.*

2.3.2 Glyph: *Perturbing agent*

Biochemical networks can be affected by external influences. Those influences can be well-defined physical perturbations, such as a the effect of a light pulse or of a change in temperature; they can also be more complex and not well-defined phenomena, for instance a biological process, an experimental setup, or a mutation. For these situations, SBGN provides the *perturbing agent* glyph. We do not use the word *perturbation* to avoid the misunderstanding with the influence that the *perturbing agent* has on the map.

SBO Term:

SBO:0000405 ! perturbing agent

Container:

A *perturbing agent* is represented by a modified hexagon having two opposite concave faces, as illustrated in Figure 2.6.

Label:

A *perturbing agent* is identified by a label placed in an unbordered box containing a string of characters. The characters can be distributed on several lines to improve readability, although this is not mandatory. The label box must be attached to the center of the *perturbing agent* container. The label may spill outside of the container.

Auxiliary items:

A *perturbing agent* does not carry any auxiliary unit. [DOES-IT NOT? IS-IT AN ENTRY POINT? WHAT ABOUT THE EXISTENCE AND LOCATION VARIABLES?]

Figure 2.6: *The Entity Relationship glyph for* perturbing agent.

2.3.3 Glyph: *Outcome*

2.3.3.1 *Introduction*

In Entity Relationship diagram, an *outcome* represents the actualisation of a *statement* (Section 2.6) or an *influence* (Section 2.7). For instance, if an *interaction* represents a non-covalent binding, the *outcome* represents the complex. If an *interaction* represents a genetic interaction, for instance derived from genetic screenings, the *outcome* represents the result of the presence of the two polymorphisms. If an *assignment* represents the phosphorylation of a protein, the *outcome* represents the phosphorylated form of this protein.

SBO Term:

SBO:0000409 ! interaction outcome

Container:

An *outcome* is represented by a black dot located on the arc of a *statement* (Section 2.6) or an *influence* (Section 2.7). The diameter of the dot has to be larger than the thickness of the arc.

Label:

An *outcome* has no identity on its own and does not carry any label.

Auxiliary items:

An *outcome* does not carry any auxiliary items.

Figure 2.7: *The Entity Relationship glyph for* outcome.

2.4 Auxiliary units

2.4.1 Glyph: *Unit of information*

When representing biological entities, it is often necessary to convey some abstract information about the entity's function that cannot (or does not need to) be easily related to its structure. The SBGN *unit of information* is a decoration that can be used in this situation to add information to a glyph. Some example uses of a *unit of information* include characterizing a logical part of an entity, the information encoded in the entity, information about the physical environment, or the specific type of biological entity it is decorating.

SBO Term:

Not applicable.

Container:

A unit of information is represented by a rectangle. The long side of the rectangle should be oriented parallel to the border of the *entity* being annotated by the *unit of information*. The center of the bounding box of a *state of information* should be located on the mid-line of the border of the *entity*.

Label:

A *unit of information* is identified by a label placed in an unbordered box containing a string of characters. The characters can be distributed on several lines to improve readability, although this is not mandatory. The label box must be attached to the center of the container. The label may spill outside of the container.

The label defines the information carried by the *unit of information*. For certain predefined types of information having controlled vocabularies associated with them, SBGN defines specific prefixes that must be included in the label to indicate the type of information in question. The controlled vocabularies predefined in SBGN Entity Relationship Level 1 are described in Section 2.2 and summarized in the following list:

`mt` entity material type
`ct` entity conceptual type

Auxiliary items:

A *unit of information* does not carry any auxiliary items.

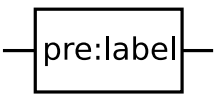

Figure 2.8: *The Entity Relationship glyph for* unit of information.

2.4.2 Glyph: *State variable*

Many biological entities such as molecules can exist in different *states*, meaning different physical or informational configurations. These states can arise for a variety of reasons. For example, macromolecules can be subject to post-synthesis modifications, wherein residues of the macromolecules (amino acids, nucleosides, or glucid residues) are modified through covalent linkage to other chemicals. Other examples of states are alternative conformations as in the closed/open/desensitized conformations of a transmembrane channel, and the active/inactive forms of an enzyme.

SBGN provides a means of associating one or more *state variables* with an entity; each such variable can be used to represent a dimension along which the state of the overall entity can vary. When an entity can exist in different states, the state of the whole entity (i.e., the SBGN object) can be described by the current values of all its *state variables*, and the values of the *state variables* of all its possible components, recursively.

In SBGN Entity Relationship Level 1, *state variables* are also used to describe the localisation in compartments (a transport is therefore described as a state variable assignment, see Section 2.6.1).

SBO Term:

Not applicable.

Container:

A *state variable* is represented by a "stadium" container, that is two hemicercles of same radius joined by parellel segments, as shown in Figure 2.9. The parallel segament axis should be tangent to the border of the glyph of the *EN* being modified by the *state variable*. The center of the bounding box of a *state variable* should be located on the mid-line of the border of the *EN*.

Label:

A *state variable* is identified by a label placed in an unbordered box containing a string of characters. The characters can be distributed on several lines to improve readability, although this is not mandatory. The label box must be attached to the center of the container. The label may spill outside of the container.

Auxiliary items:

A *state variable* does not carry any auxiliary items.

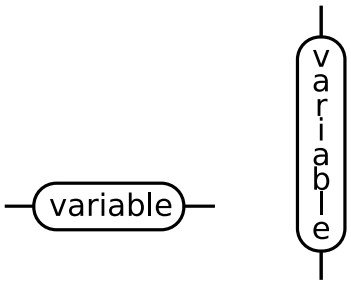

Figure 2.9: *The Entity Relationship glyph for* state variable.

A *state variable* does not necessarily have to be Boolean-valued. For example, an ion channel can possess several conductance states; a receptor can be inactive, active and desensitized; and so on. As another example, a *state variable* "ubiquitin" could also carry numerical values corresponding to the number of ubiquitin molecules present in the tail.

The state variable is assigned state-values (see Section 2.6.1). Those values are contained in a glyph similar to the *stateVariable*, although not carried by another *EN*.

Two state variables are predefined. The variable *existence* is used to represent the creation or destruction of entities, as seen on Figure 2.10 on the next page . *Existence* can take two values, true (T) or false (F). The variable is represented by a circle vertically divided in two. One hemicircle is black, and the other white.

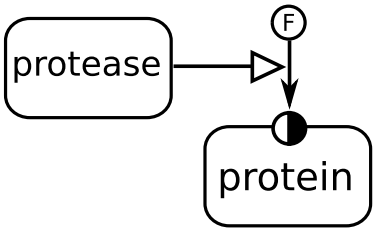

Figure 2.10: *Using the* state variable existence *to represent the degradation of a protein by a protease (i.e. the influence of the protease on the existence of the protein.*

The variable *location* is used to represent the physical location of an entity, as seen on Figure 2.11 . *Location* can take any value, but there can be only one *location* per entity. The variable is represented by a circle containing a slanted pin.

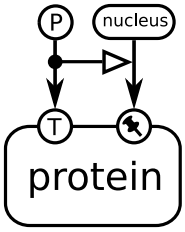

Figure 2.11: *Using the* state variable location *to represent the positive effect of a phosphorylation on the nuclear location of a protein (i.e. the influence of a phosphorylation assignment to the nuclear assignment). Note that this does not tell-us anything about the location of protein in the absence of phosphorylation.*

2.5 Glyph: *Submap*

A *submap* is used to encapsulate entities and relationships (including all types of nodes and edges) within one glyph.

TO BE FILLED.

2.6 Statements

2.6.1 Glyph: *Assignment*

Assignment is used to describe the setting of a state variable to a certain value. The assignment, represented by an harpoon arrow, goes from a variable value, represented by a floating *state variable* to a variable identification, represented by a *state variable* attached to the entity affected by the assignment. The result of an assignment is represented by *outcomes*, that is by filled dots on the arrow. The result of an *assignment* can be represented by any number of *outcomes*.

[NLN]

NLN: The arrowhead of assignment is currently the same than of interactions. It was suggested that we either one or the other or both. But no single combination

SBO Term:

non-applicable

Origin:

A state-variable (section Section 2.4.2) on its own containing a variable value.

Target:

A state-variable (section Section 2.4.2) carried by a interactor (section Section 2.3), containing a variable identification.

End point:

The target extremity of an *assignment* carries an harpoon arrowhead.

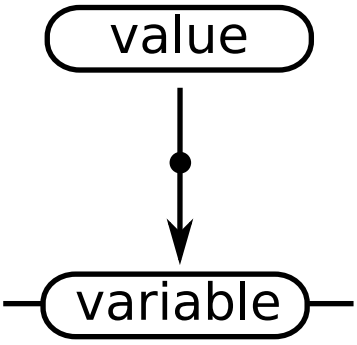

Figure 2.12: *The Entity Relationship glyph for* assignment.

2.6.2 Glyph: *Interaction*

Interaction represents an interaction between two *entity* or *outcome*, whether non-covalent physical interaction, or functional interaction, e.g. genetic interaction. Each arrowhead points to an interactor involved in the interaction. The result of the interaction is represented by *outcomes* (see section 2.3.3), that is by filled dots on the line linking the two arrowheads. The result of an interaction can be represented by any number of *outcomes*.

SBO Term:

SBO:0000342 molecular or genetic interaction

Origin:

entity Section 2.3.1 or *outcome* Section 2.3.3.

Target:

entity Section 2.3.1 or *outcome* Section 2.3.3.

End point:

Both origin and target extremities of an *interaction* carry an harpoon arrowhead.

[NLN]

NLN: TREAT THE CASE OF NON-BINARY INTERACTIONS

2.6.3 Glyph: *Observable*

A biochemical network can generate phenotypes or affect biological processes. Such processes can take place at different levels and are independent of the biochemical network itself. To represent these processes in a diagram, SBGN defines the *observable* glyph.

SBO Term:

SBO:0000358 ! process that affects an observable

Container:

An *observable* is represented by an elongated hexagon, as illustrated in Figure 2.13.

Label:

An *observable* is identified by a label placed in an unbordered box containing a string of characters. The characters can be distributed on several lines to improve readability, although this is not mandatory. The label box must be attached to the center of the *observable* container. The label may spill outside of the container.

DISCUSSION POINT HERE

Auxiliary items:

It is proposed that an *observable* glyph would stand as a short-hand for an entity representing the thing we measure, carrying a state variable existence (Section 2.4.2). An *observable* would therefore be different than the other entities because modulatory arcs would directly connect to it, while none would origin from it.

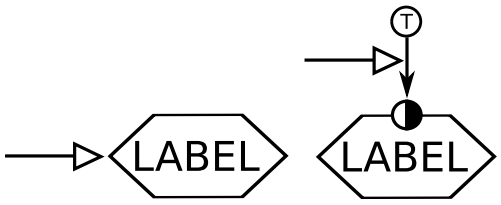

Figure 2.13: *The Entity Relationship glyph for* observable.

2.7 influences

Influence arcs represent the effect of an entity on another relationship. The symbols attached to their extremities precise their semantics. SBGN Entity Relationship Level 1's influences can be viewed as logical rules linking interactors and other rules.

2.7.1 Glyph: *Modulation*

A modulation affects the strength, or the probability, of the target relationship. Such a modulation can affect the relationship **positively or negatively**, or even both ways depending on the conditions. A *modulation* can also be used when one does not know the precise direction of the effect.

SBO Term:

SBO:0000168 ! control.

Origin:

Any *interactor* (Section 2.3) or any *logical operator* (Section 2.8).

Target:
Any *statement* (Section 2.6) or *influence* (Section 2.7).

End point:
The target extremity of a *modulation* carries an empty diamond.

Figure 2.14: *The Entity Relationship glyph for* modulation.

2.7.2 Glyph: *Stimulation*

A stimulation affects **positively** the the strength, or the probability, of the target relationship. This stimulation can be for instance a catalysis or a positive allosteric regulation.

SBO Term:
SBO:0000170 ! stimulation.

Origin:
Any *interactor* (Section 2.3) or any *logical operator* (Section 2.8).

Target:
Any *statement* (Section 2.6) or *influence* (Section 2.7).

End point:
The target extremity of a *stimulation* carries an empty arrowhead.

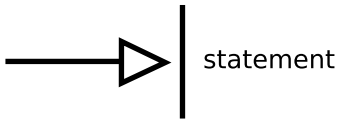

Figure 2.15: *The Process Diagram glyph for* stimulation.

2.7.3 Glyph: *Inhibition*

An inhibition **negatively** the the strength, or the probability, of the target relationship.

SBO Term:
SBO:0000169 ! inhibition.

Origin:
Any *interactor* (Section 2.3) or any *logical operator* (Section 2.8).

Target:
Any *statement* (Section 2.6) or *influence* (Section 2.7).

End point:
The target extremity of a *inhibitiom* carries a bar perpendicular to the arc.

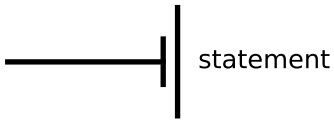

Figure 2.16: *The Process Diagram glyph for* inhibition.

2.7.4 Glyph: *Necessary stimulation*

A necessary stimulation is necessary for a relationship to take place. A relationship modulated by a necessary stimulation can only exist when this stimulation is true.

SBO Term:
SBO:0000171 ! necessary stimulation.

Origin:
Any *interactor* (Section 2.3) or any *logical operator* (Section 2.8).

Target:
Any *statement* (Section 2.6) or *influence* (Section 2.7).

End point:
The target extremity of a *necessary stimulation* carries an open arrow (to remind that it is a *stimulation*) coming after a larger vertical bar.

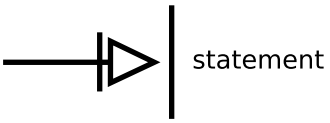

Figure 2.17: *The Process Diagram glyph for* necessary stimulation.

2.7.5 Glyph: *Absolute inhibition*

An absolute inhibition precludes the existence of another relationship.

SBO Term:
SBO:0000407 ! absolute inhibition.

Origin:
Any *interactor* (Section 2.3) or any *logical operator* (Section 2.8).

Target:
Any *statement* (Section 2.6) or *influence* (Section 2.7).

End point:
The target extremity of a *absolute inhibition* carries a double bar perpendicular to the arc (to remind that it is an *inhibition*).

Figure 2.18: *The Process Diagram glyph for* absolute inhibition.

2.7.6 Glyph: *Logic arc*

Logic arc is used to represent the fact that an interactor influences the outcome of a logic operator.

SBO Term:
SBO:0000398 - logical relationship.

Origin:
Any *interactor* (Section 2.3) or *logical operator* (Section 2.8).

Target:
Any *logical operator* (Section 2.8).

End point:
No particular symbol is used to represent a logic arc.

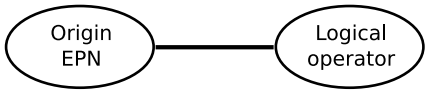

Figure 2.19: *The Entity Relationship glyph for* logic arc.

2.8 Logical operators

2.8.1 Glyph: *And*

The glyph *and* is used to denote that all the *interactor nodes* linked as input are necessary to produce the output influence.

SBO Term:
SBO:0000173 ! and.

Origin:
One interactor (section 2.3) or logical operator (section 2.8).

Target:
One modulation (section 2.7.1), stimulation (section 2.7.2), inhibition (section 2.7.3), necessary stimulation (section 2.7.4), or absolute inhibition (section 2.7.5) arc.

Node:
And is represented by a circle carrying the word "AND".

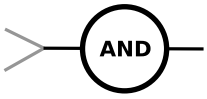

Figure 2.20: *The Entity Relationship glyph for* and. *Only two inputs are represented, but more would be allowed.*

2.8.2 Glyph: *Or*

The glyph *or* is used to denote that any of the *interactor nodes* linked as input is sufficient to produce the output influence.

SBO Term:
SBO:0000174 ! or.

Origin:
One interactor (section 2.3) or logical operator (section 2.8).

Target:
One modulation (section 2.7.1), stimulation (section 2.7.2), inhibition (section 2.7.3), necessary stimulation (section 2.7.4), or absolute inhibition (section 2.7.5) arc.

Node:
Or is represented by a circle carrying the word "OR".

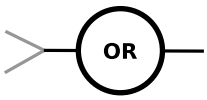

Figure 2.21: *The Entity Relationship glyph for* or. *Only two inputs are represented, but more would be allowed.*

2.8.3 Glyph: *Not*

The glyph *not* is used to denote that the output influence only happen in the absence of the input *interactor*.

SBO Term:
SBO:0000238 ! not.

Origin:
One interactor (section 2.3) or logical operator (section 2.8).

Target:
One modulation (section 2.7.1), stimulation (section 2.7.2), inhibition (section 2.7.3), necessary stimulation (section 2.7.4), or absolute inhibition (section 2.7.5) arc.

Node:
Not is represented by a circle carrying the word "NOT".

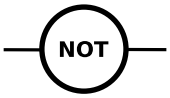

Figure 2.22: *The Entity Relationship glyph for* not.

2.8.4 Glyph: *delay*

The glyph *delay* is used to denote that the *interactor* linked as input does not produce the influence immediately.

SBO Term:

SBO:NEW ! delay.

Origin:

One interactor (section Section 2.3) or logical operator (section Section 2.8).

Target:

One modulation (section Section 2.7.1), stimulation (section Section 2.7.2), inhibition (section Section 2.7.3), necessary stimulation (section Section 2.7.4), or absolute inhibition (section Section 2.7.5) arc.

Node:

Delay is represented by a circle carrying the greek letter "τ" ("TAU").

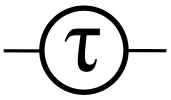

Figure 2.23: *The Entity Relationship glyph for* delay.

Chapter 3

Grammar of Entity Relationship diagrams

3.1 Overview

In this chapter, we describe how the glyphs of SBGN Entity Relationship Level 1 can be combined to make a valid Entity Relationship diagram. To do this, we must at the very least define what glyphs can be connected to each other. This is called syntax. Next, we must define rules over and above connection rules, such as whether duplicate symbols are permitted. In addition, we must define what the notation "means" — how does it represent a body of biological knowledge? This is semantics, and it is essential if a reader is to understand an SBGN diagram without external help, and a writer is to create one that reflects his understanding of a biological system.

In this section we start off by describing the concepts of the Entity Relationship notation. Next a detailed description of the syntax is provided followed by a description of the syntactic rules of the notation.

3.2 Concepts

The SBGN Entity Relationship language is more than a collection of symbols. It is a visual language that uses specific abstractions to describe the biological processes that make up a quantitative model, a signalling pathway or a metabolic network. This abstraction is the semantics of SBGN, and to describe it requires more than a definition of the symbols and syntax of the language. We first need to define the abstractions we are using.

SBGN Entity Relationship diagrams describe biological interactions involving biological entities. An *interactor* (Section 2.3), such as a molecule, influences the behaviour of other *interactor* via a relationships.

It may be convenient to think of a SBGN Entity Relationship diagrams as listing independent rules that decribe influences between interactors. Diagram can then be analysed with "what if?" queries.

3.3 Syntax

The syntax of SBGN Entity Relationship diagrams is defined in the form of an incidence matrix. An incidence matrix has arcs as rows and nodes as columns. Each element of the matrix represents the role of an arc in connection to a node. Input (I) means that the arc can begin at that node. Output (O) indicates that the arc can end at that node. Numbers in parenthesis represent the maximum number of arcs of a particular type to have this specific connection role with the node. Empty cells means the arc is not able to connect to the node. For simplicity Logical operators are treated as interactors. In addition, in Entity Relationship diagrams, any relationship can have another relationship as output.

3.3.1 Interactor Nodes connectivity definition

Arc\interactors	*entity*	*perturbation*	*observable*	*outcome*	*and*	*or*	*not*	*delay*
interaction	I			I				
assignment	O							
modulation	I	I		I	I	I	I	I
stimulation	I	I		I	I	I	I	I
necessary stimulation	I	I		I	I	I	I	I
inhibition	I	I		I	I	I	I	I
absolute inhibition	I	I		I	I	I	I	I
logic arc	I	I		I	IO	IO	IO	IO

3.3.2 Syntactic rules

The incidence matrix defining the main part of the syntax is too permissive. Additional rules allow to make the syntax definition more precise.

1.

3.4 Semantic description of Entity Relationship diagrams

3.4.1 Statements

An *interaction* linking the *interactors* A and B means: "A interacts with B". An *outcome* on an *interaction* represents the cases when the statement is true, that is when the interaction effectively exists. If the interaction is a physical interaction between molecules, the *outcome* represents the complex resulting from the interaction. It is used as follow: "when (or if) A interacts with B then ...".

An *assignement* linking a state variable value v to a *state-variable* V of an *interactor* I means: "v is assigned to V of I" or "V of I takes the value v". An*outcome* on an *assignment* represents the cases when the statement is true, that is when the variable effectively displays the value. It is used as follows: "when (or if) V of I takes the value v then ..." or more succintly "when (or if) $I\{V => v\}$ then ...".

3.4.2 Influences

A *modulation* linking an *interactor* I and a relationship R means: "If I then R is either reinforced or weakened". An *outcome* on a *modulation* represents the cases when the modulation effectively takes place. It is used as follow: "when (or if) I modulates R then ...".

A *stimulation* linking an *interactor* I and a relationship R means: "If I then R is reinforced" or "If I then the probability of R is increased". An *outcome* on a *stimulation* represents the cases when the stimulation effectively takes place. It is used as follow: "when (or if) I stimulates R then ...".

A *necessary stimulation* linking an *interactor* I and a relationship R means: "R only takes place is I. An *outcome* on a *necessary stimulation* represents the cases when the stimulation effectively takes place. It is used as follow: "when (or if) I allows R then ...".

An *inhibition* linking an *interactor* I and a relationship R means: "If I then R is weakened" or "If I then the probability of R is lowered". An *outcome* on an *inhibition* represents the cases when the inhibition effectively takes place. It is used as follow: "when (or if) I inhibits R then ...".

An *absolute inhibition* linking an *interactor* I and a relationship R means: "If I then R never takes place". An *outcome* on an *absolute inhibition* represents the cases when the absolute inhibition effectively takes place. It is used as follow: "when (or if) I blocks R then ...".

3.4.3 Logical Operators

An *and* means: "if all the inputs are true then output".

An *or* means: "if any of the input are true then output".

A *not* means: "if the input is not true then output".

A *delay* means: "if the input is true then output but not immediately".

Chapter 4

Layout Guidelines for an Entity Relationship Diagram

4.1 Introduction

The previous chapters describe the appearance and meaning of SBGN Entity Relationship Level 1 components which are *interactor nodes* as well as *connecting arcs*. The components of a Entity Relationship diagram have to be placed in a meaningful way – a random distribution with spaghetti-like connections will most likely hide the information encoded in the underlying model, whereas an elegant placement of the objects, giving a congenial appearance of the diagrams, may reveal new insights. The arrangement of components in a diagram is called a *layout*.

SBGN Entity Relationship diagrams should be easily recognizable not only by the glyphs used, but also by the general style of the layout. However, the arrangement of the components is a complex art in itself, and there is no simple rule which can be applied to all cases. Therefore this section provides guidelines for the layout of process diagrams, divided into two categories:

1. requirements, i. e. rules which **must** be fulfilled by a layout, and
2. recommendations, i. e. rules which **should** be followed if possible.

In addition, we provide a list of additional suggestions which may help in producing aesthetically more pleasant layouts, possibly easier to understand.

Those layout guidelines are independent of the method used to produce the diagram, and apply to both manually drawn diagrams as well as diagrams produced by an automatic layout algorithm. The guidelines do not deal with interactive aspects (e. g. the effect of zooming). Further information about automatic network layout (graph drawing) can be found, for example, in the books of Di Battista and co-authors [14] and Kaufmann and Wagner [15].

Please note that the color of objects do not carry any meaning in SBGN. Although one can use colors to emphasize part of a diagram or encode additional information, the meaning of the diagram should not depend on the colors. Furthermore, objects can have different sizes and size is also meaningless in SBGN. For example, a transition node may be larger than a protein node. Also the meaning of a graph should be conserved upon scaling as far as possible.

4.2 Layout guidelines

4.2.1 Requirements

Requirements are rules which **must** be fulfilled by a layout to produce a valid SBGN Entity Relationship Level 1 graph.

4.2.1.1 Node-node overlaps

Nodes are only allowed to overlap in the case that the overlapping nodes define a glyph (e. g. a *entity* composed by stacking auxiliary items such as a state variable on top of the rectangular container with rounded corners). Otherwise, nodes are not allowed to overlap (Figure 4.1). This includes the touching of nodes. Submaps are not allowed to overlap.

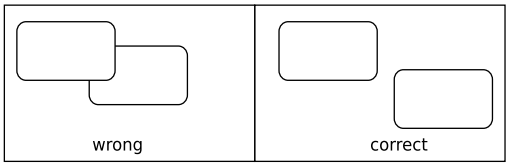

Figure 4.1: *Nodes must not overlap.*

4.2.1.2 Node-edge crossing

In case of node-edge crossing the edge must be drawn on the top of the node (Figure 4.2). See also recommendation 4.2.2.1 (crossing between edges and nodes should be avoided).

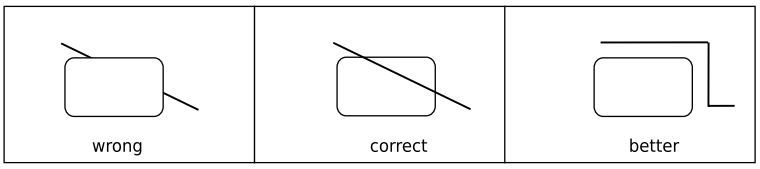

Figure 4.2: *If an edge crosses a node, the edge must be drawn on top of the node.*

4.2.1.3 Node border-edge overlaps

Edges are not allowed to overlap the border lines of nodes (Figure 4.3 on the following page).

4.2.1.4 Edge-edge overlaps

Edges are not allowed to overlap (Figure 4.4 on the next page). This includes touching of edges. Furthermore, an edge is neither allowed to cross itself nor to cross a boundary of node more than twice or other edges more than once.

4.2.1.5 Node orientation

Nodes have to be drawn horizontally or vertically, any other rotation of elements is not allowed (Figure 4.5 on the following page).

4.2.1.6 Interactions

The *interaction arcs* linking more than two *interactor nodes* are attached to a circle. Several outcomes of an interaction are not allowed to overlap (Figure 4.6 on the next page).

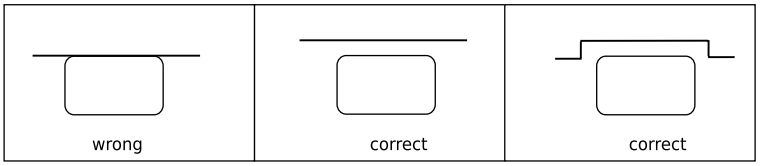

Figure 4.3: *Edges must not overlap node borders.*

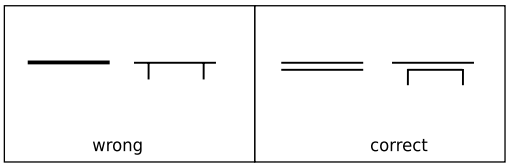

Figure 4.4: *Edges must not overlap.*

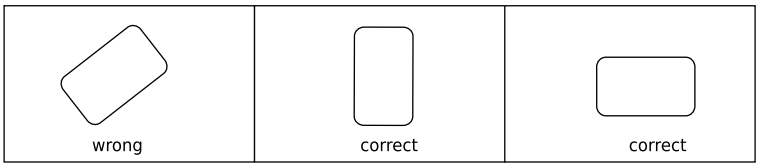

Figure 4.5: *The node orientation must be horizontally or vertically.*

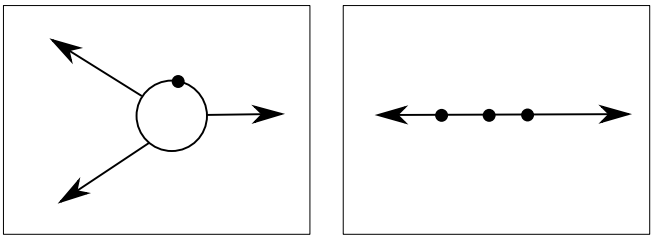

Figure 4.6: *Arcs linking more than two* interactor nodes *are attached to a circle and outcomes of an interaction are not allowed to overlap.*

4.2.1.7 Node labels

At least a part of the label (unbordered box containing a string of characters) has to be placed inside the node it belongs to. Node labels are not allowed to overlap nodes or other labels (this includes touching of other nodes or labels).

4.2.1.8 Edge labels

Edge labels are not allowed to overlap nodes. This includes touching of nodes.

4.2.2 Recommendations

Recommendations are rules which should be followed if possible to produce layouts may be easier to understand.

4.2.2.1 Node-edge crossing

Crossings between edges and nodes should be avoided. See also requirement 4.2.1.2 (in case of node-edge crossings the edge must be drawn on the top of the node).

4.2.2.2 Labels

Labels should be horizontal. Node labels should be placed completely inside the node if possible. Edge labels should be placed close to the edge and avoid overlapping the edge as well as other edge labels.

4.2.2.3 Avoid edge crossings

The amount of crossings between edges should be minimized.

4.2.2.4 Units of information

Units of information should not hide the structure of the corresponding node and should not overlap other elements.

4.2.3 Additional suggestions

Here is a list of additional layout suggestions which may help in producing aesthetically more pleasing layouts which may be easier to understand.

- Angle of edge crossings: If edge crossings are not avoidable edges should cross with an angle close to 90 degrees.
- Drawing area and width/height ratio: The drawing should be compact and the ratio between the width and the height of the drawing should be close to 1.
- Edge length: Long edges should be avoided.
- Number of edge bends: Edges should be drawn with as few bends as possible.
- Similar and symmetric parts: Similar parts of a diagram should be drawn in a similar way, and symmetric parts should be drawn symmetrically.
- Proximity information: Related elements (e. g. nodes connected by an arc or all elements within a submap) should be drawn close together.

Bibliography

[1] G. Michal. On representation of metabolic pathways. *BioSystems*, 47:1–7, 1998.

[2] K. W. Kohn. Molecular interaction map of the mammalian cell cycle control and DNA repair systems. *Molecular Biology of the Cell*, 10(8):2703–2734, 1999.

[3] P. P.-S. Chen. The entity-relationship model – toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, 1976.

[4] I. Pirson, N. Fortemaison, C. Jacobs, S. Dremier, J. E. Dumont, and C. Maenhaut. The visual display of regulatory information and networks. *Trends in Cell Biology*, 10(10):404–408, 2000.

[5] D. L. Cook, J. F. Farley, and S. J. Tapscott. A basis for a visual language for describing, archiving and analyzing functional models of complex biological systems. *Genome Biology*, 2(4):R0012.1–R0012.10, 2001.

[6] E. Demir, O. Babur, U Dogrusoz., A. Gursoy, G. Nisanci, R. Cetin-Atalay, and M. Ozturk. Patika: an integrated visual environment for collaborative construction and analysis of cellular pathways. *Bioinformatics*, 18(7):996–1003, 2002.

[7] E. Demir, O. Babur, U. Dogrusoz, A. Gursoy, A. Ayaz, G. Gulesir, G. Nisanci, and R. Cetin-Atalay. An ontology for collaborative construction and analysis of cellular pathways. *Bioinformatics*, 20(3):349–356, 2004.

[8] R. Maimon and S. Browning. Diagrammatic notation and computational structure of gene networks. In H. Kitano, editor, *Proceedings of the 2nd International Conference on Systems Biology*, pages 311–317, Madison, WI, 2001. Omnipress.

[9] H. Kitano. A graphical notation for biochemical networks. *BioSilico*, 1:169–176, 2003.

[10] H. Kitano, A. Funahashi, Y. Matsuoka, and K. Oda. Using process diagrams for the graphical representation of biological networks. *Nature Biotechnology*, 23(8):961–966, 2005.

[11] S. L. Moodie, A. A. Sorokin, I. Goryanin, and P. Ghazal. A graphical notation to describe the logical interactions of biological pathways. *Journal of Integrative Bioinformatics*, 3:36.1–36.11, 2006.

[12] N. Le Novère, S. Moodie, A. Sorokin, M. Hucka, F. Schreiber, E. Demir, H. Mi, Y. Matsuoka, K. Wegner, and H. Kitano. Systems biology graphical notation: Process diagram level 1. Nature precedings, 2008.

[13] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J.-H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le Novère, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence,

J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang. The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[14] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, New Jersey, 1998.

[15] M. Kaufmann and D. Wagner. *Drawing Graphs: Methods and Models*, volume 2025 of *Lecture Notes in Computer Science Tutorial*. Springer, 2001.