Systems Biology Graphical Notation:
Process Diagram Level 1

Release 1.0
Date: 15 August, 2008

Editors:
Nicolas Le Novère
Stuart Moodie
Anatoly Sorokin
Michael Hucka

EMBL European Bioinformatics Institute, UK
CSBE, University of Edinburgh, UK
University of Edinburgh, UK
California Institute of Technology, USA

Principal Authors:
Nicolas Le Novère
Stuart Moodie
Anatoly Sorokin
Michael Hucka
Falk Schreiber
Emek Demir
Huaiyu Mi
Yukiko Matsuoka
Katja Wegner
and
Hiroaki Kitano

EMBL European Bioinformatics Institute, UK
CSBE, University of Edinburgh, UK
University of Edinburgh, UK
IPK Gatersleben & MLU Halle, Germany
MSKCC Computational Biology Center, USA
SRI International, USA
The Systems Biology Institute, Japan
University of Hertfordshire, UK
The Systems Biology Institute, Japan

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. Bug reports and specific comments about the specification should be entered in the issue tracker http://sourceforge.net/tracker/?group_id=178553&atid=1082245.
Preface

Acknowledgements

The authors are grateful to all the attendees of the SBGN meetings, as well as to the subscribers of the sbgn-discuss@sbgn.org mailing list. The authors would like to acknowledge especially the help of Frank Bergmann, Sarala Dissanayake, Ralph Gauges, Peter Ghazal, and Lu Li. SM and AS would also like to acknowledge Igor Goryanin whose financial support and encouragement enabled us to commit the necessary time to the development of this specification. A more comprehensive list of people involved in SBGN development is available in the appendix D.


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.
## Contents

**Preface**  
1 What is the Systems Biology Graphical Notation?  
  1.1 History of SBGN development  
  1.2 The three languages of SBGN  
  1.3 SBGN levels  
  1.4 Developments, discussions, and notifications of updates  
2 Process Diagram glyphs  
  2.1 Overview  
  2.2 Controlled vocabularies used in SBGN Process Diagram Level 1  
  2.2.1 Entity pool node material types  
  2.2.2 Entity pool node conceptual types  
  2.2.3 Macromolecule covalent modifications  
  2.2.4 Physical characteristics of compartments  
  2.2.5 Cardinality  
  2.3 Entity pool nodes  
  2.3.1 Glyph: Unspecified entity  
  2.3.2 Glyph: Simple chemical  
  2.3.3 Glyph: Macromolecule  
  2.3.4 Glyph: Nucleic acid feature  
  2.3.5 Glyph: Multimer  
  2.3.6 Glyph: Source/sink  
  2.3.7 Glyph: Perturbation  
  2.3.8 Glyph: Observable  
  2.3.9 Glyph: Tag  
  2.3.10 Glyph: Unit of information  
  2.3.11 Glyph: State variable  
  2.3.12 Glyph: Clone marker  
  2.3.13 Examples of complex EPNs  
  2.4 Container nodes  
  2.4.1 Glyph: Complex  
  2.4.2 Glyph: Compartment  
  2.4.3 Glyph: Submap  
  2.5 Process nodes  
  2.5.1 Glyph: Transition  
  2.5.2 Glyph: Omitted process  
  2.5.3 Glyph: Uncertain process  
  2.5.4 Glyph: Association  
  2.5.5 Glyph: Dissociation  
  2.6 Connecting arcs  
  2.6.1 Glyph: Consumption  
  2.6.2 Glyph: Production  
  2.6.3 Glyph: Modulation  
  2.6.4 Glyph: Stimulation  
  2.6.5 Glyph: Catalysis  
  2.6.6 Glyph: Inhibition  
  2.6.7 Glyph: Trigger  
  2.6.8 Glyph: Logic arc  
  2.6.9 Glyph: Equivalence arc  
  2.7 Logical operators  
  2.7.1 Glyph: And  
  2.7.2 Glyph: Or  
  2.7.3 Glyph: Not  
3 Process Diagram grammar  
  3.1 Overview  
  3.2 Concepts  
  3.3 The conceptual model  
  3.4 Syntax  
  3.4.1 Entity Pool Nodes connectivity definition  
  3.4.2 Process Nodes connectivity definition  
  3.4.3 Containment definition  
  3.4.4 Syntactic rules  
  3.5 Semantic rules  
  3.5.1 Namespaces  
  3.5.2 Cloning  
  3.5.3 State variables  
  3.5.4 Compartment spanning  
  3.5.5 Compartments  
  3.5.6 Modulation  
  3.5.7 Submaps  
4 Layout Guidelines for a Process Diagram  
  4.1 Introduction  
  4.2 Layout guidelines  
  4.2.1 Requirements  
  4.2.2 Recommendations  
  4.2.3 Additional suggestions  
A Complete examples of SBGN Process Diagram Level 1 graphs  
B Reference card  
C Issues postponed to future levels  
  C.1 Multicompartment entities  
  C.2 Logical combination of state variable values  
  C.3 Non-chemical entity nodes  
  C.4 Generics  
  C.5 State and transformation of compartments  
D Extended list of people involved in SBGN Process Diagram Level 1 development
Chapter 1

What is the Systems Biology Graphical Notation?

The goal of the Systems Biology Graphical Notation (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 Process Diagram visual language of SBGN. As explained more fully in Section 1.2, Process 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 Process 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
CHAPTER 1. WHAT IS THE SYSTEMS BIOLOGY GRAPHICAL NOTATION?

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.

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 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; [12]): 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](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](mailto: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

Process Diagram glyphs

This chapter provides a catalog of the graphical symbols available for representing entities in Process Diagrams. There are different classes of glyphs corresponding to different classes of material or conceptual entities, containers, processes, connecting arcs, and logical operators. In Chapter 3 beginning on page 40, we describe the rules for combining these glyphs into a legal SBGN Process Diagram, and in Chapter 4 beginning on page 57, 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 Process Diagram notation with the help of an example shown in Figure 2.1 on the next page.
Figure 2.1: This example of a Process Diagram uses two kinds of entity pool nodes: one for pools of different macromolecules (Section 2.3.3) and another for pools of simple chemicals (Section 2.3.2). Most macromolecule nodes in this diagram are adorned with state variables (Section 2.3.11) representing phosphorylation states. This diagram uses one type of process node, the transition node (Section 2.5.1), and one kind of connecting arc, catalysis (Section 2.6.5). Finally, some entity pool nodes have dark bands along their bottoms; these are clone markers indicating that the same pool nodes appear multiple times in the diagram.

The diagram in Figure 2.1 is a simple diagram for part of a mitogen-activated protein kinase (MAPK) cascade. The larger nodes in the figure (some of which are in the shape of rounded rectangles and others in the shape of circles) represent biological materials—things like macromolecules and simple chemicals. The biological materials are altered via processes, which are indicated in SBGN by lines with arrows and other decorations. In this particular diagram, all of the processes happen to be the same: transitions catalyzed by biochemical entities. The directions of the arrows indicate the direction of the transitions; for example, unphosphorylated RAF kinase transitions to phosphorylated RAF kinase via a process catalyzed by RAS. Although ATP and ADP are shown as incidental to the phosphorylations on this particular graph, they are involved in the same process than the proteins getting phosphorylated. The small circles on the nodes for RAF and other entity pools represent state variables (in this case, phosphorylation sites).

The essence of the Process Diagram is change: it shows how different entities in the system transition from one form to another. The entities themselves can be many different things. In the example of Figure 2.1, they are either pools of macromolecules or pools of simple chemicals, but as will become clear later in this chapter, they can be other conceptual and material constructs as well. Note also that we speak of entity pools rather than individuals; this is because in biochemical network models, one does not focus on single molecules, but rather collections of molecules of the same kind. The molecules in a given pool are considered indistinguishable from each other. The way in which one type of entity is transformed into another is conveyed by process nodes, and links between entity pool nodes and process nodes indicate an influence by the entities on the processes. In the case of Figure 2.1, those links describe consumption Section 2.6.1, production Section 2.6.2 and catalysis Section 2.6.5, but others are possible. Finally, nodes in Process Diagrams are usually not repeated; if they do need to be repeated,
they are marked with clone markers—specific modifications to the appearance of the node (Section 2.3.12). The details of this and other aspects of Process Diagram notation are explained in the rest of this chapter.

Table 2.1 summarizes the different SBGN abstractions described in this chapter.

<table>
<thead>
<tr>
<th>Component</th>
<th>Abbrev.</th>
<th>Role</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity pool node</td>
<td>EPN</td>
<td>A population of entities that cannot be distinguished from each other</td>
<td>Specific macromolecules or other chemical species</td>
</tr>
<tr>
<td>Container node</td>
<td>CN</td>
<td>An encapsulation of one or more other SBGN constructs</td>
<td>Complexes, compartments</td>
</tr>
<tr>
<td>Process node</td>
<td>PN</td>
<td>A process that transforms one or more EPNs into one or more other EPNs</td>
<td>Transition, association, dissociation</td>
</tr>
<tr>
<td>Connecting arc</td>
<td>—</td>
<td>Links between EPNs or CNs to PNs or CNs to indicate influences</td>
<td>Production, catalysis, inhibition</td>
</tr>
<tr>
<td>Logical operators</td>
<td>—</td>
<td>Combines one or several inputs into one output</td>
<td>Boolean and, or, not</td>
</tr>
</tbody>
</table>

Table 2.1: Summary of Process Diagram components and their roles.

2.2 Controlled vocabularies used in SBGN Process Diagram Level 1

Some glyphs in SBGN Process Diagrams can contain particular kinds of textual annotations conveying information relevant to the purpose of the glyph. These annotations are units of information (Section 2.3.10) or state variable (Section 2.3.11). An example is in the case of multimers, which can have a unit of information conveying the number of monomers composing the multimer. Other cases are described throughout the rest of this chapter.

The text that appears as the unit of information decorating an Entity Pool Node (EPN) must in most cases be prefixed with a controlled vocabulary term indicating the type of information being expressed. The prefixes are mandatory except in the case of macromolecule covalent modifications (Section 2.2.3). 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 Process Diagram Level 1. They cover the following categories of information: an EPN’s material type, an EPN’s conceptual type, covalent modifications on macromolecules, the physical characteristics of compartments, and cardinality (e.g., of multimers). 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 pool node material types

The material type of an EPN indicates its chemical structure. A list of common material types is shown in Table 2.2 on the following page, 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 under participant→physical participant). The labels are defined by SBGN Process Diagram Level 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 roles. For example, a strand of RNA is a physical artifact, but its use as messenger RNA is a role.
CHAPTER 2. PROCESS DIAGRAM GLYPHS

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
<th>SBO term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-macromolecular ion</td>
<td>mt:ion</td>
<td>SBO:0000327</td>
</tr>
<tr>
<td>Non-macromolecular radical</td>
<td>mt:rad</td>
<td>SBO:0000328</td>
</tr>
<tr>
<td>Ribonucleic acid</td>
<td>mt:rna</td>
<td>SBO:0000250</td>
</tr>
<tr>
<td>Deoxribonucleic acid</td>
<td>mt:dna</td>
<td>SBO:0000251</td>
</tr>
<tr>
<td>Protein</td>
<td>mt:prot</td>
<td>SBO:0000297</td>
</tr>
<tr>
<td>Polysaccharide</td>
<td>mt:psac</td>
<td>SBO:0000249</td>
</tr>
</tbody>
</table>

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

2.2.2 Entity pool node conceptual types

An EPN’s conceptual type indicates its function within the context of a given Process Diagram. A list of common conceptual types is shown in Table 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 (conceptual entity under participant → physical participant). The labels are defined by SBGN Process Diagram Level 1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
<th>SBO term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene</td>
<td>ct:gene</td>
<td>SBO:0000243</td>
</tr>
<tr>
<td>Transcription start site</td>
<td>ct:tss</td>
<td>SBO:0000329</td>
</tr>
<tr>
<td>Gene coding region</td>
<td>ct:coding</td>
<td>SBO:0000335</td>
</tr>
<tr>
<td>Gene regulatory region</td>
<td>ct:grr</td>
<td>SBO:0000369</td>
</tr>
<tr>
<td>Messenger RNA</td>
<td>ct:mRNA</td>
<td>SBO:0000278</td>
</tr>
</tbody>
</table>

Table 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 (Section 2.3.11) 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 may 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.

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

2.2.4 Physical characteristics of compartments

SBGN Process Diagram Level 1 defines a special unit of information for describing certain common physical characteristics of compartments. Table 2.5 on the following page lists the particular values defined by SBGN Process Diagram Level 1. The values correspond to the Systems Biology Ontology branch with identifier SBO:0000255 (physical characteristic under quantitative parameter).
CHAPTER 2. PROCESS DIAGRAM GLYPHS

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
<th>SBO term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acetylation</td>
<td>Ac</td>
<td>SBO:0000215</td>
</tr>
<tr>
<td>Glycosylation</td>
<td>G</td>
<td>SBO:0000217</td>
</tr>
<tr>
<td>Hydroxylation</td>
<td>OH</td>
<td>SBO:0000233</td>
</tr>
<tr>
<td>Methylolation</td>
<td>Me</td>
<td>SBO:0000214</td>
</tr>
<tr>
<td>Myristoylation</td>
<td>My</td>
<td>SBO:0000219</td>
</tr>
<tr>
<td>Palmytoylation</td>
<td>Pa</td>
<td>SBO:0000218</td>
</tr>
<tr>
<td>Phosphorylation</td>
<td>P</td>
<td>SBO:0000216</td>
</tr>
<tr>
<td>Prenylation</td>
<td>Pr</td>
<td>SBO:0000221</td>
</tr>
<tr>
<td>Protonation</td>
<td>H</td>
<td>SBO:0000212</td>
</tr>
<tr>
<td>Sulfation</td>
<td>S</td>
<td>SBO:0000220</td>
</tr>
<tr>
<td>Ubiquitination</td>
<td>Ub</td>
<td>SBO:0000224</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
<th>SBO term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>pc:T</td>
<td>SBO:0000147</td>
</tr>
<tr>
<td>Voltage</td>
<td>pc:V</td>
<td>SBO:0000259</td>
</tr>
<tr>
<td>pH</td>
<td>pc:pH</td>
<td>SBO:0000304</td>
</tr>
</tbody>
</table>

Table 2.5: A sample of values from the physical characteristics vocabulary (Section 2.2.4).

2.2.5 Cardinality

SBGN Process Diagram Level 1 defines a special unit of information usable on multimers for describing the number of monomers composing the multimer. Table 2.6 shows the way in which the values must be written. 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Label</th>
<th>SBO term</th>
</tr>
</thead>
<tbody>
<tr>
<td>cardinality</td>
<td>N:#</td>
<td>SBO:0000364</td>
</tr>
</tbody>
</table>

Table 2.6: The format of the possible values for the cardinality unit of information (Section 2.2.5). Here, # stands for the number; for example, “N:5”.

2.3 Entity pool nodes

An entity pool is a population of entities that cannot be distinguished from each other, when it comes to the SBGN Process Diagram Level 1 map. For instance all the molecular entities that fulfill the same role in a given process form an entity pool. As a result, an entity pool can represent different granularity levels, such as all the proteins, all the instances of a given protein, only certain forms of a given protein. To belong to a different compartment is sufficient to belong to different entity pools. Calcium ions in the endoplasmic reticulum and calcium ions in the cytosol belong to different entity pools when it comes to representing calcium release from the endoplasmic reticulum.

SBGN Process Diagram Level 1 contains five glyphs representing classes of material entities: unspecified entity, simple chemical, macromolecule, nucleic acid feature, and multimer. (Specific types of macromolecules, such as protein, RNA, DNA, polysaccharide, and specific simple chemicals are not defined by SBGN Process Diagram Level 1 but may be part of future levels
CHAPTER 2. PROCESS DIAGRAM GLYPHS

of SBGN.) In addition to the material entities, SBGN Process Diagram Level 1 represents four conceptual entities: source/sink, perturbation, observable, and tag. Material and conceptual entities can optionally carry auxiliary units such as units of information, state variables and clone markers.

2.3.1 Glyph: Unspecified entity

The simplest type of EPN is the unspecified entity: one whose type is unknown or simply not relevant to the purposes of the model. This arises, for example, when the existence of the entity has been inferred indirectly, or when the entity is merely a construct introduced for the needs of the model, without direct biological relevance. These are examples of situations where the unspecified entity glyph is appropriate. (Conversely, for cases where the identity of the entities composing the pool is known, there exist other, more specific glyphs described elsewhere in the SBGN Process Diagram Level 1 specification.)

SBO Term:

SBO:0000285 ! material entity of unknown nature

Container:

An unspecified entity is represented by an elliptic container, as shown in Figure 2.2.

Label:

An unspecified 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. The label may spill outside of the container.

Auxiliary items:

An unspecified entity may carry a clone marker (Section 2.3.12).

Figure 2.2: The Process Diagram glyph for unspecified entity.

2.3.2 Glyph: Simple chemical

A simple chemical in SBGN is defined as the opposite of a macromolecule (Section 2.3.3): it is a chemical compound that is not formed by the covalent linking of pseudo-identical residues. Examples of simple chemicals are an atom, a monoatomic ion, a salt, a radical, a solid metal, a crystal, etc.

SBO Term:

SBO:0000247 ! simple chemical

Container:

A simple chemical is represented by a circular container, as depicted in Figure 2.3 on the next page.

Label:

The identification of the simple chemical is carried by an unbordered box containing a string of characters. The characters may be distributed on several lines to improve
readability, although this is not mandatory. The label box has to be attached to the center of the circular container. The label is permitted to spill outside the container.

**Auxiliary items:**
A *simple chemical* may be decorated with one or more *units of information* (Section 2.3.10). A particular *unit of information* describes the material type. A *simple chemical* may also carry a *clone marker* (Section 2.3.12).

![Label](image)

**Figure 2.3:** The Process Diagram glyph for simple chemical.

### 2.3.3 Glyph: Macromolecule

Many biological processes involve *macromolecules*: biochemical substances that are built up from the covalent linking of pseudo-identical units. Examples of macromolecules include proteins, nucleic acids (RNA, DNA), and polysaccharides (glycogen, cellulose, starch, etc.). Attempting to define a separate glyph for all of these different molecules would lead to an explosion of symbols in SBGN, so instead, SBGN Process Diagram Level 1 defines only one glyph for all macromolecules. The same glyph is to be used for a protein, a nucleic acid, a complex sugar, and so on. The exact nature of a particular macromolecule in a diagram is then clarified using its label and decorations, as will become clear below. (Future levels of SBGN may subclass the macromolecule and introduce different glyphs to differentiate macromolecules.)

**SBO Term:**

SBO:0000245 ! macromolecule

**Container:**
A macromolecule is represented by a rectangular container with rounded corners, as illustrated in Figure 2.4 on the following page.

**Label:**
A *macromolecule* 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 *macromolecule* can carry state variables that can add information about its state (Section 2.3.11). The state of a macromolecule is therefore defined as the vector of all its state variables.

A *macromolecule* can also carry one or several *units of information* (Section 2.3.10). The units of information can characterize a domain, such as a binding site. 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.

A *macromolecule* may also carry a *clone marker* (see Section 2.3.12)
2.3.4 Glyph: Nucleic acid feature

The Nucleic acid feature construct in SBGN is meant to represent a fragment of a macromolecule carrying genetic information. A common use for this construct is to represent a gene or transcript. The label of this EPN and its units of information are often important for making the purpose clear to the reader of a diagram.

SBO Term:
SBO:0000354 ! genetic entity

Container:
A nucleic acid feature is represented by a rectangular container whose bottom half has rounded corners, as shown in Figure 2.5. This design reminds that we are fundamentally dealing with a unit of information, but this information is carried by a macromolecule.

Label:
The identity of a particular Nucleic acid feature is established by a label placed in an unordered box containing a string of characters. The characters may 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 nucleic acid feature can carry state variables (Section 2.3.11) that add information about its precise state. The state of a nucleic acid feature is therefore defined as the vector of all its state variables.

A nucleic acid feature can also carry one or several units of information (Section 2.3.10). These can characterize a nucleic acid feature’s domain, such as a binding site, or an exon. Particular units of information carry the material type (Section 2.2.1) and the conceptual type (Section 2.2.2) of the nucleic acid feature.

A nucleic acid feature may also carry a clone marker (Section 2.3.12).

2.3.5 Glyph: Multimer

As its name implies, a multimer is an aggregation of multiple identical or pseudo-identical entities held together by non-covalent bonds (Thus, they are distinguished from polymers by the
fact that the later involve covalent bonds). Here, *pseudo-identical* refers to the possibility that the entities differ chemically but retain some common global characteristic, such as a structure or function, and so can be considered identical within the context of the SBGN Process Diagram. An example of this are the homologous subunits in a hetero-oligomeric receptor. SBGN Process Diagram accepts multimers of *simple chemical* (Section 2.3.2), *macromolecule* (Section 2.3.3) and *genetic entities* (Section 2.3.4).

**SBO Term:**

SBO:0000286 ! multimer

**Container:**

A *multimer* is represented by two identical containers shifted horizontally and vertically and stacked one on top of the other. Figure 2.6 illustrates the glyph.

**Label:**

A *multimer* has no identity on its own. However, the first of the monomers carries an identifying label. The label is 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 top monomer’s container. The label may spill outside of the container.

**Auxiliary items:**

A *multimer* can carry state variables that can add information about its state (Section 2.3.11). The state of a multimer is therefore defined as the vector of all its state variables. Note that a *state variable* carried by a multimer actually applies to each of the constituent monomers individually. If instead the state variables are meant to apply to the whole multimeric assembly, a *macromolecule* (Section 2.3.3) should be used instead of *multimer*. An assembly containing some state variables applicable to the components, and others state variable applicable to the assembly (for instance opening of a channel and phosphorylation of each of its subunits) should be represented by a *complex* (Section 2.4.1).

A *multimer* can also carry one or several *units of information* (Section 2.3.10). The information can characterize a domain, such as a binding site. Particular *units of information* exist for describing the material type (Section 2.2.1), the conceptual type (Section 2.2.2), and the cardinality (Section 2.2.5) of the multimer. Note that a *unit of information* carried by a multimer actually applies to each of the constituent monomers individually. If instead a *unit of information* should be applicable to the whole multimeric assembly, a *macromolecule* should be used (Section 2.3.3). An assembly containing units of information applying to the components, and others to the assembly should be represented by a *complex* (Section 2.4.1).

A *multimer* may also carry a *clone marker* (Section 2.3.12).

![Figure 2.6: The Process Diagram glyph for multimer with an additional unit of information containing the cardinality.](image)
2.3.6 Glyph: *Source/sink*

It is useful to have the ability to represent the creation of an entity or a state from an unspecified source, that is, from something that one does not need or wish to make precise. For instance, in a model where the production of a protein is represented, it may not be desirable to represent all of the amino acids, sugars and other metabolites used, or the energy involved in the protein’s creation. Similarly, we may not wish to bother representing the details of the destruction or decomposition of some biochemical species into a large number of more primitive entities, preferring instead to simply say that the species “disappears into a sink”. Yet another example is that one may need to represent an input (respectively, output) into (resp. from) a compartment without explicitly representing a transport process from a source (resp. to a target).

For these and other situations, SBGN defines a symbol for explicitly representing the involvement of an unspecified source or sink. The symbol used in SBGN is borrowed from the mathematical symbol for “empty set”, but it is important to note that it does not actually represent a true absence of everything or a physical void—it represents the absence of the corresponding structures in the model, that is, the fact that these sources or sinks are conceptually outside the scope of the diagram.

A frequently asked question is, why bother having an explicit symbol at all? The reason is that one cannot simply use an arc that does not terminate on a node, because the dangling end could be mistaken to be pointing to another node in the diagram. This is specially true if the diagram is rescaled, causing the spacing of elements in the diagram to change. The availability and use of an explicit symbol for sources and sinks is critical.

**SBO Term:**

SBO:0000291 ! empty set

**Container:**

A *source/sink* is represented by a glyph for “empty set”, that is, a circle crossed by a bar linking the upper-right and lower-left corners of an invisible square drawn around the circle. Figure 2.7 illustrates this. The symbol should only be linked to one and only one edge in a diagram.

**Label:**

An *source/sink* does not carry any labels.

**Auxiliary items:**

An *source/sink* does not carry any auxiliary items.

![Figure 2.7: The Process Diagram glyph for source/sink.](image)

2.3.7 Glyph: *Perturbation*

Biochemical networks can be affected by external influences. Those influences can be well-defined physical perturbations, such as a light pulse or 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 *perturbation* glyph.

**SBO Term:**

SBO:0000357 ! perturbation
CHAPTER 2. PROCESS DIAGRAM GLYPHS

Container:
A perturbation is represented by a modified hexagon having two opposite concave faces, as illustrated in Figure 2.8.

Label:
A perturbation 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 perturbation container. The label may spill outside of the container.

Auxiliary items:
A perturbation may carry a clone marker (Section 2.3.12).

2.3.8 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 ! observable

Container:
An observable is represented by an elongated hexagon, as illustrated in Figure 2.9.

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.

Auxiliary items:
An observable may carry a clone marker (Section 2.3.12).

2.3.9 Glyph: Tag
A tag is a named handle, or reference, to another EPN (Section 2.3) or a container node (Section 2.4). Tags can be used to identify elements in SBGN submaps (Section 2.4.3).

SBO Term:
Not applicable.
CHAPTER 2. PROCESS DIAGRAM GLYPHS

Container:
A tag is represented by a rectangle fused to an empty arrowhead, as illustrated in Figure 2.10. The symbol should only be linked to one and only one edge (i.e., it should reference only one EPN or container).

Label:
A tag 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 tag does not carry any auxiliary items.

Figure 2.10: The Process Diagram glyph for tag.

2.3.10 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 include: characterizing a logical part of an entity such as a functional domain (a binding domain, a catalytic site, a promoter, etc.), or the information encoded in the entity (an exon, an open reading frame, etc.). A unit of information can also convey 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 EPN 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 EPN.

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 Process Diagram Level 1 are described in Section 2.2 and summarized in the following list:

- **pc** container physical characteristic
- **mt** entity pool material type
CHAPTER 2. PROCESS DIAGRAM GLYPHS

ct  entity pool conceptual type
N  multimer cardinality

Auxiliary items:
A unit of information does not carry any auxiliary items.

Figure 2.11: The Process Diagram glyph for unit of information.

2.3.11 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.

SBO Term:
Not applicable.

Container:
A state variable is represented by an elliptical container, as shown in Figure 2.12 on the next page. The ellipse’s long axis should be tangent to the border of the glyph of the EPN being modified by the state variable. The center of the bounding box of a state of information should be located on the mid-line of the border of the EPN.

Label:
The identification of an instance of a state variable is carried by one or two unbordered boxes, each containing a string of characters. The characters cannot be distributed on several lines. One box is mandatory, and contains the value of the state variable. The value may be empty; an example of a situation where this might arise is an unphosphorylated phosphorylation site. The second box is optional and carries the identification of the state variable. This identification should be present if confusion is possible between several state variables (e.g., several phosphorylation sites). The center of the combination of the boxes located in the container box is superposed to the center of this container box. Optionally, the identification of the state variable can be located outside the state variable container box. This is strongly discouraged. See Figure 2.13 on the following page for some examples of problems arising if the identification of a state variable is located outside the state variable. The style of labeling of state variables encouraged by SBGN Process Diagram Level 1 is to combine a prefix representing the value of the variable with a suffix representing the variable’s name. Prefix and suffix should be separated by the symbol '@', X@Y thus meaning value X AT variable Y.
Auxiliary items:

A state variable does not carry any auxiliary items.

Figure 2.12: Examples of the Process Diagram 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. However, in all cases, a state variable on an EPN can only take one defined value. Further, an EPN’s state variable should always be displayed and always set to a value. An “empty” state variable is a state variable that is set to the value “unset”, it is not a state variable with no value. Note that the value “unset” is not synonymous to “any value” or “unknown value”.

The label of a state variable should, if possible, be displayed within the ellipse. In the top half of Figure 2.13, we show some examples of pathological cases that lead to confusion in the association between variable labels and values. Compare the discouraged examples with the recommended version in the bottom half of the figure.

Figure 2.13: (Upper part) Examples of incorrect state variables. (Lower part) Correct version.

2.3.12 Glyph: Clone marker

If an EPN is duplicated on a map, it is necessary to indicate this fact by using the clone marker auxiliary unit. The purpose of this marker is to provide the reader with a visual indication that this node has been cloned, and that at least one other occurrence of the EPN can be found in the map (or in a submap; see Section 2.4.3). The clone marker takes two forms, simple and labeled, depending on whether the node being cloned can carry state variables (i.e., whether it is a stateful EPN). Note that an EPN belongs to a single compartment. If two glyphs labelled “X” are located in two different compartments, such as ATP in cytosol and ATP in mitochondrial lumen, they represent different EPNs, and therefore do not need to be marked as cloned.

2.3.12.1 Simple clone marker

As mentioned above, the simple clone marker is the unlabeled version of the clone marker. See below for the labeled version.
CHAPTER 2. PROCESS DIAGRAM GLYPHS

SBO Term:
Not applicable.

Container:
The simple (unlabeled) clone marker is a portion of the surface of an EPN that has been modified visually through the use of a different shade, texture, or color. Figure 2.14 illustrates this. The clone marker occupies the lower part of the EPN. The filled area must be significantly smaller than the unfilled one. A marker’s surface must not exceed 30% of the EPN’s surface. A value in the range 15 and 25% is advised.

Label:
Not applicable.

Auxiliary items:
A clone marker does not carry any auxiliary items.

Figure 2.14: The Process Diagram glyph for simple clone marker applied to a simple chemical, an observable and a multimer of simple chemicals.

2.3.12.2 Labeled clone marker

Unlike the simple clone marker, the labeled clone marker includes (unsurprisingly, given its name) an identifying label that can be used to identify equivalent clones elsewhere in the diagram. This is particularly useful for stateful EPNs, because these can have a large number of state variables displayed and therefore may be difficult to visually identify as being identical.

SBO Term:
Not applicable.

Container:
The labeled clone marker is a portion of the surface of an EPN that has been modified visually through the use of a different shade, texture, or color. The clone marker occupies the lower part of the EPN glyph. The filled area must be significantly smaller than the unfilled one, but the be large enough to have a height larger than the clone marker’s label (cf below). A marker’s surface must not exceed 30% of the EPN’s surface. A value in the range 15 and 25% is advised. Figure 2.15 on the next page illustrates this.

Label:
A clone marker 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 portion of the surface of the EPN that has been modified visually). The font color of the label and the color of the clone marker should contrast with one another. The label on a labeled clone marker is mandatory.

Auxiliary items:
A clone marker does not carry any auxiliary items.
Figure 2.15: The Process Diagram glyph for labeled clone marker applied to a macromolecule, a nucleic acid feature and a multimer of macromolecules.

Figure 2.16: An example of using cloning, here for the species ATP and ADP.

2.3.13 Examples of complex EPNs

In this section, we provide examples of Entity Pool Node representations drawn using the SBGN Process Diagram Level 1 glyphs described above. The first is a representation of the calcium/calmodulin kinase II, with phosphorylation on the sites threonine 286 and 306, as well as catalytic and autoinhibitory domains. This is shown in Figure 2.17. Note the use of units of information and state variables.

Figure 2.17: An example representation of calcium/calmodulin kinase II.

The next EPN example is a representation of the glutamate receptor in the open state, with both phosphorylation and glycosylation. The entity carries two functional domains, the
CHAPTER 2. PROCESS DIAGRAM GLYPHS

ligand-binding domain and the ion pore. Figure 2.18 gives the diagram.

![Diagram of a glutamate receptor in the open state.](image)

Figure 2.18: An example of a glutamate receptor in the open state.

2.4 Container nodes

Containers are SBGN constructions that contain one or several other SBGN constructs. There are three container nodes in SBGN Process Diagram Level 1: complex, compartment, and submap.

2.4.1 Glyph: Complex

A complex node represents a biochemical entity composed of other biochemical entities, whether macromolecules, simple chemicals, multimers, or other complexes. The resulting entity may have its own identity, properties and function in an SBGN diagram.

SBO Term:

SBO:0000253 ! non-covalent complex

Container:

A complex possesses its own container box surrounding the juxtaposed container boxes of its components. This container box is a rectangle with cut-corners (an octagonal box with sides of two different lengths). The size of the cut-corners are adjusted so that there is no overlap between the container and the components. The container boxes of the components must not overlap.

Label:

The identification of a named complex is carried by an unbordered box containing a string of characters. The characters may be distributed on several lines to improve readability, although this is not mandatory. The label box has to be attached to the midway between the border of the complex’s container box and the border of the components’ container boxes.

Auxiliary items:

A complex can carry state variables (see Section 2.3.11). The state of a complex is defined by the set of the all its state variable and all the state variables of all its components. A complex can also carry one or several units of information (see Section 2.3.10). Those units of information can characterize a domain, such as a binding site. Particular units of information carry the material type and the conceptual type of the macromolecules. A complex may carry a clone marker (see Section 2.3.12).
2.4.2 Glyph: Compartment

In order to describe biochemical and cellular events, it is useful to define the notion of pools. A pool is an ensemble of participants that can be considered to be identical for the events in which they are involved. A compartment is a logical or physical structure that contains pools. A pool can only belong to one compartment. Therefore, the “same” biochemical species located in two different compartments are in fact two different pools.

SBO Term:

SBO:0000289 ! functional compartment

Container:

A compartment is represented by a surface enclosed in a continuous border or located between continuous borders. These borders should be noticeably thicker than the borders of the EPNs. A compartment can take any geometry. A compartment must always be entirely enclosed.

Label:

The identification of the compartment is carried by 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 can be attached anywhere in the container box. Note that the label can spill-over from the container box.

Auxiliary items:

A compartment can carry a certain number of units of information, that will add information for instance about the physical environment, such as pH, temperature or voltage, see Section 2.3.10. The center of the bounding box of a unit of information is located on the mid-line of the border of the compartment.

Figure 2.20: The Process Diagram glyph for compartment.
It is important to note that a compartment never contains another compartment, but may surround it. A key aspect of correctly drawing two “adjacent” compartments is that they are not separated by one line, but by two lines. Figure 2.21 provides an example of this in which a cell is shown made up of a nucleus surrounded by the cytoplasm.

![Figure 2.21](image)

**Figure 2.21:** Compartments can surround other compartments; in that case, both of the compartment’s borders must still be shown, with the result that the separation is drawn as two lines.

The example diagram in Figure 2.22 represents three adjacent compartments. Two of the compartments carry units of information. Notice that these units of information do not overlap multiple membrane boundaries.

![Figure 2.22](image)

**Figure 2.22:** Illustration of units of information and surrounding compartments.

To allow more aesthetically pleasing and understandable diagrams, compartments are allowed to overlap each other visually, but it must be kept in mind that this does not mean the top compartment contains part of the bottom compartment. Figure 2.23 on the next page shows two semantically equivalent placement of compartments:
Overlapped (hidden) part of the compartment should not contain any object which could be covered by an overlapping compartment. Figure 2.24 illustrates the problem using an incorrect diagram.

A submap is used to encapsulate processes (including all types of nodes and edges) within one glyph. The submap hides its content to the users, and display only input terminals (or ports), linked to EPNs (Section 2.3) or container nodes (Section 2.4). A submap is not equivalent to an omitted process (see Section 2.5.2). In the case of an SBGN diagram that is made available through a software tool, the content of a submap may be available to the tool. A user could then ask the tool to expand the submap, for instance by clicking on the icon for the submap. The tool might then expand and show the submap within the same diagram (on the same canvas), or it might open it in a different canvas.

**SBO Term:**
To be determined.

**Container:**
The submap is represented as a square box to remind the viewer that it is fundamentally a process.

**Label:**
The identification of the submap is carried by an unbordered box containing a string of characters. The characters may be distributed on several lines to improve readability, although this is not mandatory. The label box has to be attached to the center of the container box.

**Auxiliary items:**
A submap carries labeled terminals. When the submap is represented folded, those terminals are linked to external EPNs (Section 2.3) or containers (Section 2.4). In the unfolded view, exposing the internal structure of the submap, a set of tags point to the corresponding internal EPNs Section 2.3 or containers (Section 2.4).

![Figure 2.25: The Process Diagram glyph for submap. (Upper part) folded submap. (Lower part) content of the submap.](image)

Figure 2.25: The Process Diagram glyph for submap. (Upper part) folded submap. (Lower part) content of the submap.

Figure 2.26 represents a submap that transforms glucose into fructose-6-phosphate. The submap carries five terminals, four linked to EPNs and one linked to a compartment. The latter is particularly important in the case of EPNs present only in a compartment enclosed in a submap, and that are not linked to terminals themselves. Note that the terminals do not define a “direction”, such as input or output. The flux of the reactions is determined by the context.

![Figure 2.26: Example of a submap with contents elided.](image)

Figure 2.26: Example of a submap with contents elided.

The diagram in Figure 2.27 on the following page represents an unfolded version of a submap. Here, anything outside the submap has disappeared, and the internal tags are not linked to the corresponding external terminals. Note the tag 5, linking the compartment “mito” of the submap to the compartment “mito” outside the submap. The compartment containing Glu6P is implicitly defined as the same as the compartment containing Glu and Fru6P. There is no ambiguity because if Glu and Fru6P were in different compartments, one of them should have been defined within the submap.
2.5 Process nodes

Process nodes represent processes that transform one or several EPNs into one or several different EPNs. SBGN Process Diagram Level 1 defines a generic transition (Section 2.5.1), as well as four more specific ones: the omitted process (Section 2.5.2), the uncertain process (Section 2.5.3), the association (Section 2.5.4) and the dissociation (Section 2.5.5). In future levels of the SBGN Process Diagram notation, more transitions may be defined. (One can even envision the development of a controlled vocabulary of transitions, as is done now for EPNs; see Section 2.2.)

2.5.1 Glyph: Transition

A transition is a process transforming a set of entities (represented by EPNs in SBGN Process Diagram Level 1) into another set entities.

**SBO Term:**

SBO:0000167 ! reaction

**Origin:**

One or several consumption arcs (Section 2.6.1) or one or several production arcs (Section 2.6.2).

**Target:**

One or several production arcs (Section 2.6.2).

**Node:**

A transition is represented by a square box linked to two connectors, small arcs attached to the centers of opposite sides. The consumption (Section 2.6.1) and production (Section 2.6.2) arcs are linked to the extremities of those connectors. The modulatory arcs (Section 2.6) point to the other two sides of the box. A transition connected to production arcs on opposite sides is a reversible transition. The connectors and the box move as a rigid entity.
A transition is the basic process node in SBGN. It describes a process that transforms a given set of biochemical entities—macromolecules, simple chemicals or unspecified entities—into another set of biochemical entities. Such a transformation might imply modification of covalent bonds (conversion), modification of the relative position of constituents (conformational transition) or movement from one compartment to another (translocation).

A cardinality label may be associated with consumption (Section 2.6.1) or production (Section 2.6.2) arcs to indicate the stoichiometry of the process. This label becomes a requirement when the exact composition of the number of copies of the inputs or outputs to a reaction are ambiguous in the diagram.

The example in Figure 2.29 illustrates the use of a transition node to represent the phosphorylation of a protein in a Process Diagram.

![Figure 2.29: Phosphorylation of a protein.](image)

The example in Figure 2.30 illustrates the use of a transition node to represent a reaction between two reactants that generates three products.

![Figure 2.30: A reaction that generates three products.](image)

The example in Figure 2.31 illustrates the use of a transition node to represent a translocation. The large round-cornered rectangle represents a compartment border (see Section 2.4.2).

![Figure 2.31: A translocation.](image)

The example in Figure 2.32 on the following page illustrates the use of a transition node to represent the reversible opening and closing of an ionic channel in a Process Diagram.
CHAPTER 2. PROCESS DIAGRAM GLYPHS

2.32: Reversible opening and closing of an ionic channel.

When such a reversible process is asymmetrically modulated, it should be represented by two different processes in a Process Diagram. Figure 2.33 illustrates the use of two transition nodes to represent the reversible activation of a G-protein coupled receptor. In the absence of any effector, an equilibrium exists between the inactive and active forms. The agonist stabilises the active form, while the inverse agonist stabilises the inactive form.

2.33: The reversible activation of a G-protein coupled receptor.

The example in Figure 2.34 presents the conversion of two galactoses into a lactose. Galactoses are represented by only one simple chemical, the cardinality being carried by the consumption arc.

2.34: Conversion of two galactoses into a lactose.

2.5.2 Glyph: Omitted process

Omitted processes are processes that are known to exist, but are omitted from the diagram for the sake of clarity or parsimony. A single omitted process can represent any number of actual processes. The omitted process is different from a submap. While a submap possess an explicit content that is hidden in the main map, the omitted process does not “hide” anything within the context of the diagram, and cannot be “unfolded”.

SBO Term:
To be determined.

Origin:
One or several consumption arcs (Section 2.6.1) or one or several production arcs (Section 2.6.2).

Target:
One or several production arcs (Section 2.6.2).
Chapter 2. Process Diagram Glyphs

Node:
Omitted processes are represented as a transition in which the square box contains a two parallel slanted lines oriented northwest-to-southeast and separated by an empty space.

Figure 2.35: The Process Diagram glyph for omitted.

2.5.3 Glyph: Uncertain process
Uncertain processes are processes that may not exist. A single uncertain process can represent any number of actual processes.

SBO Term:
To be determined.

Origin:
One or several consumption arcs (Section 2.6.1) or one or several production arcs (Section 2.6.2).

Target:
One or several production arcs (Section 2.6.2).

Node:
Uncertain processes are represented as a transition which square box contains a question mark.

Figure 2.36: The Process Diagram glyph for an uncertain process.

2.5.4 Glyph: Association
The association between one or more EPNs represents the non-covalent binding of the biological objects represented by those EPNs into a larger complex.

SBO Term:
SBO:0000177 ! binding.

Origin:
One or more consumption arcs (Section 2.6.1).

Target:
One production arc (Section 2.6.2).

Node:
An association between several entities is represented by a filled disc linked to two connectors, small arcs attached on point separated by 180 degrees. The consumption (Section 2.6.1) and production (Section 2.6.2) arcs are linked to the extremities of those connectors. The connectors and the disc move as a rigid entity.
CHAPTER 2. PROCESS DIAGRAM GLYPHS

The example in Figure 2.38 illustrates the association of cyclin and CDC2 kinase into the Maturation Promoting Factor.

Figure 2.38: Association of cyclin and CDC2 kinase into the Maturation Promoting Factor.

Figure 2.39 gives an example illustrating the association of a pentameric macromolecule (a nicotinic acetylcholine receptor) with a simple chemical (the local anesthetic chlorpromazin) in an unnamed complex.

Figure 2.39: The association of a pentameric macromolecule with a simple chemical in an unnamed complex.

An association does not necessarily result in the formation of a complex; it can also produce a multimer, or a macromolecule (although the latter case is semantically borderline). Figure 2.40 gives an example of this, using the formation of hemoglobin.

Figure 2.40: Formation of hemoglobin.
2.5.5 Glyph: **Dissociation**

The dissociation of an *EPN* into one or more *EPNs* represents the rupture of a non-covalent binding between the biological entities represented by those *EPNs*.

**SBO Term:**

SBO:0000180 ! dissociation.

**Origin:**

One *consumption* arc (Section 2.6.1).

**Target:**

One or more *production* arc (Section 2.6.2).

**Node:**

A *dissociation* between several entities is represented by two concentric circles. A simple empty disc could be, in some cases, confused with the *catalysis* (section Section 2.6.5). Moreover, the existence of two circles reminds the dissociation, by contrast with the filled disc of the *association* (Section 2.5.4).

![Figure 2.41: The Process Diagram glyph for dissociation.](image)

The example in Figure 2.42 illustrates the dissociation of the small and large ribosomal subunits from a messenger RNA.

![Figure 2.42: Dissociation of the small and large ribosomal subunits from a messenger RNA.](image)

2.6 Connecting arcs

Connecting arcs are lines that link *EPNs* and *PNs* together. The symbols attached to their extremities indicate their semantics.

2.6.1 Glyph: **Consumption**

*Consumption* is the arc used to represent the fact that an entity affects a process, but is not affected by the process.

**SBO Term:**

To be determined.
Origin:
Any EPN (Section 2.3).

Target:
Any process node (Section 2.5).

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

A cardinality label may be associated with consumption (Section 2.6.1) or production (Section 2.6.2) arcs, indicating the stoichiometry of a process. This label is a number enclosed in a rectangle with one of the long sides adjacent to the consumption arc. The cardinality is required to eliminate ambiguity when the exact composition, or the number of copies, of the inputs or outputs to a reaction are ambiguous from the diagram. An example is a multimer of 6 subunits dissociating into 2 monomers and 2 dimers. Without stoichiometry labels another result, such as 4 monomers and 1 dimer could be inferred. Once assigned to one arc connecting to a transition node, cardinality should be represented on all consumption and production arcs connected to that transition node to avoid misinterpretation.

Omitted cardinality on one edge only should not be treated as cardinality of 1, but as an unspecified cardinality. In most cases, the exact value may be derived from the context, but unless cardinality is explicitly shown, it should be considered as unspecified. In the case where the stoichiometry of some part of the process is not known, or undefined, a question mark (?) should be used within the cardinality label of the corresponding arcs.

```
\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{consumption_glyph}
\caption{The Process Diagram glyph for consumption.}
\end{figure}
```

2.6.2 Glyph: Production

Production is the arc used to represent the fact that an entity is produced by a process. In the case of a reversible transition, the production arc also acts as a consumption arc.

SBO Term:
To be determined.

Origin:
Any process node (Section 2.5).

Target:
Any EPN (Section 2.3).

End point:
The target extremity of a production carries a filled arrowhead.

A cardinality label can be associated with a production arc indicating the stoichiometry of a process.

```
\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{production_glyph}
\caption{The Process Diagram glyph for production.}
\end{figure}
```
CHAPTER 2. PROCESS DIAGRAM GLYPHS  

Figure 2.45 illustrates the use of consumption/production arc cardinality labels to represent the stoichiometry of a process.

![Cardinality for production arcs.](image)

2.6.3 Glyph: Modulation

A modulation affects the flux of a process represented by the target transition. Such a modulation can affect the process **positively or negatively**, or even both ways depending on the conditions, for instance the concentration of the intervening participants. A modulation can also be used when one does not know the precise direction of the effect.

**SBO Term:**
- SBO:0000168 ! control.

**Origin:**
- Any EPN (Section 2.3) or any logical operator (Section 2.7).

**Target:**
- Any process node (Section 2.5).

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

![The Process Diagram glyph for modulation.](image)

Figure 2.47 on the following page represents the effect of nicotine on the transition between closed and open states of a nicotinic acetylcholine receptor. High concentrations of nicotine
open the receptor while low concentrations can desensitize it without opening.

![Figure 2.47: Modulation of nicotinic receptor opening by nicotine.](image)

### 2.6.4 Glyph: Stimulation

A stimulation affects **positively** the flux of a process represented by the target transition. This stimulation can be for instance a catalysis or a positive allosteric regulation. Note that *catalysis* exists independently in SBGN, see Section 2.6.5.

**SBO Term:**

SBO:0000170 ! stimulation.

**Origin:**

Any *EPN* (Section 2.3) or any logical operator (Section 2.7).

**Target:**

Any *process node* (Section 2.5).

**End point:**

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

![Figure 2.48: The Process Diagram glyph for stimulation.](image)

### 2.6.5 Glyph: Catalysis

A catalysis is a particular case of stimulation, where the effector affects positively the flux of a process represented by the target transition. The positive effect on the transition is due to the lowering of the activation energy of a reaction.

**SBO Term:**

SBO:0000172 ! catalysis.
Origin:
Any EPN (Section 2.3) or any logical operator (Section 2.7).

Target:
Any process node (Section 2.5).

Node:
The target extremity of a catalysis carries an empty circle.

Figure 2.49: The Process Diagram glyph for catalysis.

2.6.6 Glyph: Inhibition

An inhibition negatively affects the flux of a process represented by the target transition. This inhibition can be for instance a competitive inhibition or an allosteric inhibition.

SBO Term:
SBO:0000169 ! inhibition.

Origin:
Any EPN (Section 2.3) or any logical operator (Section 2.7).

Target:
Any process node (Section 2.5).

Node:
The target extremity of an inhibition carries a bar perpendicular to the arc.

Figure 2.50: The Process Diagram glyph for inhibition.

2.6.7 Glyph: Trigger

A trigger effect, or absolute stimulation, is a stimulation that is necessary for a process to take place. A process modulated by a trigger can only occur when this trigger is active.

SBO Term:
SBO:0000171 ! necessary stimulation.

Origin:
Any EPN (Section 2.3) or any logical operator (Section 2.7).

Target:
Any process node (Section 2.5).

Node:
The target extremity of a trigger carries an open arrow (to remind that it is a stimulation) coming after a larger vertical bar.
The example in Figure 2.52 below describes the transcription of a gene X, that is the creation of a messenger RNA X triggered by the gene X. The creation of the protein X is then triggered by the mRNA X. (Note that the same example could be represented using the gene as reactant and product, although it is semantically different.)

The example in Figure 2.53 below describes the transport of calcium ions out of the endoplasmic reticulum. Without IP3 receptor, there is not calcium flux, therefore, one cannot use a stimulation. The trigger instead represents this absolute stimulation.

2.6.8 Glyph: Logic arc

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

**SBO Term:**

To be determined.
CHAPTER 2. PROCESS DIAGRAM GLYPHS

Origin:
Any EPN (Section 2.3) or logical operator (Section 2.7).

Target:
Any logical operator (Section 2.7).

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

2.6.9 Glyph: Equivalence arc
Equivalence Arc is the arc used to represent the fact that all entities marked by a tag are equivalent.

SBO Term:
To be determined.

Origin:
Any EPN (Section 2.3).

Target:
Tag (Section 2.3.9).

End point:
No particular symbol is used to represent an equivalence arc.

2.7 Logical operators

2.7.1 Glyph: And
The glyph and is used to denote that all the EPNs linked as input are necessary to produce the output.

SBO Term:
SBO:0000173 ! and.

Origin:
More than one EPN (section 2.3) or logical operator (section 2.7).

Target:
One modulation (section 2.6.3), stimulation (section 2.6.4), catalysis (section 2.6.5), inhibition (section 2.6.6) or trigger (section 2.6.7) arc.
Node: And is represented by a circle carrying the word “AND”.

![AND](image)

Figure 2.56: The Process Diagram glyph for and. Only two inputs are represented, but more would be allowed.

2.7.2 Glyph: Or

The glyph or is used to denote that any of the EPNs linked as input is sufficient to produce the output.

SBO Term:
SBO:0000174 ! or.

Origin:
More than one EPN (section 2.3) or logical operator (section 2.7).

Target:
One modulation (section 2.6.3), stimulation (section 2.6.4), catalysis (section 2.6.5), inhibition (section 2.6.6) or trigger (section 2.6.7) arc.

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

![OR](image)

Figure 2.57: The Process Diagram glyph for or. Only two inputs are represented, but more would be allowed.

2.7.3 Glyph: Not

The glyph not is used to denote that the EPN linked as input cannot produce the output.

SBO Term:
SBO:0000238 ! not.

Origin:
One EPN (section 2.3) or logical operator (section 2.7).

Target:
One modulation (section 2.6.3), stimulation (section 2.6.4), catalysis (section 2.6.5), inhibition (section 2.6.6) or trigger (section 2.6.7) arc.

Node:
Not is represented by a circle carrying the word “NOT”.
Figure 2.58: The Process Diagram glyph for not.
Chapter 3

Process Diagram grammar

3.1 Overview

In this chapter, we describe how the glyphs of SBGN can be combined to make a valid SBGN 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 biological pathway? This is semantics, and it is essential if a reader is to understand a 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 Process Diagram 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 Process Diagram 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.

The Process Diagram in SBGN describes biological processes involving biological entities. An Entity pool node (Section 2.3), such as a molecule, is converted into other EPNs via a process. It may be convenient to think of a SBGN Process Diagram as describing the flow of a fluid. In this analogy, each EPN represents a tank of fluid, which may be emptied via a pipe (the consumption arc, Section 2.6.1) that is connected to a valve (the process node, Section 2.5), which in turn is connected to other pipes (the production arcs, Section 2.6.2) that fill other tanks (EPNs). A valve can be controlled in relation to the amount of fluid in another tank. This precise nature of the relationship between valve opening and amount of fluid determines the nature of the modulation.

Finally, the system needs a source of fluid from one or more external sources (source, section Section 2.3.6) and somewhere for it to drain away (the sink).

3.3 The conceptual model

In order to formalize the conceptual representation described above and the semantic rules that we will describe later, we have developed a UML Domain Object Model of SBGN. This defines what would otherwise be vaguely defined in the SBGN specification, and enables us to define clearly what is a valid SBGN diagram.
The model is separated into two layers. The Visual Layer (Figure 3.1 on the next page) directly maps to the glyphs used to visualise SBGN and reflects the syntax of the notation. The Conceptual Layer describes what is “meant” by the visual SBGN representation (Figure 3.2 on page 43). It is important in defining concepts such as Entity types and state variables that are implicit in the Visual Layer.
Figure 3.1: UML model of SBGN visual representation. Class names shown in italics are abstract classes and do not correspond to a glyph. Classes in non-italicised script with the suffixes "Node" and "Arc" correspond to node and arc glyphs respectively.
Figure 3.2: UML model of SBGN conceptual representation. The main difference between visual and conceptual representations is that the conceptual model is non-redundant with only one instance of an EntityPool with a given identity. Each class has a mapping back to the Visual Layer.
The Classes shown in the UML model are described in more detail in the tables below. In particular the tables define the properties that uniquely identify each component of the notation. This is important in defining the rules for duplication and the submap interface later.
Table 3.1: A data dictionary for the Visual Layer of the SBGN Conceptual Model. Attributes that uniquely identify a class are followed by (I) and required attributes by (R); it is implied that identifying attributes are also required. The default cardinality of an attribute is one (or zero or one if it is not required) unless otherwise specified. A standard notation for cardinality is used, for example {0...*} should be read zero, one or many. The Glyph column states the Glyph the class maps to, unless the class is purely conceptual and so defined as Abstract.

<table>
<thead>
<tr>
<th>Visual Layer Data Dictionary</th>
<th>Name</th>
<th>Attributes</th>
<th>Glyph</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>MapDiagram</td>
<td>name (I)</td>
<td>Name of the map.</td>
<td>N/A</td>
<td>The diagram is the visualisation of a SBGN map.</td>
</tr>
<tr>
<td>SubmapDiagram</td>
<td>mainMapName (I)</td>
<td>Name of the main map this is a sub-map of.</td>
<td>N/A</td>
<td>A SBGN diagram used to define a SBGN submap</td>
</tr>
<tr>
<td>BasicEntityNode</td>
<td>name (I)</td>
<td>Name of the sub-map.</td>
<td>Abstract</td>
<td>The most fundamental node type. Any node that can be drawn on a MapDiagram inherits from this.</td>
</tr>
<tr>
<td>ConsumableNode</td>
<td></td>
<td></td>
<td>Abstract</td>
<td>Nodes that can be consumed and are connected at the start of a consumption arc.</td>
</tr>
<tr>
<td>ProducableNode</td>
<td></td>
<td></td>
<td>Abstract</td>
<td>Nodes that can produced and are linked to the end of a production arc.</td>
</tr>
<tr>
<td>ModulatableNode</td>
<td>modulatingNodes (I) {0...*}</td>
<td>Any number of nodes that modulate this one.</td>
<td>Abstract</td>
<td>Nodes representing a process that can be modulated and so are linked to the end of a modulating arc.</td>
</tr>
<tr>
<td>ModulatingNode</td>
<td></td>
<td></td>
<td>Abstract</td>
<td>Nodes that can modulate modulatable nodes. They are linked to the origin of a modulating arc.</td>
</tr>
<tr>
<td>CompartmentNode</td>
<td>name (I)</td>
<td>Name of compartment being defined. At present the name of the compartment is not restricted, i.e. a controlled vocabulary is not used, any number of Units of Information.</td>
<td>Compartment</td>
<td></td>
</tr>
<tr>
<td>EntityPoolNode (EPN)</td>
<td>name (I)</td>
<td>The name of the entity. If no compartment is specified then this is implicitly defined as the “default” compartment. Any number of Units of Information.</td>
<td>Abstract</td>
<td>Nodes that inherit from this type are generally chemical species that can be produced, consumed or transformed in chemical processes. It refers to a population (or pool) of such species.</td>
</tr>
<tr>
<td>StatefulEntityPoolNode</td>
<td>As EPN</td>
<td>Any number of StateDescriptions associated with this EPN.</td>
<td>Abstract</td>
<td>All EPN that inherit from this class may have state variables assigned to them.</td>
</tr>
<tr>
<td>SimpleChemicalNode</td>
<td>As EPN</td>
<td></td>
<td>SimpleChemical</td>
<td></td>
</tr>
</tbody>
</table>

continued on next page
### Name

<table>
<thead>
<tr>
<th>Name</th>
<th>Attributes</th>
<th>Glyph</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnspecifiedEntity-Node</td>
<td><strong>As EPN</strong></td>
<td><strong>UnspecifiedEntity</strong></td>
<td>The SourceNode can only be associated with one consumption arc. Note that this shares the same Glyph as SinkNode.</td>
</tr>
<tr>
<td>SourceNode</td>
<td><strong>As EPN</strong></td>
<td><strong>Source/Sink</strong></td>
<td>The SourceNode can only be associated with one consumption arc. Note that this shares the same Glyph as SinkNode.</td>
</tr>
<tr>
<td>SinkNode</td>
<td><strong>As EPN</strong></td>
<td><strong>Source/Sink</strong></td>
<td>The SinkNode can only be associated with one production arc. Note that this shares the same Glyph as SourceNode.</td>
</tr>
<tr>
<td>ObservableNode</td>
<td>name (I)</td>
<td><strong>Observable</strong></td>
<td></td>
</tr>
<tr>
<td>PerturbationNode</td>
<td>name (I)</td>
<td><strong>Perturbation</strong></td>
<td></td>
</tr>
<tr>
<td>MacromoleculeNode</td>
<td><strong>As StatefulEntityPoolNode</strong></td>
<td><strong>Macromolecule</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>macromoleculeType (I)</td>
<td></td>
<td>The type of the macromolecule. Currently there are no constraints on permitted types.</td>
</tr>
<tr>
<td>NucleicAcidFeature-Node</td>
<td><strong>As StatefulEntityPoolNode</strong></td>
<td><strong>NucleicAcidFeature</strong></td>
<td></td>
</tr>
<tr>
<td>ComplexNode</td>
<td><strong>As StatefulEntityPoolNode</strong></td>
<td><strong>Complex</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>subunits (I) [0 . . . *, 0 . . . *]</td>
<td></td>
<td>A list of EPNs that together compose the complex. The set of state descriptions of any Sub-units that are StatefulEntityPoolNodes.</td>
</tr>
<tr>
<td>Multimer-MacromoleculeNode</td>
<td><strong>As MacromoleculeNode</strong></td>
<td><strong>Multimer-Macromolecule</strong></td>
<td>Note that all sub-units have the same StateDescription values.</td>
</tr>
<tr>
<td></td>
<td>numberOfSubunits (I)</td>
<td></td>
<td>The number of identical sub-units of this macromolecule.</td>
</tr>
<tr>
<td>MultimerSimple-ChemicalNode</td>
<td><strong>As SimpleChemicalNode</strong></td>
<td><strong>MultimerSimple-Chemical</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>numberOfSubunits (I)</td>
<td></td>
<td>The number of identical sub-units of this simple chemical.</td>
</tr>
<tr>
<td>MultimerComplex-Node</td>
<td><strong>As ComplexNode</strong></td>
<td><strong>MultimerComplex</strong></td>
<td>Note that all sub-units have the same StateDescription values.</td>
</tr>
<tr>
<td></td>
<td>numberOfSubunits (I)</td>
<td></td>
<td>The number of identical sub-units of this complex.</td>
</tr>
<tr>
<td>UnitOfInformation</td>
<td>owningEPN (I)</td>
<td><strong>Unit Of Information</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>prefix</td>
<td></td>
<td>Identity of the owning EntityPoolNode (not displayed, but implied by “ownership” of the glyph).</td>
</tr>
<tr>
<td></td>
<td>annotation</td>
<td></td>
<td>PREFIX (I) Annotation (R)</td>
</tr>
<tr>
<td>StateDescription</td>
<td>owningStatefulEPN</td>
<td><strong>State Variables</strong></td>
<td>Identity of the owning StatefulEntityPoolNode (not displayed, but implied by “ownership” of the glyph).</td>
</tr>
</tbody>
</table>

**continued on next page**
<table>
<thead>
<tr>
<th>Name</th>
<th>Attributes</th>
<th>Glyph</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable</td>
<td>The name (identification) of the “state variable” glyph associated a StatefulEntityPoolNode. The definition of a value for the given variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ProcessNode</td>
<td>As ModulatableNode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>producableNodes(I)</td>
<td>The ProducableNodes linked to the process by a ProductionArc.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>consumableNodes(I)</td>
<td>The ConsumableNodes linked to the process by a ConsumptionArc.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OmittedProcessNode</td>
<td>As Process Node</td>
<td></td>
<td>Process</td>
</tr>
<tr>
<td>UncertainProcessNode</td>
<td>As Process Node</td>
<td></td>
<td>Omitted Process</td>
</tr>
<tr>
<td>AssociationNode</td>
<td>As Process Node</td>
<td></td>
<td>Association</td>
</tr>
<tr>
<td>DissociationNode</td>
<td>As Process Node</td>
<td></td>
<td>Dissociation</td>
</tr>
<tr>
<td>LogicGateNode</td>
<td>gateType(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>modulatingNodes(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>modulatedNode(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANDNode</td>
<td>As LogicGateNode</td>
<td></td>
<td>And</td>
</tr>
<tr>
<td>ORNode</td>
<td>As LogicGateNode</td>
<td></td>
<td>Or</td>
</tr>
<tr>
<td>NOTNode</td>
<td>As LogicGateNode</td>
<td></td>
<td>Not</td>
</tr>
<tr>
<td>SubMapNode</td>
<td>subMapName(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>compartment[R]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TagNode</td>
<td>subMapName(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>identifier(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SubMapTerminalNode</td>
<td>subMapName(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>identifier(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ProductionArc</td>
<td>processNode(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>producableNode(I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConsumptionArc</td>
<td>consumableNode(I)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

continued on next page
### Table 3.2: A data dictionary for the Conceptual Layer of the SBGN Model. Attributes that uniquely identify a class are followed by (I) and required attributes by (R): it is implied that identifying attributes are also required. The default cardinality of an attribute is one (or zero or one if it is not required) unless otherwise specified. A standard notation for cardinality is used, for example \{0 . . . *\} should be read zero, one or many. A mapping to a class in the Visual Layer is provided where appropriate.
<table>
<thead>
<tr>
<th>Name</th>
<th>Attributes</th>
<th>Description</th>
<th>Visual Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EntityType</strong></td>
<td>Name (I)</td>
<td>The name of the EntityType.</td>
<td>This is the type of the EntityPool.</td>
</tr>
<tr>
<td></td>
<td>type (I)</td>
<td>The type of the Entity. One of: SimpleChemical, UnspecifiedEntity, Complex,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Macromolecule, and NucleicAcidFeature. Annotation must be associated with the</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>type and be general to all Entity Pool Node. If it were associated with the</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Species then this would be synonymous with a state description.</td>
<td></td>
</tr>
<tr>
<td><strong>MultimericEntityType</strong></td>
<td>As EntityType</td>
<td>An Entity that can form homo-multimeric complexes. All copies share the same</td>
<td></td>
</tr>
<tr>
<td></td>
<td>cardinality (I)</td>
<td>The number of subunits in the monomer.</td>
<td></td>
</tr>
<tr>
<td><strong>StatefulEntityType</strong></td>
<td>As EntityType</td>
<td>An EntityType that can have state variables associated with it.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stateVariables 0 . . . *</td>
<td>The set of EntityTypes that the complex composed of.</td>
<td></td>
</tr>
<tr>
<td><strong>ComplexEntityType</strong></td>
<td>As StatefulEntityType</td>
<td>The set of EntityTypes that the complex composed of.</td>
<td>Implicitly defined by a ComplexNode.</td>
</tr>
<tr>
<td></td>
<td>cardinality (I)</td>
<td>The number of subunits in the monomer.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>subunits 0 . . . *</td>
<td>The set of EntityTypes that the complex composed of.</td>
<td>Implicitly defined by a MacromoleculeNode and MultimerMacromoleculeNode</td>
</tr>
<tr>
<td><strong>MacromoleculeEntity-</strong></td>
<td>macromoleculeType (I)</td>
<td>The type of the macromolecule. Suggest types are protein, mRNA, miRNA,</td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td></td>
<td>trNA, rRNA, DNA, polysaccharide. See also controlled vocabularies for material</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>type 2.2.1 and conceptual type 2.2.2</td>
<td></td>
</tr>
<tr>
<td><strong>StateVariable</strong></td>
<td>name (I)</td>
<td>The name of the state variable.</td>
<td>The state variable is assigned to a StatefulEntityType and can have one or</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>more values assigned to it when used in a StatefulEntityPool.</td>
</tr>
<tr>
<td><strong>StateValue</strong></td>
<td>value (I)</td>
<td>The value of the state.</td>
<td>A value that can be assigned to a state variable.</td>
</tr>
<tr>
<td><strong>EntityPool</strong></td>
<td>entityType (I)</td>
<td>The EntityType of the pool.</td>
<td>An instantiation of an EntityType.</td>
</tr>
<tr>
<td><strong>StatefulEntityPool</strong></td>
<td>As EntityPool stateValueSets 1 . . . *</td>
<td>A set of state values assigned to a the StateVariables of the associated StatefulEntityType.</td>
<td>StatefulEntityPoolNode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ComplexEntityPool</strong></td>
<td>As StatefulEntity subunitValues (I) 0 . . . *</td>
<td>The set of StateValues for all subunits.</td>
<td>The instantiation of a complex that is defined by it’s own state and the states of its subunits.</td>
</tr>
</tbody>
</table>

continued on next page
<table>
<thead>
<tr>
<th>Name</th>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annotation</td>
<td>prefix (I)</td>
<td>A classifier for the annotation. Annotation that is associated with a Entity-Type or Compartment.</td>
</tr>
<tr>
<td></td>
<td>annotation</td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td>process (I)</td>
<td>The process that this is the output of.</td>
</tr>
<tr>
<td></td>
<td>product (I)</td>
<td>The EPN that is the product of the process.</td>
</tr>
<tr>
<td>Input</td>
<td>substrate (I)</td>
<td>The EPN that is the input to the process.</td>
</tr>
<tr>
<td></td>
<td>process (I)</td>
<td>The process that this is the input of.</td>
</tr>
<tr>
<td>Control</td>
<td>type (R)</td>
<td>The type of the control. One of Modulation, Stimulation, Inhibition, Trigger, Catalysis or LogicGate.</td>
</tr>
<tr>
<td>LogicGate</td>
<td>type (R)</td>
<td>The type of LogicGate. One of AND, OR, NOT.</td>
</tr>
<tr>
<td>LogicAdapter</td>
<td></td>
<td>The LogicGate takes a set of Boolean inputs (T or F) from an EntityPool which is a non-Boolean quantity. Conceptually, the adapter converts the continuous quantity into a discrete Boolean quantity.</td>
</tr>
<tr>
<td>Perturbation</td>
<td>name</td>
<td>Name of the perturbation.</td>
</tr>
<tr>
<td>Observable</td>
<td>name</td>
<td>Name of the observable.</td>
</tr>
<tr>
<td>Source</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sink</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CollapsedSubMap</td>
<td>referredSubMap (I)</td>
<td>Submap that this refers too.</td>
</tr>
<tr>
<td>SubMapEntity-Mapping</td>
<td>identifier (I)</td>
<td>Identifier in Visual layer to link EntityPool-Nodes in the main map and submap. The EntityPool node that is mapped between the submap and the main map.</td>
</tr>
</tbody>
</table>

Visual Mapping:
- UnitOfInformation
- ProductionArc
- ConsumptionArc
- ModulationArc
- LogicGateNode
- LogicArc
- PerturbationNode
- ObservableNode
- SourceNode
- SinkNode
- SubMapTerminalNode and TagNode
3.4 Syntax

The syntax of SBGN State Transition 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 EPNs.

3.4.1 Entity Pool Nodes connectivity definition

<table>
<thead>
<tr>
<th>Arc\EPN</th>
<th>macromolecule</th>
<th>simple chemical</th>
<th>unspecified entity</th>
<th>multimer</th>
<th>complex</th>
<th>nucleic acid feature</th>
<th>tag</th>
<th>source/sink</th>
<th>perturbation</th>
<th>observable</th>
<th>submap</th>
</tr>
</thead>
<tbody>
<tr>
<td>consumption</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>production</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>modulation</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stimulation</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>catalysis</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>inhibition</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>trigger</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>logic arc</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>equivalence arc</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td>O(1)</td>
</tr>
</tbody>
</table>

3.4.2 Process Nodes connectivity definition

<table>
<thead>
<tr>
<th>Arc\PN</th>
<th>transition</th>
<th>omitted process</th>
<th>uncertain process</th>
<th>association</th>
<th>dissociation</th>
<th>and</th>
<th>or</th>
<th>not</th>
</tr>
</thead>
<tbody>
<tr>
<td>consumption</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O(1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>production</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
</tr>
<tr>
<td>modulation</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
</tr>
<tr>
<td>stimulation</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
</tr>
<tr>
<td>catalysis</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
</tr>
<tr>
<td>inhibition</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
</tr>
<tr>
<td>trigger</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
<td>I(1)</td>
</tr>
<tr>
<td>logic arc</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O(1)</td>
<td>O(1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>equivalence arc</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O(1)</td>
<td>O(1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.4.3 Containment definition

There are two EPN types which allows containment in SBGN: compartment and complex. The next table describe relationship between EPN elements of SBGN and these two containers. Plus sign means that the element is able to be contained within a container. An empty cell means containment is not allowed.
### 3.4.4 Syntactic rules

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

1. **EPNs.** That rules are applicable to all EPNs (2.3)
   
   (a) If *macromolecule* has more than one *state variable*, all *state variables* should have a name;

   (b) All *state variables* of the *macromolecule* should have different names;

   (c) *Complex* should consists of different EPNs. If two or more elements of the complex are identical they should be replaced by multimer.

2. **Source/Sink** (2.3.6).
   
   (a) *source/sink* is allowed to have only one link attached to it. It could be either *consumption* or *production* link.

3. **Process.** That rules are applicable to all PNs (2.5)
   
   (a) PN should have nonzero number of *consumption* links.

   (b) PN should have nonzero number of *production* links.
(c) All substrates of the transition should be different. If several copies of the same EPN are involved in the process, cardinality label of consumption arc should be used.

(d) All products of the transition should be different. If several copies of the same EPN are produced in the process, cardinality label of production arc should be used.

(e) Once cardinality label set to one arc of the EPN all other arcs should make their cardinality visible, even if it is undefined. In a case cardinality is undefined or unknown question mark (“?”) should be placed as cardinality label.

(f) The EPN is not allowed to be a substrate and product of the same process. This rule is applicable to cloned EPNs as well.

(g) PN should have only one Catalysis arc connected to it. If there more than one catalyst known for the process several PNs should be drawn.

(h) PN should have only one Trigger arc connected to it. If there more than one trigger known for the process several PNs should be drawn or explicit logic gate should be used.

4. Association

(a) Composition of the complex or multimer produced by association should be identical to set of association substrates.

(b) If association produces complex, than number of consumption arcs should be two or more.

(c) If association produces multimer, than it can have only one consumption arc. In that case if substrate is not a multimer, then the number of monomers in a product multimer should be equal to cardinality of that arc. If substrate is another multimer, then cardinality of the consumption arc should be equal to ratio of number of monomers in product and in substrate multimers.

5. Dissociation

(a) Composition of complex or multimer consumed by dissociation should be identical to set of products of the process.

(b) If complex is consumed by the dissociation, than the process should have two or more production arcs.

(c) If multimer is split by dissociation process, than one production arc could be connected to the process node. In that case if product is not a multimer, then the number of monomers in a substrate multimer should be equal to cardinality of that arc. If product is another multimer, then cardinality of the production arc should be equal to ratio of number of monomers in substrate and in product multimers.

3.5 Semantic rules

3.5.1 Namespaces

The notation has a concept of a namespace within which entities with the same identifying attributes are regarded as identical. The SBGN namespaces are shown in table 3.3.

<table>
<thead>
<tr>
<th>Namespace Scope</th>
<th>Entities affected</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MapDiagram</td>
<td>CompartmentNode, SubMapDiagram, EquivalenceNode</td>
<td></td>
</tr>
</tbody>
</table>

| continued on next page |
CHAPTER 3. PROCESS DIAGRAM GRAMMAR

continued from previous page

<table>
<thead>
<tr>
<th>Namespace scope</th>
<th>Entities Effected</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompartmentShape</td>
<td>BasicEntityNode</td>
<td>If no compartment is drawn then all BasicEntityNodes are assumed to belong to an invisible “default” compartment.</td>
</tr>
<tr>
<td>EntityType</td>
<td>StateVariable, Annotation</td>
<td></td>
</tr>
<tr>
<td>ComplexType</td>
<td>EntityType</td>
<td></td>
</tr>
</tbody>
</table>

3.5.2 Cloning

SBGN only allows identical nodes to duplicated on a map if they are explicitly marked as such. This is using a labeled clone marker with all the nodes that are identical. The details are shown in table 3.4.

Table 3.4: Duplication rules.

<table>
<thead>
<tr>
<th>Node</th>
<th>Can Duplicate?</th>
<th>Indication</th>
<th>Additional Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompartmentNode</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SimpleChemicalNode</td>
<td>Y</td>
<td>Simple clone marker</td>
<td></td>
</tr>
<tr>
<td>UnspecifiedEntityNode</td>
<td>Y</td>
<td>Simple clone marker</td>
<td></td>
</tr>
<tr>
<td>SourceNode</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SinkNode</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perturbation</td>
<td>Y</td>
<td>Simple clone marker</td>
<td></td>
</tr>
<tr>
<td>Observable</td>
<td>Y</td>
<td>Simple clone marker</td>
<td></td>
</tr>
<tr>
<td>MultimerChemicalEntity</td>
<td>Y</td>
<td>Simple clone marker</td>
<td></td>
</tr>
<tr>
<td>StatefulEntityPoolNode</td>
<td>Y</td>
<td>Labeled clone marker</td>
<td></td>
</tr>
<tr>
<td>MacromoleculeNode</td>
<td>Y</td>
<td>Labeled clone marker</td>
<td></td>
</tr>
<tr>
<td>MultimerMacromoleculeNode</td>
<td>Y</td>
<td>Labeled clone marker</td>
<td></td>
</tr>
<tr>
<td>NucleicAcidFeatureNode</td>
<td>Y</td>
<td>Labeled clone marker</td>
<td></td>
</tr>
<tr>
<td>ComplexNode</td>
<td>Y</td>
<td>Labeled clone marker</td>
<td></td>
</tr>
<tr>
<td>TransitionNode</td>
<td>Y</td>
<td>None</td>
<td>Duplication is implied when all EPNs linked to the TransitionNode are marked as clones.</td>
</tr>
<tr>
<td>OmittedProcessNode</td>
<td>Y</td>
<td>As Transition Node</td>
<td></td>
</tr>
<tr>
<td>UncertainProcessNode</td>
<td>Y</td>
<td>As Transition Node</td>
<td></td>
</tr>
<tr>
<td>AssociationNode</td>
<td>Y</td>
<td>As Transition Node</td>
<td></td>
</tr>
<tr>
<td>DissociationNode</td>
<td>Y</td>
<td>As Transition Node</td>
<td></td>
</tr>
<tr>
<td>LogicalOperatorNode</td>
<td>Y</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>ANDNode</td>
<td>Y</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>ORNode</td>
<td>Y</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>NOTNode</td>
<td>Y</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

3.5.3 State variables

States variables are very simple. The variable can have a name. If the name is set, it should be displayed. The variable can take any value. The names used in the controlled vocabulary of post-translational modification in section 2.2 are reserved. Authors should avoid attaching any other meaning to them.

An entity pool node has a set of state variables, which is the union of all the state variables used in the diagram. All stateful EPNs must explicitly display the same set of state variables and each state variable must be assigned a value or be assigned the value “Not Set”.

3.5.4 Compartment spanning

In all cases an EPN cannot belong to more than one compartment. However, an EPN can be drawn over more than one compartment. In such cases the decision on which is the owning compartment is deferred to the drawing tool or the author. ComplexNodes may contain EPNs which belong to different compartments and in this way a complex can be used to describe an entity that spans more than one compartment.
This restriction makes it impossible to represent in a semantically correct way a macromolecule that spans more than one compartment — for example a receptor protein. It is clearly desirable to be able to show a macromolecule in a manner that the biologist expects (i.e. spanning from the outside through the membrane to the inside). Therefore, the author is recommended to draw the macromolecule across compartment boundaries, but the underlying SBGN semantic model will assign it to only one. The assignment to a compartment may be decided by the software drawing tool or the author. Note that this has implications for auto-layout algorithms as they will only be able to treat such entity nodes as contained within a compartment and will have no way of knowing a macromolecule spans a compartment.

The current solution is consistent with other Systems Biology representations such as SBML and BioPAX. For more information about the problems representing membrane spanning proteins and the rationale behind the current solution see Section C.

### 3.5.5 Compartments

The layout of compartments in an SBGN diagram does not imply anything about the topology of compartments in the cell. Compartments should be bounded and may overlap. However, adjacency and the nesting of compartments does not imply that these compartments are next to each other physically or that one compartment contains the other.

### 3.5.6 Modulation

It is implied, but not defined explicitly that a process has a rate at which it converts its input EPNs to its output EPNs. This concept is important in understanding how SBGN describes process modulation.

1. A process with no modulations has an underlying “basal rate” which describes the rate at which it converts inputs to outputs.
2. Modulation changes the basal rate in an unspecified fashion.
3. Stimulation is a modulation that’s effect is to increase the basal rate.
4. Inhibition is a modulation that’s effect is to decrease the basal rate.
5. The above types of modulation, when assigned to the same process are combined and have a multiplicative effect on the basal rate of the process.
6. Modulators that do not interact with each other in the above manner should be drawn as modulating different process nodes. Their effect is therefore additive.
7. At most one trigger can be assigned to a process. Two triggers would imply an implicit Boolean AND or OR operator. For clarity only one trigger can be assigned to a process and such combinations must be explicitly expressed as the Boolean operators.
8. At most one catalysis can be assigned to a process. A catalysis modulation implies that the exact biochemical mechanism underlying the process is known. In this context two catalysis reactions cannot be assigned to the same process as they are independent reactions. Other EPNs that modulate the catalysis can be assigned to the same process as modulators, stimulators, and inhibitors and will have a multiplicative modulation on the reaction rate defined by the catalysis.

### 3.5.7 Submaps

Submaps are a visual device that allow a map to be split into several views. They remain, however, part of the main map and share its namespace. As a test of validity it should be possible to reintroduced a submap into the main map by eliminating the SubMapNode and merging the equivalent EntityPoolNodes in both maps.
3.5.7.1 Rules for mapping to submaps
An EntityPoolNode in the main map can be mapped to one in the submap using a TagNode in the submap and SubMapTerminals (see Section 2.4.3) in the main map. For a mapping between map and submap to exists the following must be true:

1. The identifiers in the TagNode and SubMapTerminals must be identical.
2. The EntityPoolNodes must be identical.

3.5.7.2 Requirement to define a mapping
If a map and submap both contain the same EntityPoolNode, then a mapping between them must be defined as above.

3.5.7.3 Cloning consistency
If an EntityPoolNode is cloned in the main map then it must also be marked as cloned in the submap even if there is only one copy of the EntityPoolNode in the submap. The converse rule also applies where the EntityPoolNode is cloned in the submap, but not the main map.
Chapter 4

Layout Guidelines for a Process Diagram

4.1 Introduction

The previous chapters describe the appearance and meaning of SBGN Process Diagram Level 1 components. Objects are EPNs, transition nodes, container nodes, logical operators as well as connecting arcs. The components of a Process 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 Process Diagrams should be easily recognisable 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 [13] and Kaufmann and Wagner [14].

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 Process Diagram Level 1 graph.

4.2.1.1 Node-node overlaps

Nodes are only allowed to overlap in two cases:

1. the overlapping nodes define a glyph (e.g. a multimer composed by stacking of two containers representing the monomers).

2. nodes overlapping compartments (e.g. a complex placed on the compartment border).

Otherwise, nodes are not allowed to overlap (Figure 4.1). This includes the touching of nodes. Touching is not allowed apart from the case where it has a specific meaning, e.g. two macromolecules touching each other within a complex because they form the complex. Submaps are not allowed to overlap.

![Figure 4.1: Nodes must not overlap.](image)

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.](image)

4.2.1.3 Node border-edge overlaps

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

4.2.1.4 Edge-edge overlaps

Edges are not allowed to overlap (Figure 4.4 on the following 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 next page).
4.2.1.6 Node-edge connection

The arcs linking the square glyph of a transition to the consumption and production arcs are attached to the center of opposite sides (Figure 4.6). The modulatory arcs are attached to the other two sides, but not necessarily all to the center, as several modifiers can affect the same transition node. A transition connected to production arcs on opposite sides is a reversible transition.

Figure 4.6: Arcs between a transition and the consumption and production arcs must be attached to the center of opposite sides, modulatory arcs must be attached to the other two sides.

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.
4.2.1.9 **Compartments**

If a transition has all participants in the same compartment the transition node and all edges/arcs have to be in this compartment. If a transition has participants in at least two different compartments, the transition node has to be either in a compartment where the transition has at least one participant or in the empty space.

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. Some crossings may be unavoidable, e.g. the crossing between an edge and a compartment border or an edge and a complex (if the edge connects an element inside the complex with something outside). 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 **Branching of association and dissociation**

The branching points of association and dissociation nodes should be placed close to the symbol of the transition, if possible at a distance comparable than, or smaller to, the diameter of the symbol defining the transition (Figure 4.7).

![Figure 4.7](image)

**Figure 4.7:** Branching points should be close to association and dissociation symbols.

4.2.2.5 **Units of information**

Units of information should not hide the structure of the corresponding node and should not overlap other elements (Figure 4.8 on the following page).
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.

- Placement of substrates and products of a transition: Substrate and product nodes should be placed on different sides of the transition node.

- 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 a transition or all elements within a compartment) should be drawn close together.

- Directional information: Subsequent processes (e.g. a sequence of reactions) should be drawn in one direction (e.g. from top to bottom or from left to right).

- Compartments: Different compartments should have different background shade or color.
Appendix A

Complete examples of SBGN Process Diagram Level 1 graphs

The following diagrams present complete examples of SBGN Process Diagrams representing Biological processes. They by no mean exhaust the possibilities of SBGN Process Diagram Level 1.

Figure A.1 presents an example of metabolic pathway, that exemplifies the use of the EPNs simple chemical, macromolecule, and clone marker, the PNs transition, and the connecting arcs consumption, production and catalysis.

![Glycolysis diagram](attachment:glycolysis.png)

**Figure A.1:** Glycolysis. This example illustrates how SBGN can be used to describe metabolic pathways.

Figure A.2 on the following page presents an example of signalling pathway, that exemplifies in addition the use of the EPNs observable, and state variable, the containers complex, compartment and submap, the PNs association, and the connecting arcs stimulation. Note the complex IGF and IGF receptor, located on the boundary of the compartment. This position is only for user convenience. The complex has to belong to a given compartment in SBGN Process Diagram Level 1.

Figure A.3 on page 64 is an expanded version of the submap present on the diagram present in Figure A.2 on the following page. It shows the use of tag.

Figure A.4 on page 65 introduces an SBGN Process Diagram that spans several compart-
Figure A.2: Insulin-like Growth Factor (IGF) signalling. This example shows the use of compartments and how details can be hidden by using a submap. The submap is shown on Figure A.3 on the following page.

Note that the compartment “synaptic vesicle” is not contained in the compartment “synaptic button” but overlaps it. The simple chemical “ACh” of the “synaptic vesicle” is not the same EPN than the “ACh” of the “synaptic button” and of “synaptic cleft”. The situation is similar with the compartments “ER” and “muscle cytosol”. The diagram exemplifies the use of the PN omitted and dissociation, and the connecting arc trigger.

Figure A.5 on page 65 introduces the use of SBGN Process Diagram Level 1 to encode gene regulatory networks. It also show the use of the EPNs Source and the logical operator and.
Figure A.3: A submap of the previous diagram showing the MAPK cascade.
Figure A.4: Neuronal/Muscle signalling. A description of inter-cellular signalling using SBGN.

Figure A.5: Activated STAT1α induction of the IRF1 gene. An example of gene regulation using logical operators.
Appendix B

Reference card

Print this summary of SBGN symbols for a quick reference.
Appendix C

Issues postponed to future levels

C.1 Multicompartment entities

The problem of entities, such as macromolecules, spanning several compartments proved to be a challenge for the community involved in the development of SBGN Process Diagram Level 1. It was thus decided to leave it for a future Level. It turns out there is at the moment no obvious solution satisfactory for everyone. Three broad classes of solutions have been identified so far:

- One can systematically locate an EPN in a given compartment, for instance a transmembrane receptor in a membrane. However, the reactions of this entity with entities represented by EPN in other compartments, such as extracellular ligands and second messenger systems, will create artificial transport reactions.

- One can represent the domains of proteins in different compartments by macromolecules, and link all those macromolecules in a complex spanning several compartments. However, such a representation would be very confusing, implying that the domains are actually different molecules linked through non-covalent bonds.

- One can accept macromolecules that span several compartments, and represent domains as units of information. Those units of information should then be located in given compartments. To make a full use of such a representation, one should then start and end connecting arcs on given units of information, something prohibited by the current specification.

C.2 Logical combination of state variable values

The value of a state variable has to be perfectly defined in SBGN Process Diagram Level 1. If a state variable can take the alternative values 'A', 'B' and 'C', one cannot attribute it values such as 'non-A', 'A or B', 'any' or 'none'. As a consequence some biochemical processes cannot be easily represented because of the very large number of state to enumerate. The decision to forbid such a Boolean logic lies in the necessity of maintaining truth path all over an SBGN map.

C.3 Non-chemical entity nodes

The current specification cannot represent combinations of events and entities. For instance a variable "voltage" cannot be controlled by a difference of concentration between different entities, such as a given ion in both sides of a membrane.

C.4 Generics

SBGN Process Diagram Level 1 does not provide mechanisms to sub-class EPNs. There is no specific means of specifying that macromolecules or nucleic acid features X1, X2 and X2 are
APPENDIX C. ISSUES POSTPONED TO FUTURE LEVELS

subclasses of X. Therefore, any process that applies to all the subtypes of X has to be triplicated. That situation can easily generate combinatorial explosions of the number of EPNs or PNs.

C.5 State and transformation of compartments

In SBGN Process Diagram Level 1 a compartment is a stateless entity. It cannot carry state variables, and cannot be subjected to process modifying a state. As a result, a compartment cannot be transformed, moved, split or merged with another. If one wants to represent the transformation of a compartment, one has to create the start and end compartments, and represent the transport of all the EPNs from one to the other. This is not satisfactory, and should be addressed in the future.
Appendix D

Extended list of people involved in SBGN Process Diagram Level 1 development

Here is a more comprehensive list of person who have been actively involved in SBGN development, either by their help designing the languages, their comments on the specification, help with development infrastructure or any other useful input.


We aim this list to be rather complete. We are very sorry if we forgot someone, and will be grateful if you notified us any omission.
Bibliography


