Systems Biology Graphical Notation: Process Description language Level 1

Version 2.0

Date: October 31, 2011

Disclaimer: This is a working draft of the SBGN Process Description Level 1 Version 2.0 specification. It is not a normative document.

Editors:

Stuart Moodie	*Edinburgh, UK*
Nicolas Le Novère	*EMBL European Bioinformatics Institute, UK*
Emek Demir	*Sloan-Kettering Institute, USA*
Huaiyu Mi	*University of Southern California, USA*
Alice Viléger	*University of Manchester, UK*

Contents

Chapter 1

Introduction

The goal of the **S**ystems **B**iology **G**raphical **N**otation (SBGN) is to standardize the graphical/visual representation of biochemical and cellular processes. SBGN defines comprehensive sets 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. For a general description of SBGN, one can read:

> Nicolas Le Novère, Michael Hucka, Huaiyu Mi, Stuart Moodie, Falk Schreiber, Anatoly Sorokin, Emek Demir, Katja Wegner, Mirit I Aladjem, Sarala M Wimalaratne, Frank T Bergman, Ralph Gauges, Peter Ghazal, Hideya Kawaji, Lu Li, Yukiko Matsuoka, Alice Villéger, Sarah E Boyd, Laurence Calzone, Melanie Courtot, Ugur Dogrusoz, Tom C Freeman, Akira Funahashi, Samik Ghosh, Akiya Jouraku, Sohyoung Kim, Fedor Kolpakov, Augustin Luna, Sven Sahle, Esther Schmidt, Steven Watterson, Guanming Wu, Igor Goryanin, Douglas B Kell, Chris Sander, Herbert Sauro, Jacky L Snoep, Kurt Kohn & Hiroaki Kitano. The Systems Biology Graphical Notation. *Nature Biotechnology* **27**, 735 - 741 (2009). http://dx.doi.org/10.1038/nbt.1558

This document defines the *Process Description* visual language of SBGN. Process Descriptions are one of three views of a biological process offered by SBGN. It is the product of many hours of discussion and development by many individuals and groups.

1.1 SBGN levels and versions

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 need a vast language that most newcomers would shun as being too complex. Thus, the SBGN community followed an idea used in the development of other standards, i.e. stratify language development into levels.

A *level* of one of the SBGN languages 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. Within *levels*, *versions* represent small evolution of a language, that may involve new glyphs, refined semantics, but no fundamental change of the way maps are to be generated and interpreted. 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 or version. In this way, the development of SBGN languages is envisioned to proceed in stages, with each higher levels adding richness compared to the levels below it.

1.2 Developments, discussions, and notifications of updates ~34~

The SBGN website (http://sbgn.org/) is a portal for all things related to SBGN. It provides a web ~35~
forum interface to the SBGN discussion list (sbgn-discuss@caltech.edu) and information about ~36~
how anyone may subscribe to it. The easiest and best way to get involved in SBGN discussions is to ~37~
join the mailing list and participate. ~38~

Face-to-face meetings of the SBGN community are announced on the website as well as the mail- ~39~
ing list. Although no set schedule currently exists for workshops and other meetings, we envision ~40~
holding at least one public workshop per year. As with other similar efforts, the workshops are likely ~41~
to be held as satellite workshops of larger conferences, enabling attendees to use their international ~42~
travel time and money more efficiently. ~43~

Notifications of updates to the SBGN specification are also broadcast on the mailing list and an- ~44~
nounced on the SBGN website. ~45~

1.3 Note on typographical convention ~46~

The concept represented by a glyph is written using a normal font, while a `glyph` means the SBGN ~47~
visual representation of the concept. For instance "a biological process is encoded by the SBGN PD ~48~
`process`". ~49~

Chapter 2

Concepts

To set the stage for what follows, we first give a brief overview of some of the concepts in the Process Description language with the help of an example shown in Figure 2.1.
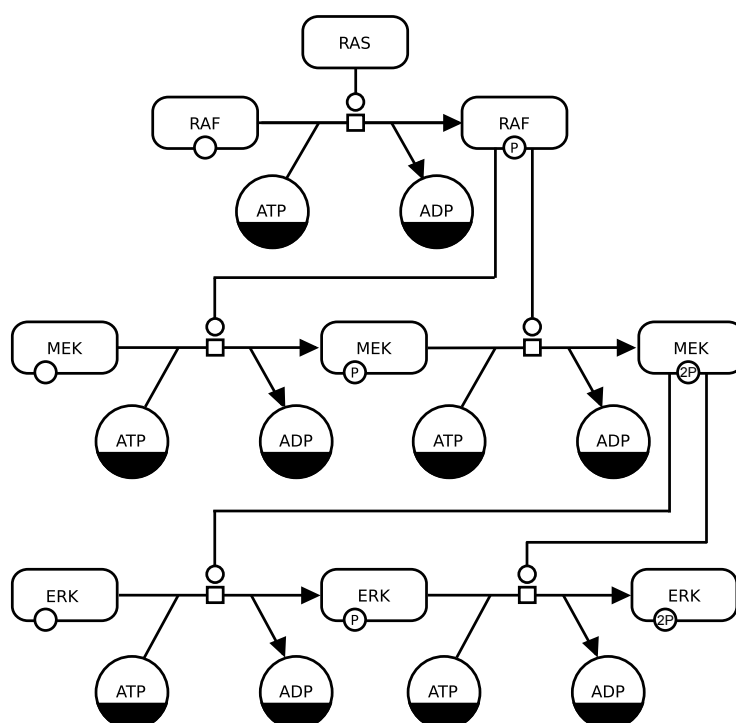
Figure 2.1: *This example of a Process Description uses two kinds of entity pool nodes: one for pools of different macromolecules (Section 3.2.11) and another for pools of simple chemicals (Section 3.2.7). Most macromolecule nodes in this map are adorned with state variables (Section 3.2.17) representing phosphorylation states. This map uses one type of process node, the process node (Section 3.2.15), and three kind of connecting arc, consumption (Section 3.2.19), production (Section 3.2.19) and catalysis (Section 3.2.19). Finally, some entity pool nodes have dark bands along their bottoms; these are clone markers (Section 3.3.1) indicating that the same pool nodes appear multiple times in the map.*

The map in Figure 2.1 is a simple map 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 Process Description language by lines with arrows and other decorations. In this particular map, all of the processes happen to be the same: processes catalyzed by biochemical entities. The directions of the arrows indicate the direction of the processes; for example, unphosphorylated RAF kinase processes to phosphory-

3

lated RAF kinase via a process catalyzed by RAS. Although ATP and ADP are shown as incidental to 61
the phosphorylations on this particular graph, they are involved in the same process as the proteins 62
getting phosphorylated. The small circles on the nodes for RAF and other entity pools represent state 63
variables (in this case, phosphorylation sites). 64

The essence of the Process Descriptions is *change*: it shows how different entities in the system 65
process from one form to another. The entities themselves can be many different things. In the ex- 66
ample of Figure 2.1, they are either pools of macromolecules or pools of simple chemicals, but as 67
will become clear later in this chapter, they can be other conceptual and material constructs as well. 68
Note also that we speak of *entity pools* rather than individuals; this is because in biochemical network 69
models, one does not focus on single molecules, but rather collections of molecules of the same kind. 70
The molecules in a given pool are considered indistinguishable from each other. The way in which 71
one type of entity is transformed into another is conveyed by a *process node* and links between en- 72
tity pool nodes and process nodes indicate an influence by the entities on the processes. In the case 73
of Figure 2.1, those links describe consumption Section 3.2.19, production Section 3.2.19 and catal- 74
ysis Section 3.2.19, but others are possible. Finally, nodes in Process Descriptions are usually not 75
repeated; if they do need to be repeated, they are marked with *clone markers*—specific modifica- 76
tions to the appearance of the node (Section 3.3.1). The details of this and other aspects of Process 77
Description notation are explained in the following chapters. 78

2.1 Definitions and Nomenclature 79

2.1.1 Language versus notation 80

SBGN specifications propose symbols, ways to organise them, but also semantic rules to analyse the 81
resulting representations. SBGN "drawings" can be translated into English, but also into computer 82
readable formats. Those specifications really propose true languages. SBGN is therefore made up of 83
three languages. 84

2.1.2 What are the languages? 85

PD is a language that permits the description of all the processes taking place in a biological system. 86
The ensemble of all these processes constitute a Description. **ER** is a language that permits the de- 87
scription of all the relations involving the entities of a biological system. The ensemble of all these 88
relations constitute a Relationship. **AF** is a language that permits the description of the flow of activity 89
in a biological system. 90

2.1.3 Nomenclature 91

The three languages of SBGN should be referred to as: 92
- the Process Description language. 93
- the Entity Relationship language. 94
- the Activity Flow language. 95

  96

Abbreviated as: 97
- the PD language. 98
- the ER language. 99
- the AF language. 100

  101

A specific representation of a biological system in one of the SBGN languages should be referred to 102
as: 103
- a Process Description map. 104
- an Entity Relationship map. 105
- an Activity Flow map. 106

Abbreviated as:

- a PD map.
- an ER map.
- an AF map.

The corpus of all SBGN representations should be referred to as:
- Process Descriptions.
- Entity Relationships.
- Activity Flows.

The capitalization is important. PD, ER and AF are names of languages. As such they must be capi-
talized in English. This is not the case of the accompanying noun (language or map).

2.1.4 Graph, diagram or map?

A graph is a very technical term that belongs to mathematics and is uncommon in biology. Dia-
gram is a concept that encompasses more than just graph. Examples are Venn diagrams for instance.
Therefore, we recommend using the term map for SBGN representations. Those representations ef-
fectively permit users to travel and orient themselves in a biological system. Map is also the term
most frequently used by the different communities, whether in metabolism, signaling or genomics.

Chapter 3

Concepts and Glyphs

3.1 Introduction

Although ultimately defined by its glyphs, SBGN Process Description Level 1 these glyphs represent concepts that are related to varying degrees and ultimately can be organised hierarchically. This pro- vides us with a useful way or organising the glyphs and thinking about Process Description. Therefore in this chapter we describe the conceptual structure of SBGN Process Description Level 1 and place each glyph within this structure. For each glyph or concept we describe its physical appearance and define syntax and other usage rules. An overview of this hierarchy is provided in figures 3.1 and **??**.

Figure 3.1: *Organisation of the node glyphs within SBGN Process Description language. All UML classes (boxes) correspond to Process Description node glyphs except those with italicised names, which are organisational groupings. They correspond to the groupings used elsewhere in this document.*

3.2 Definitions

3.2.1 SBGNElement

All the glyphs in SBGN Process Description Level 1 inherit from `SBGNElement`. This is an abstract or conceptual class that helps organise Process Description conceptually. `SBGNElement` (figure 3.2) has a single attribute id that is an identifying attribute. This means that all SBGN elements defined here, which ultimately extend `SBGNBase`, can all be uniquely identified from each other. This makes sense if you think that a glyph drawn on a map is distinct from another glyph drawn on the map. The id attribute reflects this and is not shown explicitly in a Process Description map.

Name: SBGN Element · 143

SBO Term: Not defined · 144

Identifying Attributes: · 145

  id:identity the implicit identity of an SBGN element ensuring all instances are unique. · 146

Permitted Child Glyphs: N/A · 147

Special constraints or rules: N/A · 148

Cloning: N/A · 149

3.2.2 Map · 150

The Map (figure 3.2) is a concrete class the extends the SBGNElement. It describes the visible container · 151
upon which all glyphs are drawn. The map can take any pattern or colour, but as SBGN is 'colour · 152
blind' this does not convey any meaning in itself. · 153

Name: Map · 154

SBO Term: Not defined · 155

Identifying Attributes: · 156

  glyphs:SBGNGlyph The collection of glyphs held by the map. · 157

Permitted Child Glyphs: All SBGN glyphs that extend SBGNGlyph · 158

Special constraints or rules: A map is valid if it is empty (although not very useful) · 159
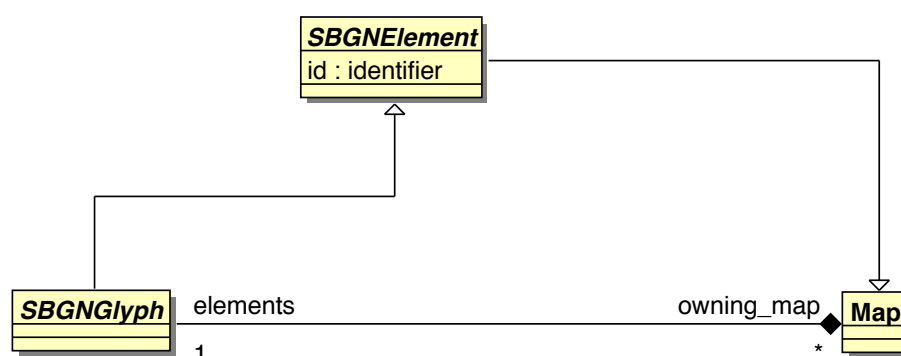
Cloning: N/A · 160

Figure 3.2: *UML definition of Map and SBGNElement.*

3.2.3 Glyphs, Auxiliary Units, Nodes and Arcs · 161

In addition to the map elements within SBGN are either descendants of SBGNGlyph (glyphs) or SBGNAuxiliaryUnit · 162
(auxiliary units). See figure 3.3. Glyphs are the fundamental building blocks of the Process Descrip- · 163
tion language and are the only elements that can be drawn on a map (MAP). Auxiliary units in contrast · 164
cannot and are used to adorn glyphs and providing additional information that may be useful to the · 165
reader. These can provide annotation (annotation), state information (state variable) or indi- · 166
cate duplication of entity pool nodes (clone marker). Glyphs are further subdivided divided into · 167
nodes (SBGNGlyph) and edges (SBGNArc) that define the graph structure inherent in a Process De- · 168
scription map. · 169

3.2.4 Entity pool nodes · 170

An entity pool is a population of entities that cannot be distinguished from each other, when it comes · 171
to the SBGN Process Description Level 1 map. For instance all the molecular entities that fulfill the · 172
same role in a given process form an entity pool. As a result, an entity pool can represent different · 173
granularity levels, such as all the proteins, all the instances of a given protein, only certain forms of a · 174
given protein. To belong to a different compartment is sufficient to belong to different entity pools. · 175
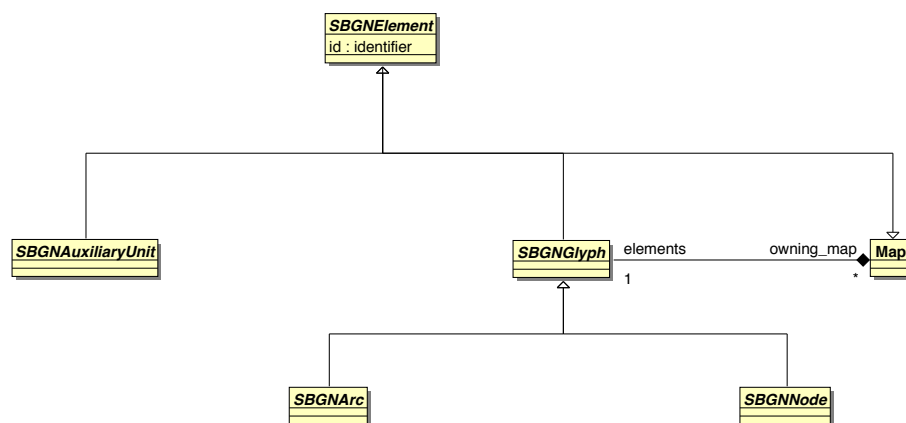
Figure 3.3: *UML definition of the glyph, auxiliary unit, nodes and arcs.*

Calcium ions in the endoplasmic reticulum and calcium ions in the cytosol belong to different entity 176
pools when it comes to representing calcium release from the endoplasmic reticulum. 177

The Process Description contains five glyph types representing classes of material entities: `unspecified` 178
`entity` (Section 3.2.8), `simple chemical` (Section 3.2.7), `macromolecule` (Section 3.2.11), `nucleic` 179
`acid feature` (Section 3.2.12) and `complex` (Section 3.2.13). (Specific types of macromolecules, 180
such as protein, RNA, DNA, polysaccharide, and specific simple chemicals are not defined by Pro- 181
cess Description but may be part of future levels of SBGN.) In addition to the material entities, Pro- 182
cess Description represents a conceptual entities: `perturbing agent` (Section 3.2.9). 183
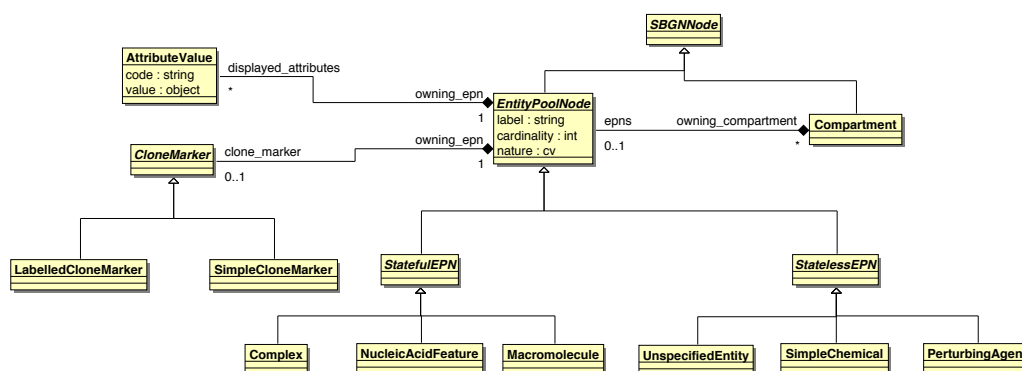

Figure 3.4: *UML definition of the entity pool node and its descendant glyphs.*

The `EntityPoolNode` (figure 3.4) is the common ancestor of all these glyphs and defines at- 184
tributes that are common to all its descendants. It must belong to a compartment (c.f. section 3.2.16) 185
and can contain a clone marker if it is cloned. Note that not all EPNs can be cloned. 186

EntityPoolNode 187

SBO Term: Not defined 188
Identifying Attributes: 189
    label:`string` The name that identifies the entity in the Process Description map. EPNs with 190
        the same label should be from the same entity. the string cannot be empty and must start 191
        and end with a non-space character. Any Unicode character is acceptable. 192
    cardinality:`int` The number of copies of the entity. Must be a positive non-zero integer. 193
    nature:`cv` The nature of the entity pool node as defined by a controlled vocabulary. 194
    owning_compartment:`Compartment` The compartment that this EPN belongs too. 195
    clone_marker:`CloneMarker` The clone marker decorator. See section 3.3.1 for its use. 196

displayed_attributes:`AttributeValue` One or more decorators used to display attribute values. ₁₉₇

Permitted Child Glyphs: Annotation ₁₉₈

Special constraints or rules: 1. If cardinality > 1 then the descendant glyph must be displayed as a ₁₉₉ multimer. 2. If the EPN is drawn directly on a `Map` then owning_compartment is assigned to an ₂₀₀ invisible default compartment. nature can only use the material type (section **??**), conceptual ₂₀₁ type (section **??**) or physical characteristics (section **??**) controlled vocabularies. ₂₀₂

Cloning: Defined by descendants. ₂₀₃

3.2.5 Empty Set ₂₀₄

It is useful to have the ability to represent the creation of an entity or a state from an unspecified ₂₀₅ source, that is, from something that one does not need or wish to make precise. For instance, in a ₂₀₆ model where the production of a protein is represented, it may not be desirable to represent all of ₂₀₇ the amino acids, sugars and other metabolites used, or the energy involved in the protein's creation. ₂₀₈ Similarly, we may not wish to bother representing the details of the destruction or decomposition of ₂₀₉ some biochemical species into a large number of more primitive entities, preferring instead to simply ₂₁₀ say that the species "disappears into a sink". Yet another example is that one may need to represent ₂₁₁ an input (respectively, output) into (resp. from) a compartment without explicitly representing a ₂₁₂ transport process from a source (resp. to a target). ₂₁₃

For these and other situations, SBGN defines a single glyph to handle these situations represent- ₂₁₄ ing the involvement of an external pool of entities. The symbol used in SBGN is borrowed from the ₂₁₅ mathematical symbol for "empty set", but it is important to note that it does not actually represent a ₂₁₆ true absence of everything or a physical void—it represents the absence of the corresponding struc- ₂₁₇ tures in the model, that is, the fact that the external pool is conceptually outside the scope of the ₂₁₈ map. ₂₁₉

A frequently asked question is, why bother having an explicit symbol at all? The reason is that ₂₂₀ one cannot simply use an arc that does not terminate on a node, because the dangling end could ₂₂₁ be mistaken to be pointing to another node in the map. This is specially true if the map is rescaled, ₂₂₂ causing the spacing of elements in the map to change. The availability and use of an explicit symbol ₂₂₃ for sources and sinks is critical. ₂₂₄
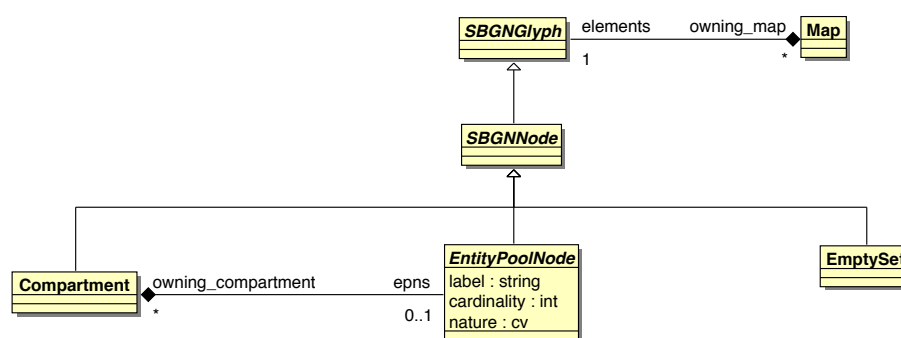

Figure 3.5: *The UML definition of the* `EmptySet` *and its context in relation to other elements of the Process Description language.*
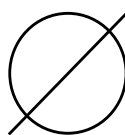
The definition of the `Empty Set` is shown in figure 3.5. The empty set is not an EPN as it does not ₂₂₅ represent a single pool of entities and does not share any of the other attributes of an EPN, nor does ₂₂₆ it belong to a particular compartment. ₂₂₇

Glyph: `Empty Set` 228

Figure 3.6: *The* `empty set` *glyph.*

SBO Term: SBO:0000291 ! empty set 229
Permitted Child Glyphs: None 230
Special constraints or rules: 231
Cloning: All instances of `Empty Set` can be regarded as identical therefore not special decoration is 232
used to indicate replication on the map. 233

Container Represented by the mathematical symbol for "empty set", that is, a circle crossed by a 234
bar linking the upper-right and lower-left corners of an invisible square drawn around the circle ($\emptyset$). 235
Figure 3.6 illustrates this. The symbol should be linked to one and only one edge in a map. 236

Label None 237
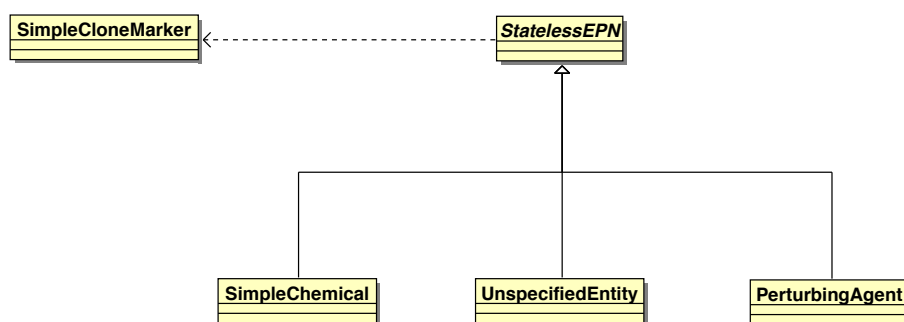
3.2.6 Stateless Entity Pool 238

The `StatelessEPN` (figure 3.7) represents a pool where the entities do not change 'state'. In other- 239
words the entities do not undergo any physical change that is useful to record in a Process Descrip- 240
tion map. Therefore these glyphs cannot be assigned a state-variable. 241

Figure 3.7: *UML definition of the stateless entity pool node and its descendant glyphs.*

StatelessEPN 242

SBO Term: Not defined 243
Permitted Child Glyphs: 244
Special constraints or rules: 1. if a clone marker is used it must be of type `SimpleCloneMarker`. 245
Cloning: Defined by descendants. 246

3.2.7 Simple chemical 247

A simple chemical in SBGN is defined as the opposite of a macromolecule (Section 3.2.11): it is a 248
chemical compound that is *not* formed by the covalent linking of pseudo-identical residues. Exam- 249
ples of simple chemicals are an atom, a monoatomic ion, a salt, a radical, a solid metal, a crystal, 250

etc. The complex can be represented by a monomeric glyph (`Simple chemical monomer`) and a ₂₅₁ multimeric glyph (`Simple chemical multimer`). ₂₅₂

Glyph: `Simple chemical monomer` ₂₅₃

SBO Term: SBO:0000247 ! simple chemical ₂₅₄
Permitted Child Glyphs: ₂₅₅
Special constraints or rules: ₂₅₆
Cloning: Yes. ₂₅₇
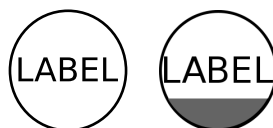
Figure 3.8: *The Process Description glyph for* `simple chemical`.

Container A `simple chemical` is represented by a circular container, as depicted in Figure 3.8. To ₂₅₈ avoid confusion with the Unspecified Entity (3.2.8), this glyph must remain a circle and cannot be ₂₅₉ deformed into an eclipse. ₂₆₀

Label The identification of the `simple chemical` is carried by an unbordered box containing a ₂₆₁ string of characters. The characters may be distributed on several lines to improve readability, al- ₂₆₂ though this is not mandatory. The label box has to be attached to the center of the circular container. ₂₆₃ The label is permitted to spill outside the container. ₂₆₄

Glyph: `Simple chemical multimer` ₂₆₅

SBO Term: SBO:0000421 ! multimer of simple chemicals ₂₆₆
Permitted Child Glyphs: ₂₆₇
Special constraints or rules: ₂₆₈
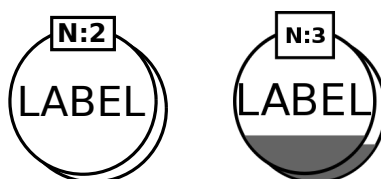**Cloning:** Yes. ₂₆₉

Figure 3.9: *The Process Description glyph for* `multimer`.

Container A `simple chemical multimer` is represented by two identical containers shifted hori- ₂₇₀ zontally and vertically and stacked one on top of the other. Figure 3.9 illustrates the glyph. ₂₇₁

Label A `multimer` has no identity on its own. However, the first of the monomers carries an identi- ₂₇₂ fying label. The label is placed in an unbordered box containing a string of characters. The characters ₂₇₃ can be distributed on several lines to improve readability, although this is not mandatory. The label ₂₇₄ box must be attached to the center of the top monomer's container. The label may spill outside of ₂₇₅ the container. ₂₇₆

3.2.8 Unspecified entity 277

The simplest type of EPN is the `unspecified entity`: one whose type is unknown or simply not 278
relevant to the purposes of the map. This arises, for example, when the existence of the entity has 279
been inferred indirectly, or when the entity is merely a construct introduced for the needs of a map, 280
without direct biological relevance. These are examples of situations where the *unspecified entity* 281
glyph is appropriate. (Conversely, for cases where the identity of the entities composing the pool *is* 282
known, there exist other, more specific glyphs described elsewhere in the specification.) 283

Glyph: `Unspecified entity` 284

SBO Term: SBO:0000285 ! material entity of unspecified nature 285
Permitted Child Glyphs: 286
Special constraints or rules: 1. The cardinality of the unspecified entity is always 1, i.e., (there is no 287
multimer glyph. 288
Cloning: Yes 289

Container An `unspecified entity` is represented by an elliptic container, as shown in 3.10. Note 290
that this must remain an ellipse to avoid confusion with the Simple Chemical glyph, which is a circle 291
(c.f. 3.2.7). 292

Label An `unspecified entity` is identified by a label placed in an unbordered box containing a 293
string of characters. The characters can be distributed on several lines to improve readability, al- 294
though this is not mandatory. The label box must be attached to the center of the container. The 295
label may spill outside of the container. 296

Figure 3.10: *The Process Description glyph for* `unspecified entity`*.*

3.2.9 Perturbing Agent 297

Biochemical networks can be affected by external influences. Those influences can be the effect of 298
well-defined physical perturbing agents, such as a light pulse or a change in temperature; they can 299
also be more complex and not well-defined phenomena, for instance the outcome of a biological 300
process, an experimental setup, or a mutation. For these situations, SBGN provides the `perturbing` 301
`agent` glyph. It is an EPN, and represents the amount to perturbing agent applied to a process. 302
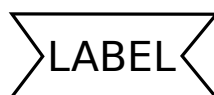
Glyph: `Perturbing agent` 303

Figure 3.11: *The Process Description glyph for* `perturbing agent`*.*

SBO Term: SBO:0000405 ! perturbing agent 304
Permitted Child Glyphs: A `perturbing agent` can optionally carry one or more `units of information` 305
(Section 3.2.17). 306

Special constraints or rules: 1. A `perturbing agent` can optionally carry one or more `units of` 307
`information` (Section 3.2.17). A controlled vocaulary is available to describing the physical 308
characteristic of the perturbing agent (see Section 3.4.4). 309

Cloning: Yes 310

Container A `perturbing agent` is represented by a modified hexagon having two opposite con- 311
cave faces, as illustrated in Figure 3.11. 312

Label A `perturbing agent` is identified by a label placed in an unbordered box containing a string 313
of characters. The characters can be distributed on several lines to improve readability, although this 314
is not mandatory. The label box must be attached to the center of the `perturbing agent` container. 315
The label may spill outside of the container. 316

3.2.10 Stateful Entity Pool 317

Stateful entity pools can undergo physical changes, for example chemical modifcation or conforma- 318
tional change, which we wish to record in a Process Description map. This information is captured 319
via the `StateVariable`, as can be seen in figure 3.12). Replicated `StatefulEPS`s are indicated by the 320
`Labelled Clone Marker` decoration. Note that not all glyphs that are descendants of `StatefulEPN` 321
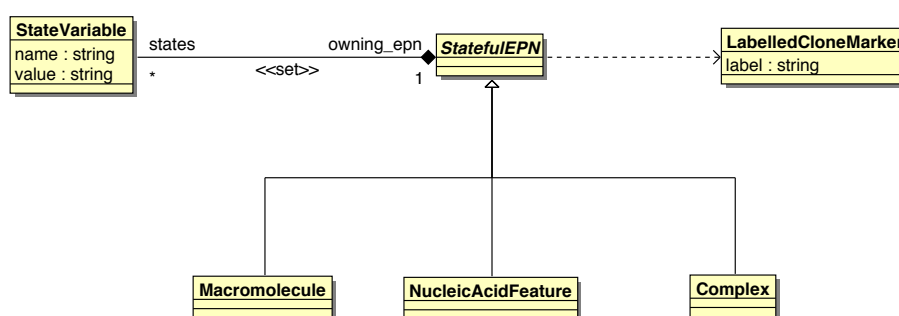can be cloned at all. 322

Figure 3.12: *UML definition of the stateful entity pool node and its descendant glyphs and its association with state variables.*

StatefulEPN 323

SBO Term: Not defined 324

Identifying Attributes: 325
    states:`StateVariable` The state variables associated with this EPN. 326

Permitted Child Glyphs: It can contain one or more `StateVariable` glyphs associated with it. 327

Special constraints or rules: 1. Two or more state variables with the same name are permitted. 2. State 328
variables with no name set are permitted. 3. A `LabelledCloneMaker` must be used to indicate 329
cloning. 4. `StatefulEPN`s that are identical and so decorated with a `LabelledCloneMarker` 330
must use the same label to indicate that they are part of the same 'clone'. 331

Cloning: 332

3.2.11 Macromolecule 333

Many biological processes involve *macromolecules*: biochemical substances that are built up from 334
the covalent linking of pseudo-identical units. Examples of macromolecules include proteins, nu- 335
cleic acids (RNA, DNA), and polysaccharides (glycogen, cellulose, starch, etc.). Attempting to define 336
a separate glyph for all of these different molecules would lead to an explosion of symbols in SBGN, 337

so instead, SBGN Process Description Level 1 defines only one glyph for all macromolecules. The 338
same glyph is to be used for a protein, a nucleic acid, a complex sugar, and so on. The exact nature of 339
a particular macromolecule in a map is then clarified using its label and decorations, as will become 340
clear below. (Future levels of SBGN may subclass the `macromolecule` and introduce different glyphs 341
to differentiate between types of macromolecules). This has two associated glyphs. One where 342

Glyph: `Macromolecule monomer` 343
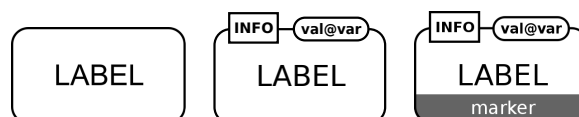
Figure 3.13: *The Process Description glyph for `macromolecule`, shown plain and unadorned on the left, and with with an additional state variable and a unit of information in the right and the cloned form on the right.*

SBO Term: SBO:0000245 ! macromolecule 344
Permitted Child Glyphs: 345
Special constraints or rules: 1. cardinality must equal 1. 346
Cloning: Yes. 347

Container A macromolecule is represented by a rectangular container with rounded corners, as il- 348
lustrated in Figure 3.13. 349

Label A `macromolecule` is identified by a label placed in an unbordered box containing a string of 350
characters. The characters can be distributed on several lines to improve readability, although this 351
is not mandatory. The label box must be attached to the center of the container. The label may spill 352
outside of the container. 353

Glyph: `Macromolecule multimer` 354

Figure 3.14: *The Process Description glyph for `macromolecule multimer`, shown plain and un-adorned on the left, and with with an additional state variable and a unit of information on the right.*

SBO Term: SBO:0000420 ! multimer of macromolecules 355
Permitted Child Glyphs: 356
Special constraints or rules: 1. cardinality must be greater than 1. 357
Cloning: Yes. 358

Container A `multimer` is represented by two identical containers shifted horizontally and vertically 359
and stacked one on top of the other. Figure 3.14 illustrates the glyph. 360

Label As monomer 361

3.2.12 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 nature are often important for making the purpose clear to the reader of a map.

Glyph: `Nucleic acid feature monomer`

This glyphs represents a monomeric macromolecule.

SBO Term: SBO:0000354 ! informational molecule segment

Container: A `nucleic acid feature` is represented by a rectangular container whose bottom half has rounded corners, as shown in Figure 3.15. 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.

Permitted Child Glyphs: A `nucleic acid feature` can carry state variables (Section 3.2.17) 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 3.2.17). 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 3.4.1) and the conceptual type (Section 3.4.2) of the `nucleic acid feature`.

Cloning: Labeled Clone Marker

Figure 3.15: *The Process Description glyph for `nucleic acid feature monomer`, shown plain and unadorned on the left and with an additional state variable and a unit of information in the middle and the cloned form on the right.*

Glyph: `Nucleic acid feature multimer`

This glyphs represents a multimeric macromolecule.

SBO Term: SBO:0000419 ! multimer of informational molecule segments

Container: A `Nucleic acid feature multimer` is represented by two identical containers shifted horizontally and vertically and stacked one on top of the other. Figure 3.9 illustrates the glyph.

Label: As monomer glyph.

Permitted Child Glyphs: A `nucleic acid feature` can carry state variables (Section 3.2.17) 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 3.2.17). 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 3.4.1) and the conceptual type (Section 3.4.2) of the `nucleic acid feature`.

A `nucleic acid feature` may also carry a `clone marker` (Section 3.3.1).

Figure 3.16: *The Process Description glyph for* `nucleic acid feature monomer`*, shown plain and unadorned on the left and with an additional state variable and a unit of information in the middle and the cloned form on the right.*

3.2.13 Complex   399

A `Complex` represents a biochemical entity composed of other biochemical entities, whether macro-   400
molecules, simple chemicals, multimers, or other complexes (figure 3.17). The `Complex` can de-   401
scribed its composition by the set of `Subunit`s it contains (see figure **??**). This description is entirely   402
optional and is their to assist the user with a visual shorthand about the composition of the complex.   403
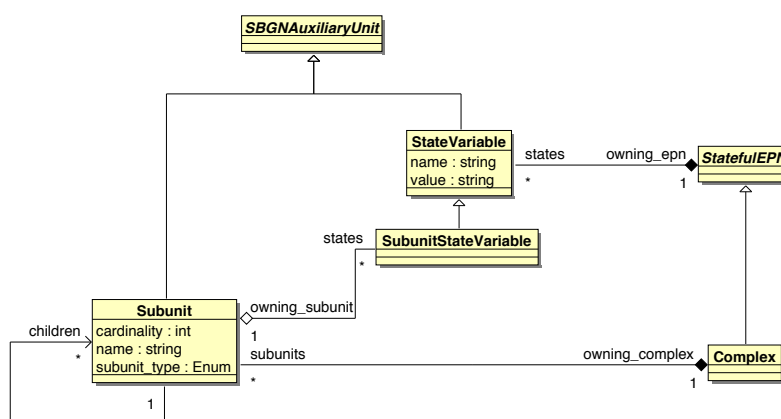
Figure 3.17: *The UML definition of the* `Complex` *and its associated* `subunits`*. In particular this describes organisation of the state variables that belong to both the subunit, but also the complex.*

    subsubsectionDefinition   404
Name: `Complex`   405
SBO Term: Not defined   406
Identifying Attributes: 407
    subunits:`'sbgnclasssubunit` The subunits that describe the composition of this complex.   408
Special constraints or rules: • Once a set of subunits are defined then they are always defined.   409
    subsubsectionVisual Appearance   410
    The Complex is represented by two glyphs, the `Complex Monomer` which represents a `Complex`   411
where the cardinality is one and the `Complex Multimer` where the cardinality is greater than that.   412

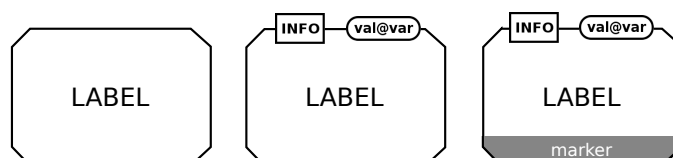### Complex Monomer   413

Figure 3.18: *The* `complex` *glyph.*

SBO Term: SBO:0000253 ! non-covalent complex   414

Permitted Child Glyphs: A `complex` can carry state variables (see Section 3.2.17). The state of a $_{415}$
complex is defined by the set of the all its state variable and all the state variables of all its com- $_{416}$
ponents. A `complex` can also carry one or several `units of information` (see Section 3.2.17). $_{417}$
A `complex` may carry a `clone marker` (see Section 3.3.1). $_{418}$

Cloning: Labeled Clone Marker $_{419}$

Container A `complex` possesses its own container box surrounding the juxtaposed container boxes $_{420}$
of its components. This container box is a rectangle with cut-corners (an octagonal box with sides of $_{421}$
two different lengths). The size of the cut-corners are adjusted so that there is no overlap between $_{422}$
the container and the components. The container boxes of the components must not overlap. $_{423}$

Label The identification of a `named complex` is carried by an unbordered box containing a string of $_{424}$
characters. The characters may be distributed on several lines to improve readability, although this is $_{425}$
not mandatory. The label box has to be attached to the midway between the border of the complex's $_{426}$
container box and the border of the components' container boxes. $_{427}$

Complex Multimer $_{428}$
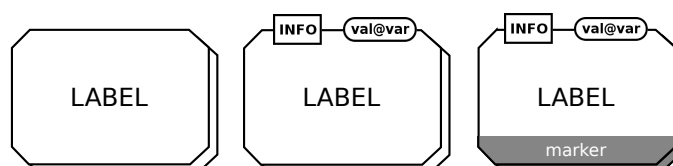
Figure 3.19: *The Complex Multimer glyph.*

SBO Term: SBO:0000418 ! multimer of complexes $_{429}$
Permitted Child Glyphs: As monomer. $_{430}$
Cloning: Labeled Clone Marker $_{431}$

Container A `Complex Multimer` is represented by two identical `Complex` containers shifted hori- $_{432}$
zontally and vertically and stacked one on top of the other. Figure 3.19 illustrates the glyph. $_{433}$

Label As monomer. $_{434}$

3.2.14 Subunits $_{435}$

A complex can optionally be decorated with subunit symbols (`Subunit` see figure 3.17) to describe $_{436}$
the composition of the complex. The symbols available are equivalent to those used by the EPN $_{437}$
glyphs including the `complex`. Therefore it is possible to describe complexes within complexes. Sub- $_{438}$
units may contain labels. $_{439}$

Subunit $_{440}$

SBO Term: Not defined $_{441}$
Identifying Attributes: $_{442}$
owning_complex:`Complex` The complex that owns the subunit. $_{443}$
cardinality:`int` The number of copies of the subunit. $_{444}$
name:`string` The name of the subunit. $_{445}$
subunit_type:`enum` The type of the subunit, for example macromolecule, simple chemical etc. $_{446}$
One type must be chosen from an enumerated set of types (see section **??**). $_{447}$

states:SubunitStateVariable The state variables assigned to this subunit. 448

Permitted Child Glyphs: It can contain zero, one or more SubunitStateVariables. 449

Special constraints or rules: 1. Two or more state variables with the same name are permitted. 2. State 450 variables with no name set are permitted. 3. Subunits can also contain subunits. There is no 451 limit on such nesting. The namespace rules below apply. 4. Each state variable actually be- 452 longs to the Complex. 5. The subunit defines a namespace for its state variables, e.g. subunit 453 "A" assigned a state variable "P@Ser202" and a subunit "B" assigned the same state variable can 454 be distinguised as A:P@Ser202 and B:P@Ser202. 455

Cloning: No 456

The example in figure Figure 3.20 illustrates the use of subunits in a complex shows an equivalent 457 compex without subunits. This is an import point. For every Complex drawn with subunits it will 458 always be possible to drawn an equivalent version that does not use contains subunits. 459
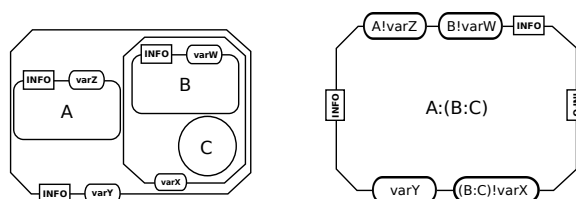
Figure 3.20: *Both these complex glyphs are equivalent. The one on the left is described using sub-unit decorators, the one on the right describes the same thing without them.*

Examples of complex EPNs 460

In this section, we provide examples of Entity Pool Node representations drawn using the SBGN Pro- 461 cess Description Level 1 glyphs described above. 462

Figure 3.21 represents calcium/calmodulin kinase II, with phosphorylation on the sites threonine 463 286 and 306, as well as catalytic and autoinhibitory domains. Note the use of *units of information* and 464 *state variables*. 465
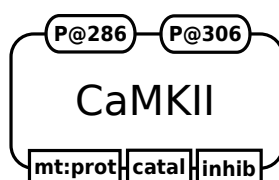

Figure 3.21: *An example representation of calcium/calmodulin kinase II.*

Figure 3.22 represents the glutamate receptor in the open state, with both phosphorylation and 466 glycosylation. The entity carries two functional domains, the ligand-binding domain and the ion 467 pore, and its chemical nature is precided. 468
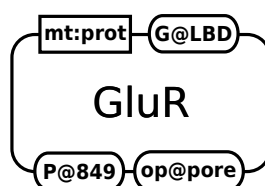

Figure 3.22: *An example of a glutamate receptor in the open state.*

3.2.15 Process

Process nodes represent processes that transform one or several entity pools into one or several entity pools, identical or different. Note that a process may be used to represent or summarise more than one known process. SBGN Process Description Level 1 defines a generic `process` (Section 3.2.15), as well as five more specific ones: the `omitted process` (Section 3.2.15), the `uncertain process` (Section 3.2.15), the `association` (Section 3.2.15), the `dissociation` (Section 3.2.15), and the `phenotype` (Section 3.2.15).

Identifying Attributes:
- Instance - all entities are unique

glyphRules A process cannot be "orphaned". It must always be connected to another node by one or more arcs.

Non-stoichiometric process

Process that does not necessarily result in measurable change of entity pools, and which does necessarily have a defined start and end point.

Glyph: Phenotype 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 map, SBGN defines the `phenotype` glyph.

SBO Term: SBO:0000358 ! phenotype

Container: A `phenotype` is represented by an elongated hexagon, as illustrated in Figure 3.23.

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

Permitted Child Glyphs: A `phenotype` may carry a `clone marker` (Section 3.3.1).

Figure 3.23: *The Process Description glyph for* `phenotype`.

Glyph: Submap 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 3.2.4). A `submap` is not equivalent to an `omitted process` (see Section 3.2.15). In the case of an SBGN description 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 representing the `submap`. The tool might then expand and show the `submap` within the same map (on the same canvas), or it might open it in a different canvas. In the case of an SBGN description made available in a book or a website, the content of the `submap` may be available on another page, possibly accessible via an hyperlink on the `submap`.

SBO Term: SBO:0000395 ! encapsulating process

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.

Permitted Child Glyphs: A `submap` carries labeled terminals. When the `submap` is represented folded, those terminals are linked to external `EPNs` (Section 3.2.4). In the unfolded view, exposing the

internal structure of the `submap`, a set of `tags` point to the corresponding internal EPNs Section 3.2.4.
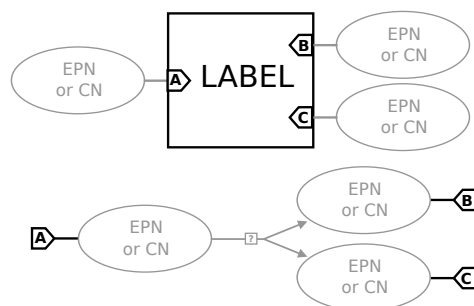

Figure 3.24: *The Process Description glyph for* `submap`*. (Upper part) folded submap. (Lower part) content of the submap.*

Figure 3.25 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.
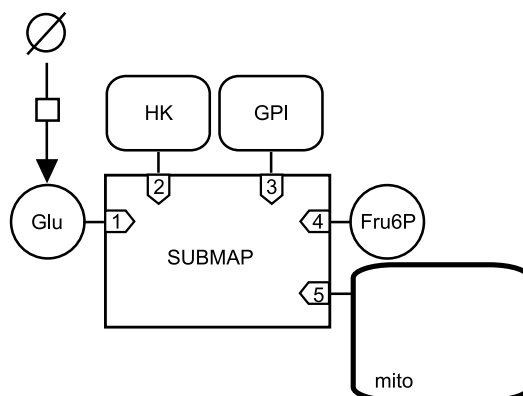
Figure 3.25: *Example of a submap with contents elided.*

The map in Figure 3.26 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`.
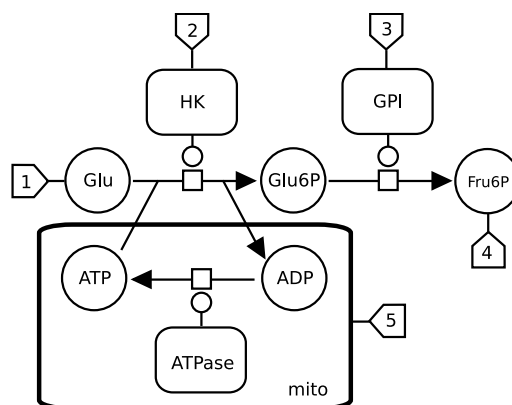
Figure 3.26: *Example of an unfolded submap. The unfolded submap corresponds to the folded submap of Figure 3.25.*

Logical operators 523

Glyph: And The glyph and is used to denote that all the EPNs linked as input are necessary to 524
produce the output. 525
SBO Term: SBO:0000173 ! and. 526
Origin: More than one EPN (section 3.2.4) or logical operator (section 3.2.15). 527
Target: One modulation (section 3.2.19), stimulation (section 3.2.19), catalysis (section 3.2.19), inhi- 528
bition (section 3.2.19) or necessary stimulation (section 3.2.19) arc. 529
Node: And is represented by a circle carrying the word "AND". 530
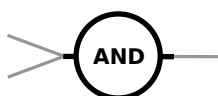
Figure 3.27: *The Process Description glyph for and. Only two inputs are represented, but more would be allowed.*

Glyph: Or The glyph or is used to denote that any of the EPNs linked as input is sufficient to 531
produce the output. 532
SBO Term: SBO:0000174 ! or. 533
Origin: More than one EPN (section 3.2.4) or logical operator (section 3.2.15). 534
Target: One modulation (section 3.2.19), stimulation (section 3.2.19), catalysis (section 3.2.19), inhi- 535
bition (section 3.2.19) or necessary stimulation (section 3.2.19) arc. 536
Node: Or is represented by a circle carrying the word "OR". 537
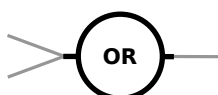

Figure 3.28: *The Process Description glyph for or. Only two inputs are represented, but more would be allowed.*

Glyph: Not The glyph not is used to denote that the EPN linked as input cannot produce the 538
output. 539
SBO Term: SBO:0000238 ! not. 540
Origin: One EPN (section 3.2.4) or logical operator (section 3.2.15). 541

Target: One modulation (section 3.2.19), stimulation (section 3.2.19), catalysis (section 3.2.19), inhi- 542
bition (section 3.2.19) or necessary stimulation (section 3.2.19) arc. 543
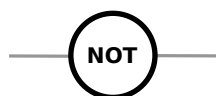**Node:** `Not` is represented by a circle carrying the word "NOT". 544

Figure 3.29: *The Process Description glyph for* `not`.

Stoichiometric process 545

A process with measurable change in the quantities of entity pools consumed and produced. 546

Glyph: `Process` A generic stoichiometric process that transforms a set of entity pools (represented 547
by EPNs in SBGN Process Description Level 1) into another set of entity pools. 548
SBO Term: SBO:0000375 ! process 549
Origin: One or several `consumption` arcs (Section 3.2.19) or one or several `production` arcs (Sec- 550
tion 3.2.19). 551
Target: One or several `production` arcs (Section 3.2.19). 552
Node: A process is represented by a square box linked to two connectors, small arcs attached to the 553
centers of opposite sides. The consumption (Section 3.2.19) and production (Section 3.2.19) 554
arcs are linked to the extremities of those connectors. The modulatory arcs (Section 3.2.19) 555
point to the other two sides of the box. A `process` connected to `production` arcs on opposite 556
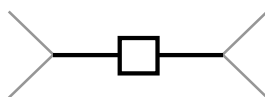sides is a reversible process. 557

Figure 3.30: *The Process Description glyph for* `process`.

A process is the basic process node in SBGN. It describes a process that transforms a given set of 558
biochemical entities—macromolecules, simple chemicals or unspecified entities—into another set 559
of biochemical entities. Such a transformation might imply modification of covalent bonds (conver- 560
sion), modification of the relative position of constituents (conformational process) or movement 561
from one compartment to another (translocation). 562

A cardinality label may be associated with `consumption` (Section 3.2.19) or `production` (Sec- 563
tion 3.2.19) arcs to indicate the stoichiometry of the process. This label becomes a requirement when 564
the exact composition of the number of copies of the inputs or outputs to a reaction are ambiguous 565
in the map. 566

A process is regarded as reversible if both 'sides' of the process are connected to `production` arcs 567
(see section 4.2.2). 568

The example in Figure 3.31 illustrates the use of a `process` node to represent the phosphorylation 569
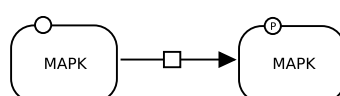of a protein in a Process Description. 570

Figure 3.31: *Phosphorylation of the protein MAP kinase.*

The example in Figure 3.32 illustrates the use of a `process` node to represent a reaction between 571
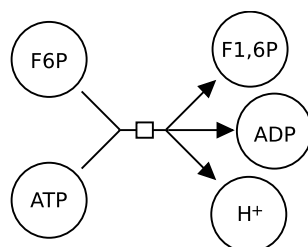two reactants that generates three products. 572

Figure 3.32: *Reaction between ATP and fructose-6-phosphate to produce fructose-1,6-biphosphate, ADP and a proton.*

The example in Figure 3.33 illustrates the use of a `process` node to represent a translocation. The 573
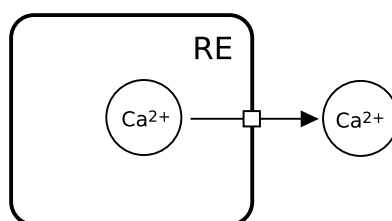large round-cornered rectangle represents a compartment border (see Section 3.2.16). 574

Figure 3.33: *Translocation of calcium ion out of the endoplasmic reticulum. Note that the `process` does not have to be located on the boundary of the `compartment`. A `process` is not attached to any `compartment`.*

The example in Figure 3.34 illustrates the use of a `process` node to represent the reversible open- 575
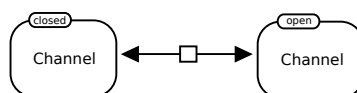ing and closing of an ionic channel in a Process Description. 576

Figure 3.34: *Reversible opening and closing of an ionic channel.*

When such a reversible process is asymmetrically modulated, it must be represented by two dif- 577
ferent processes in a Process Description. Figure 3.35 illustrates the use of two `process` nodes to 578
represent the reversible activation of a G-protein coupled receptor. In the absence of any effector, 579
an equilibrium exists between the inactive and active forms. The agonist stabilises the active form, 580
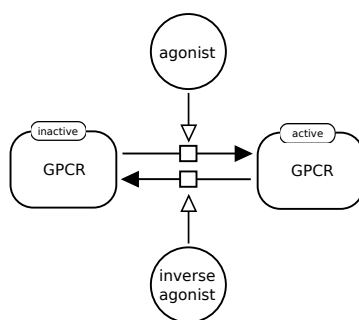while the inverse agonist stabilises the inactive form. 581

Figure 3.35: *The reversible activation of a G-protein coupled receptor.*

The example in Figure 3.36 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.
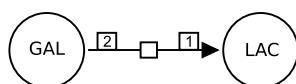

Figure 3.36: *Conversion of two galactoses into a lactose.*

Glyph: `Omitted process` Omitted processes are processes that are known to exist, but are omitted from the map 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` references to an explicit content, that is hidden in the main map, the `omitted process` does not "hide" anything within the context of the map, and cannot be "unfolded".

SBO Term: SBO:0000397 - omitted process.

Origin: One or several `consumption` arcs (Section 3.2.19) or one or several `production` arcs (Section 3.2.19).

Target: One or several `production` arcs (Section 3.2.19).

Node: An `omitted process` is represented by a `process` in which the square box contains a two parallel slanted lines oriented northwest-to-southeast and separated by an empty space.
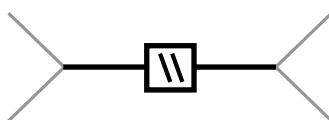

Figure 3.37: *The Process Description glyph for `omitted process`.*

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

SBO Term: SBO:0000396 ! uncertain process.

Origin: One or several `consumption` arcs (Section 3.2.19) or one or several `production` arcs (Section 3.2.19).

Target: One or several `production` arcs (Section 3.2.19).

Node: An `uncertain process` is represented by a `process` which square box contains a question mark.
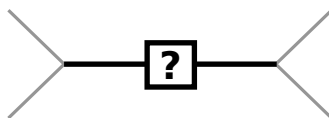
Figure 3.38: *The Process Description glyph for an* `uncertain process`.

Glyph: `Association`   The association between one or more EPNs represents the non-covalent bind-   604
ing of the biological objects represented by those EPNs into a larger complex.   605
SBO Term: SBO:0000177 ! non-covalent binding.   606
Origin: One or more `consumption` arcs (Section 3.2.19).   607
Target: One `production` arc (Section 3.2.19).   608
Node: An `association` between several entities is represented by a filled disc linked to two connec-   609
tors, small arcs attached on point separated by 180 degrees. The consumption (Section 3.2.19)   610
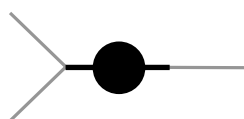and production (Section 3.2.19) arcs are linked to the extremities of those connectors.   611

Figure 3.39: *The Process Description glyph for* `association`.

The example in Figure 3.40 illustrates the association of cyclin and CDC2 kinase into the Matura-   612
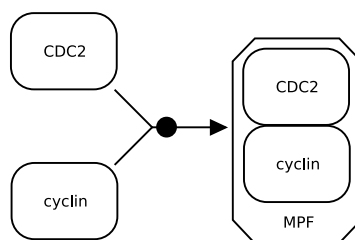tion Promoting Factor.   613

Figure 3.40: *Association of cyclin and CDC2 kinase into the Maturation Promoting Factor.*

Figure 3.41 gives an example illustrating the association of a pentameric macromolecule (a nico-   614
tinic acetylcholine receptor) with a simple chemical (the local anesthetic chlorpromazin) in an un-   615
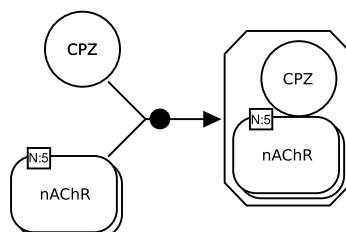named complex.   616

Figure 3.41: *The association of a pentameric macromolecule with a simple chemical in an un-named 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). 3.42 gives an example of this, using the formation of hemoglobin.
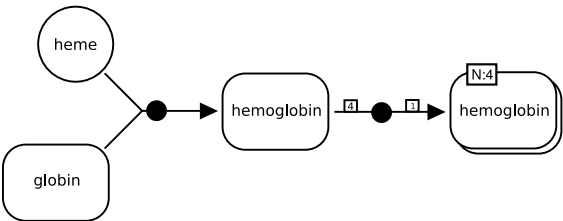

Figure 3.42: *Formation of hemoglobin.*

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 3.2.19).
Target: One or more `production` arc (Section 3.2.19).
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 3.2.19). Moreover, the existence of two circles reminds the dissociation, by contrast with the filled disc of the `association` (Section 3.2.15).

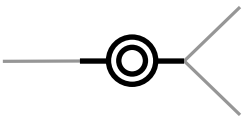**Figure 3.43:** *The Process Description glyph for `dissociation`.*

The example in Figure 3.44 illustrates the dissociation of the small and large ribosomal subunits from a messenger RNA.
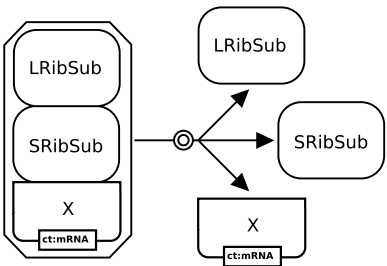

Figure 3.44: *Dissociation of the small and large ribosomal subunits from a messenger RNA.*

3.2.16 Container

Canvas

Identifying Attributes:

- identity

Special constraints or rules:

If one `Compartment` is drawn on the canvas then all EPNs must be drawn in a `Compartment` and cannot be drawn directly onto the Canvas[1]. 636 637

SBO Term: Not assigned 638

Container: None 639

Label: None 640

Permitted Child Glyphs: The Canvas can have `Container` glyphs and any EPN, Process and Arcs as children. However all Auxiliary glyphs cannot be children of the Canvas. 641 642

Cloning: N/A 643

Glyph: `Compartment` 644

A compartment is a logical or physical structure that contains entity pool nodes. An EPN can only belong to one compartment. Therefore, the "same" biochemical species located in two different compartments are in fact two different pools. 645 646 647

SBO Term: SBO:0000290 ! physical compartment 648

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. 649 650 651 652

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. 653 654 655 656

Permitted Child Glyphs: 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 3.2.17. The center of the bounding box of a `unit of information` is located on the mid-line of the border of the compartment. 657 658 659 660
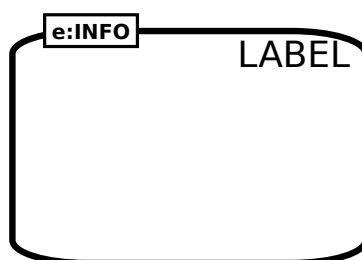

Figure 3.45: *The Process Description glyph for* `compartment`*.*

To allow more aesthetically pleasing and understandable maps, 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 3.46 shows two semantically equivalent placement of compartments: 661 662 663 664

---

[1]All EPNs must beling to a compartment. When no compartment is drawn then all EPNs can be thought of as belonging to an imaginary "default" compartment. But when one or more EPNs are assigned to an explicit compartment then for clarity all compartments must be explicity drawn and EPNs assigned to them.
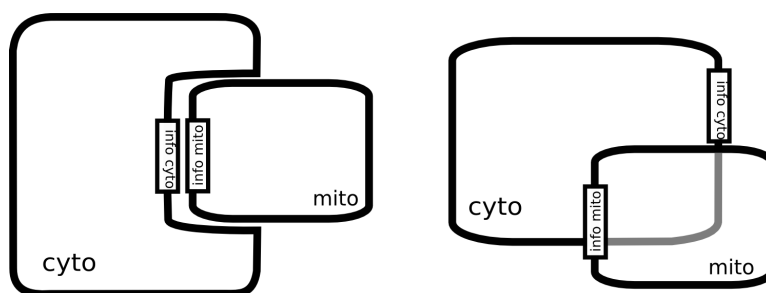
Figure 3.46: *Overlapped compartments are permitted, but the overlap does not imply containment.*

Overlapped (hidden) part of the compartment should not contain any object which could be covered by an overlapping compartment. Figure 3.47 illustrates the problem using an incorrect map.
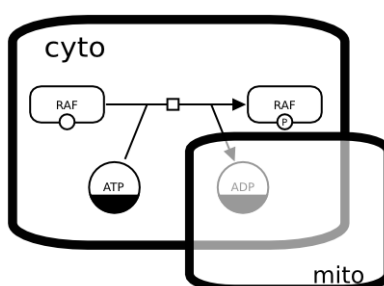

Figure 3.47: *Example of an **incorrect** map. Overlapped compartments must not obscure other objects.*

3.2.17 Auxiliary units

Auxiliary units are glyphs that decorate other glyphs, providing additional information that may be useful to the reader. These can provide annotation (`unit of information`), state information (`state variable`) or indicate duplication of entity pool nodes (`clone marker`).

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 `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 midline 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 prede-

fined types of information having controlled vocabularies associated with them, SBGN de-   689
fines specific prefixes that must be included in the label to indicate the type of information   690
in question. The controlled vocabularies predefined in SBGN Process Description Level 1 are   691
described in Section 3.4 and summarized in the following list: 692

pc container physical characteristic   693
mt entity pool material type   694
ct entity pool conceptual type   695
N multimer cardinality   696

Permitted Child Glyphs: A `unit of information` does not carry any auxiliary items.   697
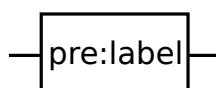

Figure 3.48: *The Process Description glyph for* `unit of information`*.*

Glyph: `State variable`   698

Many biological entities such as molecules can exist in different *states*, meaning different physical or   699
informational configurations. These states can arise for a variety of reasons. For example, macro-   700
molecules can be subject to post-synthesis modifications, wherein residues of the macromolecules   701
(amino acids, nucleosides, or glucid residues) are modified through covalent linkage to other chem-   702
icals. Other examples of states are alternative conformations as in the closed/open/desensitized   703
conformations of a transmembrane channel, and the active/inactive forms of an enzyme.   704

SBGN provides a means of associating one or more `state variables` with an entity; each such   705
variable can be used to represent a dimension along which the state of the overall entity can vary.   706
When an entity can exist in different states, the state of the whole entity (i.e., the SBGN object) can be   707
described by the current values of all its `state variables`, and the values of the `state variables`   708
of all its possible components, recursively.   709
Identifying Attributes: 710

- owning EPN   711
- name (optional)   712
- value   713

Special constraints or rules: 714
None.   715
SBO Term: Not applicable.   716
Container: A `state variable` is represented by a '"stadium" container, that is two hemicercles   717
of same radius joined by parallel segments, as shown in Figure 3.49. The parallel segment   718
axis should be tangent to the border of the glyph of the EPN being modified by the `state`   719
`variable`. The center of the bounding box of a `state variable` should be located on the   720
mid-line of the border of the EPN.   721
Label: The identification of an instance of a `state variable` is carried by one or two unbordered   722
boxes, each containing a string of characters. The characters cannot be distributed on several   723
lines. One box is mandatory, and contains the value of the `state variable`. The value may be   724
empty; an example of a situation where this might arise is an unphosphorylated phosporyla-   725
tion site. The second box is optional and carries the identification of the `state variable`. The   726
center of the combination of the boxes located in the container box is superposed to the center   727
of this container box. In earier version of the Process Description specification, the identifi-   728
cation of the `state variable` could be located outside the `state variable` container box.   729
This is now forbidden. The style of labeling of `state variables` encouraged by SBGN Process   730

Description Level 1 is to combine a prefix representing the value of the variable with a suffix   731
representing the variable's name. Prefix and suffix should be separated by the symbol '@', X@Y   732
thus meaning *value X* AT *variable Y*. The label of a `state variable` should, if possible, be   733
displayed within the boundary of the glyph.   734

Permitted Child Glyphs: A `state variable` does not carry any auxiliary items.   735
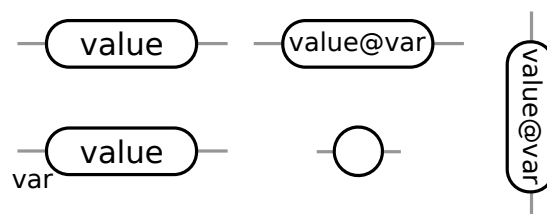
Cloning: Not cloned.   736

Figure 3.49: *Examples of the Process Description glyph for* `state variable`.

A `state variable` does not necessarily have to be Boolean-valued. For example, an ion chan-   737
nel can possess several conductance states; a receptor can be inactive, active and desensitized; and   738
so on. As another example, a `state variable` "ubiquitin" could also carry numerical values cor-   739
responding to the number of ubiquitin molecules present in the tail. However, in all cases, a `state`   740
`variable` on an EPN can only take *one* defined value. Further, an EPN's `state variable` should   741
always be displayed and always set to a value. An "empty" `state variable` is a `state variable`   742
that is set to the value "unset", it is not a `state variable` with no value. Note that the value "unset"   743
is *not* synonymous to "any value" or "unknown value".   744

Glyph: `Annotation`   745

SBGN Process Description Level 1 defines a glyph to add additional information to a map, that does   746
not modify the semantic of the the graph. This glyph can be used to add free text, or links to external   747
information.   748

SBO Term: SBO:NEW   749

Container: An `annotation` is represented by a rectangular container with a folded corner, as illus-   750
trated in Figure 3.50. This container is linked to the annotated element in a way that cannot be   751
mistaken for a relationship, for instance a callout, a thick edge, a dashed line etc. The link ends   752
up on the border of the annotated element.   753

Label: An `annotation` contains information placed in an unbordered box containing a string of   754
characters. The characters can be distributed on several lines to improve readability, although   755
this is not mandatory. The label box must be attached to the center of the container. The label   756
may spill outside of the container.   757

Permitted Child Glyphs: An `annotation` does not carry any auxiliary unit.   758

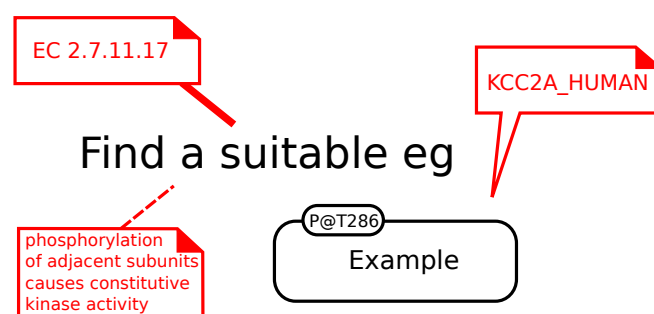**Figure 3.50:** *The Process Description glyph for* `annotation`.

Figure 3.51: *Example of* `annotations` *adding information to the description of the trans-phosphorylation of CaMKII. Note that three different types of links are used between annotation nodes and annotated elements. However, it is recommended to use a consistent scheme whithin a map.*

3.2.18 Reference nodes   759

Reference nodes handle links or relationships between elements of a map and sub-map. At present   760
there is only one reference glyph, `tag`, which can be used in a map refered to by a `submap` (Sec-   761
tion 3.2.15) or as an auxilary unit on the `submap`. The `clone marker` can also provide additional   762
reference mechanisms and is discussed below (Section 3.3.1).   763

Glyph: `Tag`   764

A `tag` is a named handle, or reference, to another EPN (Section 3.2.4) or compartment (Section 3.2.16).   765
`Tags` are used to identify those elements in `submaps` (Section 3.2.15).   766
SBO Term: Not applicable.   767
Container: A `tag` is represented by a rectangle fused to an empty arrowhead, as illustrated in Fig-   768
   ure 3.52. The symbol should be linked to one and only one edge (i.e., it should reference only   769
   one EPN or compartment).   770
Label: A `tag` is identified by a label placed in an unbordered box containing a string of characters.   771
   The characters can be distributed on several lines to improve readability, although this is not   772
   mandatory. The label box must be attached to the center of the container. The label may spill   773
   outside of the container.   774
Permitted Child Glyphs: A `tag` does not carry any auxiliary items.   775

Figure 3.52: *The Process Description glyph for* `tag`.

3.2.19 Arcs   776

Arcs are lines that link `EPNs` and `PNs` together. The symbols attached to their extremities indicate   777
their semantics.   778

Process Arcs   779

Glyph: `Consumption` `Consumption` is the arc used to represent the fact that an entity pool is con-   780
sumed by a process, but is not produced by the process.   781
SBO Term: SBO:0000394 ! consumption.   782
Origin: Any EPN (Section 3.2.4).   783
Target: Any `process node` (Section 3.2.15).   784

End point: No particular symbol is used to represent a consumption. 785

A cardinality label may be associated with `consumption` (Section 3.2.19) or `production` (Section 3.2.19) arcs, indicating the stoichiometry of a process. This label is a number enclosed in a 787 rectangle with one of the long sides adjacent to the consumption arc. The cardinality is required to 788 eliminate ambiguity when the exact composition, or the number of copies, of the inputs or outputs 789 to a reaction are ambiguous from the map. An example is a multimer of 6 subunits dissociating into 790 2 monomers and 2 dimers. Without stoichiometry labels another result, such as 4 monomers and 1 791 dimer could be inferred. Once assigned to one arc connecting to a process node, cardinality should 792 be represented on all `consumption` and `production` arcs connected to that process node to avoid 793 misinterpretation. 794

Omitted cardinality on one edge only should not be treated as cardinality of 1, but as an unspec- 795 ified cardinality. In most cases, the exact value may be derived from the context, but unless cardinal- 796 ity is explicitly shown, it should be considered as unspecified. In the case where the stoichiometry of 797 some part of the process is not known, or undefined, a question mark (?) should be used within the 798 cardinality label of the corresponding arcs. 799

Figure 3.53: *The Process Description glyph for* `consumption`.

Glyph: `Production` `Production` is the arc used to represent the fact that an entity pool is produced 800 by a process. In the case of a reversible process, the `production` arc also acts as a `consumption` arc. 801
SBO Term: SBO:0000393 ! production. 802
Origin: Any `process node` (Section 3.2.15). 803
Target: Any EPN (Section 3.2.4). 804
End point: The target extremity of a `production` carries a filled arrowhead. 805

A cardinality label can be associated with a `production` arc indicating the stoichiometry of a 806 process. 807

Figure 3.54: *The Process Description glyph for* `production`.

Figure 3.55 illustrates the use of consumption/production arc cardinality labels to represent the 808 stoichiometry of a process. 809
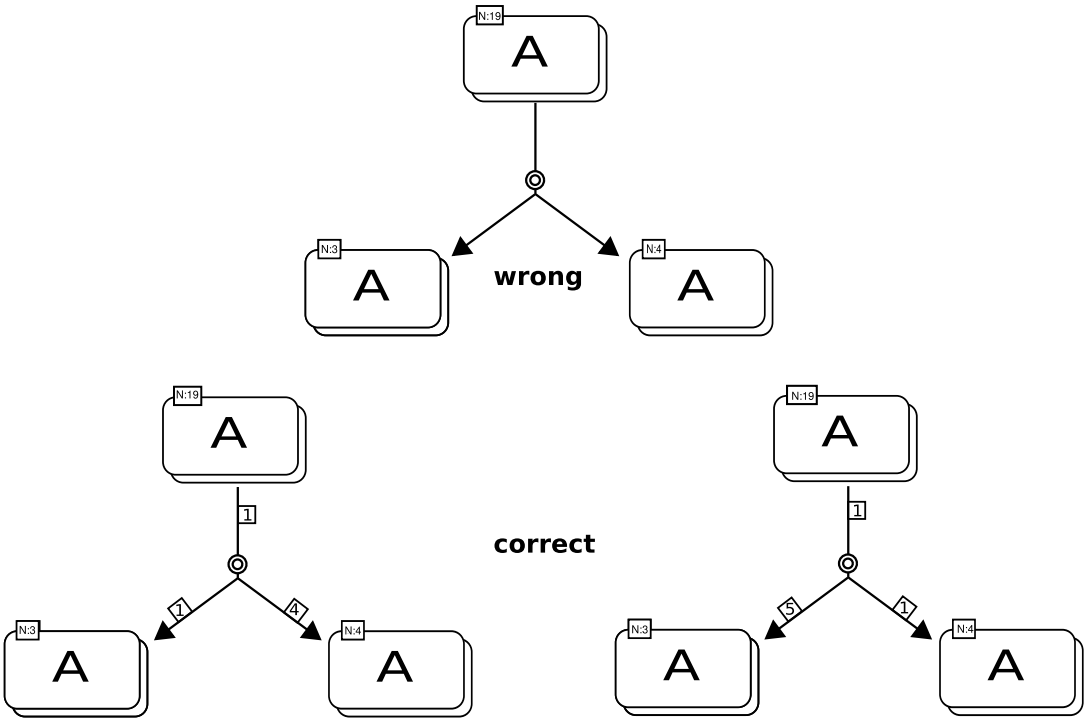
Figure 3.55: *Cardinality for production arcs.*

Modulation Arcs 810

Glyph: Modulation A modulation affects the flux of a process represented by the target process. 811
Such a modulation can affect the process **positively or negatively**, or even both ways depending on 812
the conditions, for instance the concentration of the intervening participants. A modulation can 813
also be used when one does not know the precise direction of the effect. 814
SBO Term: SBO:0000168 ! control. 815
Origin: Any EPN (Section 3.2.4) or any logical operator (Section 3.2.15). 816
Target: Any process node (Section 3.2.15). 817
End point: The target extremity of a modulation carries an empty diamond. 818

Figure 3.56: *The Process Description glyph for modulation.*

Figure 3.57 represents the effect of nicotine on the process between closed and open states of a 819
nicotinic acetylcholine receptor. High concentrations of nicotine open the receptor while low con- 820
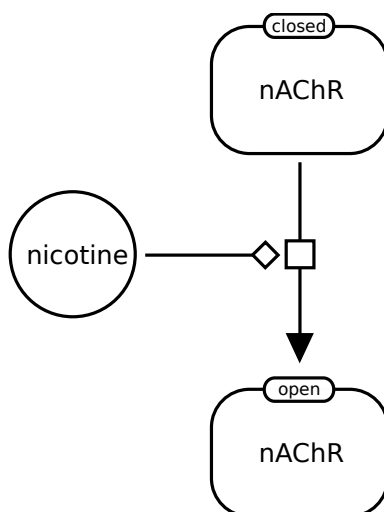centrations can desensitize it without opening. 821

Figure 3.57: *Modulation of nicotinic receptor opening by nicotine.*

Glyph: `Stimulation` A stimulation affects **positively** the flux of a process represented by the target   822
process. This stimulation can be for instance a catalysis or a positive allosteric regulation. Note that   823
`catalysis` exists independently in SBGN, see Section 3.2.19.   824
SBO Term: SBO:0000170 ! stimulation.   825
Origin: Any EPN (Section 3.2.4) or any logical operator (Section 3.2.15).   826
Target: Any `process node` (Section 3.2.15).   827
End point: The target extremity of a `stimulation` carries an empty arrowhead.   828

Figure 3.58: *The Process Description glyph for `stimulation`.*

Glyph: `Catalysis` A catalysis is a particular case of stimulation, where the effector affects positively   829
the flux of a process represented by the target process. The positive effect on the process is due to the   830
lowering of the activation energy of a reaction.   831
SBO Term: SBO:0000172 ! catalysis.   832
Origin: Any EPN (Section 3.2.4) or any `logical operator` (Section 3.2.15).   833
Target: Any `process node` (Section 3.2.15).   834
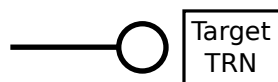**Node:** The target extremity of a `catalysis` carries an empty circle.   835

Figure 3.59: *The Process Description glyph for `catalysis`.*

Glyph: `Inhibition` An inhibition **negatively** affects the flux of a process represented by the target   836
process. This inhibition can be for instance a competitive inhibition or an allosteric inhibition.   837
SBO Term: SBO:0000169 ! inhibition.   838
Origin: Any EPN (Section 3.2.4) or any `logical operator` (Section 3.2.15).   839
Target: Any `process node` (Section 3.2.15).   840
Node: The target extremity of an `inhibition` carries a bar perpendicular to the arc.   841
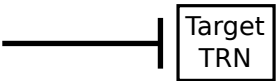
Figure 3.60: *The Process Description glyph for* `inhibition`.

Glyph: `Necessary stimulation` A necessary stimulation, is one that is necessary for a process   842
to take place. A process modulated by a necessary stimulation can only occur when this necessary   843
stimulation is active.   844
SBO Term: SBO:0000171 ! necessary stimulation.   845
Origin: Any EPN (Section 3.2.4) or any `logical operator` (Section 3.2.15).   846
Target: Any `process node` (Section 3.2.15).   847
Node: The target extremity of a `necessary stimulation` carries an open arrow (to remind that it   848
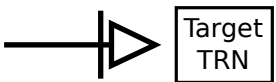is a `stimulation`) coming after a larger vertical bar.   849

Figure 3.61: *The Process Description glyph for* `Necessary Stimulation`.

The example in Figure 3.62 below describes the transcription of a gene X, that is the creation of   850
a messenger RNA X triggered by the gene X. The creation of the protein X is then triggered by the   851
mRNA X. (Note that the same example could be represented using the gene as reactant and product,   852
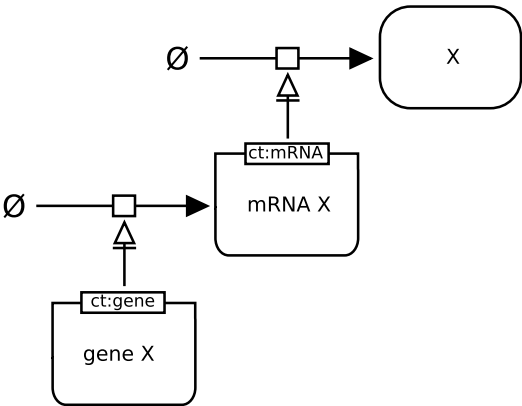although it is semantically different.)   853

Figure 3.62: *The creation of a messenger RNA X triggered by the gene X.*

The example in Figure 3.63 below describes the transport of calcium ions out of the endoplasmic   854
reticulum. Without IP3 receptor, there is not calcium flux, therefore, one cannot use a `stimulation`.   855
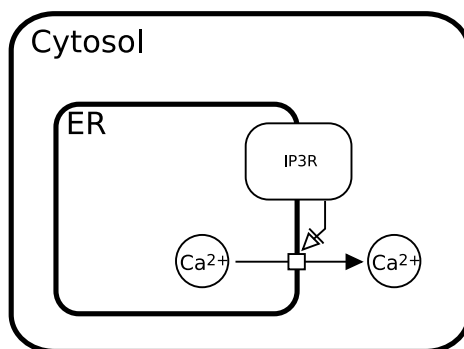The Necessary Stimulation instead represents this absolute stimulation.   856

Figure 3.63: *The transport of calcium ions out of the endoplasmic reticulum into the cytosol. Note that IP3R crosses both compartment boundaries. This is allowed, but the Macromolecule should only belong to one of the compartments see section C.1 for more discussion of this issue.*

Others 857

Glyph: `Logic arc` `Logic arc` is used to represent the fact that an entity influences the outcome 858
of a logic operator. 859
SBO Term: SBO:0000398 ! logical relationship. 860
Origin: Any EPN (Section 3.2.4) or `logical operator` (Section 3.2.15). 861
Target: Any `logical operator` (Section 3.2.15). 862
End point: No particular symbol is used to represent a logic arc. 863
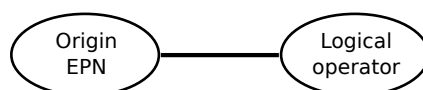

Figure 3.64: *The Process Description glyph for* `logic arc`.

Glyph: `Equivalence arc` `Equivalence arc` is the arc used to represent the fact that all entities 864
marked by a `tag` are equivalent. 865
SBO Term: Not applicable. 866
Origin: Any EPN (Section 3.2.4). 867
Target: `Tag` (Section 3.2.18). 868
End point: No particular symbol is used to represent an `equivalence arc`. 869
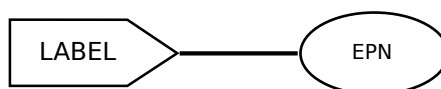

Figure 3.65: *The Process Description glyph for* `Equivalence arc`.

3.3 Decorators 870

3.3.1 Glyph: `Clone marker` 871

If an EPN is duplicated on a map, it is necessary to indicate this fact by using the `clone marker` 872
auxiliary unit. The purpose of this marker is to provide the reader with a visual indication that this 873
node has been cloned, and that at least one other occurrence of the EPN can be found in the map (or 874
in a submap; see Section 3.2.15). The clone marker takes two forms, simple and labeled, depending 875
on whether the node being cloned can carry state variables (i.e., whether it is a stateful EPN). Note 876
that an EPN belongs to a single compartment. If two glyphs labelled "X" are located in two different 877

compartments, such as ATP in cytosol and ATP in mitochondrial lumen, they represent different   878
EPNs, and therefore do not need to be marked as cloned.   879

Simple clone marker   880

As mentioned above, the `simple clone marker` is the unlabeled version of the `clone marker`. See   881
below for the labeled version.   882
SBO Term: Not applicable.   883
Container: The simple (unlabeled) `clone marker` is a portion of the surface of an EPN that has been   884
modified visually through the use of a different shade, texture, or color. Figure 3.66 illustrates   885
this. The `clone marker` occupies the lower part of the EPN. The filled area must be smaller   886
than the unfilled one.   887
Label: Not applicable.   888
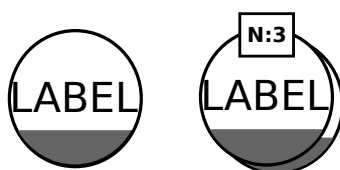**Permitted Child Glyphs:** A `clone marker` does not carry any auxiliary items.   889

Figure 3.66: *The Process Description glyph for* `simple clone marker` *applied to a* `simple chemical` *and a* `multimer` *of* `simple chemicals`*.*

Labeled clone marker   890

Unlike the `simple clone marker`, the `labeled clone marker` includes (unsurprisingly, given its   891
name) an identifying label that can be used to identify equivalent clones elsewhere in the map. This   892
is particularly useful for stateful EPNs, because these can have a large number of state variables dis-   893
played and therefore may be difficult to visually identify as being identical.   894
SBO Term: Not applicable.   895
Container: The labeled `clone marker` is a portion of the surface of an EPN that has been modified   896
visually through the use of a different shade, texture, or color. The `clone marker` occupies the   897
lower part of the EPN glyph. The filled area must be smaller than the unfilled one, but the be   898
large enough to have a height larger than the `clone marker`'s label (cf below).   899
Label: A `clone marker` is identified by a label placed in an unbordered box containing a string of   900
characters. The characters can be distributed on several lines to improve readability, although   901
this is not mandatory. The label box must be attached to the center of the container. The label   902
may spill outside of the container (the portion of the surface of the EPN that has been modified   903
visually). The font color of the label and the color of the clone marker should contrast with one   904
another. The label on a `labeled clone marker` is mandatory.   905
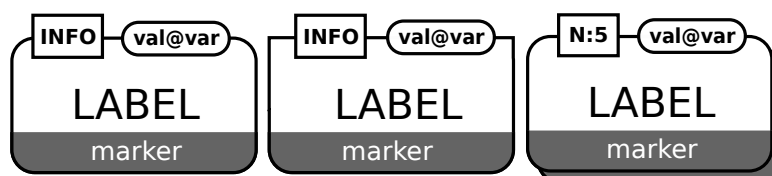**Permitted Child Glyphs:** A `clone marker` does not carry any auxiliary items.   906

Figure 3.67: *The Process Description glyph for* `labeled clone marker` *applied to a* `macromolecule`*, a* `nucleic acid feature` *and a* `multimer` *of* `macromolecules`*.*

Figure 3.68 contains an example in which we illustrate the use of `clone markers` to clone the 907
species ATP and ADP participating in different reactions. This example also demonstrates the chief 908
drawbacks of using clones: it leads to a kind of dissociation of the overall network and multiplies 909
the number of nodes required, requiring more work on the part of the reader to interpret the result. 910
Sometimes these disadvantages are offset in larger maps by a reduction in the overall number of line 911
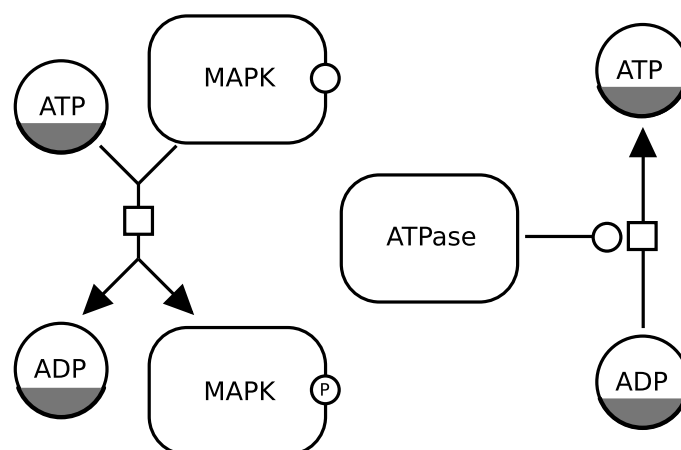crossings, but not always. In general, we advise that cloning should be used sparingly. 912

Figure 3.68: *An example of using cloning, here for the species ATP and ADP.*

3.4 Controlled vocabularies 913

Some glyphs in SBGN Process Descriptions can contain particular kinds of textual annotations con- 914
veying information relevant to the purpose of the glyph. These annotations are `units of information` 915
(Section 3.2.17) or `state variable` (Section 3.2.17). An example is in the case of multimers, which 916
can have a unit of information conveying the number of monomers composing the multimer. Other 917
cases are described throughout the rest of this chapter. 918

The text that appears as the unit of information decorating an Entity Pool Node (EPN) must in 919
most cases be prefixed with a controlled vocabulary term indicating the type of information being 920
expressed. The prefixes are mandatory except in the case of macromolecule covalent modifications 921
(Section 3.4.3). Without the use of controlled vocabulary prefixes, it would be necessary to have 922
different glyphs to indicate different classes of information; this would lead to an explosion in the 923
number of symbols needed. 924

In the rest of this section, we describe the controlled vocabularies (CVs) used in SBGN Process 925
Description Level 1. They cover the following categories of information: an EPN's material type, an 926
EPN's conceptual type, covalent modifications on macromolecules, the physical characteristics of 927
compartments, and cardinality (e.g., of multimers). In each case, some CV terms are predefined by 928
SBGN, but unless otherwise noted, *they are not the only terms permitted*. Authors may use other CV 929
values not listed here, but in such cases, they should explain the term's meanings in a figure legend 930
or other text accompanying the map. 931

3.4.1 Entity pool node material types 932

The material type of an EPN indicates its chemical structure. A list of common material types is 933
shown in Table 3.1, but others are possible. The values are to be taken from the Systems Biology On- 934
tology (`http://www.ebi.ac.uk/sbo/`), specifically from the branch having identifier `SBO:0000240` 935
(*material entity* under *entity*). The labels are defined by SBGN Process Description Level 1. 936

The material types are in contrast to the *conceptual types* (see below). The distinction is that 937
material types are about physical composition, while conceptual types are about roles. For example, 938

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

Table 3.1: *A sample of values from the* material types *controlled vocabulary (Section 3.4.1).*

a strand of RNA is a physical artifact, but its use as messenger RNA is a role.   939

3.4.2 Entity pool node conceptual types   940

An EPN's *conceptual type* indicates its function within the context of a given Process Description. A   941
list of common conceptual types is shown in Table 3.2, but others are possible. The values are to   942
be taken from the Systems Biology Ontology (http://www.ebi.ac.uk/sbo/), specifically from the   943
branch having identifier `SBO:0000241` (*conceptual entity* under *entity*). The labels are defined by   944
SBGN Process Description Level 1.   945

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

Table 3.2: *A sample of values from the* conceptual types *vocabulary (Section 3.4.2).*

3.4.3 Macromolecule covalent modifications   946

A common reason for the introduction of state variables (Section 3.2.17) on an entity is to allow ac-   947
cess to the configuration of possible covalent modification sites on that entity. For instance, a macro-   948
molecule may have one or more sites where a phosphate group may be attached; this change in the   949
site's configuration (i.e., being either phosphorylated or not) may factor into whether, and how, the   950
entity can participate in different processes. Being able to describe such modifications in a consistent   951
fashion is the motivation for the existence of SBGN's covalent modifications controlled vocabulary.   952
   Table 3.3 lists a number of common types of covalent modifications. The most common values   953
are defined by the Systems Biology Ontology in the branch having identifier `SBO:0000210` (*addition*   954
of a chemical group under *interaction→process→biochemical or transport reaction→*   955
biochemical reaction→conversion). The labels shown in Table 3.3 are defined by SBGN Process De-   956
scription Level 1; for all other kinds of modifications not listed here, the author of a Process Descrip-   957
tion must create a new label (and should also describe the meaning of the label in a legend or text   958
accompanying the map).   959

3.4.4 Physical characteristics   960

SBGN Process Description Level 1 defines a special unit of information for describing certain com-   961
mon physical characteristics. Table 3.4 lists the particular values defined by SBGN Process Descrip-   962

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

Table 3.3: *A sample of values from the* covalent modifications *vocabulary (Section 3.4.3).*

tion Level 1. It is anticipated that these will be used to describe the nature of a `perturbing agent` (section 3.2.9) or a `phenotype` (section 3.2.15).

Name	Label	SBO term
Temperature	pc:T	SBO:0000147
Voltage	pc:V	SBO:0000259
pH	pc:pH	SBO:0000304

Table 3.4: *A sample of values from the* physical characteristics *vocabulary (Section 3.4.4).*

3.4.5 Cardinality

SBGN Process Description Level 1 defines a special unit of information usable on multimers for describing the number of monomers composing the multimer. Table 3.5 shows the way in which the values must be written. Note that the value is an positive non-zero integer, and not (for example) a range. There is no provision in SBGN Process Description Level 1 for specifying a range in this context because it leads to problems of entity identifiability.

Name	Label	SBO term
cardinality	N:#	SBO:0000364

Table 3.5: *The format of the possible values for the* cardinality *unit of information (Section 3.4.5). Here, # stands for the number; for example, "N:5".*

Chapter 4

Validation Rules

4.1 Overview

The previous chapter descibed the glyphs and described rules specific to individual glyphs. In this chapter the rules that govern the interactions of the glyphs in SBGN Process Description Level 1 and that span an Process Description diagram as a whole.

4.2 Semantic rules

4.2.1 EPNs

1. All `state variables` associated with a Stateful Entity Pool Node should be unique and not duplicated within that node.

2. If a state variable is used in one EPN then is must be used in all equivalent stateful EPNs[1].

3. EPNs should not be orphaned (i.e. they must be associated with at least one arc.

4.2.2 Process Nodes

As described in Section 3.2.15, the `consumption` and `production` arcs converge before connecting to the process node (Figure 4.1). This defines the EPNs that are the input and outputs of an irreversible process. Since, processes can be reversible in the following rules we refer to these groupings as the "left-hand-side" (LHS) and "right-hand-side" (RHS) of the process[2]. For convenience we will also collectively refer to the `consumption` and `production` arcs as *flux* arcs.
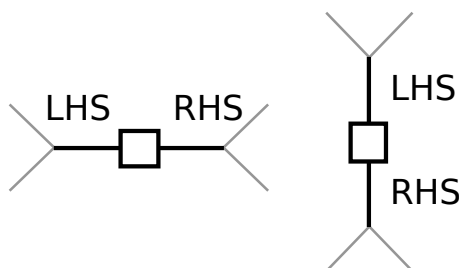
Figure 4.1: *An illustration of the "sidedness" of a process. The designation of LHS and RHS is essentially arbitrary.*

---

[1]A stateful EPN is equivalent if the EPNs are identical when their state descriptions are ignore.

[2]Note this designation is purely for grouping and is used even then the sides of the reaction are above and below the process.

Flux Arcs ₉₈₉

1. All process nodes (with the exception of `phenotype`) must have a LHS and RHS. ₉₉₀

2. All EPNs on the LHS of a process must be unique. ₉₉₁

3. All EPNs on the RHS of a process must be unique. ₉₉₂

4. All `phenotype` glyphs must be associated with at least one modulation arc. ₉₉₃

5. The EPNs that make up the LHS of the process should be consistent with the RHS, i.e. the process should constitute a balanced biochemical reaction. _{994 995}

6. Once the stoichiometry of a flux arc is displayed in a map then all other flux arcs should display theit stoichiometry make. _{996 997}

7. If the stoichiometry is undefined or unknown this should be indicated by the use of a question mark ("?"). _{998 999}

8. If more than one set of stoichiometries can be applied to the flux arcs of the process then the stoichiometry of the flux arcs must be displayed. _{1000 1001}

Association ₁₀₀₂

1. An `Association` is always an irreversible process. ₁₀₀₃

Dissociation ₁₀₀₄

1. An `Dissociation` is always an irreversible process. ₁₀₀₅

Modulation ₁₀₀₆

As discussed in Chapter 2, it is implied, but not defined explicitly that the process has a rate at which it converts its LHS EPNs to its RHS EPNs (and vice-versa in the case of a reversible process). This concept is important in understanding how the Process Description language describes process modulation. _{1007 1008 1009 1010}

1. A `process` with no modulations has an underlying "basal rate" which describes the rate at which it converts inputs to outputs. _{1011 1012}

2. A `modulation` changes the basal rate in an unspecified fashion. ₁₀₁₃

3. A `stimulation` is a modulation that increases the basal rate. ₁₀₁₄

4. An `inhibition` is a modulation that decreases the basal rate. ₁₀₁₅

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. _{1016 1017}

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. _{1018 1019}

7. At most one `necessary stimulation` can be assigned to a process node. Two `necessary stimulations` would imply an implicit AND or OR operator. For clarity only one `necessary stimulation` can be assigned to a `process`, and such combinations must be explicitly expressed using `logical operators`. _{1020 1021 1022 1023}

8. At most one `catalysis` can be assigned to a `process`. Modulation by a catalysis arc implies that the exact biochemical mechanism underlying the process is known. In this context two `catalysis` cannot be assigned to the same process node as they represent independent reactions. Other EPNs can be assigned to the same process as a catalysis, such as modulators, stimulators, and inhibitors, and will have a multiplicative modulation on the reaction rate defined by the catalysis.

Reversible Processes

A process is deemed to be reversible if it has `production` arcs on both the LHS and RHS of a process node Figure 4.2. Semantically, the `production` arc can be thought of as allowing a reversible flow of entities between the `process` and the EPN. A `consumption` arc only permits an irreversible flow from the EPN to the `process`. In this way, the `consumption` arc forces the `process` to be irreversible. `Consumption` arcs cannot be associated with both sides of a `process` as this would prohibit any flow through the `process`.
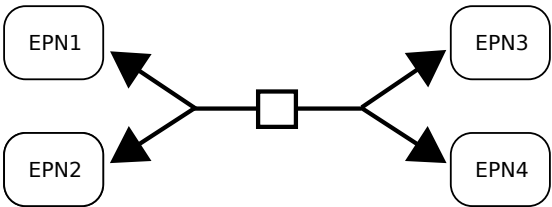
Figure 4.2: *A valid reversible process. A process is reversible if its LHS and RHS contain only* `production` *arcs.*

1. A mixture of `consumption` and `production` arcs on the same side of a `process` is not permitted.

2. A `sink` cannot be linked to a reversible process as it only receives entities, and so would effectively make the process irreversible[3].

3. The semantics of `modulation` is the same as for irreversible processes, .i.e. the amount of entity in the modulation pool affects the rate of the process.

4.2.3 Cloning

SBGN allows identical nodes to be duplicated on a map if they are explicitly marked as such. This is done using a `clone marker`. The details are shown in table 4.1.

Table 4.1: *Duplication rules.*

Node	Can be duplicated	Indication	Additional Rules
Compartment	N		
SimpleChemical	Y	Simple clone marker	
UnspecifiedEntity	Y	Simple clone marker	
Source	N		
Sink	N		
Perturbing Agent	Y	Simple clone marker	
Phenotype	N		
MultimerChemicalEntity	Y	Simple clone marker	
StatefulEntityPool	Y	Labeled clone marker	
Macromolecule	Y	Labeled clone marker	
MultimerMacromolecule	Y	Labeled clone marker	
			continued on next page

---

[3]A `source` can only be associated with a `consumption` arc so this rule does not apply in this case.

		continued from previous page	
Node	Duplicate?	Indication	Additional Rules
NucleicAcidFeature	Y	`Labeled clone marker`	
Complex	Y	`Labeled clone marker`	
Subunit	N		
Process	N		
OmittedProcess	N		
UncertainProcess	N		
Association	N		
Dissociation	N		
LogicalOperator	Y	None	
AND	Y	None	
OR	Y	None	
NOT	Y	None	

4.2.4 Compartment spanning

1048

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. A `complex` may contain EPNs which belong to different `compartments` and in this way a `complex` can be used to describe entities that span more than one compartment.

This restriction makes it impossible to represent in a semantically correct way a macromolecule that spans more then 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 pool 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.

4.2.5 Submaps

1066

The submap is a visual device that allows the detail of an Process Description map to be exported into another Process Description map and replaced by a `submap` glyph, which acts as a place-holder. This is described and illustrated in Section 3.2.15. In the following discussion we will refer to the original map as the *main* map and the map containing the export detail as the submap.

1. For a valid mapping between an EPN in the map and submap to exist the identifiers in the `tag` and the submap terminal must be identical and their associated entity pool nodes must be identical.

2. If the same EPN is present in the map and a submap, then they must be mapped to each other.

3. Since the main map and submap share the same namespace, an EPN that is cloned in the main map must also be marked as cloned in the submap — even if there is only one copy of the EPN in the submap. The converse applies when the EPN in the submap is cloned[4].

---

[4]This has the additional benefit of ensuring that main maps and submaps do not need to be modified if the submap is exanded and collapsed by a viewing or editing tool.

Chapter 5

Layout Rules for a Process Description

5.1 Introduction

The previous chapters describe the appearance and meaning of SBGN Process Description Level 1 components. Here we describe rules governing the visual appearence and asthetics of the Process Description language. The components of a Process Description 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 maps, may reveal new insights. The arrangement of components in a map is called a *layout*.

SBGN Process Descriptions 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 rules for the layout of process description maps, 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 rules are independent of the method used to produce the map, and apply to both manually drawn maps as well as maps produced by an automatic layout algorithm. The rules 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 [1] and Kaufmann and Wagner [2].

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

5.2 Requirements

Requirements are rules which **must** be fulfilled by a layout to produce a valid Process Description map.

5.2.1 Node-node overlaps

Nodes are only allowed to overlap in two cases when they are allowed to contain other nodes — as described in Chapter 3. Otherwise, nodes are not allowed to overlap (Figure 5.1). This includes the

touching of nodes. Touching is not allowed apart from the case where it has a specific meaning, e.g. 1112
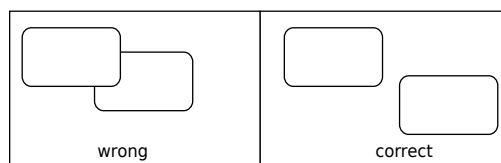two macromolecules touching each other within a complex because they form the complex. 1113

Figure 5.1: *Nodes must not overlap.*

5.2.2 Node-edge crossing 1114

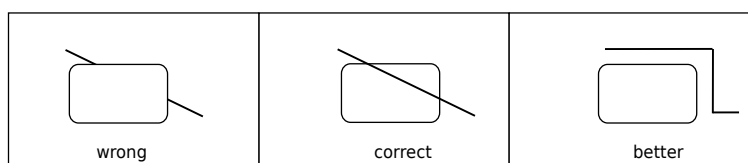Edges must be drawn on the top of a the node (Figure 5.2). See also recommendation Section 5.3.1. 1115

Figure 5.2: *If an edge crosses a node, the edge must be drawn on top of the node.*

5.2.3 Node border-edge overlaps 1116

Edges are not allowed to overlap the border lines of nodes (Figure 5.3). 1117
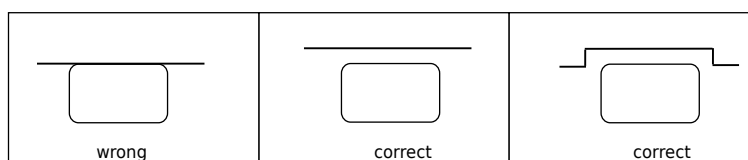
Figure 5.3: *Edges must not overlap node borders.*

5.2.4 Edge-edge overlaps 1118

Edges are not allowed to overlap (Figure 5.4). This includes touching of edges. Furthermore, an edge 1119
is neither allowed to cross itself nor to cross a boundary of node more than twice or other edges more 1120
than once. 1121

5.2.5 Node orientation 1122

Nodes have to be drawn horizontally or vertically, any other rotation of elements is not allowed (Fig- 1123
ure 5.5). 1124

5.2.6 Node-edge connection 1125

1. The arcs linking the square glyph of a `process` to the `consumption` and `production arcs` are 1126
   attached to the center of opposite sides (Figure 5.6). 1127
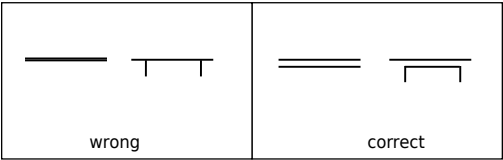
Figure 5.4: *Edges must not overlap.*
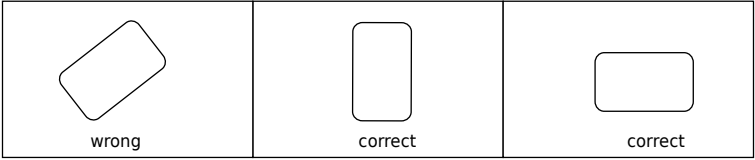
Figure 5.5: *The node orientation must be horizontally or vertically.*

2. The modulatory arcs are attached to the other two sides, but not necessarily all to the center, as several modifiers can affect the same process node.
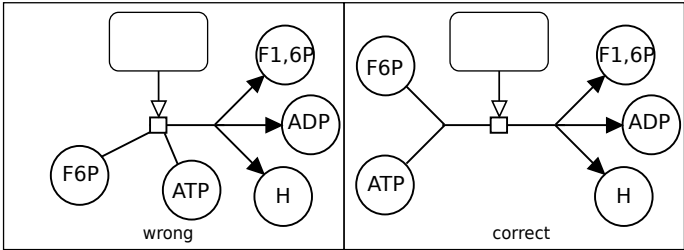

Figure 5.6: *Arcs between a* `process` *and the* `consumption` *and* `production arcs` *must be attached to the center of opposite sides, modulatory arcs must be attached to the other two sides.*

5.2.7 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 other nodes or other labels (this includes touching of other nodes or labels).

5.2.8 Edge labels

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

5.2.9 Compartments

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

5.3 Recommendations

Recommendations are rules which should be followed if possible and generally should improve the clarity of the diagram.

5.3.1 Node-edge crossing 1144

Situations where edges and nodes cross should be avoided. Note that some crossings may be un- 1145 avoidable, e.g. the crossing between an edge and a compartment border or an edge and a complex 1146 (if the edge connects an element inside the complex with something outside). 1147

5.3.2 Labels 1148

Labels should be horizontal. Node labels should be placed completely inside the node if possible. 1149 Edge labels should be placed close to the edge and avoid overlapping the edge as well as other edge 1150 labels. 1151

5.3.3 Avoid edge crossings 1152

The amount of crossings between edges should be minimized. 1153

5.3.4 Branching of `association` and `dissociation` 1154

The branching points of `association` and `dissociation` nodes should be placed closed to the 1155 symbol of the process, if possible at a distance comparable than, or smaller to, the diameter of the 1156 symbol defining the process (Figure 5.7). 1157
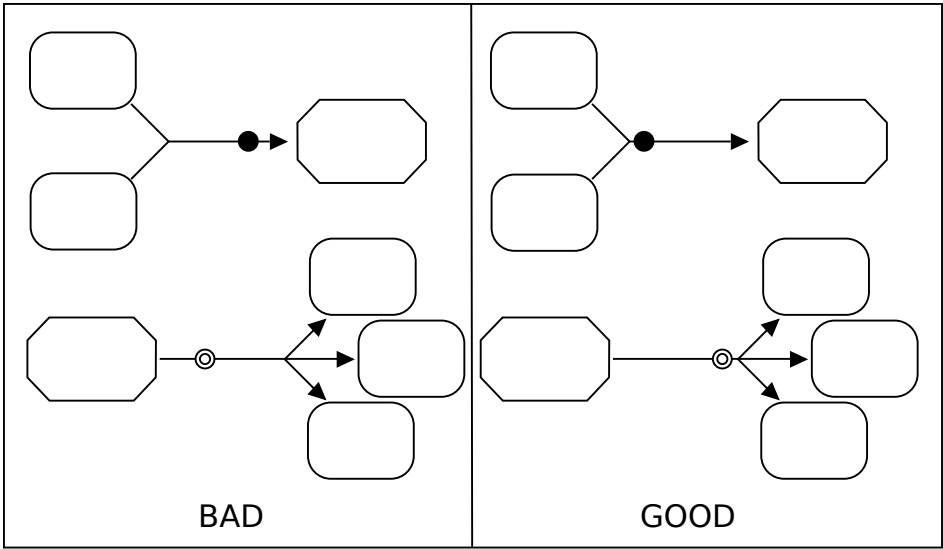
Figure 5.7: *Branching points should be close to association and dissociation symbols.*

5.3.5 Units of information 1158

Units of information should not hide the structure of the corresponding node and should not overlap 1159 other elements (Figure 5.8). 1160

5.4 Additional suggestions 1161

Here is a list of additional layout suggestions which may help improve the asthetics and clarity of 1162 Process Description maps. 1163

- Angle of edge crossings: If edge crossing cannot be avoided then the edges should cross with 1164 an angle close to 90 degrees. 1165
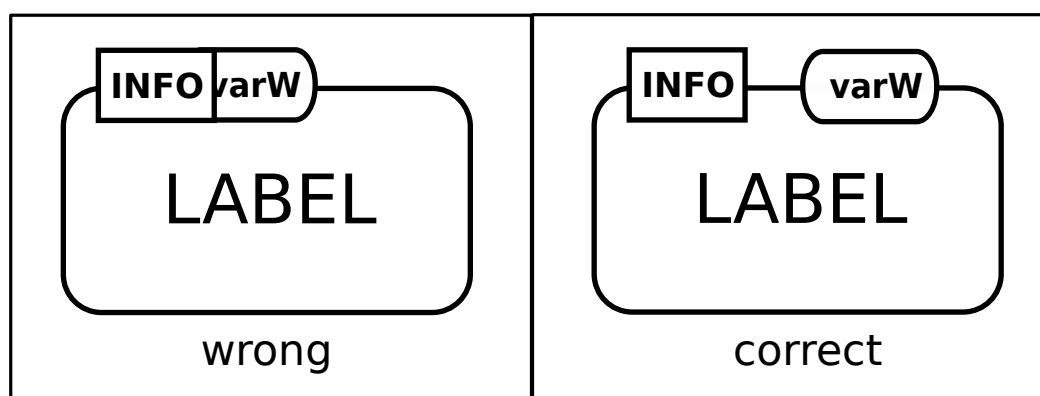
Figure 5.8: *Units of information should not overlap with any other element.*

- 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 map should be drawn in a similar way, and symmetric parts should be drawn symmetrically.

- Proximity information: Related elements (e. g. nodes connected by a process 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: It can help clarity to use a different background shade or color for each compartment.

Chapter 6

Acknowledgments

Here we acknowledge those people and organisations the assisted in the develeopment of this and 1180
previous releases of the SBGN Process Description language specification. First we specifically ac- 1181
knowledge those who contributed directly to each revision of the specification document, followed 1182
by a comprehensive acknowledgement of contributors that attended workshops and forum meet- 1183
ings or in some other way provided input to the standard. Finally, we acknowledge the bodies that 1184
provided financial support for the development of the standard. 1185

6.1 Level 1 Release 1.0

The specification of was written by Nicolas Le Novère, Stuart Moodie, Anatoly Sorokin, Michael 1187
Hucka, Falk Schreiber, Emek Demir, Huaiyu Mi, Yukiko Matsuoka, Katja Wegner and Hiroaki Ki- 1188
tano. In addition, the specification benefited much from the help of (in alphabetical order) Frank 1189
Bergmann, Sarala Dissanayake, Ralph Gauges, Peter Ghazal, Lu Li, and Steven Watterson. 1190

6.2 Level 1 Release 1.1

The specification of SBGN PD Level 1.1 was written by Stuart Moodie and Nicolas Le Novère, with 1192
contributions from (in alphabetical order) Frank Bergmann, Sarah Boyd, Emek Demir, Sarala Wimalaratne, 1193
Yukiko Matsuoka, Huaiyu Mi, Falk Schreiber, Anatoly Sorokin, Alice Villéger. 1194

6.3 Level 1 Release 1.2

The specification of SBGN PD Level 1.2 was modified by Stuart Moodie, with contributions from (in 1196
alphabetical order) Sarah Boyd, Nicolas Le Novère, Huaiyu Mi. 1197

6.4 Level 1 Release 1.3

The specification of SBGN PD Level 1.3 was modified by Stuart Moodie, with contributions from (in 1199
alphabetical order), Tobias Czauderna, Nicolas Le Novère, Anatoly Sorokin. 1200

6.5 Comprehensive list of acknowledgements

Here is a more comprehensive list of people who have been actively involved in SBGN development, 1202
either by their help designing the languages, their comments on the specification, help with devel- 1203
opment infrastructure or any other useful input. We intend this list to be complete. We are very sorry 1204
if we forgot someone, and would be grateful if you could notify us of any omission. 1205

6.6 Financial Support

Appendix A

Complete examples of Process Description Maps

The following maps present complete examples of SBGN Process Descriptions representing Biological processes. They by no mean exhaust the possibilities of SBGN Process Description Level 1.

Figure A.1 presents an example of metabolic pathway, that examplifies the use of the `EPNs simple chemical`, `macromolecule`, and `clone marker`, the `PNs process`, and the `connecting arcs consumption`, `production` and `catalysis`.
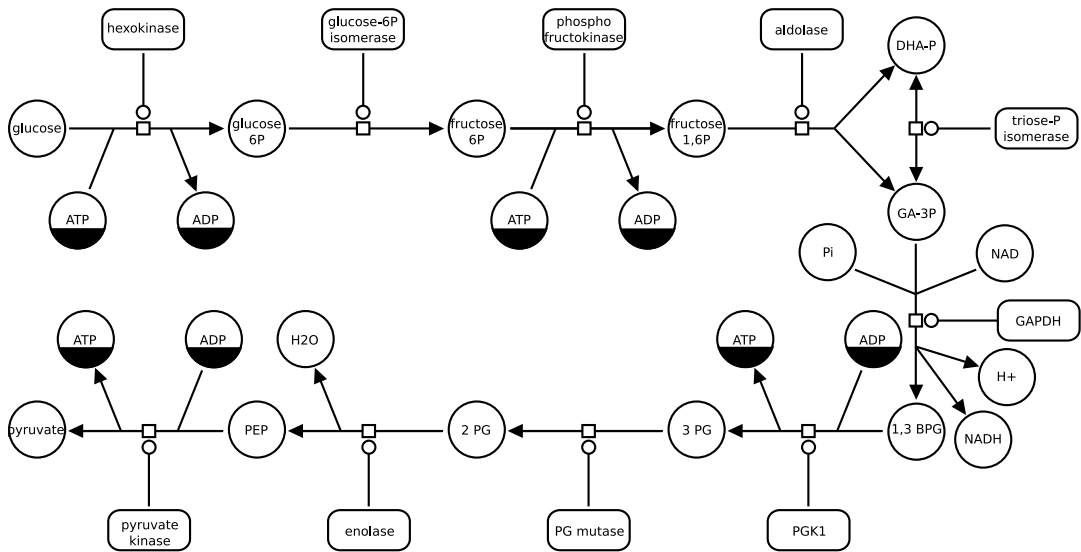
Figure A.1: *Glycolysis. This example illustrates how SBGN can be used to describe metabolic pathways.*

Figure A.2 presents an example of signalling pathway, that examplifies in addition the use of the `EPNs phenotype`, 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 Description Level 1.

Figure A.3 is an expanded version of the submap present on the map present in Figure A.2. It shows the use of `tag`.

Figure A.4 introduces an SBGN Process Description that spans several compartments. 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

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246
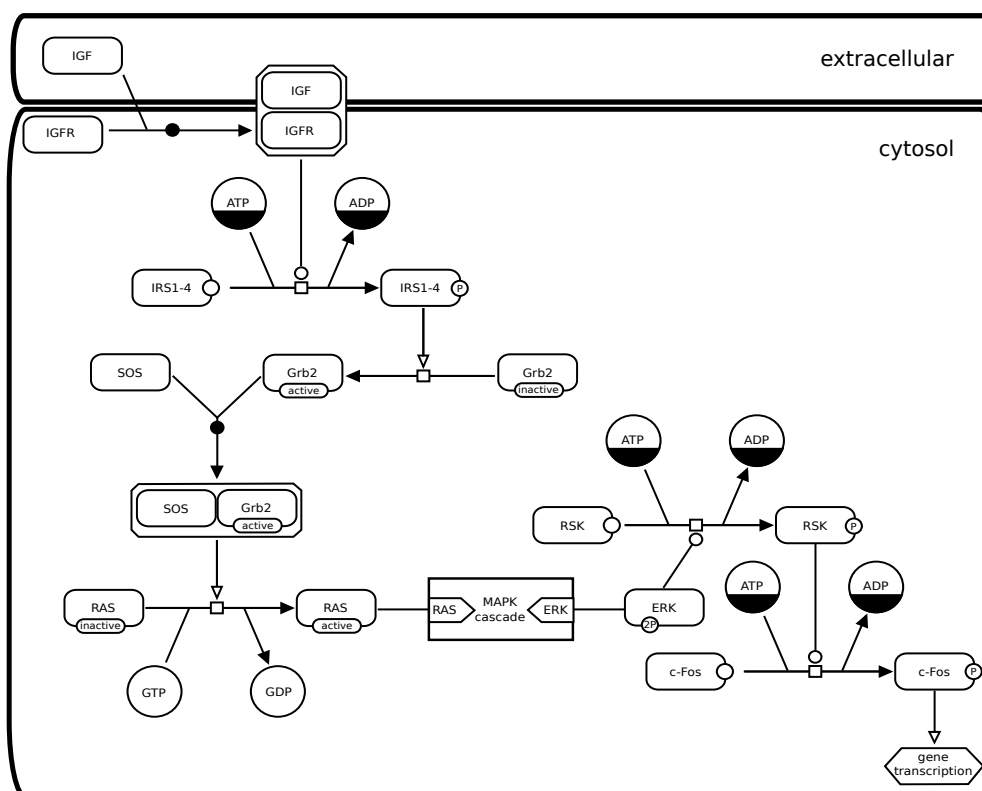
Figure A.2: *Insulin-like Growth Factor (IGF) signalling. This example shows the use of compartments and how details can be hidden by using a submap. The submap is shown on Figure A.3.*

the "synaptic button" and of "synaptic cleft". The situation is similar with the compartments "ER" and "muscle cytosol". The map exemplifies the use of the PN omitted and dissociation, and the
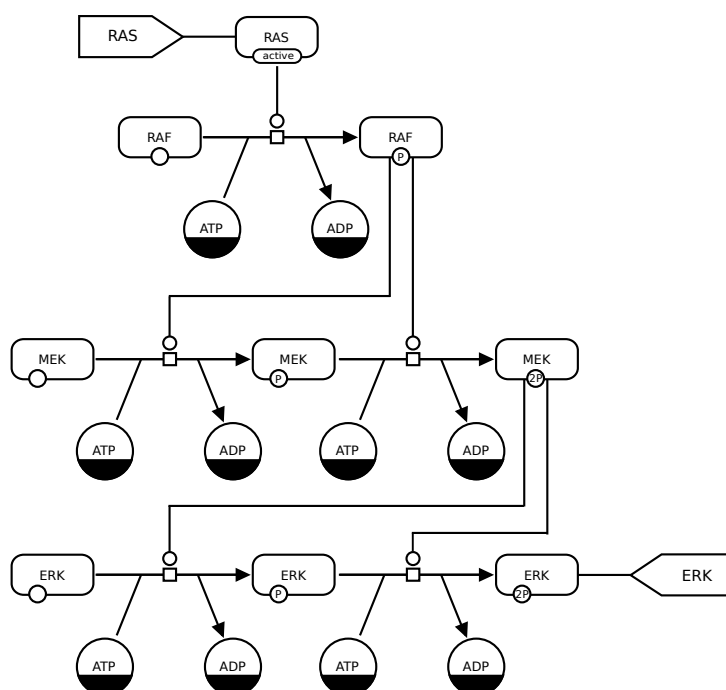
Figure A.3: *A submap of the previous map showing the MAPK cascade.*

connecting `arc` necessary stimulation.
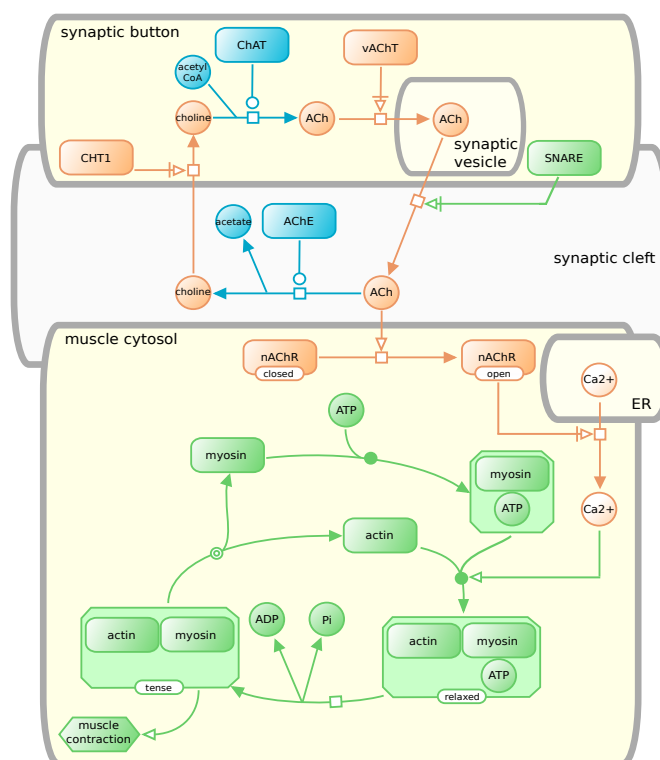
Figure A.4: *Neuronal/Muscle signalling. A description of inter-cellular signalling using SBGN.*

Figure A.5 introduces the use of SBGN Process Description Level 1 to encode gene regulatory networks. It also show the use of the `EPNs Source` and the `logical operator` and.
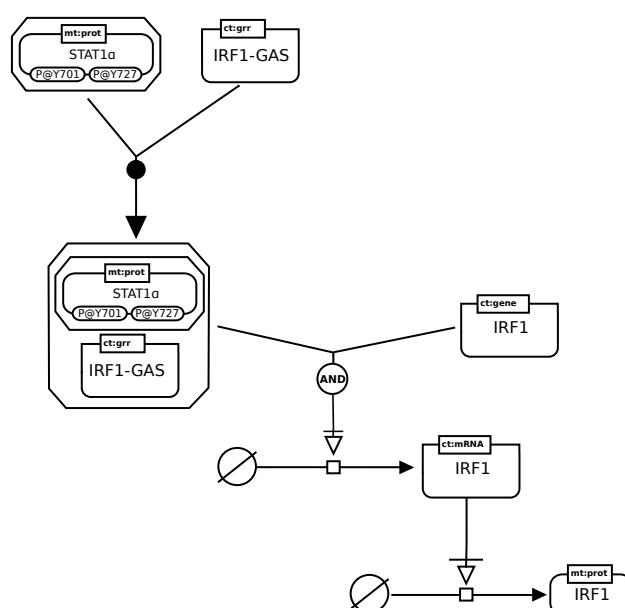
Figure A.5: *Activated STAT1α induction of the IRF1 gene. An example of gene regulation using logical operators.*

Appendix B

Reference card

Print the summary of SBGN symbols on the next page for a quick reference.

1255

Entity Pool Nodes

- unspecified entity — LABEL
- simple chemical — LABEL
- macromolecule — LABEL
- nucleic acid feature — LABEL
- perturbing agent — LABEL
- source
- sink
- complex — LABEL, INFO, varZ, varY, varX, LABEL, INFO, varW

Auxiliary Units

- unit of information — pre:label
- state variable — val@var
- clone marker — LABEL, marker, (LABEL)
- multimers — N:2 LABEL, N:5 LABEL, N:2 LABEL, N:5 LABEL

Reference Nodes

- submap — EPN or CN, LABEL, A, B, C, EPN or CN, EPN or CN
- tag — LABEL

Process Nodes

- process
- omitted process
- uncertain process
- association
- dissociation
- phenotype — LABEL

Container Nodes

- compartment — LABEL, e:INFO

Connecting arcs

- consumption — Source EPN, N, N, Target EPN
- production — Target EPN
- modulation — Target PN
- stimulation — Target PN
- catalysis — Target PN
- inhibition — Target PN
- necessary stimulation — Target PN
- logic arc — Source EPN, Logical operator
- equivalence arc — LABEL, EPN

Logical Operators

- and operator — AND
- or operator — OR
- not operator — NOT

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 Description 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 differ-
  ent molecules linked through non-covalent bonds.

- On 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 spec-
  ification.

C.2 Logical combination of state variable values

The value of a `state variable` has to be perfectly defined in SBGN Process Description 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 vari-
able "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 1286

SBGN Process Description Level 1 does not provide mechanisms to sub-class `EPNs`. There is no spe- 1287
cific means of specifying that `macromolecules` or `nucleic acid features` X1, X2 and X2 are sub- 1288
classes of X. Therefore, any process that applies to all the subtypes of X has to be triplicated. That 1289
situation can easily generate combinatorial explosions of the number of `EPNs` or `PNs`. 1290

C.5 State and transformation of compartments 1291

In SBGN Process Description Level 1 a `compartment` is a stateless entity. It cannot carry `state` 1292
`variables`, and cannot be subjected to process modifying a state. As a recult, a `compartment` cannot 1293
be transformed, moved, split or merged with another. If one want to represent the transformation of 1294
a compartment, one has to create the start and end compartments, and represent the transport of all 1295
the `EPNs` from one to the other. This is not satisfactory, and should be addressed in the future. 1296

Appendix D

Revision History

D.1 Version 1.0 to Version 1.1

Below are the changes incorporated into Version 1.1 of the SBGN Process Description Level 1 specification.

Description	Tracker ID
Regarding modulation of reversible processes, changed "should" to "must" be represented by two `process`	
Removed "The connectors and the box move as a rigid entity" in the definition of `process`	
Changed the definition of process node to "represent processes that transform one or several EPNs into *one or several EPNs, identical or different*"	
Changed SBO term of `compartment` From SBO:0000289 (functional compartment) to SBO:0000290 (physical compartment)	
Reoganised classification of glyphs	
Reoganised glyph section to reflect the above changes	
Revised reference card to reflect changes in glyph organisation	
Revised logic operators throughout spec to make sure input and output arcs meet before attaching to the glyph - as with processes.	
Added enumerated rules to grammar section. This is probably not complete, but should help the implementation of semantc validation by software tools. The hope is this will be refined as tools start validating maps.	
Updated UML maps and data dictionary to be consistent with rest of changes to spec.	
Definition of cardinality is ambiguous	2840996
`Sink and source` are lumped together	2726435
SBO terms are incorrect or missing.	2841261
`Compartment` description is confusing and contradictory.	2841122
`Clone marker` fill percentages unhelpful.	2841114
Use of CV for physical charactetistic not clear.	2841085
Definition of Cardinality is ambiguous.	2840996
input to AND on IFN example.	2804326
more SBO terms for `multimers`	2803593
Legend of figure 2.20 is incorrect	2803537
legend of figure 3.2	2802990
	continued on next page

continued from previous page	
Description	Tracker ID
Compartment colouring	2745703
Errors in diag a4.	2664912
Change name of trigger glyph.	2664908
Transition should be renamed process.	2664862
Converting arcs tautological.	2664843
Example invalid.	2545870
consumption and production.	2388317
Should require circles to be distinguishable from ellipses	2219388
Figure 2.53	2162619
Reference card: production	2104471
Figure 2.42 is wrong	2104465
Mistake in the multi-cellular example	2395488
Should not prevent processes having identical in and out	2664933
No description of linking to subunit rules.	2545810
Extensively revised the grammar section. The UML diagrams have been simplified to show glyph taxonomy, and the data dictionary has been pruned to just show glyph identity. The some syntax rules have been moved into semantics and the rules reformulated to make them easier to understand.	
Eliminated duplicate rules in layout section and revised text slightly.	
Phenotype cloning?	2989007
Perturbing agent description	2940021

1303

D.2 Version 1.1 to Version 1.2

1304

Below are the changes incorporated into Version 1.2 of the SBGN Process Description Level 1 specification.

1305
1306

Description	Tracker ID
Perturbing agent description	2940021
Members of complex touching	2849273
PD Reference card error for submap glyph	3029242

1307

D.3 Version 1.2 to Version 1.3

1308

Below are the changes incorporated into Version 1.3 of the SBGN Process Description Level 1 specification.

1309
1310

Description	Tracker ID
Incorrect editor on title page	
Typos in acknowledgements	
Fixed typo in item on catalysis in section 4.2.2.	
State variables figure 2.6 V1.2	3090543

1311

Bibliography

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

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