Additional file 1: The formal description and full list of 40 rules.

A rule is a form of reactive rules, i.e., event-condition-action rules. When the event happens, the corresponding condition is evaluated and the action is executed. Some rules are a form of condition-action rules that directly evaluate the specified conditions with no event. For such rules, if the conditions are satisfied then the action is applied. Relations used in rules are in typewriter type and the details are as follows:

- **Unary relations** are classes defined in CSO. \( \text{Process}(p_1) \) and \( \text{Connector}(c_1) \) mean that \( p_1 \) and \( c_1 \) are instances of the \textit{Process} and \textit{Connector} classes, respectively.
- **Binary relations** in all capital letters are properties defined in CSO, which map instances of a class to instances of another class. \( \text{CONNECTOR}(p_1, c_1) \) means that \( p_1 \) and \( c_1 \) are related via the \textit{CONNECTOR} property.
- **Binary relations** in a Hungarian notation, i.e. capital letters for dividing multiple words in a name except the first letter of the name, are user-defined properties by using CSO properties. The \( \text{hasInput}(p_1, e_1) \) property implies that \( p_1 \) has an input entity \( e_1 \). This kind of relations is needed to define relations among more than three classes. The details will be described later.
- **Pre-defined instances** in CSO and variables for instances are in italics. For the pre-defined instances in CSO, the apostrophe prefix is used to distinguish it from a variable, such as \'FT\_phosphorylated and \'ME\_Binding.

For the details of CSO and its schema, please refer to [16].

**Criterion 1: validation for structurally correct models**

**Rule 1.** If there is given one process and one entity, then there should be only one connector between them. Otherwise, alert.

\[
\begin{align*}
\text{Event} & \quad \text{Process}(x_1) \land \text{Entity}(x_2) \\
\text{Condition} & \quad \neg \left[ \exists x_3 \ \text{CONNECTOR}(x_1, x_3) \land \{ \text{InputProcessBiological}(x_3) \lor \text{InputAssociationBiological}(x_3) \lor \text{InputInhibitorBiological}(x_3) \lor \text{OutputProcessBiological}(x_3) \} \land \text{ENTITY}(x_3, x_2) \right] \\
\text{Action} & \quad \text{alert}
\end{align*}
\]

**Criterion 2: validation for biologically correct models**

To represent rules for Criterion 2, we first describe three properties that are not defined in CSO. As described before, we are interested in biological interactions and the related four types of connectors. We define that an input entity is the entity that is connected to a process via one of three input connectors \( \text{InputAssociationBiological}, \text{InputInhibitorBiological}, \) and \( \text{InputProcessBiological} \). In particular, the entity connected to a process via \( \text{InputProcessBiological} \) is called the input-process entity in order to distinguish it from the other two types of input entities. Lastly, the output entity is defined as the entity connected to a process via \( \text{OutputProcessBiological} \). Formally, the three properties are as follows:

\[
\begin{align*}
\exists x_1, \exists x_2, \exists x_3 & \quad [\text{hasInput}(x_1, x_3) \equiv \text{CONNECTOR}(x_1, x_2) \land \{ \text{InputProcessBiological}(x_2) \lor \text{InputAssociationBiological}(x_2) \lor \text{InputInhibitorBiological}(x_2) \} \land \text{ENTITY}(x_2, x_3)] \\
\exists x_1, \exists x_2, \exists x_3 & \quad [\text{hasInputProcess}(x_1, x_3) \equiv \text{CONNECTOR}(x_1, x_2) \land \text{InputProcessBiological}(x_2) \land \text{ENTITY}(x_2, x_3)]
\end{align*}
\]
\[ \exists x_1, \exists x_2, \exists x_3 \ [\text{hasOutput}(x_1, x_3) \equiv \text{CONNECTOR}(x_1, x_2) \land \text{OutputProcessBiological}(x_2) \land \text{ENTITY}(x_2, x_3)] \]

In the rules, sameAs defines an equality relationship between two instances or two values, whereas differentFrom defines an inequality relationship.

[Group 1: rules that need cardinality and type constraints]

**Rule 2.** It needs only one input and one output entities, but should not have any regulator entities such as input associate and input inhibitor entities.

Event \( \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Autocleavage) \)

Condition \( \neg [\exists x_2, \exists x_3 \ \text{hasInput}(x_1, x_2) \land \text{hasInputProcess}(x_1, x_3)] \)

**Rule 3.** It needs at least two inputprocess entities and one output entity whose type is Complex.

Event \( \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Binding) \)

Condition \( \neg [\exists x_2, \exists x_3 \ \text{hasInputProcess}(x_1, x_2) \land \text{Complex}(x_2) \land \text{hasOutput}(x_1, x_3)] \)

**Rule 4.** It needs at least two inputprocess entities and one output entity. One inputprocess entity should have a type as Dna and one output entity as Complex.

Event \( \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_DNABinding) \)

Condition \( \neg [\exists x_2, \exists x_3 \ \text{hasInputProcess}(x_1, x_2) \land \text{Dna}(x_2) \land \text{hasOutput}(x_1, x_3) \land \text{Complex}(x_3)] \)

**Rule 5.** It needs one input entity and one output entity as Dna.

Event \( \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_DNAReplication) \)

Condition \( \neg [\exists x_2, \exists x_3 \ \text{hasInput}(x_1, x_2) \land \text{Dna}(x_2) \land \text{hasOutput}(x_1, x_3) \land \text{Dna}(x_3)] \)

**Rule 6.** It needs only one inputprocess entity whose type is Complex and at least two output entities.

Event \( \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Dissociation) \)

Condition \( \neg [\exists x_2, \exists x_3 \ \text{hasInputProcess}(x_1, x_2) \land \text{Complex}(x_2) \land \text{hasOutput}(x_1, x_3)] \)

**Rule 7.** It needs at least one inputprocess entity whose name is GTP and one output entity whose name is GDP.

Event \( \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_GDPGTPExchange) \)

Condition \( \neg [\exists x_2, \exists x_3 \ \text{hasInputProcess}(x_1, x_2) \land \text{hasOutput}(x_1, x_3) \land \text{SmallMolecule}(x_2) \land \text{NAME}(x_2, 'GTP) \land \text{SmallMolecule}(x_3) \land \text{NAME}(x_3, 'GDP)] \)

**Rule 8.** It needs only one inputprocess and at least one output entity both of whose types are SmallMolecule.
Rule 9. It needs at least one input process and at least one output entity both of whose types are SmallMolecule.

Rule 10. It needs only one input process entity whose type is Protein or Complex.

Rule 11. It needs one input process entity whose type is Protein.

Rule 12. It needs one input process entity whose type is Protein, Complex, Sma, mRNA, or SmallMolecule.

[Group 2: cardinality and FEATURETYPE property constraints]
In the following rules, \( \text{hasFeature}(x_1, x_2) \) implies that an entity \( x_1 \) has a feature type as \( x_2 \) when \( x_2 \) is a predefined term for FeatureType in CSO and the formal definition is as follows:

\[
\exists x_1, \exists x_2, \exists x_3 \ [\text{Entity}(x_1) \land \text{hasFeature}(x_1, x_2) \equiv \text{SEQUENCEFEATURE}(x_1, x_3) \land \text{SequenceFeature}(x_3) \land \text{FEATURETYPE}(x_3, x_2) \land \text{FeatureType}(x_2)]
\]

Note that the instances with prefix (') such as \( 'ME_Autophosphorylation \) and \( 'FT_phosphorylated \) are pre-defined terms (instances) in CSO.

Rules 13–24. It needs at least one input process and one output entities both of whose uni-molecule references (XREF) are same. The output entity should have a feature type which is a pre-defined value.
Rule 25. It needs only one inputprocess and one output entities, but should not have any regulator entities such as input associate and input inhibitor entities.

\[
\text{Event} \quad \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Autophosphorylation')
\]

\[
\text{Condition} \quad \neg \exists x_2, x_3, x_4, x_5, x_6 \: \text{hasInputProcess}(x_1, x_2) \land \text{hasOutput}(x_1, x_3)
\]

\[
\land \text{Entity}(x_2) \land \text{Entity}(x_3) \land \text{FEATURETYPE}(x_1, 'FT_Autophosphorylated')
\]

\[
\land \text{UNIFICATIONXREF}(x_2, x_4) \land \text{UNIFICATIONXREF}(x_3, x_5) \land \text{sameAs}(x_4, x_5)
\]

\[
\land \text{hasInput}(x_1, x_6) \land \text{Entity}(x_6)
\]

\[
\text{Rules 26–27. It needs only one inputprocess entity and at least one output entity. The inputprocess entity should have a defined feature type. For example, a phosphorylated entity can be dephosphorylated. So the inputprocess entity has a feature type as 'FT_phosphorylated.'}
\]

\[
\text{Event} \quad \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Dephosphorylation/ME_Debiquitination')
\]

\[
\text{Condition} \quad \neg \exists x_2, x_3, x_4, x_5 \: \text{hasInputProcess}(x_1, x_2) \land \text{FEATURETYPE}(x_2, 'FT_phosphorylated/FT_ubiquitinated') \land \text{hasOutput}(x_1, x_3)
\]

\[
\land \text{UNIFICATIONXREF}(x_2, x_4) \land \text{UNIFICATIONXREF}(x_3, x_5) \land \text{sameAs}(x_4, x_5)
\]

[Group 3: cardinality and STOICHIOMETRY property constraints]

In CSO, the stoichiometric coefficient is the property of the connector connecting one process and one inputprocess entity, because the same entity can be involved in many processes and the stoichiometric coefficient will be different depending on the involved processes. In these rules, \(\text{hasStoichiometry}\) implies that given a process \(x_1\), the participating inputprocess entity \(x_3\) has \(x_7\) as its stoichiometric coefficient.

\[
\exists x_1, x_2, x_3, x_4, x_5, x_6, x_7 \: \text{hasInputProcess}(x_1, x_3) \land \text{hasStoichiometry}(x_3, x_7)
\]

\[
\equiv \text{CONNECTOR}(x_1, x_2) \land \text{InputProcessBiological}(x_2) \land \text{ENTITY}(x_2, x_3) \land \text{CONNECTORSIMULATIONPROPERTY}(x_2, x_4) \land \text{CONNECTORKINETIC}(x_4, x_5) \land \text{PARAMETER}(x_5, x_6) \land \text{KEY}(x_6, \text{stoichiometry}) \land \text{VALUE}(x_6, x_7)
\]

Rule 28. It needs only one inputprocess entity whose stoichiometry coefficient is equal to 2 and only one output entity whose type is Complex.

\[
\text{Event} \quad \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Dimerization')
\]

\[
\text{Condition} \quad \neg \exists x_2, x_3, x_4 \: \text{hasInputProcess}(x_1, x_2) \land \text{Entity}(x_2) \land \text{hasStoichiometry}(x_2, x_3) \land (x_3 = 2) \land \text{hasOutput}(x_1, x_4) \land \text{Complex}(x_4)
\]

Rule 29. It needs only one inputprocess entity whose stoichiometry coefficient is from 3 to 20 and only one output entity whose type is Complex.

\[
\text{Event} \quad \text{Process}(x_1) \land \text{BIOLOGICALEVENT}(x_1, 'ME_Oligomerization')
\]

\[
\text{Condition} \quad \neg \exists x_2, x_3, x_4 \: \text{hasInputProcess}(x_1, x_2) \land \text{Entity}(x_2) \land \text{hasStoichiometry}(x_2, x_3) \land (3 \leq x_3 \leq 20) \land \text{hasOutput}(x_1, x_4) \land \text{Complex}(x_4)
\]

Rule 30. It needs only one inputprocess entity whose stoichiometry coefficient is 21 or more and only one output entity whose type is Complex.
It needs only one input process and only one output entity. In addition, the location of two
whose location is cytoplasm.

It needs only one input process whose type is mRNA and only one output entity whose location is cytoplasm.

It needs only one input process and only one output entities both of whose types are SmallMolecule, but the location of two entities should be different.

It needs only one input process and only one output entities both of whose types are SmallMolecule, but the location of two entities should be different.
Event \( \text{Process}(x_1) \wedge \text{BIOL OGICALEVENT}(x_1, 'ME_Transcription') \)

Condition \( \neg [\exists^{=1} x_2 \text{ hasOutput}(x_1, x_2) \wedge \text{mRNA}(x_2) \wedge \text{CELLCOMPONENT}(x_2, 'CC_Nucleoplasm') ] \)

**Rule 37.** It needs only one input entity whose type is mRNA located in nucleoplasm, and only one output entity whose type is Protein located in cytoplasm.

Event \( \text{Process}(x_1) \wedge \text{BIOL OGICALEVENT}(x_1, 'ME_Translation') \)

Condition \( \neg [\exists^{=1} x_2, \exists^{=1} x_3 \text{ hasInput}(x_1, x_2) \wedge \text{Entity}(x_2) \wedge \text{CELLCOMPONENT}(x_2, 'CC_Nucleoplasm') \wedge \text{hasOutput}(x_1, x_3) \wedge \text{Protein}(x_3) \wedge \text{CELLCOMPONENT}(x_3, 'CC_Cytoplasm') ] \)

**Criterion 3: validation for systematically correct models**

In the following three rules, the action complements the given model by adding new instances (\textit{add-instance}) and properties (\textit{add-property}). The variable in braces, e.g. \( \{x_2\} \), denotes a new instance ID. Furthermore, the reverse properties are used, e.g. \( \text{ENTITY}^{-}(x_1, x_4) \) is equal to \( \text{ENTITY}(x_4, x_1) \).

**Rule 38.** \% Rule for starting entities

**Condition** \( \text{Entity}(x_1) \wedge \neg \text{Complex}(x_1) \wedge \forall x_4 \{ \text{ENTITY}^{-}(x_1, x_4) \wedge \text{Input}(x_4) \} \)

**Action**
- \textit{add-instance} Process\( \langle x_2 \rangle \), OutputProcessBiological\( \langle x_4 \rangle \)
- \textit{add-property} BIOL OGICALEVENT\( \langle x_2 \rangle, 'ME_{UnknownProduction} \),
  
  \text{CONNECTOR}\( \langle x_2 \rangle, \langle x_4 \rangle \), \text{ENTITY}\( \langle x_4 \rangle, x_1 \)

**Rule 39.** \% Rule for starting complexes

**Condition** \( \text{Complex}(x_1) \wedge \forall x_5 \{ \text{ENTITY}^{-}(x_1, x_5) \wedge \text{Input}(x_5) \} \)

**Action**
- \textit{add-instance} Process\( \langle x_2 \rangle \), OutputProcessBiological\( \langle x_4 \rangle \)
- \textit{add-property} BIOL OGICALEVENT\( \langle x_2 \rangle, 'ME_{Binding} \),
  
  \text{CONNECTOR}\( \langle x_2 \rangle, \langle x_4 \rangle \)

  for \( \forall x_3 \text{ENTITY}(x_1, x_3) \wedge \text{Entity}(x_3) \) do

  - \textit{add-property} \text{CONNECTOR}^{-}(x_3, \langle x_i \rangle)
  - \textit{add-instance} \text{InputProcessBiological}(\langle x_i \rangle)

**Rule 40.** \% Rule for degrading entities

**Condition** \( \{ \text{Protein}(x_1) \lor \text{Complex}(x_1) \lor \text{mRNA}(x_1) \lor \text{SmallMolecule}(x_1) \} \wedge \neg \{ \text{Process}(x_2) \wedge \text{BIOL OGICALEVENT}(x_2, 'ME_{UnknownDegradation} \wedge \text{hasInputProcess}(x_2, x_1) \}) \)

**Action**
- \textit{add-instance} Process\( \langle x_3 \rangle \), InputProcessBiological\( \langle x_4 \rangle \)
- \textit{add-property} BIOL OGICALEVENT\( \langle x_3 \rangle, 'ME_{UnknownDegradation} \),
  
  \text{CONNECTOR}\( \langle x_3 \rangle, \langle x_4 \rangle \), \text{ENTITY}\( \langle x_4 \rangle, x_1 \)