



Quantifying SBGN-PD

Stuart Moodie

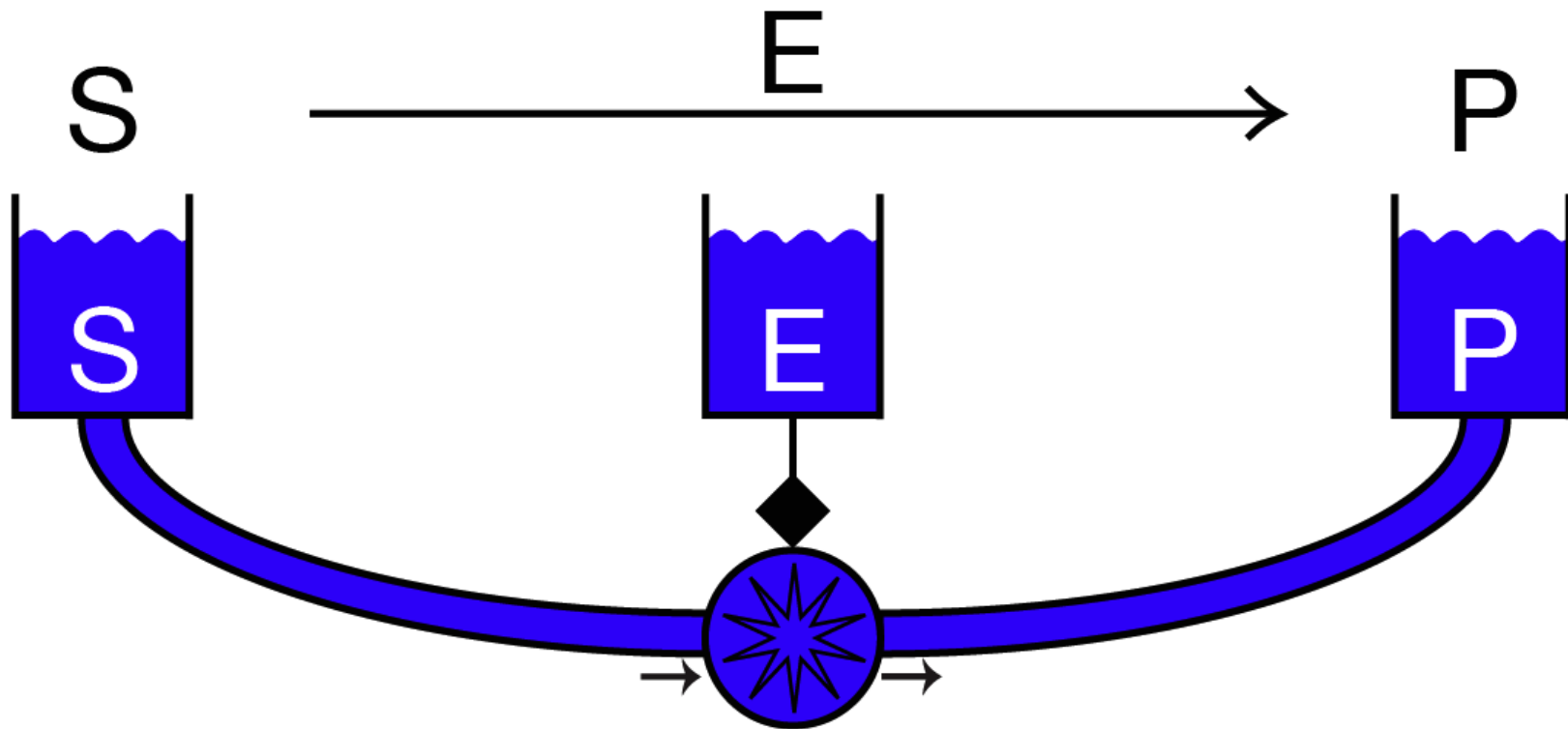
School for Informatics, University of Edinburgh

SBGN 5.5, Wittenberg 22 April 2010

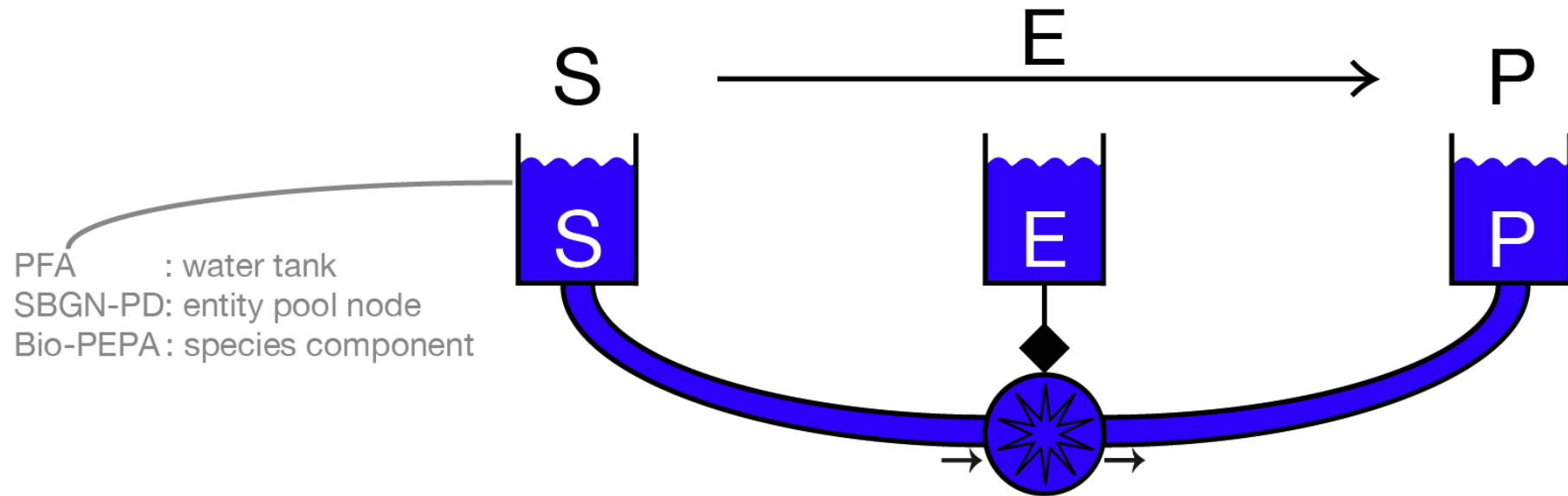
SBGN Process Diagrams are based on the Process Flow Abstraction








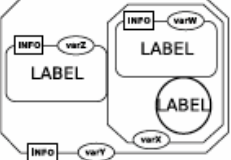


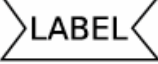
SBGN Process Diagrams are based on the Process Flow Abstraction



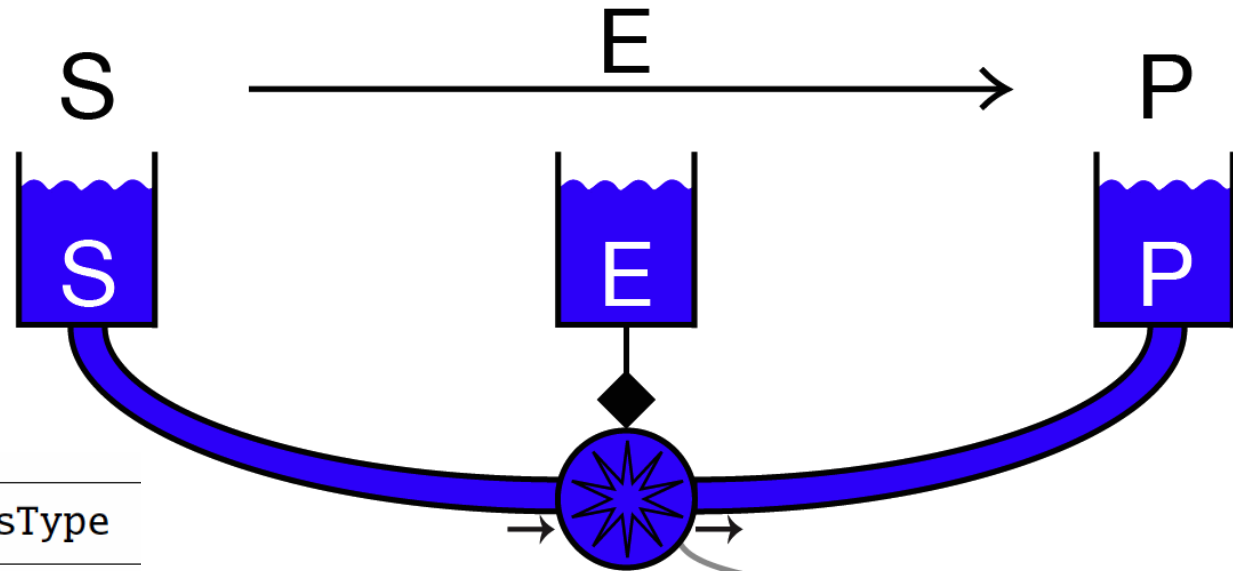
Pools of Entities are like Water Tanks



Entity Pool Nodes in SBGN-PD

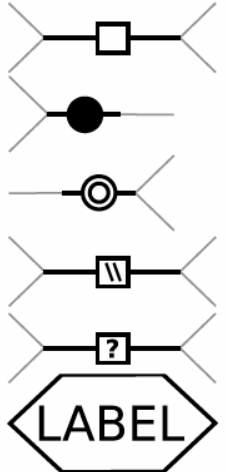
SBGN-PD glyph	EPNType	class type	comment
	Unspecified	material	EPN with unknown specifics
	SimpleChemical	material	EPN
	Macromolecule	material	EPN
	NucleicAcidFeature	material	EPN
	-	material	EPN multimer specified by cardinality
	Complex	container	EPN; arbitrary nesting allowed
	Source	conceptual	external source of molecules
	Sink	conceptual	removal from the system
	PerturbingAgent	conceptual	external influence on a reaction

Processes are like Pumps



SBGN-PD glyph

ProcessType



Process

Association

Dissociation

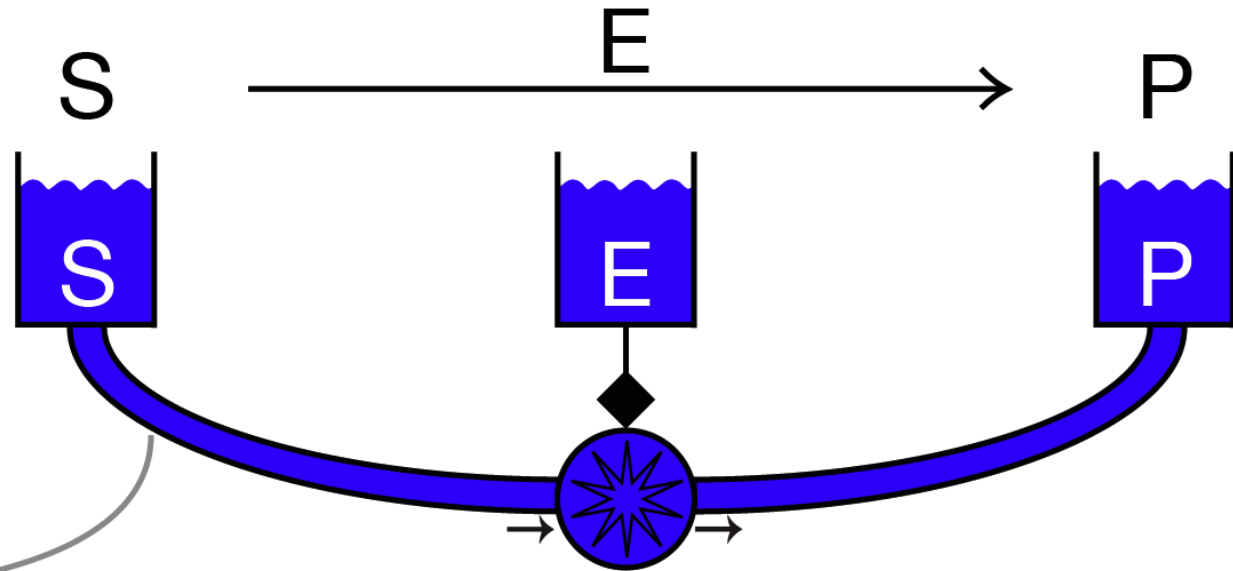
Omitted

Uncertain

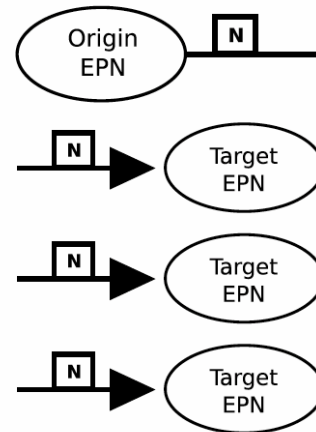
Observable

PFA : pump
 SBGN-PD: process
 Bio-PEPA: action

Consumption and Production Arcs are like Pipes

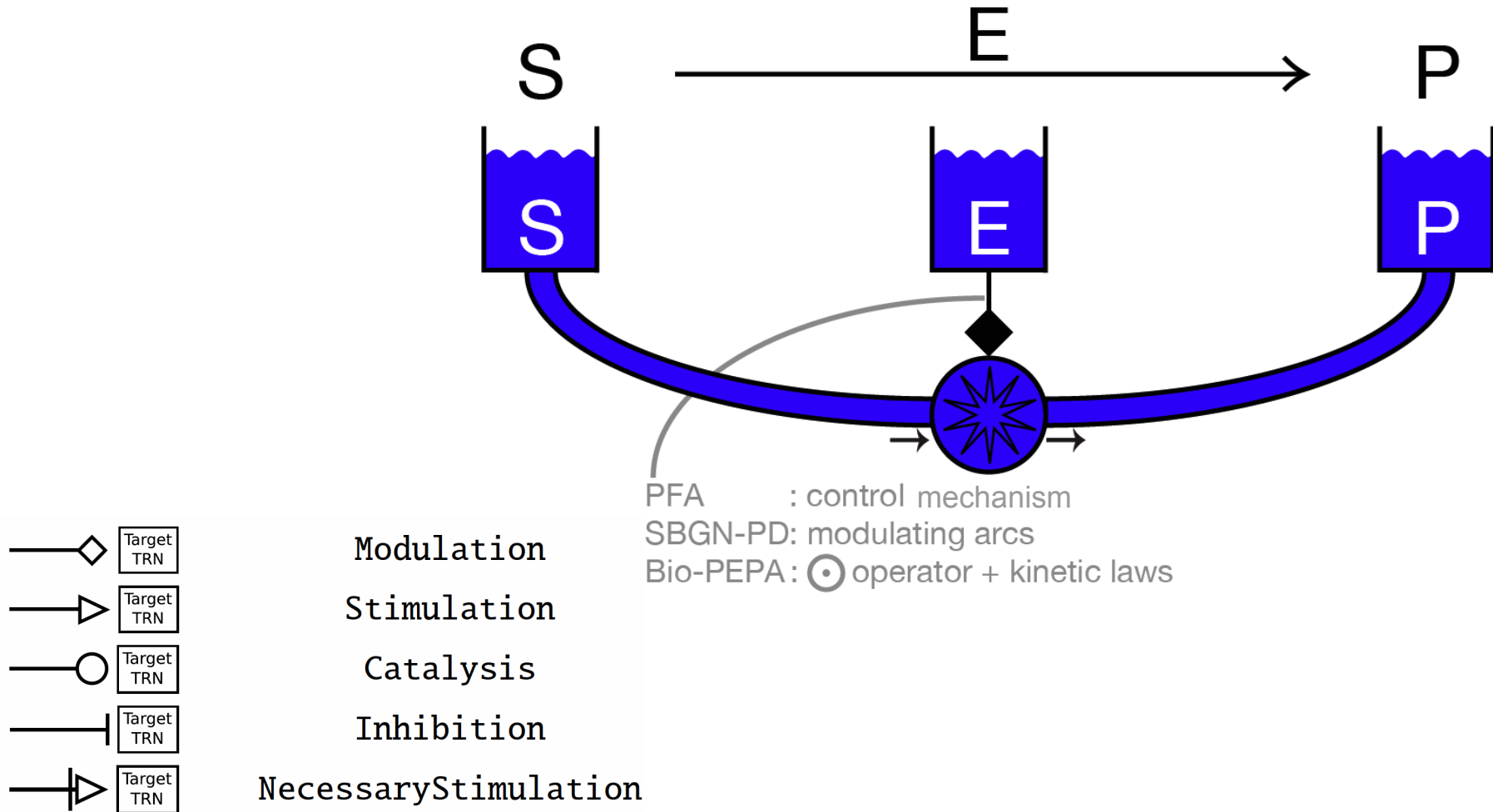


PFA : pipes
 SBGN-PD: consumption/production arcs
 Bio-PEPA: $\uparrow\downarrow$ operators



Consumption
 Production
 LeftHandSide
 RightHandSide

Modulating Arcs are like Electronics



Adding Quantitative Extensions to a Graphical Standard

1. Find a textual representation
=> ***SBGNtext*** defined as
proxy representation for SBGN-PD
2. Add extensions to the textual representation
in order to hold quantitative information

Extending Entity Pool Nodes

1. Ensure a meaningful unique name as identifier
2. Add an `InitialMoleculeCount` to provide a precise starting point for simulations

Extending Process Nodes

1. Ensure a meaningful unique name as identifier
2. Add a `ForwardPropensityFunction` to describe how this process changes the system
3. Add a `BackwardPropensityFunction` if the process is reversible

Extending Consumption and Production Arcs

1. Ensure an automatic unique name as identifier
2. They already store
 - one process
 - one entity pool node
 - the stoichiometry
 - what type of arc they are
3. Add a locally unique `ManualEquationArcID`

Extending Modulating Arcs

1. Ensure an automatic unique name as identifier
2. They already store
 - one process and one entity pool node
 - what type of arc they are
3. Add a locally unique `ManualEquationArcID`
4. Add quantitative properties $\{ (Name = Value)_n \}$

Instantiating Propensity Functions

Replace all aliases in generic propensity functions:

1. Parameters:

```
<par: ManualEquationArcID.QuantitativePropertyName>
```

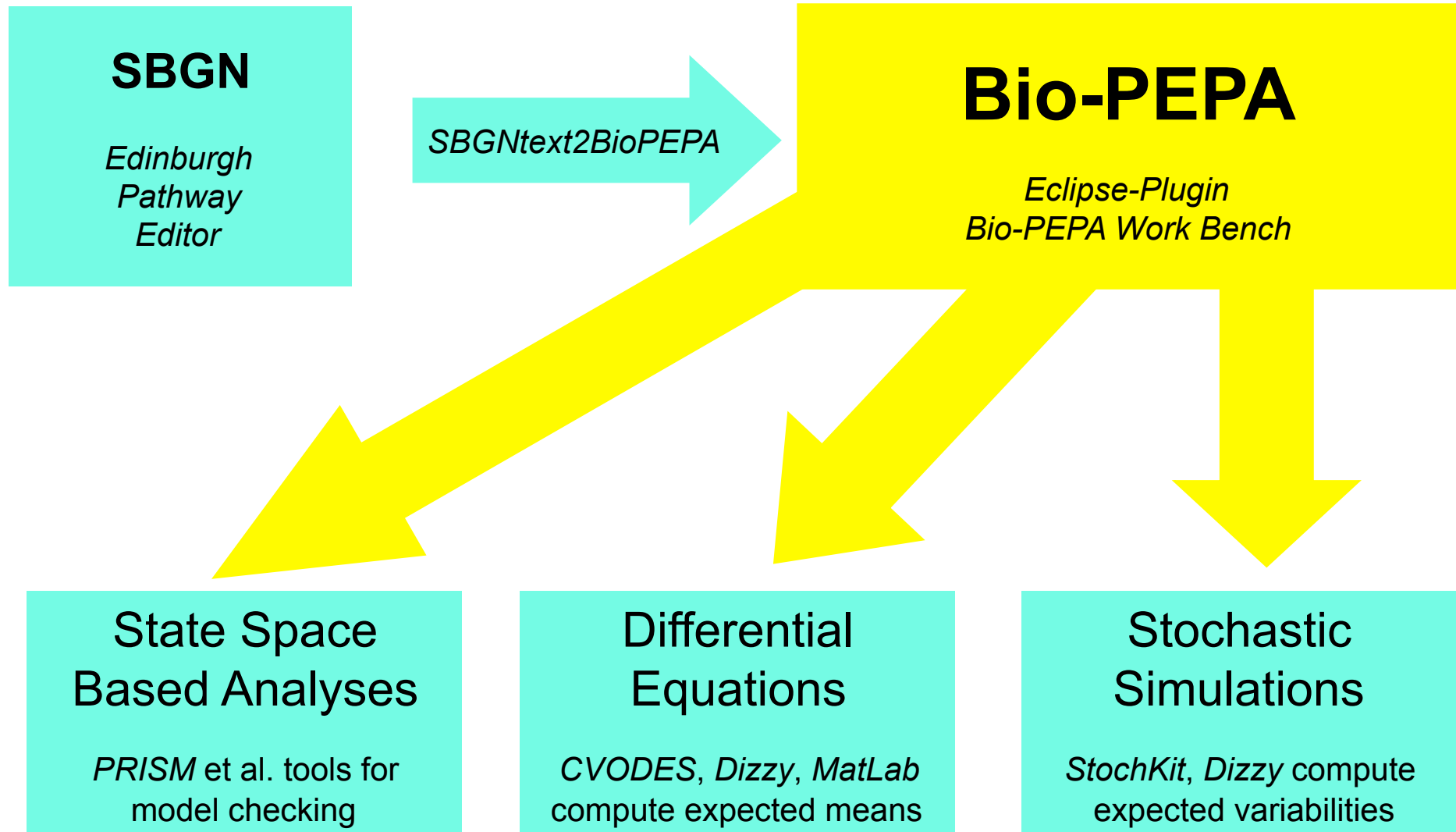
2. Entity Pool Node - molecule count:

```
<ent: ManualEquationArcID >
```

Why Bio-PEPA?

- A good example for a quantitative modelling system, but the mapping is easy to apply to other systems
- Bio-PEPA
 - is a stochastic process algebra
 - represents the model independently from solvers
 - supports a large variety of mathematical techniques
 - active development constantly expands the repertoire of analysis techniques

Modelling Workflow



Bio-PEPA Syntax Core





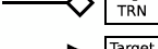




- The key is the definition of 'sequential components', where each component defines a chemical species.
- Each sequential component lists all reactions that a species can be involved in, where "+" combines different reactions

species = (reaction, stoichiometry) **OP** species + (react...

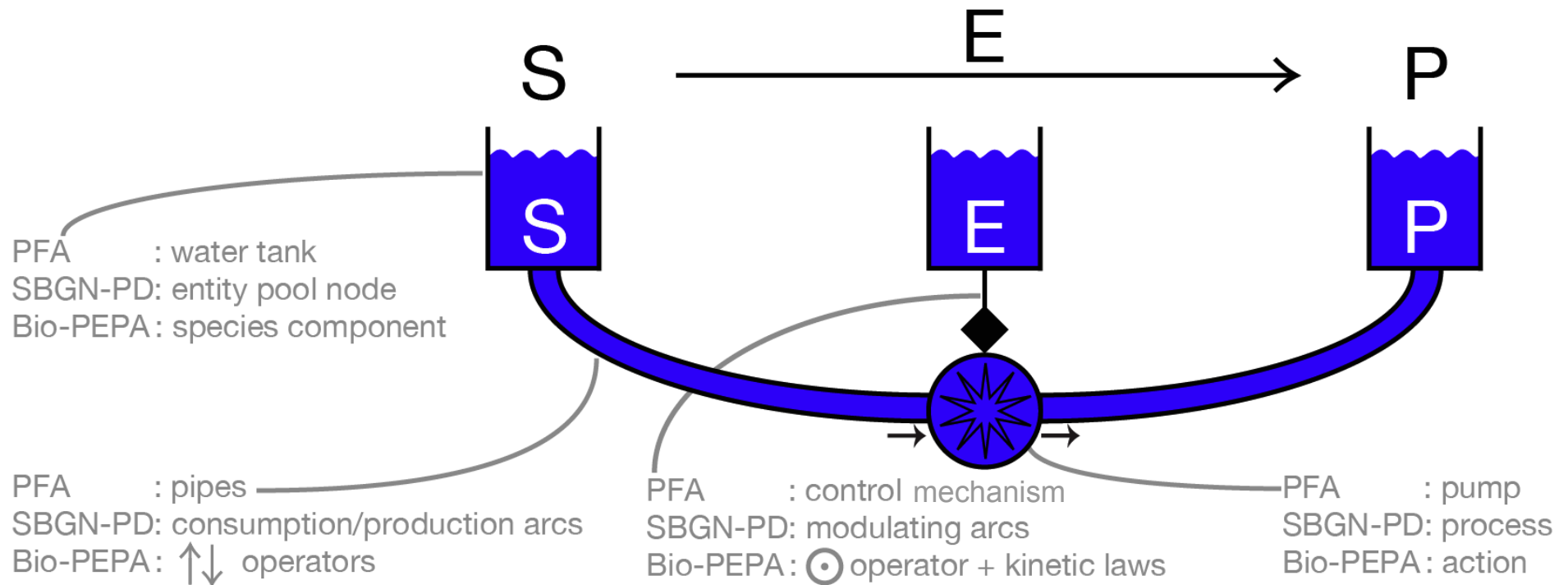
OP paper	OP code	Role of species	Meaning
↓	= <<	= reactant	= is decreased by the reaction
↑	= >>	= product	= is increased by the reaction
⊕	= (+)	= activator	= e.g. an enzyme
⊖	= (-)	= inhibitor	= e.g. an enzyme
⊙	= (.)	= modifier	= is not changed by the reaction

Mapping SBGNtext to Bio-PEPA

- Entity Pool Nodes => sequential species components
- Processes => actions
- Propensity functions => kinetic laws of actions
- Quantitative properties => parameters in kinetic laws
- Arc types => operator in sequential species component

SBGN-PD glyph	ArcType	Bio-PEPA symbol	Bio-PEPA code
	Consumption	↓	<<
	Production	↑	>>
	LeftHandSide	↓ and ↑	<< and >>
	RightHandSide	↑ and ↓	>> and <<
	Modulation	⊙	(.)
	Stimulation	⊕	(+)
	Catalysis	⊕	(+)
	Inhibition	⊖	(-)
	NecessaryStimulation	⊙	(.)

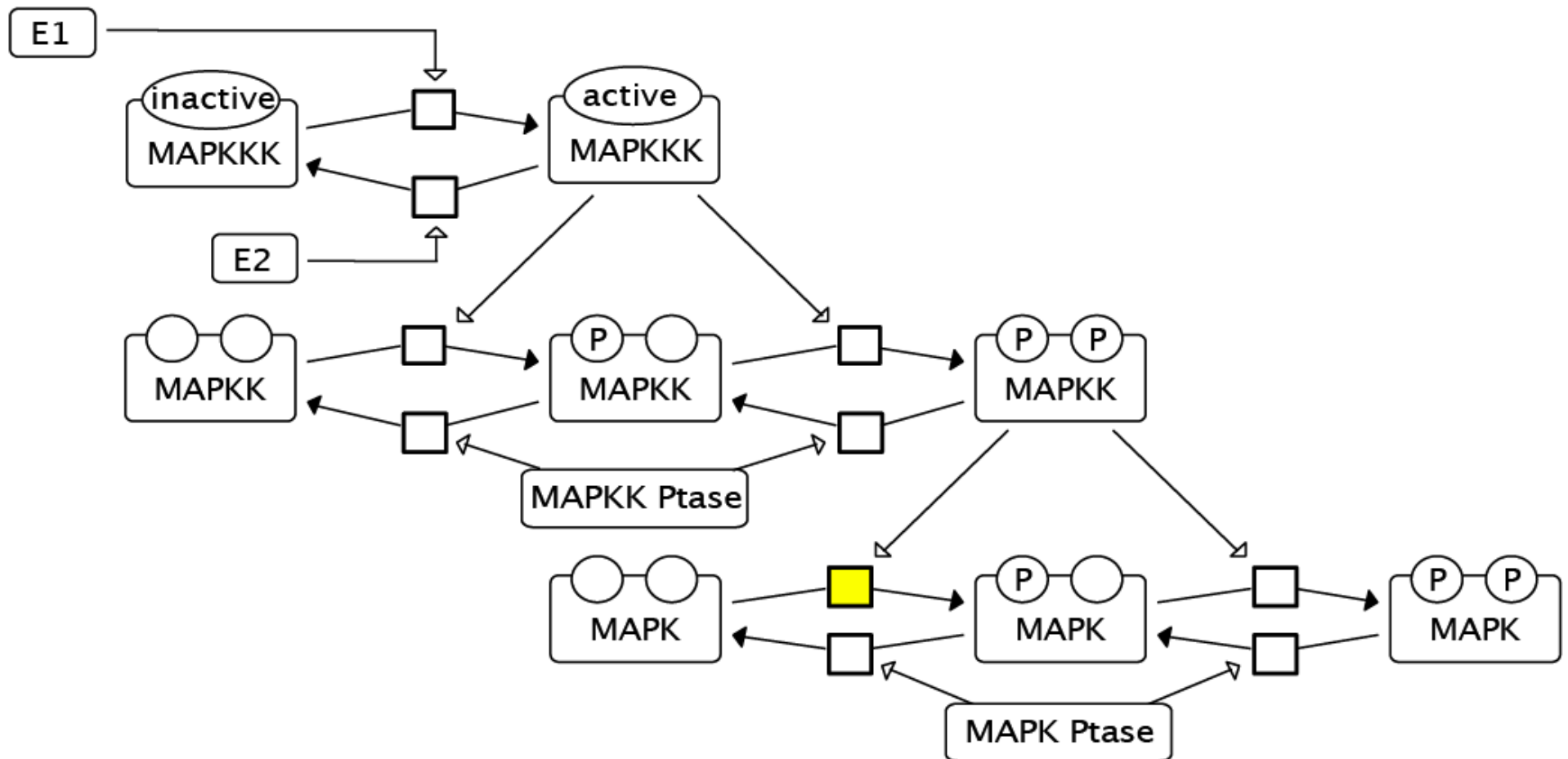
SBGNtext2BioPEPA Overview



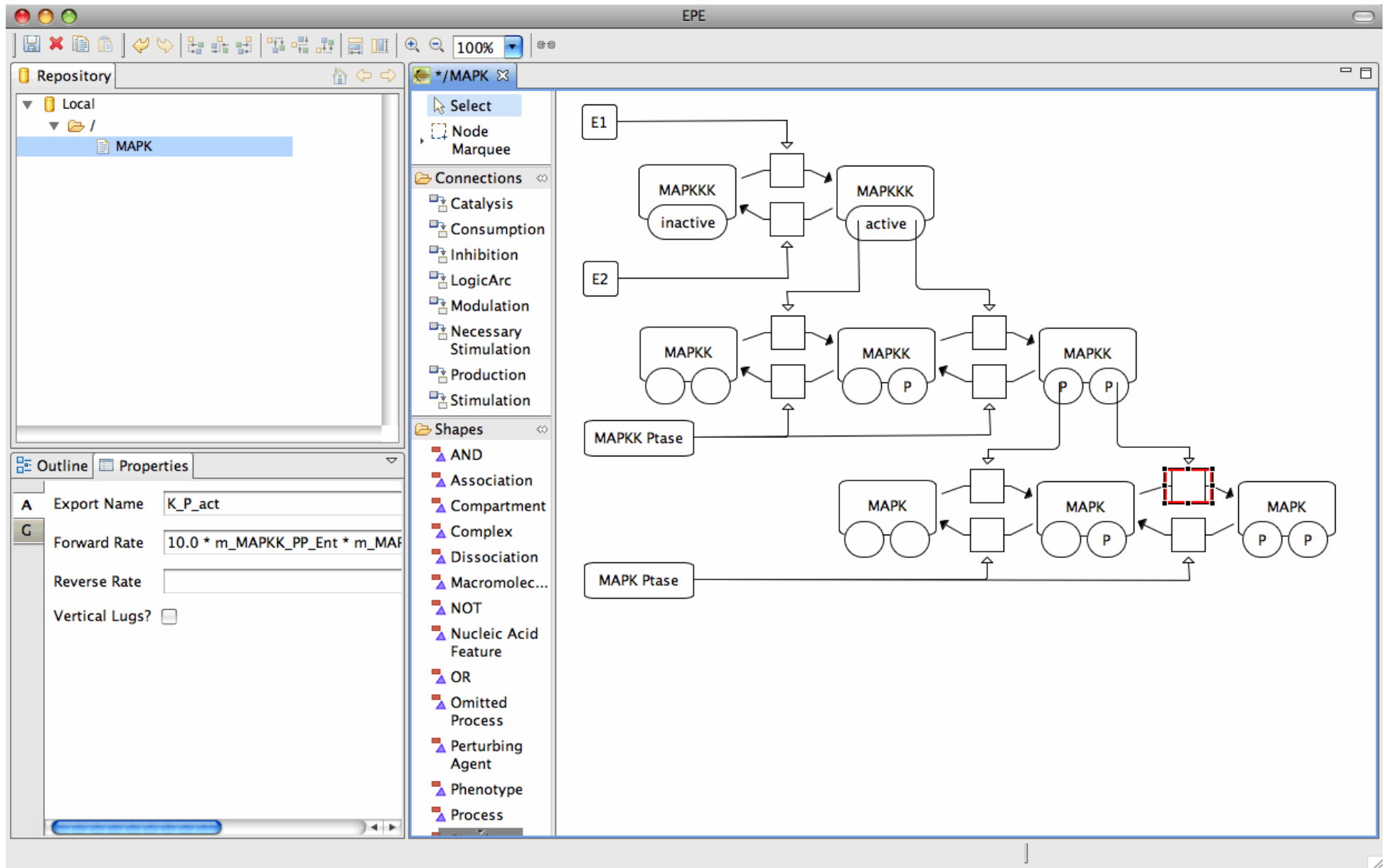
SBGNtext2BioPEPA Implementation

- Parse SBGNtext file and store all details
- Walk over treemaps of
 - entities
 - processes
 - arcs
 - parameters
- Compile the corresponding Bio-PEPA output

SBGN-PD: MAPK Signaling Cascade



Example: Edinburgh Pathway Editor



```

...// header
EntityPoolNode m_MAPKK_PP {
    EPNName = "MAPKK" ;
    EPNType = Macromolecule ;
    EPNState = { ( Label = "P" ) ( Label = "P" ) } ;
    InitialMoleculeCount = 0 ;
} ... // 11 more EPNs
ProcessNode K_P_act {
    ProcessType = Process;
    PropensityFunction =
        "<par: enz.kcat> * <ent: enz> * <ent: subs>
        / (<par: enz.Km_CountCell> * <ent: subs>)" ;
} ... // 9 more ProcessNodes
Arc st27 {
    ManualEquationArcID = enz ;
    ArcType = Stimulation ;
    Entity = m_MAPKK_PP ;
    Process = K_P_act ;
    Stoichiometry = 1 ;
    QuantitativeProperties = {
        ( Km_CountCell = 300.0 )
        ( kcat = 10.0 )
    } ;
} ... // 29 more Arcs
...// ending

```

SBGNtext

SBGNtext2BioPEPA

Java command-line tool that translates
SBGNtext => Bio-PEPA code

Freely available Parser for SBGNtext:
<http://csbe.bio.ed.ac.uk/SBGNtext2BioPEPA/index.php>

```

st27_Km_CountCell = 300;
st27_kcat = 10.0;
    ... // 18 more parameters

kineticLawOf K_P_act :
    st27_kcat *
    m_MAPKK_PP * m_MAPK_P /
    ( st27_Km_CountCell * m_MAPK_P )
; ... // 9 more kinetic laws

m_MAPKK_PP =
    ( K_act , 1 ) (+) m_MAPKK_PP +
    ( K_P_act , 1 ) (+) m_MAPKK_PP +
    ( KK_P_act , 1 ) >> m_MAPKK_PP +
    ( KK_PP_deact , 1 ) << m_MAPKK_PP
; ... // 11 more sequential species components

// model component with initial counts per cell
m_MAPKK_PP [0] <*> m_MAPK_P [0] <*>
    ... // 9 more species components [initial count]

```

Bio- PEPA

Conclusions

- The process flow abstraction gives us “quantified” SBGN-PD for free
- Mapping from PD -> model representations straightforward
- PD -> SBML possible too

Thank you!



- Laurence Loewe (Did the work & made these slides!)
- Jane Hillston

- Anatoly Sorokin
(Discussions)

Funding:

- BBSRC + EPSRC

Further reading

Websites

EPE: <http://www.pathwayeditor.org>

Tool: <http://csbe.bio.ed.ac.uk/SBGNtext2BioPEPA/index.php>

Bio-PEPA: <http://www.biopepa.org/>

Additional technical report describing SBGNtext:

Loewe, Moodie, Hillston (2009) School for Informatics, Uni Edinburgh
<http://www.inf.ed.ac.uk/publications/report/1334.html>

Published

Loewe L, Moodie S, Hillston J

Nov 2009, 2nd CompMod Workshop 2009, Eindhoven, Netherlands