Bond Graph Modeling In Simscape

Pršić Dragan, Nedić Novak, Dubonjić Ljubiša, Djordjević Vladimir

***Abstract*—**Modeling is a complex process realized through several levels of abstraction. Since each level has its own ontological primitives, a problem of model transformation from one level to another appears. In order to decrease discontinuities in a development process, this paper discusses a bond graph model library implemented in Simscape. Simscape is a software tool intended for modeling and simulation of physical systems in Simulink environment. Thanks to this library, it is possible to use physical network and bond graph approach in modeling, within the same model, on two different levels. In other words, for both structure and behavior description a unique notation is used. Besides the library of basic bond graph elements, an example of a model of a component used as interface between a bond graph and other Simscape domains is also given.

Application of Simscape bond graph library is illustrated through an example of a hydraulic system model. The model combines standard Simscape and bond graph blocks.

***Keywords*—**Bond graphs, Simscape, Object oriented modeling, Hydraulic system

I. Introduction

EVEN besides the adopted methodology, modeling is a creative process that is difficult to formalize. A lot of things depend on knowledge, experience and intuition of a modeler. Methodology provides a set of concepts, ontological primitives, which help in model description but don't say how a model is formed. Recommendations from experience, modeling heuristics on how to conduct a modeling process are usually given, however no general rule exists.

At the beginning of model forming, even informally, we need to adopt a conceptual framework in order to move easily through modeling space. Modeling is usually structured through three levels of abstraction (Fig. 1) [1], [2] where each level represents a different point of view of a system.

The modeling process is usually stared at the technical components level (TCL model). Subsystems/components and their mutual relations are identified in the system. For graphical representation of subsystems/components, more or less standard symbols that indicate their function are used.

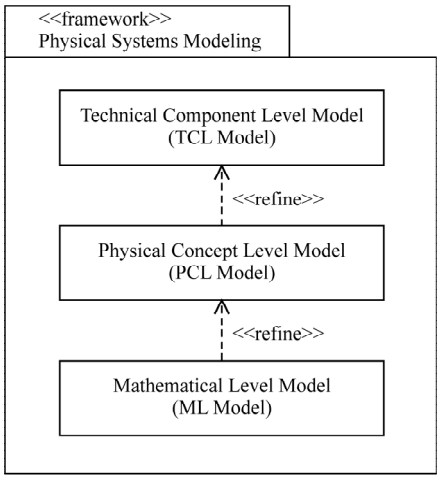


Fig. 1 Physical Systems Modeling Framework

Relations between the model elements are drawn following real relations that exist between the system components. This is how network representation of systems that describes system structure (function) is reached. Each nod from such network can be a component or a subsystem that can further be decomposed. A typical example of such a model is any hydraulic system diagram. The final model from this phase is defined by interface, as for the whole system so for each component individually, which should be realized by the models on next levels.

After completed structural decomposition, we move to describing behavior of the system, i.e. to physical concept level (PCL). Depending on purpose of the model and desired exactness, each component is attached by relevant physical processes and their mutual relations. In other words, we qualitatively describe behavior of the system. The process is not straightforward because different physical concepts can be attached to single component. The idea is to describe the system with physical terms (inertia, friction, leakage, resistance, capacitance, inductance, etc.), which are much closer to a domain expert, and to put the mathematical side of the problem into background. Here, we must take care about a contract that a component, through its interface, overtook in the previous modeling phase. This is the most important phase in modeling because it requires not only good knowledge of the system, but also ability and experience in deciding which processes to include in the model and which to leave out. For a

Manuscript submitted February 16, 2012. This work was supported by the Serbian Ministry of Education and Science under project TR 33026.

Pršić, Dragan. Department of Mechanical Engineering in Kraljevo, University of Kragujevac, Kragujevac, Serbia (phone: +381 36 383-377; fax: +381 36 383-378; e-mail: prsic.d@mfkv.kg.ac.rs).

Nedić, Novak. Department of Mechanical Engineering in Kraljevo, University of Kragujevac, Kragujevac, Serbia (e-mail: nedic.n@mfkv.kg.ac.rs).

Dubonjić, Ljubiša. Department of Mechanical Engineering in Kraljevo, University of Kragujevac, Kragujevac, Serbia (e-mail: dubonjic.lj@mfkv.kg.ac.rs).

Djordjević, Vladimir. Department of Mechanical Engineering in Kraljevo, University of Kragujevac, Kragujevac, Serbia (e-mail: vladadj@bluenet.rs).

qualitative description of behavior, we usually use graphic notation: bond graphs, generalized networks or linear graphs.

At the last level, the mathematical abstraction level (ML), physical processes and their interactions are described with mathematical relations. These relations are referred to as the constitutive relations. Not even this process is straightforward because different mathematical descriptions can be attached to the same physical mechanism. Even the form of a mathematical model can be different. For example, a model can be implemented as a set of differential and algebraic equations (DAEs) or as an executable simulation program. However, it is important that the changes on this level do not affect model changes on previous levels.

It can be seen from the aforesaid that modeling is a process comprised of series of transformations. Basic problem is a large conceptual distance of a starting and a final domain of a model. On one side, there is a user who forms an image of a real system using domain terms and notation (e.g. electrical and hydraulic diagrams), and on the other side, a strictly formalized syntax and semantics of low level (e.g. DAEs or computer programs).

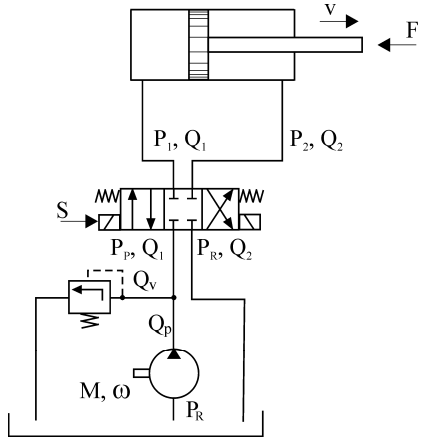
II. Simscape Approach to Modeling

In order to enable use of a unified language for modeling on all three levels of abstraction, MathWorks [3] has expanded its Simulink® package (MATLAB®, R2007a) with Simscape™ tool. Simscape is intended for modeling, simulation and analysis of multi domain physical systems. Modeling is based on generalized network concept and on generalized Kirchhoff's First Law [4]. Instead of mathematical blocks (Simulink), blocks that are appropriate for physical components (Simscape), such as pump, valve, cylinder, orifice, etc., are used for systems model construction. Connections that correspond to the energy flows and signal flows are established between the components. In such a way, the model structure suits the structure of a real system, and the model formation process itself resembles the assembling of the real system. Thus, unlike the Simulink where the system is modeled on the third level of application (ML), modeling in Simscape begins on the first level (TCL).

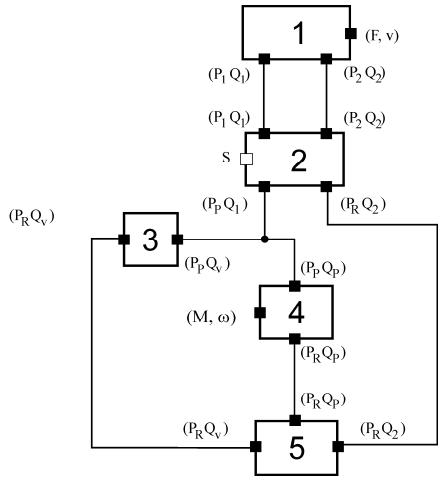
For the purpose of illustration, Fig. 2a shows a schematic view of a hydraulic system, and Fig. 2b shows a suitable TCL model based on generalized network approach.

Similarity of the system structure and the model structure (e.g. cylinder corresponding to block 1, directional valve corresponding to block 2, etc.) can be seen from the picture. That similarity is even more obvious when each block in the model is presented by a domain symbol of a given component. Therefore, model creation is an intuitive process that resembles the assembling of a real system. User chooses suitable blocks from the library and then connects them with lines following the system structure.

On a user level, each component is presented by means of its interface. This is kind of a contract a component enters into with the rest of the system. The interface is determined with a number and type of connectors through which energy and information are exchanged with the outside world.



(a) Schematic view



(b) Network model

Fig. 2 Hydraulic system

In Simscape terminology, these connectors are called physical conserving ports, i.e. physical signal ports. Ports are conceptual places through which energy/signal enters into/exits from a component. Physical conserving ports are bidirectional and represent physical connections. Physical signal ports are unidirectional and carry signals between blocks. For example, block 2 has four energy ports and one signal port (two connectors to the cylinder, one to the pump, one the reservoir and one connector for the control signal). Energy ports have their own type depending on the energy domain they belong to (electrical, hydraulic, thermal, mechanical). Each port type has specific through-across variables associated with it. For example, block 1 (cylinder) has two hydraulic (with associated P_1, Q_1 and P_2, Q_2 variables) and one mechanical port (v, F).

Each energy port is joined by two power conjugated

variables (across and through type) whose product determines the energy flow through the port. The across-through division is based on how the variables are measured. The across type quantities are observed as differences between two spatially different points. The complementary through type quantities are observed at one single point.

Connections between the components (blocks) are modeled with lines that represent physical connections transmitting energy or physical signal connections. Energetic connections are non-causal (bidirectional). We do not have to specify energy flow direction when connecting components, just as we do not have to specify this information when connecting real physical components. Physical signal links are unidirectional and are similar to Simulink signal links.

Connection lines have a role of the model's distribution mechanism. They introduce constraints in energy distribution in the system. In case of direct interconnection between two blocks, the same amount of power that leaves one block enters the other block. In other words, the link represents instantaneous energy transfer between the nodes [5]. From mathematical point of view, the interconnection forces power variable to be equal. In case when three or more components need to be connected, a physical connection line cannot branch. Then, at every moment in time, exactly the same amount of power flowing into the branch point also flows out of it. From mathematical point of view, connected components share the same across variable while the through variable divides between them.

Basic building elements in Simscape are blocks that represent engineering components such as resistor, mass, spring, valve, chamber, etc. They are organized in libraries according to technical discipline and function they perform. The blocks can be described in two ways: as a network of lower level blocks (structural model) or as basic elements implementing physical behavior by a system of mathematical equations (behavioral model). The problem is that same ontologies are used on two levels of modeling (technical component and physical concept), so a passage from one level to another is not explicit. That's why there is a potential risk to wrongly equalize the component with a physical process associated with it. For example, the block "Piston Chamber" models fluid compressibility in a chamber created by the piston in a cylinder. Most often the "Piston Chamber" component and a physical process of "fluid compressibility" coincide under real operating conditions. However, if we do not want to include fluid compressibility with a simpler model, we cannot include the "Piston Chamber" block even though it really exists. This problem is expressed even more in the electrical domain where there is even greater coincidence between the components and physical processes [1].

Our opinion is that clearer division between the concepts used on two levels of modeling needs to be made. The idea is to use the Physical Network Approach (PNA) on the level of components on one side, because the model structure corresponds with the system structure, and to use the Bond Graph Approach (BGA) on the level of physical concepts on the other side, because we pass from the structure to behavior modeling, i.e. description of energy processes in the system.

Reasons why bond graph formalism is used as a passage between a technical component and mathematical levels are the following [5], [6]:

- PNA and BGA integrate lumped parameters and energy based ontology by multiport mechanism (object oriented paradigm).
- bond graph model can be algorithmically converted into the mathematical model.
- graphical representation of physical process is uniform across different domains.

Introducing two approaches, the following effects are accomplished:

- formal division of two different views on the system, and therefore avoidance of mistakes of identification of components and physical processes,
- additional increase of model modularity because different behavior can be attached to the same structure.

The problem appearing with the introduction of two approaches into a unique modeling process is the presence of discontinuities that occur within the development process [7]. These discontinuities are caused by the lack of formal relationships between different notations. They also make it difficult to trace linkages between the system requirements and the implementations that are supposed to satisfy them. For the passage between the two approaches to modeling to be smoother, the idea is to use a unique notation in both approaches. Since the number of basic BG elements is restricted, a solution that imposes is formation of the Simscape bond graph library. Connection between the PNA and BGA, i.e. TCL and PCL, is realized on the component level - the basic structural element. The component defines the interface implemented by the bond graph.

Sections that follow show the development of certain basic bond graph blocks in Simscape.

III. 3. Bond Graph Library in Simscape

For construction of new Simscape blocks on the basic level, which do not exist in the Foundation Library (FL), Simscape™ Language is available to us. The Simscape Language (SL) is a textual, object-oriented, model description language that allows us to define behavioral model of custom blocks [8]. SL only expands the possibilities of the Simscape modeling environment, but it cannot be used as an individual language. In other words, by means of the SL a new behavioral model is formed, whereas Simscape graphical editor is still used for formation of structural models.

The new basic model is called the component model and it consists of an interface and its implementation that describes internals behavior. Towards other components, the interface is defined with number and type of bidirectional ports, nodes in

Simscape terminology, which relate power-conjugated variables in terms of physical connections. A component can contain different types of nods, but the connection can be established only between two nods of the same type. The type of a nod is defined by a domain it belongs to, i.e. type of energy flowing through it. Simscape contains in itself several predefined domains, such as hydraulic, mechanical, electrical, and so on. These domains are included in the FL and they represent the basis for construction of Simscape Foundation blocks. For example, part of a hydraulic domain description is shown in Fig. 3.

```
domain hydraulic
% Hydraulic Domain
% Copyright 2005-2008 The MathWorks, Inc.
%...

  variables
    p = {0 , 'Pa' };
  end

  variables(Balancing = true)
    q = {0 , 'm^3/s' };
  end

end
```

Fig. 3 Hydraulic domain model (partial view [3])

Central part of the domain model is represented by a declaration of variables (with initial values and units) which define the energy flow through the given domain. The block starting with a keyword variables contains a list of across variables of the given domain. The block starting with variables(Balancing=true) declares the domain's through variables. In the previous example, variables p and q are of across and through type respectively.

Although during construction of new blocks we can use predefined Simscape domains, however during development of the BG library of basic elements we must define a new domain. The reason is that a nod from one domain can be connected only to another nod from the same domain. On the other hand, BG is a multi-domain (domain-independent) approach to physical system modeling and linking to some of the domain from Simscape FL would narrow the possibility to apply the BG blocks. This can be explained with the fact that Simscape FL domains refer to the TC level of modeling, whereas the BG refers to PC level of modeling (Fig. 1). Thus, we need to define a new domain that will serve as a base for defining energy ports of all BG blocks.

Major problem during forming of BG domains originates from different conceptual interpretation of connections used in Simscape and BG. Network based connections are used in Simscape, while in BG we use port based connections. On the TC level of modeling (structure description), physical connection lines describe real connections in the system, i.e. depict physical structure of the system. During the transition to the PC level of modeling (behavior description) these connections get a new meaning because they implicitly contain a mechanism for distribution of energy. In other words, Simscape connections combine ideal connections and mechanism for energy distribution. During transition on the PC level of modeling, where the physical structure of the system loses its sense, these concepts should be divided because they represent different terms in the OO terminology. This is exactly what has been done in the BG. Edges represent perfect, point to point connections: port variables at one side of the edge are equal to port variables at the other side. Structure of interconnection is defined by means of separate blocks.

Each energy interaction can be described by means of two lower-level mechanisms that relate across and through variables of connected nodes. These mechanisms are referred to as identity and balance [6], or as equality and sum-to-zero ([9], [10]) mechanisms. The identity (equality) mechanism introduces constraint that all the variables involved must be equal. The balance (sum-to-zero) mechanism represents the implementation of generalized Kirchhoff Current Law.

Since in BG only ideal (point-to-point) connections are used, in the domain model we must only use identity mechanism. Through variables declaration blocks must stay empty. This is how we arrive to BG domain model shown in Fig. 4.

```
domain BondGraph
% Bond Graph Domain
%...

  variables
    effort = {0, '1' };
    flow = {0, '1' };
  end

end
```

Fig. 4 Bond graph domain model

As shown, interaction in the BG domain is described with two dimensionless values (effort, flow) initially set to zero. Only identity relation is established between corresponding values of two different ports.

We should notice that in the SL there is no connect statement. For example, there is no possibility to use the connect equation connect(A.p, B.p), like in Modelica [11], in order to define the connection between the two components. In other words, with the SL we can build only libraries of basic behavior models, but not structural models. Relations between the models and sub models are defined in Simscape graphic editor.

The port can only have one edge connected to. The problem is that in Simscape there is no possibility to introduce constraints that will not allow branching of connection lines (bond always connects exactly two ports). This is the reason the Simscape solver does not have the possibility to discover this syntax error, so attention should be paid during creating of bond graph models and sub models.

After defining the type of port used in the BG models, we continue with implementation of the library of basic elements in the SL. The library is based on the OO mechanism of

inheritance supported by both the SL and the BG [8], [12]. At the top of hierarchy we define the OnePortMechanism class, which is a super class for all other BG elements. The class is shown in Fig. 5.

```
component (Hidden=true) OnePortMechanism
% Generic multiport with one port

  nodes
    p1=BondGraph.BondGraph; % p1
  end

  variables
    f1 = {0, '1' };
    e1 = {0, '1' };
  end

  function setup
    across(f1,p1.flow,[]);
    across(e1,p1.effort,[]);
  end

end
```

Fig. 5 The generic class *OnePortMechanism*

Component specification begins with keyword component after which the component name is stated. Attribute Hidden practically marks that we are talking about an abstract class that cannot be instanced, but serves as a base class.

Interface components (nodes section) are defined in the first part. The interface capsules the interior of the component model and it overtakes all the interaction with the environment on itself. In the example above, connection to the outside world is realized by means of the p1 instance, whose type is defined by the BondGraph domain model (Fig. 4).

The block starting with keyword variables declares private (local) class variable which mathematically describe behavior of a given component (constitutive relations).

In the setup section we establish a relation between local variables of the given component and its interface.

Besides physical connections lines in the component interaction with the environment, signals also can participate. Signals are usually used for modulation of a parameter of the components. In such way, although the amount of power flowing through physical signal lines is negligible, influence of the signal on other energy values in constitutive relation is not negligible. In that case, physical signal port also needs to be opened in the component's interface. Therefore the ModOnePortMechanism (Fig. 6) class, as a generic class for all BG elements having a modulation port, is defined at the beginning.

```
component ModOnePortMechanism
< BondGraph.OnePortMechanism
%
inputs
  modS={1,'1'};% modS
end

end
```

Fig. 6 The generic class *ModOnePortMechanism*

Thanks to the inheritance mechanism, it is sufficient to declare only the signal port (inputs block) in the new class.

With specialization of abstract classes OnePortMechanism or ModOnePortMechanism, models of all basic BG elements with one port (modulated or non-modulated), can be defined. Fig. 7 shows a model of a non-modulated R element (resistive element).

```
component R < BondGraph.OnePortMechanism
%Dissipator

  parameters
    R={1,'1'};
  end

  equations
    e1==R*f1;
  end

end
```

Fig. 7 R element

Equations block defines the model's constitutive relation. The equations of models are described on non-causal, implicit manner. Operator "= =" is not assignment statements but specify continuous mathematical equality between expressions on left and right-hand side. Internal description does not depend on the context the model is used in because the assignment of appointing the input-output roles is overtaken by the compiler. In other words, model causality is determined after designing the system model, and not during the component model design. In such a way, the usability of the component model and modeling flexibility increase because the same interface can be realized in different ways (polymorphism) not disturbing the system model structure.

Note that the component model with one port, besides the interface, inherits also positive reference direction (PRD) pointing inward.

For modeling of energy storage in the system, available are two types of energy stores. The C storage element with one port is shown below (Fig. 8). It is a dynamic element that inserts phase shift between the input and the output signal. If the element has integral causality, phase lag appears and initial condition needs to be defined. In case of differential causality, there is a phase lead and there is no need to define starting conditions. In the SL, integral causality is preferred and there is no possibility to insert causality restrictions. In the equations section, constitutive relations are stated in a declarative way

(in mathematical sense), and the compiler itself determined their causality.

```
component C < BondGraph.OnePortMechanism
%C Storage

  parameters
    C={1,'s'};%parameter C
    x0={0,'1'}; %initial value for state
  end

  variables
    state={0,'s'}; % conserved quantity
  end

  function setup
    state=x0;
  end

  equations
    f1==state.der;
    e1==state/C;
  end

end
```

Fig. 8 C storage element

The I storage element is a dual form of the C storage element, so the model is similar to the previous one except that power variables e and f have replaced roles.

During modeling of basic BG elements with two ports (TF and GY), we start from a generic class shown in Fig. 9.

```
component  (Hidden=true) TwoPortMechanism
< BondGraph.OnePortMechanism
% Generic class for TwoPortMechanism

  nodes
    p2=BondGraph.BondGraph; % p2:right
  end

  variables
    e2 = {0 , '1' };
    f2 = {0 , '1' };
  end

  function setup
    across(e2,p2.effort,[]);
    across(f2,p2.flow,[]);
  end

end
```

Fig. 9 The generic class TwoPortMechanism

The TwoPortMechanism component inherits the OnePortMechanism class and adds a new energy port p2 which is associated with power conjugated variables e2 and f2 by means of the across( ) function.

On the basis of this generic class we reach a two-port transformer model (Fig. 10).

```
component TF < BondGraph.TwoPortMechanism
%Transformer

  parameters
    m={1,'1'};%e1=m*e2; m*f1=f2
  end

  equations
    e1==m*e2;
    m*f1==f2;
  end

end
```

Fig. 10 Two-Port Transformer

This kind of process neither stores nor produces energy, it is power conservative. In every moment, energy flow on one port is equal to energy flow on another port. No power should be lost or created by this mechanism. This implicitly means that one port is the input port, in relation power direction, and the other one is output port. However, constitutive relation form does not depend on which port is input and which is output. In other words, we don't care about positive direction of energy flow when we connect this mechanism with others. Ports p1 and p2 can change places because positive direction of energy flow is inherited from the environment. Yet, value of the m parameter depends on the connecting manner.

Model of the second type of reversible energy transformer with two ports (GY) can be acquired in a similar manner as the previous model. For two-port mechanisms with modulation port, a generic class that inherits the TwoPortMechanism class and introduce signal port are used.

In order to describe the energy exchange between the basic passive and active elements in the model, we use power conservative distribution mechanisms. In the BG, the interconnections structure is separate model part [5]. There are only two types of dual version distribution mechanisms needed: so-called 0-junction and 1-junction.

Each junction structure element is multiport with three or more ports. Since each such multiport can be presented as a set of three port mechanisms (Kirchhoff junction structure), in this paper, we only present models of distribution mechanisms with three ports.

The specification of a 0-junction in Simscape Language is given in Fig. 11.

J_0 component inherits the abstract class ThreePortMechanism, which is acquired by introduction of a third energy port. Constitutive relation of distribution mechanisms in conducted in accordance with the principle of power conservation. Exactly the same amount of power that flows into the distribution mechanism also flows out of it. Apart from that, with both types of mechanism, one of the conjugate quantities sum up to zero taking into account their signs.

```
component J_0 < BondGraph.ThreePortMechanism
%0 Junction

  parameters
    DirectionP1={1,'1'};%Power direction P1
    DirectionP2={1,'1'};%Power direction P2
    DirectionP3={1,'1'};%Power direction P3
  end

  equations
    e1==e2;
    e2==e3;

DirectionP1*f1+DirectionP2*f2+DirectionP3*f3==0;
  end

end
```

Fig. 11 0-junction with three ports

This means that, on the mathematical level, positive reference direction of the energy flow must be added to each nod. This direction does not represent actual direction of energy flow, but time-invariant reference direction in relation to which energy flow has positive or negative value.

The following convention is used in the SL: energy flow through a given nod is positive if the component energy potential increases, i.e. if the element consumes energy. In other words, the component through variable is positive if it "flows" in the direction of decrease of the component across variable.

Same rule is used in this paper, but the realization manner is different. Namely, there are two restrictions:

- in the SL, PRD is defined for the component. In the BG domain, PRD must be defined for the nod (there are one-port mechanisms).
- in the SL, the through() function is used for defining positive direction of the through variable. In the BG domain, two across variables (effort and flow) are used, so the previous function cannot be used.

In order to mark the PRD, we insert the parameter DirectionPi, i=1,2,3, one for each nod of the component, where DirectionPi=1 (default value) if power flows into the mechanism, i.e. DirectionPi=-1 if the power flows out of the mechanism.

The problem is that in the SL there is no built in logic for PDR control of distribution mechanism ports. For example, if the PRD on one end of the connection line is positive, then on the other end it must be negative. In other words, the parameter DirectionP of each distribution mechanism port connected to the passive element must have value -1.

The 1-junction is the dual form of the 0-junction. The model structure is as shown on Fig. 11 but with changed roles of energy values.

We will mention one more type of two-port mechanisms that do not belong to BG elements, but are necessary to connect the BG model with the network model on the component level. Like we mentioned earlier, the library of basic BG elements is used for modeling the internal dynamics of certain components, but for modeling of systems we use Simscape environment based on the physical network approach. Since we cannot directly connect the ports belonging to the BG domain with ports belonging to other domains (hydraulic, electrical, mechanical), we need an interface that will connect the values of different domains. Fig. 12 shows an interface that connects the BG and the hydraulic domain.

```
component  BGHydInterface
% BG-Hydraulic Interface

  nodes
    hyd=foundation.hydraulic.hydraulic; % hyd:left
    bg=BondGraph.BondGraph; % bg:right
  end

  parameters (Hidden=true)
    Kp={1,'Pa'};
    Kq={1 , 'm^3/s' };
  end

  variables
    p = {0 , 'Pa' };
    q = {0 , 'm^3/s' };
    e = {0 , '1' };
    f = {0 , '1' };
  end

  function setup
    through(q,hyd.q,[]);
    across(p,hyd.p,[]);
    across(f,bg.flow,[]);
    across(e,bg.effort,[]);
  end

  equations
    p==Kp*e;
    q==Kq*f;
  end

end
```

Fig. 12 Bond Graph - Hydraulic Interface

The interface comprises of two nods: one belonging to the BG domain and the other belonging to a domain the connection is made with. Since the nods are of different types, the interface cannot inherit the generic class of a two-port mechanism (Fig. 9). In the equations section, connection between energy values of one and the other domain is established. The connection is established with parameters (Kp, Kq) in order to secure compatibility of physical units. Similar interfaces must be used to connect BG domains with other energy domains.

IV. Application Example

Aiming to illustrate the application of Simscape bond graph library, we use model of the hydraulic circuit shown in Fig. 2. This system's Simscape model is shown on Fig. 13. All blocks, except for the Cylinder block, represent the Simscape components. For cylinder modeling, used are the BG blocks shown in this paper. Thus, two different approaches to modeling are used in the same model. On the top-level, we use generalized network approach so that the model topology could correspond to topology of a real system. On the component level, we can choose standard Simscape or bond graph blocks (generalized network or bond graph approach).

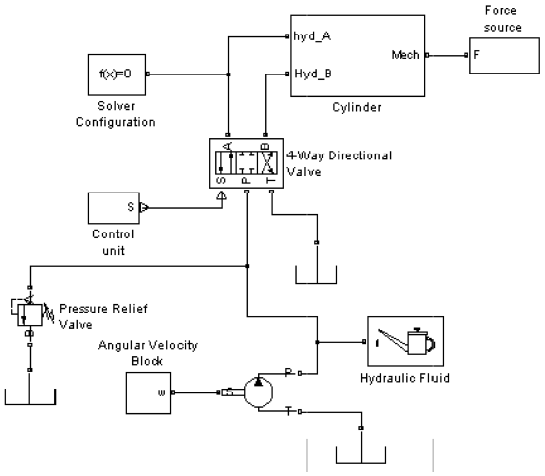


Fig. 13 Simscape model of Hydraulic Circuit

BG model of a cylinder is shown on Fig. 14, and the corresponding Simscape model, on Fig. 15. Graphic editor is used for model creation. Each BG element is presented with an icon of an interface composed of ports of the given block. Following the BG structure, we draw lines between the ports representing ideal, point to point connections.

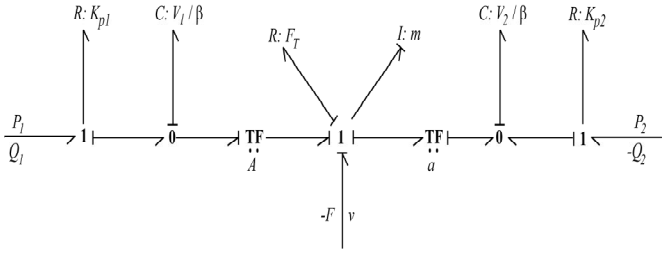


Fig. 14 Bond graph model of double acting hydraulic cylinder

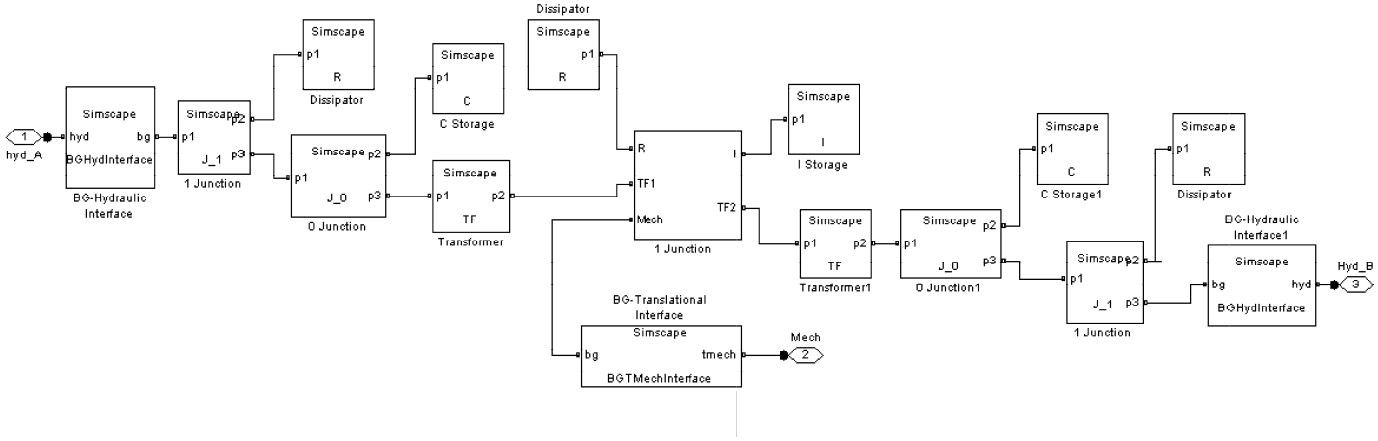


Fig. 15 Simscape model of double acting hydraulic cylinder

I. Conclusion

Our opinion is that the Simscape has a possibility of describing bond graph models using the library of basic bond graph elements. This provides us an opportunity to use two approaches to modeling within the same environment: the Physical Network Approach and the Bond Graph Approach. On the system level, we think about components and their mutual relations (structure). For each component, we define an interface in a given context. On the component level, we pass to realization of the interface. The focus is shifted from structure to behavior. We are interested in relevant physical processes and their mutual interaction.

There are two main reasons that benefit the creation of the Simscape bond graph library. Both approaches are based on a paradigm of object-oriented modeling. The system is observed as a hierarchically organized network of abstractions from the real world. For each abstraction there is division on an interface and the realization (encapsulation). The same interface can be realized on different ways (polymorphism). A new abstraction can be defined by upgrading the existing one (inheritance). Apart from that, the Simscape compiler supports the defining of constitutive relations of BG elements in a non-causal, implicit manner. Although certain conclusions about the model can be derived on the basis of causality, determining the order of calculation is subject to mathematical interpretations of physical concepts.

When using the Simscape bond graph library, attention should be paid to two restrictions: only ideal, point-to-point connections can be used between the blocks in the BG. Structure of the interconnection is defined with separate blocks. Since only the graphic editor is used in Simscape for

structure defining, this means that not any line branching connecting the blocks in the BG model is allowed. The Simscape graphic editor does not have the possibility to check this condition. The second restriction is related to the use of distribution mechanism (0, 1 - junctions) in the model. We must manually define the positive reference direction for each port of these mechanisms. There is no control of assignment of these values in Simscape.

REFERENCES

[1] A.P.J. Breunese, J.L. Top, J.F. Broenink, and J.M. Akkermans, "Libraries of reusable models: Theory and application", *Simulation* vol. 71 (1), July 1998, pp. 7-22.

[2] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, Second Edition, Addison Wesley Professional, 2005.

[3] http://www.mathworks.com/- Official site of the company The Math Works.

[4] *Simscape™3, User's Guide*, The MathWorks, 2009.

[5] W. Borutzky, "Bond Graphs, A Methodology for Modelling Multidisciplinary Dynamic Systems", SCS Publishing House, Erlangen, San Diego, 2004.

[6] U. Söderman, "Conceptual Modelling of Physical Systems - a frame of reference", Ph.D. dissertation, Linköping University, Sweden, 1995.

[7] B. Selic, G. Gullekson, and P. Ward, *Real-Time object-oriented modeling*, John Wiley & Sons, New York, 1994.

[8] *Simscape™3, Language Guide*, The MathWorks, 2009.

[9] J.F. Broenink, "Bond-graph modeling in Modelica™", 9th European Simulation Symposium, Passau Germany, 1997, pp. 137-141.

[10] J.F. Broenink, "Object-oriented modeling with bond graphs and Modelica", International Conference on Bond Graph Modeling and Simulation, the Western MultiConference (ICBGM'99), San Farancisco, Simulation Series Vol. 31, No.1, pp 163-168, (1999).

[11] Modelica Language Documents - Version 3.1, http://www.modelica.org/documents/

[12] W. Borutzky, "Bond graph modelling and simulation of multidisciplinary systems - An introduction", *Simulation Modelling Practice and Theory* vol. 17 (1), Jan. 2009, pp.3-21.