Integrating System Dynamics with Object-Role Modeling and Petri Nets

P. (Fiona) Tulinayo[1], S.J.B.A (Stijn) Hoppenbrouwers [1],
Patrick van Bommel [1], H.A. Erik Proper[1,2]

[1]Institute of Computing and Information Sciences, Radboud University Nijmegen
Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands.
[2]Capgemini Nederland B.V.,
Papendorpseweg 100, 3528 BJ Utrecht, The Netherlands.
F.Tulinayo@science.ru.nl, stijnh@cs.ru.nl, pvb@cs.ru.nl, e.proper@acm.org

Abstract: The art of System Dynamics (SD) modeling lies in discovering and representing the feedback processes and other elements that determine the dynamics of a system. However, SD shows a lack of means for discovering and expressing precise, language-based concepts in domains. Therefore, we choose to use Object-Role Modeling (ORM) to add high quality formal conceptualization to SD modeling and Petri Nets (PNs) to bridge the gap between static ORM and Dynamic, flow-like aspects of SD. To achieve the integration of these methods, we take a step by step approach where we first identify the conceptual link between SD and ORM, then the key concepts used in these methods, which we later use to derive mappings, transition statements and elements. This helps us to better understand the underlying concepts, the connections between the model structure, and behavior in a sequential manner.

1 Introduction

Integration of methods is an approach that can be applied to complex software development, it aims for combined use of different (generic) methods for highly specific purposes [Att00]. Paige [Pai97] defines method integration as an involvement in defining relationships between different methods so that they may be productively used together to solve problems. He further gives more definitions inline with method integration in [Pai99]. In this paper we integrate System Dynamics [For61] with object-role modeling (ORM) [Hal98] and Petri nets [Mur89] i.e extract different views of the same model, compare and relate them among themselves and then combine them. Our main interest is to improve SD modeling by deploying methods and techniques from system development (ORM and Petri Nets).
To model the dynamics of key variables within a process, we use SD a method that has been in existence since 1961, developed by Jay Forrester to handle socio-economic problems with a focus on the structure and behavior of systems composed of interacting feedback loops. A review and history is given in [For61]. As a method, it has its focus on the structure and behavior of systems composed of interacting feedback loops, it also provides

a high level view of the system emphasizing the interactions between its constituent parts, as well as the impact of time on its dynamic behavior [HGM01]. The dynamics arise from the interaction of two types of feedback loops, positive and negative. Positive loops tend to reinforce or amplify whatever is happening in the system while negative loops counteract and oppose change. These loops all describe processes that tend to be self limiting, create balance and equilibrium [PL99].
There have been earlier comparative studies, concerning methods potentially complementary to SD, for example SD and Discrete-event system [BH00], [BF04]; and SD and Petri nets [Dug06]. In these comparisons the main differences between SD and these methods have been highlighted but they do not state the static conceptualization of system/process which this research looks at.

For strong verbalization, conceptualization and a fully formal link to predicate logic we use ORM which is comparable to Entity relationship (ER) diagrams in use [Che76]. It is, however, a *fact-oriented* approach for modeling information at a conceptual level [HW03]. It was initially developed in the early 70's [Hal98] and has a graphical constraint notation that is claimed to be far more expressive for data modeling purposes than, for example, Unified Modeling Language (UML) class diagrams or industrial Entity Relationships (ER) diagrams [HW03]. Halpin and Wagner do state that *'although ORM supports modeling of business terms facts, and many static integrity constraints and derivation rules, it cannot model the reactive behavior of systems which can be described using dynamic integrity constraints"*. Clearly we use SD to capture the reactive behavior of a system. [Sha05] further states that; *"....there is a strong case for starting to apply systems dynamics methods more openly in the BPM and MIS research fields, as I feel the tools and techniques available are vastly under-rated in terms of their applicability and capability to provide novel representations of real-world situations....."*. These complementary statements are the basis for our overall research direction: integration of SD with Conceptual and Process Modeling.

PNs which are abstract, formal models of information flow that were originally developed to model causal relationships between asynchronous components of a computer system [Dug06] are also used. They have been around for a considerable time and are widely used for modeling workflow systems. Various scholars e.g. [Pet81]; [Mur89] give detailed studies and their background [1].
To give an insight of the overall view of the study, we came up with Fig.1 where we sketch how we are to integrate the three methods (SD, ORM and PNs). In this illustration, two types of mappings are shown: mappings between viewpoints are what we refer to as inter-viewpoint mappings, and the mappings between specific viewpoints and integrated meta model are refereed to as viewpoint meta model mappings. We use ORM, which is a state and event reporting, fact oriented modeling technique [HW03], as a graphical representation for conceptual structure. PNs are used to model a discrete flow (in a sense that they contain sequential quantities) , and SD to model a quantitative flow (items or quantifications that flow with in the system). With these three models integrated we hope

[1] Note: all methods used in this paper are in their simplest form, this is because when dealing with complex or different methods the modeler needs to understand the underlying concepts, connections and behavior in their simplest form before moving into details. As [Ric96] states "*understanding connections between model structure and behavior comes from a sequential modeling process that is from simpler formulations to complex structures*"

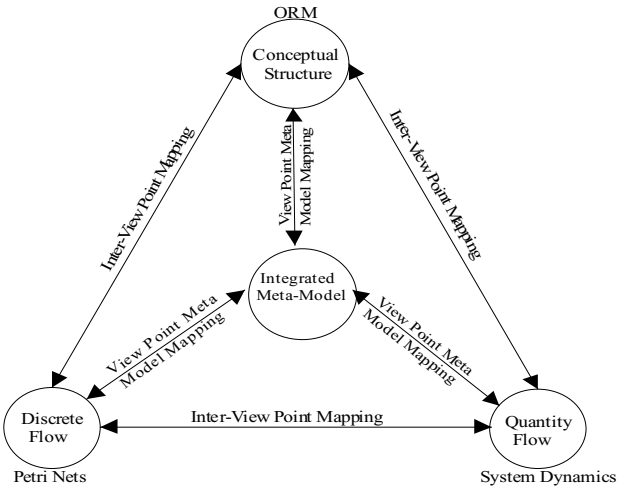

Figure 1: Abstract View of the Integration

to produce better understood models all-round, though we emphasize the use of ORM and PNs in the service of SD modeling.

The structure of this paper is as follows. In section 2 we explain the basic concepts of the three methods, then explore the conceptual links between SD and ORM; and finally SD, ORM, and PNs. We also present a step by step schema based approach for transforming an ORM domain model into an SD and PN model. In section 2.3 we use the same example to come up with a causal Loop Diagram (CLD) and Stock and Flow Diagram (SFD). In section 3 we identify the commonly used variables in the three methods, state their importance, and how they are used in the methods. We present this in two tables. In the first table we map two of the methods indicating their transitions. In the second table we add a third method (PNs), against each concept we put a transition statement to explain the similarities among these concepts. Thus this paper presents an exercise in usefully linking three existing and rather different methods in enterprize modeling.

2 Conceptual Link between SD and ORM

Before we conceptually link these methods, Lets start by explaining the basic underlying concepts of the three methods, on which we base our studies or assumptions.

The key variables we use in this paper are; System Dynamics under (A), Object Role modeling under (B), and PNs under (C) depicted in Fig.2. In SD we use *Stocks* (Stock A and Stock B), *Information links*, *Exogenous variables*, and *Flow rates* (Inflow into stock A and Outflow from Stock A into stock B). Stocks can be considered reservoirs containing quantities describing the state of the system. Flows (inflow to and outflow from the various levels) can be imagined as pipelines with a valve that controls the rate of accumulation to and from the stocks. The Exogenous variables contain information in the form of equations

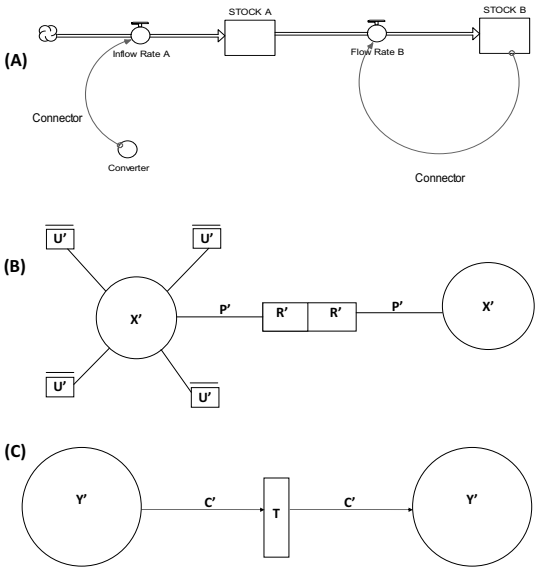

Figure 2: SD,ORM and Petri Nets Concepts

or values that can be applied to stocks, flows, and other Exogenous variables in the model [LU07].
Information links and Exogenous variables measure the quantities in levels and, through various calculations, control the rates. They appear as lines with arrows (Information links) and as circles (Exogenous variables).
In the conventions for a stock and flow diagram: (a) Information links can feed information into or out of flows and Exogenous variables but only extract information out of the stock. (b) Stocks are influenced by flows (in and out) and can influence flows or Exogenous variables but cannot be influenced by other stocks and Exogenous variables. (c) The flows can be influenced by stocks and Exogenous variables but cannot influence Exogenous variables or other flows, and Exogenous variables can influence flows or other Exogenous variables.

In ORM we use *object types* to denote entities (at type level) and *fact types* as predicate-like relationships between object types (X'). Object types are a collections of objects with similar properties, in the set-theoretical sense. Fact types (R', U') represent associations between object types, consisting of a number of roles denoting the way object types participate in that fact type. We can have, for example, *unary fact types* (U') and *binary fact types* (R').

For PNs, *Places* are denoted as (Y') which are inactive concepts analogous to in-boxes in an office workflow system. They are depicted as circles, each PN has one start place and one end place, but a number of intermediate places. *Transitions* (T) are active and represent tasks to be performed. They are depicted as rectangles in Fig.2. *Arcs* (C') are shown as connecting lines. An inward arc goes from a Place to a Transition and an outward

arc goes from a Transition to a Place. Tokens exist within Places and represent the current state of a process.

2.1 Using ORM as a Foundation for SD Models and Modeling Processes

The first step in our approach consists of three sub-steps in which an ORM diagram is augmented and replaced by process concepts that lean towards SD-like conceptualization. The second step consists of two sub-steps that concern the construction of actual CLD and SFD. We use as a working example the procedures a paper might go through en route from writing to publication (*Note that this example does not illustrate the full power of SD*). The procedures are stated as:

1. A person (author) writes an intent of submission. This can be in the form of an abstract.
2. Then the content (text of the paper) is submitted, whereby the paper becomes a submitted paper.
3. Each submitted paper receives a classification.
4. Each submitted paper is reviewed.
5. Some submitted papers are accepted and some are rejected.
6. For each accepted paper new content is submitted, which makes the paper a published paper that is added to the publications.

These statements can be represented in an ORM diagram as indicated in Fig.3:
Step 1.(a) We start with Fig.3 which is an ORM diagram of events (reported as elemen-

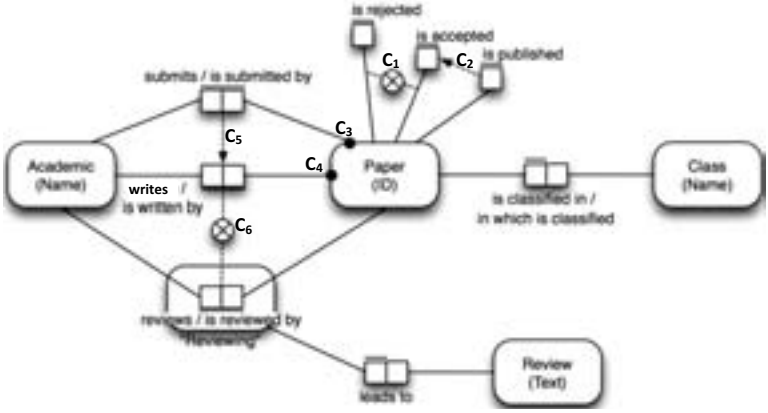

Figure 3: Paper flow concepts in ORM

tary facts) that may be observed in a domain (in this case, the reviewing domain). This approach is inline with the PSM^2 [vBFvdW97] approach. The symbol and Constraint used in fig.3 can be Verbalized as follows:

C_1 (*Exclusive*): ***each*** paper plays ***exactly one of the roles*** is rejected or is accepted.
C_2: ***each*** paper published is ***an*** accepted paper.
C_3 (*mandatory*): ***each*** paper is submitted by ***at least one*** academic.
C_4 (*mandatory*): ***each*** paper is written by ***at least one*** academic.
C_5: ***each*** Academic ***who*** submits ***a*** paper ***also*** writes ***that*** paper.
C_6 (*Exclusive*): ***each*** academic plays ***exactly one of the roles*** reviews or writes.

2.2 ORM integrated with SD and petri Nets

Step 1.(b) We add temporal dependencies between the roles associated to the paper. This leads to the flow depicted in Fig.4. The left hand side depicts the full diagram based on the facts types (event types) in the original ORM diagram. The diamond shape is the BPM [Whi04] symbol for an XOR split. Our focus is on the flow statics of submissions, reviews, acceptance, rejection and publication. This is a section of the roles connected to the object types in Fig.3. This leads to the abstracted view depicted on the right hand side.

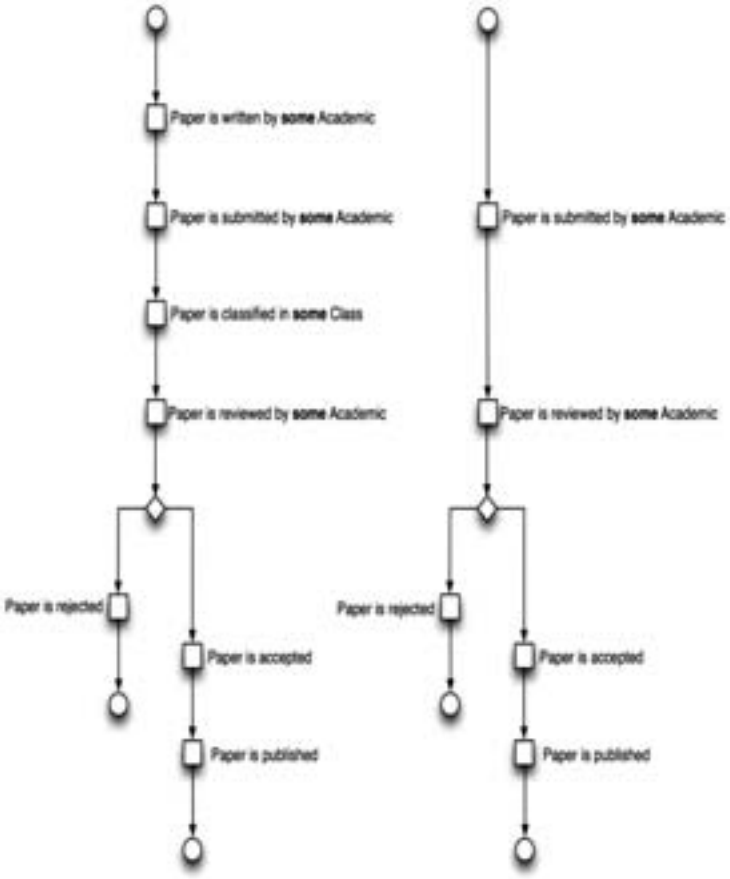

Figure 4: Paper Flow

Step 1.(c)we now make explicit the relations between, in particular, the ORM model and stock-flow model.

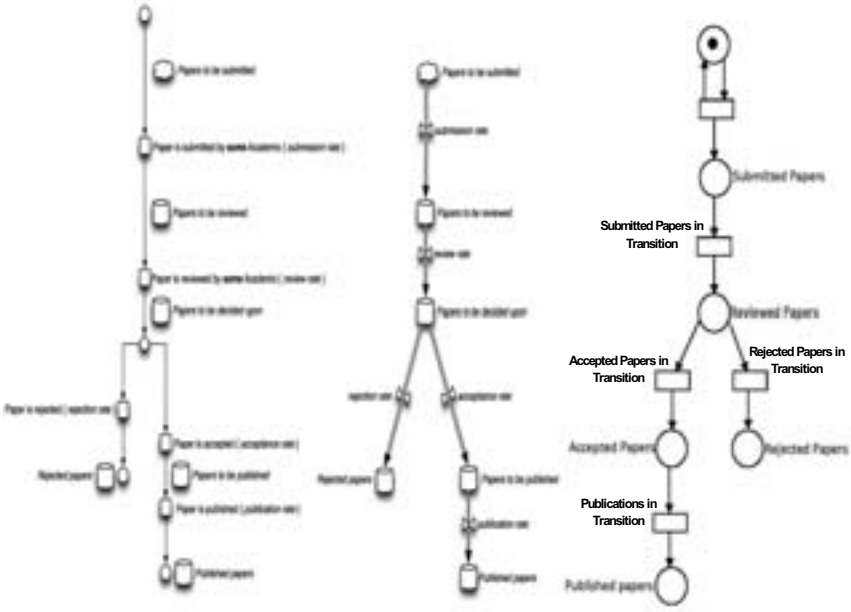

Figure 5: An integration of ORM, SD and Petri nets

In Fig.5 the left hand side shows Fig.4 (right) with some extra information and the right hand side is the Petri Net model. The extra information pertains to the flow-based interpretation. We now see stores of papers that are ready to flow from one state to another. Each time a paper "flows", this is an event (the original events as related to ORM the diagram, Fig.3). So:

- A paper is **reviewed**
- A paper is **decided upon**

Associated to the event-types, we can now also add a rate. Leading to:

- Review rate
- Acceptance rate

The model in the center depicts the SD diagram. This is the prelude to the complete SD Stock and Flow diagram as depicted in Fig.8. Note that the SD model contains five stocks, which are caused by our chosen focus on submissions, reviews, acceptance, rejection and publication explained under Step.1(b).

The PN model on the right (Fig.5) shows that every time a token (paper) if fired (submitted papers) another token is replaced. It goes through a transition (submitted papers in transition) to get to the next place (Reviewed papers) [*This holds for all places and transitions*

in this example]. Here we start to see that the transitions (PN) and Flows (SD) carryout similar jobs of transferring token (PN)/Quantities (SD) from one place (PN)/Stock (SD) to another place (PN)/Stock (SD). More of these findings are given in Tables 1 and 2.

2.3 Causal Loop Diagram, and Stock and Flow Diagram

Step 2.(a) We now embark on identifying the key variables for the SD model. In SD, two diagrams are most commonly used: causal loop diagrams (CLD) and stock-flow diagrams. A CLD (in our case, created using Vensim simulation software) helps us show the main feedback loops (feedback is defined as the transmission and return of information [RP81])in a process. Below is a clear description of a CLD or feedback loop.

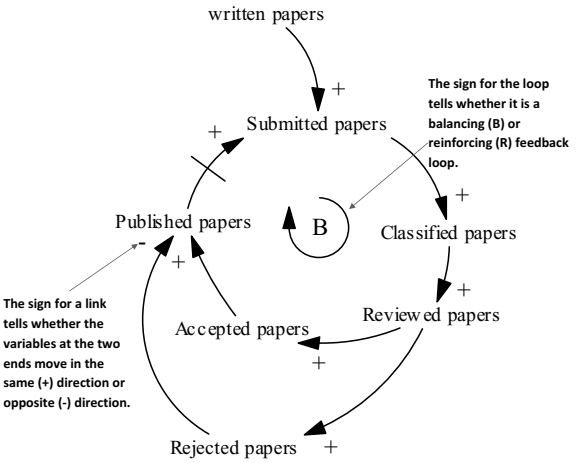

Figure 6: A causal Loop Diagram for paper flow

Figure. 6 is an annotated causal loop diagram for a simple process a paper might go through from writing to publication. This diagram includes elements and arrows (which are called *causal links*) but also includes signs (either + or -) on each link. These signs have the following meanings:

1. A causal link from one element A to another element B is *positive* (that is, +) if either (a) A adds to B or (b) a change in A produces a change in B in the *same* direction.

2. A causal link from one element A to another element B is *negative* (that is, -) if either (a) A subtracts from B or (b) a change in A produces a change in B in the *opposite* direction.

Starting from the element "*Submitted papers*" at the top of the diagram. If the Submitted papers increase then the "*Classified Papers*" also increase. Therefore, the sign on the link from "Submitted papers" to "Classified Papers" is positive. Next, if the "Classified Papers" increase, then the "*Reviewed papers*" increase. Therefore, the sign on the link

between these two elements is positive. Similarly, if the "*Reviewed papers*" increase, then the "*Accepted Papers*" increase. Hence, the sign on the link between these two elements is also positive. The Increase in " Accepted Papers " causes an increase in "*Published papers*". The next element along the chain of causal influences is the "Delay," this is because when there is an increase in "Published Papers", there is a lag (delay) before the actual increase in "submitted papers" is noticed (evident) leading to a balanced loop (B). On the other hand if the "*Reviewed papers*" increase, then the "*Rejected Papers*" increase. Therefore, the sign on the link between these two elements is also positive. The Increase in " Rejected Papers " causes a decrease in "*Published papers*". Therefore, the sign on the link from "Rejected Papers" to "Published Papers" is negative.

In addition to the signs on each link, a complete loop also is given a sign. The sign for a particular loop is determined by counting the number of minus (-) signs on all the links that make up the loop. Specifically,

1. A feedback loop is called *positive (B)*, indicated by a + sign in parentheses, if it contains an even number of negative causal links.

2. A feedback loop is called *negative (R)*, indicated by a -sign in parentheses, if it contains an odd number of negative causal links.

Thus, the sign of a loop is the algebraic product of the signs of its links. Often a small looping arrow is drawn around the feedback loop sign to more clearly indicate that the sign refers to the loop.

Step2.(b) As with a causal loop diagram, the SFD shows relationships among variables which have the potential to change over time. The SFDs have four different concepts [*Stocks, Flow-Rates, Information links (Connectors)* and *Exogenous variables (Converters)*].

The SFD is constructed here using the Powersim application. This is a simulation tool based on the SD methodology. The SFD was used to show flow dependencies and how quantities are distributed within the system. Stocks hold quantities that are subject to accumulation through inflows, or to reduction through outflows. We have the stocks as submitted papers, papers to be decided upon, rejected papers, accepted papers, and published papers. These stocks have inflows and outflows that are regulated by means of valves. The valves determine the rate at which an inflow or outflow of material applies to the stock (box). In this model there are different factors that affect the flows, and these are either positive or negative. These effects can be indicated as constants linked to the flows or stocks with a Information links. During the development of the stock and flow model a number of experimental simulations are normally run to show the different behaviors of the system studied. In our case, such simulations were also carried out, on selected simulation parameters. While doing so, the model can be paused, and each of the stocks continues to hold its quantity for observation. If the value of a particular stock is not important to the problem at hand, then it is shown as a cloud, to indicate that it is outside the boundary of the model. This procedure is also known as sensitive analysis [MR08]. From Fig.7 each rate is clearly defined as follows;
Let *X* and *Y* be the input and output of some flow $X \Rightarrow Y$:

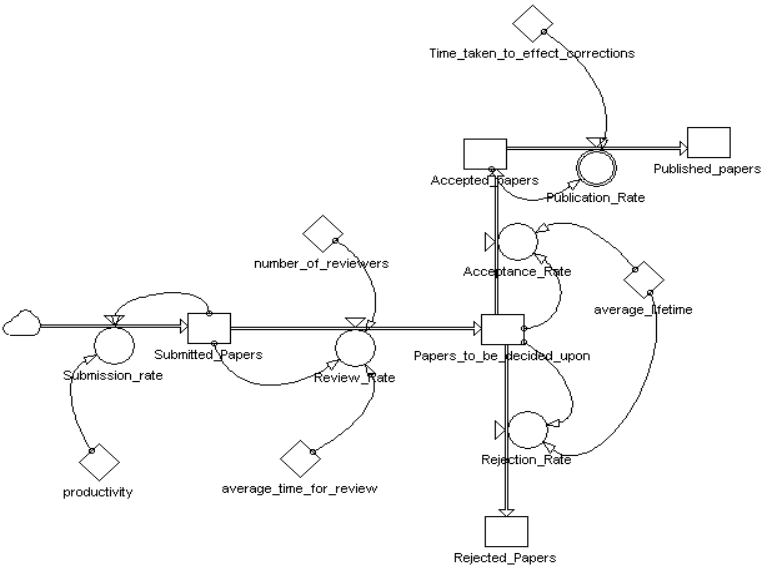

Figure 7: A stock and flow diagram for the paper Procedure

Then the rate for this flow at time *t* is defined as

$$\mathsf{Rate}(X \Rightarrow Y)\frac{\mathsf{TotalPop}_t(Y)}{\mathsf{TotalPop}_t(X)}$$

Where Totalpop $_t$(X) is the set of all instances ever of stock.

For cases where we have more outflow rates from the same stock and more inflow rates from different stocks we would have for each stock *X*:

$$\mathsf{TotalPop}_t(X)\mathsf{Pop}_t(X) \cup \bigcup_{Y:X\Rightarrow Y} \mathsf{AddedFlow}_t(X,Y)$$

$$\mathsf{AddedFlow}_t(X,Y)\mathsf{TotalPop}_t(Y) - \bigcup_{X':X'\Rightarrow Y\wedge X'\neq X} \mathsf{TotalPop}_t(X')$$

$$\mathsf{TotalPop}_t(X) \qquad \mathsf{Pop}_t(X) \cup \bigcup_{Y:X\Rightarrow Y} \left(\mathsf{TotalPop}_t(Y) - \bigcup_{X':X'\Rightarrow Y\wedge X'\neq X} \mathsf{TotalPop}_t(X') \right)$$

Note that in general this will lead to a recursive system of equation. As all instances have assigned unique locations at each moment, this recursive system of equations will have a unique location.

3 Mappings between Methods

In this we use the key variables identified in figure.2 section.2 to make explicit the relationships among these methods. This helps us in mapping the different concepts used in the methods. Apart from identifying the connections or justifiable similarities among these methods, we note their transitional statements and elements. The relationships among these methods are derived from the way concepts interact amongst themselves, or the roles they play in the process of modeling systems. Table.1 is used as starting point for the integration or combination of these techniques. We note one aspect of the methods being integrated that's SD modeling as explained earlier has dynamic properties while ORM has static properties. Static properties refer to the possible states of the system under study while dynamic properties refer to the possible transitions between the states [HEG93]. This raises an interesting question: how can we integrate a static method with a dynamic method? Our static method is ORM[2], which describes the objects that make up the states of the system as well as the integrity constraints on the states. The dynamic side of things is represented by SD [For61] and Petri nets [Pet81]. In Petri Nets, terms like Place, Transition (Link), Token and Arc are used, while in SD; Stock, Flow, information links, and exogenous variables are used. These are two completely different worlds but they both model dynamic systems.

Table 1: SD, ORM plus their transitional statements

System Dynamics	**ORM**	**Transitional Statement**
Stock	Unary fact types	They all contain "things" or act as containers.
Quantity	Objects	These can be looked at as the contents within the system.
Flows (Inflow and Outflow)	Object types	They all connect different stocks (SD) and Fact types (ORM).
Information links (connectors)	Fact types	They are both active and have movements involved that cause a change to the recipient or destination.

When mapping the different concepts used, we found connections among these concepts. *Stocks* in SD are similar (though not identical) to *unary fact types* in ORM because they both contain "*things*"(quantities for SD and Objects for ORM). *Quantity* in SD are similar to counting *Objects* in ORM. This is because we look at them as quantities that flow within the system or process. We use the term "quantity" in SD to represent the items or quantifications that flow with in the system. Next we link *flows* (inflows and out flows) in SD to *Object types*. This is because they connect different stocks (SD) and Unary fact

[2]Yet also Petri Nets which has been developed and used mainly in modeling dynamic systems and processes; they are also suitable for modeling static properties although the diagrams produced are big and complex [HEG93].

types (ORM) respectively. Finally *Information links* are linked to *fact types* because they are both active and have activities involved that cause a change to the recipient/destination. Fig.2 (A) System Dynamics, (B) Object-Role Modeling and (C) Petri nets depict the discussed variables in this paragraph.

Table 2: SD with ORM and Petri nets

System Dynamics	**ORM**	**Petri nets**	**Transitional Statement**	**Elements**
Stock	Unary fact types	Places	They all contain "things" (quantity (SD), Objects (ORM) and Tokens (PNs))	Containers
Quantity	Objects	Tokens	These can be looked at as the things that flow with in the system	Contents
Flows (Inflow and Outflow)	Object types	Transitions	They all connect and transfer items from one state to another state (*in a sequential manner*): Stocks (SD), Unary fact types, (ORM) and Places (PNs)	Homogeneous connectors
Information links (connectors)	Fact types	Arcs	They are all active and involve activities that cause a change to the recipient or destination	Heterogeneous connectors

In Table.2 *Stocks* in SD are similar (though not identical) to *unary fact types* in ORM and *Places* in PNs because they all act as containers of quantities (SD), Objects (ORM) and tokens (Petri Nets). We refer to their elements as *containers* because of their purpose which is holding items. *Quantity* in SD are similar to *counting Objects* in ORM and *Tokens* in PNs because they are the "Things/contents" that flow within the system or process and are held in stocks/places/unary fact types. we refer to their elements as *contents*. *Flows* (inflows and out flows) in SD are linked to *Object types* in ORM and *Transitions* in Petri nets because they all connect different stocks (SD), Unary fact types (ORM) and Places (Petri Nets), and transfer objects (ORM)/tokens (PN)/ Quantity (SD). We refer to their element as Homogeneous connectors because they all connect and transfer similar concepts. Finally we link *information links* (SD), fact types (ORM) and Arcs (Petri Nets) because they are all active and have activities involved that cause a change to the recipient. Although, transfers are made through them, they do not carry similar items thats why we refer to their elements are as Heterogeneous connectors.

4 Conclusion and Further Research

In this paper we have identified the key features in three modeling methods, and mapped them to show their relationships, transitions and elements. We have shown the extent to which the features of ORM static models can be transformed (with added information)

into Petri Net and SD models. The ORM methodology equips the modeler with strong conceptualization of the domain which is key in developing any model. By combining SD concepts with ORM and Petri Nets style modeling, we manage to better capture the static part of the model, and to link it with the dynamic aspect.

This research is part of larger project aiming at improving SD modeling by deploying methods and techniques from system development. An expectation is that the models produced this way are better understood with less errors than is currently the case. This will have to be empirically confirmed. With higher quality SD models in place, decision makers and stakeholders should be able to make better decisions concerning their enterprize and its processes. We will apply the approach presented in context of various case domains. We will further develop and refine the method (its models as well as the stepwise process): By devoting more attention to formalizing its syntax and semantics, but also to operationalization of the modeling procedures. In addition, we intend to use the techniques suggested in this paper in collaborative settings (Group Model Building, [RH08]), which is a sub discipline with in the field of SD. Finally, we intend to explore further links between SD and process modeling (already initiated by the Petri Net involvement).

References

[Att00] C. Attiogb'e. Formal Methods Integration for Software Development: Some Locks and Outlines. In *Technical Report: 00.8*. IRIN, University of Nantes, 2000.

[BF04] A. Borshchev and A. Filippov. From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools. *22nd International Conference of the System Dynamics Society*, 2004.

[BH00] S.C. Brailsford and N.A. Hilton. A Comparison of Discrete Event Simulation and System Dynamics for Modelling Healthcare Systems. *Proceedings of ORAHS, Glasgow Caledonian University*, pages 18 – 39, 2000.

[Che76] P.P. Chen. The entity-Relationship model-Towards a unified view data. *ACM Transactions of database systems*, 1(1):9–36, March 1976.

[Dug06] J. Duggan. A Comparison of Petri Net and System Dynamics Approaches for Modelling Dynamic Feedback Systems. *24th International Conference of the Systems Dynamics Society, Nijmegen, The Netherlands,*, July 2006.

[For61] J.W. Forrester. *Industrial Dynamics.* Productivity Press; Student Edition edition, edition, 1961.

[Hal98] T. Halpin. *'Object-Role Modeling (ORM/NIAM)', Handbook on Architectures of Information Systems.* Springer Berlin Heidelberg, 1998.

[HEG93] C. A. Heuser, Meira Peres. E, and Richter. G. Towards a complete conceptual model: Petri nets and entity-relationship diagrams. *Information Systems*, 18(5):275 – 298, July 1993.

[HGM01] J.C. Hustache, M. Gibellini, and P.L. Matos. A System Dynamics Tool for Economic Performance Assessment in Air Traffic Management. *4th USA/Europe Air Traffic Management R and D Seminar*, pages 3 – 7, December 2001.

[HW03] T Halpin and G. Wagner. Modeling reactive Behavior in ORM. *LNCS , Springer-Verlag Berlin Heidelberg*, 2813:567–569, October 2003.

[LU07] J.D Leaver and C. P. ; Unsworth. System dynamics modeling of spring behavior in the Orakeikorako geothermal field. *Elsevier Science, Oxford, ROYAUME-UNI*, 36(2):101–114, April 2007.

[MR08] B. Mutschler and M. Reichert. On Modeling and Analyzing Cost Factors in Information Systems Engineering. In *Advanced Information Systems Engineering*, pages 510–524. Springer Berlin / Heidelberg, June 2008.

[Mur89] T. Murata. Petri Nets: Properties, analysis and applications. *Proc. of the IEEE*, 77:541–580, April 1989.

[Pai97] R. Paige. A meta-method for formal method integration . *Proc. Formal Methods Europe, LNCS*, 1313:473–494, September 1997.

[Pai99] R. Paige. When are methods complementary? . *ISM*, 41(3):157–162, February 1999.

[Pet81] J.L. Peterson. *Petri Net Theory and the Modeling of System.* Prentice-Hall PTR, Upper Saddle River, NJ, USA, edition, 1981.

[PL99] D. Pfahl and K. Lebsanft. Integration of system dynamics modeling with descriptive process modeling and goal-oriented measurement. *Journal of Systems and software*, 41:135–150, April 1999.

[RH08] E. Rouwette and S.J.B.A. Hoppenbrouwers. Collaborative systems modeling and group model building: a useful combination? *26th International Conference of the System Dynamics Society*, December 2008.

[Ric96] G. Richardson. Problems for the future of system dynamics. *System Dynamics Review*, 12:141–157, December 1996.

[RP81] George P. Richardson. and Alexander L. Pugh. *Introduction to System Dynamics Modeling with DYNAMO*. MIT Cambridge, MA, USA., edition, 1981.

[Sha05] A. M. Sharif. Industrial Viewpoint can systems dynamics be effective in modeling dynamic business systems? *Business Process Management Journal*, 11(3):612–615, 2005.

[vBFvdW97] P. van Bommel, P.J.M. Frederiks, and T.P. van de Weide. Object-Oriented Modeling based on Logbooks. . *The Computer Journal*, 39(9):793–799, February 1997.

[Whi04] S.A. White. Business Process Modeling Notation (BPMN) Version 1.0. *BPMI. Org*, May 2004.