# Adding More Functionality to the Matrix Vector Transition Net for Interaction Modelling

A. Spiteri Staines

***Abstract*—** Petri nets have been used for system modelling for the past three decades. However for representing complexity at the architectural level there are certain issues. Modern systems have many complex states and connections that are not easily visible. A Matrix Vector Transition Net (MVTN) introduced in previous work is explained. This model can represent certain types of complexities in a compact form. The MVTN is based on Petri net like semantics. The MVTN is an executable structure that is more expressive for certain classes of system modelling problems. The inputs and outputs of this structure can be matrices or vectors. This work presents the addition or combination of Petri net structures to the MVTN for enhanced modelling. These can be added places or an entire net. Several toy examples from communicating systems are presented to show the enhanced MVTN. The resultant models preserve properties similar to Petri nets and are both symbolic and executable models. Results and findings are discussed.

***Keywords*—** Colored Petri Nets, Matrix-Vector Transition Net, Petri Nets, System Modelling and Interaction

## I. Introduction

MODERN computer systems have become increasingly popular in different fields in the modern world. The use of these systems varies from complex control in fields like avionics to business organizations and industry. Cloud computing and computer networking are based on grid technologies, complex interacting elements and many different types of time-spatial dependent configurations. Devising computer systems is becoming increasingly complex. Formal specification methods and notations have been around for a number of decades and they strive to deal with these issues. These can be used to prove the reliability and correctness of such structures. Petri net formalisms are types of models that have found extensive use for constructing visual and representational models of various types of systems. Naturally, formal modelling structures are important for various reasons ranging from verification and validation, checking and representation purposes.

Modern computer systems are highly dependent on organized communication and computations related to the correct specification and interconnectivities of different system elements. This principle can be extended to other types of systems and formations.

The complexities of underlying structures have a far reaching effect that is not easily visible to the untrained eye. Sometimes the underlying structures are not obvious, but global events or activities have a much more complex and deeper effect than can normally be seen. These are not just characterized by computations but also by the correctness of the specification and the interconnectedness of different modules, components or operating parts. These parts transfer information with others. Due to the complex relationships of systems, different structures of their networking and connectedness do exist. This implicates extensive dependencies on other systems.

This is seen in modern open systems like computer networking, mobile networking, social networking, grid computing, transport systems, logistics, etc [24]. E.g. if a network is considered from a global perspective, each element in the network can have a state but all the elements connected together form a composite global state. If one element changes state the global state is bound to change.

Unfortunately many formal methods including ordinary Petri nets deal with the axiomatic or lower level parts of global systems.

Normal place transition Petri nets and some other classes cannot really model the complex intricacies of communication in complex distributed systems. Still they are very useful for symbolic and structural representation in restricted form. The reasons are that when ordinary Petri nets were created, they were never intended to model at this level. A disadvantage of Petri nets is that for complex structures it is possible to derive models that have over 50 places and transitions making them very difficult to read and comprehend. Vertices, density and localization of arc connectivities will pose problems to the construction and interpretations of the net, especially if the size of the net is very large.

Previously in [13], [14] another modelling notation based on the MVTN (matrix vector transition net) approach has been suggested. This is based on ordinary Petri net like semantics but instead of using ordinary places and transitions, vectors or matrices are used and the input and output arcs are used with respective functions. This structure is useful for certain problems and systems that have multiple inputs and outputs that can be grouped or even inputs that can have real values. This work builds upon the previous work, showing how the modelling approaches used are suitable to extend for other

Anthony (Tony) Spiteri Staines, is with the Department of Information Systems, Faculty of ICT, University of Malta, (corresponding phone: 00356-21373402,e-mail: toni_staines@yahoo.com)

complex abstract representation purposes.

## II. Related Works

Petri nets are expressive formalisms with visual counterparts. They have at least three decades of coverage. They are extensively documented and well supported in literature and have numerous publications including books.

Place transition nets have been used to model different types of systems ranging from communication systems and network protocols, real time systems, distributed databases, software and hardware design, etc. Here some examples of Petri nets are briefly explained and discussed.

In [1] place transition nets explain how supervisory control can take place in distributed systems. Communication between elements is like a connection layer based circuit. Petri nets were used to create Systolic networks in [2]. These are a combination of Petri nets and other notations. They are useful for modelling interconnected processors like grid, circuit or mesh topologies and can be decomposed.

In orthogonal transformations from CPN's (colored Petri nets), matrices with sets are used to represent the networks [3]. Colored Petri nets have been extensively used for modelling complex systems like train systems, transportation and logistics. CPN's and higher order nets allow the colored place to contain complex types, possibly even sets and matrices [4]-[10].

The actor model is a higher order net structure useful for modelling information systems and distributed systems based on workflow concepts [11],[12]. The concept of a processor element, instead of a simple transition can be used to develop circuits and certain topologies. Petri nets are useful in workflow systems and have the ability to properly model the processes at varying levels of detail and formalization [2]-[4], [6]-[8].

In [22] it is shown how simple Petri nets can be translated into digraphs with some loss of information. Petri nets exhibit properties that are similar to graphs in nature. The MVTN makes use of this visual property.

UML activity diagram translation into Petri nets can be formalized using triple graph grammars (TGGs) as in [23]. The transformation is bi-directional in the sense that the starting point can be the Petri net or the activity diagram. Both are based on graph notations of nodes and edges.

In [25] a compact colored Petri net has been used for modelling fault diagnosis. This work illustrates the possibility of how higher level nets can summarize information. A net with a restricted number of places can represent a different number of error codes without the need to complicate the net. This is also possible if more error codes are required in the future or if the error codes do change. Several classes of Petri nets exist [26]. Some classes are more complex and others are simple or elementary. The simple classes are easier to handle however there is loss of information if higher order nets are transformed into simpler nets. In [27] it is shown how task graphs can be represented as Petri nets. It can be clearly noted that Petri nets are more expressive and detailed than task graphs. The transformation of task graphs into Petri nets is a simple operation.

In previous work it has been shown how the MVTN notation can successfully model complex communication between different processing elements using a Petri net oriented type of behavior [13]. An example of an abstract switch was used as a case study [14]. The reachability marking graph can be constructed for the firing of each processor/element. Instead of ordinary places the structure presented uses matrices or vectors for inputs and outputs [13]. The inputs and output arcs were assigned respective functions that must match the dimensions of the respective input and output matrices. The operations of the net can be represented using basic matrix algebra [15], [16]. Simultaneously, the structural complexity of the network is kept reduced or simplified. It is possible to apply traditional solutions or modelling methods to many system structures provided that the traditional solutions are used in new ways and preserve certain fundamental properties. Having different views and representation models of a system can offer better insight to what is happening. Thus, multiple views definitely offer better reasoning about complex systems. It is suggested that the MVTN is used in a combined approach for this purpose.

## III. Background

The major objective of this work is to devise simple solutions to represent a wide range of applications and systems. As it is impossible to meet optimally all the objectives of formal modelling using diagrammatic notations, the focus is on the reliability and correctness of system structures. System verification in reduced form can be based on i) proof of correctness, ii) proof or termination of some processing activity and iii) conditional correctness.

There is a demand for large scale nets to represent real world systems. Large scale nets must exhibit Petri net like behavior. These are: well foundedness, well formed, some form of execution, conservative behavior, formally verifiable, no loss of information, etc. The matrix vector transition net (MVTN) or the simpler called matrix transition net (MTN) has been precisely defined for these purposes [13]. In essence the MVTN is a modified Petri net that has much more detail in places and the arc connectivities.

Complexity factors between interfacing and linking components imply a difficulty to represent and model this interaction. Petri nets can help with this. However ordinary Petri nets were never intended to model at these levels. Colored Petri nets and higher order nets are far better [5]-[8]. If systems evolve it is also necessary that the models for system description evolve. Evolution in systems implies that modelling power must increase. This is required because still no proper visual notations readily available. It is possible to identify a gap of missing information between real world complex models and those that are represented using ordinary Petri nets. The solution of the MVTN is possible.

The MVTN is a higher order net that can be considered to be a type of CPN, but that offers the use of matrices and vectors [13].

Many Petri net models are unfortunately an oversimplification of the real world system and scenarios. When system representation is done using graph structures and network models, the models will not fully describe all the aspects of the system. Petri nets are appropriate means for describing the partial ordering or sequencing of actions and events. However from the viewpoint of causal ordering Petri nets are representative of the state or states of a system. The graphical and mathematical properties of these nets guarantee the preservation of partial event ordering because of precise rules and restricted actions. The MVTN will similarly preserve these properties.

Large scale Petri nets can be created, however they are not compact and easily readable [17]-[19]. Because of the limiting factors in ordinary Petri nets, alternative modelling approaches like colored Petri nets, higher order nets, graph structures, structured diagrams, etc. have all been suggested. Some classes of higher order nets are just representative structures and not really executable. A Petri net without a processing element is just a graphical description according to modern system architectural views. A Petri net has to be processed or executed for some change in state. Low level behavior cannot be easily replicated by high level structures and vice-versa. Many system structures are based on repeatable patterns and replication of certain parts.

## IV. Problem Definition

In today's world there is an ever increasing complexity in the distribution and connectivity of hardware, middleware and software. New approaches to represent these complexities are important for software engineering to be taken to the next level. This implies that new methods of system representation must be developed. One interesting area is that of symbolic modelling structures related to Petri net like behavior.

The main problem is to represent complex system structures and the connections using more detailed models that can effectively show the detailed communication and its sequence. The usefulness of Petri nets comes from its execution and the possibility to have a change in state. The challenge of the MVTN is that it has to imitate Petri net like behavior and be capable of representing the change in state when an event takes place [13].

Various reasons can be given for motivating the use of the MVTN and extending it. Matrix structures can be derived from ordinary Petri nets to show their execution. Matrices and vectors have been chosen because they can contain more information than ordinary places. Matrices can represent extensive data sets or results and the global state of a complex system. From an architectural perspective, circuit behavior is easily abstracted. In essence if a system can be viewed as a set of interconnected, related components passing messages or information, then the inputs and outputs can be grouped into matrices or vectors.

Petri nets and the MVTN are based on digraphs having simple edge and node types. The MVTN can include matrices, row and column vectors and other constructs like elementary Petri net places. The MVTN is more complex than an ordinary place transition net and the execution introduces new problems.

The model is similar in principle to higher order net structures and can represent the complexity involved in communication architectures. These structures offer improved and compacted visual representation. This is more representational of what is happening in the real world. The MVTN contains the possibility for expansion and expressiveness as regards to complexity.

The problem is to enhance even further the MVTN. The MVTN will be combined with other Petri net constructs. This work explains and shows how this is possible up to a certain level of detail.

The usage of matrices and vectors instead of ordinary places allows the possibility of creating new diverse types of models in addition to the standard types. The MVTN is not exclusive and even normal Petri net places can be added, increasing the expressiveness and complexity. Matrices from a mathematical perspective are considered extraordinary forms of algebra that can express multidimensional views and complex data representations in a compressed form. Combining the MVTN with Petri net structures offers even more modelling possibilities and exploration.

## V. Problem Solution

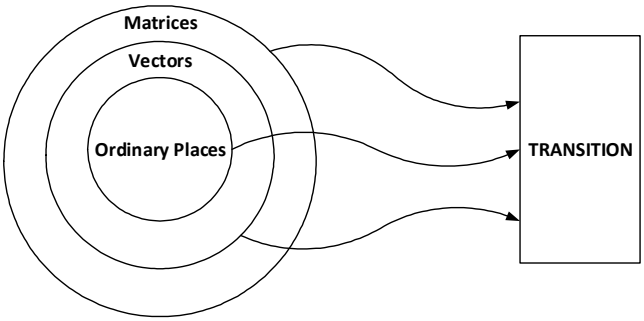


Fig. 1 Relationships between Places, Vectors and Matrices for the MVTN

The proposed solution is to use the MVTN in conjunction with other basic Petri net constructs like places, transitions, etc. The complete definition and explanation of the MVTN is not shown because it would take up too much space. They are explained to some extent in [13].

The main relationship between Petri net places, vectors and matrices is briefly shown in fig.1.

This section outlines the practical implementation of the MVTN. For this purpose, a system can be described as a finite set of interconnected elements (e1, e2, e3,….,en). Basically, element e1 may have at least one input e1? and at least one

output e1! . A system can be defined as [SYSTEM] i.e. the set of processing elements or components of the system. The global communication process is defined as a sequence communicating process: Seq SYSTEM, and communicating process =<e1, e2,.> in any given order. # seq SYSTEM > 0. The global process of communication implies that sub processes can be identified. In each process data is transferred from one component to another. The components or elements can also be considered to be individual processors or processing elements. Given that the net is composed of input matrices and output matrices or vectors: The system state change can be given as (MS,time) $\mapsto$ (MS',time') where MS implies the matrix set of the net and $\mapsto$ denotes a single step transition. Formally the MVTN can be expressed as a system that links up a set of processes or elements. The simple execution of the basic system can be expressed as a set of (inputs, transitions, outputs) i.e. (Inp1$\mapsto$T1$\mapsto$Out1). The change in state as the result of a transition is reflected by the change in the input and output values

Some basic properties about the MVTN are i) global transitions, ii) use of matrices and vectors, iii) complete vs partial transition firing, iv) separation of outputs and inputs, v)

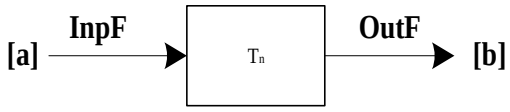


Fig. 2 Generic Form of the MVTN

transition firing by vectors, matrices and places are similar to what happens in Petri nets and other Petri net classes. Other issues like i) concurrency , ii) parallelism, iii) choice or conflict, iv) transition enabling and disabling, v) results of firing a transition, vi) symbolic reachability or marking graph construction and more can be considered to be similar to what happens with Petri net structures [13]. In principle only the input and output types of the net have been changed. The underlying functionality is not modified at all.

For proper firing basically all the input conditions must be satisfied. I.e. the matrices, vectors or places must have sufficient values in them to satisfy the input function. The input and output functions are analogous to Petri net arc weights or arc expressions in colored Petri nets. The input and output functions do not need to be of the same order at all.

Different combinations are possible depending on the modelling requirements. When a processing element does not have an even amount of inputs to form a proper matrix, zero values can be added to solve this. When a transition fires the inputs and output values normally change. Thus, if there are some output values and an output function then the output values must change accordingly, when a transition takes place. Theoretically it is possible to have no output values but just input values as is the case with Petri nets.

For a very oversimplified generic or general form of the MVTN represented in fig.2, transition firing can be basically given as Inp = Inp – Inp_function and Out = Out + Out_ function.

## VI. Some Toy Case studies

This section depicts some simple examples of the use of the matrix vector transition net and its combination with Petri net structures. Some generic communication or interaction

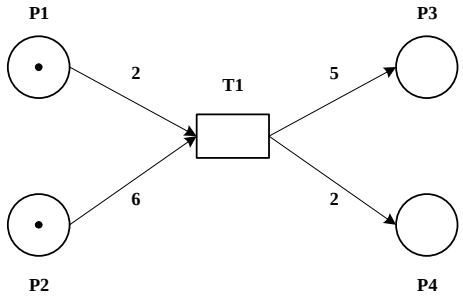


Fig. 3 Normal Place Transition Net

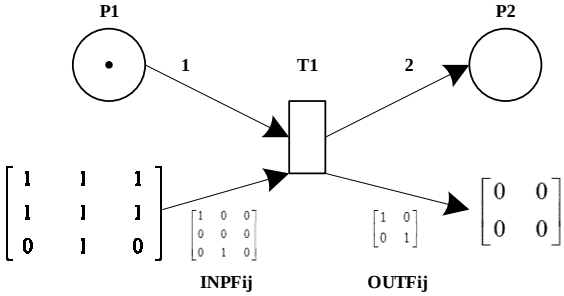


Fig. 4 MVTN with Places Added

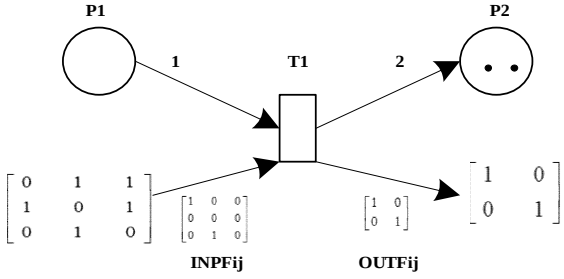


Fig.5 MVTN after Firing T1

structures and their behavior are illustrated.

### A. Basic MTVN combined with normal Places

A normal Petri net diagram in fig. 3 is shown. This is compared with the MVTN counterpart in fig.4. Please note that there is no relationship whatsoever between fig. 3 and fig. 4. It is just an example. It is immediately obvious that the MVTN structures are more detailed and expressive. The diagram in fig. 4. shows how the MVTN combines with ordinary Petri net places.

The basic idea behind this structure opens many new dimensions and possibilities of modelling. The net in fig. 4 is an executable structure where a token can be removed from P1 and the input matrix has sufficient values for transition enabling and firing. I.e. the values in the input matrix are equal or greater than the values in the input function. Hence there are sufficient values for firing T1 i.e. this implies that transition T1 is enabled for firing.

Fig. 5 shows the resultant state of the net in fig. 4 after the firing of transition T1. The status of the net in fig. 5 is non-live one in Petri net terminology. This implies a dead state with no further activity being possible and marking change. In ordinary terms the activities of the net have reached a termination point or conclusion.

### B. Possible Combinations

The diagram in fig. 6 depicts a more complex structure. Even though the inputs to T1 are a matrix and a place P1, the output is a simple place P2 that connects to T2. The net in fig. 6. is executable. However after T2 fires then it will become a non-live net. The place in P2 is a symbolic representation of what can be a i) buffer, ii) store or iii) a switch, etc. that will activate T2 when the right conditions are present.

The diagram in fig. 6 shows that it is possible to have a separate Petri net and MVTN and join both together. The MVTN and the Petri net could be executing in parallel. Hence structures running in parallel can be connected. This example can obviously be extended to other classes of Petri nets for more detailed extensive modelling as required.

The structures shown are executable. After T1 fires one token is placed in P2 and T2 and T3 are activated simultaneously, (see fig. 8). However it is possible for only one to fire. Firing one automatically disables the other. This type of structure presents non-determinism, i.e. different types of behavior and outcomes are possible. This structure could be modified or restricted to guarantee more deterministic behavior.

In Petri net terminology the non-determinism where either T2 or T3 can fire, represents choice or conflict. If T2 fires the net goes into a dead state, (see fig. 9) without further activity

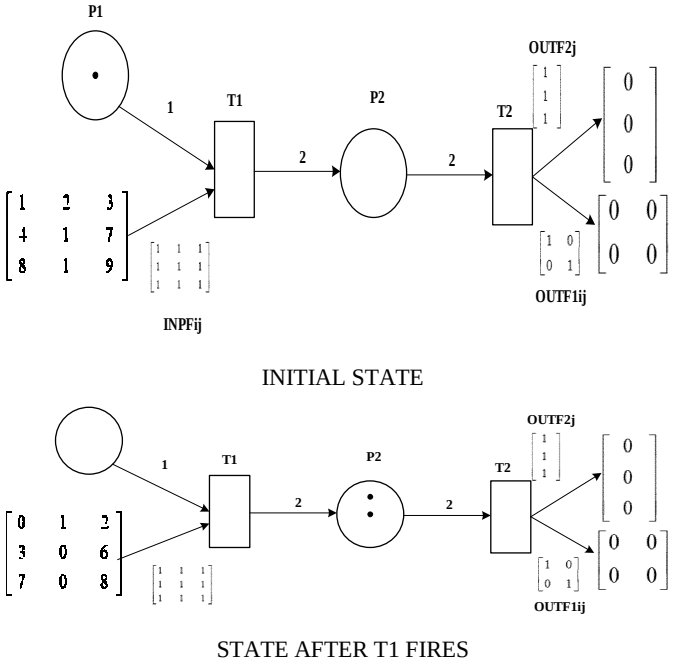


Fig. 6 MVTN structure using a buffer like place connection or communication

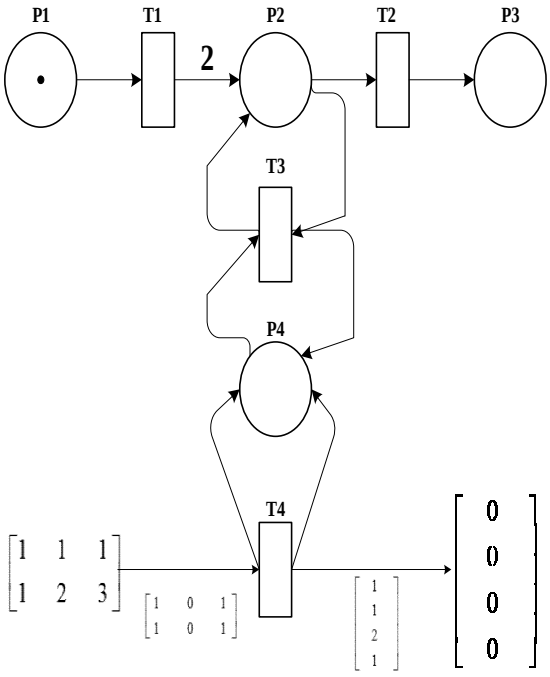


Fig. 7 Separate MVTN and Petri net connecting together

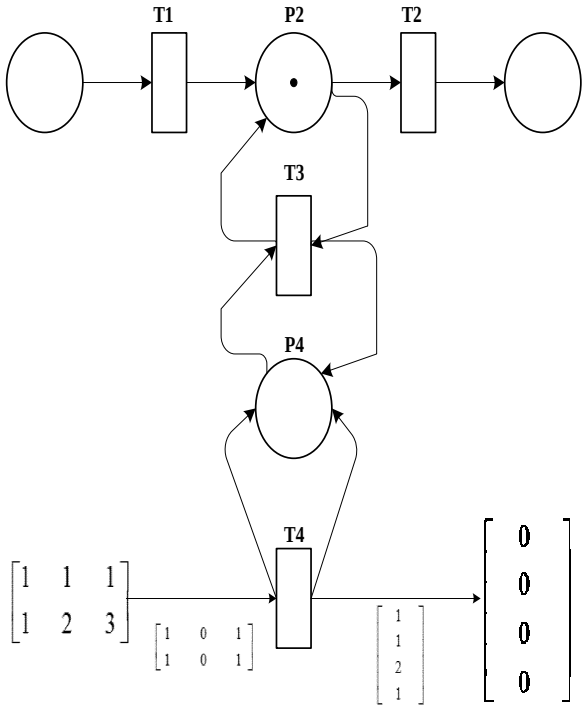


Fig. 8 MVTN State After T1 Fires

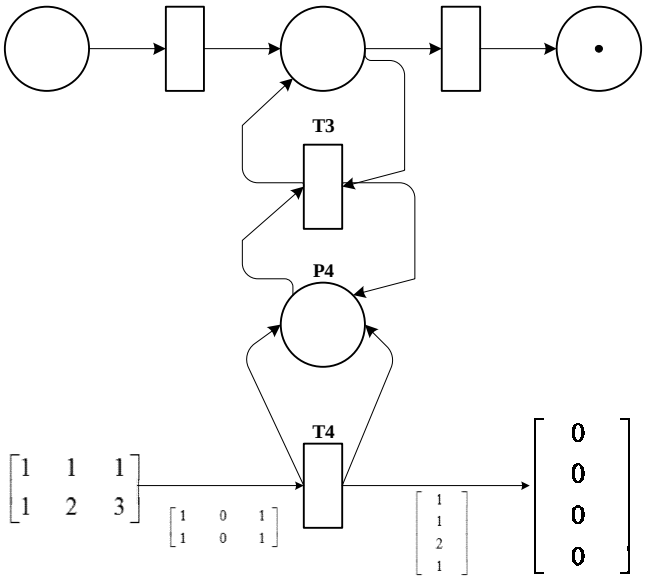


Fig. 9 MVTN State After T2 Fires

being possible. On the other hand if T3 fires some further activity is possible as the subnet below becomes activated. If T2 does not fire T3 can fire.

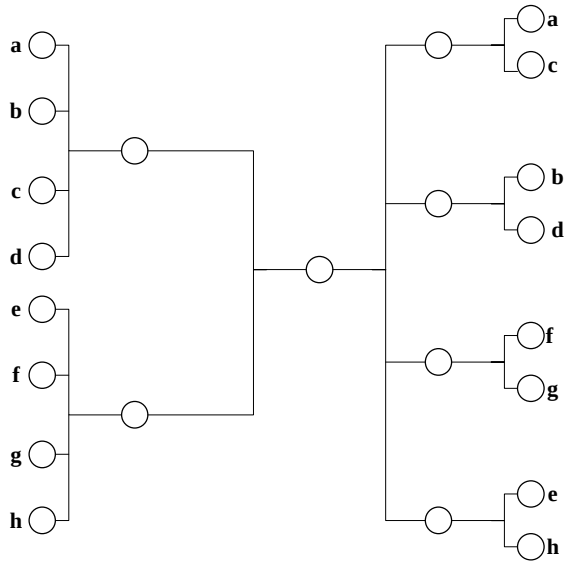


Fig. 10 Symbolic representation of network

### C. *Message Communication and Decomposition*

This example is about pure message communication. In real systems normally at some level there is assembly and disassembly of information or composition or decomposition. This example compares to packet assembly /disassembly, multiplexing and other forms of computer or mobile networking.

The initial block network diagram in fig. 10 shows active ports which communicate to a particular source or entity. The same analogy can be applied for a message that has to be split up into different sources and rerouted or packet assembly and disassembly.

The symbolic MVTN combined with petri net places for input and output is an executable structure. This is shown in fig. 11. Letters are used to symbolically represent the input and output sources respectively. The letters are an abstraction or symbolic representation of possibly real values. At the minimum the MVTN allows for proper representation. The concepts presented in the MTV net can easily be extended to model PTP channels and message splitting algorithms. Fig. 12 and 13 show this operational model with simple transmit values.

### D. *Message Sending between Workstations*

A very simple example of message passing between three stations is depicted in fig. 14. This behavior is typical of message passing between different objects in many modern systems where communication protocols are exchanged between workstations. This behavior can be extended to include more objects and entities.

This behavior is shown using a basic message sequencing chart (MSC). Message sequence charts are widely used to represent various communication scenarios and a vast amount of literature exists about MSCs.

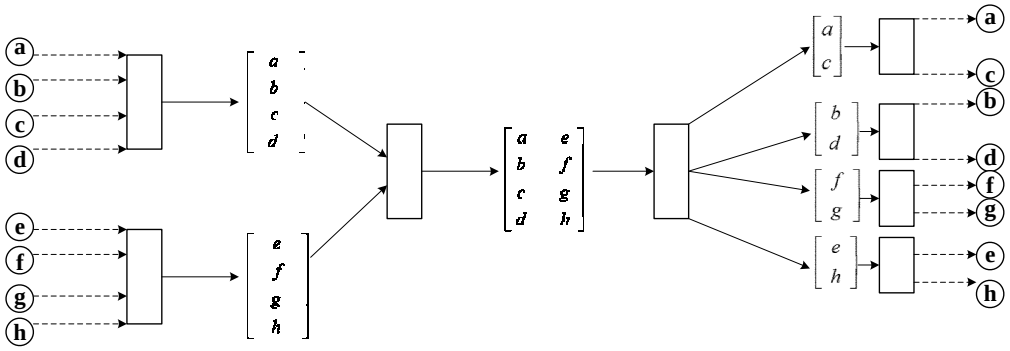

Fig. 11 Generic MVTN for the network assembly/ disassembly structure

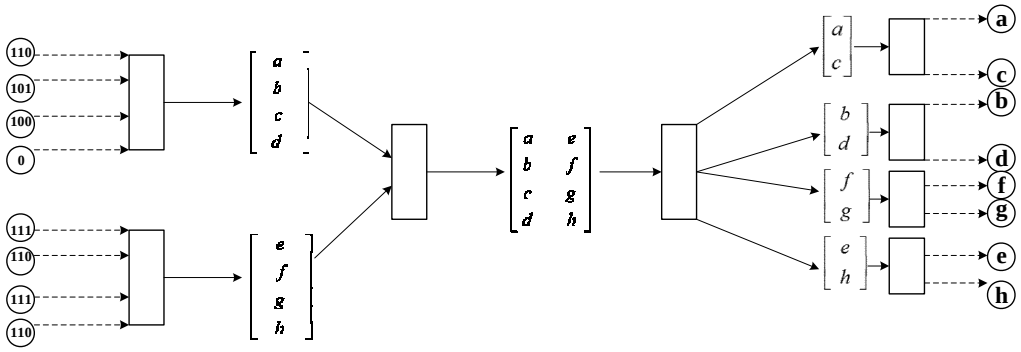

Fig. 12 MVTN for the network assembly/ disassembly structure with binary input values

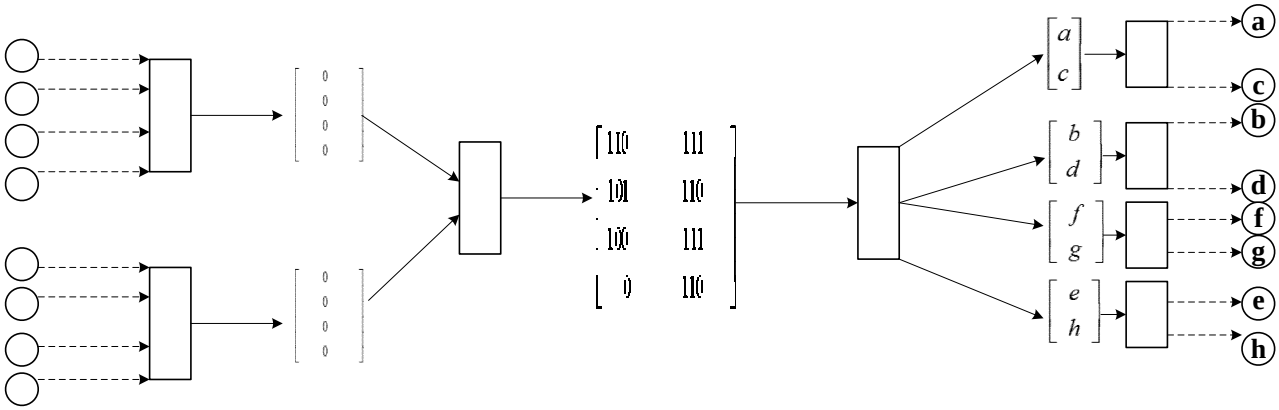

Fig. 13 MVTN for the network assembly/ disassembly structure after firing initial transitions

The MSC in fig. 14 describes the sequencing of events. Fig. 15 shows the MVTN for the whole sequence in fig. 14. The MVTN in fig. 15 is an executable model that contains various states, synchronization and control mechanisms.

## VII. Results

The example in section 6A shows how Petri net places have been combined with the MVTN. The structure is executable. After firing the main transition, it becomes a dead structure. I.e. no further states are possible in Petri net terms. In this case the composite marking of this system is rather simple because it is restricted as there are only a few possible markings.

The example depicted in section 6 B, fig.6. is a bit more complex. Actually there are more states and the behavior of

this net is typical of a buffer, channel or port. Communication is asynchronous, i.e. in one direction only. The MVTN allows complexity to be modeled in this structure. The actual transfer of information can be shown in detail. This model can be modified to show two way communication and add other buffers if needed. This level of modelling can be represented using colored Petri nets but not with ordinary Petri nets.

Fig. 7 depicts some new challenging behavior. Conflict or choice have been deliberately included. The outcomes in this net are undetermined and limited sets of possible markings exist (see fig. 8 and 9). The marking graph for this structure will prove to be more difficult to construct. The main observation from this is that the MVTN does indeed preserve Petri net like behavior. The MVTN allows the modelling of different types of conditions that are normally associated with normal Petri nets. The MVTN or the Petri net can be both used to switch or trigger each other or vice-versa depending on how the net is structured. Fig. 7 indicates that communication between the MVTN and ordinary Petri nets can be done using certain synchronization points as required.

In section 6 C fig. 10, an abstract system that represents real world communication is shown. Fig. 10 just shows that this is possible. Fig. 11 is the executable counterpart of fig. 10. For simplicity's sake in fig. 10 and fig. 11 letters which are representative of the actual values have been used. The MVTN structures are fully executable. They have different states and a reachability graph or marking graph can be constructed. If this is too complex a symbolic marking graph can be used. This will reduce the complexity of the description. These models are useful for representing different types of architectures and component based systems. The more complex the models the more time consuming is their construction. The models are decomposable into Petri nets which will result in information loss. The approach in fig. 8 can be used for modelling other things like parallel algorithms or processing where specific decomposition is needed. This structure is similar to that of a control flow graph. Fig. 12 and 13 show the functional model with real data.

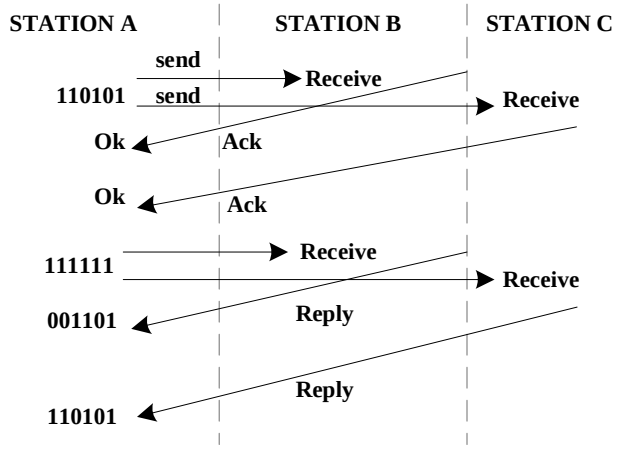


Fig. 14 Typical Message Sequence Chart for Three Workstations

In section 6 D fig. 14 a typical message sequencing activity between three work-stations is shown. The message sequence chart in fig. 14 represents this. The model in fig. 14 is not an executable one. The MVTN for fig. 14 is shown in fig. 15. This is an executable model that can be used for modelling and tracing errors. This model is quite complex and detailed. The messages are shown using vectors this time, but obviously the vectors could be replaced by matrices. The input function is omitted and represented symbolically using ?a, similarly for output !a is used. This can also imply that whatever value is inputted is accepted and outputted. This model is also useful to see at which current state the system is in. I.e. if station A is waiting for a reply from station B to continue broadcasting. These states can be clearly depicted in this network. It is also possible to identify deadlock and concurrency issues from the net. The major concepts that are applicable in Petri nets can be used for this model. The net in fig. 15 is live but after firing all the transitions it goes into a non-reversible state. This depicts exactly what is truly happening in the message sequencing chart. The MVTN can thus be used to verify the correctness and operation of the MSC in a detailed manner.

The modelling approach of the MVTN can be combined with other approaches like those explained in [19]. This would possibly be more useful for practical real world problems.

The MVTN is applicable to real world problems and abstract representation of system structures. This is possible for systems where inputs and outputs can be grouped into matrices. For these systems the MVTN can prove to be quite useful for modelling. The MVTN approach can definitely be combined with other notations from Petri nets and even other Petri net classes if this is required. This can open up a lot of new exploratory modelling. This has been clearly demonstrated in this paper.

If grouping is not possible then this approach might not be suitable for use. For simple problems it is will be better to use Petri nets.

The models can be used for other problems like processor modelling, pipeline architectures [20], hardware modelling [21], concurrent systems etc. and even in other fields like: travel, transportation and logistics modelling, etc.

## VIII. Conclusions

This work has briefly shown the usefulness of the MVTN approach to model interaction and communication in different systems. Obviously the MVTN is based on Petri net like semantics and can be successfully combined with other Petri net components. The best use of this modelling approach seems to be where there are systems that have input and output components or elements that can be easily grouped to form matrices or vectors. The limitations of this type of modelling is that it requires a lot of time to construct more complex and detailed models and the state explosion problem inherent in traditional Petri nets can occur here also. Thus when using this approach, it is important to keep the representation simplified.

Also if there is no grouping the approach is not suitable.

New uses for the MVTN can definitely be considered and a lot of exploratory modelling can be done using this technique. The models can be simplified or converted into Petri nets.

The approach is i) suitable for formal verification and simplification for certain class of problems in systems that can be grouped, ii) connectivity measures and reachability can be considered, iii) experimental results indicate that the models

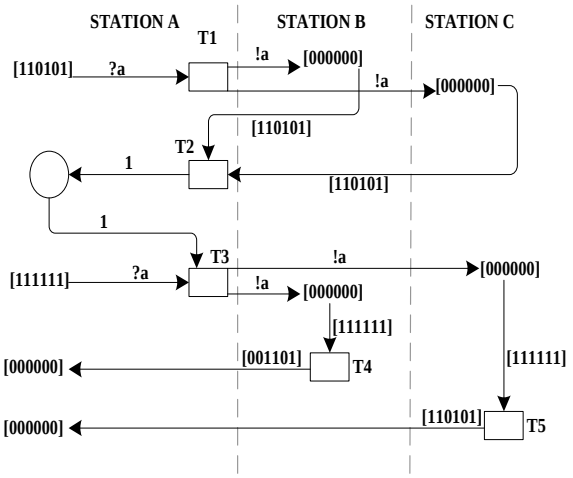


Fig. 15 Typical Message Sequence Chart for Three Workstations

can be improved and fine-tuned, iv) the models can be combined with other constructs and approaches.

## REFERENCES


[1] K.C. Wong, J.G. Thistle, R.P. Malhame, H.-H Hoang, Supervisory control of distributed systems: conflict resolution Decision and Control, 1998. Proceedings of the 37th IEEE Conference on Decision and Control, Volume: 3, 1998, pp. 3275 – 3280.

[2] A. Abellard, P. Abellard Systolic Petri Nets, Petri Nets Applications, InTech, 2010, ch. 5.

[3] E. Best, T. Thielke, Orthogonal Transformations for Coloured Petri Nets, ICATPN '97, Springer, 1997,pp. 447 – 466.

[4] K. Jensen, G. Rozenberg, High-level Petri Nets: Theory and Applications, 1st ed., Springer-Verlag, 1991, ISBN-10: 354054125X ,ISBN-13: 978-3540541257.

[5] A. Spiteri Staines, A Colored Petri Net for the France-Paris Metro, NAUN, International Journal of Computers, Issue 2,Vol. 6., 2012,pp. 111-118.

[6] CPNTools, CPN Group, Department of Computer Science, University of Aarhus, Denmark http://cs.au.dk/CPnets/

[7] L.M. Kristensen, S. Christensen, K. Jensen, The Practioner's Guide to Coloured Petri Nets, International Journal On Software Tools for Tech. Transfer (STTT), Vol. 2, Springer-Verlag, 1998,pp. 98-132.

[8] L.M. Kristensen, J.B. Jorgensen, K. Jensen, "Application of Coloured Petri Nets in System Development", Lecture Notes in Computer Science, Vol. 3098, Springer-Verlag, 2004,pp. 626-685.

[9] B. Scholz-Reiter, C. Zabel, Integration of Load Carriers in a Decentralized Routing Concept for Transport Logistics Networks, Wseas Proceedings of the 2nd International Conference on Theoretical and Applied Mechanics (TAM '11) Corfu, Greece, 2011,pp. 259-264.

[10] J. Cortadella , M. Kishinevsky , A. Kondratyev , L. Lavagno , A. Yakovlev, Hardware and Petri Net Application to Asynchronous Circuit Design, Application and Theory of Petri Nets, Lecture Notes in Computer Science, Springer, Vol. 1825, 2000,pp. 1-15.

[11] K. van Hee, Information Systems: A Formal Approach, Cambridge Univ. Press, 2009.

[12] W. Van Der Aalst, K. Max Van Hee, Workflow Management: Models, Methods, and Systems, MIT press, 2014.

[13] T. Spiteri Staines, Implementing a Matrix Vector Transition Net, British Journal of Mathematics & Computer Science, ISSN: 2231-0851,Vol.: 4, Issue.: 14, 2014, pp. 1921-1940.

[14] T. Spiteri Staines, F. Neri, A Matrix Transition Oriented Net for Modelling Distributed Complex Computer and Communication Systems, WSEAS Transactions on Systems, Vol 13, 2014, pp. 12-22.

[15] K.M. Abadir, J.R. Magnus, Matrix Algebra, Cambridge University Press, 2005.

[16] F. Ayres (jr), Theory and Problems of Matrices, Schaum's Outline Series, Schaum, 1974.

[17] M.B. Dwyer, L.A. Clarke, A Compact Petri Net Representation and its Implications for Analysis, IEEE Transactions on Software Engineering, vol. 22, issue 11, 1996,pp. 794 – 811.

[18] R.H. Sloan, U. Buy, Reduction Rules for Time Petri Nets, Acta Informatica, Vol. 33, Issue 5, Springer-Verlag, 1996, pp. 687-706.

[19] C. Knieke, B. Schindler, U. Goltz, A. Rausch, Defining Domain Specific Operational Semantics for Activity Diagrams, Technical Report Series IfI-12-04, Institut für Informatik, Technische Universität Clausthal, 2012.

[20] C.V. Ramamoorthy, H.F. Li, Pipeline Architecture, Journal ACM Computing Surveys (CSUR), Volume 9 Issue 1, 1977,pp. 61-102.

[21] A. Yakovlev, L. Gomes, L. Lavagno, Hardware Design and Petri Nets, Springer, 2000.

[22] A. Spiteri Staines, Representing Petri Net Structures as Directed Graphs, SEPADS'11, 10th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems, Cambridge UK, 2011, pp. 30-35.

[23] A. Spiteri Staines, A triple Graph Grammar (TGG) Approach for Mapping UML 2 Activities into Petri Nets, SEPADS'11, 9th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems, Cambridge UK, 2010, pp. 90-95.

[24] F. Neri, Traffic Packet Based Intrusion Detection: Decision Trees and Genetic Based **L**earning Evaluation, WSEAS Transactions on Computing, 2005, pp. 1017-1024.

[25] A. Spiteri Staines, A Compact Colored Petri Net Model for Fault Diagnosis and Recovery in Embedded and Control Systems, International Journal of Computers, NAUN, Issue 2, Vol 3, 2009, pp. 222-229.

[26] A. Spiteri Staines, Supporting Requirements Engineering With Different Petri Net Classes, International Journal of Computers, NAUN, Issue 4, Vol 4, 2010, pp. 215-222.

[27] A. Spiteri Staines, From Task Graphs to Petri Nets, International Journal of Emerging Technology and Advanced Engineering, Vol 3, Issue 5, May 2013,pp. 36-42.