Defence Science Journal, Vol. 72, No. 5, September 2022, pp. 721-731, DOI : 10.14429/dsj.72.17987

Construction and Analysis of Petri Net Model for Distributed Cyber-Physical Systems

Vikas Sood,[#,*] Malay Kumar Nema,[#] Rituraj Kumar[#] and Manisha J. Nene[$]

[#]*Centre for Artificial Intelligence and Robotics, Bengaluru – 560 093, India*
[$]*Defence Institute of Advanced Technology, Pune – 411 025, India*
[*]*E-mail: vikas-sood.cair@gov.in*

ABSTRACT

A Distributed Cyber-Physical System (DCPS) composition poses challenges in determining its emergent behaviour. These challenges occur due to (1) the appearance of causal loops of information and energy flow through cyber and physical channels and (2) inherent non-determinism in the temporally ordered flow of events within independently evolving interacting processes of Constituent Systems (CSs). Hence, there is a need to construct a model of the envisaged schematic of DCPS composition for analysis and verification of its significant properties in the conceptual design stage of the system development life cycle.

This paper presents a procedure to construct DCPS composition models in Petri net formalism using distributed abstractions. The model for each CS is obtained from elementary constructs using compositional operators. The interaction among CSs occurs through channels obtained by connecting send and receive constructs of two CSs participating in an interaction. The internal processing within a CS characterizing its primary function is abstracted in a generic pass-through construct. Representing these constructs with compositional operators results in the complete DCPS model in Petri net formalism. A toolchain with Reference net workshop (Renew) as an integrated Petri net editing and analysis platform is configured to support DCPS modelling, simulation and analysis. The Renew tool functionality has been enhanced with a plugin designed and developed by authors to facilitate the drawing of the distributed composition model.

A low-level Petri net analysis (Lola) v2.0 plugin is employed to verify the Petri net and temporal properties of the modelled DCPS scenarios. The properties of the resultant model are verified using well-established algorithms to analyze Petri nets. Further, system properties specified using temporal logic can be verified using model-checking algorithms for Petri nets. A moderately complex scenario involving interactions among six CSs illustrates the presented approach.

Keywords: Model construction; Petri net; Behavioural analysis; Distributed Cyber-Physical systems

1. INTRODUCTION

The spatial dispersal of physical and cyber (computational and communication) elements interacting through asynchronous message passing results in an overall distributed system. In a Cyber-Physical System (CPS)[1-3], computational processes interact with the physical processes through sensors and actuators. The interactions with multiple sensors to actuation loops result in a Distributed Cyber-Physical System (DCPS) instance[4-5]. The challenges in the composition of DCPS arise as the integrated design may miss out on capturing the correct sequencing of interactions on cyber and physical channels.

Behavioural analysis of the envisaged system on an abstracted model can minimize the cost of rework in the System Development Life Cycle (SDLC). The formal models and verification mechanisms present scope for establishing satisfaction of system specifications against logical properties. Petri nets[6-7] are well-established formalisms to model concurrent and distributed systems. A Petri net model is a bipartite graph with vertices as places and transitions connected by directed edges (arcs) as flow elements. A Petri net represents passive (places) and active (transitions) system elements. The actions of the modelled system depend on a limited set of conditions, restrictions, etc., forming the local environment for the action under consideration. Petri nets model system dynamics by firing transitions (atomic actions) depending on their local environment (set of input and output places connected to the transition). This principle of locality is the basis of the superiority of Petri nets in modelling concurrency through transitions with disjoint localities firing concurrently. The dual representation in graphical and algebraic form provides tool support for constructing and analyzing system models in an interactive, user-friendly manner. The underlying algebraic form's expressibility and computability power are retained, allowing verification during the model construction stage. The Petri nets can model a great number of systems[8]. Hence, there is a need to establish a set of elementary building blocks encapsulating relevant details to construct larger systems for the domain of interest.

This work presents an approach to represent distributed abstractions from a set of elementary Petri net constructs. An algebra to capture distributed abstractions with compositional operations is described. A net component-based plugin has been developed for the Reference net workshop (Renew)

Received : 17 March 2022, Revised : 21 July 2022
Accepted : 18 August 2022, Online published : 1 November 2022

tool. The plugin provides a drag and drops functionality for constructing distributed models from relatively independent Constituent Systems (CSs). The mapping of CSs for modelling DCPS scenarios is presented. The applicability of a two-machine production cells scenario as a DCPS instance has been illustrated. The focus while developing the scenario has been to capture nonterminating causal loops of cyber-physical interactions.

The rest of the paper is organized as follows. Section 2 provides details of related work and distinguishes contributions of the work reported in this paper. Section 3 describes the proposed methodology to obtain a Petri net model of DCPS composition from distributed abstractions. Section 4 establishes a mapping of distributed abstractions to elements of a cyber-physical system. Section 5 brings out toolchain setup and enhancements. Section 6 demonstrates the proposed approach with an illustrative application scenario verifying the analysis results for the targeted behavioural properties. Section 7 discusses the aspects of interactions within the ICT processes of DCPS and justifies the selection of toolchains. The conclusion for the reported work is provided in Section 8. The indicative future work is provided in Section 9.

2. RELATED WORK

The prominent formalisms used in the modelling of distributed scenarios are:

(a) Process algebra[9], including CSP, CCS, and ACP with CADP, Concurrency Workbench, and mCRL2 as modelling and verification tools
(b) Petri nets[10] can model control flow within distributed abstractions and provide data models in its high-level extensions. Pipe+, Renew, CPN Tools, and TAPAAL are popular GUI editors and Analysis tools
(c) Actor Models[11] with Ptolemy-II[12] and Rebeca Modelling language[13] providing tool support for modelling and verification

A Petri net as a modelling formalism is intuitive and captures much structural information about the system. A body of analysis techniques has been developed for studying system behaviour. Low-level Petri-nets are supported with matured verification and analysis approaches and tools. Renew is the only tool that incorporates extension mechanisms as plugins. It also provides analysis capabilities as a plugin among the available toolsets. Further, Renew provides a net component plugin to encapsulate patterns for modelling systems as Petri nets.[14] The net component plugin has been used to design the basic component structure for capturing distributed abstraction. The Low-Level Petri net Analysis (LoLA) plugin is used to analyze the dynamic properties of a distributed system.[15] In addition to Petri net-specific techniques like reducing boundedness, liveness, and reversibility properties, model checking with CTL* temporal properties is also supported.[15-17]

Though frameworks for modelling Cyber-Physical Systems using Petri net have been reported, a systematic approach with distributed components for modelling system compositions has not been attempted earlier. A method for formal verification using UML has been recently reported[18]. The approach provides a translator from UML Statechart diagrams to Labelled Transition System (LTS) and provides model-checking support for LTL and CTL property specifications. The approach requires representing behaviour as a global state space and hence does not retain the structure of the composition. The approach is suitable for the standalone embedded system as global state space is comprehensive to the modeller.

An approach to obtain Petri net from SysML Activity diagram specifications is presented by Messaoud et al.[19]. The model checking is supported with temporal specifications. The approach requires the specification of the complete model as an activity diagram followed by translation to the Petri net module for verification. A methodology covering the specification stage to the final implementation of the controller in the distributed devices is presented by Grobelna *et al.* [20]. The system is decomposed into separate modules, and each module participates in forming a distributed system. The specification is verified by applying the model-checking technique against predefined behavioural requirements. The focus is again on top-down decomposition and arriving at an FPGA implementation consistent with the model.

The DCPS conceptual modelling as a network with nodes of type Cyber, Physical, and CPS defined in terms of elements of a capability set using reference net framework provides an alternative approach using high-level features like synchronous channels and hierarchical organization[3]. The approach has been enhanced with a user interface plugin. The plugin facilitates prototyped execution of simulated model to visualize user interactions over multiple user interfaces concurrently[4]. The verification analysis is limited to simulation as unique high-level features of reference nets like synchronous channel communication, and the net-within-net paradigm of reference net formalism lacks matured verification techniques[21].

The proposed approach allows modelling of individual CS capturing local state space and connecting the components with remote terminals to achieve the global state space. The presented framework as a procedure focuses on verifying the control flow within a distributed composition and is a novel attempt in this direction. The presented procedure results in a low-level Petri net model suitable for CTL* based verification. The domain-specific action for DCPS can be enhanced with functional inscriptions to achieve a DCPS prototype where process interactions remain verifiable through model-checking as future work.

3. PROPOSED APPROACH

3.1 Preliminaries: Petri Net

Definition 1: A Petri net graph (PNG) is a 4-tuple

$$PNG = (P, T, F, W) \tag{1}$$

such that;

(1) $P = \{p_1, p_2, p_3, \ldots\} \& T = \{t_1, t_2, t_3, \ldots\}$ are finite sets of Places and Transitions, respectively, where $P \cap T = \phi$ and $P \cup T \neq \phi$.
(2) $F \subset (P \times T) \cup (T \times P)$ is a set of arcs or flow relations.
(3) $W : F \to \{1, 2, 3, \ldots\}$ is an arc multiset. The count (or weight) for each arc is a measure of arc multiplicity.

There is a corresponding graphical representation of the PNG where places are drawn as circles and transitions as

rectangular bars connected with directed arcs. A PNG of four places and three transitions with the directed arcs is shown in Fig. 1(a).

Definition 2: A Petri net (PN) is a marked PNG

$$PN = (P, T, F, W, M) \quad (2)$$

where, $M : P \to \{0,1,2,\ldots\}$ is a place multiset of the PNG. $M(p_i)$ represents the number of tokens present at place $p_i \in P$ depicted as a marking.

The behaviour of a system modelled in Petri net formalism is studied as an evolution of PN. $M=M_0$ is known as initial marking using a single rule of transition firing. Figure 1(b) shows a PN as marked PNG with $M_0 = [2,0,0,0]$.

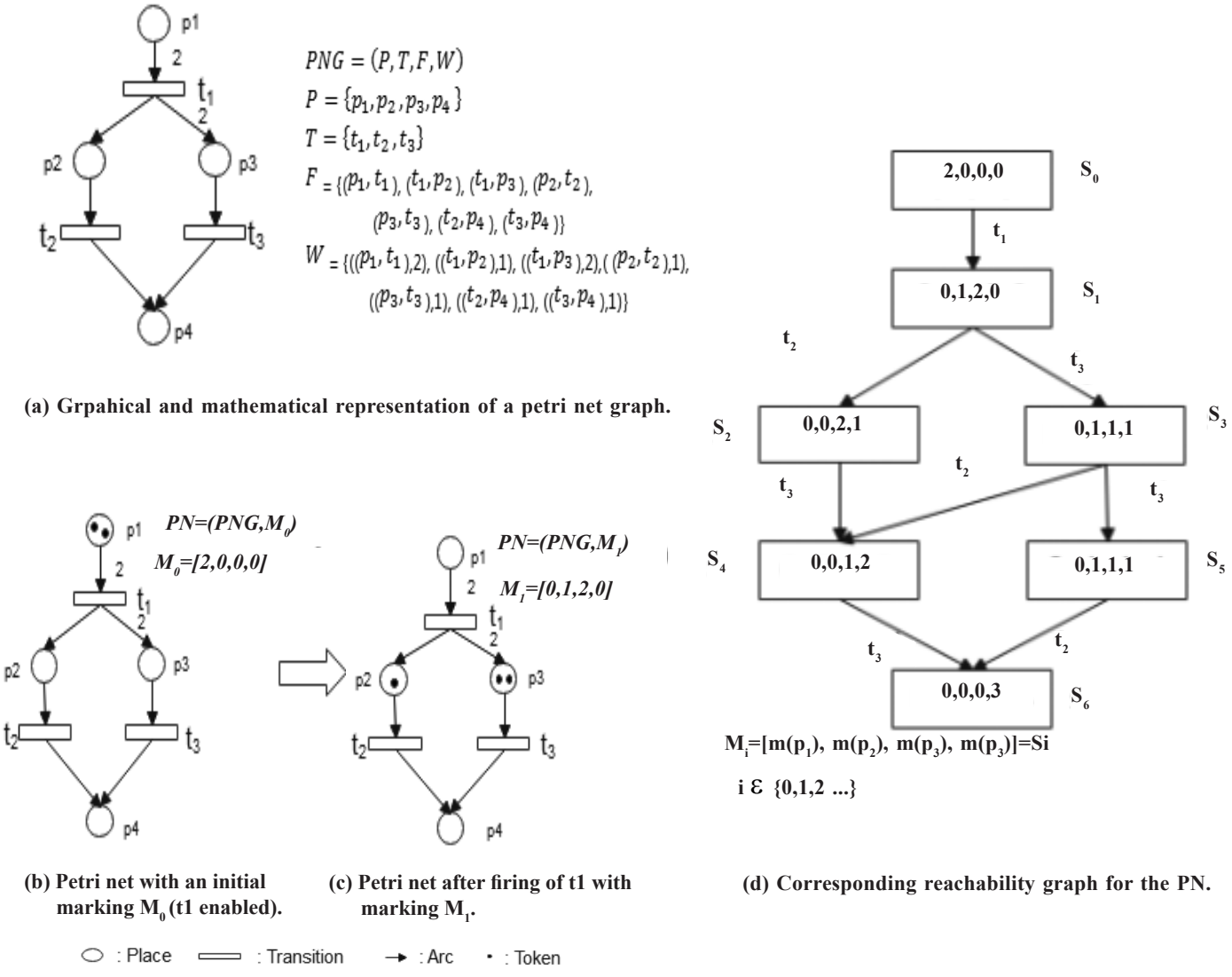
Definition 3: PN transition firing semantics

(1) A transition $t_i \in T$ a PN is enabled if all the input places ${}^0t \in P$ are marked with at least $w(p,t_i)$ tokens, where $w(p,t_i)$ the weight of the arc from place $P \in {}^0t_i$ to t_i. Here notations ${}^0t_i, t_i^0$ denote pre, post set of the transition t_i.

(2) An enabled transition t_i may or may not fire (depending on event occurrence).

The firing of t_i removes $w(p,t_i)$ tokens from $P \in {}^0t_i$ and adds $w(t_i,p)$ tokens to each of its output places $p \in t_i^0$. Fig. 1(b) shows a PN with $M_0 = [2,0,0,0]$, resulting in enabling t_i.

Figure 1(c) shows the PN with $M_1 = [0,1,2,0]$ after firing of t_i. The dynamics of a PN can be analyzed as a simulation run. The simulations help in debugging and model improvements but cannot establish the absence of errors. The model checking offers a fully automated mechanism to establish satisfaction with system specifications. The specifications as temporal formulae are verified over a transition system as a dynamic model of the system.



(a) Grpahical and mathematical representation of a petri net graph.

(b) Petri net with an initial marking M_0 (t1 enabled).

(c) Petri net after firing of t1 with marking M_1.

(d) Corresponding reachability graph for the PN.

Figure 1. A Petri net with four places and three transitions with initial marking and subsequent application of firing rule on transition t_i. The corresponding reachability graph is also depicted.

A reachability graph of a PN is a directed graph where successive states constitute vertices starting from M_0 and are connected through the directed edge(s). The directed edge represents the transition action affecting the change of the current state to the next state. The reachability graph is a transition system with a single initial state. It is the primary structure for verifying system specifications represented in temporal logic. The reachability graph for the example PN is shown in Fig. 1(d).

3.2 Model-Checking and Temporal Logic

Model-checking uses temporal logic for automated verification. In contrast to propositional and predicate logic, a temporal logic formula is not statically true or false in a model. Instead, the model contains several states, and the formula can be true in some states and false in others. Model M is a transition system, and the properties φ are formulas in temporal logic. The verification encompasses three things:

(a) A model M , where PN's reachability/ coverability graph acts as a transition system

(b) The property is specified as a temporal logic formula φ . LTL, CTL, and CTL* encoding are prevalent formalisms for specifying φ

(c) A verification algorithm with inputs M and φ and output Yes/No/Error indicating satisfaction/ non-satisfaction/ error condition (out of memory), respectively

The syntax of CTL* involves two classes of formulas:

(a) State formulas, which are evaluated in states:

$$\varphi := p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid A[\alpha] \mid E[\alpha]$$

(b) Path formulas, which are evaluated along paths:

$$\alpha := \varphi \mid (\neg\alpha) \mid (\alpha \wedge \alpha) \mid (\alpha U \alpha) \mid (G\alpha) \mid F\alpha \mid (X\alpha)$$

where p is an atomic formula, α is any path formula. φ any state formula.

State operator X, F, G and U represents the next state, future state (eventually), globally (always), and until. Path operators A and E represent all paths and a path (possibility), respectively. The CTL* subsumes LTL and CTL formulas. LoLA2.0 - A Low-level Petri net-analyzer used for verification in this study supports the CTL* property specification.

3.3 Distributed System

Definition 4: A distributed system (DS) is a tuple of labelled interactions of Constituent Systems (CSs).

$$DS = (A, \Lambda, I)$$
$$A = \{CS_i : 1 \le i \le n; n \ge 2 \,\&\, n \in \mathbb{N}\}$$
$$\Lambda = (l, <)$$
$$I = \{(CS_i, l, CS_j) \subseteq A \times \Lambda \times A; i \ne j\} \quad (3)$$

The CSs are composed of elementary constructs joined through compositional operators to achieve models of relatively independent CSs. The set of interactions (I) among participating CSs provides the dynamic behaviour to the overall distributed system. The interactions are labelled in an ordered manner to provide unique naming and to incorporate causal dependence among interactions. The detailed representation and semantics are presented in the following sections.

3.4 Elementary Constructs

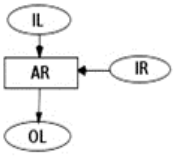
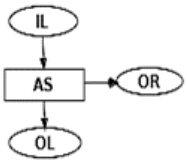
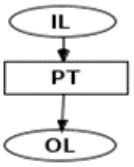
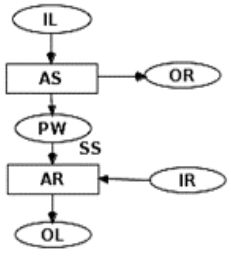
Considering atomic action within a distributed composition of CS, elementary constructs (EC) of Type Asynchronous Send (AS), Asynchronous Receive (AR), and Pass-Through (PT) are identified.

$$EC = AS \mid SS \mid AR \mid PT \mid \varepsilon \quad (4)$$
$$PNG(EC) = (P, T, F) \quad (5)$$
$$P = I \cup O; |I|, |O| \ge 1 \quad (6)$$
$$I = I_L \cup I_R \quad (7)$$
$$O = O_L \cup O_R \quad (8)$$

where, I_L / I_R and O_L / O_R are local/ remote input and local/ remote outputs, respectively.

Each EC depends on local/ remote inputs to perform elementary actions within the CS. Additionally, Synchronous Send (SS) is a sequential combination of AS followed by the AR element. In SS, the sending thread waits till the synchronizing action between sending and receiving CSs has been completed. The EC ε null action is incorporated as terminating elements. The PT element is mapped to the atomic action of CS through label assignment. A PT consumes local input and produces local output on its firing. The algebraic description and corresponding graphical notation for each EC are provided in Table 1.

Table 1. Elementary constructs as PNG

Algebraic Description	Graphical Notation
$PNG(AR) = (P, T, F)$ $P = \{IL, OL, IR\}$ $T = \{AR\}$ $F = \{(IL, AR), (IR, AR), (AR, OL)\}$	
$PNG(AS) = (P, T, F)$ $P = \{IL, OL, OR\}$ $T = \{AS\}$ $F = \{(IL, AR), (IR, AR), (AR, OL)\}$	
$PNG(PT) = (P, T, F)$ $P = \{IL, OL\}$ $T = \{PT\}$ $F = \{(IL, PT), (PT, OL)\}$	
$PNG(SS) = (P, T, F)$ $P = \{IL, OR, PW, IR, OL\}$ $T = \{AS, AR\}$ $F = \{(IL, AS), (AS, OR), (AS, PW), (PW, AR), (AR, IR), (AR, OL)\}$	

3.5 Constituent System Composition

The atomic actions within a CS proceed in a sequential or parallel configuration with non-deterministic choice, concurrent executions, and their synchronization with a possibility of repeated execution. A CS is obtained by applying the following compositional operation on local inputs (IL) and local outputs (OL) of the CS elements:

(a) Sequence (;): The causal dependence of atomic actions is indicated with a sequential operator where post-action can only occur after the pre-action has been completed.

(b) Non-deterministic choice ($\oplus$): The choice of one or more sets of actions in the subsequent execution step involves a choice operation.

(c) Concurrency ($\curlywedge$): The concurrent or overlapped execution of two or more sets of actions in the subsequent step involves concurrency operation.

(d) Synchronisation ($\curlyvee$): The scenarios with concurrent or overlapped execution of two or more sets of actions being synchronised in the subsequent step involve a synchronised operation.

(e) Repetition (CS^*): The scenarios with repeated execution of two or more sets of actions involves Repetition operation.

The algebra for the composition of CS from ECs is depicted in eq. 9. The approach is generic enough to construct a combination of PNG with modelling power to represent sequential and concurrent flow (with synchronization). The PNG obtained using place fusion for the above operations is depicted in Table 2.

$$CS = EC \left| CS1; CS2 \mid CS1 \oplus CS2 \right| \\ CS1 \curlywedge CS2 \left| CS1 \curlyvee CS2 \right| CS^* \tag{9}$$

For providing domain-specific context (Pragmatics) to each CS, Transitions are labelled to represent domain-specific actions, and local places are labelled with domain-specific states.

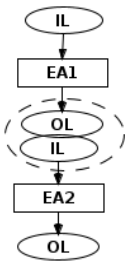
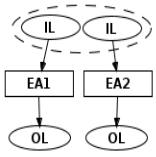
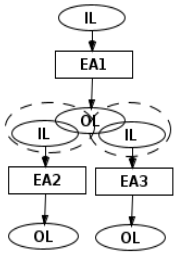
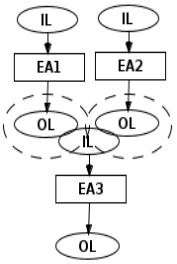
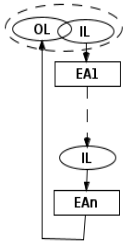
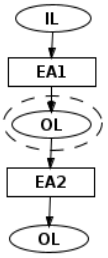
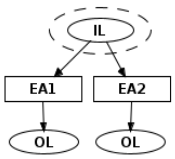
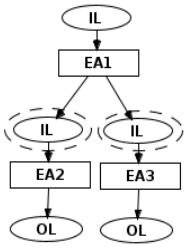
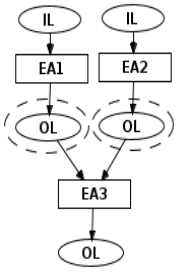
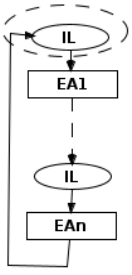
3.6. Integrating Distributed Whole and Initialization

Once the CSs have been modelled, the distributed whole is achieved by fusing one CS's remote output places to the other CS's remote input places with the identical label. The resultant structure is a PNG. An initial marking is assigned to the PNG. The PN obtained is analyzed as the simulation runs or verified through property-checking algorithms.

4. MAPPING DISTRIBUTED ABSTRACTIONS TO CYBER-PHYSICAL SYSTEM

A Cyber-Physical System involves the interaction of processes in the cyber layer with the processes in the physical layer through sensors and actuators. On the other end, the cyber part may interact with Human processes through the human-computer interface for situational assessment and external action by the Human agents to achieve the system objectives. The interacting processes in human, cyber, and physical layers are depicted in Fig. 2. The Cyber-Physical Interface (CPI) connects the cyber and physical layers with sensors and actuators. The interaction between the cyber and physical layer processes takes place through the physical channel. The exchange of information through CPI is asynchronous, and physical processes do not wait for synchronization with the Cyber Processes for action. Hence, there cannot be a SS element in CPI. A sensor is represented as an AR element, and an actuator is represented as AS element. The Domain-specific sensors/actuators are labelled with AS/AR for representing these atomic actions.

Table 2. Composition operators and graphical composition using place fusion

Operator	Sequence	Choice	Concurrency	Synchronization	Repetition
Expression	$EC1; EC2$	$EC1 \oplus EC2$	$EC1; EC2 \curlywedge EC3$	$EC1 \curlyvee EC2; EC3$	CS^*
Place Fusion					
Post Fusion					

(a) The exchange of network messages in the cyber channel is also asynchronous. However, using the SS element on the sending side blocks the process until the synchronization message is received on the remote input place, which meets all the requirements of synchronous message exchange.

(b) The Physical process indicates its status by exciting the deployed sensor element. The excitation is represented using AS element within the physical process execution. Similarly, the actuator action stimulates the physical process, and this aspect is captured as an AR element within the physical process model.

(c) Human action through HCI is entirely determined by the design of actions in the cyber process. The PT element abstracts interaction by human role and thus necessarily need not be required to be explicitly modelled.

Application of the elemental mapping to the distributed abstractions achieves the Petri net model for DCPS compositions.

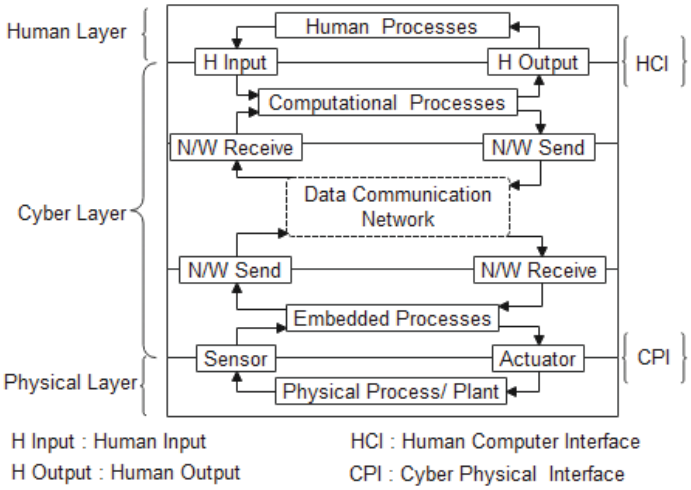


Figure 2. Schematic of a cyber-physical system as a distributed set of interacting processes in the cyber, physical and human layers.

5. TOOLCHAIN SETUP AND ENHANCEMENTS

The CS and subsequently integrated DS models are achieved by applying place fusion as per required compositional/ integration operations. The procedure to achieve PNG for CSs is incorporated within the Renew tool as a DS Model component. The DS Model plugin has been designed and developed to support the above-described procedure for DCPS model construction. The DS Model plugin integrated with the Renew editor and analysis tool provides drag and drop functionality for drawing graphical PNG(EC).

A snapshot of the enhanced view of Renew tool with the DS Model palette is shown in Fig. 3. One instance for each EC type represented as a PNG using the DS Model palette is also displayed in Fig. 3. For a modeller, the activity involves selecting and placing EC from the palette and performing place fusion to apply composition operation to achieve the PNG model of the distributed setup. LOLA 2.0 plugin is utilized to analyze the PN model for a scenario under study.

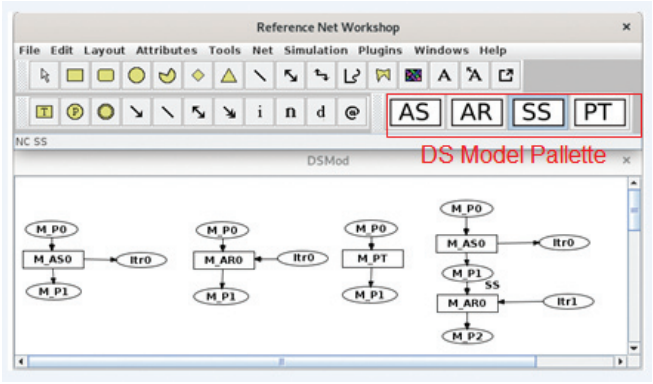


Figure 3. DS model palette and usage in Reference Net Workshop.

Note: We designed and developed DS Model Plugin that provides interactive placement of Petri net graphs. An instance of each Elementary Construct is shown on the drawing in the editor.

6. ILLUSTRATIVE SCENARIO

6.1 Scenario Description

A hypothetical two-machine production cell system comprising the following components and operational descriptions is considered.

(1) Conveyor belts (C1 & C2): There are two belts, the C1 carries raw items (Ri) as input to this production system from the outside world, and C2 carries the finished items (Fi) to the outside world. These belts form part of this production system's environmental entities being sensed and acted.

(2) Machine1 (M1)**:** It senses the availability of a raw item's lot (Ri) on C1 and waits for the occurrence of the Raw item's available (E1) event. On detection of E1, it requests the robot to load Ri on M1 and carries out processing activities to transform Ri into a partially finished item (Pi). Through operation1 (Op1). After completion of Op1, it intimates the Robot (R) to unload the Pi from M1 to Buffer (B) after checking the availability of empty buffer space. This sequence of activities is repeated to process Ris.

(3) Machine2 (M2): It senses the availability of a Pi on B and waits for the occurrence of a Partially processed item available (E2) event. On detection of E2, it requests the robot to load the Pi on M2. It carries out processing activities to transform Pi abstracted as hypothetical operation2 (Op2). After completing Op2, the availability of a slot in C2 is checked. It intimates the R to unload the Fi from M2.

(4) Robot (R): It receives a request from M1 to load Ri from C1 and unload Pi after Op1 to Buffer(B). Similarly, it receives a request from M2 to load Pi from B and to unload Fi after Op2 to C2.

(5) Buffer(B): It acts as a transient store with a fixed capacity (seven for this scenario) for Pi produced by M1 waiting for transportation by R to M2.

6.2 Constituent System and Interaction Identification

The schematic of the scenario is depicted in Fig. 4. The functional description is provided for each of the participating CS. The Interactions have been labelled in an ordered manner. Further, the channels of interactions have been classified as cyber or physical to identify the type of communications among the components (Asynchronous/ Synchronous).

6.3 Constituent System Model Construction

From the schematic of the distributed setup, the six CSs are identified. The CSs, namely Input Conveyor Belt, Machine1, Robot, Buffer, Machine2, and Output Conveyor Belt, interact on cyber and physical channels to generate the dynamic behaviour of the overall DCPS. Using Elementary Constructs and composition operators, the resultant PNG for each CSs is provided in Figure 5. The places and transactions are relabelled to represent the role of EC in CS. The mapping of labels to domain-specific descriptions provides pragmatics to the modelled scenario.

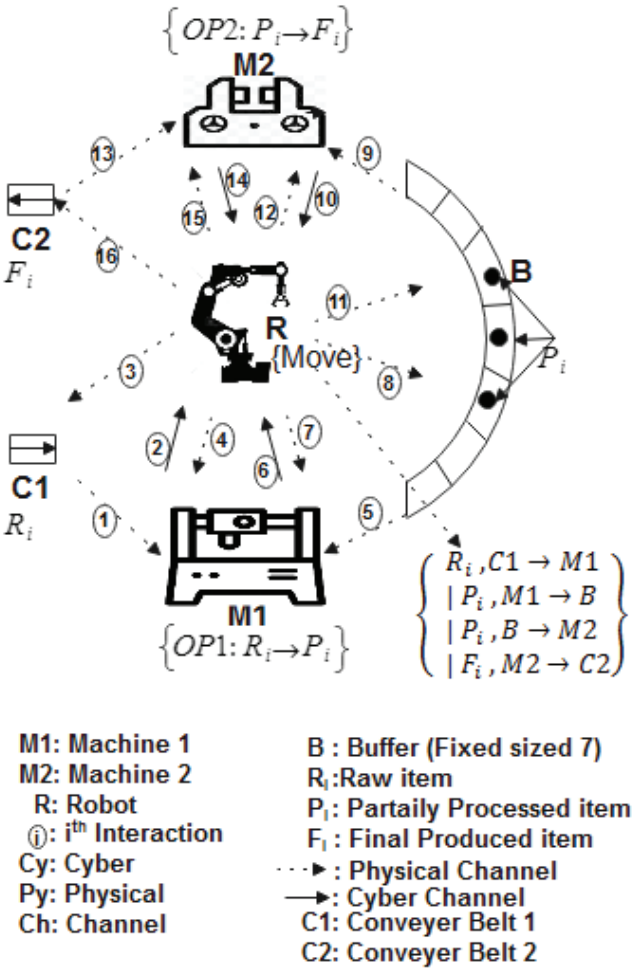
6.4 Integrated PNG Model and Analysis

Integrated PNG achieved after applying place fusion to identically labelled Remote Inputs and Outputs is shown in Figure 6. The initial marking is applied with Tokens ([]) placement in the places representing initial conditions for the scenario. The following properties are directly verified on the invocation of the LOLA plugin.

(a) Quasi-Liveness: This signifies that every transition in the net with initial marking is non-dead (live).
(b) Liveness: Indicates that a firing sequence can be enabled for all reachable markings and thus may fire.
(c) Deadlock freedom: Indicates whether the modelled system can exhibit deadlock-free operation or not.
(d) Reversibility: Depicts whether the initial marking is reachable from every reachable state.
(e) Boundedness: Indicates whether all the places are bounded

For the model depicted in Figure 6, all these properties are satisfied. The temporal properties are added to the Model as CTL* formulas through the tasks UI of the LOLA plugin. For the two machine production cells scenario following properties are specified and verified.

(1) Buffer Capacity: The buffer capacity is always seven.
(2) Absence of buffer underflow and overflow.
(3) The eventuality of a deadlock.
(4) Mutual exclusive execution of Robot transport function.



ⓘ	Ch	Interaction Description
1	Py	M1 senses readiness of Ri at C1
2	Cy	M1 messages R to move Ri to M1
3	Py	R action for moving Ri at C1 completed
4	Py	R action for moving Ri to M1 completed
5	Py	M1 senses B for buffer availability
6	Cy	M1 messages R to move Pi to B
7	Py	R action for moving Pi to B at M1 completed
8	Py	R action for moving Pi to B completed
9	Py	M2 senses Pi availability at B
10	Cy	M2 messages R to move Pi to M2
11	Py	R action for moving Pi at B completed
12	Py	R action for moving Pi to M2 completed
13	Py	M2 senses C2 for dispatch of Fi.
14	Cy	M2 messages R to move Fi to C2
15	Py	R action for moving Fi at M2 completed
16	Py	R action for moving Fi at C2 completed

Figure 4. Schematic setup of two machines production cell scenario. The labelled interactions over the cyber and physical channels are also indicated and described.

The verification task inserts the specifications as CTL* formulae with the model as an integral element. The execution of the verification command of the LOLA plugin displays the status of the satisfied property in green colour and a not satisfied property in red colour, as shown in Fig. 6. In case of deviation from desired behaviour, the simulation can be used to debug the model. The revised model could be repeatedly verified to comply with the behavioural specifications.

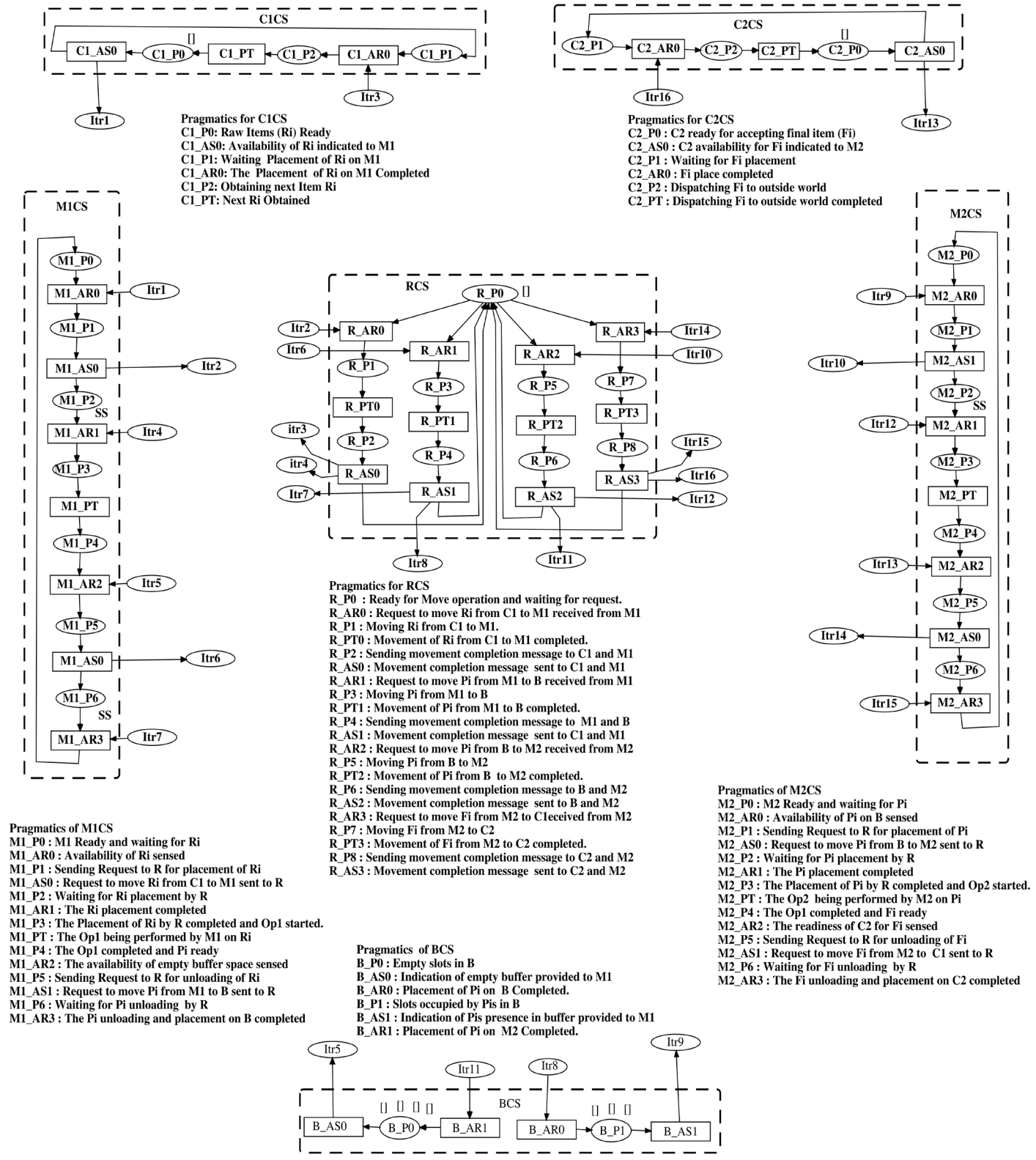


Figure 5. The constituent system compositions for six constituent systems of two machine production cell system obtained after applying composition operators on elementary constructs. The pragmatics for places and transitions of constituent systems are also indicated.

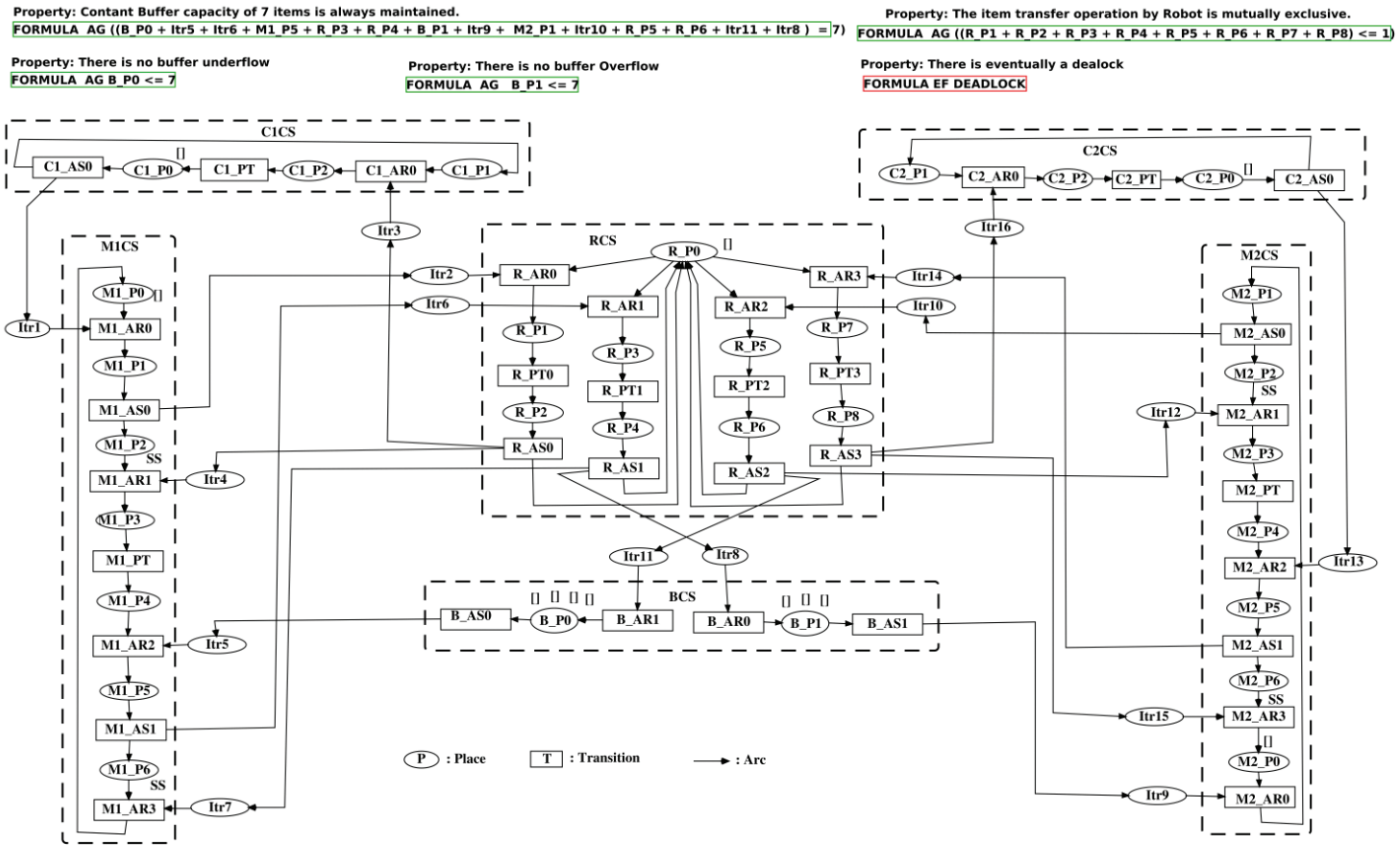


Figure 6. The overall petri net model of the two machines production cells system.

Note: The model is achieved by applying steps of modelling CSs and fusing CSs interaction by merging the label of remote outputs of the sending (AS and SS) elements with the remote input terminal of AR elements. The CTL* properties are also embedded into the model. The Green/Red colour signifies Satisfaction/ Non- the satisfaction of properties after verification through LOLA 2.0 plugin functionality.

7. DISCUSSION

The ICT processes on the Human-computer interface drive the information exchange with humans. The ICT processes interacting with physical processes are reactive, i.e., they take inputs from the environment continuously and respond to the environmental stimuli. Hence, human processes can be represented with a human input/ output action as a PT element where Human-in-the-loop decision-making is required. The scenario for two machine production cell systems has been kept simple by only concentrating on cyber-physical system interactions.

The usage of the Net component plugin allows the design of domain-specific patterns to ease the representation of PN models in a systematic and verifiable way. This is the first effort known to the authors facilitating domain-specific modelling in a graphical framework of Petri nets supporting formal representation and verification. The unique plugin support of the Renew tool facilitates enhancements.

The presented work utilizes a Low-level Petri net analysis tool (LOLA v2.0) for verification of the CTL* properties of the system specifications. Performance analysis using Timed CTL for the correctness of timing requirements and Probabilistic CTL* are not supported by LOLA v2.0 as well as any of the Renew plugins. The timed arc models are possible in Renew but can only be studied for simulation analysis. TAPPAL[22] is one of the timed arc Petri net modelling and analysis tools suitable for this purpose. It lacks a plugin feature for enhancing the modelling capacities of the tool.

Another aspect of the performance analysis is probabilistic logic for performance analysis supported by Stochastic Petri nets with tool support provided by GreatSPN[23] and PIPE[24]. Renew tool through inscriptions allow the representation of probabilistic models, but the analysis is limited to the simulations only. As far as the study of different scenarios within a model is concerned, the model checker evaluates the temporal query on all/required traces of the execution to provide property satisfaction results.

8. CONCLUSION

An approach focusing on behavioural analysis of distributed systems, in general, using Petri nets is presented. An end-to-end procedure for model-checking with Petri net verification with untimed dynamics has been developed and applied to modelling Distributed Cyber-Physical Systems. A scenario capturing interactions of cyber, physical and cyber-physical constituent systems with causal loops of information flow is demonstrated. Incorporating the human-computer interaction in the Pass-Through (PT) construct subsumes the human role in the envisaged system.

The presented approach is significant as modelling

distributed scenarios requires intricate knowledge of formal methods. Employing Petri nets, Process Algebra, or languages like Erlang and Promela do not map directly to the terminology of applied engineering domains. The presented work enables system engineers/ architects and analysts to construct models in prevalent domain terminology and gain the benefits of underlying matured analysis capabilities of the Petri net formalism. This should motivate the incorporation of other domain-specific plugins on relevant action meta-models and their usage with supported unified toolchains.

9. FUTURE WORK

The proposed procedure can be validated for various distributed scenarios with properties expressing domain-specific liveness, safety and fairness requirements. This can be taken up as a separate exercise depending upon the domain of interest. Also, mapping the asynchronous send receive element for capturing cyber-physical layer interaction can be replaced by sense and actuate elements, making the presented framework more specific for modelling DCPS scenarios like containment monitoring, water and electric SCADA, industrial automation etc. The inscription retaining properties of low-level process flow can be used for enhancements for obtaining interactive user prototypes. The plugins to integrate the Timed CTL and Stochastic CTL analysis module using the existing implementation in other tools can enhance the overall analysis capabilities of the Renew tool chain setup.

REFERENCES

1. Baheti, R. & Gill, H. Cyber-physical systems. The impact of control technology, 2011, **12**(1), 61-166.
2. R. Rajkumar, LeeI. , Sha, L. and Stankovic, J. Cyber-physical systems: The next computing revolution, in Design automation conference, 2010, 731–736. doi:10.1145/1837274.1837461
3. Lee, E.A. The past, present and future of cyber-physical systems: A focus on models, sensors, 2015, **15**(3), 4837-4869.
4. Sood, V.; Nema, M.K.; Kumar, R. & Nene, M.J. Conceptual verification of distributed cyber physical systems using reference nets. *Procedia computer science*, 2020, **171**, 81-90. doi:10.1016/j.procs.2020.04.009
5. Sood, V.; Nema, M.K.; Kumar, R. and Nene, M.J. A framework for prototyping distributed cyber-physical systems with reference nets, simulation modelling practice and theory, 2022, **117**, 102488. doi:10.1016/j.simpat.2022.102488
6. Murata, T. Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 1989, **77**(4), 541-580. doi:10.1109/5.24143
7. Narahari, Y. Petri Nets-Overview and foundations. *Resonance*, 1999, **4(**8**)**, 58-69. https://www.ias.ac.in/article/fulltext/reso/004/08/0058-0069
8. Girault, C. & Valk, R. Petri nets for systems engineering: a guide to modeling, verification, and applications, springer science & business media, 2013. https://citations.springernature.com/book?doi=10.1007/978-3-662-05324-9
9. Baeten, J.C. A brief history of process algebra, theoretical computer science, 2005, **335**(2-3),131-146. doi:10.1016/j.tcs.2004.07.036
10. Giua, A. & Silva, M. Petri nets and automatic control: A historical perspective, *Annual reviews in control*, 2018, **45**, 223-239. doi:10.1016/j.arcontrol.2018.04.006
11. Agha, G.A. Actors: A model of concurrent computation in distributed systems, Massachusetts Inst. of Tech. Cambridge Artificial Intelligence Lab, 1985. http://hdl.handle.net/1721.1/6952
12. Ptolemaeus, C. System design, modeling, and simulation using Ptolemy II, Ptolemy. org Berkeley, 2014. http://ptolemy.org/books/Systems
13. Sirjani, M. Rebeca: Theory, applications, and tools. In international symposium on formal methods for components and objects, 2006,102-126. doi:10.1007/978-3-540-74792-5_5
14. Cabac, L. & Moldt, D. Net components: concepts, tool, praxis, petri nets and software engineering, international workshop, PNSE, 2009, **9**, 17-33. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.723.8390&rep=rep1&type=pdf#page=23
15. Wolf, K. Petri net model checking with LoLA 2, application and theory of petri nets and concurrency. PETRI NETS 2018. *Lecture Notes in Computer Science*, 2018, **10877**, 351-362. doi:10.1007/978-3-319-91268-4_18
16. Merz, S. Model checking: Modeling and verification of parallel processes: 4th Summer School, *MOVEP* 2000 Nantes, France, June 19--23, 2000 Revised Tutorial Lectures, 2001, **2067,** 3-38. doi:10.1007/3-540- 45510-8_1
17. Clarke, Edmund M.; Henzinger, T.A.; Veith, H. ; Bloem, R. and others. Handbook of model checking, Springer, 2018. doi:10.1007/978-3-319-10575-8
18. Kochaleema, K. & Kumar, G.S. Generic methodology for formal verification of UML models. *Def. Sci. J.*, 2022, **72**(1), 40-48. doi:10.14429/dsj.72.17228
19. Messaoud, Rahim; Boukala-Ioualalen, Malika & Hammad, Ahmed . Petri nets based approach for modular verification of SysML Requirements on Activity Diagrams, PNSE@ Petri Nets,2014, 233-248. http://ceur-ws.org/Vol-1160/paper14.pdf
20. Iwona, Grobelna; Wiśniewski, Remigiusz; Grobelny, Michał & Wiśniewska, Monika . Design and verification of real-life processes with application of Petri nets. *IEEE trans. on systems, man, and cybernetics: Sys.,* 2016, **47**(11), 2856-2869. doi:10.1109/TSMC.2016.2531673
21. Sven, Willrodt; Moldt, Daniel & Simon, Michael . Modular model checking of reference nets: MoMoC. PNSE@ Petri Nets, 2020,181-193. http://ceur-ws.org/Vol-2651/paper12.pdf

22. David, A.; Jacobsen, L.; Jacobsen, M., Jørgensen, K.Y.; Møller, M.H. & Srba, J. TAPAAL 2.0: Integrated development environment for timed-arc petri nets. International conference on tools and algorithms for the construction and analysis of systems, 2012, 492-497. doi:10.1007/978-3-642-8756-5_36
23. Amparore, Elvio G. Stochastic modelling and evaluation using GreatSPN. *SIGMETRICS Perform. Eval. Rev*, 2022, **49**(4),87-91. doi:10.1145/3543146.3543165
24. Llado, Catalina M. PIPE 2.7 Overview A Petri Net tool for performance modeling and evaluation. *SIGMETRICS Perform. Eval. Rev.*, 2022, **49**(4), 76–80. doi:10.1145/3543146.3543163

CONTRIBUTORS

Mr Vikas Sood received BTech Degree from REC (NIT), Hamirpur and ME degree from PSG Tech, Coimbatore. Currently pursuing a PhD in the CSE department at DIAT, Pune. He is working at CAIR-DRDO, Bangalore since 2001. His areas of interest include System Sciences, Cyber-Physical Systems, Command and Control Systems, Distributed Computing, Modeling, and Simulation of complex systems.
He has conceptualized, implemented the Renew plugin, developed the complete approach, and written the manuscript.

Dr. Malay Kumar Nema received BE degree and ME degree from GEC(JEC), Jabalpur. He received PhD degree from IIT, Bombay and currently working at CAIR-DRDO, Bangalore. His areas of interest include Multi-sensor data fusion, Signal Processing, Applied Machine learning, Progressive image and video formats, Software Development Life Cycle.
He has provided the mapping to represent distributed cyber-physical systems using the four elementary constructs as a basis set.

Dr Rituraj Kumar received his master's from IISc Bangalore and PhD from IIT Delhi. Currently, he is working at CAIR-DRDO, Bangalore since 1994. His areas of interest include Net-Centric systems, Survivable Network Analysis, Critical Infrastructure Protection, and End Point Security.
He has provided guidance in theoretical aspects and overall design and presentation of the proposed methodology.

Dr Manisha Jitendra Nene received a PhD degree in computer science and engineering from the Defence Institute of Advanced Technology (DIAT), DU. She is currently the Head and Director Incharge of the School of Computer Engineering and Mathematical Sciences (SoCE&MS), DIAT. Her areas of interest are wireless sensor networks, cyber-physical systems, cognitive radio, analysis of algorithms, high-performance computing, and modelling and simulation.
She supervised the overall research work from conceptualization to presentation. She also contributed to the property specification and verification aspects of the presented scenario. She has reviewed and provided valuable guidance in various internal revisions of the manuscript.