I.J. Information Technology and Computer Science, **2014, 10, 63-69**
Published Online September 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2014.10.09

Verification of Generic Ubiquitous Middleware for Smart Home Using Coloured Petri Nets

Madhusudanan. J
Research Scholar, Pondicherry University, Puducherry
E-mail: contactmadhu@gmail.com

Anand. P
PG Scholar, SMVEC, Puducherry
E-mail: sivaananth26@gmail.com

Hariharan. S
PG Scholar, SMVEC, Puducherry
Email: hari1691@gmail.com

Dr. **V. Prasanna Venkatesan**
Associate Professor, Pondicherry University, Puducherry
Email: prasanna_v@yahoo.com

Abstract— Smart home is a relatively new technology, where we applied pervasive computing in all the aspects, so as to make our jobs or things that we normally do in-side the home in a very easier way. Originally, a smart home technology was used to control environmental systems such as lighting and heating; but recently the use of smart technology has been developed so that almost any electrical component within the home can be included in the system. Usually in pervasive computing, a middleware is developed to provide interaction between the user and device. In previous, a middleware is only suitable for specific Smart Home architecture, that can't be applicable to any other architecture but the Generic Ubiquitous Middleware is suitable for different Smart Home architecture. This paper proposes that any smart home can be built with single architecture and it is verified using a Coloured Petri Nets tool. We have given a verification model of various Smart home Environments.

Index Terms— Smart Home, Coloured Petri Nets, Context.

I. Introduction

A vision of pervasive computing [1] is to enable computer-based services to be made available everywhere (Ubiquitous) and also to support intuitive human usage. But yet, appear to be invisible to the user. It is also referred to as Ubiquitous computing.

The pervasive computing plays an important role in Smart Home Technology. Smart home technology was used to control environmental systems such as lighting and heating; but recently the use of smart home technology has been developed so that almost any electrical component within the home can be included in the system.

In Smart Home, users interact with devices using pervasive middleware. Several Middleware were developed for the interaction among human and devices like Context-aware middleware [2], Interplatform Service Oriented middleware, and Sensor Fusion Based middleware. However, these middleware were only applicable to certain smart home environment.

So, Generic Ubiquitous Middleware (GUM) was developed to show how interaction takes place among human and devices. This GUM middleware can be applicable to any kind of smart home environments. This paper shows the verification of various smart home environments using GUM with the help of Coloured Petri Nets (CPN) tool.

Coloured Petri Nets (CP-nets or CPNs) [3] is a Mathematical graphical language. It is mostly used for modelling, editing, simulating the concurrent systems and also used to analyze their properties.

CPN is a kind of discrete modelling language, which combines the capabilities of Petri Nets [4] along with high level programming language capability. CPN supports the extensions with time, colour and hierarchy. CPN is based on standard ML. The major advantages of using CPN tool are Gain insight, analysis and specification.

This paper is organized as follows: Section 2 represents a related work, Section 3 presents Architecture of Generic Ubiquitous Middleware for smart home, Section 4 depicts Need for Verification, Section 5 shows Architecture of CPN for smart home, Section 6 represents Various Smart Home environment, Section 7 depicts Verification of Smart Home using Coloured Petri Net and Section 8 presents final Conclusion.

II. Related Work

Generally, Verification of smart environment [5] is done in several ways. Some of them are formal

verification, another kind of verification is done with the help of tools, verification using case study that is not much efficient. The formal verification method for situation definitions and demonstrate its feasibility and efficiency.

The formal verification mechanism consists of three Algorithms. Some of the drawbacks are Fast verification, verification of dynamic and temporal situation, automated situation fix and uncertainty.

There are various verification and model checking tools for the logics. For the logic LTL some well-known tools are NuSMV [6], SPIN [7], VIS [8] and TRP++ [9]. NuSMV is an extension of the model checking tool SMV [10], which is a software tool for the formal verification of finite state systems.

Unlike SMV, NuSMV provides facility for LTL model checking. Spin is an LTL model checking system, which supports all correctness requirements expressible in LTL, but it can also be used as an efficient on-the-fly verifier for more basic safety and liveness properties. VIS is a symbolic model checker supporting LTL.

VIS is able to synthesize finite state systems and or verify properties of the systems that have been specified hierarchically as a collection of interacting finite state machines. TRP++ is a resolution based theorem proven for LTL. TRP++ is based on resolution method for LTL.

The best-known tools for the logic TCTL are UPPAAL [11] and KRONOS [12]. UPPAAL is an integrated tool environment for modelling, validation and verification of real time systems modelled as networks of timed automata, extended with data types. The tool can be used for the automatic verification of safety and bounded liveness properties of real-time systems.

KRONOS is a tool developed to verify complex real-time systems, where the components of real-time systems are modelled by timed automata and the correctness requirements are expressed in the real-time temporal logic TCTL.

AcPeg tool which accepts descriptions of access control models and evaluates queries. It treats the case that the sets of agents and other resources are finite (this case is decidable).

It is currently not able to deal with non-bounded sets of resources, a problem known to be undesirable. ProVerif was used to evaluate the safety of the environment. It isn't always able to give an answer. It may even introduce false attacks.

PRISM [13] and APMC [14] are tools which can be used to model check PCTL formulae. PRISM is a probabilistic model checker. It is a tool for formal modelling and analysis of systems which exhibit random or probabilistic behaviour.

It supports three types of probabilistic models: discrete-time Markov chains (DTMCs), continuous-time Markov chains (CTMCs) and Markov decision processes (MDPs), plus extensions of these models with costs and rewards. The property specification language is mainly PCTL; however, the tool also supports the temporal logics CSL and LTL.

III. Architecture of Generic Ubiquitous Middleware for Smart Home

The Generic Ubiquitous Middleware [15] is interaction between the environment and devices. Generic Ubiquitous Middleware is suitable to any type of smart home. The architecture of Generic Ubiquitous Middleware is shown in Fig.1. Generic Middleware consists of three aspects they are Infrastructure, device and context aspects. Infrastructure and device used to create environment design. Environment design should incorporate with location, user and devices. Design the location, devices involved and user as per the requirements. In the environment integrate the devices in the corresponding locations.

In context aspect different context are identified and parsed in the environment. After Infrastructure and device aspects designed context aspect helps to identify the environment scenario developed in the previous aspects.

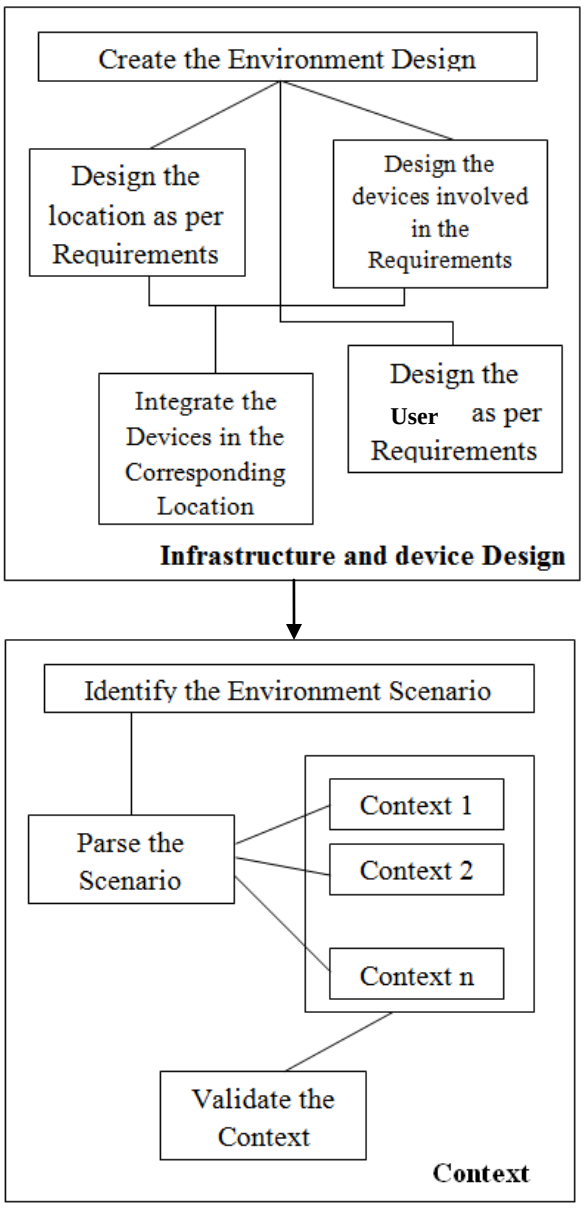

Fig. 1. Architecture of Generic Middleware for smart home

The different context present in the environment is split i.e. parse and validate the context. Different kind of smart homes are present. Generic Middleware architecture is suitable for any number of smart homes. Consider the smart home consists of different devices and different location. Different devices like fan, TV, light, AC, etc., Different locations are hall, bedroom and kitchen. These can be designed in Infrastructure and Device aspects. Context is processed based on the environment created in previous aspects.

IV. Need for Verification

Verification is the process of analyzing the properties, accuracy or validating the smart home environment. Mathematical graphical language [16] should present for analyzing behavior and properties. Statistics must be generated during verification that contains various properties like Liveness properties, Fairness properties, Home properties and Boundness properties.

The verification properties also contain symmetry, fairness and no deadlock. CPN is a mathematical graphical language that process with all the properties mentioned. In addition it process state space analysis. A state space is set of values which the process can take it consists of set of states that a problem can be in. state space is implicit.

V. Architecture of Coloured PetriNets for Smart Home

Coloured Petri Nets [17] is a Mathematical graphical language. It is mostly used for modelling, editing, simulating and to analyze their properties. The architecture of Coloured Petri Net for smart home is shown in Fig.2.

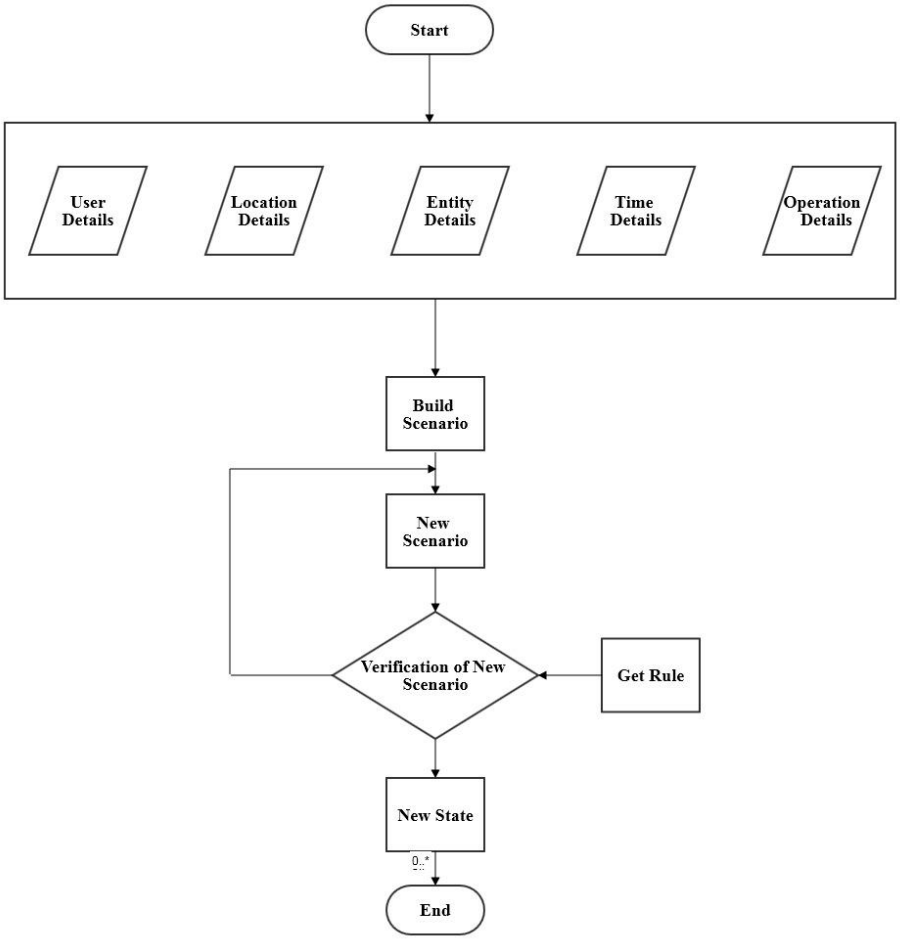

Fig. 2. Architecture of Coloured Petri Nets

The architecture shows the flow of verification is done for the smart home. Initially, a scenario of smart home is built using the details obtained from several contexts like user context, location context, time context, entity context, operation context etc. The new scenario will be given as input. The CPN tool starts verification of scenario for the smart home. It compares the new scenario with the rule already provided.

VI. Modelling of Various Smart Home Environments

Different types of Smart Homes are present based on the different user. The different kind of Smart Home environments was developed using the Generic Ubiquitous Middleware (GUM). GUM [18] is used to

design and process different environments. In this work we have developed three different Smart Home Environments to make verification of the scenario. They are shown in the following cases.

A. Case 1: Smart Home1

Smart Home is a interaction between the devices or interaction between user and devices. Fig.3 shows the model of Smart Home1 environments, which consists of two bed rooms, a guest room and a kitchen. It also consist of devices like Air conditioner, beds, bureaus, doors, fan, lights, TV, washing machine, music players, micro oven, Computer, Sofa, etc. We consider a scenario to execute the interaction between the devices and Users in a Smart Home1 Environment.

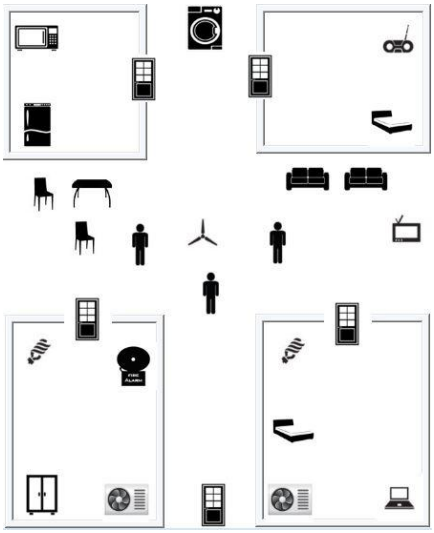

Fig. 3. Model of Smart Home 1

Fig.4 illustrates the executed representation of given scenario for the Smart Home1 environment. User enters into the bedroom at evening AC will be turned ON automatically. Likewise user can move to any location at any time the device will interact with the user according to the user needs.

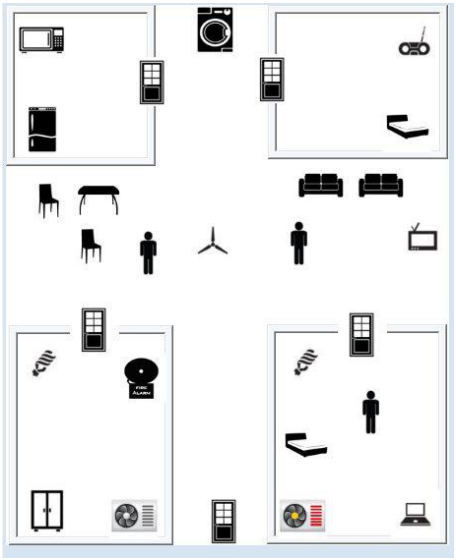

Fig. 4. Execution of a Scenario in Smart Home 1

B. Case 2: Smart Home 2

Smart Home2 consists of different number of user and different location with several devices. Fig.5 shows the model of Smart Home2 environments, which consists of bed room, a dining room and hall. It also consist of devices like doors, fans, lights, TV, music players, computer, chairs, tables, radio etc. We consider a scenario to execute the interaction between the devices and Users in a Smart Home2 Environment.

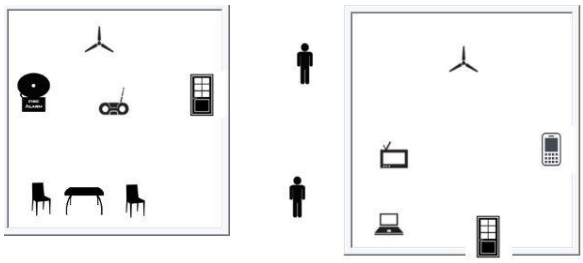

Fig. 5. Model of Smart Home 2

The Fig.6 shows the executed representation of given scenario for the Smart Home2 environment. User enters into the bedroom at evening TV will be turned ON automatically. Likewise user can move to any location at any time the device will interact with the user according to the user needs.

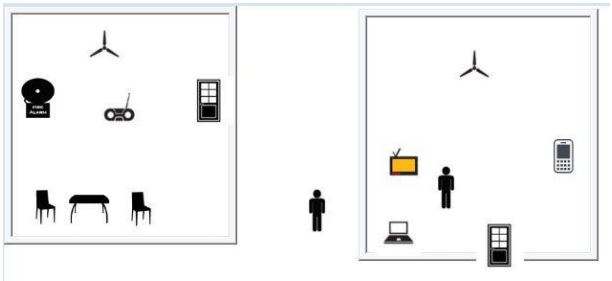

Fig. 6. Execution of a Scenario in Smart Home 2

C. Case 3: Smart Home 3

Smart Home3 consists of different number of user and different location with several devices. Fig.7 shows the architecture of Smart Home3 environments, which consists of bed room, a kitchen and hall. It also consists of devices like doors, lights, TV, micro-oven, chairs, tables, sofa etc. We consider a scenario to execute the interaction between the devices and Users in a Smart Home3 Environment.

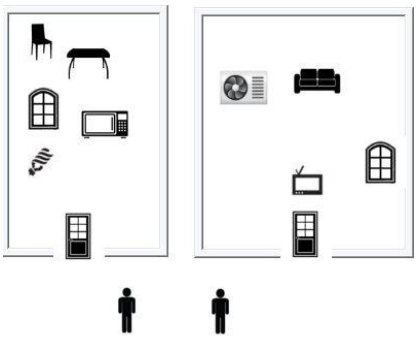

Fig. 7. Model of Smart Home 3

Fig.8 shows the executed representation of given scenario for the Smart Home3 environment. User enters into the bedroom at Morning TV will be turned ON automatically. Likewise user can move to any location at any time the device will interact with the user according to the user needs.

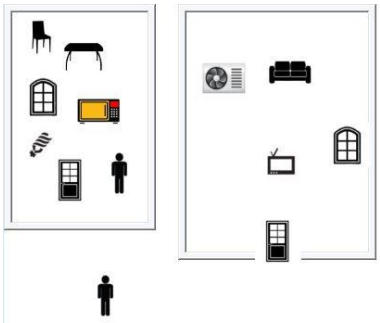

Fig. 8. Execution of a Scenario in Smart Home 3

VII. Verification of Various Smart Home Environment Using CPN

In this section, a verification of three different smart home environments built using GUM is represented in the following cases.

A. Case 1: Smart Home1

This model consists of scenario builder used to build scenario with combination of location, user, entity, operation and time. The built scenario is check with the rule file and forms a new state. The verified result is store in the new state. In smart home1 environment there are three user, three bed rooms and a kitchen incorporate with several devices. Device should interact with user and provides what users want. User enters into bed room at evening then AC will be turned ON likewise device must interact with user at several session that is verify through CPN model that is shown in Fig.9.

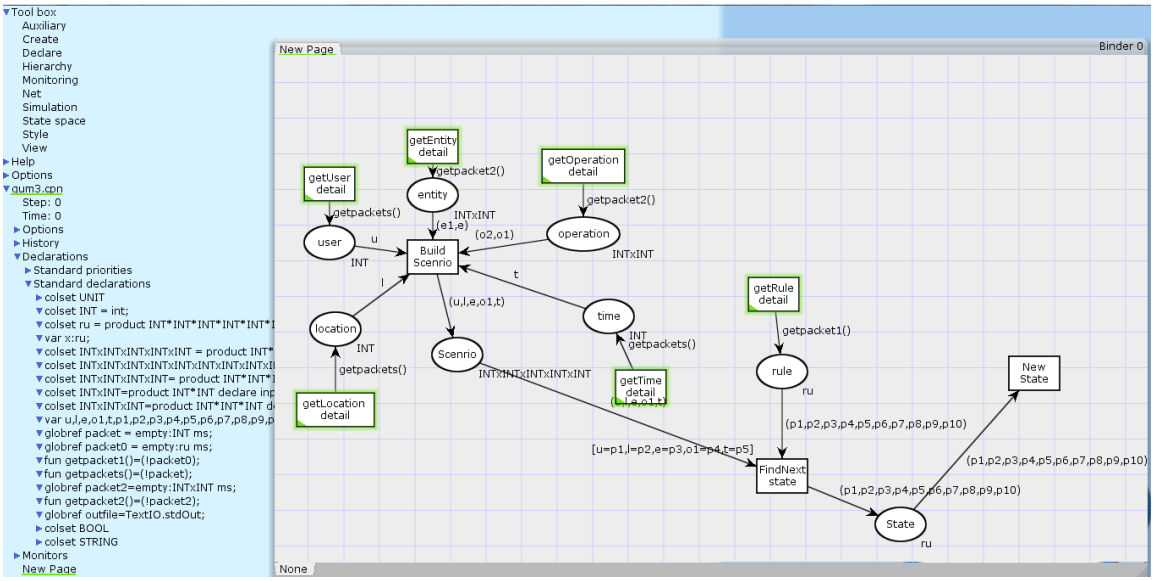

Fig. 9. CPN Model

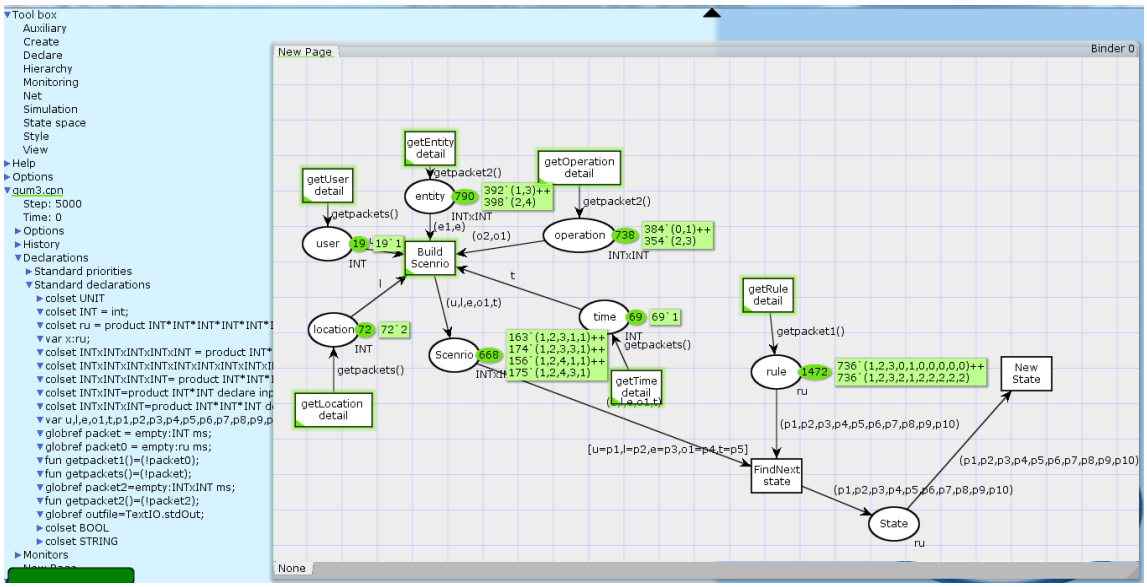

Fig. 10. Verified Model of Smart Home

Table 1 shows the scenario built for the Smart Home1 environment and their numerical representations and the table 2 shows the rule that are given as the input for the Petri Net model and the result. From comparing the above two tables, we verify that no infinite sequence occurs. Verified model is shown in Fig.10.

Table 1. Scenario and Numerical Representation of Smart Home 1

Scenario for Smart Home1	Numerical Representation
If Anand is \Entering/ in (Hall) at the <Morning> TV turn ON	(111121)
If Hari is \Sleeping/ in (Room) at the <Evening> AC turn ON	(222241)
If Guest is \Entering/ in (GuestRoom) at the <Evening> Light turn ON	(314231)

Table 2. Rule and Result for Smart Home 1

Rule for Smart Home1	Result
(111121)	121
(222241)	241
(314231)	431

In this model, verification is done based on the numerical representation. The input files give in numerical representation. In the environment each device, location and user have the numerical representation. Rule files also process as numerical representation. The scenario checks with the rule file and verified result stored in new state that can be represented as final result.

B. Case 2: Smart Home 2

Smart Home2 consists of two users and three locations. Several devices are present in these three locations. A user enters into the bed room at evening then TV will turn ON this can check with rule file apply to the model. In smart home2, this work verifies with three different scenarios and rule file, the verified scenario is stored as a result. Smart Home2 is verified similar to the verification of Smart Home1.

Table 3. Scenario and Numerical Representation of Smart Home 2

Scenario for Smart Home2	Numerical Representation
If owner is \Sleeping/ in (Hall) at the <Evening> Fan turn ON	(121211)
If Guest is \Entering/ in (Room) at the <Evening> TV turn ON	(212221)
If owner is \Entering/ in (Kitchen) at the <Morning> Light turn ON	(113131)

The table 3 shows the scenario built for the Smart Home2 environment and their numerical representations and the table 4 shows the rule that are given as the input for the Petri Net model and the result. From comparing the above two tables, we verify that no infinite sequence occurs. Verified model is shown in Fig.10.

Table 4. Rule and Result for Smart Home 2

Rule for Smart Home2	Result
121212	111
212222	221
113132	331

C. Case 3: Smart Home 3

Smart Home3 consists of two users and three locations. Several devices are present in these three locations. A user entering into hall at evening then Light will turn ON this can check with rule file apply to the model shown in diagram. In smart home3 this work verify with three different scenarios and rule file the verified scenario is stored as a result. Smart Home3 is also verified similar to verification of Smart Home1.

Table 5. Scenario and Numerical Representation of Smart Home 3

Scenario for Smart Home3	Numerical Representation
If owner is \Sleeping/ in (Room) at the <Morning> TV turn OFF	(122122)
If Guest is \Entering/ in (Hall) at the <Evening> Light turn ON	(211231)
If owner is \Entering/ in (Kitchen) at the <Morning> Oven turn ON	(113111)

Table 6. Rule and Result for Smart Home 3

Rule for Smart Home3	Result
122121	222
211232	131
113112	311

The table 5 shows the scenario built for the Smart Home3 environment and their numerical representations and the table 6 shows the rule that are given as the input for the Petri Net model and the result. From comparing the above two tables, we verify that no infinite sequence occurs. Verified model is shown in Fig. 10.

The state space report is generated for these three different cases of smart home. After creating state space report it is found that no infinity sequence occurs during the state space analysis and it also shows there is no deadlock. It is shown in the following table 7.

Table 7. State Space analysis of Generic Ubiquitous Middleware for Various Smart Home Environment

	Smart Home1	Smart Home2	Smart Home3
State Space Node	49631	32146	33146
SCC Node	49631	32146	33146
Fairness Property	No infinite Sequence	No infinite Sequence	No infinite Sequence
Deadlock	No	No	No

VIII. Conclusion

In this paper, Coloured Petri Net tool is used to verify Generic Ubiquitous middleware for smart homes. This work considers three different smart homes as three different cases. The proposed architecture is applicable to different smart home environments. The three cases of smart home is applied to the proposed architecture and verified using the input scenario. This work gives the results for verification of three different smart homes. Our future work focuses on verification of Generic Ubiquitous Middleware for various environments like Smart Bank, Smart Hospital, etc.,

References

[1] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges", To appear in IEEE Personal Communications, 2001.

[2] Kristian Ellebæk Kjær, "A Survey Of Context-Aware Middleware", IEEE, 2010.

[3] K. Jensen, L.M. Kristensen, "Formal Definition of Non-hierarchical Coloured Petri Nets", springer 2009.

[4] Karima Mahdi, Raida Elmansouri, Allaoua Chaoui, "On Transforming Business Patterns to Labeled Petri Nets Using Graph Grammars", IJITCS 2013.

[5] Kurt Jensen, Lars Michael Kristensen, LisaWells "Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems", springer 2007.

[6] A. Cimatti, E. Clarke, F. Giunchiglia, M. Roveri. NuSMV: A New Symbolic Model Verifier. In Proceedings of International Conference on Computer-Aided Verification (CAV'99). Pp. 495–499. 1999.

[7] G. J. Holzmann. The Spin Model Checker. Addison-Wesley, 2003.

[8] T. V. Group. VIS: A System for Verification and Synthesis. In Proceedings of the 8th International Conference on Computer Aided Verification. Pp. 428–432. 1996.

[9] U. Hustadt, B. Konev. TRP++ 2.0: A Temporal Resolution Prover. In Proceedings of Conference on Automated Deduction (CADE-19). Pp. 274–278. 2003.

[10] K. L. McMillan. Symbolic Model Checking. Kluwer Academic Publishing, 1993.

[11] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, W. Yi. UPPAAL — a Tool Suite for Automatic Verification of Real–Time Systems. In Proceedings of Workshop on Verification and Control of Hybrid Systems III. Lecture Notes in Computer Science 1066, pp. 232–243. Springer Verlag, 1995.

[12] M. Bozga, C. Daws, O. Maler, A. Olivero, StavrosTripakis, S. Yovine. Kronos: A Model-Checking Tool for Real-Time Systems. In CAV '98: Proceedings of the 10th International Conference on Computer Aided Verification. Pp. 546–550. Springer Verlag, 1998.

[13] A. Hinton, M. Kwiatkowska, G. Norman, D. Parker. PRISM: A Tool for Automatic Verification of Probabilistic Systems. In Proc. 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06). Lecture Notes in Computer Science 3920, pp. 441–444. Springer, 2006.

[14] T. Hérault, R. Lassaigne, F. Magniette, S. Peyronnet. Approximate Probabilistic Model Checking. In Proc. 5th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'04). Lecture Notes in Computer Science 2937, pp. 307–329. Springer, 2004.

[15] A. Kalaiselvi, V. Indhumathy, J. Madhusudanan, V. Prasanna Venkatesan, "Implementation of Generic Context Middleware for Context-Aware Applications", International Journal of Engineering Research & Technology (IJERT), 2012.

[16] Jerome Hugues, Thomas Vergnaud, Laurent Pautet, "On the Formal Verification of Middleware Behavioral Properties", ELSEVIER, 2005.

[17] Azza K. Nabih, Mostafa M. Gomaa, Hossam S. Osman, Gamal M. Aly, "Modeling, Simulation, and Control of Smart Homes Using Petri Nets", International Journal of Smart Home, 2011.

[18] V. Indhumathy, A. Kalaiselvi, J. Madhusudanan, V. Prasanna Venkatesan, "A Generic Simulation Tool for Pervasive Devices Application", International Journal of Engineering Research & Technology (IJERT), 2012.

Authors' Profiles

Mr.J.Madhusudanan, Associate Professor, Department of Computer Science and Engineering in Sri Manakula Vinayagar Engineering College. He holds M.E in Computer Science and Engineering and pursuing his Ph.D in Banking Technology from Pondicherry University, India. His areas of Interest are Context Aware Computing, Pervasive Middleware, Ubiquitous Computing and Smart Banking. He has many International and National Journal Publications.

Mr. P. Anand. PG Scholar, Sri Manakula Vinayagar Engineering College, Puducherry. He holds B.E in Computer Science and Engineering and pursuing his M.Tech Computer Science and Engineering in Sri Manakula Vinayagar Engineering College, Puducherry. His areas of Interest are Context Aware Computing, Pervasive Middleware, Ubiquitous Computing.

Mr. S. Hariharan. PG Scholar, Sri Manakula Vinayagar Engineering College, Puducherry. He holds B.E in Computer Science and Engineering and pursuing his M.Tech Networking in Sri Manakula Vinayagar Engineering College, Puducherry. His areas of Interest are Computer Networks, Context Aware Computing, Pervasive Middleware, Ubiquitous Computing.

Dr. V. Prasanna Venkatesan, Associate Professor, Dept. of Banking Technology, Pondicherry University, Puducherry, India. He has more than 20 years teaching and research experience in the field of Computer Science and Engineering. His research interest includes software engineering, Business intelligence, Software Architecture and banking technology. He has designed Multilingual Compiler. He has many international Journal publications. He is co–author of the book titled as Service Composition and Orchestration: Concepts and Approaches published by Vdm Verlag Dr. Müller e.K.