International Journal of Hybrid Information Technology
Vol.6, No.5 (2013), pp.25-36
http://dx.doi.org/10.14257/ijhit.2013.6.5.03

An Approach Based on Colored Petri net for Analysing and Modelling the Aspects

K. Santhi and G. Zayaraz

Research Scholar/CSE, Professor/CSE
Pondicherry Engineering College, Puducherry, India

santhikrishnan@pec.edu, gzayaraz@pec.edu

Abstract

A software system is built by a number of concerns where a concern is composed by one or several functional requirements or non-functional requirements or both. In the software system development one of the essential principles is separation of concerns. Generally in the software system, separation of concerns is the process of breaking into many modules with meager overlapping each other. On account of that there may be few specific concerns which are not to be located into a single module which are said to be crosscutting concerns. Aspect-oriented programming (AOP) provides techniques for managing cross-cutting concerns into a single manageable module called, Aspect. AOP is used to solve many problems such as tangling and scattering representations. But the identification and specification of crosscutting concerns and considering it as aspects is not a trouble-free job. This proposed method uses colored Petri Nets for defining the concerns and its requirements and also for executing the final modeled software system which monitors transitions based on that it will determine the join-points in a desired moment –Before, After, Around and Replace operators by the action of Aspect module. Fan in analysis techniques is used for finding the dominant aspect in the system.

Keywords: *After operator, Around operator, Aspect oriented programming, Before operator, concerns, colored Petri net, crosscutting concern, Fan in analysis*

1. Introduction

At present, the developing software is used to support a model wherein the data and their related methods are defined as independent entities called "objects". Object oriented programming (OOP) paradigm provides a technique of modularization of basic concerns [1]. OOP creates a coupling between core and crosscutting concerns. Some broadly based functionality like authorization, caching, exception management, security, *etc…*, cannot be efficiently compartmentalized by using object oriented programming [2]. Such a broadly based concern are defined as crosscutting concerns since they affect the whole program and the crosscutting concerns should be centralized as a single location wherever it is possible [2]. Modularization is not a one-size-fits-all strategy scenario.

AOP is a complementary programming paradigm of OOP which aims to boost up the modularity by allowing the partition of cross-cutting concerns [3]. It accomplishments to tackle the flaw of OOP while performing the cross-cutting concerns. On the other hand, due to the requirement of patterns for addressing the cross-cutting concerns of a software model, AOP is still in an immature state [13, 21, 23]. In AOP the crosscutting concerns are implemented as aspects instead of fusing them into the core modules.

ISSN: 1738-9968 IJHIT

Many suitable methods which are used for identifying the aspects are Prune dependency rule, fan-in-analysis, Theme/Doc, Line co- change, concernMapper, Formal concept analysis, event traces and clone detection [1]. These methods are used to detect the crosscutting concerns in the various levels of software development process and such detection at initial stage is cheaper, faster and more advantageous than doing the modifications later on-the-fly [12, 19, 22, 23].

Rashid [9] presented ARCaDe tools for modularization and composition of crosscutting concerns in Aspect Oriented Requirement Engineering level itself. His work is accomplished by considering the Viewpoints model [10], used to provide the incorporation of mixed requirements which are specified from multiple viewpoint. Vahdat Abdelzad [8] used petrinets for identification of aspects with the drawbacks that join point or match point could not be determined in the situation of imposing aspects like after, before, around and replace operators to logical entities.

In this paper, we propose a method which extends the work have been done in [8] that permit the user at requirement phase to detect the interactions between crosscutting concerns by using colored Petri net. The operators like Before, After, Around and Replace are used to generate composition specifications for every match point which can be used for guiding the process of developing composition rules at requirement phase itself

2. Background

Petri net is first introduced by Carl Adam Petri in 1962 in his PhD [4]. Petri net is the graphical and mathematical modeling tool. It is a directed bipartite graph consists of places, transitions and tokens are represented by circles, bars and bullets respectively. It is used for the description of distributed system, concurrent execution of any system. Petri nets are useful tool in modeling and analysis of systems [4, 5, 7, 8, 12].

A *net* is a tuple $N = (P, T, A, \Sigma, C, N, E, G, I)$ [5] where:

- P stands for set of *places*.
- T stands for set of *transitions*.
- A stands for set of *arcs*

In CPNs sets of places, transitions and arcs are mutually pair wise disjoint $P \cap T = P \cap A = T \cap A = \emptyset$

- Σ is a set of color sets contains all possible colors, operations and functions used within CPN.
- C is a color function which maps places in P into colors in Σ.
- N is a node which maps A into $P \times T \cup T \times P$.
- E is an arc expression function which maps each arc $a \in A$ into the expression e.
- G is a guard function which is defined from T into expressions E
- I is an initialization function which is defined from P to closed expressions.

The structure of Petri net which is shown in Figure1 consists of set of nodes belong to two different classes and the edges to connect only nodes of different classes.The state is

represented by the distribution of tokens over places (also referred to as marking). Rules to be considered for constructing the Petri net graph, i) Connections are directed ii) No connections between two places or two transitions iii) Places may hold zero or more tokens.

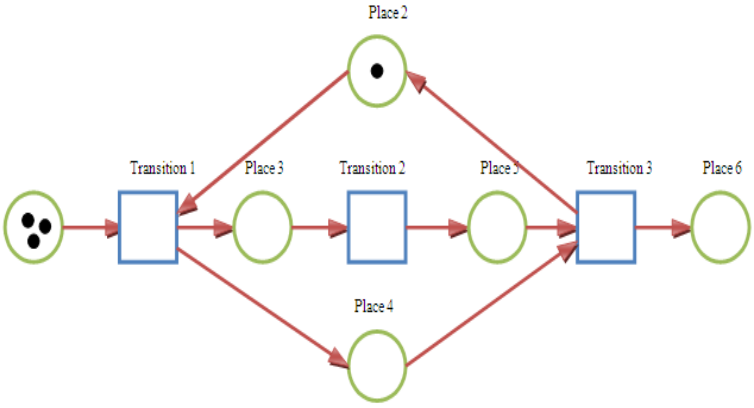

Figure 1. Petri net graph

A Petri net produces dynamic information using the Enabling rule and Firing rule,

- An enabled transition is marked with at least w (p, t) tokens, where w (p, t) is the weight of the arc from p to t.
- It may or may not fire (depending on whether or not the event actually takes place).
- A firing of an enabled transition t removes w(p,t) tokens from each input place p of t and adds w(t,p) tokens to each output place p of t, where w(t,p) is the weight of the arc from t to p

Enabled transition:

A transition is enabled if each of its input places contains at least one token

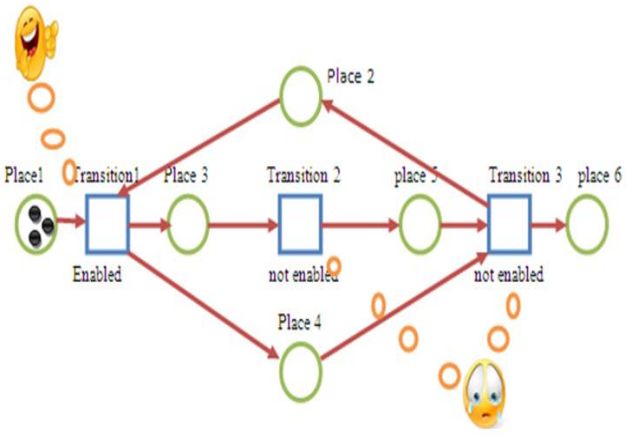

Figure 2. Enabled transition Petri net

Firing transition:

- An enabled transition can fire (*i.e.*, it occurs).
- When it fires it consumes a token from each input place and produces a token for each output place.

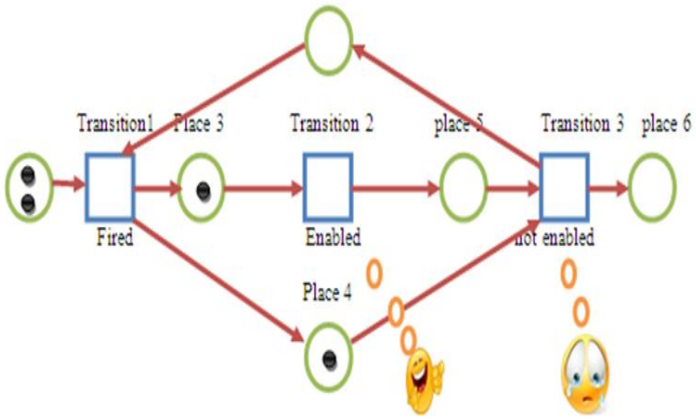

Figure 3. Fired transition Petri net

Multiple transitions may be enabled, but only one can fire at a time being non-deterministic.

3. Properties of Petri Nets

A. Liveness

A transition is said to be live if it is potentially fire able in any marking of R (M1) [7, 11]. A transition is dead in M if it is not potentially fire able; if the PN enters marking M the dead transition cannot fire any more.

B. Safeness

A place is safe if the token count does not exceed 1 in any marking of R (M1). A PN is safe if each place is safe.

C. Boundedness

A simple generalization of safeness is the concept of boundedness. A place is bounded with bound k, if the token count does not exceed k in any marking of R (M1). A PN is k-bounded if each place is k-bounded [7, 8].

D. Reachability

A marking M_n is said to be reachable from a marking M_0 if there exists a sequence of firings that transforms M_0 to M_n

4. Procedure used for identifying aspects and the most dominating Aspects

To identify the aspects these steps are to be followed and satisfied respectively.

Step 1

Prescribing the system through concerns is an essential thing for the proposed system. Since, from the Lexical analysis of the system's text, concerns are obtained

Step 2

Each and every concern which is associated with the requirements should be specified, since the requirements are obtained through any traditional requirement engineering approach. By this method, each and every requirement is taken as an independent Petri Net.

Step 3

For each requirement define a requirement net and for a requirement net find logical entities. The following defines requirement net as

A requirement net is a 2-tuple RN= (PN, LE) [8]

Where,

PN is a Petri Net which is following: |P|=2, |T|=1, |F|=2.

LE= (O_1, O_2 . . . O_n), is a set of logical entities.

Step 4

As the requirement engineers with analyzing purpose of a concern and associated requirements may elicit the execution orders. So the execution orders are specified for each concern in order to constitute the concern net.

A concern net is a 2-tuple CN= (SoR, SoE) [8]

Where,

SoR= (RN_1, RN_2 . . . RN_n) (n>0), it is a finite set of requirement nets.

SoE= (EO_1, EO_2. . . EO_n) (n>0), it is a finite set of execution orders.

An execution order is a sequence of requirement nets which present the following,

EO= (RN_1, RN_2 . . . RN_n).

Step 5

Based on the definitions defined in the previous steps the concern net for each concern is obtained. For constituting concern nets, get the requirements nets and execution order. The execution of each concern net implicates that the proper token is placed in the first place of concern net. If there is not enough token in first place then concern net cannot be executed correctly in the final Petri Net model. Due to this it is impossible to identify the aspects in the system.

Step 6

In the requirement nets and concern nets identify the dependencies, restrictions and relationships among them. Produce a new place as temporary place when dependency occurs between two requirement nets and a token of dependency is placed in it, The complete execution of the system is caused by the names of token net and requirement net.

Step 7

After specifying the transitions of each concern net having two or more entrances, if value of entrance tokens is distinct then both entrance token and transition token are taken as token1, token2.

Step 8

Determine the logical entities related with requirement nets. The logical entity has the tangling problem arise in logical entities when they are belong to two requirement nets having 2-tuple (token1, token2).

Step 9

When the identification of crosscutting concerns is made, the match points are obtained by considering both the functional and non functional requirements of the system. Generally in the match point both the functional crosscutting concerns and non-functional cross cutting will meet each other.

Step 10

In a match point the execution of a program occurs in which aspects are associated with each other. Now the aspect takes an action in this match point in a desired moment like Before, After, Around and Modify.

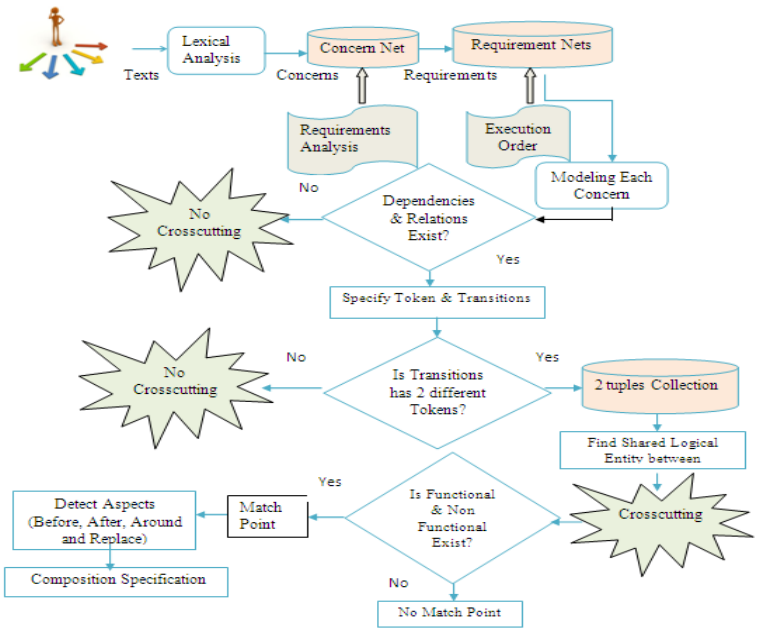

Figure 4. Aspect analyzer using coloured Petri net

Algorithm 1: Identify Crosscutting & Match Point

while (! EOF (text)**) do**
Generate Lexemes //Do Lexical Analysis
end while
while (! EOF (concern)**) do**
Perform Requirement Analysis
For each requirement
{
Construct requirement Net (RN) = (PN, CE) //PN pertinent
Construct **(**LE= (O_1, O2, O3……., On)**)** //LE logical entity
}
end while
Construct Concern Net & Execution Order
if (Dependency && Relations**) then**
Specify Tokens and Transition T
else
Declare No crosscutting exist in T
end if
if (count(Deg^- (T)>1**) then**
{
T *is designated* as Crosscutting
if (count (Candidate Aspects (T) > =1)**) then**
T *is designated* as Match Point
//Generate Composition Specification
for each Candidate Aspect
if (CA(i) > T**) then** T→ CA(i) //Before Operator
else if (CA(i) < T**) then** T ←CA(i) //After Operator
else if (CA(i)|| T**) then** T =>CA(i) //Around Operator
else CA(i) ⋯⋯> T //Modify Operator
end if
end for
end if
end if

Case study

In order to demonstrate how our approach works in practice, this section presents a systematic case study of modeling and analysis of Movie Theatre Chain System. Integrated on-line tickets can be reserved, paid, bought and cancelled through on-line itself with sales facility. Main facilities considered by this system are on-line user registration, ticket booking, payment, cancellation on request and transaction statistics. To perform these processes first the consult of portal takes place which checks for the Availability and accessibility of the system. In on-line user registration where the user and system exchange the data between them which takes place parallel to access control. In the on-line ticket booking the data exchange and access control are performed then the ticket payment is done by data exchange and access control with security. On-line tickets booking and payment requires efficiency

(response-time), precision and security; whereas consult of portal requires availability and user registration needs security.

Now we must look for imposing the aspects with functional and non- functional concerns and also we should identify the match points where these two concerns will meet each other. Finally, the composition specification will be derived.

The concerns considered for this application are,
Concern C1: User Login
Concern C2: Reserve Ticket
Concern C3: Reservation Confirmation
Concern C4: Reservation cancellation
Concern C5: Logging

Table 1. Concern nets and requirement nets

S. No	Concern	Requirements
1.	User Login	1.User Registration(R11) 2. Accessibility(R_{12})
2.	Reserve Ticket	1. check ticket availability(R21) 2. Ticket Reservation(R22)
3.	Reservation Confirmation	1. Payment Process(R31) 2. Display report(R32)
4.	Reservation Cancellation	1.verify and update the reservation (R41) 2.Refund payment(R42) 3. Access Registration(R43)
5.	Logging	1.log the file(R51) 2. Save log file(R52)

Table 2. Execution of concern nets

S. No	Concerns	Execution order
1.	User Login	RN_{11},RN_{12}
2.	Reserve Ticket	RN_{21},RN_{22},RN_{23}
3.	Reservation Confirmation	RN_{31},RN_{32},RN_{33}
4.	Reservation Cancellation	RN_{41},RN_{42}
5.	Logging	R_{51},R_{52}

By identifying the requirement nets for each concern the execution order for the concerns are derived. To enable the firing transition, the respective tokens and color set are given to each place which is represented by circles in Figure 5. By giving the tokens the transition will be enabled and fired and then the other transitions will gets enabled by receiving the tokens. Identify the dependencies between the concern nets and requirements nets; if it exists temporary place is created with token.

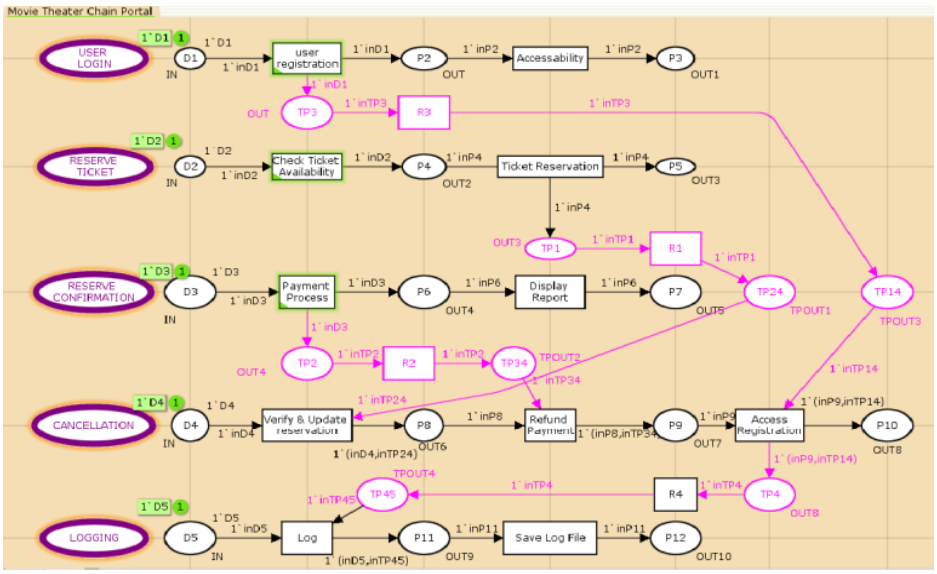

Figure 5. Identification of functional crosscutting concerns

The result is monitored and analyzed by colored Petri Nets, the temporary places which are colored represent that the dependency has exist between the concern nets and their respective requirements. By considering the functional concerns, the cross cutting concerns which are identified in the Movie theatre Chain System are Ticket booking, Ticket payment and User Registration [14, 17]. Both the customers and administrators are needed to check the ticket availability in order to make the reservation. If the availability of ticket is there then the ticket is allocated and reserved after that only the payment for the ticket will be remitted. These are separated and modulated as aspects with advices and point cuts.

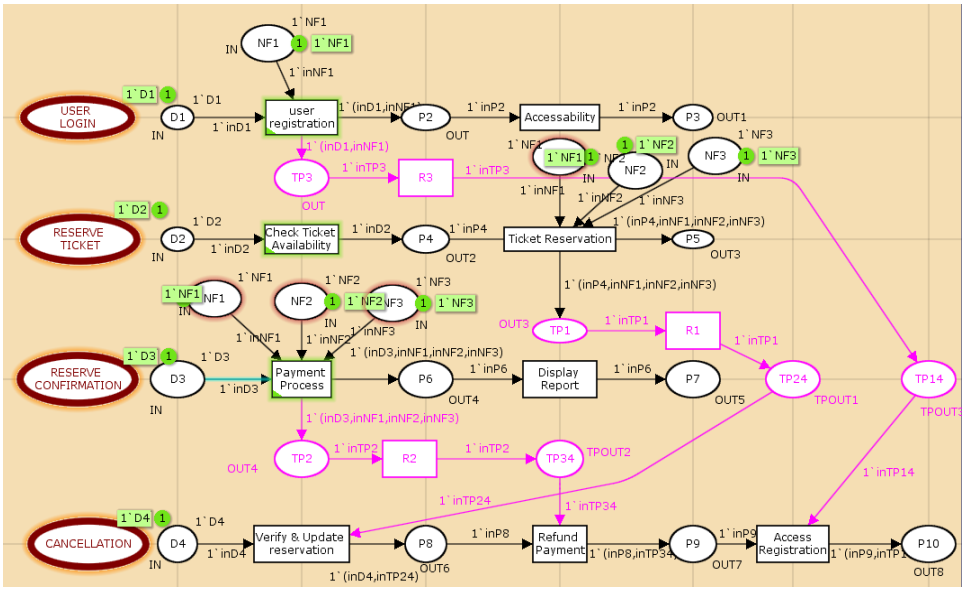

Figure 6. Match point identification

By considering the nonfunctional concerns like security, availability, precision and response time along with its functional concerns the match point exist [14, 21]. With this both functional and non-functional concerns the match point which is identified in the system are User Registration, Ticket Reservation and Ticket payment as in Figure 6.

By considering the operators such as Before, After and Around, the composition specification for the match point Ticket payment is given below,

- Response time(RT) around Ticket payment(TP) TP =>RT
- Security .Availability(S.AV) before Ticket payment(TP) TP →S.AV
- Precision after Ticket payment(TP) Precision → TP
- Security .Accuracy (S.AC) around Ticket payment(TP) TP=> S.AC
- Ticket Reservation (TR) before Ticket Payment TP→TR

In the similar way the composition specification for the other two match points will be identified.

Table 3. Identification of scattering and tangling

	Fan In & Fan out Analysis	
Functional Concerns	Tangling	Scattering
User Registration	2	2
Ticket Payment	4	2
Ticket Reservation	4	2

Among the crosscutting concerns the most tangled concerns are Ticket Payment and Ticket Reservation. All the functional concerns of the system are equally scattered to each other. Structural complexity which deals with association of functional concerns and can be calculated by using Henry-Kafura formula [14],

$$C_T = \sum_{i=1}^{n} C_{pi} \text{ where } \quad Cp = (\text{fan-in X fan-out})^2$$

Now, for our case study,

Cp for user registration is = $(2 * 2)^2 = 9$.

Cp for Ticket Payment is = $(4 * 2)^2 = 64$

Cp for Ticket Reservation is = $(4 * 2)^2 = 64$

C_T for the entire case study is = 125.

As high fan-in (tangling) shows better design structure in which module has been used heavily which shows re-usability of module, hence reduces redundancy in code. High fan out (scattering) shows poor design structure in which module depends highly on other module, which also increases maintainability cost [20].

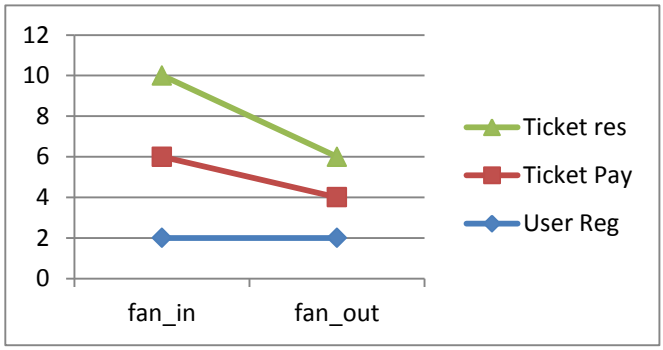

Figure 6. Representation of scattering and tangling

5. Conclusion and future work

The concerns and the requirements for the Movie Theatre Chain portal System and their dependencies are identified. By this the crosscutting concerns is derived and the imposing aspects like Before, After, Around and Replace for match point has been executed by using the colored Petri net. The research work in future can be forwarded in the direction to apply the same case study to generate the composition rules from the composition specifications which can be used for guiding the process of composition at requirement stage itself.

References

[1] A. Kaur and K. Johari, "Identification of Crosscutting Concerns: A Survey", International Journal of Engineering Science and Technology, vol. 1, no. 3, (**2009**), pp. 166-172.
[2] S. A. Vidal and C. A. Marcos, "Toward automated refactoring of crosscutting concerns into aspects", The Journal of Systems and Software, (**2013**), pp. 1482– 1497, http://dx.doi.org/10.1016/j.jss.2012.12.045.
[3] A. Rashid, P. Sawyer, A. Moreira and J. Araujo, "Early Aspects: A model for aspect oriented requirements engineering", In proceedings of the IEEE joint International conference on requirements Engineering *(RES02)*, IEEE, **(2002)**, pp. 199-202, http://dx.doi.org/10.1109/ICRE.2002.1048526.
[4] L. Guan, X. Li and H. Hu, "A Petri Net-Based Approach for Supporting Aspect-Oriented Modeling" 2nd IFIP/IEEE International Symposium on Theoretical Aspects of Software Eng'g., TASE, (**2008**), pp. 83-90.
[5] K. Jensen, "A brief introduction to coloured petri nets", Tools and Algorithms for the Construction and Analysis of Systems", Springer Berlin Heidelberg, vol. 1217, (**1997**), pp. 203-208.
[6] L. Guan, X. Li, H. Hu and J. Lu, "A Petri net-based approach for supporting aspect-oriented modeling", Frontiers of Computer Science in China, vol. 2, issue 4, (**2008**), pp. 413-423, http://dx.doi.org/10.1007/s11704-008-0041-8.
[7] T. Murata, "Petri Nets; Properties, Analysis and Applications", Proceeding of the IEEE, vol. 77, no. 4, (**1989**), pp. 541-580.
[8] V. Abdelzad and F. S. Aliee, "A Method Based on Petri Nets for Identification of Aspects", Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on Early Aspects, R. Chitchyan, S. Zsachaler (Eds.), vol. 2, no. 1, **(2010)**, pp. 43-49.
[9] R. Awais, A. Moreira and J. Araújo, "Modularisation and composition of aspectual requirements", Proceedings of the 2nd international conference on Aspect-oriented software development. ACM, (**2003**), pp. 11-20, http://dx.doi.org/10.1145/643603.643605.
[10] J. Lee, K. -H. Hsu, S. -J. Lee and W. -T. Lee, "Discovering Early Aspects through Goals Interactions", Software Engineering Conference (APSEC), 2012 19th Asia-Pacific, vol. 1, (**2012**), pp. 97-106, http://dx.doi.org/10.1109/APSEC.2012.64.

[11] C. He and C. Tu, "GPRN: A Hierarchical Framework for Aspect-oriented Requirement Modeling", Int. J. Digital Content Technology and its Applications, vol. 5, no. 2, **(2011)**, pp. 165-172.
[12] L. Kong and T. Yuan, "Use case modeling approach for early aspect acquisition", ACM SIGSOFT Software Engineering Notes, vol. 34, no. 4, **(2009)**, pp. 1-6.
[13] K. Renganathan and V. Bhaskar, "Modeling, analysis and performance evaluation for fault diagnosis and Fault Tolerant Control in bottle-filling plant modeled using Hybrid Petri nets", Applied Mathematical Modelling, vol. 37, Issue 7, (**2013**), pp. 4842–4859, http://dx.doi.org/10.1016/j.apm.2012.07.059.
[14] S. Cech, "UML Extensions for Aspect Oriented Software Development", in Journal of Object Technology, vol. 8, no. 5, (**2009**), pp. 85-104, http://www.jot.fm/issues/issues 2009 04/.
[15] S. Henry and D. Kafura, "Software Structure Metrics Based on Information Flow", IEEE Transactions on Software Engineering, vol. SE-7, no. 5, **(1981)**, pp. 510-518, http://dx.doi.org/ 10.1109/TSE.1981.231113.
[16] I. Somerville, "Software Engineering", Seventh edition, Addison-Wesley, **(2005)**.
[17] R. Chitchyan, A. Rashid, P. Rayson and R. Waters, "Semantics-based composition for aspect-oriented requirements engineering", Proceedings of the 6th international conference on Aspect-oriented software development (AOSD'07), ACM, (**2007)**, pp. 36-48, http://dx.doi.org/10.1145/1218563.1218569.
[18] E. W. Dijkstra, "A Discipline of Programming", Englewood Cliffs, NJ Prentice Hall, **(1976)**.
[19] I. Krechetov, B. Tekinerdogan, A. Garcia, C. Chavez and U. Kulesza, "Towards an integrated aspect-oriented modeling approach for software architecture design", In 8th Workshop on Aspect-Oriented Modelling (AOM. 06), AOSD, vol. 6, (**2006**), http://dx.doi.org/10.1.1.64.3597.
[20] X. Yu and D. A. Lamb, "Metrics applicable to software design", Annals of Software Engineering, vol. 1, issue 1, (**1995**), pp. 23-41, http://dx.doi.org/10.1007/BF02249044.
[21] D. Mairiza and D. Zowghi, "Constructing a catalogue of conflicts among non-functional requirements", Evaluation of Novel Approaches to Software Engineering, Communications in Computer and Information Science, Springer-Verlag Berlin Heidelberg, 2011, vol. 230, (**2011**), pp. 31-44, http://dx.doi.org/10.1007/978-3-642-23391-3_3.
[22] A. R. Oliveira, J. Araújo and V. Amaral, "The VisualAORE DSL", 2010 5th Int'l Workshop on Requirements Engineering Visualization (REV), **(2010)**, pp. 11-19, http://dx.doi.org/10.1109/REV.2010.5625665.
[23] J. Herrera, I. Macia, P. Salas, R. Pinho, R. Vargas, A. Garcia, J. Araujo and K. Breitman, "Revealing Crosscutting Concerns in Textual Requirements Documents: An Exploratory Study with Industry Systems", 26th Brazilian Symposium on Software Engineering (SBES), **(2012)**, pp. 111-120, http://dx.doi.org/10.1109/SBES.2012.10.

Authors

K. Santhi

She received Bachelor degree in Computer Science from Madras University and Master degree in computer Science from Bharathidasan University also she earned Master degree in Computer Science and Engineering from Anna University. Her research interest includes Aspect oriented programming, Software Architecture, Algorithm Analysis and Design. Currently she is pursuing Doctoral degree in Pondicherry Engineering college affiliated with Pondicherry University.

Dr. G. Zayaraz

He is currently working as professor in Computer Science & Engineering at Pondicherry Engineering College, Puducherry, India. He received his Bachelor's, Master's and Doctorate degree in Computer Science & Engineering from Pondicherry University. He has published more than fifty research papers in reputed International Journals and Conferences. His areas of specialization include Software Architecture and Information Security. He is a Reviewer/editorial member for several reputed International Journals and Conferences and Life Member of CSI and ISTE.

