Modelling and Verification of Ambient Systems using Petri Nets

Alexandros Konios

School of Computing Science

Newcastle University

A thesis submitted for the degree of

Doctor of Philosophy

June 2015

I would like to dedicate this thesis to my loving wife Martha.
Without her patience, support and true love, this work would not
have been completed.

Acknowledgements

First and foremost I would like to thank my supervisor Marta Pietkie - wicz-Koutny for her constant support and useful guidance throughout my doctoral research. It has been a very constructive, productive and invaluable experience to be a PhD student under her supervision, since she motivated and inspired me to improve my skills on mathematics, formal methods and generally research through her advices and ideas.

Special thanks to the members of my supervisory team, Professor Maciej Koutny and Dr. Jason Steggles, for providing me with interesting ideas and helpful advices about the structuring of my final research topic.

I would like to acknowledge Professor Michael Harrison who helped me to comprehend the basic notions of Ambient Systems at the very beginning of my PhD and gave me suitable guidelines about related works on this cognitive field.

I would also like to thank my family for all their love, encouragement and support in all my pursuits. And most of all, I would like to thank my loving, encouraging and patient wife Martha whose unconditional and faithful support during all these years of my PhD studies is truly appreciated.

This work was funded and supported by the School of Computing Science of Newcastle University.

Abstract

The expeditious development of technology in the past decades resulted in the introduction of concurrent systems that incorporate both ubiquitous and pervasive computing, the ambient systems. These systems are named after their ability to be completely embedded in the environment in which they operate and interact with the users, in a silent and non distracting way, facilitating the completion of their tasks.

Hence, there is a growing need to introduce and develop formal techniques for computational models capable of faithfully modelling the behaviour of these systems. One way of capturing the intricate behaviours of the ambient systems is to use Petri nets, which are a modelling language that is used for the representation and analysis of concurrent systems.

Within the domain of rigorous system design, verification of systems effectively checks and guarantees the correctness of the examined models with respect to the specification.

This work investigates the modelling and the analysis of ambient systems using Petri nets. To examine the modelling of these systems, their taxonomy into Ambient Guidance Systems and Ambient Information Systems is carried out and a case study is used for the modelling of each category.

To model ambient systems, the step-modelling approach and a variant class of Coloured Petri Nets, the Ambient Petri Nets (APNs), are introduced. Step modelling approach focuses on the interaction between the system and the user and Ambient Petri Nets is a class of nets with colour-sensitive inhibitor arcs that is used especially for the structural and behavioural representation of ambient systems. For the modelling of general ambient systems, the compositionality of the Ambient Petri Nets is used.

To verify the correctness of the produced Ambient Petri Nets models, the introduction of the Transformed Ambient Petri Nets class that has no colour-sensitive inhibitor arcs is required since Charlie and generally most of the existing verification tools do not support the analysis of inhibitor nets. To address this problem, a construction is defined to translate the Ambient Petri Nets into Transformed Ambient Petri Nets. Afterwards, the Step Transition Systems are used to prove the behavioural equivalence of the nets that are associated through the construction.

Subsequently, the Transformed Ambient Petri Nets models of the chosen case studies are verified against model checking and qualitative properties. For the first category, Computation Tree Logic (CTL) is used to check the models against important properties of the ambient systems that are related to their features and their general functioning. Finally, qualitative properties consider fundamental structural and behavioural properties of Petri nets that provide useful outcome about the systems under consideration.

Publications

Conference paper:

- A. Konios, ‘Modelling Ambient Systems with Petri Nets’, In Proceedings of 13th International Conference on Application of Concurrency to System Design, PhD session, (ACSD 2013), pages 247 - 251, 2013.

Technical reports:

- A. Konios and M. Pietkiewicz-Koutny, ‘Modelling Ambient Systems with Coloured Petri Nets’, Technical report series CS-TR-1377, School of Computing Science, Newcastle University, 2013.
- A. Konios, ‘Ambient Systems and Taxonomy Approaches’, Technical report series CS-TR-1281, School of Computing Science, Newcastle University, 2011.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There is a growing need to introduce and develop formal techniques for computational models capable of faithfully modelling systems whose behaviour is of high complexity and concurrent. Concurrent systems may vary according to the characteristics and properties that influence or govern their behaviour. Examples of such systems are: reactive systems [25], distributed systems [48], ambient systems [23], etc. In this thesis, we address the modelling and analysis of the ambient systems.

To examine and analyse these intricate systems thoroughly, we model both their structure and behaviour. For that purpose, we use the formal methods [68], which are techniques that are supported by a strong, rigorous and well defined mathematical background. Formal methods are distinguished into several different techniques that can be used for the modelling and verification of systems. Some examples of such methods are VDM [10], π-calculus [60], Petri nets [49], etc.

VDM is a model-based method that is used for the description of sequential systems [52]. The advantage of using VDM is that it provides a comprehensive and precise system specification, which is based on the strong mathematical formalism of that method and that it is supported by several tools for the analysis of the modelled systems. On the contrary, the main disadvantage of VDM lie in the fact that this method does not support the effective modelling of concurrency or timing aspects of systems. Furthermore, VDM produces system specifications that are hard to be interpreted as the models are expressed in form of a textual specification language. Another formalism is π-calculus, which is used for the description of concurrent systems, and specifically for the modelling and analysing properties of concurrent computation [64]. The advantage of π-calculus is its solid, powerful and very expressive formalism that has a quite simple syntax. Furthermore, π-calculus is extended to a range of classes that can be used for the description of the different kinds of concurrent systems. On the other hand, π-

calculus uses a textual specification language to express the concurrent behaviour of the system through processes that interact with each other making the interpretation of the model very difficult, as with VDM. But, the major disadvantage of π-calculus is the lack of verification tools to support the extensive analysis of the models. Finally, Petri nets are a graphical and mathematical modelling language that can be used for the description of several kind of systems like concurrent, distributed or asynchronous systems [7]. Despite their graphical nature, Petri nets are also supported by formal semantics that define the static and dynamic aspects of the models. Moreover, Petri nets have abundance of analysis techniques and tools, which makes them a very powerful options for the modelling and verification of systems.

In this work, the formal method that is used for the modelling and analysis of the ambient systems is Petri nets. As was discussed above, this method provides graphical models that represent the structure and exhibit indirectly both the sequential and concurrent behaviour of the systems under consideration. Additionally, the graphical representation of the models makes the understanding of system specification easier and the observation of the state changes more obvious compared to the other two formalisms. To express the sequential or concurrent behaviour of the Petri net models in a more comprehensible and clearer way, we use the transition systems [46], which illustrate the state space of the models considering all the potential sequential or concurrent events that can occur within the examined systems.

Finally, to explore the correctness of the systems' models, formal verification is applied to these models examining fundamental structural and behavioural properties of Petri nets [56]. In addition, for further investigation of the models, model-checking [17] is used in order to examine specific system properties of the models that are related to the specification of the examined ambient systems.

1.1 Ambient Systems

An engaging and interesting subject of investigation in computing science are the recently introduced ambient systems which were originally developed in the late 1990's, and have already attracted the attention of the researchers. The tremendous advance of technology has contributed to the construction of such systems, which incorporate both ubiquitous and pervasive computing [4, 67] resulting in their rapid proliferation.

The main purpose of introducing and developing the ambient systems was the creation of systems that would interact with the users facilitating their daily lives. The interactivity of ambient systems with the users is expressed through the embedded technology that governs their architecture [6]. The application areas of ambient systems cover a wide spectrum of different domains, such as public or private environments [5]. These environments usually refer to public buildings like hospitals, libraries or universities and private properties like houses or companies respectively.

The need of exhaustive understanding of the ambient systems resulted in the deep and thorough study of the intricate architecture and behaviour of these systems. For that reason, the researchers focused on several different aspects of ambient systems, such as interactivity [26], mobility [14], context-awareness [40], fault-tolerance [30], user's experience [61], etc. All the above fields of investigation derive from the fact that different definitions have been given to the ambient systems throughout the past decades. These definitions may slightly differentiate the ambient systems according to the point of view and the main interest of research of each researcher, but all of them denote that these systems are governed by common key features that are related to their architecture and behaviour [14, 30, 41, 58].

Hence, to examine all the different aspects and features of the ambient systems, the researchers applied several formal methods and verification techniques, which resulted in the development of different approaches for the analysis of the these systems. For example, *Luca Cardelli* and *Andrew Gordon* [14] examine the mobility of the ambient systems. In their approach, they capture notions of locations, mobility and authorisation to move for ambient systems by using π-calculus concepts and techniques. On the other hand, *Iliasov et al.* [30] explore the fault-tolerance of the ambient system aiming to develop and describe an approach that contributes to the easiest way to achieve the fault-tolerance in ambient systems. They demonstrate their approach through a case scenario that is modelled in B-method.

Finally, the approach that is introduced in this thesis for the modelling and the analysis of ambient systems is based on the structural and behavioural representation of both the interactivity and the context-awareness of these systems using Petri nets. For that reason, two case studies of ambient systems are modelled and analysed in the rest of this work.

1.2 Modelling using Petri Nets

Petri nets are a modelling language that is used for the graphical representation and analysis of real concurrent systems [56]. Since their introduction by Carl Adam Petri [49], Petri nets have been widely used in various areas, such as biology, software engineering and concurrent programming (see, e.g., [28, 13]).

Petri nets can be distinguished into two general categories, the low level Petri nets [24] and the high level Petri nets [66]. The low level Petri nets consist of classes like Elementary nets or Condition/Event nets and Place/Transition nets. Low level nets can usually represent the behaviour of systems with low or at most medium complexity. This does not imply that a complex system cannot be modelled with low level nets, but in such a case the complexity of the modelling and the analysis of that system would be higher. On the other hand, high level nets like Coloured Petri Nets, Stochastic Petri nets and Predicate/Transition nets, can model the behaviour of more complex systems. The advantage of high level nets over the low level nets is that they can depict the flow of different data types, temporal specifications, etc. [21, 34].

Another way to distinguish the nets of Petri net classes is in context with the use of non-ordinary arcs, such as the inhibitor arcs. In particular, general Petri nets can be classified into inhibitor nets and non-inhibitor nets, where the former category refers to nets with inhibitor arcs and the latter to nets with no inhibitor arcs. The inhibitor arcs are special arcs that check the firing of net transitions with respect to the absence of resources and are used to enhance the expressiveness of the Petri net classes by both extending the kind of arcs and changing the enabling rules of these Petri net classes [35]. Despite the enriched expressiveness of the inhibitor nets, the introduction of the inhibitor arcs restrains the verification of the produced nets as the reachability of the general Petri nets cannot be generalised for the inhibitor nets due to the lack of monotonicity[1][12].

For the modelling of ambient systems, a new class of coloured inhibitor nets is defined, the Ambient Petri Nets (APNs). This class introduces a new type of inhibitor arcs, the colour-sensitive inhibitor arcs. These arcs work combining the property of the ordinary inhibitor arcs with the different data types of the resources, i.e., absence or presence of tokens in the inhibitor places and tokens of different colours. Furthermore, this class of inhibitor nets is supported by both its enabling and firing rules and its composition theory. The introduction of Ambient Petri Nets intents to provide a high level Petri net class that captures and reflects the key features and the functioning of the ambient systems in an accurate and faithful way.

1.3 Petri Nets and Transition Systems

As was mentioned, Petri nets are used for the modelling of concurrent systems providing their graphical representation. This representation usually depicts the structure of the examined systems with respect to the system components and features that are considered for their modelling. Specifically, the graphical repre-

[1]Monotonicity denotes that addition of tokens to the existing marking cannot disable already enabled transitions of that marking but only can enable other transitions that were not previously enabled [65].

sentation of the systems does not make the behaviour of the nets apparent, which is actually described by the ‘semantics’ that determine both the initial marking and the functioning (firing and enabling rules) of these nets [56].

A traditional way to capture the concurrent behaviour of the Petri net models of real systems is the *transition systems*, which are a powerful mathematical formalism that provides graphical behavioural models [47]. These models consist of states and transitions, where states refer to the condition of the system before or after an event and transitions represent the events that cause the state changes of the system. In the graphs of transition systems, states are denoted by nodes and transitions by labelled arcs[1].

According to [46], transition systems can be distinguished into two general categories, the *sequential transition systems* and the *step transition systems*. The *sequential transition systems* consider only single events for the evolution of the state changes (state space) of the system behaviour, which implies that concurrent actions that may occur within the system are not considered. On the other hand, the *step transition systems* exhibit both the sequential and the concurrent behaviour of the systems, by depicting labelled arcs with either single or multiple events respectively.

To construct the transition system of a Petri net model, we have to define the states and the transitions of the transition system out of the respective elements of the net starting with the initial state of the transition system, which is given by the initial marking of the Petri net model. Then, examining the ‘token game’ [2] by following the semantics (enabling and firing rules) of the Petri net model, we detect all the possible actions that may occur in the net and draw them as labelled arcs in the graph of the transition system. These arcs originate from the initial state and end to new states that derive from the marking change of the net. Afterwards, the same process is followed for each of the new states of the transition system till the point where there are no other new states to visit or the same states are revisited. In that case, the transition system of the net has been completed reflecting the behaviour of the net of the system.

In the rest of this thesis, for the behavioural representation of the Petri net models of ambient systems, we use the step transition systems and particularly the *concurrent reachability graphs* in order to visualise their concurrent behaviour. This fact derives from the concurrent nature of ambient systems, which implies that the concurrent behaviour of these systems should be considered in their modelling. On the contrary, the produced Petri net models of the ambient systems are analysed with respect to their sequential semantics due to the fact that Charlie

[1]The label of each arc corresponds to the action(s) that take(s) place each time.

[2] Token game refers to the circulation of tokens within a net by complying with the semantics of that net.

verification tool provides exhaustive sequential analysis but lacks in concurrent analysis as it supports only maximal concurrency, which means that it does not consider possible intermediate concurrent steps that may take place within the examined models.

1.4 Formal Verification and Model Checking of Concurrent Systems

Having modelled and exhibited the behaviour of concurrent systems like the ambient systems, through the modelling process, the primary concern of the designers is to prove that the system models have been developed properly. To ensure that, different techniques are available, such as formal verification[20] and synthesis [51]. At this point, we should mention that the main discussion point of this work as regards the 'evaluation' of the developed models is the formal verification.

Formal verification is a widely used, effective and successful technique that aims at the examination of the model correctness with respect to the system specification. In other words, formal verification checks whether the produced model of an examined system complies with the requirements of its specification. The correctness of a model is usually related to potential design, reliability or security errors that impair the system functioning. For example, a correct model should guarantee that the system design satisfies the specification and works always reliably without its information being exploited by non-authorised parties. In case that a model fails to adhere to the specification, it should be improved and rechecked till it passes the verification successfully. Otherwise, the system's malfunctioning could result in unwanted situations that range from low cost effects (loss of information) [19] to high cost effects (loss of lives) [42].

Therefore, the correctness of a model is checked with respect to the system specification by examining a number of fundamental model properties, which are usually related to the architecture (structure) and the functioning (behaviour) of the developed model [24]. Afterwards, the results of these properties are translated into meaningful outcomes the examined system.

Apart from the fundamental properties mentioned above, specific system properties can be examined through the model checking [44], which is an automatic verification approach that checks whether the model of an examined system satisfies a number of properties that is expressed in form of temporal logic formulas. Examples of the temporal logic are the Linear Temporal Logic (LTL) [59] and the Computation Tree Logic (CTL) [16] that both examine if a logic preposition is satisfied within the state space of the system[1]. The result that derives from the model checking of a logic formula is Boolean, i.e., either true or false, indicating in that way that the examined property holds or fails respectively. It should be noted that the result of the model checking can be interpreted in many different ways depending on the kind of system or the type of property that is examined. For example, a property that turns false does not necessarily imply that the system is defective. Respectively, a property that turns true does not imply that the system works properly.

The use of formal verification and, by extension, of all its approaches (*e.g.*, model checking, theorem proving, etc.) contributes in an effective way to the exhausting examination of a system model with respect to the specification. The successful application of formal verification to the analysis of different types of concurrent systems leads to the conclusion that it can check the correctness of system models whose behaviours ranges from low to very high complexity and concurrency, *e.g.*, distributed systems [15], reactive systems [2], etc.

So far, the analysis of ambient systems using Petri nets has not been thoroughly examined yet, since these systems are newly introduced. Hence, we explore this field by introducing a new approach that will efficiently cover the domain of formal modelling and verification of ambient systems in relation to Petri nets.

1.5 Modelling Approach and Motivation

The modelling approach that is discussed in the thesis is concerned with the use of formal modelling and verification techniques of interactive aspects of ambient systems. As was mentioned earlier, there is already a substantial literature on the role of formal methods in the modelling and evaluation of interactive systems [26, 27, 61], but not on the modelling and analysis of interaction of ambient systems using Petri nets. Thus, one motivation for the introduction of this approach has been to find instances of interactive behaviour that could be related to both the functioning of the system and the misbehaviour of their users and also to examine how these two factors influence the whole interactivity process. Typical properties of interest for the analysis of these systems include availability of information to both environment and users, leak of information to unauthorised users, context-awareness of the system, initiative of users with respect to available actions, response of environment to users' actions, cost of ignoring system's response and examination of deadlock cases.

The models of this approach focus on key features of interaction in the ambient systems. Consequently, a key feature of particular interest in these systems is the

[1]Model checking may suffer from the state space explosion problem [18]. In such a case, model checking cannot be effectively applied to models with infinite state space.

way that the users interact with the environment, as a result of both explicit interaction with the system, and implicit interactions that arise through changes of context. Here context could include location, or the steps that have to be taken by a user to achieve some goal. Another key feature of the ambient systems is how that interactivity of the environment with the users is expressed through system components like sensors and output devices (e.g. public and private displays). Therefore, the proposed modelling approach describes how models can be built to represent the interaction between users, devices and services, as users move within ambient environments.

The interactive nature of ambient systems is expressed, through that modelling approach, using Petri nets notation, and more specifically using a subclass of Coloured Petri Nets, the Ambient Petri Nets (APNs). The motivation behind the definition of this class of Petri nets is to satisfy three basic requirements. The first is to facilitate the modelling of real ambient systems through the compositionality of a fundamental modular APN net (basic step net), which captures a single interaction between the user and the environment. The second is to enhance the analysis of the ambient systems combining the qualitative reasoning of Petri nets and the analysis of system properties through the use of model checking. Finally, the last requirement is to enable the generic use of that class in the modelling of ambient systems. Specifically, APN class through the compositionality and modularity of the basic step nets could be used for the modelling of real systems focusing on different levels of abstraction of interactivity each time. This is motivated by the fact that the interactivity of an ambient system can be considered as a 'collection' of interactive single actions of users that are followed by the response of the environment, which all together reflect and represent the interactive behaviour of the systems. Thus, choosing an abstraction level, which could range from very low to very high, for the modelling of the interactivity of ambient systems, it could be represented by the APN nets produced through the introduced modelling approach as far as every action of a user is followed by the response of the system.

1.6 Contributions

In general, this work contributes to the modelling and verification of concurrent systems, in particular the ambient systems. The main contributions of the thesis are summarised as follows:

- A broaden definition of ambient systems that incorporates notions and characteristics of ambient systems from several definitions given to these systems up to date in available literature. (Chapter 2)

- A taxonomy approach that is based on the new ambient systems' definition and distinguishes the ambient systems into two categories: the Ambient Guidance Systems and the Ambient Information Systems. (Chapter 2)
- Definition of Ambient Petri Nets (APNs), which is a subclass of Coloured Petri Nets with inhibitor arcs. This class introduces the colour-sensitive inhibitor arcs and has its own enabling and firing rules that take into consideration these inhibitor arcs. (Chapter 3)
- The step-modelling approach, which is used to build up in a compositional way the behavioural models of the ambient systems out of building blocks that represent single actions of a user and the response of the system to these actions. (Chapter 4)
- Definitions of the *forward composition* and the *backward composition* of APN nets. (Chapter 4)
- Definition of composite step nets, which are a subclass of the APN nets that is obtained from the building blocks by applying forward and backward compositions. (Chapter 4)
- Definition of the class of Transformed Ambient Petri Nets (T-APNs), which is a subclass of Coloured Petri Nets without colour-sensitive inhibitor arcs. (Chapter 5)
- Definition of a construction of the T-APN net associated with a given composite step APN net. (Chapter 6)
- Proof of the behavioural equivalence of a composite step APN net and its associated T-APN net. (Chapter 6)
- Verification of qualitative and model checking properties of T-APN models of chosen case studies using Charlie verification tool [11]. (Chapter 7)

Some of the above contributions of this thesis were published as technical reports [36, 38] and presented at the PhD session of an international conference [37].

1.7 Thesis Overview

In this thesis, the remainder of the work is organised as follows:

Chapter 2 introduces the various definitions that have been given to the ambient systems up to date in the available literature, which are followed by our 'new' definition about these systems. Afterwards, the existing taxonomy approaches along with our newly defined taxonomy approach are presented. The last part of this chapter describes two case studies of ambient systems, the selection of which is based on the new taxonomy approach.

In chapter 3, the class of Ambient Petri Nets (APNs) that provides coloured inhibitor nets is defined. This class is accompanied by the corresponding semantics that determine the enabling and firing rules of these nets. Finally, the behaviour of the APNs is captured through the step transition systems that are defined in the last section of this chapter.

Chapter 4 discusses the step-modelling approach, which describes the modelling of ambient systems by using building blocks of APNs with specific structure, the basic step nets. In addition, nets like the root net and the composite step nets are presented for the sake of the two APN compositions, which are discussed at the second half of the chapter. These two compositions, the forward composition and the backward composition, enable the gluing of the APNs on the net places. Finally, we demonstrate how the APN models of the case studies can be created by using the two compositions of the APNs[1].

In chapter 5, a subclass of Coloured Petri Nets [9, 33], the Transformed Ambient Petri Nets (T-APNs), is defined. This class produces non-inhibitor nets. As with APNs, T-APNs come with the corresponding semantics that define the enabling and firing rules of these nets, as they are presented in the respective section of this chapter. The last part of the chapter discusses the behaviour of the APNs through the step transition systems.

Chapter 6 starts with a comparison between the nets of the APNs and the T-APNs. Next, a construction that sets up the relation of the elements of the two classes is defined. Afterwards, we illustrate the use of the construction through a generic example and highlight the correct functioning of a T-APN net that is constructed out of a given APN net. Furthermore, we describe the transformation of the APN models of the case studies into the respective T-APN models by applying the construction[2]. Finally, in this chapter, we prove that a constructed T-APN net and its associated APN net are behaviourally equivalent by showing that their sequential and step transition systems are isomorphic.

In chapter 7, we check the correctness of the produced T-APN models of the case studies by performing qualitative and model checking analysis. The discussion about the analysis of the T-APN models starts with the investigation of the qualitative properties that consist of the structural and behavioural properties, which examine fundamental static and dynamic properties of the T-APN nets [33, 54]. The remaining sections discuss the analysis of the model checking prop-

[1]The APN models of the case studies are implemented using Snoopy modelling tool [29].

[2]The constructed T-APN models of the case studies are also implemented using Snoopy.

erties that contain the ambient properties and the general properties, which are related to the examination of the key features and the general behaviour of the ambient systems respectively. At the end of the chapter, we discuss further verification aspects, such as path analysis of the reachability graphs of the models of the case studies.

Finally, chapter 8 discusses the most important outcomes of this work and suggests further improvements or additions that could be made in the future in order to achieve better modelling and verification results.

Chapter 2

Ambient Systems (AS)

The rapid development of ambient systems, which are increasingly related to the use of advanced technology, leads more and more researchers to deal with these complex environments. The need of exhaustive understanding of ambient systems resulted in the creation of different definitions for these systems. Despite the fact that the researchers, through their definitions, focus on different aspects of ambient systems, they consist of certain characteristics (features) that dominate their functionality.

To achieve a more comprehensive and easy understanding of what an ambient system is, apart from the different definitions, we also present some taxonomy approaches that will help us to become familiar with the notion of this kind of systems.

The different definitions and taxonomy approaches, which are described in sections 2.1.1 and 2.2 respectively, are used for the best possible study of ambient systems. More specifically, all of them help towards the identification of those elements of ambient systems that should be examined through the modelling of representative examples of these systems. The selection of the case studies is primarily based on our own taxonomy approach for the ambient systems (see section 2.2.5), which is stemmed from the observation of the already available definitions and taxonomies. Unlike the other approaches, this taxonomy approach is more generic and intends to categorise the ambient systems in accordance with the factors that affect their functioning and not with respect to the aesthetic, technological or technical characteristics of these systems, such as the way that the information is shown to the users, how fast that information is disseminated to them, etc. The difference between the other approaches and ours becomes apparent after the description of all the taxonomies.

Thereafter, two case studies of the ambient systems are presented in the rest of this chapter. These case studies are used later on in the thesis for the modelling and examination of the behaviour of the ambient systems using Petri nets. At the end of this chapter, two of the aforementioned taxonomies are compared by taxonomizing the systems of the case studies. This comparison takes places to show that these systems can be classified by any of the taxonomies regardless of the fact that they have been selected according to the criteria of our proposed taxonomy.

2.1 Ambient Systems

Nowadays, humans are looking for systems that make their lives easier and more pleasant with respect to the fulfilment of their everyday obligations, transactions or activities. As has already been mentioned, the tremendous advance of technology has contributed to the construction of ambient systems, which incorporate both ubiquitous and pervasive computing. Their main purpose is to serve their users so that they will not be charged with additional or time-consuming tasks in order to achieve their goals. In fact, the aim of these systems is to act in the background in a silent way that will not distract or interrupt the users from their goals, but will inform them only when is necessary. In addition, they usually facilitate the users' tasks by tracking down or predicting their potential actions. Ambient systems are named after their ability to be completely embedded in the environment in which they operate.

In the next section, we present the definitions that have been given to the ambient systems specifying both what these systems and their main characteristics are. In the last paragraphs of the section, we provide a broaden definition of the ambient systems that focuses on all those system components and features that define the interactivity and the architecture of these systems. Finally, we refer to some examples of examined ambient systems.

2.1.1 Defining Ambient Systems

Luca Cardelli and Andrew Gordon [14] define the ambient system as a bounded place where computation happens, can be nested within other ambient systems, can be moved as a whole, has a unique name and a collection of agents. The notion of the boundary determines what is inside and what outside of an ambient. The second characteristic allows an ambient to be used as part of another ambient. The 'move' property implies that a device can be connected (disconnected) to (from) a system automatically without causing problems to the system. Finally, the name of the ambient is used to control the access to the systems and the collection of agents characterises the ability of the system to run computations.

Iliasov et al in [30] define ambient systems as systems that have mobile elements, need to be context-aware and are open. Mobile elements can be code, devices, data, services and users. Context-awareness mentions the ability of the system to be adapted to the information changes and the openness refers to the fact that system components can appear or disappear.

Kristensen et al in [41] define ambient systems as environments where "users participate in several ongoing computations. Aspects as time and space are central in the systems. The systems usually identify users, collaborate intelligently, and support users in their ongoing activities. In ambient systems the participants can have the initiative. The goal is not to provide information per se, but to support activities directly in the physical world."

Russ et al [58] define ambient systems as: "An ambient information system is a system which offers its user mobile and pervasive access (mediated by sensors and effectors of their immediate environment) and which is capable of adapting itself to the particular user needs and profiles."

After the presentation of the available definitions of the ambient systems, we provide our own definition that provides a more descriptive and comprehensive description of the ambient systems. This definition incorporates both all the characteristics/features that were defined in the above definitions and all those system components that capture the interactivity of these systems. Thus:

> An ambient system is defined as a system that acts in a silent way at the background of the environment (i.e. allows the user to have the initiative), does not distract the user from his goals, predicts or tracks down the potential actions of the user, informs/alerts him periodically or when is needed and has the following characteristics (features) and components:
>
> - Is context-aware, namely it adapts itself according to the information changes. These changes can be change of location/position of an object (device) or a subject (user), change of time, temperature etc.
> - Consists of output devices and mobile elements, such as private displays (PDAs, mobiles, laptops) and public displays (boards, screens), where the (updated) information is displayed,
> - Has advanced technology, like robotic technology, sensors, effectors etc., which makes it capable of identifying, tracking down and interacting with the user or the system components/devices,
> - Is a bounded place. The dissemination and use of the data/ information is restricted by the boundary of the system,
> - The information is always available (depending on the access level of the user). Each user has access only to the informa-

tion that he is eligible for. In addition, the availability of the information requires continuous connectivity to the system,

- Devices can be automatically and directly connected (disconnected) to (from) the system without affecting its functionality.

2.1.2 Related Work on Ambient Systems

In this part, we briefly present some examples of examined ambient systems, which either guide the users to a specific place or inform them about important tasks or sessions that they should follow.

Harrison and Massink [26] have modelled a Guidance system and an Out-patient system. The first one aims to guide the visitors that are unfamiliar with an office building to a particular location in it. The Out-patient system is similar to the previous one but is located at a hospital. This system provides information to the patients that is relevant to their appointment, the waiting times and finally helps them through the guidance system to get to the suitable doctor or take the appropriate exams.

Kray et al have developed an Airport model [27]. That system simulates the steps of the airport system. In more details, it senses the location of the user and informs him about his flight, the possible next steps that he has to do and all the relating information. This model works by using sense that identifies the user and his current location.

Finally, *Luis Silva et al* have implemented a system that simulates the functionality of a smart Library [61]. This model informs the user of the books that are in stock and then guides him to the place where the book that he wants is. Firstly, the user requests a book through the library webpage and then goes to the library entry gate where a display shows his request. He enters the library and the guidance system of the library leads him to the shelf where the book is located. The system points out the exact position of the book by a light of a specific colour. The user picks up the book and then he can leave.

2.2 Taxonomy Approaches of Ambient Systems

The primary intention of the following paragraphs is to present briefly some of the developed taxonomy approaches that refer to the classification of the ambient systems with respect to criteria that are related to their features.

It is worth pointing that despite the rapid growth of the ambient systems and their embodiment in our lives, their classification is yet in early stages. This fact is justified by the difficulties that emerge during the development of such a taxonomy approach. These difficulties refer to the motivations, the goals, the range of the examined dimensions, the efficiency of the approach etc. As has been noticed, only a few different approaches have been developed or published to date.

At the end of this section, we introduce a new taxonomy approach that classifies the ambient systems into two general categories in accordance with the factors that give rise to their functioning and generally their interactivity with the users.

2.2.1 Ames' and Dey's Taxonomy Approach

Ames and Dey introduce a classification for ambient information systems that derives from their experience in developing such systems [1]. Their approach is based on eleven system dimensions that are used to assess the quality of the classified systems. These dimensions are: intrusiveness, notification, persistence, temporal context, overview to detail, modality, level of abstraction, interactivity, location, content and aesthetic.

The dimensions mentioned above are described in [1] as follows:

- Intrusiveness: displays do not demand attention, but provides information with a level of intrusiveness appropriate to the information's importance.
- Notification: devices display information constantly, but alert with a more salient cue when a certain state is reached or when information changes - they do not demand the same amount of attention all the time.
- Persistence: displays show information at an appropriate time scale and an appropriate refresh rate.
- Temporal context: if comparison with past or prediction of future is important, displays show it, reducing cognitive demands on user by not requiring them to remember other states.
- Overview to detail: displays show the right amount of detail: get an overview at a glance, and more detail if one pays attention.
- Modality: displays show information in a mode (that is, using a sense) that is not already overloaded.
- Level of abstraction: displays do not show information directly, but rather in an abstract or indirect manner. The method of displaying information should be clearly linked to the nature of the information.
- Interactivity: displays are appropriately interactive (or not), without demanding too much from the user.

- Location: displays reflect sensitivity to location and their surroundings in general, such as a quiet room vs. a noisy public plaza.
- Content: displays show information that the user cares about, or are flexible in content.
- Aesthetics: apart from being useful or valuable as information sources, the displays are also pleasing.

2.2.2 Matthews et al. Taxonomy Approach

Matthews et al. have implemented a taxonomy that focuses on three basic dimensions for the evaluation of the ambient systems: notification level, transition and abstraction [43]. The notification level refers to the importance of the data that is disseminated within the system. Notification level is divided into five levels that depict the importance of the data: Ignore, Change Blind, Make Aware, Interrupt and Demand Attention. These levels are sorted in an ascending order (from low to high). The dimension of transition represents the programmatic changes to the displays that result from the data updates. Transition is assessed using the following enumeration: interrupt, make aware and change blind. The last dimension (abstraction) describes the process that transforms the input data to any types of perceivable information that can be displayed on the screens of the ambient systems (e.g. sign, picture, caricature, letters or numbers). The abstraction is measured using two categories: feature abstraction or degradation.

2.2.3 Pousman's and Stasko's Taxonomy Approach

Pousman and Stasko define a new set of dimensions to analyse the ambient systems [53]. These dimensions are: information capacity, notification level, representation fidelity and aesthetic emphasis. Information capacity refers to the throughput of information that can be displayed by an ambient system. Information capacity is rated using five levels: low, somewhat low, medium, somewhat high and high. For instance, systems that depict only a single piece of information are characterised by low information capacity and systems that display more detailed information could belong to upper levers depending on the throughput of information that they deal with. Notification level characterises the intrusiveness of the system. In more details, a system alerts the user according to the importance of the information. At this point, Pousman and Stasko adopt, with a slight difference, the levels that are defined by *Matthews et al* [43]. Pousman and Stasko have changed the lower level from Ignore to user poll. Representation fidelity deals with the different ways under which the information can be presented by the system. The display of the information calls for representational accuracy and easy perception of the meaning of the presented information. In order to analyse these notions, Pousman and Stasko introduced the categories as follows [53]:

- INDEXICAL: measuring instruments, maps, photographs, text
- ICONIC: drawings, doodles, caricatures
- ICONIC: Metaphors
- SYMBOLIC: language symbols (letters and numbers)
- SYMBOLIC: abstract symbols

These categories have been sorted from high (indexical) to low (symbolic). Systems that cannot represent the information accurately and the user is not able to comprehend the meaning of the information easily belong to the lower layers of this categorisation. Finally, the last dimension that has been proposed is the aesthetic emphasis. Aesthetic emphasis focuses on the intention of the developer to produce a visually pleasing system. This dimension can be rated only in a subjective way. This occurs due to the fact that the evaluators of a system may have different opinion about what is aesthetically pleasing or not. The grading scale that is used in this case is identical to that of the first dimension.

Pousman and Stasko also suggest four design patterns that aim to provide a generic categorisation of the ambient information systems. These patterns derive from four different combinations of the dimensions of their proposed taxonomy. These four patterns are: Symbolic Sculptural Displays, Multiple-Information Consolidators, Information Monitor Display and High Throughput Textual Display. Figure 2.1 shows the graphs of these patterns (adapted from [53]), which are described thoroughly in the following paragraphs.

Symbolic Sculptural Displays: this pattern consists of ambient systems that provide few information to the users and their information is displayed in an abstract way.

Multiple-Information Consolidators: this pattern contains systems that can supply the user with much more information than that of the Symbolic Sculptural Displays. Finally, they make the users aware of the information changes and could be characterised as aesthetically elegant systems.

Information Monitor Displays: this pattern includes ambient systems that can provide too much information but in an iconic way (usually information displayed by metaphors) and can notify the user of the context changes in an efficient manner. As regards the aesthetic emphasis, these systems could be classified either in the medium or in the somewhat low level.

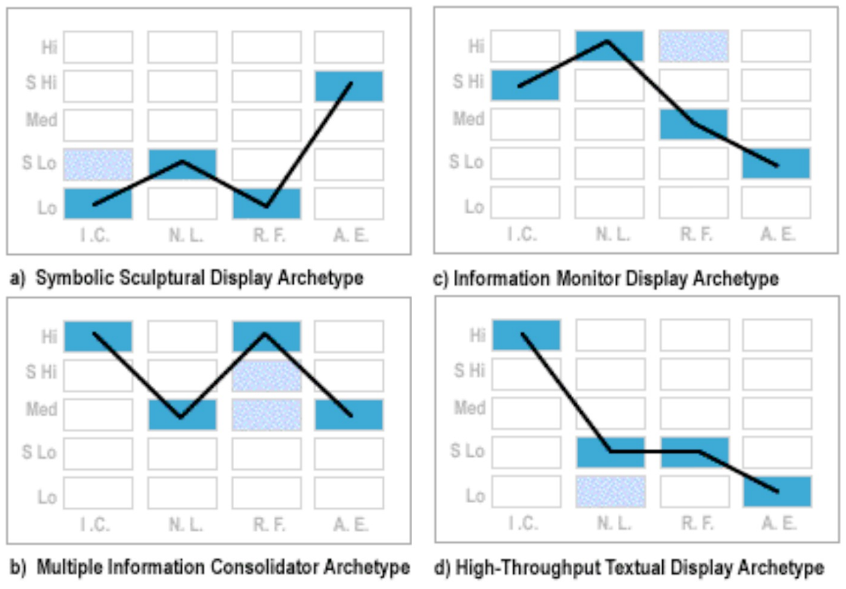


Figure 2.1: The four design patterns of Pousman and Stasko.

High Throughput Textual Display: this pattern consists of all those systems that can deal with huge throughput of information and can represent it with very simple graphics. Furthermore, the notification level of these systems can be either user poll or change blind, which means that these systems are possessed by low intrusiveness. Finally, their low aesthetic emphasis ranking proves that the primary goal of the designers of these systems is not the elegancy.

At this point, it should be mentioned that these design patterns do not cover all the ambient systems that can be analysed by this taxonomy approach. For example, there exist systems that could be classified to a different combination of dimensions or could belong to more than one pattern.

2.2.4 Tomitsch et al. Taxonomy Approach

Tomitsch et al.[63] aim at the introduction of a taxonomy approach that would have more descriptive power than that of the existing ones. Therefore, they developed a taxonomy that contains all those features that have the greatest influence on the design of the ambient systems. These system features correspond to the following dimensions: abstraction level, transition, notification level, temporal gradient, representation, modality, source and dynamic of input. Each of these dimensions uses different metrics in order to be analysed.

Abstraction level describes how the encoded data is represented by the system. This dimension can be measured using three different levels of abstraction: low, medium and high. Systems that belong to the low level represent the information in a direct and comprehensible way. Medium level systems depict the information in quite comprehensible way. Finally, high level contains all those systems that use a symbolic representation of the information.

Transition refers to the state changes of the system that result from the update of the information. More precisely, transition measures the velocity of the state changes that may occur during the operation of the system. Transition can be measured as slow, medium and fast. A system that is characterised by a slow transition does not make the user aware of the state change directly. Medium transition systems change the state of the displays faster than the slow transition systems. Finally, fast transition systems are able to immediately change the state of the display.

Notification level discusses the way under which the system alerts or notifies the user as regards the information changes. In this approach, the developers have adapted the same metric features as these in the taxonomy approach of *Matthews et al* [43] in order to analyse the notification level of the ambient systems.

Temporal gradient refers to whether the system displays a record of all the state changes of the information or not. The metric of this dimension is either history (all states are displayed) or current (only the current state is displayed).

Representation deals with the nature of the output displays, i.e. the type of the displays. This dimension uses three classification categories of the output displays: physical, integrated and 2D representation. Physical representation refers to output displays that are designed to be ambient information systems. Integrated representation includes systems that use object with embedded technology that function like ambient displays. 2D representation category consists of systems that use typical screen technology.

Modality describes the forms under which the information can be presented to the users of the system. Metric of this dimension can be: visual, tactile, olfactory, auditory, and movement.

Source refers to the locality of the information in relation to the source. In other words, it describes whether the displayed information and its source are within the same environment or not. The metric features of this dimension are classified as local, distant and virtual. Local source implies that the position of the displayed information is the same with that of the source. Distant source systems allow the dissemination of the information to mobile devices that are located in a large distance from the source. Finally, in virtual source systems, information can be provided using virtual networks like internet or mobile networks.

Location, as it is described in [63], refers to the location or context of the output devices. Three common classes of location are used: private, semi-private and public.

Dynamic of input describes the frequency or the velocity to which the data changes in relation to the source of that data. Dynamic of input is quite similar to the transition of the system. It uses the same metric features as these of transition. Slow dynamic describes rare data changes that in their turn result in rare updates of the output displays. Medium dynamic refers to frequent changes of the input data. Finally, fast dynamic implies direct and fast data changes.

Through this taxonomy, *Tomitsch et al.* try to relate the used dimensions to each other so as to identify possible patterns that could be useful for the analysis of the examined ambient systems. For instance, they have concluded that the majority of the systems with slow transition must also have a change blind notification level.

2.2.5 Introducing a Taxonomy Approach

From the different definitions given to the ambient systems and the various taxonomy approaches, it can be concluded that the interactivity of the ambient systems with the users is mostly influenced by the system features and components that are mentioned in the definition that we provided in section 2.1.1. Furthermore, it has also been observed that ambient systems could be taxinomized according to how significant is the role of the system components that are present in these systems and to the 'exogenous' factors that determine or affect the functionality of the ambient systems, such as the spatial structure of the system or the context (information) provided to the environment by the users.

Consequently, based on our definition about the ambient systems, we introduce a new taxonomy approach that classifies these systems into two categories, the Ambient Guidance Systems and the Ambient Information Systems. In both categories, we focus on the interactivity aspect of the systems, which is expressed through components and features like sensors, output devices (displays) and context-awareness. It should be mentioned that all the systems of both categories allow the users to have the initiative, but in different level of freedom. The classification of the ambient systems into one of the above categories is conducted considering as criteria both the exogenous factors and the role of the system components to the functioning of the examined systems, as mentioned above.

2.2.5.1 Ambient Guidance Systems

Every ambient system that detects/tracks down the location of the users (or agents) in order to interact with them and its behaviour is affected by both the spatial structure of the system and the context (information) provided to the environment by the users is classified in the Ambient Guidance Systems. By spatial structure, we refer to the way that the components are located within the environment of the system. For example, where the sensors and the displays are located, what is the topology of the system, etc. The interactive functioning of these systems is basically based on the strong presence of sensors and output devices (displays) which detect and advise the users respectively and the context-awareness which is expressed through the detection (use of sensors) of the users. Usually, this category of systems consists of ambient systems that guide the users to a specific location through his detection and possible redirection when is necessary.

2.2.5.2 Ambient Information Systems

Every ambient system that identifies the users (or agents) and their privileges with respect to their roles in order to interact with them and its behaviour is affected only by the context (information) provided to the environment by the users is classified in the category of the Ambient Information Systems. The functionality of these systems is dictated by the use of authentication or authorisation access control systems (passwords, cards, etc.) and output devices (displays), which identify and advise the users respectively. Note that sensors, effectors, etc. are not very frequently used in this category of systems and their use is of minor importance or little contribution to the systems behaviour. The context-awareness of these systems is expressed through the role based access control system that determines how the information is distributed to the users according to the changes of their privileges or roles. Usually, this category consists of ambient systems that provide the user with all the available information about his ongoing activities and support him towards their completion.

2.3 Case Studies of Ambient Systems

Having mentioned the key features of the ambient systems in the previous sections, we present two ambient systems that are used as case studies in order to examine their behaviour using Petri nets. These systems fully comply with the generic definition of ambient systems that is presented in section 2.1.1. Furthermore, these two systems are representative examples of Ambient Guidance Systems and Ambient Information Systems respectively, as is described in section 2.2.5.

The first system, i.e., the ambient garage, is an Ambient Guidance System that aims to guide the users to allocated parking places. On the contrary, the other system, i.e., the ambient conference room, is an Ambient Information System that helps the users to complete tasks or sessions in order to conclude about the accepted and rejected papers of a conference.

2.3.1 Ambient Garage

Starting with the description of the Ambient Guidance Systems case study, it should be mentioned that the ambient garage, which is presented below, is an extended version of an application that was developed in VRML (Virtual Reality Modelling Language) language by K. Konstantopoulos, D. Salogiannis and A. Konios during their undergraduate studies in University of Pireaus [39]. The specification of the garage is presented in the form of scenario in the following paragraph.

2.3.1.1 Scenario of Ambient Garage

The ambient garage is a bounded environment and its capacity is limited. For the sake of simplicity, it is assumed that this particular garage can host up to a certain finite number of cars and is equipped with entrances that are used to get access to the garage and cannot be used as exits. For that reason, separate exits have also been designated for the exit of the cars. In addition, the garage is always in operation and when the system reaches its maximum capacity, the cars that request access to the building have to wait in a queue until a parking bay is released and becomes available. The drivers that wait for a parking bay are informed that the garage is full by a message on their mobile device. Each of these drivers can leave the queue any time he wants without taking any further actions. The maximum length of the queue can be equal to two for each entrance.

The drivers (users) gain access to the garage only by using a pre-paid ticket. This ticket can be bought either online or from the automated ticket kiosk that is located outside of the garage where the users fill in a form with personal details like name, surname, address, mobile number etc. Once a user has been registered, the system keeps his details in the database regardless of him using the garage regularly or not. It is also assumed that any driver that attempts to enter the garage is always a registered one. After the registration, the user can buy three different types of tickets, an hourly, a daily or a monthly ticket. If a user buys a ticket, a confirmation message is sent to his mobile device (this process will be described later). The types of tickets correspond to the different sectors/areas where the car can be parked. Each sector has limited parking spaces and each ticket corresponds to one specific space (parking bay) of that sector. Each parking bay of the garage is equipped with sensors in order to allow the environment (system) to know whether a car is parked to the appropriate position or not. If that car is parked to a wrong place, a visual message is activated alerting the driver to remove his car and park it at the correct parking bay. This message is depicted by a flashing light that has the same colour as that of the instructions indicating the correct place where the car should move to. Moreover, each cars has equipment (with unique registration number) that enables the sensors of the garage to acquire the appropriate information each time that is needed. This equipment is used for the identification of each car as the unique registration number of the equipment is considered as the ID of each car. Finally, it is assumed that each driver always carries a mobile device (e.g. PDA, mobile phone) with him, where he receives all the relevant information.

Continuing the description of the system, it should be mentioned that the garage is fully equipped with sensors that recognise/detect the position of each car at any time. Every sensor of the garage covers a particular area without overlapping with other sensors. The ambient garage also has a guidance system (or signage system) that gives instructions to each driver of how he can get to the parking bay that is allocated to his car. All these instructions are shown on the public displays (boards) of the garage, assigning a specific colour for the signage (e.g. arrows) of each car. In case that a driver misses the target by following wrong instructions, the system redirects him by finding his current position and changing the relevant signage on the public displays (update of the data). When the driver finds his space and successfully parks his car, the sensors recognise that the suitable car has been parked to the space by scanning the ID of the car. Then the system sends a message to the mobile device of the user informing him about the remaining time that he has in his disposal (by showing a timer that counts down the time). Fifteen minutes before the time expires, the system transmits again an alert or notification message to the user reminding him that he has to leave the garage within the remaining time or that he can renew the ticket. If the user does not leave the garage within that time, a fine is imposed by the system. On the other hand, if the user renews the ticket, the same procedure as that of buying a ticket is repeated, i.e. the system sends a message to the mobile device notifying him with the remaining time.

Until now, we have described the procedure that takes place when the car is in the garage but we have not discussed yet what happens when a car/driver requests to enter or leave the building respectively. In the former case, the car approaches the bar of the garage, where a sensor scans the ID of the car finding out if the ticket has been paid or not. In case that it has not been paid, the access is denied to the car and a message is sent to the mobile device of the driver, informing him that he has to buy a ticket. Otherwise, the access is granted and the system lifts up the bar. When the bar is lifted up, the environment automatically assigns a parking place to the car according to the type of its tickets and guides it towards that place. In the latter case, the car/driver wants to leave the building approaching the exit bar. Another sensor identifies the car and checks if it is authorised to leave. If not, the system sends a message to the driver notifying that the fine must be paid. When the fine is paid, the driver is free to leave, otherwise the bar remains down.

2.3.2 Ambient Conference Room

The following case study is a representative example of an Ambient Information System. This system refers to an application that was developed by A. Petrou during his master's thesis [50]. This application simulates an ambient environment of a conference room, which is based on the RAHMEDE (Reception Agenda Human MEdiating Display Environment) system developed at Newcastle University [22]. The ambient conference room is a place where the conference committee decides which of the submitted papers will be accepted or rejected in order to be included or not in the conference. The detailed description of the functioning of that environment is explained in the following paragraphs through the presentation of the scenario, where the implementation of the application was based on.

2.3.2.1 Scenario of Ambient Conference Room

The behaviour of this system is presented by following a scenario that consists of sub-scenarios that cover all the possible cases that could occur in the conference room. This environment requires authorized members that can have access to the conference room using a smart card that corresponds to their ID [1]. Each of these cards is unique and it is supposed that each user always carries it with him. The users can enter the room only by swiping their cards to the card reader that is located on the door. The same process is repeated when the users want to leave the room. The ambient environment using the cards can identify who is in the room at any time.

The room is equipped with personal displays (PCs), sensors that are embedded in the personal displays and a public display. Furthermore, each personal display uses a role based access control system to identify the authorised users. This access control system is working by using a password and a username. Each time that a user enters his credentials, the system allows him to gain access to the information that corresponds to his role. The roles that can be assigned to the members of the conference are: author, spokesperson, temporary reviewer, reviewer, committee member, stand-in chair manager and chair manager. Some of the authorized members of the conference session can be assigned with more than one role.

Prior to the beginning of the conference session, the chair manager, who also has administrative privileges, can have access to the database in order to initialise the registered members, their roles and the appropriate data (submitted papers,

[1]All members are authorised to enter the room but at different phases of the conference, which means that they might be considered as unauthorised users at some point of the system operation.

initial decisions) that are needed for the conduct of the session. An assumption that all the submitted papers have already been graded (by two reviewers each) is made and is called as the initial review. According to the scores of the initial review, the system makes a suggestion about the initial decision for each paper. This initial decision will be used during the main phases of the conference (e.g. first phase).

The conference meeting consist of three different phases and a break phase. During the first phase, all the authorised members can participate in the conference session (are in the room) and the information about the submitted papers and their initial reviews is shown on the public display. In phase one, the only personal displays that can be used are those of the chair manager and the spokespersons. The display of the chair manager shows the review information of all the submitted papers and allows the chair manager to take actions, such as making/changing initial decisions for the papers (accepted or rejected). On the other hand, the displays of the spokespersons enable them to see information about the reviews of the papers that are assigned to them (each spokesperson can acquire information only for the papers that are assigned to him). If an unauthorised person gets in the room (during phase one), the system hides all the information of the public display and appears it again when that person leaves the room. The same thing happens when that person gets close to an activated personal display. In former case, the system identifies the unauthorised user by his smart card (when he swipes it, the card reader sends a signal to the system that someone enters the room and the system checks the database to realise if he is a registered member or not). In the latter case, the sensor of the personal display senses that another (unauthorised) person is within the range of the private display and hides all the information of the display. In addition, it does not matter whether the authorised user is in front of the display or not, in that case the system will hide all the information from everyone that is within that range. The first phase ends when the chair manager has assigned all the papers to the temporary reviewers and a decision about which of the papers will be reviewed in the next phases has been concluded. At the end of the first phase, the system 'forces' the members that cannot participate in the next stage to exit the room immediately in order to allow the remaining authorised members to carry on with the procedure of the conference.

The completion of the first phase is followed by that of the break. The members that can be present in the room during the break phase are all the members that have one of the following roles: chair manager or temporary reviewer. During that phase, the temporary reviewers can see (on their personal displays) information about the list of the papers that must be reviewed and can also score them. Each temporary reviewer can only receive information about the papers that are assigned to him. In contrast to the private displays, the public one displays the list of the papers under review and a timer that shows the remaining time till the end of the session. If a reviewer submits his scores for the relevant papers, the system does not let him change his mark or any other detail. As in the first phase, an unauthorised person could get in the room unexpectedly. At this case, the timer will be automatically stopped due to the interruption of the session and will start again when that person leaves the room. In case that all the reviews have been submitted their scores or the time has expired, the break phase is terminated and the conference proceeds to the next phase.

Phase two is held after the end of the break phase. In this phase, all the members of the conference should leave the room, except for the chair manager, the spokespersons and the committee members that can attend this session. The second phase can start only when the appropriate persons have remained in the room. Then, the public display of the room presents all the updated information about the paper reviews and scores, which were carried out during the previous phase. The committee holds a session to make a final decision for each paper. In this case, some conflicts between the roles can emerge. For instance, a member of the committee can also be an author of a reviewed paper. Even the chair manager could be in that situation. Therefore, the elimination of these phenomena is achieved by removing the privilege from the member that is closely related to the paper that is currently examined. That member must leave the session until a final decision has been made for that paper. After that, each 'banished' member can return to the session. If that member is the chair manager, the system sends an alert message to him through his private display and deactivates his role until the final decision of that particular paper, by that time another predetermined member temporarily replaces him as the stand-in chair manager. Conflicts can occur even if the authors of the reviewed paper are from the same institute or organisation with some of the committee members. Once again, the system forces the members that are in conflict with the currently examined paper to exit the room, otherwise the session cannot continue. It is also assumed that an unauthorised user can interrupt the process by entering the room, as in the previous phases. The system reacts to that interruption as in the previous phases. When both all the conflicts and the interruptions have been solved and all the final decisions have been made, the second phase is finished.

Finally, the last phase of the conference allows everyone to be present in the room. During that stage, the final decision for each paper is announced and all the relevant information is displayed on the public screen. In this phase, the personal displays cannot be used and it is assumed that interruptions by unauthorised users cannot affect or change the conference procedure. After the announcement of the decisions, the conference meeting is completed and all the participants leave the room.

2.4 Comparison of Taxonomy approaches

Next, we show, through two examples, how the case studies described above can be classified using the approaches mentioned in sections 2.2.3 and 2.2.4. The motive behind this illustration is firstly to compare the two approaches in order to draw some useful conclusions about which of these two taxonomies is more effective and secondly to demonstrate that representative systems of a specific taxonomy approach can be classified through the use of other approaches.

Starting with the first taxonomy example, we classify the two case studies using the approach of Pousman and Stasko [53]. Firstly, we examine the information capacity of our systems and can claim that the ambient garage can have somewhat high or high information capacity. This decision derives from the fact that the system can represent a huge amount of information on its public displays (i.e. boards). On the other hand, the ambient conference room belongs to the medium class due to the fact can depict limited information either on the public or private displays due to the confidentiality of the information shown.

The next dimension is the notification level. As has already been mentioned above, the notification level has the following classes: User poll, Change Blind, Make Aware, Interrupt and Demand Attention. The ambient garage could be listed to the Make Aware notification level. This results from the fact that the system always makes the user aware of the state changes (information changes) without interrupting or demanding attention. Contrary to the garage, the conference room sometimes notifies the user of a state change by interrupting him from his primary goal. This occurs when an “intruder” gets in the room or he gets close to a private screen. At these occasions, the system hides all the information and does not permit the completion of the task.

Analysing the two ambient systems in terms of representation fidelity, we conclude that both systems represent the information in a direct and comprehensible manner. Both of them use textual information. In addition, the ambient garage also uses iconic representation like coloured arrows. Although, it uses iconic representation; it could be included in the indexical class because the majority of the information is represented by text. Conference room also belongs to the same class.

Finally, as regards the aesthetic emphasis dimension, we have subjectively assessed both the garage and conference room and classed them in the medium class. We believe that since both systems are equipped with LCD screens as output displays, they could be characterised as aesthetic good looking systems, therefore we put them in the medium class.

The classification that was explained above is presented in the following table:

	Information Capacity	Notification level	Representational fidelity	Aesthetic emphasis
High/Demand Attention/Indexical			Ambient Garage, Conference Room	
Somewhat high/Interrupt/Iconic	Ambient Garage	Conference Room		
Medium/Make Aware/Iconic	Conference Room	Ambient Garage		Ambient Garage, Conference Room
Somewhat low/Change Blind/Symbolic				
Low/User poll/Symbolic				

Table 2.1: Rousman and Stasko taxonomy

As is noticed from the above table, these systems do not match to one of the four patterns that Pousman and Stasko have introduced, but we could claim that the ambient garage and the ambient conference room are very close to the Multiple Information Consolidators and the Information Monitor Display archetypes respectively.

Continuing with the second taxonomy approach, we classify the two systems using all the dimensions and the corresponding metrics of the approach of *Tomitsch et al* [63].

The abstraction level of both systems can be characterised as low because both system support textual representation, thus there is no or very low abstraction to the representation of the information. Examining the transition, we notice that both systems have a fast transition. That means that the users of these systems recognise the state changes immediately on the private or public displays. For example, the garage immediately notifies the user when he loses his way by redirecting him through the change of the arrows that lead to the parking space. In the conference room, a kind of fast notice is observed when an unauthorised user enters the room. The notification level in this approach is almost the same with that of the previous approach, so the garage and the conference room belong to Make Aware and Interrupt class respectively. Next step is the assessment of the temporal gradient. Both systems do not keep record of the previous states, so they are included in the 'current' class. After that, we examine the representation and the modality of the systems. Both systems use LCD screens and represent their information in a visual way (text, symbols), therefore we classify them as 2D representation and visual modality respectively. The last three dimensions that are examined are: Source, Location and Dynamic of input. As regards the Source, the garage gives the opportunity to the users to retrieve information through mobile or wireless networks, so it is characterised as virtual. On the other hand, the conference room is characterised as local because the information is available only within the room. Location of both systems can be described as a combination of public and private classes. This happens because these systems use both types of context. Finally, the Dynamic of input for both systems is fast due to the fact that they use dynamic changes of input.

Having conducted these two different taxonomies for the above ambient systems, we have concluded that these approaches analyse and focus on different aspects of the systems. To be more specific, the first one is a simple method that does not cover a huge spectrum of system design issues. On the other hand, the latter approach is more descriptive and allows the designers to understand the importance of some system features that could improve the functionality and the effectiveness of the systems that they want to develop. For that reason the second method could be applied for the creation of more useful and effective taxonomy patterns. The results of the second comparison are summarised in Table 2.2.

2.5 Concluding Remarks

In this chapter, we discussed a variety of issues that are related to the ambient systems. The basic concept of this chapter was to introduce the ambient systems and to show how these systems can be classified by using some taxonomy approaches. Firstly, we presented the different definitions that have been given to the ambient system by the researchers since their development. All these definitions helped us to understand the ambient systems and to spot all those characteristic/features and components that are of high importance for the interactive behaviour of these systems.

Afterwards, we introduced a new definition about the ambient system, which derives from a selective combination of the other definitions. In other words, this definition describes the majority of the system characteristics and covers most of the aspects that have been mentioned in all the other definitions.

Furthermore, we presented various taxonomy approaches that categorise the ambient systems with respect to different parameters such as the performance of the system, the expressiveness of the information, etc. Throughout the discussion about the taxonomies of the ambient systems, it has been noticed that all the proposed taxonomy approaches deal mainly with technical details of the systems and not with how these systems could be categorised according to the factors or the components that affect their behaviour. This led to the conclusion that further approaches could be proposed for the taxonomy of the ambient systems. For that reason, we created a generic taxonomy approach that is based on the concerns mentioned above and classifies the ambient systems according to them, i.e., 'exogenous' factors and system components. As was mentioned, the proposed taxonomy approach consists of two categories, the Ambient Guidance Systems and the Ambient Information Systems.

After the introduction of our taxonomy approach and the discussion about its categories, we chose and presented two representative case studies (one system

	Abstraction level			Transition			Notification level					Temporal gradient		Representation			Modality					Source			Location			Dynamic of input		
Systems	Low	Medium	High	Slow	Medium	Fast	Ignore	Change Blind	Make Aware	Interrupt	Demand Attention	History	Current	Physical	Integrated	2D	Visual	Tactile	Olfactory	Auditory	Movement	Local	Distant	Virtual	Private	Semi-public	Public	Slow	Medium	Fast
Ambient Garage	√					√			√				√			√	√							√	√		√			√
Conference Room	√					√				√			√			√	√					√			√		√			√

Table 2.2: Taxonomy of M. Tomitsch, K. Kappel, A. Lehner and T. Grechenig.

for each category) that will be used later on for the modelling and the analysis of the ambient systems.

Finally, in the last section of this chapter, we compared two of the taxonomy approaches by classifying the ambient systems of the case studies in order to demonstrate that the existing taxonomy approaches can be applied to any ambient system. It should been mentioned that the proper selection of the taxonomy approach that will be used for the classification of an ambient system is based each time on the parameters that we want to examine the system for.

Chapter 3

Ambient Petri Nets (APNs)

The previous chapter is devoted to the extensive discussion about the ambient systems. Moreover, it refers to the categorisation of these systems into Ambient Guidance Systems and Ambient Information Systems according to their key features and the factors that affect their behaviour. Generally, chapter 2 provides useful information about the ambient systems that could help to their analysis. Consequently, it results that in order to investigate the behaviour of the ambient systems with Petri nets, we need a class that would provide a faithful graphical representation of the systems of these two categories.

To achieve that, we introduce, in this chapter, a new class of Petri nets that aims to express and capture both the structure and the concurrent behaviour of the ambient systems. This class is called Ambient Petri Nets (APNs) and belongs to the high level of Petri nets. Actually, the APNs incorporate features from different Petri net classes in order to ease the modelling of these systems. Examples of these features are the different types of resources (tokens) and the inhibitor arcs that are 'inherited' from the Coloured Petri Nets (CPNs) and the Place/Transition nets (P/T Nets) respectively [33, 56].

More specifically, the APNs is a variant subclass of CPNs that involves a new type of arc, the colour-sensitive inhibitor arc. As with CPNs, APNs are governed by 'rules' that define the structure and the functioning of the nets of this class in relation to both their static and dynamic expressions[1] [8]. It should be mentioned that the static aspect is expressed by notation and semantics of the APNs that focus on the structure of the nets without considering any resources. On the other hand, the dynamic aspect of the APNs is given by those semantics that refer to the firing and the enabledness of the transitions and the distribution of the resources within the nets. All the notation and semantics of the APN class are defined later in this chapter.

[1]Static and dynamic expressions refer to unmarked and marked APN nets respectively.

Finally, as was mentioned above, the APN nets principally stipulate the structural representation of the ambient systems. Therefore, to analyse the behaviour of these systems, we have to produce their behavioural representation. Thus, the behaviour of an APN net becomes apparent through its step transition system, which illustrates the concurrent behaviour of the ambient system model that is related to. A special case of the step transition systems is the *concurrent reachability graphs* of the APNs, which can be used for the verification of the APN models providing useful information about their behaviour.

3.1 Preliminaries

In this section, we recall some basic notions related to multisets which are used throughout the rest of the thesis.

Definition 1 (multisets). *A* multiset *m over a set S is a mapping $m : S \to \mathbb{N}$. (Intuitively, $m(s)$ is the multiplicity of $s \in S$ in m.) We will denote by μS the set of all the multisets over S.*
We will use $\mu^{\infty} S$ to denote the set of all extended multisets *over S, i.e., mappings $m : S \to \mathbb{N} \cup \{\infty\}$. (Intuitively, the elements can now have an infinite multiplicity.)*
For every $s \in S$, we will denote by m_s a multiset over S such that $m_s(s) = 1$ and $m_s(\overline{s}) = 0$ for $\overline{s} \in S \setminus \{s\}$.
For any (extended) multiset m, $supp(m) = \{s \in S \mid m(s) \neq 0\}$ is its support.

Sometimes we will use $m = \{p_1, p_2, p_2\}$ to denote a multiset such that $m(p_1) = 1$, $m(p_2) = 2$ and $supp(m) = \{p_1, p_2\}$.

Definition 2 (operations and relations on multisets).
Let m and m' be multisets over S, and $X \subseteq S$.

- *The* sum *of m and m' is the multiset $m + m'$ over S such that, for all $s \in S$, $(m + m')(s) = m(s) + m'(s)$.*
- *The* difference *between m and m' is the multiset $m - m'$ over S such that, for all $s \in S$, $(m - m')(s) = max\{0, m_1(s) - m_2(s)\}$.*
- *The scalar multiplication of a multiset m by a natural number n is the multiset $n \cdot m$ over S such that, for all $s \in S$, $(n \cdot m)(s) = n \cdot m(s)$.*
- *m is a* sub-multiset *of m' if, for all $s \in S$, $m(s) \leq m'(s)$. We denote this by $m \leq m'$.*
- *$m|_X \in \mu X$ is the* restriction *of m to X if, for all $s \in X$, $m|_X(s) = m(s)$.*

As has already been mentioned, multisets and their operations are used for all the definitions of this work and especially for those of the APN and T-APN class respectively. These classes are expressed in terms of multisets in order to provide universal definitions that will cover the modelling of general ambient systems. But, for the sake of the application of our modelling approach (see chapter 4) to the chosen case studies, we simplify their definitions considering that the maximum multiplicity that is used for the representation of the users in these cases is equal to one. This happens because we consider that both classes are colour safe, which derives from the fact that each colour of tokens is used as ID for the users[1]. Nevertheless, these classes could be used in a more general way for the modelling of ambient systems where colour safeness is not a prerequisite. Such an example could be ambient systems where the different colours of tokens correspond to a group of users with the same ID, allowing in that way the existence of multiple tokens of the same colour in order to represent possible group activities. For that reason, these new classes are defined using multisets extending their use and making them universal, as was discussed earlier.

3.2 Ambient Petri Nets Class

Having described all the necessary multiset notions for the definition of the APN class, we proceed to the introduction of this class of nets, which aims at the modelling of general ambient systems. The discussion about the APN class takes place in three parts, starting with the general definition of the class followed by the definitions of the semantics and the step transition system of the APNs.

Definition 3 (Ambient Petri Net)**.** *An* Ambient Petri Net *is a tuple*

$$\mathcal{N} = (P, T, Pre, Post, I, Cl, C, K, M_0, G),$$

where:

- *P is a finite set of places.*
- *T is a finite set of transitions disjoint from P.*
- *Pre, $Post : T \to \mu P$ are the pre and post mappings.*
- *$I \subseteq P \times T$ is a set of inhibitor arcs. We assume that:*
  $\forall\ p \in P\ \forall\ t \in T$: $(Pre(t)(p) > 0 \vee Post(t)(p) > 0) \Rightarrow (p,t) \notin I$.
- *Cl is a non-empty finite set of non-empty colour sets. We will call it the* structuring *set.*

[1]This will become clear after the discussion about the modelling approach.

- *$C : P \cup T \to Cl$ is a colour function used to structure places and transitions:*
  - *$\widetilde{P} = \{(p, g) \mid p \in P \wedge g \in C(p)\}$ is the set of structured places.*
  - *$\widetilde{T} = \{(t, c) \mid t \in T \wedge c \in C(t)\}$ is the set of structured transitions.*
- *K is a constant function that defines the capacities of places.*
- *$M_0 \in \mu\widetilde{P}$ is an initial marking satisfying $M_0 \leq K$. In general, any $M \in \mu\widetilde{P}$ such that $M \leq K$ is a marking.*
- *$G \subseteq P$ is a set of gluing (or interface) places.*

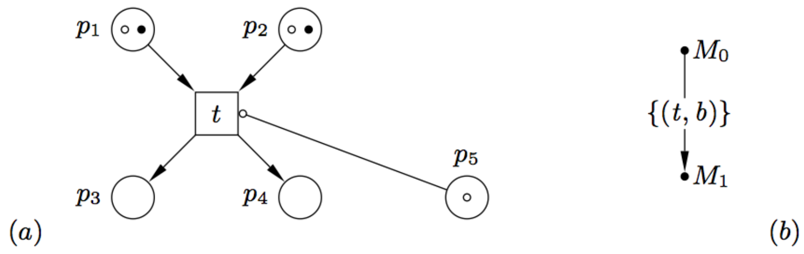


Figure 3.1: An Ambient Petri Net $\mathcal{N}$, where the capacity K is a constant function 1 (a), and its concurrent reachability graph $CRG(\mathcal{N})$ (b). In the diagram, each $\circ$ token represents w, and each $\bullet$ token represents b.

The meaning and graphical representation of P (places) and T (transitions) are as in the standard net theory. The set of places, G, specified as the last element of the tuple, will be important in the composition of Ambient Petri Nets. The directed arcs of the net are given by the *Pre* and *Post* mappings. For example, for the only transition of the APN net in Figure 3.1 we have: $Pre(t) = \{p_1, p_2\}$ and $Post(t) = \{p_3, p_4\}$. We will call p_1 and p_2 the *pre-places* of t, and p_3 and p_4 the *post-places* of t. In general, any $p \in supp(Pre(t))$ is a pre-place of t and any $p \in supp(Post(t))$ is a post-place of t. In this thesis, $Pre(t)$ and $Post(t)$ will be sets for any $t \in T$. This means that every directed arc of an APN net has the weight 1. We will also use the following 'dot' notation:

$$\begin{array}{rcll}
{}^\bullet t & = & supp(Pre(t)) & \text{(pre-set of } t \in T\text{),}\\
t^\bullet & = & supp(Post(t)) & \text{(post-set of } t \in T\text{),}\\
{}^\bullet p & = & \{t \in T \mid p \in supp(Post(t))\} & \text{(pre-set of } p \in P\text{),}\\
p^\bullet & = & \{t \in T \mid p \in supp(Pre(t))\} & \text{(post-set of } p \in P\text{).}
\end{array}$$

If $p \in {}^{\bullet}t$ then we will denote the arc from p to t by (p, t). Respectively, if $p \in t^{\bullet}$ we will denote the arc from t to p by (t, p). It is noted that the above 'dot' notation can also be extended for the sets of places and transitions and will be denoted as ${}^{\bullet}P$, $P^{\bullet}$ and ${}^{\bullet}T$, $T^{\bullet}$ respectively.

An *inhibitor arc* $(p, t) \in I$ will be represented in diagrams by an edge ending with a small circle, like (p_5, t) in Figure 3.1. We will call p_5 an *inhibitor place* of transition t.

The Cl set is used for structuring places and transitions, by equipping them with colour sets as it is used as co-domain in the colour function C. Places can hold coloured tokens, representing different kinds of resources or agents. The set of structured places is given by $\widetilde{P}$. A structured transition $(t, c) \in \widetilde{T}$ represents an action that operates in a colour 'mode' given by c. In Figure 3.1, $Cl = \{\{w\}, \{b, w\}\}$, where b stands for the black colour and w stands for the white colour. $\widetilde{P}$ for the APN net in Figure 3.1 is the set $\{p_1, p_2, p_3, p_4\} \times \{b, w\} \cup \{(p_5, w)\}$ assuming that $C(p_i) = \{b, w\}$ for $i \in \{1, \ldots, 4\}$ and $C(p_5) = \{w\}$. In diagrams, a structured place is represented by a coloured token that resides in this place. For example, (p_1, b) means that a black token was placed in place p_1. Also, $\widetilde{T}$ (for the APN in Figure 3.1) can be defined as $\{(t, b), (t, w)\}$, assuming that $C(t) = \{b, w\}$. Structured transitions only become apparent when we look at the net's behaviour (see Figure 3.1(b)).

The initial marking M_0 (which cannot exceed the capacities given by K) specifies, for each structured place p, the number of tokens of each colour held in p. The initial marking M_0 of the net in Figure 3.1 is $\{(p_1, b),\ (p_1, w),\ (p_2, b),\ (p_2, w),\ (p_5, w)\}$. In this work, APNs will always be *safe coloured Petri nets*, because we will allow at most one token of a given colour per place. This is motivated by the fact that colours will be used as identities given to agents or other entities interacting within an ambient environment. As a consequence, the capacity K will be a constant function returning 1 for each colour of tokens per structured place.

3.3 Semantics of APN nets

After the explanation of the net elements, we continue with the definition of the semantics of the APNs, which indicate the functioning of the APN nets by providing the firing and the enabling rules of the class.

Accordingly, to define the semantics of an APN net we need to extend Pre and $Post$ mappings using structured places and transitions. $\widetilde{Pre}$, $\widetilde{Post} : \widetilde{T} \to \mu\widetilde{P}$ are defined as follows:

$$\widetilde{Pre}(t, c)(p, g) = \begin{cases} 0 & \text{if } c \neq g \\ Pre(t)(p) & \text{if } c = g \end{cases} \qquad (*)$$

and

$$\widetilde{Post}(t,c)(p,g) = \begin{cases} 0 & \text{if } c \neq g \\ Post(t)(p) & \text{if } c = g \end{cases} \qquad (**)$$

$\widetilde{Pre}(t,c)$ denotes, for every place p, the number of tokens of colour c that t needs from place p in order to fire. $\widetilde{Post}(t,c)$ denotes, for every place p, the number of tokens of colour c that t will deposit in p after being fired. Notice that $Pre(t)(p)$ and $Post(t)(p)$ in the formulas above would be 1 or 0 for APN nets, as the weights of the direct arcs are all 1 and 0 stands for the non-presence of direct arcs respectively.

We can now extend $\widetilde{Pre}$ and $\widetilde{Post}$ functions to steps of structured transitions $U = \{(t_1,c_1),\ldots,(t_n,c_n)\} \in 2^{\widetilde{T}}$ as follows: $\widetilde{Pre}(U) = \widetilde{Pre}(t_1,c_1)+\cdots+\widetilde{Pre}(t_n,c_n)$, and similarly for $\widetilde{Post}$. The above extension is needed as the proposed semantics will be a *step semantics* rather than a *sequential semantics.*

Furthermore, we can define a set of *colour-sensitive inhibitor arcs* based on the set of inhibitor arcs I:

$$\widetilde{I} = \{((p,c),(t,c)) \mid (p,t) \in I \wedge c \in C(p) \cap C(t)\}.$$

The meaning of a colour-sensitive inhibitor arc will depend on the colours with which we equip transition t and the tokens in its inhibitor place p. The arc will be blocking transition t if t is in mode c and there is a token c in p, and will have no effect on t otherwise.

Now we can describe the semantics of an APN net $\mathcal{N}$. A step of structured transitions U is *enabled* at marking M if the following hold:

- $\widetilde{Pre}(U) \leq M$.
- $M + \widetilde{Post}(U) \leq K$.[1]
- $(M \times U) \cap \widetilde{I} = \varnothing$.[2]

We denote this by $M[U\rangle_{\mathcal{N}}$. An enabled step U can fire producing the marking $M' = M - \widetilde{Pre}(U) + \widetilde{Post}(U)$. This will be denoted by $M[U\rangle_{\mathcal{N}}M'$.[3] That means that a step is enabled if the pre-places of all the transitions in the step, working in certain colour modes, have sufficient number of tokens of appropriate colour. Furthermore, an enabled step U satisfies that the tokens added to the marking of

[1] If we want to allow self-loops in the net, this condition should take a weaker form: $M - \widetilde{Pre}(U) + \widetilde{Post}(U) \leq K$.

[2] This condition means that there are no $(p,c) \in M$ and $(t,c) \in U$ such that $((p,c),(t,c)) \in \widetilde{I}$.

[3] $\mathcal{N}$ can be omitted if it is clear from the context.

the APN net, after the firing of U, do not violate the capacity of the net places, as is expressed by the second condition. Finally, for the step to be enabled, none of the transitions of the step should have its inhibitor places (if any) marked by tokens of the colour equivalent to its working mode colour.

The transitions of an enabled step U can fire concurrently, which implies that they are not in conflict with each other for the resources of the pre-places, but they might be in conflict with other transitions that do not belong to step U. A conflict between two transitions is defined as follows:

Definition 4 (Conflict). *Two enabled structured transitions $(t,c),(t',c') \in \widetilde{T}$ are in conflict at a marking M if they can fire separately, i.e., $M[(t,c)\rangle_{\mathcal{N}}$ and $M[(t',c')\rangle_{\mathcal{N}}$, but not concurrently, i.e., not $M[\{(t,c),(t',c')\}\rangle_{\mathcal{N}}$.*

For the APN in Figure 3.1, the step $\{(t,b)\}$ is enabled at M_0 and can fire, removing black tokens from p_1 and p_2 (see $(*)$), and adding one black token to each of p_3 and p_4 (see $(**)$). However, the steps $\{(t,w)\}$ and $\{(t,b),(t,w)\}$ are not enabled at M_0 and cannot be fired as $((p_5,w),(t,w)) \in \widetilde{I}$. Furthermore, a step with more than one structured transition (t,b), like $\{(t,b),(t,b)\}$, is also not enabled, because of a lack of sufficient number of black tokens in p_1 and p_2.

3.4 Step Transition System of APN nets

The full execution semantics of an APN net will be captured using a transition system where arcs are labelled by executed steps, the *step transition system*. A step transition system is defined as follows:

Definition 5 (Step Transition System). *A* step transition system *over T is a triple $STS = (Q, \Delta, q_0)$ consisting of a set of* states Q*, set of arcs $\Delta \subseteq Q \times 2^{\widetilde{T}} \times Q$ and the* initial *state $q_0 \in Q$.*

The definition of the step transition system is followed by that of the concurrent reachability graph, which is given below:

The *concurrent reachability graph $CRG(\mathcal{N})$* of $\mathcal{N}$ is a step transition system $CRG(\mathcal{N}) = ([M_0\rangle, \Delta, M_0)$ over T where:

$$[M_0\rangle = \{M_n \mid \exists U_1, \ldots, U_n \; \exists M_1, \ldots M_{n-1} \; \forall 1 \le i \le n \; : \; M_{i-1}[U_i\rangle M_i\} \qquad (3.1)$$

is the set of reachable markings and $(M, U, M') \in \Delta$ *iff* $M[U\rangle M'$. Figure 3.1(b) shows the concurrent reachability graph of the APN net in Figure 3.1(a). Furthermore, we will call $U_1 \ldots U_n$, as in the formula (3.1), a *step sequence* and write $M_0[U_1 \ldots U_n\rangle M_n$.

To describe the behaviour of the APNs through the step transition system, we take as example the net in Figure 3.1(a) and its generated concurrent reachability graph (see Figure 3.1(b)). We start capturing the behaviour of the net by examining the enabledness of transition t, as it is the only transition of this net. According to the enabling rules of the APNs, t is enabled for the black token at the initial marking M_0, as is described in section 3.3. This means that the transition can fire at least once, i.e., under the black mode. After the firing of enabled step (t, b), a new state results and is described by the marking $M_1 = \{(p_1, w), (p_2, w), (p_3, b), (p_4, b), (p_5, w)\}$. Now, checking the enabledness of t at M_1, it turns that the transition cannot be executed further since it is not enabled. So, the non-existence of enabled transition leads to a dead state[1] for the net, which is its terminal state defined by M_1. Hence, to draw the concurrent reachability graph of an APN net, we start from the initial marking M_0 and we generate an arc that links to a new marking (state) every time that a step is executed. This arc is labelled with the step that is executed each time. In this case, the concurrent reachability graph of $\mathcal{N}$ is shown in Figure 3.1(b) and consists of two states, M_0 and M_1, which are linked with the arc that is named after the step (t, b), as was described earlier.

Consequently, examining the enabledness of all the transitions of an APN net and executing every time its enabled transitions, concurrently or not, we can construct the concurrent reachability graph of that net starting from the initial state that is usually given by the marking M_0.

3.5 Concluding Remarks

In this chapter, we defined the class of the APNs by describing both the static and the dynamic aspects of the APN nets. The static aspect of the APNs is related to those elements that determine the structure of these nets, such as the places, the transition, the arcs, the capacity, etc. On the contrary, the dynamic aspect of the APNs concentrates on how the marked APN nets behave with respect to the semantics mentioned above. The behaviour of APNs is captured by the step transition systems and in particular by the concurrent reachability graphs of the nets. All the above notions are fundamental principles of the APN class and are needed for the correct functioning of the nets.

Thus, having completed the description of the APNs, it can be concluded that the APNs are a well defined class with strong mathematical background that can be used for the faithful modelling of general ambient systems. Its intention is not to prove that ambient systems can only be modelled with this class of Petri nets, but the main motivation behind the introduction of this class of Petri nets

[1]Dead state refers to a state where no transition is enabled.

is to reflect the features of these systems in an accurate and flexible way. Finally, the APN class provides simple, readable and understandable models of ambient systems.

Chapter 4

Modelling with Ambient Petri Nets (APNs)

In the previous chapter, we described the Ambient Petri Nets (APNs) that resulted from the need of faithfully modelling of general ambient systems. In this chapter, it is examined how this new class could be used in the modelling of the ambient systems.

In this work, the modelling of ambient systems focuses on the interactivity between the system and the user, as the functioning of the these systems is based mainly on interactive tasks. Therefore, it is crucial to examine, through the APN nets, how the user can affect the system's actions and vice versa.

The interactivity of the ambient systems is expressed in terms of the detection or the prediction of the user's action and the advices (notification or feedback) that the system provides to the users after their actions. Apart from the interactivity, another interesting feature of these systems is examined, the context-awareness. Context-awareness indicates the ability of the system to interact with the user in respect of the information provided each time that an action takes place. For the modelling of both the interactivity and the context-awareness of the ambient systems, the proper system components should be chosen in order to represent them.

Having described the behaviour of the ambient systems (see chapter 2), it can be concluded that the interaction between the user and the system is expressed by means of system components like sensors and output devices (private or public displays). Hence, to examine the ambient systems, we model these components in the context of user's behaviour and system's response. Another aim of this chapter is to present the modularity issues that emerge throughout the modelling of the case studies. In particular, a building block that expresses the interaction between the user and the system is described. Basically, the modelling of the ambient systems using APNs is based on the repetitive use of that block.

For that reason, the step-modelling approach is introduced aiming at the construction of APN models for the ambient systems. This approach describes how the APN model of an ambient system is built up taking into consideration both the system components and the modularity issues. The rationale behind this approach is explained in detail in the following sections.

Finally, the composition of the APNs is addressed in the last part of this chapter through the introduction of two compositional operators that indicate two different ways of expanding the APN nets by gluing them on their places. Thereafter, these compositions are used for the construction of the APN models of the ambient systems presented in section 2.3.

4.1 Step-Modelling Approach

The name of the step-modelling approach derives from the logic that governs the ambient systems, namely that the user cannot be at two different places or perform two different important tasks at the same time. So, a user as an entity usually performs actions *step by step*, but he can execute tasks (actions) concurrently with the system's actions and with other users' actions.

As has already been mentioned, a fundamental building block is introduced for modelling purposes. The repetitive and proper use of which could result in the development of the APN models for the ambient systems of the case studies. This building block describes a single action of a user, and the way in which this particular action leads to the change of the state for both the system and the user. The structure and the components of the building block are discussed in this section explaining their significant role in the construction of the APN models.

4.1.1 Building Block

Starting the description of the step-modelling approach, we introduce the *unidirectional step*, or *unidirectional step system*, which is part of the building block.

> *A unidirectional step is used when an action is permitted to happen only in one direction. In this case, the user or the system cannot retrieve the previous condition/state. An example of such a step is given in Figure 4.1.*

Note that the unidirectional step in Figure 4.1 is a very simple net consisting of two places and one transition. In order to distinguish these places and the transition, we will use the notation $P_{ST} = \{p_1, p_2\}$ and $T_{ST} = \{t_1\}$. A formal definition of a unidirectional step will be given later in section 4.2.

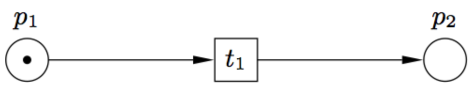


Figure 4.1: A unidirectional step.

Having introduced the unidirectional step — the first part of the introduced building block — we now specify the control system which decides how the system should respond depending on the information received through the action of the user.

The control system of a unidirectional step is used to synchronise the user's action with the response of the system.

In this section, the structure and the functioning of the unidirectional step with control system are intuitively described through the following example. But, to explain the control system of the building block explicitly, an extensive discussion is held in section 4.1.2.

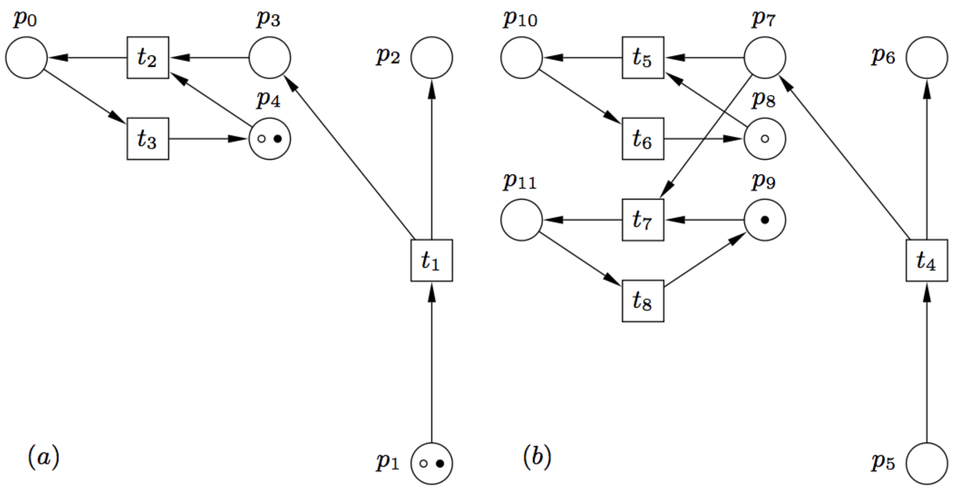


Figure 4.2: A unidirectional step with control system (a), and unidirectional step with control system and multiple decision points (b).

Figure 4.2 shows two unidirectional steps with control systems. Assume that the net in Figure 4.2(b) represents a step that follows the step described by the net in Figure 4.2(a). We now explain how a system that is composed of these two unidirectional steps with control systems works. The net of that system is shown in Figure 4.3. It should be mentioned that the composition of the unidirectional steps with control systems is formally defined in section 4.3.

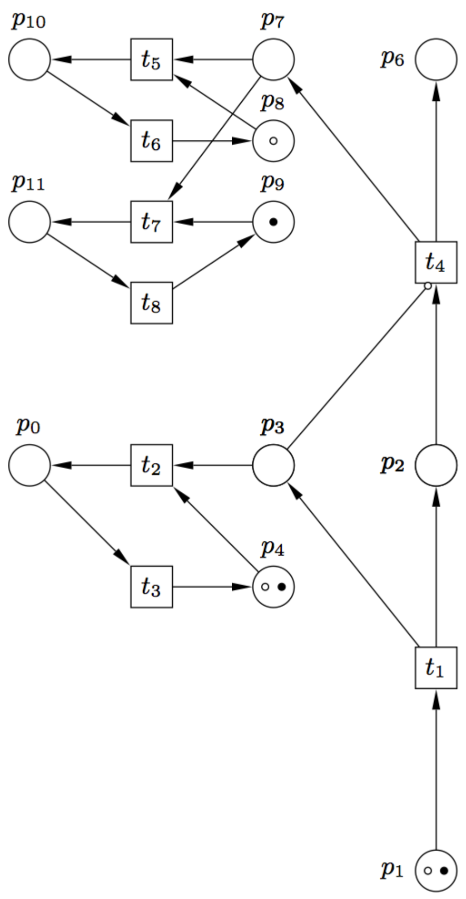


Figure 4.3: A composition of two unidirectional step building blocks.

The following simulation of the combined steps demonstrates the functioning of the control system. The control system of the first unidirectional step consists of the places p_0, p_3 and p_4 and the transitions t_2 and t_3, as is shown in Figure 4.2(a). Respectively, the control system of the second unidirectional step, shown in Figure 4.2(b), consists of the places p_7, p_8, p_9, p_{10} and p_{11} and the transitions t_5, t_6, t_7 and t_8.

Now, we start the simulation from place p_1, assuming that the token game[1] starts with the firing of transition t_1 for the black token. By executing the transition, the black token is removed from place p_1 and one black token is deposited

[1]By the term 'token game', we refer to the distribution of tokens after the firing of transitions, as it is defined by the semantics.

in each of the p_2 and p_3 places. At this point, it is said that p_3 'controls' p_2 as transition t_4 cannot fire when there are tokens of the same colour in p_3 and p_2 due to the colour-sensitive inhibitor arc between p_3 and t_4 (see Figure 4.3). The presence of the colour-sensitive inhibitor arc derives from the composition of the unidirectional step, which will be described later in section 4.3. The only way to execute transition t_4 (in the black mode) is by removing the black token from place p_3 (by executing t_2). In other words, the only way for the user to execute the next task/action is by obtaining first the appropriate response from the system. This will become clearer in the following section, where the elements of the control system of the building block are described.

4.1.2 Structure and Functioning of Control System

The necessity to describe and explain both the elements and the functioning of the control system arises from the simulation demonstrated in the example above. Hence, the control system (CS) of a unidirectional step could be described as a component of the building block that consists of:

- the *control place* (p_c),
- the *database places* (P_d),
- the *response places* (P_r) and
- transitions whose pre- and post-places are either control, database or response places.

The control place (p_c) is used to control the corresponding place of the step (i.e., the post-place of the transition of the step). In other words, there is a one-to-one correspondence between the control places and the post-places of transition of the steps. The control place is also used for the depositing of tokens reflecting in that way the detection of the user's action. The detection of a user's action prompts the system to give the proper response to the user. It is important to note that each control system has only one control place.

The database places (P_d) store all the useful information about the configuration of the system in form of tokens. For instance, after firing a transition that has as pre-places the control place and a database place, the system can respond to the user's action according to the information that is given through the token that resides in the control place. For example, in Figure 4.2(b), if we assume that a black token is stored in the control place, in this example p_7, then the system will check the colour of the token, will also check its database places, and will then execute the transition that is enabled giving the proper advice to the user. In this case, the system realises that the colour of the token is black, checks the database place p_9 (where the information about the black token is located) and finds out that the user associated with the black token must be provided with the information that is indicated by the post-place p_{11} of the executed transition t_7.

The response places (P_r) usually represent the advice that is given to the user through the output devices of the system (public displays, private displays, etc.). An example of a response place is place p_{11}, as was mentioned earlier.

The number of the response and database places of a control system is determined by the number of different advices that the system can provide with respect to the number of different options that a user has after making an action, which is described by the associated step. For instance, lets assume that the first step of Figure 4.3 represents a car that moves from the entrance of the garage (p_1) to another position (p_2). Lets also assume that when a car arrives at p_2, it can only go forward. Then the system will only have a database place that includes all the users and a response place that shows the same advice to all of them. Continuing with the second step of Figure 4.3, we assume that at place p_6 each user has two options (go left or forward). Thus, if a (black) user is at place p_6, then the system should provide the correct advice regardless of what the choice of the user will be after that advice. In this example, the system has allocated a parking spot to the black user and depending on where that parking place is (according to the topology of the garage), it gives directions to the user. Hence, being at place p_6, the system should advise the user to go towards the appropriate direction, by executing transition t_7. However, as is explained by our definition of the ambient systems (see section 2.1.1), a user always has the initiative, which means that he is not obliged to follow the advice of the system. On the other hand, the system always has to provide the correct information to the user. Regarding the advice that is given to the user through the response places of the control system, it should be noticed that the labels of these places represent the nature of the advice (e.g. instructions like forward, backward, right or left can be represented by the respective labels in case of a guidance system). Finally, the absence or presence of tokens in these places indicates whether the advice has been given or not.

It should be mentioned that the different response or database places cannot be merged into one response or database place per control system respectively. This happens due to the fact that the different colours of tokens that reside into those places of the control system do not represent the data that corresponds to the advices given by the system but to the IDs of the users that are related to those advices. Furthermore, the several response and database places of the different unidirectional steps are used to capture the update of the data at different instances of the system's functioning. For example, a token that is hosted in any response or database places of the net always represents the ID of a particular user, but according to in which response or database place it resides, the system provides the respective advice showing in that way that the data and the advices are updated after each user's action. Therefore, merging these places and using the colours of tokens to represent the advices of the system instead of having different places to represent them would not effectively reflect the update of the data as the colours of tokens would be always linked to particular advices during the operation of the system.

To complete the description of the control system, we explain the role of its transitions. There exist two types of transitions: the retrieve transitions (T_r) and the emptying transitions (T_{em}). The retrieve transitions have as pre-places the response places and as post-places the database places. These transitions are used to refill the database places with those tokens that have been removed from those places recently. The importance of the 'refill' process lies in the fact that the database places could be used more than once by a user if that user wants to repeat the same action in the future[1]. On the other hand, the emptying transitions have as pre-places the database places and the control place and as post-places the response places. The emptying transitions are responsible for the advice that is given to the user as their firing results in the depositing of tokens into the response places. Furthermore, the emptying transitions control the state of the control places, and by extension, the state of the next step transition, i.e., $t \in T_{ST}$. For example, this means that if an advice has not been given to a user, due to the fact that the system has not responded to a user's action yet (i.e., the respective emptying transition has not been executed), the user is not allowed to proceed to another task (because a token still resides in the control place) till he gets the advice, 'controlling' in that way (via the colour-sensitive inhibitor arc) the interaction.

Another important issue is the colour-sensitive inhibitor arc, the formal definition of which is given in section 3.3. Here, we simply describe the usefulness of the colour-sensitive inhibitor arc for the APN models, and the difference between that arc and the ordinary inhibitor arc [32]. The colour-sensitive inhibitor arc 'prevents' a user from taking the next action before the response of the system for his current one. We could say that this restriction (presence of colour-sensitive inhibitor arc in a model) synchronises the system's actions with those of the user (the user acts and the system reacts). The difference between the colour-sensitive inhibitor arc and the ordinary inhibitor arc is that a colour-sensitive inhibitor arc permits the firing of a transition for all the data types (colours) of the tokens except for those that reside in the inhibitor places of the transition. The example

[1]It should be noted that the retrieve transitions are not used to depict the update of data. The data is always the same for each unidirectional step. The update of data is represented by the use of the different unidirectional steps, which carry the appropriate information for each user's action at a particular time. Thus, a delayed execution of a retrieve transition does not affect the next action of a user.

of Figure 4.4 demonstrates the operation of a colour-sensitive inhibitor arc[1]. The transition t_4 can fire in the white mode but not in the black one. This is due to the fact that a black token is in place p_3 and blocks the execution of the transition for that particular colour of token. We could interpret this situation by saying that the 'white' user has obtained a response from the system (see place p_0) and can proceed to the next step, while the 'black' user still awaits for the system's advice. Contrary to the colour-sensitive inhibitor arc, the ordinary inhibitor arc blocks the firing of a transition for any token that resides in its pre-places. This happens in case that at least one token of any type is in the inhibitor place of the transition. Generally, the ordinary inhibitor arcs lack this extra flexibility that is not needed in the P/T nets since there is only one type of tokens.

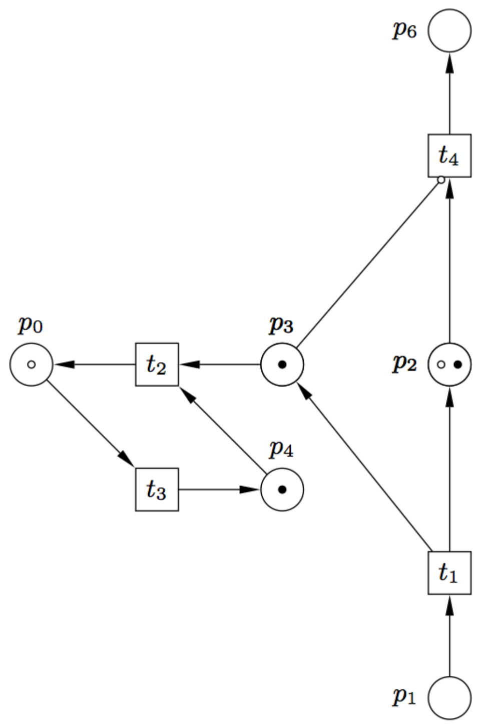


Figure 4.4: A colour-sensitive inhibitor arc.

4.2 Defining Building Block

In this section, we define the basic step net, which captures the intuitive concept of a unidirectional step with control system. The definition of that net follows

[1]For the sake of simplicity the control system of the second unidirectional step is omitted.

from the above discussion about the building block and its components: the step system and the control system. Regarding the functioning of the basic step nets, it is determined by the initial marking of the APN net that may be constructed out of several basic step nets. The marking of the APN net usually requires only the database places of the basic step nets to be marked. All the other places should be unmarked except for those step places of the net that belong to the initial marking. Finally, the initial state of an ambient system will be provided by a *root net*, which is described at the end of this section.

4.2.1 Basic Step Nets

At this point, the *basic step net*, which captures the structure of the introduced building block, is defined. Thereafter, the general structure of a basic step net is presented in Figure 4.5.

Definition 6 (basic step net)**.** *A* basic step net *is an APN net*

$$\mathcal{N}_S = (P_S, T_S, Pre_S, Post_S, I_S, Cl, C_S, K_S, M_0^S, G_S)$$

consisting of two parts: step system *and* control system *satisfying the following structural conditions:*

- *$P_S = P_{ST} \uplus P_{CS}$, where P_{ST} is a set of places of the step system and P_{CS} is a set of places of the control system. More specifically, $P_{ST} = \{p_s, p_f\}$ contains the* starting *(p_s) and* finishing *(p_f) place of the unique step transition t_s, while $P_{CS} = P_r \uplus P_d \uplus \{p_c\}$ is built out of three subsets such that (below $n \geq 1$):*

  1. *$P_r = \{p_r^1, \ldots, p_r^n\}$ is a set of* response *places.*
  2. *$P_d = \{p_d^1, \ldots, p_d^n\}$ is a set of* database *places.*
  3. *p_c is a unique* control *place of the finishing place p_f associated with transition t_s. We will denote this by $p_c = cp^{t_s}(p_f)$.*

- *$T_S = T_{ST} \uplus T_{CS}$, where T_{ST} is a set of transitions of the step system and T_{CS} is a set of transitions of the control system. More specifically, $T_{ST} = \{t_s\}$ contains a unique transition, called a* step *transition. $T_{CS} = T_r \uplus T_{em}$ is built out of two subsets:*

  1. *$T_r = \{t_r^1, \ldots, t_r^n\}$ is a set of* retrieve *transitions.*
  2. *$T_{em} = \{t_{em}^1, \ldots, t_{em}^n\}$ is a set of* emptying *transitions.*

- *Pre_S, $Post_S : T_S \to \mu P_S$ are such that:*

$$
\begin{aligned}
Pre_S(t^i_{em}) &= \{p_c, p^i_d\} && \textit{for } i = 1, \ldots, n.\\
Post_S(t^i_{em}) &= \{p^i_r\} && \textit{for } i = 1, \ldots, n.\\
Pre_S(t^i_r) &= \{p^i_r\} && \textit{for } i = 1, \ldots, n.\\
Post_S(t^i_r) &= \{p^i_d\} && \textit{for } i = 1, \ldots, n.\\
Pre_S(t_s) &= \{p_s\}.\\
Post_S(t_s) &= \{p_f, p_c\}.
\end{aligned}
$$

- $I_S = \varnothing$.

- *All the places except for the database places are unmarked. The database places are marked by at least one token of some allowed colour.*

$$
\begin{aligned}
M^S_0(p, c) &= 0 && \textit{, for every } p \in P_{ST} \cup P_r \cup \{p_c\} \textit{ and } c \in C_S(p).\\
M^S_0(p, c) &\neq 0 && \textit{, for every } p \in P_d \textit{ and some } c \in C_S(p).
\end{aligned}
$$

- $G_S = \{p_s\}$.

A similar structure to that of the basic step net has been previously defined in [62] representing a circuit in a tracker-bouncer architecture. The structural difference between the tracker-bouncer and the basic step net lies on the number of places and transitions used in these nets. The structure of tracker consists of two step systems and the bouncer only of two places and two transitions that are linked with each other in a cyclic way resembling in some sense the structure of a control system without a control place, but with only one response place, one database place, one retrieve transition and one emptying transition.

4.2.2 Root Net

Apart from the basic step net that is defined above, the root net that is important for the development of the APN models of the ambient systems is defined. The root net is significant for the composition of the APNs since it indicates the 'starting' state of the APN model of the examined ambient system. The composition of the APNs is described in section 4.3.

Definition 7 (root net)**.** *A* root net *is defined as an APN net that has two places: a place p that acts as both starting and finishing place of a 'collapsed' step and*

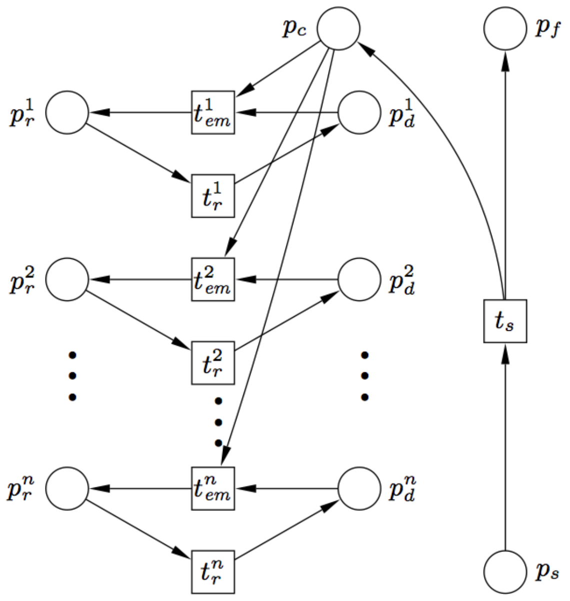


Figure 4.5: General structure of an unmarked basic step net.

its control place $p_c = cp^{\epsilon}(p)$, where ϵ denotes the 'collapsed' transition of the collapsed step.[1] *The root net will be denoted by:*

$$\mathcal{N}_{Rt} = (P_{Rt}, T_{Rt}, Pre_{Rt}, Post_{Rt}, I_{Rt}, Cl, C_{Rt}, K_{Rt}, M_0^{Rt}, G_{Rt}),$$

where

- $P_{Rt} = \{p, p_c\}$,
- $T_{Rt} = \varnothing$,
- Pre_{Rt}, $Post_{Rt}$ *are empty functions,*
- $I_{Rt} = \varnothing$,
- $G_{Rt} = \{p\}$.

A root net is used to provide the initial information of the system by marking each of its places with some chosen set of tokens. The place p will be marked with the set of initial users of the system. For example, in Figure 4.6(a, b), we have two users represented by black and white tokens. The control place, p_c, is provided to define the initial restrictions. The root net of Figure 4.6(a) states no

[1] We can extend our notation to: $p \in {}^{\bullet}\epsilon$ and $p \in \epsilon^{\bullet}$.

restrictions about any of the allowed users, while the root net of Figure 4.6(b) states restrictions for the 'black' user: this user will be stopped from proceeding any further. For example, in the model of the ambient garage this could mean that the owner of the car represented by the black token has not paid the fee and is not allowed to use the garage.

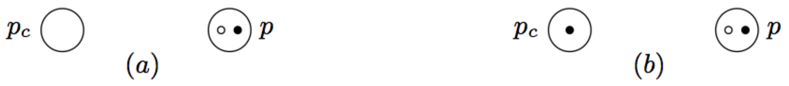


Figure 4.6: Root nets (a, b).

The use of the root net will become clear in the next section, where it is demonstrated, through examples, how a root net can be composed with other APN nets (i.e., composite step nets) in order to construct the APN model of the ambient systems.

4.3 Composing Ambient Petri Nets (APNs)

In the previous section, we introduced the building block (or basic step net) that is extracted from the intuitive modelling of the case studies (see section 2.3). As has already been mentioned, the basic step net is used for the compositional construction of APN models for the ambient systems.

Composition is a fundamental technique for constructing large system models out of simpler ones. A number of different approaches have been developed as regards the composition of Petri nets in the past years, based on composition of places or transitions [3, 57]. In this section, we introduce two composition operators for the APNs: the *forward* composition operator and the *backward* composition operator.

The composition of APNs that is described in the following paragraphs is based on the gluing of net places. More specifically, these two compositions describe how two APN nets can be glued with respect to the places of their *step systems*. Moreover, when any of these compositions is conducted, colour-sensitive inhibitor arcs are generated between the control places of the first net and the step transition of the second net.

After the definition of the composition operators, we provide representative examples that demonstrate how each of these composition operators of the APNs works. The repetitive use of the forward and the backward composition operators could result in the construction of the behavioural models for the ambient systems.

Using the two composition operators for the modelling of the APN nets of the ambient garage and the ambient conference room repetitively, we developed their structural and behavioural models, which are presented at the end of the chapter. Finally, we discuss the produced APN nets with respect to the scenarios given in section 2.3.

4.3.1 Forward Composition (FC) of APNs

In this section, we define the forward composition operator that enables the 'sequential' composition of two APN nets. By 'sequential', we mean the gluing of two APN nets that is conducted between the two nets only on one pair of gluing places, creating in that way a sequence of glued nets. An example could be a gluing that takes place between the finishing place p_f of the first net and the starting place p_s of the second net.

In addition to the definition of the forward composition operator, we introduce the forward composite step nets that result from the definition of the operator. Finally, for better understanding of the forward composition, we provide a representative example.

4.3.1.1 Defining Forward Composition Operator

The following definition describes how the forward composition operator is applied to the participating nets of the composition resulting in the creation of a composed APN net.

Definition 8 (forward composition).
Let us assume that

$$\mathcal{N} = (P, T, Pre, Post, I, Cl, C, K, M_0, G)$$

is an Ambient Petri Net such that, for every place $g \in G$ and $t \in {}^\bullet g$, there is a control place in $t^\bullet$ denoted by $cp^t(g)$. Moreover, let

$$\mathcal{N}_S = (P_S, T_S, Pre_S, Post_S, \varnothing, Cl, C_S, K_S, M_0^S, G_S)$$

be a basic step net (as in definition 6) such that $P \cap P_S = T \cap T_S = \varnothing$, and let $g \in G$ be a place satisfying: $C(g) = C_S(p_s) = C_{SET}$ and $K(g, c) = K_S(p_s, c)$, for every $c \in C_{SET}$.[1]

The forward composition *of $\mathcal{N}$ and $\mathcal{N}_S$ w.r.t. the gluing pair (g, p_s) of places is an Ambient Petri net*

$$\mathcal{N} \oplus^g \mathcal{N}_S = \mathcal{N}' = (P', T', Pre', Post', I', Cl, C', K', M_0', G'),$$

[1] Note that p_s is the only gluing place of $\mathcal{N}_S$.

where the different components are defined as follows:

- $P' = P \cup (P_S \setminus \{p_s\})$ *and* $T' = T \cup T_S$.
- $Pre', Post' : T' \to \mu P'$ *are defined by:*

$$Pre'(t)(p) \quad = \begin{cases} Pre(t)(p) & \textit{for } t \in T \textit{ and } p \in P, \\ Pre_S(t)(p) & \textit{for } t \in T_S \textit{ and } p \in P_S \setminus \{p_s\}, \\ Pre_S(t)(p_s) & \textit{for } t \in T_S \textit{ and } p = g, \\ 0 & \textit{otherwise.} \end{cases}$$

  and

$$Post'(t)(p) \quad = \begin{cases} Post(t)(p) & \textit{for } t \in T \textit{ and } p \in P, \\ Post_S(t)(p) & \textit{for } t \in T_S \textit{ and } p \in P_S \setminus \{p_s\}, \\ Post(t)(p_s) & \textit{for } t \in T \textit{ and } p = g, \\ 0 & \textit{otherwise.} \end{cases}$$

- $I' \subseteq P' \times T'$ *is defined by:* $I' = I \cup \{(cp^t(g), t_s) \mid t \in {}^\bullet g \textit{ in } \mathcal{N}\}$, *where* $t_s \in T_S$ *is* $\mathcal{N}_S$*'s unique step transition.*
- $C' : P' \cup T' \to Cl$ *is defined by:* $C'|_{P \cup T} = C$ *and* $C'|_{(P_S \setminus \{p_s\}) \cup T_S} = C_S$.
- K' *is defined by:* $K'|_{\widetilde{P}} = K$ *and* $K'|_{\widetilde{P_S \setminus \{p_s\}}} = K_S|_{\widetilde{P_S \setminus \{p_s\}}}$.
- $M'_0 \in \mu\widetilde{P'}$ *is defined by:* $M'_0|_{\widetilde{P}} = M_0$ *and* $M'_0|_{\widetilde{P_S \setminus \{p_s\}}} = M^S_0|_{\widetilde{P_S \setminus \{p_s\}}}$.
- $G' = G \cup \{p_f\}$; *moreover, the control places of the gluing places in* G' *are inherited from the component nets.*

Although in the above definition $\mathcal{N}$ is a general Ambient Petri Net, for the composition operator to work as desired, $\mathcal{N}$ should have a specific structure. More precisely, it should be a *forward composite step net.* In what follows, X_{Basic} denotes all the basic step nets, and X_{Root} all the root nets.

Definition 9 (forward composite step nets)**.** *The set of* forward composite step nets X^F_{Com} *is defined inductively as follows:*

- $X_{Root} \subset X^F_{Com}$.
- *If* $\mathcal{N} \in X^F_{Com}$ *and* $\mathcal{N}_S \in X_{Basic}$, *then* $\mathcal{N} \oplus^g \mathcal{N}_S \in X^F_{Com}$.

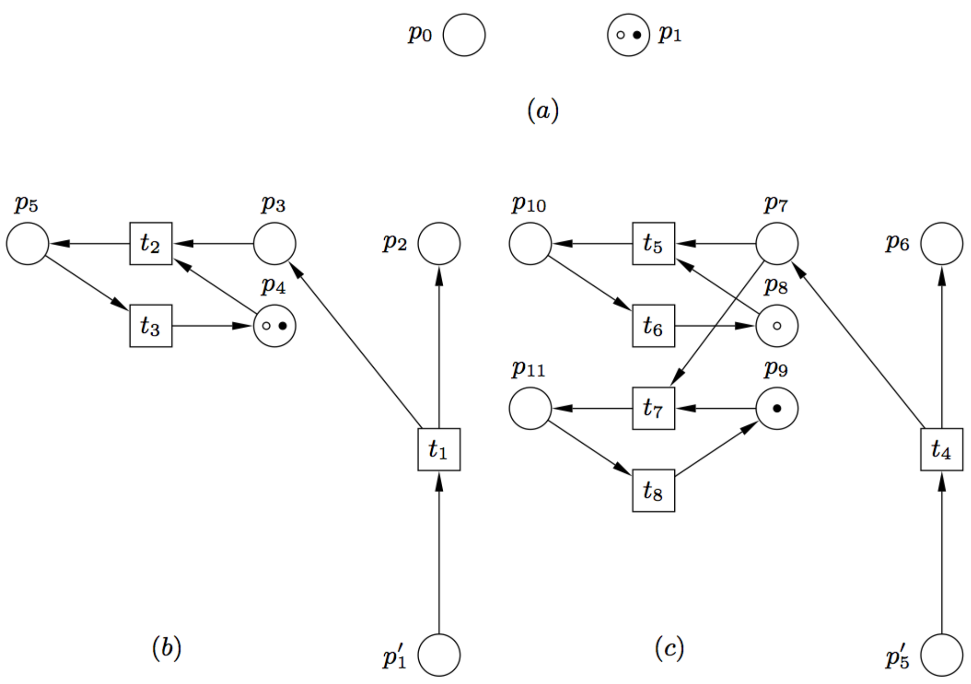


Figure 4.7: $\mathcal{N}_1 \in X_{Root}$ (a), $\mathcal{N}_2 \in X_{Basic}$ (b) and $\mathcal{N}_3 \in X_{Basic}$ (c).

Note that for every gluing place of a net $\mathcal{N} \in X^F_{Com}$ there will be at least one control place. Indeed, the only gluing place of a root net has its unique control place associated with collapsed transition ϵ, and this transition can be considered both as pre- and post-transition of this gluing place. Also, the place added to the set of gluing places after a forward composite net is extended by a basic step net (p_f of the basic step net) comes with its control place as well. It is noted that the forward composition can be applied on the same gluing place of a composite step net several times extending the net, from that gluing place, by a basic step net each time. Finally, applying the forward composition, we can produce only directed tree graphs, such as path (or linear tree) graphs, caterpillar trees and star graphs. On the contrary, the use of forward composition cannot result in polytree graphs, which can be produced with the help of the second composition operator that will be presented later on.

In the forward composite step nets, there is always a unique control place for every gluing place of $\mathcal{N} \in X^F_{Com}$. As a result, the added set of colour-sensitive inhibitor arcs (in definition 8) will be just a singleton set. However, this will change when we introduce the second composition operator allowing transitions to 'come back' to some of the existing places or to link not connected places.

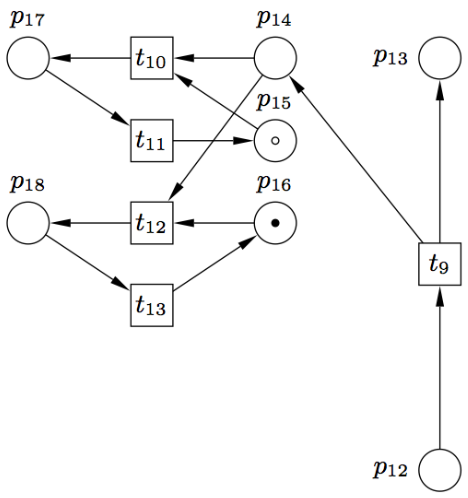


Figure 4.8: $\mathcal{N}_4 \in X_{Basic}$, an example of a basic step net.

4.3.1.2 An Example of Forward Composition of APNs

Now, we provide an example of a forward composition in order to explain definition 8.

Suppose that we have two nets, $\mathcal{N}_1 \in X_{Root}$ and $\mathcal{N}_2 \in X_{Basic}$, as shown in Figure 4.7$(a, b)$. We can (forward) compose them by choosing (p_1, p_1') as a gluing pair of places and constructing $\mathcal{N}_1 \oplus^{p_1} \mathcal{N}_2$. In the process of composing them, a colour-sensitive inhibitor arc will be inserted between place p_0 and transition t_1. The resulting net will belong to the set X^F_{Com}. We can extend it further by a step defined by the net $\mathcal{N}_3 \in X_{Basic}$ (see Figure 4.7(c)), by gluing places p_2 and p_5'. The resulting net will contain an inserted colour-sensitive inhibitor arc between p_3 and t_4. Again the resulting net belongs to the set X^F_{Com}. Let's call it $\mathcal{N}$. At this point, the forward composite step net $\mathcal{N}$ has a set of gluing points with three elements: $\{p_1, p_2, p_6\}$. Any of them can be a starting point of future extensions. Suppose we select p_2 as the next gluing point with a net $\mathcal{N}_4$ of Figure 4.8 describing the next step. We will use (p_2, p_{12}) as the next gluing pair.

The resulting net is shown in Figure 4.9. The two component nets, $\mathcal{N}$ and $\mathcal{N}_4$, can be composed according to definition 8, and their composition fuses the places of the gluing pair and removes p_{12}. A new colour-sensitive inhibitor arc is added between the control place $p_3 = cp^{t_1}(p_2)$ and the step transition t_9. Note that the set ${}^\bullet p_2$ has only one element, t_1, and so only one inhibitor arc is added. Finally, p_{13}, which is the p_f place of the net $\mathcal{N}_4$, is added to the gluing places of the resulting net.

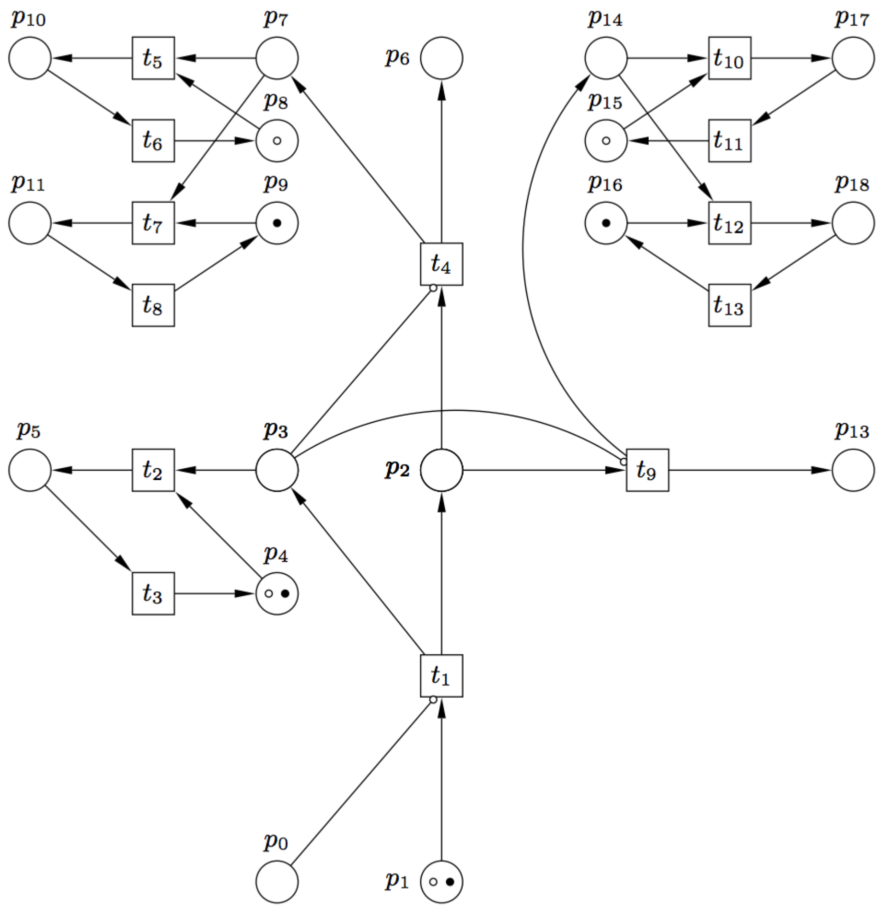


Figure 4.9: A forward composition of a forward composite net $\mathcal{N}$ and a basic step net $\mathcal{N}_4$.

4.3.2 Backward Composition of APNs

Having defined the operator for the forward composition of APNs, one can notice that it does not allow a step to return to an already existing place. Therefore, we introduce another composition operator which will use two gluing pairs of places. This operator also enables the creation of 'cyclic' APN nets.

Definition 10 (cyclic/reversible APN net)**.** *A marked Ambient Petri Net with initial marking M_0, is cyclic/reversible if from every reachable marking M it is possible to retrieve M_0 (i.e., $M \in [M_0\rangle \implies M_0 \in [M\rangle$).*

Apart from the definition of the backward composition, we define all those terms that are needed for that composition. Finally, we provide three examples of backward composition for better comprehension of the composition. It should be mentioned that the first example refers to a new net, the bidirectional step.

4.3.2.1 Defining Backward Composition Operator

In this section, the backward composition of the APNs is introduced. As was mentioned earlier, the basic step net has only one gluing place. Hence, to define the backward composition operator, we extend the definition of the basic step net. After that, we proceed with the definition of the composition.

Definition 11 (one step net)**.** *A* one step net *is defined as a basic step net, with the only difference, namely its set of gluing places contains both the starting and finishing place: $G_S = \{p_s, p_f\}$.*

We will denote by X_{One} the set of all one step nets.

Definition 12 (backward composition)**.**
Let $\mathcal{N}$ be an Ambient Petri Net, and $\mathcal{N}_S$ be a one step net, as in definitions 8 and 11, respectively (the two nets should have disjoint sets of places and transitions). Moreover, let $g_1, g_2 \in G$ be places satisfying:

$$C(g_1) = C(p_s) = C^1_{SET}, \qquad C_S(g_2) = C_S(p_f) = C^2_{SET}$$

as well as, for all $c \in C^1_{SET}$ and $c' \in C^2_{SET}$,

$$K(g_1, c) = K_S(p_s, c), \qquad K(g_2, c') = K_S(p_f, c').$$

The backward composition *of $\mathcal{N}$ and $\mathcal{N}_S$ w.r.t. the gluing pairs (g_1, p_s) and (g_2, p_f) of places is an Ambient Petri Net:*

$$\mathcal{N} \oplus^{g_1}_{g_2} \mathcal{N}_S = \mathcal{N}' = (P', T', Pre', Post', I', Cl, C', K', M'_0, G')$$

where the different components are defined as follows:

- *$P' = P \cup (P_S \setminus \{p_s, p_f\})$ and $T' = T \cup T_S$.*
- *Pre', $Post' : T' \to \mu P'$ are defined by:*

$$Pre'(t)(p) = \begin{cases} Pre(t)(p) & \text{for } t \in T \text{ and } p \in P, \\ Pre_S(t)(p) & \text{for } t \in T_S \text{ and } p \in P_S \setminus \{p_s, p_f\}, \\ Pre_S(t)(p_s) & \text{for } t \in T_S \text{ and } p = g_1, \\ Pre(t)(p_f) & \text{for } t \in T \text{ and } p = g_2, \\ 0 & \text{otherwise.} \end{cases}$$

and

$$Post'(t)(p) = \begin{cases} Post(t)(p) & \textit{for } t \in T \textit{ and } p \in P, \\ Post_S(t)(p) & \textit{for } t \in T_S \textit{ and } p \in P_S \setminus \{p_s, p_f\}, \\ Post(t)(p_s) & \textit{for } t \in T \textit{ and } p = g_1, \\ Post_S(t)(p_f) & \textit{for } t \in T_S \textit{ and } p = g_2, \\ 0 & \textit{otherwise.} \end{cases}$$

- $I' \subseteq P' \times T'$ *is defined by:*

  $$I' = I \cup \{(cp^t(g_1), t_s) \mid t \in {}^{\bullet}g_1 \textit{ in } \mathcal{N}\} \cup \{(cp^{t_s}(p_f), t) \mid t \in g_2^{\bullet} \textit{ in } \mathcal{N}'\},$$

  where $t_s \in T_S$ *is* $\mathcal{N}_S$*'s unique step transition.*

- $C' : P' \cup T' \to Cl$ *is defined by:* $C'|_{P \cup T} = C$ *and* $C'|_{(P_S \setminus \{p_s, p_f\}) \cup T_S} = C_S$.

- K' *is defined by:* $K'|_{\widetilde{P}} = K$ *and* $K'|_{\widetilde{P_S \setminus \{p_s, p_f\}}} = K_S|_{\widetilde{P_S \setminus \{p_s, p_f\}}}$.

- $M'_0 \in \mu\widetilde{P'}$ *is defined by:* $M'_0|_{\widetilde{P}} = M_0$ *and* $M'_0|_{\widetilde{P_S \setminus \{p_s, p_f\}}} = M_0^S|_{\widetilde{P_S \setminus \{p_s, p_f\}}}$.

- $G' = G$*; moreover, the control places of the gluing places in* G' *are inherited from* $\mathcal{N}$ *except for* g_2 *that has an additional control place* $cp^{t_s}(g_2)$ *which was* $cp^{t_s}(p_f)$ *in* $\mathcal{N}_S$.[1]

The basic difference between definitions 8 and 12 is in the generation of the colour-sensitive inhibitor arcs and the set of gluing places in the resulting net. In the backward composition, the set of colour-sensitive inhibitor arcs includes the colour-sensitive inhibitor arcs of $\mathcal{N}$ together with the two sets of colour-sensitive inhibitor arcs created during the composition. One of the sets is the same as in definition 8; it contains arcs that are needed to make sure that the user takes instructions (from the system) after executing some transition t and proceeding to execute t_s. Now, more colour-sensitive inhibitor arcs are needed in order for the user to take advices (from the system) after executing transition t_s and proceed to the execution of some subsequent transition t. This is due to the fact that backward composition introduces 'cycles' into the resulting net, and we not only need to consider execution sequences containing tt_s, but as well execution sequences containing t_st. The gluing places of $\mathcal{N}'$ are simply the gluing places of $\mathcal{N}$. Note that we do not require that g_1 and g_2 are different places. If they coincide, then both p_s and p_f can be glued with the same place ($g_1 = g_2$),

[1] So now the user executing some existing transition $t \in g_2^{\bullet}$ in $\mathcal{N}'$ will not be able to proceed without 'taking the instructions' from the system following the execution of transition t_s.

creating a self-loop. Furthermore, it is noted that once a backward composition has been performed on specific gluing pairs, e.g. (g_1, p_s) and (g_2, p_f), then no other backward composition can be applied on these gluing pairs, but it can be conducted on other gluing pairs that include places of these pairs. At this point, it should be mentioned that the use of forward and backward compositions can construct several types of directed graphs. For instance, the combination of these two compositions can result in cyclic graphs, directed acyclic graphs (DAGs) (e.g. rhombus graph) and polytree graphs (e.g. N-graph) considering an ordering in the way that these compositions are conducted. Examples of combining the two composition operators in order to create some of the aforementioned types of graphs are illustrated in section 4.3.2.2 through the construction of cyclic graphs (e.g. bidirectional step), polytree graphs (N-graph), etc.

In the above definition, as in definition 8, $\mathcal{N}$ is a general APN net. However, we are really interested in nets which can be derived compositionally forming the set X_{Com} of composite step nets.

Definition 13 (composite step nets)**.** *The set of* composite step nets X_{Com} *is defined inductively as follows:*

- $X^F_{Com} \subset X_{Com}$.
- *If* $\mathcal{N} \in X_{Com}$ *and* $\mathcal{N}_S \in X_{Basic}$*, then* $\mathcal{N} \oplus^g \mathcal{N}_S \in X_{Com}$*, provided that the forward composition is applied.*
- *If* $\mathcal{N} \in X_{Com}$ *and* $\mathcal{N}_S \in X_{One}$*, then* $\mathcal{N} \oplus^{g_1}_{g_2} \mathcal{N}_S \in X_{Com}$*, provided that the backward composition is applied.*

4.3.2.2 Examples of Backward Composition of APNs

As has already been mentioned above, this section describes three different examples of the backward composition. Finally, the *bidirectional step* is introduced through the demonstration of the first example.

4.3.2.2.1 Example of a Bidirectional Step

The backward composition of nets enables the creation of *bidirectional steps*, which are composed of two unidirectional steps with control systems.

A bidirectional step allows the user to retrieve the previous state by undoing the last action (note that a unidirectional step does not allow the retrieval of previous states). The construction of bidirectional steps turned out to be necessary for the modelling of ambient systems. The simplest way to create a bidirectional step is by gluing two unidirectional steps with control system w.r.t. the places of their step systems (see definition 6).

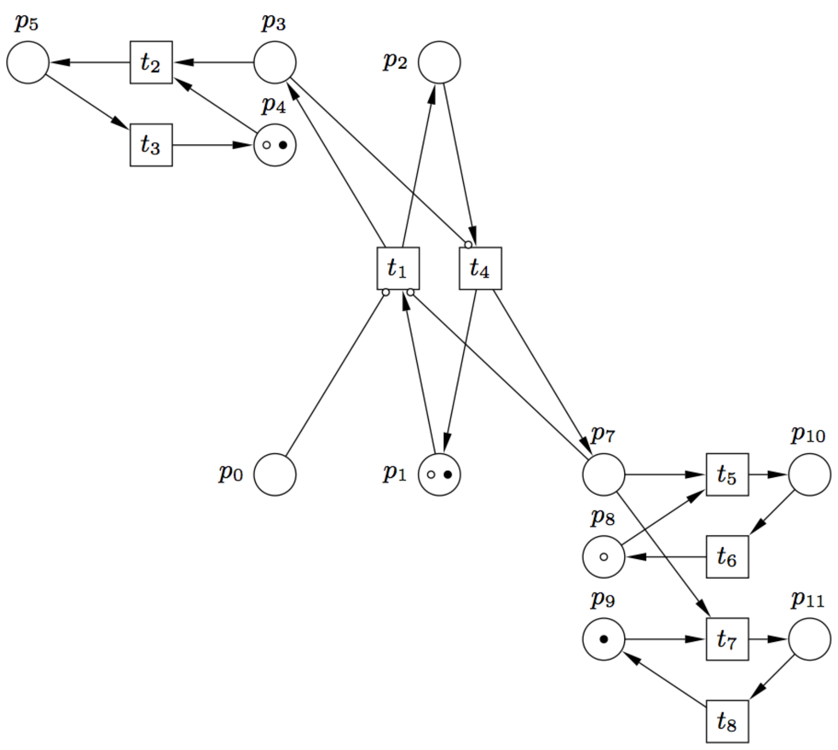


Figure 4.10: A bidirectional step resulting from the composition of unidirectional steps of Figure 4.2.

An example of a bidirectional step is given in Figure 4.10. The net of this figure is generated by following the definition of backward composition for the forward composite net $\mathcal{N} = \mathcal{N}_1 \oplus^{p_1} \mathcal{N}_2$ (see Figure 4.7 for nets $\mathcal{N}_1$ and $\mathcal{N}_2$) and $\mathcal{N}_3$ from Figure 4.7 treated as a one step net. For the composition of these two nets, $(g_1, p_s) = (p_2, p'_5)$ and $(g_2, p_f) = (p_1, p_6)$[1] were chosen as the two gluing pairs ($\mathcal{N} \oplus^{p_2}_{p_1} \mathcal{N}_3 = \mathcal{N}'$). In this case, the places that are removed during the composition of the two nets are p'_5 and p_6 respectively. Furthermore, the set of colour-sensitive inhibitor arcs of the resulting net consists of the colour-sensitive inhibitor arcs that connect the transitions t_4 and t_1 with the places p_3 and p_7 respectively, together with the colour-sensitive inhibitor arc (between t_1 and p_0) that was added after $\mathcal{N}_1$ and $\mathcal{N}_2$ were composed. Indeed, knowing that p_f is glued with g_2, $p_f = p_6$, $t_s = t_4$, $g_1 = p_2$, $g_2 = p_1$ and ${}^\bullet p_2 = \{t_1\} = p_1^\bullet$ in $\mathcal{N}$ (and $\mathcal{N}'$), we

[1] p_s and p_f are the starting and finishing places of $\mathcal{N}_3$.

obtain, from definition 12, the following:

$$\begin{array}{rcccl} I' & = & I & \cup & \{(cp^t(g_1), t_s) \mid t \in {}^\bullet g_1 \text{ in } \mathcal{N}\} \cup \{(cp^{t_s}(p_f), t) \mid t \in g_2^\bullet \text{ in } \mathcal{N}'\} \\ & = & \{(p_0, t_1)\} & \cup & \{(cp^t(p_2), t_4) \mid t \in {}^\bullet p_2 \text{ in } \mathcal{N}\} \cup \{(cp^{t_4}(p_6), t) \mid t \in p_1^\bullet \text{ in } \mathcal{N}'\} \\ & = & \{(p_0, t_1)\} & \cup & \{(cp^{t_1}(p_2), t_4)\} \cup \{(cp^{t_4}(p_6), t_1)\} \\ & = & \{(p_0, t_1)\} & \cup & \{(p_3, t_4)\} \cup \{(p_7, t_1)\} \end{array}$$

Notice that the place p_1 of the resulting net has two associated control places: p_0 and p_7. Place p_0 'controls' the movements of the users starting from place p_1 after they arrive there by executing an empty transition ϵ ($p_0 = cp^\epsilon(p_1)$). Place p_7, on the other hand, was a control place of the place p_6 in the net $\mathcal{N}_3$ ($cp^{t_4}(p_6)$). Now, after composing $\mathcal{N}$ and $\mathcal{N}_3$, where p_1 and p_6 were glued, p_7 'controls' the movements of the users starting from place p_1 after they arrive there by executing transition t_4 ($p_7 = cp^{t_4}(p_1)$). So, $p_7 = cp^{t_4}(p_6)$ in $\mathcal{N}_3$, but $p_7 = cp^{t_4}(p_1)$ in $\mathcal{N}'$.

4.3.2.2.2 Example of a N-graph

An example of a N-graph is given in Figure 4.11. The net of this figure is produced by following the definition of backward composition for the net $\mathcal{N}$, which is treated as a composite net that consists of two parallel forward composite step nets (N_6 and N_7), and the N_4 from Figure 4.8. The forward composite step nets N_6 and N_7 are generated by forward composing the nets N_2 and N_3 of Figure 4.7 with the root nets of Figures 4.7(a) and 4.6(b) respectively. For the backward composition of N and N_4, $(g_1, p_s) = (p, p_{12})$ and $(g_2, p_f) = (p_2, p_{13})$[1] were chosen as the two gluing pairs ($\mathcal{N} \oplus^p_{p_2} \mathcal{N}_4 = \mathcal{N}'$). Applying definition 12 and gluing the two nets w.r.t. these places results in N'. By carrying out the composition, the places p_{12} and p_{13} are removed from the structure of the resulting net as they are fused with the places p and p_2 respectively. The set of transitions of the resulting net consists of the transitions of $\mathcal{N}$ and $\mathcal{N}_4$. We recall that p_f is glued with g_2 and $p_f = p_{13}$. Since $t_s = t_9$, $g_1 = p$, $g_2 = p_2$, ${}^\bullet p = \{\epsilon\}$ and $p_2^\bullet = \varnothing$ in $\mathcal{N}$ (and $\mathcal{N}'$), the set of colour-sensitive inhibitor arcs of the resulting net can be computed as follows:

$$\begin{array}{rcccl} I' & = & I & \cup & \{(cp^t(g_1), t_s) \mid t \in {}^\bullet g_1 \text{ in } \mathcal{N}\} \cup \{(cp^{t_s}(p_f), t) \mid t \in g_2^\bullet \text{ in } \mathcal{N}'\} \\ & = & I & \cup & \{(cp^t(p), t_9) \mid t \in {}^\bullet p \text{ in } \mathcal{N}\} \cup \{(cp^{t_9}(p_{13}), t) \mid t \in p_2^\bullet \text{ in } \mathcal{N}'\} \\ & = & I & \cup & \{(cp^\epsilon(p), t_9)\} \cup \varnothing \\ & = & I & \cup & \{(p_c, t_9)\}, \end{array}$$

where $I = \{(p_0, t_1), (p_c, t_4)\}$.

[1] p_s and p_f are the starting and finishing places of $\mathcal{N}_4$.

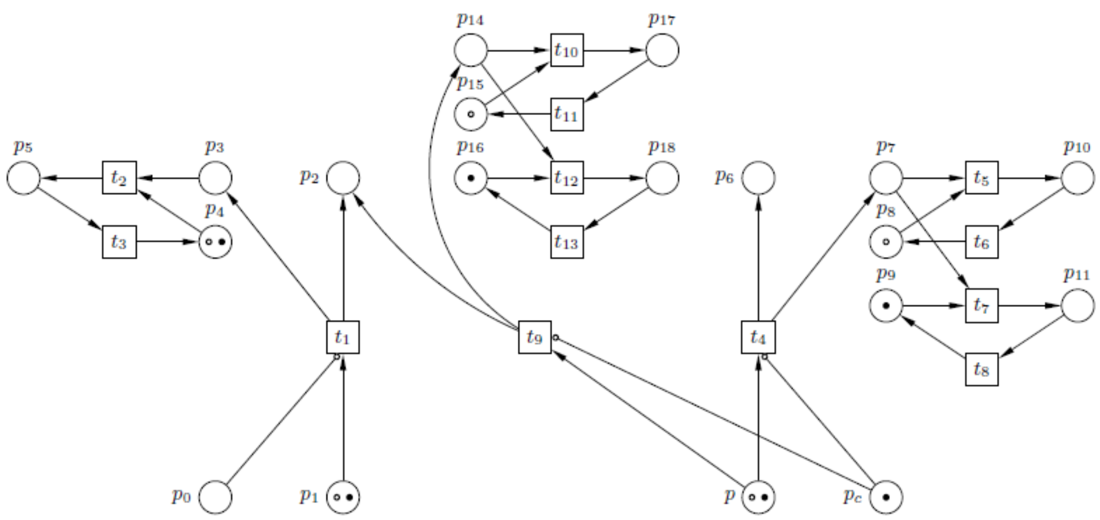


Figure 4.11: A N-graph.

4.3.2.2.3 Another Example of Backward Composition

We now present a more generic example of the backward composition.

Let $\mathcal{N}$ be a composite step net obtained by using the forward composition operator on nets $\mathcal{N}_1$, $\mathcal{N}_2$ and $\mathcal{N}_3$ from Figure 4.7: $\mathcal{N} = (\mathcal{N}_1 \oplus^{p_1} \mathcal{N}_2) \oplus^{p_2} \mathcal{N}_3$. A net similar to $\mathcal{N}$ appears in Figure 4.3 (we only need to add an empty place p_0 and a colour-sensitive inhibitor arc between t_1 and p_0). We now backward compose $\mathcal{N}$ with net $\mathcal{N}_4$ from Figure 4.8 treated as one step net, obtaining $\mathcal{N}'$. The two chosen gluing pairs for these two nets are $(g_1, p_s) = (p_2, p_{12})$ and $(g_2, p_f) = (p_1, p_{13})$[1]. Applying definition 12 and gluing the two nets w.r.t. these places results in the net of Figure 4.12. By carrying out the composition, the places p_{12} and p_{13} are removed from the structure of the resulting net as they are fused with the places p_2 and p_1 respectively. The set of transitions of the resulting net consists of the transitions of $\mathcal{N}$ and $\mathcal{N}_4$. We recall that p_f is glued with g_2 and $p_f = p_{13}$. Since $t_s = t_9$, $g_1 = p_2$, $g_2 = p_1$ and ${}^\bullet p_2 = \{t_1\} = p_1^\bullet$ in $\mathcal{N}$ (and $\mathcal{N}'$), the set of colour-sensitive inhibitor arcs of the resulting net can be computed as follows:

$$\begin{aligned}
I' &= I \;\cup\; \{(cp^t(g_1), t_s) \mid t \in {}^\bullet g_1 \text{ in } \mathcal{N}\} \cup \{(cp^{t_s}(p_f), t) \mid t \in g_2^\bullet \text{ in } \mathcal{N}'\} \\
&= I \;\cup\; \{(cp^t(p_2), t_9) \mid t \in {}^\bullet p_2 \text{ in } \mathcal{N}\} \cup \{(cp^{t_9}(p_{13}), t) \mid t \in p_1^\bullet \text{ in } \mathcal{N}'\} \\
&= I \;\cup\; \{(cp^{t_1}(p_2), t_9)\} \cup \{(cp^{t_9}(p_{13}), t_1)\} \\
&= I \;\cup\; \{(p_3, t_9)\} \cup \{(p_{14}, t_1)\},
\end{aligned}$$

[1] p_s and p_f are starting and finishing places of $\mathcal{N}_4$ here.

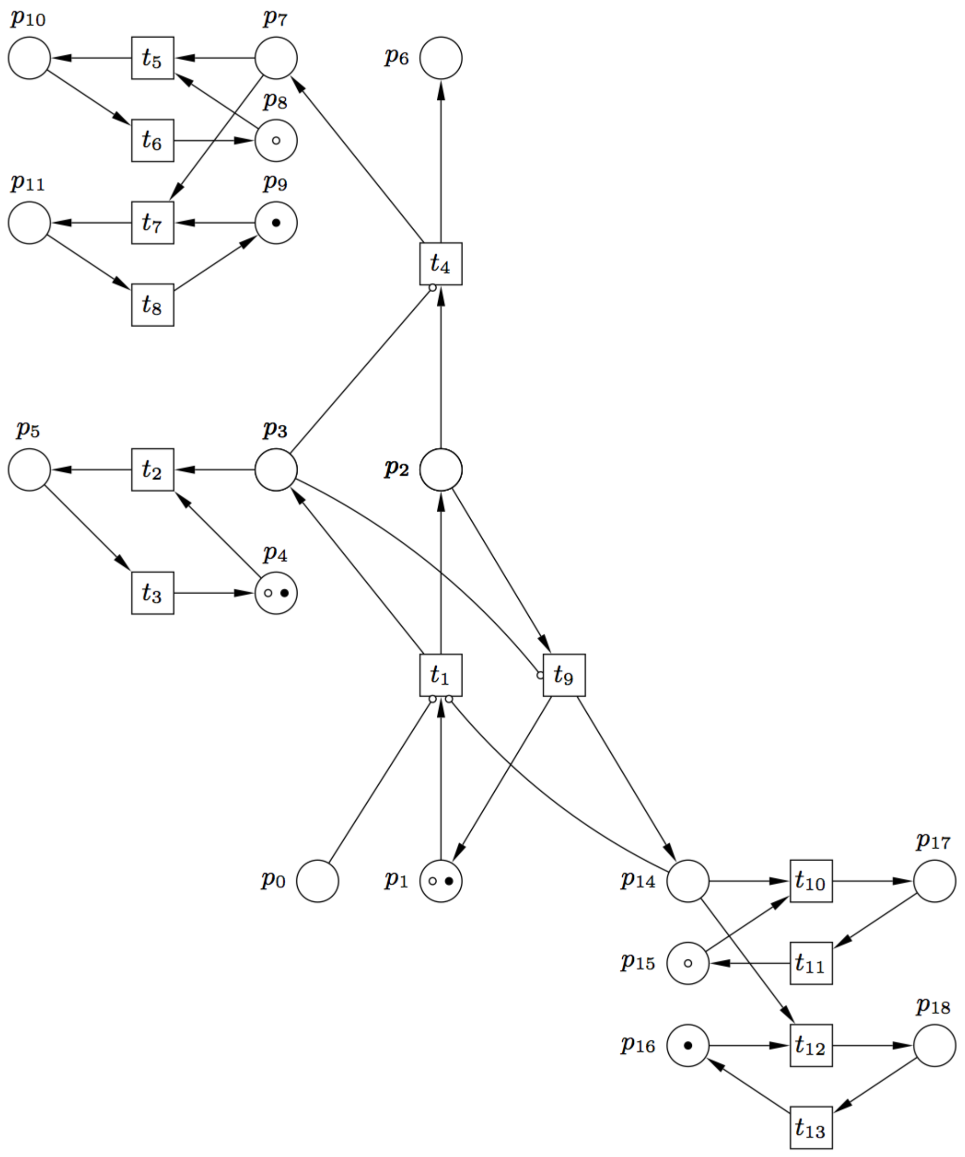


Figure 4.12: A composition of a forward composite step net and a one step net.

where $I = \{(p_0, t_1), (p_3, t_4)\}$. Note that $p_{14} = cp^{t_9}(p_{13})$ in $\mathcal{N}_4$, but now, in the context of $\mathcal{N}'$, $p_{14} = cp^{t_9}(p_1)$.

4.4 General Properties of composite APNs

Having defined the class of APNs and its composition operators that can be used for construction of composite APN nets, we now present some qualitative properties of these nets. These properties, which are presented below, are considered as general in the sense that they are meaningful for any concurrent system, not only for those modelled with APNs. We start the demonstration of the general properties with the description of the basic behavioural properties, which are followed by the discussion about some indicative structural properties of the APNs.

4.4.1 Behavioural Properties

The first behavioural property that is discussed is that of *liveness*, which is related to the potential fireability in all the reachable markings of an APN net.

Definition 14 (Liveness). *Let N be an APN net and $t \in T_N$.*

- *t is called live iff $\forall M \in [M_N\rangle\ \exists M' \in [M\rangle$ such that t is M'-enabled.*
- *N is called live iff $\forall t \in T_N : t$ is live.*

Additionally to the above definition, we extend the notion of *liveness* to the markings of an APN.

Definition 15 (Liveness of marking). *A marking M of an APN net N is live iff $\forall t \in T_N\ \exists M' \in [M\rangle$ such that t is M'-enabled.*

Then from the above definitions of *liveness*, we get the following.

Lemma 1 (Live APN net). *An APN net N is live iff all markings $M \in [M_N\rangle$ are live.*

Proof. N is live $\iff \forall t \in T_N : t$ is live $\iff \forall t \in T_N\ \forall M \in [M_N\rangle\ \exists M'$ such that t is $M' - enabled \iff \forall M \in [M_N\rangle\ M$ is live. □

The next behavioural property, i.e., *boundedness*, characterises the finiteness of the places and by extension the finiteness of the state space of the APNs.

Definition 16 (Boundedness of place). *A place p is k-bounded if there exists a natural number k, which represents an upper bound for the number of tokens on this place in all reachable markings of the Ambient Petri net: $\exists k \in \mathbb{N}_0 : \forall M \in [M_0\rangle : M((p, g)) \leq k$.*

To extend the notion of the *boundedness* to an APN net, we use the following definition:

Definition 17 (Boundedness of an APN net)**.** *An APN net N is*

- *k-bounded if all its places are k-bounded and*
- *structurally bounded if it is bounded in any initial marking.*

Finally, the last behavioural property that is presented is that of the *reversibility.* This property characterises the ability of recovering the initial marking from any reachable marking of an APN net. The definition of reversible APN nets has already been given in definition 10 as the notions of reversible and cyclic APNs are considered the same. Thus, in other words, reversibility implies a cyclic behaviour of a system.

4.4.2 Structural Properties

The static nature of APNs can be analysed through the structural properties, which depend on the arrangement of places, transitions and arcs of these nets. These properties characterise the structure of the APN nets independently of their marking. In this section, we discuss a few indicative structural properties of APNs.

The first structural property that is presented is that of *purity*, which indicates the absence of self-loops in the APN nets.

Definition 18 (Pure APN net)**.** *An APN net is pure if*

$$\forall p \in P, \forall t \in T : \mathit{Pre}(t)(p) \neq 0 \implies \mathit{Post}(t)(p) = 0.$$

From the above definition, we get the following:

Lemma 2 (Pure composite APN net)**.** *A composite APN net N, which is made out of forward or backward compositions, is pure.*

Proof. From definition 6, it follows that a basic step net has no places and transitions that are directly connected in both directions. So, according to definition 18, this net is pure. Furthermore, from definitions 8 and 12, it turns that a composite APN net N that consists of several basic step nets is pure as well since no new direct arcs are added between the places and the transition of the produced net during the application of forward or backward compositions. □

Finally, the last structural property that is presented is *ordinariness*, which denotes that the weights of all the arcs of an APN net are equal to one.

Definition 19 (Ordinary APN net)**.** *An APN net is ordinary if* $\forall p \in P, \forall t \in T:$ $Pre(t)(p) \neq 0 \implies Pre(t)(p) = 1 \wedge Post(t)(p) \neq 0 \implies Post(t)(p) = 1$.

From the above definition, we get the following:

Lemma 3 (Ordinary composite APN net)**.** *A composite APN net N, which is made out of forward or backward compositions, is ordinary.*

Proof. From definitions 3 and 6, it follows that all the pre and post arcs of a basic step net have weights equal to one as $Pre(t)(p)$ and $Post(t)(p)$ take the value one for every existing arc (see p. 36). So, according to definition 19, this net is ordinary. Furthermore, from definitions 8 and 12, it turns that a composite APN net N that consists of several basic step nets is ordinary as well since all the pre and post arcs of the produced net consists of the already existed pre and post arcs of the composed basic step nets, which all have weights equal to one. □

4.5 APN Models of Case Studies

In section 4.3, we discussed the definition of the compositions of the APNs and we demonstrated their applicability through some generic examples. Furthermore, in section 4.4, we defined some general properties of composite APN nets. Primary concern of this section is the implementation of the APN models for the ambient garage and the ambient conference room. For the modelling of these systems, the step-modelling approach, the APN class and the compositions of the APNs are applied considering both the specification and the system features.

Modelling precisely these systems by following all the requirements of their specifications, it is noticed that the representation of the behavioural model of the whole system is often impractical or not possible. Even in cases that this is feasible, the analysis of these models is a laborious process due to the chaotic and incomprehensible design. Therefore, the modelling of the ambient systems focuses on a particular aspect of these systems, the interactivity (see section 2.2.5). The produced APN models capture the interactive behaviour of the ambient systems, which is expressed through system components like the sensors, the output devices and the context-awareness. In other words, the APN models represent how all the above features contribute to the interaction of the ambient systems with the users.

4.5.1 APN Model of Ambient Garage

To model the behaviour of the Ambient Guidance Systems, we use a representative system of this category, the ambient garage. As is already known, the behaviour of the Ambient Guidance Systems depends on their spatial structure and the context provided throughout their operation. Furthermore, the main concern of the Ambient Guidance Systems is to guide the users, helping in that way, towards the accomplishment of their goals (e.g. to reach a destination). In the case of the ambient garage, the goal of each user is to park his car to the allocated parking bay.

For that reason, the modelling of the ambient garage focuses on the system interactivity in terms of guidance and redirection of the users and on how the actions taken by the users affect the system's response or the progress of the users' goal. As a result, all the activities before the entry of the cars to the garage and after their exit from it are not considered in the modelling process. Thus, to capture the interactivity of the system, the sensors, the output displays and the context-awareness are designed, as these system components express the interactive nature of the garage.

In addition, for the modelling of the ambient garage, a topological plan of the garage is provided. This topological plan provides all the necessary information about the spatial structure of the ambient garage and specifies the system components that are modelled. The topology of the ambient garage is presented in Figure 4.13. It should be noted that the interactivity of the ambient garage is modelled in a similar way for any topological plan. The only difference, regarding the modelling of ambient garages with different topologies, lies in the size and the resources (colours of tokens) of the produced APN models.

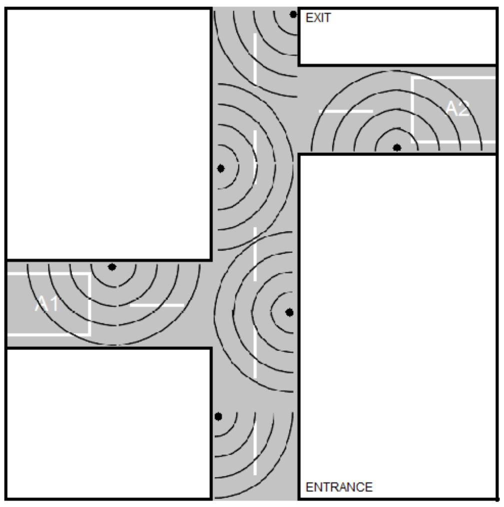


Figure 4.13: Topological plan of ambient garage.

Figure 4.13 shows that the garage is equipped with an entrance, an exit, six sensors that are used for the detection of the users and two parking bays. The number of the parking bays implies that maximum two users can interact with the system during its operation. Furthermore, it is assumed that every area that is covered by a sensor has a public display that shows the advices of the system to the users. These advices are determined with respect to the current position of the detected users and the location of their allocated parking bays, providing always the shortest path. In case that a user misses or ignores one of the advices, the system redirects him through a new advice that is provided at his new position.

Having described the main points of the modelling process of the ambient garage, we proceed with the implementation of the APN model. Applying the step-modelling approach, we start with the identification of all the possible actions of all the users, within the garage, in accordance with the topology and the specification of the garage. The actions of the users are as follows:

(*i*) Move from the entrance to the first junction,

(*ii*) Move from the first junction to the parking bay $A1$,

(*iii*) Move from $A1$ to the first junction,

(*iv*) Move from the first junction to the second one,

(*v*) Move from the second junction to the parking bay $A2$,

(*vi*) Move from $A2$ to the second junction,

(*vii*) Move from the second junction to the first one,

(*viii*) Move from the second junction to the exit.

As is noticed from the above actions, there exist three cases where the users can return to a previously visited area. These cases enable the users either to exit the garage after they have successfully parked their cars to the parking bays or to get back on the right track after a missed or an ignored advice. Finally, actions like moving from the first junction to the entrance or moving from the exit to the second junction are not allowed. This follows from the specification of the garage, which mentions that the users cannot exit from the entrance and cannot enter from the exit respectively. For the representation of these actions, we use the basic step nets. More specifically, these actions are represented by the step transitions of the basic step nets.

Having explained the step transitions of the basic step nets that correspond to the actions listed above, we describe the step places of the respective actions.

As is mentioned earlier, the users can move from an area to another and vice verse, excluding the areas of the entrance and exit where the users can move in one direction only. Each of these areas is monitored by a sensor that detects the motion of the user. Thus, in the APN model of the ambient garage, the step places represent the sensors that monitor the areas of the garage.

Afterwards, the control system of the basic step nets is described. The control system represents the database and the output displays of the system. The database consists of the database places, where all the information about the users is held. The number of the database places is determined according to the number of different advices that the system can potentially provide for a particular action of the users. On the contrary, the output display of each area is represented by the response places of the control system. Each of these response places represents a specific advice that is given to a user and is shown on the public display.

Having described how the sensors and the output displays are presented in the APN model, we continue with a discussion about the representation of the context-awareness of the ambient garage. The context-awareness is expressed through the execution of the emptying transitions of the basic step nets. As is known, the pre-places of the emptying transitions consist of control places and database places. For the execution of the emptying transitions, both pre-places should host tokens of the same colour. The tokens that reside in the control places represent the IDs of the users that perform a particular action and the tokens that reside in the database places represent the data that is related to the actions of these users. Thus, the consumption of tokens with the same colour from these places shows how the user ID and the system data are linked resulting in the advice of the system for a particular action.

From all the above, it results that every basic step net shows the interaction between the users' action and the system's response. To model the whole interactivity of the garage, eight different basic step nets are needed. Thereafter, we produce the APN model of the garage by gluing these basic step nets on the appropriate step places. Now, as regards the compositions of the APNs, it follows that five forward and three backward compositions should be applied for the implementation of the APN net of the ambient garage.

At this point, we present the APN model of the ambient garage through the construction of a small part of that net, as is presented in Snoopy [29]. For the implementation of the APN model, the structuring set Cl contains only two different colours, 1 and 2. These colours correspond to the IDs of the users. Furthermore, the following labels are given to the step places (sensors) of the net: $entrance$ for the sensor of the entrance, $p1$ for the sensor of the first junction, $p2$ for the sensor of the second junction, $A1$ for the sensor of the parking bay $A1$, $A2$ for the sensor of the parking bay $A2$ and $exit$ for the sensor of the exit. Moreover, for modelling purposes, it should be noted that the parking bays $A1$ and $A2$ are allocated to the users 1 and 2 respectively. This assumption is made in order to define what colours of tokens will reside in the database places of each basic step net. The subnet of the APN model is shown in Figure 4.14.

Another important issue is the authorised and the unauthorised users of the system. These users are represented by the use of the coloured tokens in the appropriate net places. At this point, it should be mentioned that the initial state of an APN net is given by the root net. In this case the root net of the APN net consists of the places *entrance* and *ctrl_entrance*.

For the modelling of the ambient garage, it is assumed that the APN model deals only with authorised users. Regarding the unauthorised users, they can be modelled in the APN net by putting the appropriate colours of tokens in the *ctrl_entrance* place, which implies that the users with IDs that correspond to the colours of these tokens are not allowed to enter the garage.

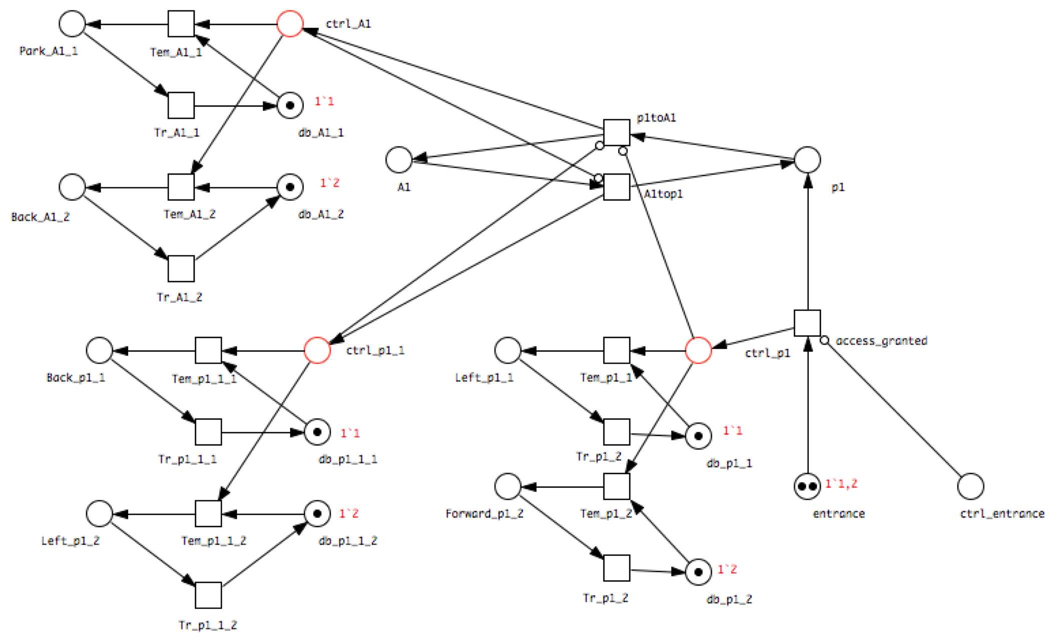


Figure 4.14: Fragment of the APN model of the ambient garage of Figure 4.15.

To construct the net of Figure 4.14, we first apply a forward composition between the root net and the first basic step net that represents the user moving from the *entrance* to $p1$. Then, another forward composition takes place between the above net and the basic step net that represents the user moving from $p1$ to the parking bay $A1$. This composition is conducted by gluing the two nets on the place $p1$. Finally, a backward composition is carried out between the net produced after the latest forward composition and the basic step net that represents the move of the user from $A1$ back to $p1$. The backward composition merges the places $A1$ and $p1$ of these two nets.

Following the same procedure for the gluing of the remaining basic step nets results in the construction of the full APN model of the ambient garage, which is shown in Figure 4.15.

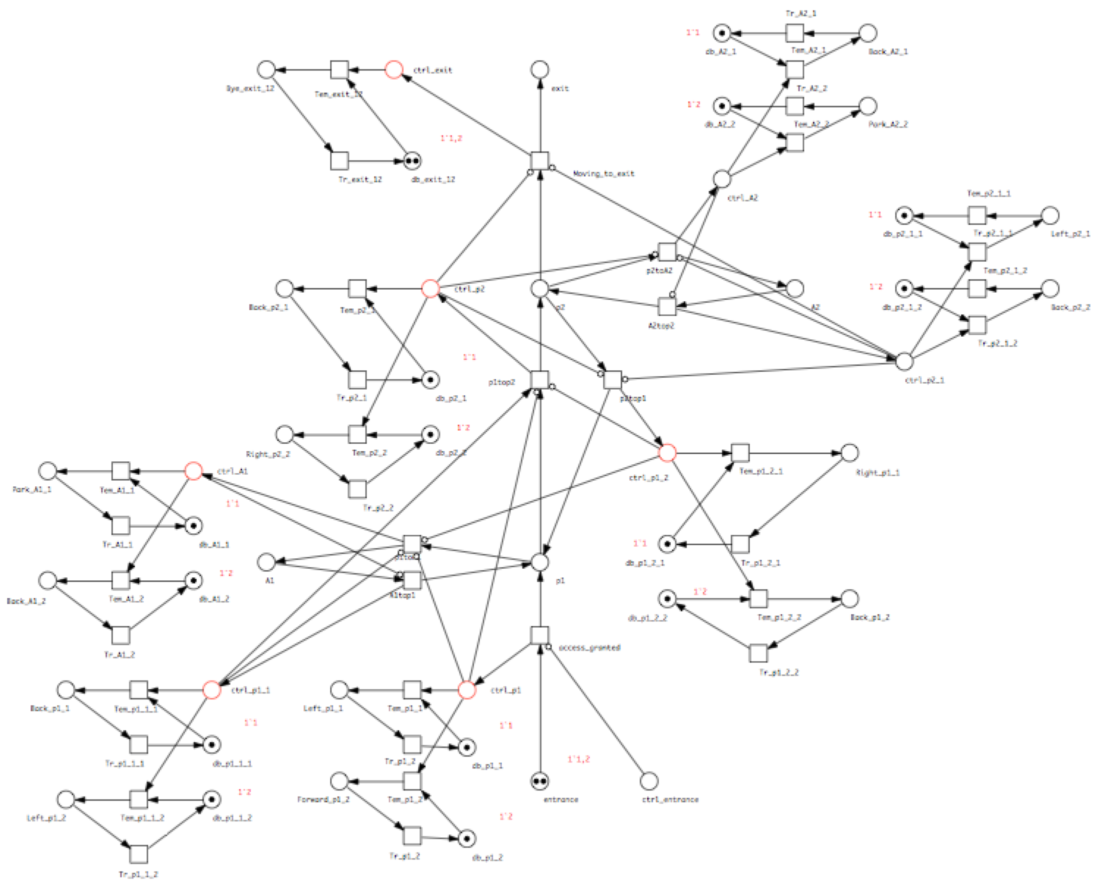


Figure 4.15: The APN net of ambient garage in Snoopy.

At this point, it should be noted that Snoopy does not support an upper bound for the capacities of the net places making the implementation of the garage model less realistic as the absence of limit on the capacity of places implies that all the cars being in the garage can potentially be at the same area, which is not possible to happen in real systems. To make the model of the garage more realistic, we propose a theoretical solution for the bounding of the net places of the produced model. As is known, the model in Figure 4.15 is an APN net with colour-sensitive inhibitor arcs, which makes it very difficult in practice to bound the net places by adding more places, transitions or arcs in the model since the addition of these elements would not comply with the definitions of the composite step nets and composition operators of the APN class. Moreover, adding extra tokens in the net, which could be used for the bounding of the places, is not practically feasible since the inhibitor nets like the model of the ambient garage face the problem of monotonicity. On the contrary, we could bound the places of the APN model of the garage by modifying the enabledness rules of the APN class. To be more specific, by adding one more condition to the already existing rules, we could bound the net places through the limitation of the total number of tokens that could reside in any of the net places. That additional enabledness condition is defined as follows

$$\forall p \in P, \forall c_i \in C(p) : \ \sum_{i=1}^{n} K((p, c_i)) \ \leq \ell$$

with $\ell \in \mathbb{N}$ being the upper bound of each net place.

Therefore, taking the total number of coloured tokens that are hosted in every net place and setting it less than or equal to ℓ, we can bound the places of an APN net. In the case of the above model, $\ell = 1$ setting the upper bound for the total number of tokens per place equal to one.

4.5.2 APN Model of Ambient Conference Room

In this section, we model the behaviour of an Ambient Information System, the ambient conference room. Contrary to the Ambient Guidance Systems, the behaviour of the Ambient Information Systems is affected only by the information provided throughout their operation. Thus, the main responsibility of the Ambient Information Systems is to help the users to complete their goals (e.g. to finish a task) by providing the correct advices to them and by protecting the information from possible leaks.

In the case of the ambient conference room, the system provides the advices via the public and the private displays of the room and prevents the leak of information through the detection of the unauthorised users by the sensors. Another way, for the system, to prevent the leak of information to unauthorised users is through the role based access control system that is used for the identification of the users.

The modelling of the ambient conference room focuses on the system interactivity during all the phases of the conference. To express that interactivity, we model the tasks, the sensors, the output displays and the context-awareness of the conference room, as in the case of the ambient garage.

At this point, we present all the assumptions that are made for the modelling of the ambient conference room. Firstly, it is assumed that all the participants leave the room after the end of the first phase, apart from the temporary reviewers that will score the papers during the break. Furthermore, it assumed that an unauthorised user can enter the room during the break interrupting the session, so the conference session can be interrupted only during the break. Furthermore, the only roles that are considered during the modelling are the roles of the temporary reviewer, the spokesperson and the member of the committee. This assumption is made in order to reduce the size of the produced model. It is also assumed that after the announcement of the final decision, the conference ends and all the participants leave the room. It should be noted that the preparation needed before the conference is out of the scope of the modelling. For that reason, we assume that everything is set up for the opening of the conference, such as distribution of roles, papers, etc. Finally, the model of the conference room is of high abstraction level since a detailed description of the conference procedure would lead to the creation of a model with enormous number of basic step nets. For that reason, every basic step net that is used for the representation of an interaction between the user and the environment describes a more generic aspect of it.

Having described the main points and assumptions of the modelling of the ambient conference room, we proceed to the implementation of the APN model. As with the modelling of the ambient garage, we use the step-modelling approach to define of all the possible actions of the users, throughout the phases of the conference, as it is described in the specification of the system. These actions are as follows:

(*i*) Participants enter the room for the first phase,

(*ii*) The temporary reviewers sit in front of the private displays and enter their credentials for the start of the break phase,

(*iii*) An unauthorised user enters the room or private display areas during the break phase,

(*iv*) The unauthorised user leaves the room,

(*v*) The temporary reviewer continue with break phase scoring the papers,

(*vi*) The temporary reviewers log out after the end of the break phase,

(*vii*) The unauthorised user leaves the room and waits for the final decision after the end of the break phase, (*viii*) The authorised spokesperson participates in the second phase,

(*ix*) A member of the committee is in conflict with a paper and exits the room,

(*x*) The replaced user waits for the decision about that paper in order to return to the session,

(*xi*) The committee makes the final decision at the end of the second phase,

(*xii*) The final decision is announced to all the participants during the last phase.

As in the ambient garage, all the actions of the users are expressed by step transitions of the respective basic step nets. From the specification, it follows that these actions are in a particular order. For example, the first action always precedes the second one and so on. This implies that the basic step net that represents an action is glued with the next action via a forward composition. This composition is applied to all the actions, except for the case of the fourth action. For the basic step net of this action a backward composition is carried out. This happens because the unauthorised user stops the break phase by his presence and starts it again by leaving the room. This means that the system returns to the state that it was before the interruption.

In the APN model of the ambient conference room, the step places represent either tasks, detection of user or general conditions. For example, a place that represents a task is the place *home_page*, where the user has logged in to his personal computer and acquires information about the papers that he has to score. Another place that represents a user's detection is the place *participants* that corresponds to the card reader of the door. Finally a place that represents a general condition of the user is the place *entering_room* that refers to the case where a participant is in the room.

Moving to the control system of the basic step nets, it represents the database and the output displays of the system, as happens with the respective control systems of the APN net of the ambient garage. It is also noticed that the context-awareness is expressed in exactly the same way as in the case of the ambient garage.

Now, for the implementation of the APN model, the structuring set Cl consist of two different colours, 1 and 2. These colours refer to the IDs of the users that either perform a task or get a particular role. The APN model of the ambient conference room is explained through the construction of a small part of that model. This net is shown in Figure 4.16.

Another important modelling issue is the representation of both the authorised and the unauthorised users of the system. These users are represented by the use of the coloured tokens in the appropriate net places. At this point, it should be mentioned that the initial state of an APN net is given by the root net. In this case the root net of the APN net consists of the places *participants* and *ctrl_participants*.

According to the specification, the initial state of the ambient conference room consists authorised users only, as all the participants can attend the first phase of the conference. But, in case that we want to model unauthorised users, they can be modelled by putting the appropriate colours of tokens in *ctrl_participants*. In that case, the users with IDs that correspond to the colours of these tokens are not allowed to participate in the conference from the very beginning of the whole process.

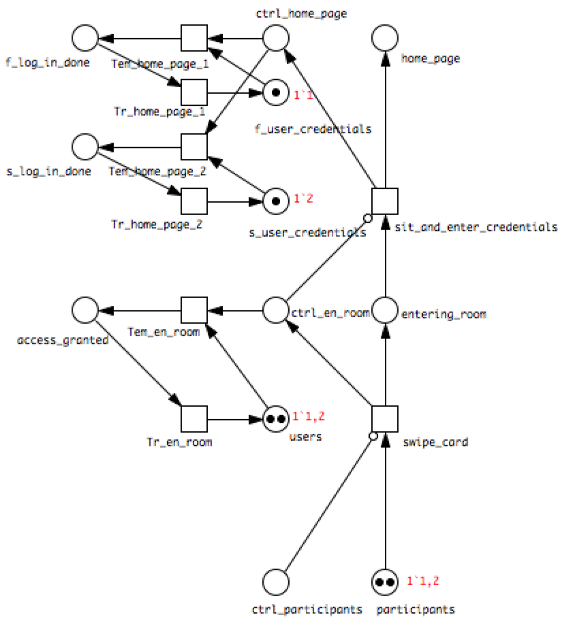


Figure 4.16: Fragment of the APN model of the ambient conference room of Figure 4.17.

To construct the net of Figure 4.16, we first apply a forward composition between the root net and the first basic step net that represent the case where the participants enter the room. This composition is carried out by gluing the two nets on the place *participants*. Finally, we compose the produced net with the second basic step net that represents the case where the temporary reviewers sit in front of the private displays and enter their credentials. This composition is conducted by merging the two nets on the place *entering_room*.

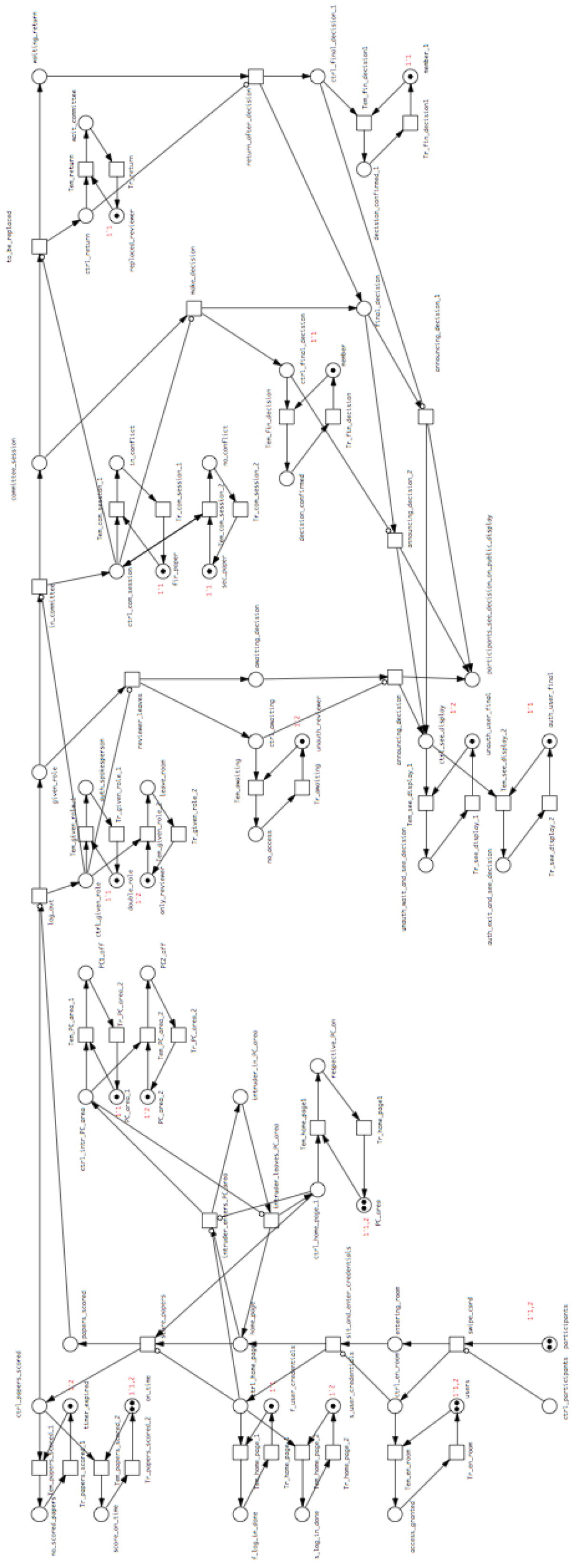


Figure 4.17: The APN net of ambient conference room in Snoopy.

Following the same procedure for the gluing of the remaining basic step nets results in the construction of the full APN model for the ambient conference room, which is presented in Figure 4.17.

Contrary to the APN model of the ambient garage, that of the ambient conference room is not needed to have bounded net places. That derives from the fact that the places in this case do not represent physical spaces in the room, but different roles, tasks or credentials given to particular participants of the conference.

For example, there could exist multiple tasks performed by many users at the same time allowing in that sense the presence of several coloured tokens into a net place without being necessary to bound the places using a specific limit for them. The only limit that is imposed on these places is that they can host only one token from each of the available colours given by the structuring set Cl. Thus, these places are limited by the cardinality of the structuring set Cl. Finally, it should be mentioned that in case that we want to bound the places of the net that represents the conference room, we could do it in exactly the same way as in section 4.5.2.

4.6 Concluding Remarks

Applying the APN class to the modelling of the case studies resulted in the creation of the step-modelling approach. This approach enables the development of models for general ambient systems through the repetitive use of a structural building block, which represents the interaction between the user and the system. This interaction is expressed by the token circulation between the two components of the building block, the unidirectional step and the control system, which represent the user's and the system's actions respectively.

To explain the structure of the building block, we described its elements and we sorted them into different categories. Each of these categories represents a feature of the ambient system that is examined, such as sensors, displays, database, etc.

Afterwards, the definition of both the basic step net and root net have been introduced. These nets are used for the composition of the APNs that was addressed in the second part of the chapter. Briefly, the composition of the APNs described how two or more building blocks and the root net can be glued, via their repetitive use that was mentioned above. The composition of the APNs uses two operators, one for the extension of the nets by a step forward and another for the creation of 'cyclic nets' in the case of going backward to some already existing state. These two operators lead to a flexible compositional approach as the nets can be extended by gluing on every 'step' place and not only on specific places (as it happens with other compositional approaches [57]).

Having outlined the rationale behind the step-modelling approach, it is concluded that this approach can successfully deal with the modelling of ambient systems with APNs. It should be mentioned that after the definition of the APNs composition, it follows that the step-modelling approach is supported by a strong and well defined mathematical background that consists of the APN class and the composition of the APNs. Furthermore, it has been shown that the APN net that consist of composite basic step nets that were composed with each other by applying either forward or backward compositions could lend common structural and behavioural features to the produced APN models, as was discussed in section 4.4. Finally, using the step-modelling approach along with all its 'mathematical support' resulted in the development of the APN models for the ambient systems of the case studies (see section 4.5).

Chapter 5

Transformed Ambient Petri Nets (T-APNs)

In this chapter, we deal with the class of the Transformed Ambient Petri Nets (T-APNs). This class is defined for verification purposes, which derive from the fact that existing verification tools like Charlie[11] and CPN tools [55] do not support the verification of inhibitor nets, like APNs.

Therefore, to verify the APNs, they should be translated into a verifiable class of Coloured Petri Nets with no colour-sensitive inhibitor arcs. This class is the Transformed Ambient Petri Nets that is mentioned above. To maintain the functionality given to the APNs by the presence of the colour-sensitive inhibitor arcs in the T-APNs, all the colour-sensitive inhibitor arcs of the APNs should be replaced by the appropriate number of directed arcs in the T-APNs. Furthermore, additional net elements should be used for the definition of the T-APNs, such as arc inscriptions, guards, etc.

At this point, it should be mentioned that the scope of this chapter is not to relate the APN class with the T-APN class or to compare these two classes (both are addressed in chapter 6), but to present the definition of the T-APNs and to mention the importance of this class.

The importance of the T-APN class lies in the fact that it is used for the expression of the output models of the construction for given APNs, as it will be described in section 6.2. The T-APN models produced by the construction can be verified using Charlie verification tool [11] and the results of the verification are used for the observation and the analysis of the ambient systems, which will be discussed later in chapter 7.

In the next sections, we define the T-APN class by describing the notations and the semantics used in that class and more specifically in its firing and enabling rules. Finally, we refer to the behaviour of the T-APNs through a discussion about the step transition systems of this class.

5.1 Transformed Ambient Petri Nets Class

In this section, we introduce the Transformed Ambient Petri nets (T-APNs) by providing the definition of the class and explaining the elements of these nets.

Definition 20 (Transformed Ambient Petri Net)**.** *A* Transformed Ambient Petri Net *is a tuple*

$$\mathcal{N} = (P, T, Pre, Post, Cl, \overline{Cl}, C_P, C_T, Ins, K, Gd, M_0, G),$$

where:

- *P is a finite set of places.*
- *T is a finite set of transitions disjoint from P.*
- *Pre, $Post : T \to \mu P$ are the pre and post mappings.*
- *Cl, $\overline{Cl}$ are non-empty finite sets of non-empty colour sets, such that $(\bigcup Cl) \cap (\bigcup \overline{Cl}) = \varnothing$ and there exists a bijection f: $(\bigcup Cl) \to (\bigcup \overline{Cl})$. We will call Cl and $\overline{Cl}$ the structuring sets.*
- *$C_P : P \to CSets$ is a colour function used to structure places:*
  - *$\widehat{P} = \{(p, g) \mid p \in P \land g \in C_P(p)\}$ is the set of structured places, where $CSets \subseteq 2^{(\bigcup Cl) \cup (\bigcup \overline{Cl})}$ is a non-empty subset of non-empty colour sets from $2^{(\bigcup Cl) \cup (\bigcup \overline{Cl})}$.*
- *$C_T : T \to Cl$ is a colour function used to structure transitions:*
  - *$\widetilde{T} = \{(t, c) \mid t \in T \land c \in C_T(t)\}$ is the set of structured transitions.*
- *$Ins : A \to Cl \cup \overline{Cl} \cup \{\varnothing\}$ is a colour function used to assign colour sets (inscriptions) to the arcs of the net, where:*
  - *$A = \bigcup\limits_{t \in T} A^t_{in} \cup \bigcup\limits_{t \in T} A^t_{out} \cup \{no-arc\}$ is the set of arcs of the net,*
  - *$A^t_{in} = \{(p, t) \mid p \in supp(Pre(t))\}$, the set of incoming arcs of transition t,*
  - *$A^t_{out} = \{(t, p) \mid p \in supp(Post(t))\}$, the set of outgoing arcs of transition t,*
  - *$no-arc$ is an 'empty' arc (no connection) for which $Ins(no-arc) = \varnothing$,*
  - *$A^t = A^t_{in} \cup A^t_{out}$, the set of arcs connected to transition t,*
  - *$\widehat{A} = \{(a, ins) \mid a \in A \land ins \in Ins(a)\}$ is the set of structured arcs.*

- *K is a constant function that defines the capacities of places.*
- *$Gd : \widetilde{T} \rightarrow \{true, false\}$, is a Boolean guard function defined as follows:*

  For $t \in T_G$:
  $$Gd((t,c)) = \begin{cases} true & \text{if } (c, f(c)) \in B_t \\ false & \text{otherwise} \end{cases}$$
  where B_t is a set of binding pairs:
  $$B_t = \{(x, f(x)) \mid x \in C_T(t)\}$$
  and for $t \in T_{NG}$:
  $$Gd((t,c)) = true$$
  In the above, $T = T_G \cup T_{NG}$, where T_G is a set of transitions with guards and T_{NG} is a set of transitions 'without' guards[1]*.*
- *$M_0 \in \mu\widehat{P}$ is an initial marking satisfying $M_0 \leq K$. In general, any $M \in \widehat{P}$ such that $M \leq K$ is a marking.*
- *$G \subseteq P$ is a set of gluing (or interface) places.*

Having finished with the definition of the class, a small T-APN net that illustrates most of the elements of the T-APNs is shown in Figure 5.1. Furthermore, this figure presents the step transition system of the net, which will be described later on in section 5.3. In this section, the net of that figure is used as an example for the description of the net elements.

The meaning and graphical representation of P (places) and T (transitions) are as in the standard net theory. The set of places, G, will be important in the composition of the Transformed Ambient Petri Nets, as it represents the set of gluing places of a T-APN net. The directed arcs of the net are given by the *Pre* and *Post* mappings. For example, for the only transition of the T-APN of Figure 5.1 we have: $Pre(t) = \{p_1, p_2, p_5\}$ and $Post(t) = \{p_3, p_4\}$. We will call p_1, p_2 and p_5 the *pre-places* of t, and p_3 and p_4 the *post-places* of t. In general, any $p \in supp(Pre(t))$ is a pre-place of t and any $p \in supp(Post(t))$ is a post-place of t. In this class, $Pre(t)$ and $Post(t)$ will be sets for any $t \in T$. This means that every directed arc of an T-APN net has the weight 1. We also use the following

[1]Guards are applied to all the transitions, but we consider that every $t \in T_{NG}$ is a transition with 'no guard' since as we will see later its enabledness will not be affected by the guard value, which is constantly set to *true*.

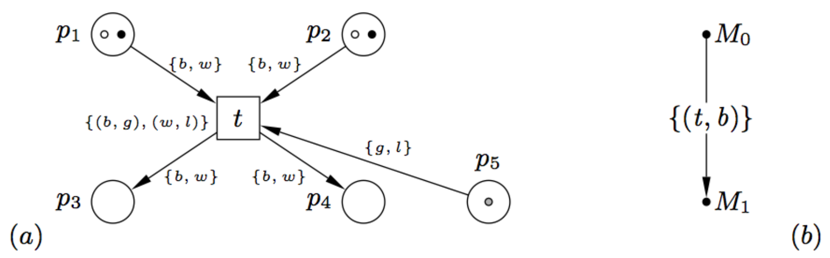


Figure 5.1: A Transformed Ambient Petri Net $\mathcal{N}$, where the capacity K is a constant function 1 (a), and its concurrent reachability graph $CRG(\mathcal{N})$ (b). In the diagram, each ○ token represents w, each ● token represents b and each ● token represents g.

'dot' notation:

$$\begin{aligned}
{}^{\bullet}t &= \mathit{supp}(\mathit{Pre}(t)) && \text{(pre-set of } t \in T\text{)},\\
t^{\bullet} &= \mathit{supp}(\mathit{Post}(t)) && \text{(post-set of } t \in T\text{)},\\
{}^{\bullet}p &= \{t \in T \mid p \in \mathit{supp}(\mathit{Post}(t))\} && \text{(pre-set of } p \in P\text{)},\\
p^{\bullet} &= \{t \in T \mid p \in \mathit{supp}(\mathit{Pre}(t))\} && \text{(post-set of } p \in P\text{)}.
\end{aligned}$$

If $p \in {}^{\bullet}t$ then we denote the arc from p to t by (p, t). Respectively, if $p \in t^{\bullet}$ we denote the arc from t to p by (t, p). It should be mentioned that the above 'dot' notation can also be extended for the sets of places and transitions and is denoted as ${}^{\bullet}P$, $P^{\bullet}$ and ${}^{\bullet}T$, $T^{\bullet}$ respectively.

The C_P and C_T functions are used for structuring places and transitions respectively, by equipping them with colours. Places can hold coloured tokens, representing different kinds of resources or agents of the T-APNs or the 'internal' functioning of the net. Every structured place of an T-APN can potentially host either identity, special or both colours of tokens. The set of structured places is given by $\widehat{P}$. A structured transition $(t, c) \in \widetilde{T}$ represents an action that operates in a colour 'mode' given by c. As already mentioned, Cl and $\overline{Cl}$ are non-empty finite sets of colour sets that represent the group of agents or user-identities of an ambient system. More specifically, sets of Cl give the unique colour IDs of the users and those of $\overline{Cl}$ provide the 'complement' colours of the IDs of $\bigcup Cl$, which are used for the 'internal' functioning of the net. Cl and $\overline{Cl}$ will be called *structuring* sets as they are used in the colour functions that structure the places, the transitions and the arcs.

In Figure 5.1, $CSets = \{\{g\}, \{b, w\}\}$, $Cl = \{b, w\}$ and $\overline{Cl} = \{g, l\}$, where b stands for the black colour, w for the white colour, g for grey colour and l for light grey colour. $\widehat{P}$ for the T-APN net in Figure 5.1 is the set $\{p_1, p_2, p_3, p_4\} \times \{b, w\} \cup$

$\{p_5\} \times \{g\}$ assuming that $C_P(p_i) = \{b, w\}$ for $i \in \{1, \dots, 4\}$ and $C_P(p_5) = \{g\}$. In diagrams, a structured place is represented by a coloured token that resides in this place. For example, (p_1, b) means that a black token was placed in place p_1.

Also, $\widetilde{T}$ (for the T-APN in Figure 5.1) can be defined as $\{(t, b), (t, w)\}$, assuming that $C_T(t) = \{b, w\}$. In this case the transition t can operate in the two colour modes defined by $C_T(t)$, the black mode and the white mode. The colour mode of a transition is restricted when a guard is present. For example, if $t \in T_G$, then the guard function Gd, for a given $(t, c) \in \widetilde{T}$, indicates whether transition t can fire in mode c with respect to the set of *binding pairs* defined for this transition by B_t. On the contrary, if $t \in T_{NG}$, then the guard function Gd, for a given $(t, c) \in \widetilde{T}$, indicates that transition t fires in mode c without considering the set of *binding pairs* as it is always set to true. We will represent the guards of $t \in T_G$, in net diagrams, by writing next to each transition the set of *binding pairs* for which it can fire in mode c. The guards of $t \in T_{NG}$, in net diagrams, will be omitted as they do not depend on the set of *binding pairs* in order to evaluate to true. If, for example, $B_t = \{(b, g), (w, l)\}$ (as in Figure 5.1), then in the Snoopy modelling tool [29] the guard of t could be expressed as follows:

$$(Cl = b \; \& \; \overline{Cl} = g) \;\; | \;\; (Cl = w \; \& \; \overline{Cl} = l)$$

while in the CPN tools [55] it would be expressed as:

$$(Cl = b \;\; andalso \;\; \overline{Cl} = g) \;\; orelse \;\; (Cl = w \;\; andalso \;\; \overline{Cl} = l)$$

Generally, a guard in Snoopy modelling tool or in CPN tool can be expressed in different ways due to the different languages that are used in those two tools. A transition t cannot be executed in mode c if the guard evaluates to false for (t, c). Structured transitions only become apparent when we look at the net's behaviour (see Figure 5.1(b)). The net's behaviour in this example is determined by the guard of the structured transition. The (t, b) of the $CRG(\mathcal{N})$ indicates that the transition with the limitations imposed by the guard fires in black mode for the black and grey pair of tokens. We assume that a pair of tokens that belongs to the set of *binding pairs* B_t and for which a transition fires, gives rise to a colour mode determined by the colour of the first element of the pair. This can be done assuming that a transition treats the complementary colours given by the bijection f of the set of *binding pairs* as identical to their associated colours. By complementary colours, we define the images of the colours of $\bigcup Cl$ given by the bijection f: $(\bigcup Cl) \to (\bigcup \overline{Cl})$.

As already mentioned, a guard is a Boolean function that imposes additional constraints on the enabling of the transitions. It should be noted that the transitions with 'no guard' (T_{NG}) are considered enabled if they satisfy the marking requirements as the ‘missing’ guards of these transitions always evaluate to true. It should be mentioned that a quite similar formalism to the above is given in [45] describing the ‘conditional’ and ‘unconditional’ events of Conditional Partial Ordered Graphs.

The initial marking M_0 (which cannot exceed the capacities given by K) specifies, for each place p, the number of tokens of each colour held in p. The initial marking M_0 for the net of Figure 5.1 is $\{(p_1, b), (p_1, w), (p_2, b), (p_2, w), (p_5, g)\}$. T-APNs are *safe coloured Petri nets* (only one token of a given colour is allowed per place), as it happens with the APNs. This is motivated by the fact that colours will be used as identities given to agents or other entities interacting within an ambient environment. As a consequence, the capacity K will be a constant function returning 1 for each colour of tokens, per place.

As already described, the arcs of the T-APNs have inscriptions on them, which represent the colours of tokens that an arc can carry. This can be expressed using the Ins colour function. In our running example in Figure 5.1, we have two disjoint colour sets: the colour set $\bigcup Cl = \{b, w\}$ and the special colour set $\bigcup \overline{Cl} = \{g, l\}$. Set $\{b, w\}$ represents the identities of the agents and set $\{g, l\}$ represents the special set of colours that replace the colours of $\{b, w\}$, for net functioning reasons, according to the set of *binding pairs* B_t. The arcs $(p_1, t), (p_2, t), (t, p_3), (t, p_4)$ and (p_5, t) in Figure 5.1(a) get the inscriptions $\{b, w\}$, $\{b, w\}$, $\{b, w\}$, $\{b, w\}$ and $\{g, l\}$ respectively, which means that the first four arcs can carry black and white tokens and the last one can carry grey and light grey tokens. The set of *binding pairs* for the transition t in Figure 5.1 is $\{(b, g), (w, l)\}$, i.e., black is associated with grey and white with light grey respectively ($f(b) = g$ and $f(w) = l$). That set actually provides the pairs of colours of tokens, for which a transition may be enabled. Therefore, the set of *binding pairs* is important for the association of the colour sets Cl and $\overline{Cl}$ and the operation of the guard function (Gd).

5.2 Semantics of T-APN nets

Having described the net elements, we continue with the definition of the semantics of the T-APNs, which indicate the functioning of the T-APN nets by providing the firing and the enabling rules of the class.

To define the semantics for the firing of a T-APN net we need to extend the Pre and $Post$ mappings using the structured places and transitions. $\widehat{Pre}$, $\widehat{Post}$: $\widetilde{T} \to \mu\widehat{P}$ are defined as follows:

$$\widehat{Pre}(t, c)(p, g) = \begin{cases} Pre(t)(p) & \text{if } (g = c \vee g = f(c)) \wedge g \in Ins((p, t)) \\ 0 & \text{otherwise} \end{cases} \qquad (\dagger)$$

and

$$\widehat{Post}(t,c)(p,g) = \begin{cases} Post(t)(p) & \text{if } (g = c \vee g = f(c)) \wedge g \in Ins((t,p)) \\ 0 & \text{otherwise} \end{cases} \qquad (\dagger\dagger)$$

$\widehat{Pre}(t,c)$ denotes, for every place p, the number of tokens of colour g that t needs from place p in order to fire when the colour mode of the transition is 'compatible' with g. $\widehat{Post}(t,c)$ denotes, for every place p, the number of tokens of colour g that t will deposit in p after being fired. Notice that $Pre(t)(p)$ and $Post(t)(p)$ in the formulas above would be 1 for T-APN nets, as the weights of the directed arcs are all 1.

We can now extend $\widehat{Pre}$ and $\widehat{Post}$ functions to steps of structured transitions $U = \{(t_1,c_1),\ldots,(t_n,c_n)\} \in 2^{\widetilde{T}}$ as follows: $\widehat{Pre}(U) = \widehat{Pre}(t_1,c_1)+\cdots+\widehat{Pre}(t_n,c_n)$, and similarly for $\widehat{Post}$. The above extension is needed as the proposed semantics will be a *step semantics* rather than a *sequential semantics*.

Now we describe the semantics of a Transformed Ambient Petri Net $\mathcal{N}$. A step of structured transitions U is *enabled* at marking M if the following hold:

- $\widehat{Pre}(U) \leq M$.
- $M + \widehat{Post}(U) \leq K$.[1]
- $\forall\ (t_i,c_i) \in U,\ \mathit{with}\ i = 1,\ldots,n,\ Gd((t_i,c_i)) = \mathit{true}.$

We denote this by $M[U\rangle_{\mathcal{N}}$. An enabled step U can fire producing the marking $M' = M - \widehat{Pre}(U) + \widehat{Post}(U)$. This will be denoted by $M[U\rangle_{\mathcal{N}}M'$.[2] That means that a step is enabled if the pre-places of all the transitions in the step, working in certain colour modes (taking into account their multiplicities), have sufficient number of tokens of appropriate colour. Furthermore, an enabled step U satisfies that the tokens added to the current marking of the T-APN net, after the firing of U, do not violate the capacity of the net places. Finally, for the step U to be enabled, it should be ensured that the guard functions of the structured transitions of the step evaluate to true for the appropriate colours. It should also be mentioned that an enabled step may fire if the inscriptions ensure that the appropriate colours of tokens can be carried by the structured arcs, as is described in $\widehat{Pre}$ and $\widehat{Post}$ of the T-APNs.

For example, for the T-APN in Figure 5.1, the step $\{(t,b)\}$ is enabled at M_0 as the pre-places of transition t host both the adequate number and the proper

[1] If we want to allow self-loops in the net, this condition should take a weaker form: $M - \widehat{Pre}(U) + \widehat{Post}(U) \leq K$.

[2] $\mathcal{N}$ can be omitted if it is clear from the context.

colours of tokens, i.e., a black token in places p_1 and p_2 respectively and a grey token in place p_5 . Firing the enabled step $\{(t, b)\}$ removing the black tokens from the places p_1 and p_2 and the grey token from the place p_5 (see (†)) and adding one black token to the places p_3 and p_4 (see (††)) respectively, it can be noticed that the capacity of the post-places p_3 and p_4 is not violated as these places were previously empty. Additionally, the guard of transition t evaluates to true as the colours of tokens that reside into the pre-places of that transition belong to the set of *binding pairs*. Contrary to the enabled step $\{(t, b)\}$, the steps $\{(t, w)\}$ and $\{(t, b), (t, w)\}$ are not enabled at M_0 and cannot be fired as the complementary coloured token of w (i.e., l) is missing from p_5. Finally, the structured transition (t, b) cannot fire twice, since the step $\{(t, b), (t, b)\}$ is not enabled due to the lack of sufficient number of tokens in the places p_1, p_2 and p_5.

5.3 Step Transition System of T-APNs

The full execution semantics of a T-APN net will be captured using a transition system where arcs are labelled by executed steps, the step transition system. The *step transition system* of the T-APNs is defined in exactly the same way as the step transition system of the APNs is described in definition 5. Furthermore, from that definition, it follows that the *concurrent reachability graph* of a T-APN net is a kind of a step transition system.

Hence, to describe the behaviour of the T-APNs through the step transition systems, we take as an example the net of Figure 5.1(a) and its concurrent reachability graph (shown in Figure 5.1(b)). We start capturing the behaviour of the net by examining the enabledness of transition t, as it is the only transition in this net. According to the enabling rules of the T-APNs, transition t is enabled for the black token at the initial marking M_0 (see section 5.2). This means that the transition can fire at least once under the black mode. After the firing of enabled step $\{(t, b)\}$, a new state results and is described by the marking $M_1 = \{(p_1, w), (p_2, w), (p_3, b), (p_4, b)\}$. Now, checking the enabledness of t at M_1, it turns that the transition cannot be executed further since it is not enabled. So, the non-existence of enabled transition leads to a dead state for the net, which is the terminal state defined by M_1. Thus, to draw the concurrent reachability graph of $\mathcal{N}$, we start from the initial marking M_0 and we generate an arc that links to a new marking (state) every time that a step is executed. This arc is labelled with the step that is executed each time. In this case, the concurrent reachability graph of $\mathcal{N}$ is shown in Figure 5.1(b) and consists of two states, M_0 and M_1, which are linked with the arc that is named after the step $\{(t, b)\}$, as was described earlier.

Consequently, examining the enabledness of all the transitions of a T-APN net and executing every time its enabled transitions, concurrently or not, we can construct the concurrent reachability graph of that net starting from the initial state that is given by the marking M_0.

5.4 Concluding Remarks

In this chapter, we defined the class of the T-APNs by describing both the static and the dynamic aspects of it. The static aspect of the T-APNs is related to those element that determine the structure of these nets, such as the places, the transition, the arcs, the capacity, etc. On the contrary, the dynamic aspect of the T-APNs concentrates on how the marked T-APN nets behave with respect to the semantics mentioned above.

The behaviour of T-APNs can be captured by the step transition systems and in particular by the concurrent reachability graphs of the nets, as with the APNs. It should be mentioned that all the above notions are fundamental principles of the class of the T-APNs and are needed for the correct and proper functioning of the nets.

Finally, an important inference that derives from this chapter is that the theory behind the T-APN class is based on a strong mathematical background and is well written.

Chapter 6

Relating APNs with T-APNs

Chapters 3 and 5 discuss the classes of Ambient Petri Nets and Transformed Ambient Petri Nets respectively, which provide different kind of coloured Petri nets. As has already been mentioned in those chapters, the nets of these classes are used for different purposes.

In this chapter, we describe a construction of a T-APN net associated with a given APN net showing how the nets of these two classes are related. Consequently, demonstrating that these nets are related and more specifically that they behave in exactly the same way, it can be concluded that the verification results of the T-APN nets could be applied to the structural and the behavioural analysis of the APN nets in a reasonable and meaningful way. Hence, an APN net that cannot be verified directly by the verification tools, could be 'verified' through an equivalent T-APN net.

The discussion about the construction is conducted into two stages. Firstly, the two classes of nets are compared with regard to their structure, their behaviour and generally their formalisms. Finally, we provide the formal definition of the construction which is based on the comparison that is carried out in the first stage.

Thereafter, we illustrate the working of the construction by giving an example of a constructed T-APN net out of a given associated APN net. In addition, we provide another example that examines the behaviour of two nets intuitively, i.e., a constructed T-APN nets and its associated APN net. Finally, after that example, we demonstrate the translation of the APN models of the case studies into the respective T-APN models through the application of the construction.

In the last section of this chapter, we prove the behavioural equivalence of the nets of these two classes by showing that their concurrent reachability graphs are isomorphic.

6.1 Comparing APNs with T-APNs

To compare the nets of these two classes, we first outline the general situation with an example that sets the key similarities and differences of the nets. Thereafter, having set the general framework of the comparison through that example, a detailed description takes place.

6.1.1 Setting the Scene of the Comparison

As has already been mentioned above, an example of a visual comparison of two simple nets, an APN net and a T-APN net, is presented. The aim of this example is to 'visualise' the differences and the similarities of both the static (structure) and the dynamic (behaviour) aspects of these nets.

The first net in Figure 3.1 shows an APN net with an enabled transition t at the initial marking and its reachability graph that reveals the colour mode under which the enabled transition fired. Similarly, the net in Figure 5.1 depicts a T-APN net with an enabled transition at the initial marking. In addition, this figure shows the concurrent reachability graph of that particular T-APN net.

From the visual comparison of these two nets, it can be noticed that the nets consist of the same number of places and transitions respectively. On the other hand, some structural and functioning differences between these nets are: the number and the colours of the tokens used in both nets, the guard of the transition used in the T-APN net, the colour-sensitive inhibitor arc used in the APN net and the inscripted arcs used in the T-APN net. Moreover, from the comparison of the concurrent reachability graphs of the two nets derives that these nets behave in exactly the same way.

All the observations made above could set the general framework of the discussion about the comparison that is described in details in the following section. Finally, it is also explained how this framework helps us to define a construction of a T-APN net associated with a given APN net. By 'associated' we mean that two nets represent the same behaviour (their concurrent reachability graphs are isomorphic).

6.1.2 Comparing Structure and Functioning of APNs and T-APNs

At this point, we compare the APNs with T-APNs and we explain how these two classes of nets are associated with each other and in what way. As has already been mentioned, T-APNs result from the need of converting the APNs to a net class that could be a valid input for Charlie verification tool [11]. Therefore, T-

APNs could be used for the 'translation' of APNs into Coloured Petri Nets with no colour-sensitive inhibitor arcs.

Hence, APN nets are transformed into T-APN nets maintaining the expressive power of APNs that is given by the presence of colour-sensitive inhibitor arcs by replacing those arcs with direct arcs. These T-APN nets are non-inhibitor nets and are supported by Charlie for the verification process.

To transform the APNs into T-APNs, we have to consider that the functionality of the APNs is actually based on the colour-sensitive inhibitor arcs, which are not part of the T-APNs. Hence, to conduct the transformation we have to tackle this problem, i.e. the representation of the colour-sensitive inhibitor arc in the T-APN nets. This is done by introducing a new colour set, $\overline{Cl}$, which is used for the structuring of the places and the arcs, a guard function that controls the behaviour of the transitions with respect to the set of *binding pairs* that defines the set of valid pairs of coloured tokens that can fire the transitions, the inscriptions that describe the colours that each structured arc can carry and by adding some more directed arcs to the net. The use of all these four introduced elements contribute to the preservation of the colour-sensitive inhibitor arc functionality in the T-APNs.

Due to the introduction of all the above elements, the T-APNs have more elements in their tuple than the APNs. But, let's start the comparison of the two classes from their main differences and similarities. This will help, later on, to establish a construction that results from that comparison.

Let

$$\mathcal{N} = (P, T, Pre, Post, I, Cl, C, K, M_0, G)$$

be an APN net and

$$\mathcal{N}' = (P', T', Pre', Post', Cl', \overline{Cl}, C_P, C_T, Ins, K', Gd, M_0', G')$$

be a T-APN net associated with $\mathcal{N}$.

It can be noticed from the above that the additional elements in the T-APN tuple are the $\overline{Cl}, C_P, C_T, Ins$ and Gd. Three of them, the special colour set ($\overline{Cl}$), the inscriptions (Ins) and the guards (Gd) have already been described above and were introduced in order to replace the colour-sensitive inhibitor arcs' functionality in the T-APN nets. Both C_P and C_T are the colour functions that structure the places and the transitions of the T-APNs respectively, substituting the function C that is used in the APNs. APNs have only one colour function to structure the places and the transitions of a net as the places and the transitions of the APN nets take colour values from the same structuring set, the Cl.

Having mentioned that the two nets use different 'colour resources' resulting in the use of different colour functions, it can be concluded that the places and the transitions of these nets may host different colours of tokens and work in different or similar colour modes respectively. Consequently, we should define a relation between the colour function C of a APN net and the respective colours functions C_P and C_T of a T-APN net in order to build our construction. The C_P and C_T functions use different ranges in their definitions because we want to satisfy that the places can host any combination of colours given by the structuring sets and that the transitions can operate only in colour modes that are given by one structuring set only. For instance, in Figure 5.1, places p_1, p_2, p_3, p_4 host tokens of identity colours and p_5 hosts a token of special colour, but all these net places can potentially host all the colours (i.e., b,w,g and l). On the contrary, transition t can operate only under black or white mode. In other words, the colour mode of transition t is strictly defined by the set of identity colours, Cl. Regarding the direct arcs of the two nets, they are defined in the same way for both nets, i.e. as mappings between the places and the transitions and vice versa but the arcs of the T-APNs carry inscriptions on them.

As is noticed from the comparison of the formalisms of the two nets, the presence of colour-sensitive inhibitor arcs makes the definition of the APNs more succinct compared to that of the T-APNs, as elements like $\overline{Cl}, Ins$ and Gd are not needed for the operation of the APN nets.

Examining the two nets, it follows that the capacity of the places of the APN net should be linked to that of the places of the T-APN net with two different ways since the places of the T-APN net take colours of tokens from two different sets. Thus, the capacity relation should be defined for both the identity and the special colours of tokens.

Regarding the initial markings, it should be noticed that the inhibitor place of the APN net is empty as the transition of the net is enabled and that the respective place of the T-APN net hosts a token of a special colour. Furthermore, the colours of the tokens that reside in all the other places are the same in both nets. Hence, the definition of the relation between the two markings is necessary for the construction. Finally, the gluing places of the two nets are similarly defined and are used for the same purpose in both nets.

As has already been described in 6.1.1, the similarities and the differences of the two nets can be easily spotted through the comparison of Figures 3.1 and 5.1.

6.2 Defining Construction

Following what was mentioned above, we define the construction of a T-APN net that is associated to a given APN net. This construction is defined by describing the relation of the tuple elements of the given APN net with the respective elements of the constructed T-APN net.

6.2.1 Preliminaries of Construction

At this point, we mention all these notions that are used for the definition of the construction concentrating on a particular sub-class of the APN nets, the composite step nets (see definition 13). Therefore, for a given composite step net $\mathcal{N} = (P, T, Pre, Post, I, Cl, C, K, M_0, G)$, we refer to the constructed net as the T-APN net associated with $\mathcal{N}$ and denote it by $\varphi(\mathcal{N}) = (P', T', Pre', Post', Cl', \overline{Cl}, C_P, C_T, Ins, K', Gd, M'_0, G')$. Thereafter, it is explained what it means $\mathcal{N}$ and $\varphi(\mathcal{N})$ to be equivalent.

The given net in the following construction is a composite step net (composed out of n basic or one step nets and a root net with an index 0).

Then:

$$P = P_R \cup P_D \cup P_C \cup P_{SF} \tag{$\star$}$$

where

$P_R = \bigcup_{i=0}^{n} P_r^i$,

$P_D = \bigcup_{i=0}^{n} P_d^i$,

$P_C = \bigcup_{i=0}^{n} \{p_c^i\}$,

$P_{SF} \subset \bigcup_{i=0}^{n} P_{ST}^i$,

with $P_r^i, P_d^i, \{p_c^i\}$ and P_{ST}^i being the sets of response, database, control (or inhibitor) and step places of the components of the given composite APN net $\mathcal{N}$ respectively, as they are described in section 4.2.1. P_{SF} is a subset of the step places used to specify the set of starting and finishing places of the given composite step net by removing the duplicates of the gluing places of the composed nets.

Another set of places of $\mathcal{N}$ needed for the definition of the construction is the set of inhibitor places:

$$P_I = \{p \in P \mid \exists_{t \in T}\ (p, t) \in I\} \tag{$\star\star$}$$

This set refers to a sub-set of the set of control places, the inhibitor places. An inhibitor place is always a control place but not the other way round. This set is necessary for the definition of the $Post'$ mapping of the constructed T-APN net for the specification of the additional arcs and their location.

Using indices for all the set of places of $\mathcal{N}$, we can easily identify every place from P_R, P_D and P_C and the sub-net of $\mathcal{N}$ they originate from. For instance, as is mentioned above, index 0 stands for the root net, so p_c^0 is a control place of the root net, $P_{ST}^0 = \{p^0\}$ is the only step place of the root net and $P_r^0 = \varnothing$, $P_d^0 = \varnothing$ are the response and database places of the root net respectively.

The transitions of $\mathcal{N}$ are composed of retrieve, emptying and step transitions as follows:

$$T = T_R \cup T_{EM} \cup T_{STEP} \qquad (\star\star\star)$$

where:
$T_R = \bigcup_{i=1}^{n} T_r^i$,
$T_{EM} = \bigcup_{i=1}^{n} T_{em}^i$,
$T_{STEP} = \bigcup_{i=1}^{n} T_{ST}^i \cup \{\epsilon\} = \bigcup_{i=1}^{n} \{t_s^i\} \cup \{\epsilon\}$.

We can easily trace every transition of the composite APN net back to the component it comes from. The root net does not contribute to T, though it has one 'collapsed' transition denoted by ϵ if we consider that its only step place p^0 is both starting and finishing place of ϵ.

Furthermore, we observe that for every $t \in T_{STEP}$ there exists exactly one post-place p of t such that $p \in P_{SF}$. We will denote such a place p_t.

Finally, for the sake of simplicity the notations Cl_{ID} and Cl_{SPEC} will be used in the construction for the sets of identity and special colours respectively. These sets are defined as follows:
$Cl_{ID} = \bigcup Cl$ and
$Cl_{SPEC} = \bigcup \overline{Cl}$.

The indices of the sets of places and transitions will not be used in the definition of the construction, but are useful for the precise description of the structure of the given APN net $\mathcal{N}$.

6.2.2 Formal Definition of Construction

Having described all the necessary information about the places, the transitions and the colour-sets of the original composite step net $\mathcal{N}$, we continue with the construction of the T-APN net associated with $\mathcal{N}$.

Definition 21 (Construction). *Let $\mathcal{N} = (P, T, Pre, Post, I, Cl, C, K, M_0, G)$ be a composite step APN net. A T-APN net $\varphi(\mathcal{N}) = (P', T', Pre', Post', Cl', \overline{Cl}, C_P, C_T, Ins, K', Gd, M_0', G')$ associated with $\mathcal{N}$ is defined as follows:*

(i) $P' = P$.

(ii) $T' = T$.

(iii) Pre', $Post' : T' \to \mu P'$ are defined as follows:

$$Pre'(t)(p) = \begin{cases} 1 & \textit{if } (p,t) \in I, \\ Pre(t)(p) & \textit{otherwise.} \end{cases}$$

$$Post'(t)(p) = \begin{cases} 1 & \textit{if } Post(t)(p) = 0 \wedge \big((t \in T_{EM} \wedge p \in \\ & P_C \cap P_I \wedge Pre(t)(p) > 0) \vee (t \in T_{STEP} \\ & \wedge\, p = cp^{\widehat{t}}(p_t) \wedge \widehat{t} \in (T_{STEP} \setminus \{t\}))\big), \\ Post(t)(p) & \textit{otherwise.} \end{cases}$$

(iv) $Cl' = Cl$.

(v) $\overline{Cl}$ *is a structuring set built as a complementary set to the structuring set* Cl *(given by bijection* $f{:}(\bigcup Cl) \to (\bigcup \overline{Cl})$*)* .

(vi)

$$C_P(p) = \begin{cases} C(p) & \textit{if } p \in (P_D \cup P_R \cup P_{SF}), \\ C(p) \cup \{f(g) \mid g \in C(p)\} & \textit{if } p \in P_C. \end{cases}$$

(vii) $\forall\ t \in T$:

$$C_T(t) = C(t).$$

(viii)

$$Ins(a) = \begin{cases} \{f(col) \mid col \in Cl_{ID}\} & \textit{if } a \in Pre_{REP} \vee a \in Post_{NEW}, \\ \{col \mid col \in Cl_{ID}\} & \textit{otherwise.} \end{cases}$$

where $Pre_{REP} = \{(p,t) \mid (t,p) \in I\}$ *and* $Post_{NEW} = \{(t,p) \mid Post(t)(p) = 0 \wedge Post'(t)(p) = 1\}$.

(ix)

$$K'((p,c)) = \begin{cases} K((p,c)) & \textit{if } p \in P \wedge c \in Cl_{ID}, \\ 1 & \textit{if } p \in P \wedge c \in Cl_{SPEC}. \end{cases}$$

(x) $Gd : \widetilde{T}' \to \{true, false\}$, *is a Boolean guard function defined as follows:*

For $t \in T_G$*:*

$$Gd((t,c)) = \begin{cases} true & \textit{if } (c, f(c)) \in B_t \\ false & \textit{otherwise} \end{cases}$$

where B_t *is a set of binding pairs:*

$$B_t = \{(x, f(x)) \mid x \in C(t)\}$$

and for $t \in T_{NG}$*:*

$$Gd((t,c)) = true$$

where $T' = T_G \cup T_{NG}$.[1]

(xi) Initial marking M'_0 *is defined as follows:*

$$M'_0((p,g)) = \begin{cases} M_0((p,g)) & \mathit{if}\, p \in P \setminus \{p^0_c\} \wedge g \in Cl_{ID}, \\ 1 - M_0((p,g)) & \mathit{if}\, p = p^0_c \wedge g \in Cl_{ID}, \\ 1 - M_0((p, f^{-1}(g))) & \mathit{if}\,\big(p \in (P_C \cap P_I) \wedge (\exists_{t \in T_{STEP} \setminus \{\epsilon\}} : \\ & Post(t)(p) = 0 \wedge Post'(t)(p) = 1 \wedge \\ & M_0((p_t, f^{-1}(g))) > 0) \wedge g \in Cl_{SPEC}\big) \\ & \vee \big(p \in (P_C \cap P_I) \wedge (\exists_{t' \in T_{EM}} : \\ & Post(t')(p) = 0 \wedge Post'(t')(p) = 1) \wedge \\ & (\exists_{t \in T_{STEP} \setminus \{\epsilon\}} : p = cp^t(p_t) \wedge \\ & M_0((p_t, f^{-1}(g))) > 0) \wedge g \in Cl_{SPEC}\big) \\ & \vee \big(p = p^0_c \wedge g \in Cl_{SPEC}\big), \\ 0 & otherwise. \end{cases}$$

where f^{-1} *is the inverse function of bijection* f*.*

(xii) $G' = G$.

At this point, we explain, step by step, the construction defined above by providing the meaning of each relation separately.

(i) In the construction above it can be noticed that the number of places of the constructed T-APN is equal to the number of places in the APN net. Moreover, it can be concluded that for every place of the APN net there exists a respective place in the constructed T-APN net.

(ii) The same remarks, like those of the places above, can be applied to the transitions.

(iii) For the *Pre*′ mapping of the constructed $\varphi(\mathcal{N})$ net, it can be noticed that every colour-sensitive inhibitor arc of $\mathcal{N}$ is replaced by a direct arc. Additionally, every direct arc of the *Pre* mapping of $\mathcal{N}$ remains structurally the same in the constructed $\varphi(\mathcal{N})$ net. For the *Post*′ mapping of the $\varphi(\mathcal{N})$ net

[1] T_G consists of all those transitions of the T-APN net that are associated with transitions of the APN net that have inhibitor places. The remaining transitions of the T-APN net belong to T_{NG}.

we notice that new direct arcs are added to the T-APN net. Firstly, new direct arcs are added starting from the step transitions and ending at all the control places of the post-places of those transitions. These additional arcs are generated to maintain the functionality of the net when either a unidirectional step is followed by a bidirectional one or a bidirectional step is followed by another bidirectional step. Secondly, new direct arcs are added between all the emptying transitions and the respective control places, starting from the emptying transitions and ending at those control places. Thirdly, new direct arcs are added between the step transitions and the control place of the root net when the 'starting' place (or root place) of the $\varphi(\mathcal{N})$ net can be revisited in the future. In other words, new direct arcs that connect the step transitions that have as post-place the 'starting' place of the $\varphi(\mathcal{N})$ net with the control place of the root net are added when the 'starting' place is place of one or more bidirectional steps. The last case is a special case of the first case with the new direct arcs between the step transitions and the control places. All the above 'post' cases are described in the first part of the $Post'$ mapping. Finally, regarding the rest of the post mapping of the constructed T-APN net $\varphi(\mathcal{N})$, all the other post arcs remain the same as in $\mathcal{N}$.

(iv) The Cl' structuring set of the $\varphi(\mathcal{N})$ net that defines the set of the identity colours of the net is the same as the Cl structuring set of $\mathcal{N}$.

(v) The $\overline{Cl}$ structuring set provides the 'special' colours of tokens that are used to maintain the functioning of the constructed $\varphi(\mathcal{N})$ net the same as that of the APN net $\mathcal{N}$.

(vi) The C_P function of the $\varphi(\mathcal{N})$ net is related to the colour function C of the APN net through the function above. If the places of $\varphi(\mathcal{N})$ to be structured are database, response or step places then in both nets these places will be structured taking colours that derive from function C. On the other hand, all the control places of $\varphi(\mathcal{N})$ will be structured taking colours that derive from both function C and the set of complementary colours, i.e., the special colours.

(vii) The transitions of both nets are structured taking colours from the same structuring set as is defined above by the relation of the two colour functions C_T and C.

$(viii)$ Actually, Ins function defines what kind of coloured tokens can be carried by each arc of $\varphi(\mathcal{N})$. This function is not related with any colour function of the $\mathcal{N}$ as all the arcs of $\mathcal{N}$ are not considered as structured and can carry any colour of the set Cl, but it could be related to the structuring set Cl to show how the colours of the arc inscriptions can derive from that set. Contrary to $\mathcal{N}$, in $\varphi(\mathcal{N})$ not all the arcs can carry the same colours of tokens as the inscriptions are given by the disjoint structuring sets Cl and $\overline{Cl}$ (see section 5.1). This is necessary for maintaining the behaviour of $\mathcal{N}$ in the T-APN net $\varphi(\mathcal{N})$. The inscriptions of the arcs are decided according to where these arcs belong to. In few words, all the newly added direct arcs carry tokens of special colours only, which is described by the first point of the function and all the other arcs carry tokens identity colours as described by the last point of that function.

(ix) The capacity of the places of $\mathcal{N}$ is maintained the same for the respective places of the constructed net $\varphi(\mathcal{N})$ when the colours of the tokens belong to the identity set of colours. Otherwise, K' returns 1 for each place of $\varphi(\mathcal{N})$, when the colours of the tokens of that place belong to the special set of colours.

(x) The APN net $\mathcal{N}$ has no guards. Guards are introduced in $\varphi(\mathcal{N})$ in order to replace in some sense the absence of the colour-sensitive inhibitor arcs, which are doing in $\mathcal{N}$ what the guards do in $\varphi(\mathcal{N})$. From the theory of the T-APNs it follows that every structured transition is accompanied by a guard, but only the guards of T_G transitions depend on the set of *binding pairs* B_t for their evaluation to either true or false. Thus, the relation B_t shows how the set of *binding pairs* of $\varphi(\mathcal{N})$ can be constructed through the colour function C of $\mathcal{N}$. As is known, the guards of the T_{NG} transitions are set to true. It should be mentioned that all the transitions that belong to T_G are those transitions of which the respective transitions in $\mathcal{N}$ have at least one inhibitor place. All the other transitions of $\varphi(\mathcal{N})$ belong to the T_{NG} set of transitions. Hence, from definition 21(ii) and ($\star\star\star$), it follows that $T_{STEP} = T_G$ and $T_{EM} \cup T_R = T_{NG}$.

(xi) The initial marking of $\varphi(\mathcal{N})$ depends on the marking of $\mathcal{N}$ out of which $\varphi(\mathcal{N})$ is constructed. More specifically, the initial marking of $\varphi(\mathcal{N})$ is defined with respect to the places that host the tokens and with respect to the colours of tokens that are held in each place of $\mathcal{N}$. If the place of $\varphi(\mathcal{N})$ that we examine is database, response or step place or control place of the same building block with that of the step place controlled by it and the colour of token in the place of $\mathcal{N}$ belongs to the 'identity' set of colours, then the marking of the examined place of $\varphi(\mathcal{N})$ remains the same as it is in the respective place of $\mathcal{N}$. This case is described in the first point of the function. If the place that we examine is the root control place and the colour of token in the respective place of $\mathcal{N}$ belongs to the 'identity' set again, then the marking of that place is defined by the second point of the marking function. On the other hand, if the place of $\varphi(\mathcal{N})$ that we examine is a control place of a step place that belongs to a different building block from that of the step place that is 'controlled' by it, then the marking of the control place is defined according to the formula that is given in the third part of the function. It should be noted that the initial marking of the root control place and all the other control places of the 'starting' place of the net is defined using the special colours given by the formula of the third condition of the initial marking relation. The following paragraphs describe different examples of both the second and third case of the construction of the initial marking. If the place that is examined is the control place of the root net of the $\varphi(\mathcal{N})$ net, then the marking of this place takes the values of all the complementary colours of the tokens that reside in the step place 'controlled' by it, when the respective control place of the root net of $\mathcal{N}$ holds no tokens at all. Otherwise, if the control place of the root net of $\mathcal{N}$ hosts at least one token, then the control place of the root net of the $\varphi(\mathcal{N})$ net takes the values of all the complementary colours of the tokens that reside in the respective step place, excluding that or those complementary value(s) that correspond(s) to the colour(s) of the token(s) that reside(s) in the control place of the root net of $\mathcal{N}$. If the places that are examined are the control places of the 'starting' place of the $\varphi(\mathcal{N})$ net, then the marking of these places takes the values of all the special colours of the tokens that reside in the step place 'controlled' by each of them, when the respective control place of the root net of $\mathcal{N}$ holds no tokens at all. On the other hand, when the control place of the root net of $\mathcal{N}$ holds at least one token, then the control places of the 'starting' place of the $\varphi(\mathcal{N})$ net take the values of all the complementary colours of the tokens that reside in the 'starting' place except for that or those complementary value(s) that correspond(s) to the colour(s) of the token(s) that reside(s) in the control place of the root net of $\mathcal{N}$.

(xii) All the gluing places of $\mathcal{N}$ are considered as gluing places of the constructed $\varphi(\mathcal{N})$ net. More specifically, the places of the $\varphi(\mathcal{N})$ net that correspond to the gluing places of $\mathcal{N}$, are the gluing places of the constructed $\varphi(\mathcal{N})$.

6.3 A Construction Example

After the definition of the construction, we demonstrate the application of the above definition through an example of the construction of a T-APN net $\varphi(\mathcal{N})$ out of a given APN net $\mathcal{N}$ just by following the relations between the elements of the two nets that are given by that definition. The given APN net $\mathcal{N}$ is presented in Figure 6.1.

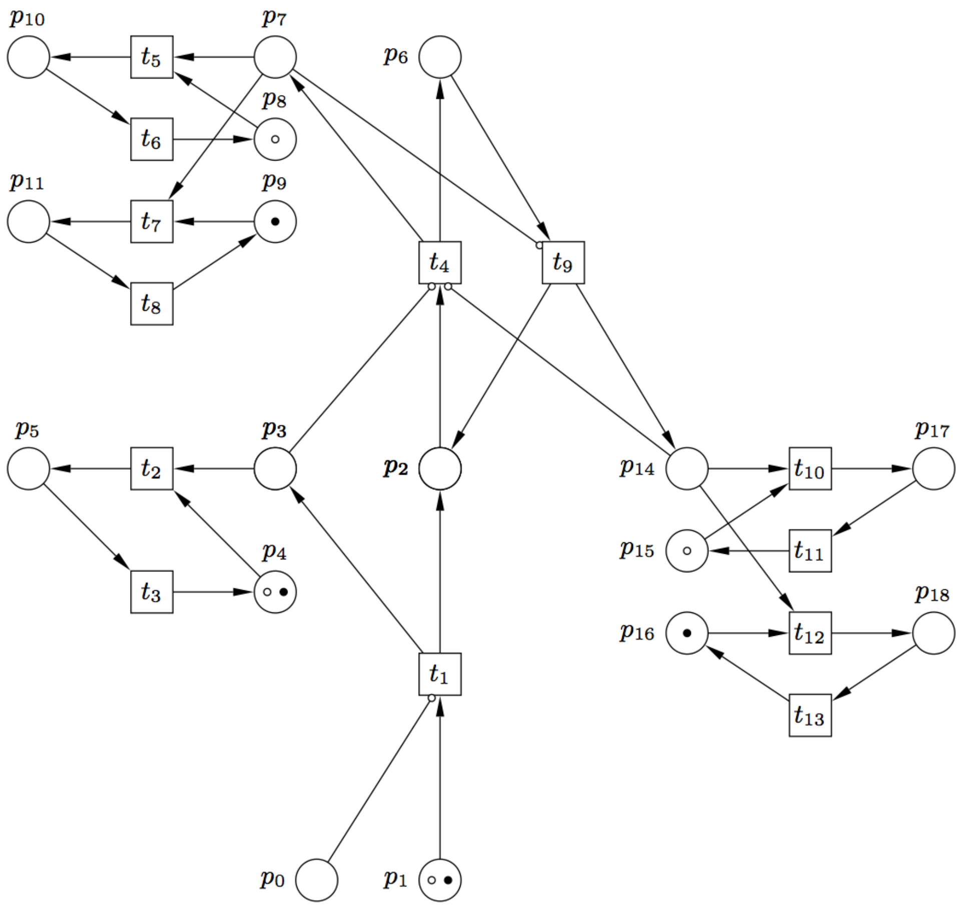


Figure 6.1: The APN net $\mathcal{N}$.

Knowing the structure of the given APN net $\mathcal{N}$ in Figure 6.1 and following the construction we build the respective T-APN net $\varphi(\mathcal{N})$, which is associated to $\mathcal{N}$. It should be noted that in the following example, the labelling of the elements of both nets is exactly the same, which means that the label of the places, the transitions, etc. are the same for $\mathcal{N}$ and $\varphi(\mathcal{N})$. This does not imply that two labels with the same value refer to the same element but to respective elements of the two nets. For example, place p_1 of $\varphi(\mathcal{N})$ is the respective place of p_1 of $\mathcal{N}$.

To proceed with the construction of $\varphi(\mathcal{N})$, first we have to identify every place and every transition and to sort them into the proper subset of places and transitions respectively. As is known from the theory of the APNs (see section

4.2.1), every composite step net has a specific structure that consists of database, response, control and step places and of emptying, retrieve and step transitions.

Starting with the places of $\mathcal{N}$, we could easily define the set P of all the places of the net, where:

$$P = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}\}$$

But, it is also known that for every net of composite step nets, P consists of subsets of special places:

$$P = P_R \cup P_D \cup P_C \cup P_{SF}$$

In this case the response, database, control and step places of $\mathcal{N}$ are as follows:

$P_R = \{p_5, p_{10}, p_{11}, p_{17}, p_{18}\}$,
$P_D = \{p_4, p_8, p_9, p_{15}, p_{16}\}$,
$P_C = \{p_0, p_3, p_7, p_{14}\}$,
and $P_{SF} = \{p_1, p_2, p_6\}$.

At this point we should mention that the control places of the 'starting' place are needed for the construction of the initial marking of the $\varphi(\mathcal{N})$ net and are as follows:

$$p_c^0 = \{p_0\}$$

which means that the 'starting' place p_1 has only one control place in this particular example, the place p_0. Finally, the set of inhibitor places P_I in this case is the same as the set of control places P_C. So, $P_I = \{p_0, p_3, p_7, p_{14}\}$.

Having defined all the sets of places, we continue with the construction of the places of the $\varphi(\mathcal{N})$ net according to definition 21(i), where it is described that for every place of the APN net $\mathcal{N}$ there exists a respective place in the T-APN net $\varphi(\mathcal{N})$. This means that for every database, response, control and step place of $\mathcal{N}$ there exists the respective database, response, control and step place in $\varphi(\mathcal{N})$, which implies that firstly the number of places in both nets is the same and secondly all the following:

$P'_R = P_R = \{p_5, p_{10}, p_{11}, p_{17}, p_{18}\}$,
$P'_D = P_D = \{p_4, p_8, p_9, p_{15}, p_{16}\}$,
$P'_C = P_C = \{p_0, p_3, p_7, p_{14}\}$ and
$P'_{SF} = P_{SF} = \{p_1, p_2, p_6\}$.

After the description of the construction for the places of $\varphi(\mathcal{N})$, we continue with the construction of the transitions of that net out of the transitions of $\mathcal{N}$. As is known, $\mathcal{N}$ comprises retrieve, emptying and step transitions:

$$T = T_R \cup T_{EM} \cup T_{STEP}$$

where:

$$T = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}\}$$

with
$T_R = \{t_3, t_6, t_8, t_{11}, t_{13}\}$,
$T_{EM} = \{t_2, t_5, t_7, t_{10}, t_{12}\}$ and
$T_{STEP} = \{t_1, t_4, t_9\}$.

Having defined the transitions of the APN net $\mathcal{N}$ and following once again the construction and particularly the second point that refers to the construction of the transitions of $\varphi(\mathcal{N})$, it follows that for every transition of $\mathcal{N}$ there exists a respective transition in $\varphi(\mathcal{N})$. This means that for every retrieve, emptying and step transition of $\mathcal{N}$ there exists the respective retrieve, emptying and step transition in $\varphi(\mathcal{N})$, which implies that the number of transitions in both nets is the same and that:
$T'_R = T_R = \{t_3, t_6, t_8, t_{11}, t_{13}\}$,
$T'_{EM} = T_{EM} = \{t_2, t_5, t_7, t_{10}, t_{12}\}$ and
$T'_{STEP} = T_{STEP} = \{t_1, t_4, t_9\}$.
Thus, from definition 21 (ii), (x), it follows that $T'_{STEP} = T_G$ and $T'_R \cup T'_{EM} = T_{NG}$. The sets of T_G and T_{NG} transitions will be used later for the guards.

Next, we identify all the pre and post arcs of $\mathcal{N}$ starting with the pre mapping first. The set of pre arcs, denoted by Pre_{ALL} is divided into direct pre arcs and inhibitor arcs, which are denoted by Pre_{DIR} and I respectively and are defined as follows:

$$Pre_{ALL} = Pre_{DIR} \cup I$$

where:
$Pre_{DIR} = \{(p_1, t_1),\ (p_3, t_2),\ (p_5, t_3),\ (p_4, t_2),\ (p_2, t_4),\ (p_7, t_5),\ (p_7, t_7),\ (p_8, t_5),$ $(p_9, t_7),\ (p_{10}, t_6), (p_{11}, t_8), (p_{14}, t_{10}), (p_6, t_9),\ (p_{14}, t_{12}), (p_{15}, t_{10}), (p_{16}, t_{12}),\ (p_{17}, t_{11}),$ $(p_{18}, t_{13})\}$
and $I = \{(p_0, t_1), (p_3, t_4), (p_7, t_9), (p_{14}, t_4)\}$.

According to point three that defines the construction of the pre arcs of the $\varphi(\mathcal{N})$ net with respect to the pre arcs of $\mathcal{N}$, every arc that belongs to the inhibitor arcs of $\mathcal{N}$ is replaced by the respective direct pre arc and every arc that belongs to the direct pre arcs of $\mathcal{N}$ corresponds to a direct pre arc in $\varphi(\mathcal{N})$. This implies that the total number of pre arcs of $\varphi(\mathcal{N})$ is the same as the number of pre arcs of $\mathcal{N}$ since that replacement retains the number of the pre arcs the exactly the same.

Hence,

$$Pre'_{ALL} = Pre'_{DIR} \cup Pre_{REP}$$

with
$Pre'_{DIR} = Pre_{DIR} = \{(p_1, t_1),\ (p_3, t_2),\ (p_5, t_3),\ (p_4, t_2),\ (p_2, t_4),\ (p_7, t_5),\ (p_7, t_7),$ $(p_8, t_5),\ (p_9, t_7), (p_{10}, t_6), (p_{11}, t_8),\ (p_{14}, t_{10}),\ (p_6, t_9),\ (p_{14}, t_{12}),\ (p_{15}, t_{10}),\ (p_{16}, t_{12}),$ $(p_{17}, t_{11}), (p_{18}, t_{13})\}$ and $Pre_{REP} = \{(p_0, t_1), (p_3, t_4), (p_7, t_9), (p_{14}, t_4)\}$, with Pre_{REP} being the set of direct arcs that replaces the respective set of inhibitor arcs.

Continuing with the post mapping of $\mathcal{N}$ we have the following set of all the post arcs of $\mathcal{N}$, denoted by $Post_{ALL}$:

$Post_{ALL} = \{(t_1, p_2), (t_1, p_3),\ (t_2, p_5),\ (t_3, p_4),\ (t_4, p_6),\ (t_4, p_7), (t_5, p_{10}), (t_6, p_8),$ $(t_7, p_{11}), (t_8, p_9), (t_4, p_{14}), (t_9, p_2), (t_{10}, p_{17}), (t_{11}, p_{15}), (t_{12}, p_{18}), (t_{13}, p_{16})\}$.

Now we provide the respective post mapping of $\varphi(\mathcal{N})$ using point three again that also defines the construction of the post arcs of $\varphi(\mathcal{N})$ with respect to the post arcs of the APN net $\mathcal{N}$. According to the construction of the post mapping for every arc that belongs to the set of post arcs of $\mathcal{N}$ there exists a respective post arc in $\varphi(\mathcal{N})$. Furthermore, new direct post arcs are added to the $\varphi(\mathcal{N})$ net between the emptying transitions and every inhibitor place and between step transitions and the control places of step places that are revisited. In this example, the new post arcs between the emptying transition and the inhibitor places are $\{(t_2, p_3), (t_5, p_7), (t_7, p_7), (t_{10}, p_{14}), (t_{12}, p_{14})\}$ as these arcs comply with the requirements of adding a new arc since all these transitions are emptying transition, the 'destination' places of these arcs are all inhibitor places and finally there exists a pre arc for every pair of the above post arcs. Finally, the remaining post arcs that are added to $\varphi(\mathcal{N})$ are $\{(t_1, p_{14}), (t_9, p_3)\}$ which are arcs that connect the step transitions t_1 and t_9 with the control places p_{14} and p_3 of the step place p_2 respectively, which is a step place that is revisited as it belongs to a bidirectional step.

All the newly added post arcs will be denoted by $Post_{NEW}$ and are as follows: $Post_{NEW} = \{(t_2, p_3), (t_5, p_7), (t_7, p_7), (t_{10}, p_{14}), (t_{12}, p_{14}), (t_1, p_{14}), (t_9, p_3)\}$.

From all the above, it can be concluded that: $Post'_{ALL} = Post_{ALL} \cup Post_{NEW} =$ $\{(t_1, p_2), (t_1, p_3), (t_2, p_5),\ (t_3, p_4), (t_4, p_6),\ (t_4, p_7), (t_5, p_{10}), (t_6, p_8), (t_7, p_{11}), (t_8, p_9),$ $(t_4, p_{14}), (t_9, p_2), (t_{10}, p_{17}), (t_{11}, p_{15}), (t_{12}, p_{18}), (t_{13}, p_{16})\} \cup \{(t_2, p_3), (t_5, p_7), (t_7, p_7),$ $(t_{10}, p_{14}),\ (t_{12}, p_{14}), (t_1, p_{14}),\ (t_9, p_3)\} = \{(t_1, p_2),\ (t_1, p_3), (t_2, p_5), (t_3, p_4), (t_4, p_6),$ $(t_4, p_7),\ (t_5, p_{10}),\ (t_6, p_8),\ (t_7, p_{11}),\ (t_8, p_9),\ (t_4, p_{14}),\ (t_9, p_2),\ (t_{10},\ p_{17}),\ (t_{11}, p_{15}),$ $(t_{12}, p_{18}), (t_{13}, p_{16}), (t_2, p_3), (t_5, p_7), (t_7, p_7), (t_{10}, p_{14}), (t_{12}, p_{14}), (t_1, p_{14}), (t_9, p_3)\}$.

Having completed the construction of places, transitions and arcs, we could say that we have developed the structure of $\varphi(\mathcal{N})$, which will be presented later as a graph. Moving from the structural parts of the nets to the functional ones, we will deal with the colour functions and the structuring sets that are used in both nets and we will show how they are related through our example.

Firstly, we start with the structuring set Cl. In our example, Cl for $\mathcal{N}$ is the structuring set of both places and transition and is as follows:

$$Cl = \{b, w\}$$

As is known, this means that the colours used in $\mathcal{N}$ are black and white. In definition 21(iv), Cl' is equal to Cl signifying that the same structuring set is also used in $\varphi(\mathcal{N})$.

For the construction of $\varphi(\mathcal{N})$ we need the complementary set of Cl, the $\overline{Cl}$ which is a set that consists of the complementary values of the Cl given by the bijection f. We assume that in our example, $\overline{Cl} = \{g, l\}$, which means that the complementary colours that will be used in $\varphi(\mathcal{N})$ will be the grey (g) and the light grey (l), with $f(b) = g$ and $f(w) = l$.

In $\mathcal{N}$, the colour function C structures all the places of the net using the structuring set Cl allowing the places to host either black or white tokens or both. For the structuring of the places of $\varphi(\mathcal{N})$ we use the colour function C_P which is related to the colour function C of $\mathcal{N}$ as stated in definition 21(vi). Checking each place of the T-APN net $\varphi(\mathcal{N})$, we can define how it is structured with respect to function C depending on where this place belongs to. For example all the response, database and step places of $\varphi(\mathcal{N})$ are structured using C_P which is set equal to C. On the other hand, all the control places of $\varphi(\mathcal{N})$ are structured by taking values from both C and the complementary set of colours.

Thus in $\varphi(\mathcal{N})$, the places $P'_C = \{p_0, p_3, p_7, p_{14}\}$ can host tokens of black, white, grey and light grey colour and the places $P'_R = \{p_5, p_{10}, p_{11}, p_{17}, p_{18}\}$, $P'_D = \{p_4, p_8, p_9, p_{15}, p_{16}\}$ and $P'_{SF} = \{p_1, p_2, p_6\}$ host tokens of black and white colours only.

Applying the construction to the colour function C_T for the structuring of the transitions of $\varphi(\mathcal{N})$, we notice that all the transitions of $\varphi(\mathcal{N})$ are structured in the same way as the transitions of $\mathcal{N}$, which means that all the transition of the constructed net work under black and white modes, as they take their values from function C, since $C_T = C$.

The last colour function that is needed for the functioning of the T-APN net $\varphi(\mathcal{N})$ is the Ins function that structures the arcs of $\varphi(\mathcal{N})$. To find the inscription, we first check where each arc of $\varphi(\mathcal{N})$ belongs to and according to that we decide the inscription of the arc. For example, all the arcs of $\varphi(\mathcal{N})$ that correspond to the arcs of $\mathcal{N}$ that belong to the pre and post arcs apart from those arcs that belong to the inhibitor arcs take their inscription from the Cl structuring set. In our case, all these arcs carry black and white tokens and their inscription will be written as $\{b, w\}$. Contrary to the arcs above, all the newly added arcs and the arcs that replace the inhibitor arcs of $\mathcal{N}$ carry tokens of the complementary colours of Cl. So, these arcs carry grey and light grey tokens and their inscription will be written as $\{g, l\}$.

Regarding the capacities of the places of the constructed net, we define the capacity of each place of $\varphi(\mathcal{N})$ in accordance with the capacity of the respective place in $\mathcal{N}$ only if the colour of token of the place is an identity colour. In that case, the capacity is the same for both places. Otherwise, if the colour of a token is a special colour then the capacity of the place is 1. In our example, we consider that every place of $\mathcal{N}$ has capacity 1 per colour of token, which implies that all the places of $\varphi(\mathcal{N})$ have also capacity equal to one for any colour of token, either identity colour or special colour.

The guards of the transitions T_G of $\varphi(\mathcal{N})$ evaluate to true for the set of *binding pairs* B_t and the guards of the transition T_{NG} are always set to true for that specific net. Considering that the transitions of $\mathcal{N}$ have no guards and that the guards of $\varphi(\mathcal{N})$ are Boolean functions that either depend on the set of *binding pairs* or not, we can only relate the transitions of the two nets only by showing that the set of *binding pairs* B_t derives from function C, as is described in definition 21(x). In our example, we know that function C takes values from Cl, which means that the variable x belongs to that set, *i.e.*, $x \in \{b, w\}$. Furthermore, $f(x)$ is given by bijection f, which means that $f(x)$ takes values from Cl_{SPEC}, which means that $f(x) \in \{g, l\}$.

The set of *binding pairs* of $\varphi(\mathcal{N})$ is as follows:

$$B_t = \{(b, g), (w, l)\}$$

with $f(b) = g$ and $f(w) = l$, as has already been mentioned.

All the step transitions are followed by guards that evaluate to true for any of the *binding pairs*. For all the other transitions of $\varphi(\mathcal{N})$, the guards always evaluate to true.

The most important part of the construction is the translation of the initial marking of $\mathcal{N}$ into the respective initial marking for the T-APN net $\varphi(\mathcal{N})$. Having defined all the places of both nets, we can easily proceed to the construction of the initial marking of $\varphi(\mathcal{N})$. As is described by definition 21(xi) , every place of $\varphi(\mathcal{N})$ retains the colours and the number of tokens that reside in the respective places of $\mathcal{N}$, except some special cases for the control places of the step places including the control places of the 'starting' place of the net, which will take values according to the second and third condition of the construction for the initial marking.

Specifically, checking every place of $\varphi(\mathcal{N})$ we concluded that the initial marking of the net is:

$$M'_0 = \{(p_0, g), (p_0, l), (p_1, b), (p_1, w), (p_4, b), (p_4, w), (p_8, b), (p_9, w), (p_{15}, b), (p_{16}, w)\}$$

as the places p_4, p_8, p_9, p_{15} and p_{16} are database places that are marked in $\mathcal{N}$, so they should remain marked with the same colours in $\varphi(\mathcal{N})$. Additionally, checking all the response places $p_5, p_{10}, p_{11}, p_{17}$ and p_{18} of $\mathcal{N}$ we noticed that none of them holds any token, therefore the respective places of $\varphi(\mathcal{N})$ will not hold tokens as well. As regards the step places p_1, p_2 and p_6 of $\mathcal{N}$, only p_1 is marked with black and white tokens, thus p_1 of $\varphi(\mathcal{N})$ will contain black and white tokens. Finally, regarding the initial marking of the control places of a step place, we could say that we have to check only those control places that their step place is marked or those that are control places of the 'starting' place of the net as is described by the second and third condition of the initial marking construction since the respective control place of an 'empty' control place of $\mathcal{N}$ will remain empty in $\varphi(\mathcal{N})$ net as well. As a result, we have to check the control places of p_1 of $\varphi(\mathcal{N})$ and decide how many and what colour of tokens they will host. In this case, the only control place of p_1 is the place p_0 which is a root control place and due to the fact that it satisfies all the prerequisites of the third condition, p_0 is marked using the formula of that condition of the initial marking construction taking the values g and l.

Finally, the last element that should be constructed in order to complete the construction of the T-APN net $\varphi(\mathcal{N})$ is its gluing places. To do so, we have to find the gluing places of $\mathcal{N}$ first and then to use the appropriate relation of the construction to define the gluing places of $\varphi(\mathcal{N})$. As is known from the composition of the APNs, every step place of an APN net that consists of composite step nets is a gluing place of that net. Since $\mathcal{N}$ is an APN net that consists of composite step nets, the set of gluing places consists of all step places of the net, *i.e.*, $G = \{p_1, p_2, p_6\}$. From the construction, we know that $G' = G$, thus the set of gluing places of $\varphi(\mathcal{N})$ is $G' = \{p_1, p_2, p_6\}$, which consists of the step place p_1, p_2 and p_6 of $\varphi(\mathcal{N})$.

The completion of the construction of the T-APN net $\varphi(\mathcal{N})$ that is associated with the given APN net $\mathcal{N}$ resulted in the net of Figure 6.2.

Observing the two nets, we notice that the structural differences of the two nets lie in the additional arcs of $\varphi(\mathcal{N})$. Moreover, we notice that the initial markings of the two nets differ as regards the number and the colours of the tokens.

6.4 Firing a Constructed T-APN net

In this section, we describe how a constructed T-APN $\varphi(\mathcal{N})$ net associated with a given APN net $\mathcal{N}$ behaves playing the token game in compliance with the enabling and the occurrence rules defined by the theory of the T-APNs. Moreover, we provide the concurrent reachability graph of both nets showing through the traces (step sequences) of the graphs that their behaviour is identical.

Due to the fact that the nets used in the previous example generate enormous reachability graphs, which makes it difficult to depict them in a clear and comprehensible way, a simple APN net, N_1, with just one token is used to construct

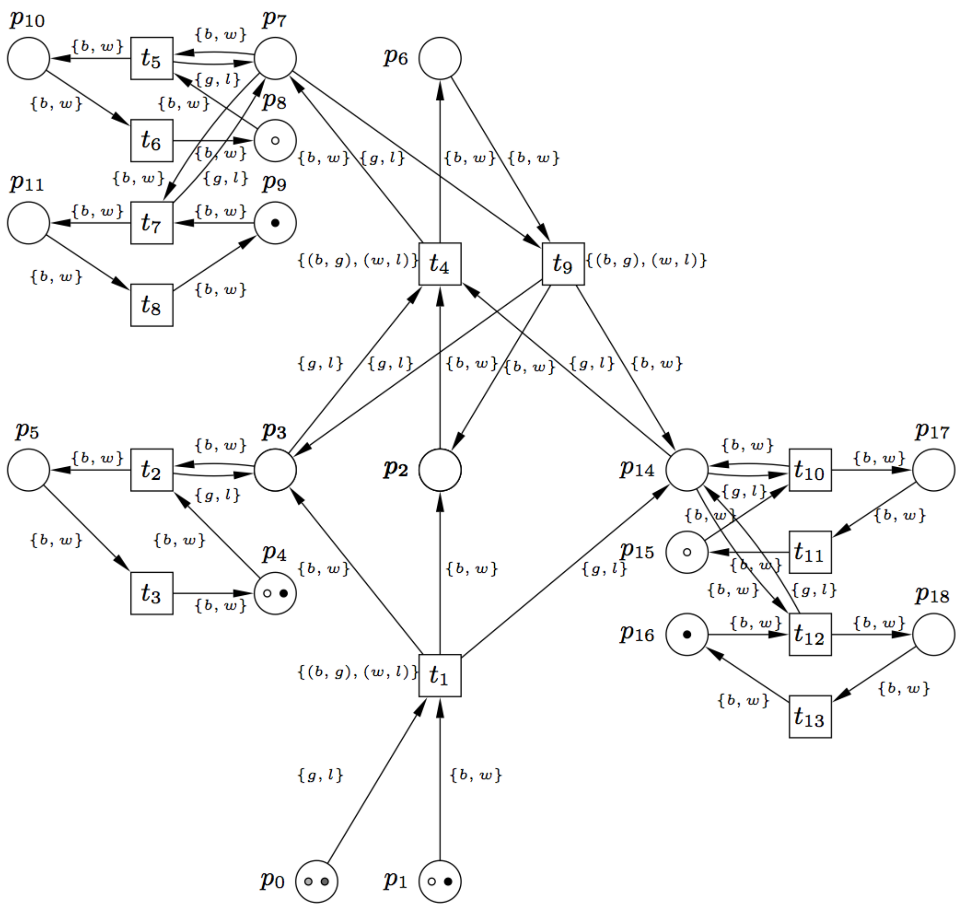


Figure 6.2: The constructed T-APN net $\varphi(\mathcal{N})$ associated with the APN net $\mathcal{N}$ from Figure 6.1.

the T-APN net $\varphi(N_1)$ through definition 21. The given APN net N_1 that will be used for the construction is shown in Figure 6.3.

Following exactly the same process as that described in the example of section 6.3, we construct the T-APN net $\varphi(N_1)$ of Figure 6.4. At this point we do not explain the construction of $\varphi(N_1)$ in detail, but we briefly provide all the necessary information, such as the colour functions, the structuring sets, the set of places, the set of *binding pairs*, etc.

The sets of places of N_1 are:

$P_R = \{p_5, p_9\}$,

$P_D = \{p_4, p_8\}$,

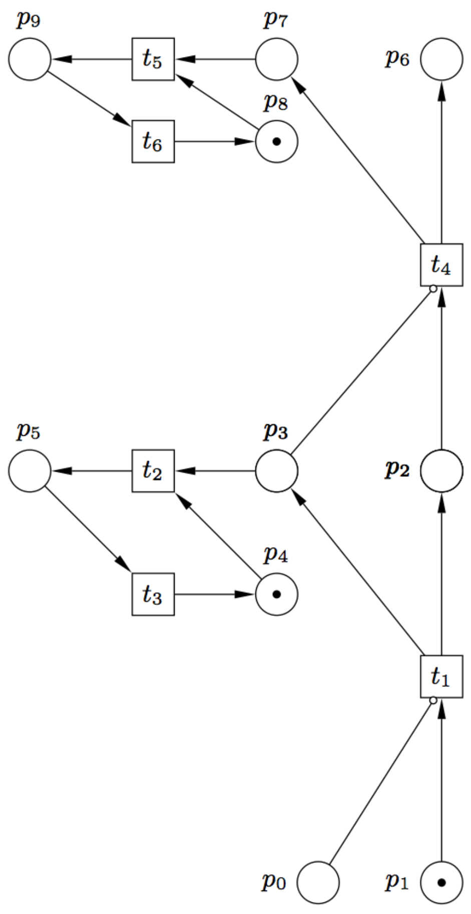


Figure 6.3: An APN net N_1.

$P_C = \{p_0, p_3, p_7\}$,
$P_{SF} = \{p_1, p_2, p_6\}$
and $P_I = \{p_0, p_3\}$.

The sets of places of $\varphi(N_1)$ are:
$P'_R = \{p_5, p_9\}$,
$P'_D = \{p_4, p_8\}$,
$P'_C = \{p_0, p_3, p_7\}$ and
$P'_{SF} = \{p_1, p_2, p_6\}$.

The sets of transitions of N_1 are:
$T_R = \{t_3, t_6\}$,
$T_{EM} = \{t_2, t_5\}$
and $T_{STEP} = \{t_1, t_4\}$.

The sets of transitions of $\varphi(N_1)$ are:
$T'_R = \{t_3, t_6\}$,
$T'_{EM} = \{t_2, t_5\}$
and $T'_{STEP} = \{t_1, t_4\}$.

Thus, from definition 21 (ii), (x), it follows that $T'_{STEP} = T_G$ and $T'_R \cup T'_{EM} = T_{NG}$. As with the previous example, the sets of T_G and T_{NG} transitions will be used later for the guards.

The sets of pre arcs of N_1 are:
$Pre_{DIR} = \{(p_1, t_1), (p_3, t_2), (p_4, t_2), (p_5, t_3), (p_2, t_4), (p_7, t_5), (p_8, t_5), (p_{10}, t_6)\}$ and $I = \{(p_0, t_1), (p_3, t_4)\}$.

The sets of pre arcs of $\varphi(N_1)$ are:
$Pre'_{DIR} = \{(p_1, t_1), (p_3, t_2), (p_4, t_2), (p_5, t_3), (p_2, t_4), (p_7, t_5), (p_8, t_5), (p_{10}, t_6)\}$ and $Pre_{REP} = \{(p_0, t_1), (p_3, t_4)\}$.

The set of post arcs of N_1 are:
$Post_{ALL} = \{(t_1, p_2), (t_1, p_3), (t_2, p_5), (t_3, p_4), (t_4, p_6), (t_4, p_7), (t_5, p_9), (t_6, p_8)\}$.

The set of post arcs of $\varphi(N_1)$ is:
$Post'_{ALL} = Post_{ALL} \cup Post_{NEW} = \{(t_1, p_2), (t_1, p_3), (t_2, p_5), (t_3, p_4), (t_4, p_6), (t_4, p_7), (t_5, p_9), (t_6, p_8)\} \cup \{(t_2, p_3)\} = \{(t_1, p_2), (t_1, p_3), (t_2, p_5), (t_3, p_4), (t_4, p_6), (t_4, p_7), (t_5, p_9), (t_6, p_8), (t_2, p_3)\}$.

The structuring set Cl contains only one element which is the black colour, so $Cl = \{b\}$ and the structuring set $\overline{Cl}$ contains only the grey colour, $\overline{Cl} = \{g\}$.

Regarding the structuring of the places of N_1, all the places can host black tokens. The same happens for all the places of $\varphi(N_1)$ except for the control places of the net that can host both black and grey tokens. The structuring of the transitions of N_1 through the function C allows the transitions of the net to work under the black mode. The transitions of $\varphi(N_1)$ work under the same colour mode as that of the transitions of N_1 as C_T is set equal to C.

The inscriptions for both the newly added arc and the arcs that replace the inhibitor arcs in $\varphi(N_1)$ are set equal to $\{g\}$ and for all the other arcs of the net is set equal to $\{b\}$.

In this example, all the places of N_1 have capacity equal to one due to the fact that we have only one colour of token each time in every place. Thus, using definition 21(ix) results that all the places of $\varphi(N_1)$ either maintain the same capacity for the identity colours of tokens of that of N_1 or get capacity equal to 1 for the special colours of tokens. In both cases, the capacity of each place of $\varphi(N_1)$ is one for this specific example, which ensures that every place is 1-bounded in this particular case.

Continuing with the set of *binding pairs*, it consists of only one pair, the $\{(b, g)\}$ pair, since the colour function that in some sense determines the *binding* *pairs* takes the value of only one colour. As a consequence this colour has a unique image given by bijection f and by taking these two colours we make the binding set $B_t = \{(b, g)\}$ with $f(b) = g$. Thus, this set is used as guard in our graphs for all the step transitions of $\varphi(N_1)$ as $T'_{STEP} = T_G$. The guards of all other transitions are always true, since $T'_R \cup T_{EM'} = T_{NG}$.

The initial marking of N_1 is:

$$M_0 = \{(p_1, b), (p_4, b), (p_8, b)\}.$$

Applying the construction for the initial marking (definition 21(xi)) and knowing the control places of the marked step places, it follows that place p_0 is the only control place of a marked step place in this example. Thus, we get the following initial marking for the T-APN net $\varphi(N_1)$:

$$M'_0 = \{(p_0, g), (p_1, b), (p_4, b), (p_8, b)\}.$$

Finally, the gluing places of $\varphi(N_1)$ are the places p_1, p_2 and p_6, which are all the step places of the net.

Although, the demonstration of the token game of the APN net N_1 is out of the scope of this example, we have produced the concurrent reachability graph of N_1, which is shown in Figure 6.5.

The states (markings) of the concurrent reachability graph of N_1 are as follows:

$M_0 = \{(p_1, b), (p_4, b), (p_8, b)\}$,
$M_1 = \{(p_2, b), (p_3, b), (p_4, b), (p_8, b)\}$,
$M_2 = \{(p_2, b), (p_5, b), (p_8, b)\}$,
$M_3 = \{(p_2, b), (p_4, b), (p_8, b)\}$,
$M_4 = \{(p_5, b), (p_6, b), (p_7, b), (p_8, b)\}$,
$M_5 = \{(p_4, b), (p_6, b), (p_7, b), (p_8, b)\}$,
$M_6 = \{(p_5, b), (p_6, b), (p_9, b)\}$,
$M_7 = \{(p_4, b), (p_6, b), (p_9, b)\}$,
$M_8 = \{(p_5, b), (p_6, b), (p_8, b)\}$
and $M_9 = \{(p_4, b), (p_6, b), (p_8, b)\}$.

On the contrary, the token game of the constructed $\varphi(N_1)$ net is extensively described starting from the initial state of the net that is indicated by the marking M'_0. As is stated by the marking M'_0, the initial state of $\varphi(N_1)$ consists of the places p_0, p_1, p_4 and p_8, which hold a grey, a black, a black and a black token respectively. Knowing the initial state, to fire the token game of $\varphi(N_1)$, we check if there exists any enabled transition for the given marking M'_0. A transition of $\varphi(N_1)$ is considered as enabled and may fire if all three conditions of the enabledness of the T-APNs are satisfied (see section 5.2). Firstly, the pre-places of the transition that is checked for enabling should contain adequate number and

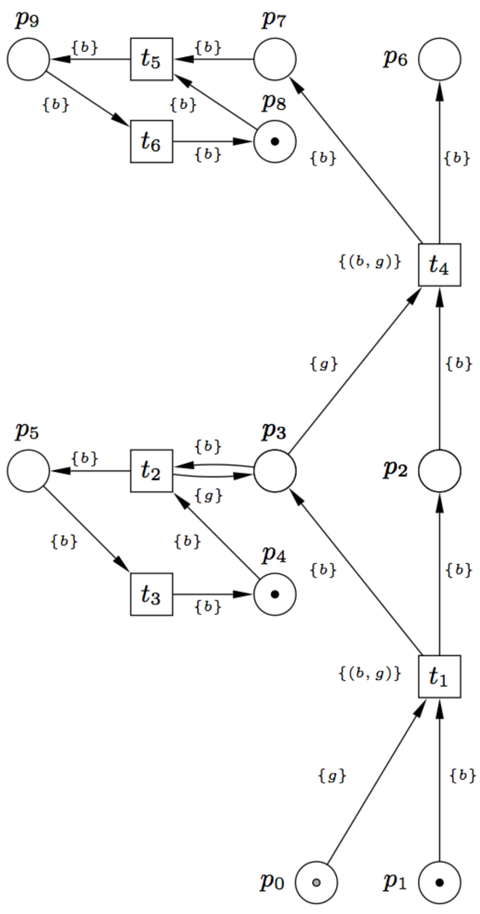


Figure 6.4: The constructed T-APN net $\varphi(N_1)$ associated with the APN net N_1 from Figure 6.3.

colours of tokens that will be consumed during the transition firing. Secondly, the number of tokens that will be deposited into the post-places of the fired transition plus the number of tokens that already reside into these places should not exceed the capacity of the places. Finally, if the prior two conditions hold, then we check whether the guard of that transition evaluates to true or false.

So, we should check which of the transitions t_1, t_2 and t_5 are enabled at M_0' as these are the only transitions that have tokens in their pre-places. Transitions t_2 and t_5 are not enabled as only one of their pre-places holds the appropriate number of tokens, violating in that way the first condition of the enabledness. Checking t_1, the first condition holds as both the appropriate number and colours of tokens that are required for the transition firing already exist in each pre-place

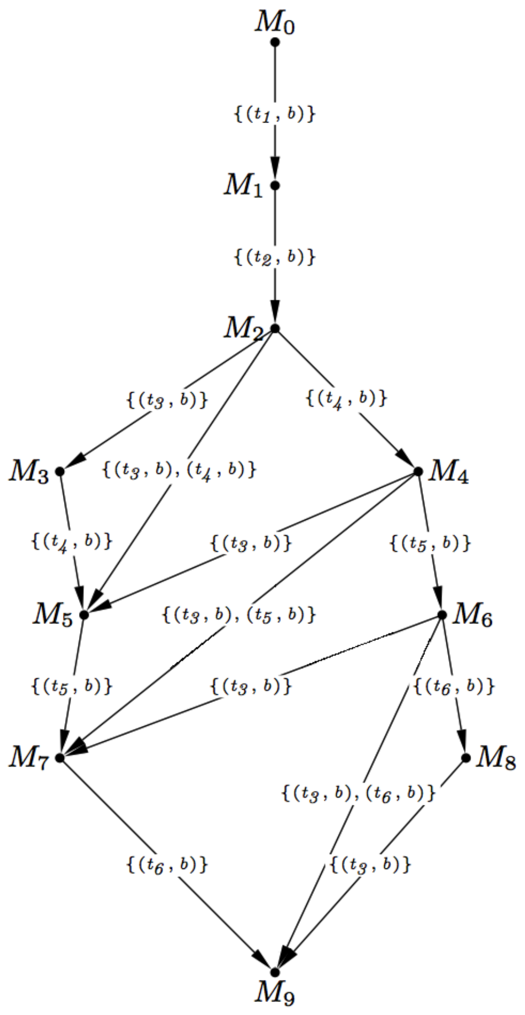


Figure 6.5: The concurrent reachability graph of the APN net N_1 ($CRG(N_1)$).

of t_1 and also comply with the inscriptions of the arcs that connect the pre-places with that particular transition. For example, the inscriptions of the arcs (p_0, t_1) and (p_1, t_1) are $\{g\}$ and $\{b\}$ respectively. Consequently, the first condition holds due to the fact that a grey token resides in p_0 and a black one in p_1. Furthermore, the second condition holds as the number of tokens that will be deposited in the post-places after the firing will not exceed the number defined by the capacity of these places. Additionally, the colours of the tokens comply with these described by the inscriptions of the post arcs (t_1, p_2) and (t_1, p_3) which both carry/deposit black tokens. Finally, the guard of t_1 evaluates to true for the grey and black pair of tokens as it is a valid binding pair.

Firing the enabled transition t_1 we consume the grey token from p_0 and the black token from p_1 and we put one black token to each of the post-places, p_2 and p_3. Then, checking the transitions t_2 and t_4 for firing, we notice that transition t_4 is not enabled as the first condition does not hold due to the fact that black colour of p_3 does not comply with the inscription of the (p_3, t_4) arcs which is set to $\{g\}$ as described by the $\widehat{Pre}$ in section 5.2. On the other hand, transition t_2 is enabled since all the conditions are met. The first condition holds because the pre-places p_3 and p_4 are equipped with a black token each and the inscriptions of the respective arcs are set to $\{b\}$, thus the tokens can be consumed. The second conditions is also satisfied since the black and the grey token that are deposited into the post-places p_5 and p_3 respectively do not violate the capacity restrictions of the places and comply with the inscriptions of the respective post arcs. The arc (t_2, p_5) has inscription $\{b\}$ and the arcs (t_2, p_5) has inscription $\{g\}$. Finally, t_2 is a T_{NG} transition and according to the T-APNs theory this means that the guard of this transition is always set to true.

Firing the enabled transition t_2 we consume the black tokens from p_3 and p_4 and deposit a black token in p_5 and a grey token in p_3. At this point, we should check the firing of the transition t_3 and t_4, which are both enabled. Firstly, transition t_3 is enabled because the pre-place p_5 holds the proper number and colour of token, which complies with the inscription of the pre arc (p_5, t_3). Secondly, the black token that is deposited in the post-place p_4, after the firing of t_3, can be carried by the arc (t_3, p_4) and does not violate the capacity restrictions of the place as p_4 was empty before the firing. Finally due to the fact that t_3 is a T_{NG} transition, it follows that the last condition is always met. Transition t_4 is enabled because the pre-places p_2 and p_3 hold the proper number and colours of tokens. The colours of these tokens also comply with the inscriptions of the pre arcs (p_2, t_4) and (p_3, t_4) respectively. Secondly, t_4 satisfies the second enabledness condition as the depositing of black tokens in the post-places p_6 and p_7, after the firing of t_4, does not violate the capacity restrictions of these post-places as they were empty before the firing. Furthermore, the tokens can be carried by the arcs (t_4, p_6) and (t_4, p_7) as both arcs have $\{b\}$ as inscriptions. Finally, the third condition of enabledness is satisfied due to the fact that the guard of t_4 evaluates to true for the pair (b, g).

Having two enabled transition t_3 and t_4, they may fire separately or concurrently. Let's assume that they fire separately. In that case, we examine what happens after the firing of each transition. Assuming that t_3 fires first, we deposit a black token in the post-place p_4. Now, checking again for potentially enabled transitions in $\varphi(N_1)$ after the firing of t_3, it results that the only enabled transition of the net is t_4, which was examined and proved enabled earlier. Therefore, we execute transition t_4, which deposits a black token in each of the places p_6 and p_7. At this point, it should be mentioned that if t_4 had fired first, the potentially enabled transitions would be t_3 and t_5 and if t_3 and t_4 had fired concurrently, the only transition that could be potentially enabled is t_5.

But, let's continue this process from the point where the firing of t_4 succeeded that of t_3 resulting in the depositing of a black token in each of the places p_6 and p_7 as was mentioned above. After the execution of t_4, the only transition that could be checked is t_5. Transition t_5 is enabled because the pre-places p_7 and p_8 hold the proper number and colours of tokens. The colours of these tokens also comply with the inscriptions of the pre arcs (p_7, t_5) and (p_8, t_5) which are both $\{b\}$. Secondly, t_5 is enabled due to the fact that the black token that is placed to the post-place p_9, after the firing of t_5, does not violate the capacity restrictions of the post-place as p_9 was empty before the firing. Furthermore, the deposited token can be carried by the arc (t_5, p_9) as its inscription is set to $\{b\}$. Finally due to the fact that t_5 is a T_{NG} transition, it turns that the last condition is always met.

After the firing of t_5, we check whether t_6 is enabled or not. Transition t_6 is enabled firstly because the pre-place p_9 holds the proper number and colour of token. The colour of this token also complies with the inscription of the pre arc (p_9, t_6) which is $\{b\}$. Secondly, t_6 is enabled because the black token that is placed to the post-place p_8, after the firing of t_6, does not violate the capacity restriction of the post-place as p_8 was empty before the firing. This token can be carried by the arc (t_6, p_8) as its inscription is set to $\{b\}$. Finally due to the fact that t_6 is a T_{NG} transition, it follows that the last condition is always met.

Finally, we observe that after the firing of t_6, no other transition is enabled as the marking consists of tokens that reside into places that cannot create the proper conditions for further transition firing. This marking is the 'terminal' state of $\varphi(N_1)$ and is as follows: $\{(p_4, b), (p_6, b), (p_8, b)\}$.

It should be noted that all the concurrent executions of the enabled transitions are omitted in the description above, but they are considered in the concurrent reachability graph of $\varphi(N_1)$, which is shown in Figure 6.6. Their description was skipped for the sake of simplicity and comprehension as the complexity of the writing would rise dramatically. Nevertheless, an example of enabled transitions that could have been executed concurrently was given above.

The states (markings) of the concurrent reachability graph of $\varphi(N_1)$ are as follows:

$M'_0 = \{(p_0, g), (p_1, b), (p_4, b), (p_8, b)\}$,
$M'_1 = \{(p_2, b), (p_3, b), (p_4, b), (p_8, b)\}$,
$M'_2 = \{(p_2, b), (p_3, g), (p_5, b), (p_8, b)\}$,
$M'_3 = \{(p_2, b), (p_3, g), (p_4, b), (p_8, b)\}$,

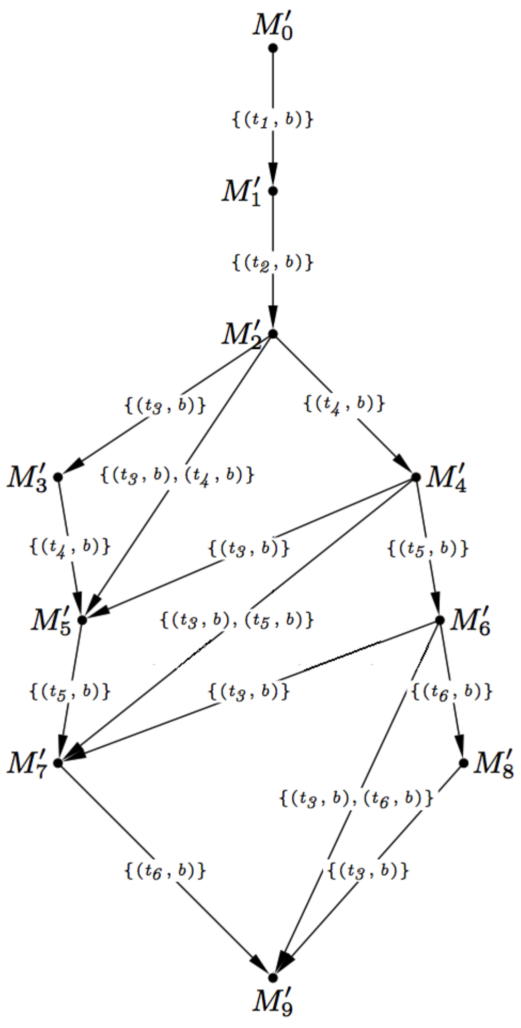


Figure 6.6: The concurrent reachability graph of the constructed T-APN net $\varphi(N_1)$ $(CRG(\varphi(N_1)))$.

$M'_4 = \{(p_5, b), (p_6, b), (p_7, b), (p_8, b)\}$,
$M'_5 = \{(p_4, b), (p_6, b), (p_7, b), (p_8, b)\}$,
$M'_6 = \{(p_5, b), (p_6, b), (p_9, b)\}$,
$M'_7 = \{(p_4, b), (p_6, b), (p_9, b)\}$,
$M'_8 = \{(p_5, b), (p_6, b), (p_8, b)\}$
and $M'_9 = \{(p_4, b), (p_6, b), (p_8, b)\}$.

Comparing the reachability graphs of the two nets, N_1 and $\varphi(N_1)$, we can show that the two nets behave in the same way by demonstrating that every possible trace of the concurrent reachability graph of N_1 can also be produced by the concurrent reachability graph of $\varphi(N_1)$.

In the paragraphs below we provide all the traces or step sequences of the concurrent reachability graph of N_1 in Figure 6.5 as they resulted from the token game of the net. Before the presentation of the step sequences, we should explain how these sequences are written and read. We denote a step sequence by writing one step after the other with the first executed step being always written on the left of the sequence. Finally, the steps used in the step sequences as are follows:

$U_1 = \{(t_1, b)\}$,
$U_2 = \{(t_2, b)\}$,
$U_3 = \{(t_3, b)\}$,
$U_4 = \{(t_4, b)\}$,
$U_5 = \{(t_5, b)\}$,
$U_6 = \{(t_6, b)\}$,
$U_7 = \{(t_3, b), (t_4, b)\}$
$U_8 = \{(t_3, b), (t_5, b)\}$
and $U_9 = \{(t_3, b), (t_6, b)\}$.

Hence, the step sequences of N_1 are:

(1) $U_1U_2U_3U_4U_5U_6$,
(2) $U_1U_2U_4U_3U_5U_6$,
(3) $U_1U_2U_4U_5U_3U_6$,
(4) $U_1U_2U_4U_5U_6U_3$,
(5) $U_1U_2U_7U_5U_6$,
(6) $U_1U_2U_4U_8U_6$ and
(7) $U_1U_2U_4U_5U_9$.

If the behaviour of the two nets is equivalent, then taking the concurrent reachability graph of $\varphi(N_1)$ we should be able to generate, for every single step executed in N_1, exactly the same step sequences. Consequently, for the steps given above, U_1, U_2, ..., U_9, we should get the sequences (1) to (7).

An easy way to determine if the step sequences of the N_1 can also be generated by $\varphi(N_1)$ is by trying to apply the step sequences to the reachability graph of Figure 6.6 in order to figure out whether all of them can by applied or not. If all the step sequences are applicable then the constructed T-APN net $\varphi(N_1)$ behaves identically to the given APN net N_1.

Examining the concurrent reachability graph of $\varphi(N_1)$ with respect to the produced step sequences of N_1, we realise that $\varphi(N_1)$ can generate exactly the same sequences. To make it more specific, $\varphi(N_1)$ does not produce other step sequences than those produced by N_1, which makes the two nets behaviourally equivalent.

6.5 T-APN Models of Case Studies

In this section, we discuss the constructed T-APN models associated with the APN models of the chosen case studies. Due to the fact that the APN models of the ambient garage and the ambient conference room are significantly large, we concentrate on the presentation of the constructed T-APN models[1] that are associated with the fragmented APN models shown in section 4.5[2]. Demonstrating that the construction works for these models, it can be deduced that the full T-APN models of the two cases studies can be produced by applying the construction to the full APN models. Finally, the aim of this section is not to explain the construction of the T-APN models in detail but to give a flavour of the elements of the T-APN models that will be used for the verification of the ambient garage and the ambient conference room respectively.

6.5.1 Constructed T-APN Model of Ambient Garage

The construction of the T-APN net of the ambient garage is based on the APN net presented in Figure 4.14, which consists of forward and backward compositions (see sections 4.3.1 and 4.3.2). Applying the construction on that APN net, the constructed T-APN net of Figure 6.7 is produced.

From the construction, it follows that the number of places and transitions of the produced T-APN model is the same as that of places and transitions of the given APN net. Furthermore, it follows that all the pre and post arcs of the APN net are maintained in the T-APN net.

On the contrary, as it can be noticed from the comparison of Figures 4.14 and 6.7, the constructed T-APN net differs from the respective APN net as regards the structuring sets, the additional arcs, the arc inscriptions, the guards of the transitions and the colours of tokens that reside in the places of the net.

Starting with the structuring sets, it should be mentioned that the structuring set of identity colours of the T-APN net is the same as that of the APN net. The structuring set of special colours of the T-APN net is given by bijection f, which has as domain the values of the set of identity colours. In the T-APN net of Figure 6.7, x stands for the set of identity colours and xy represents the set of special colours. The structuring set x consists of the elements 1 and 2 which represent the IDs of the users. On the other hand, the set of special colours xy contains the elements 0 and 3, with $f(1) = 0$ and $f(2) = 3$.

The additional arcs of the T-APN net consist of all the pre arcs that replace the respective colour-sensitive inhibitor arcs of the APN net, the post arcs between

[1]The constructed T-APN model are produced using Snoopy modelling tool [29].

[2]The naming of the nodes (places and transitions) of the constructed T-APN models is adapted from the naming of the nodes of the associated APN models.

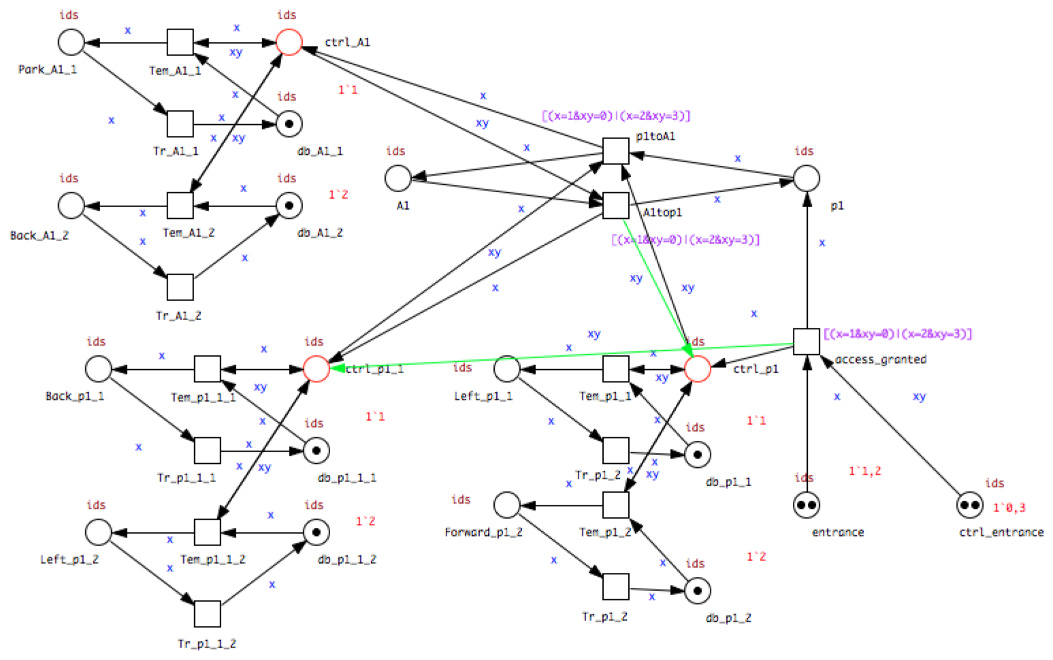


Figure 6.7: Constructed T-APN net associated with APN net of Figure 4.14.

the emptying transitions and the control places of the building blocks and the post arcs that are used for the ‘internal’ functioning of the net when there exists a bidirectional step (e.g. green arcs in Figure 6.7).

Continuing with the arc inscriptions, it should be mentioned that all the new direct arcs of the T-APN net take xy as inscription and all the other direct arcs take x as inscription.

For the guards of the transitions of the constructed T-APN net, we consider two different cases, the guards of T_G and T_{NG} transitions respectively. Regarding the guards of the T_G transitions, the set of *binding pairs* for which the step transitions are enabled is represented in Snoopy [29] by the following function:

$$[(x = 1\&xy = 0) \mid (x = 2\&xy = 3)]$$

On the other hand, the guards of all the other transitions are always true as they belong to the T_{NG} transitions. Hence, they can be enabled only by fulfilling the first two conditions of the enabling rules of the T-APNs (see section 5.2) since the third one is always satisfied. For that reason, the guards of the T_{NG} are discarded from the T-APN model of the ambient garage.

Finally, the initial marking of the T-APN net with respect to the initial marking of the APN net is as follows:

$$M_0 = \{(ctrl\_entrance, 0), (ctrl\_entrance, 3), (entrance, 1), (entrance, 2),\\ (db\_p1\_1, 1), (db\_p1\_2, 2), (db\_p1\_1\_1, 1), (db\_p1\_1\_2, 2), (db\_A1\_1, 1), (db\_A1\_2, 2)\}$$

where *ctrl_entrance* is the only control place that carries special tokens at M_0.

The presence of the structuring set of special colour, additional arcs, arc inscriptions and guards in the T-APN net of the ambient garage is needed for the representation of the colour-sensitive inhibitor arcs that are used in the associated APN net, keeping in that way the functioning of the two nets the same.

At this point, applying the construction to the APN net of Figure 4.15, the constructed T-APN net of the ambient garage is created and is shown in Figure 6.8.

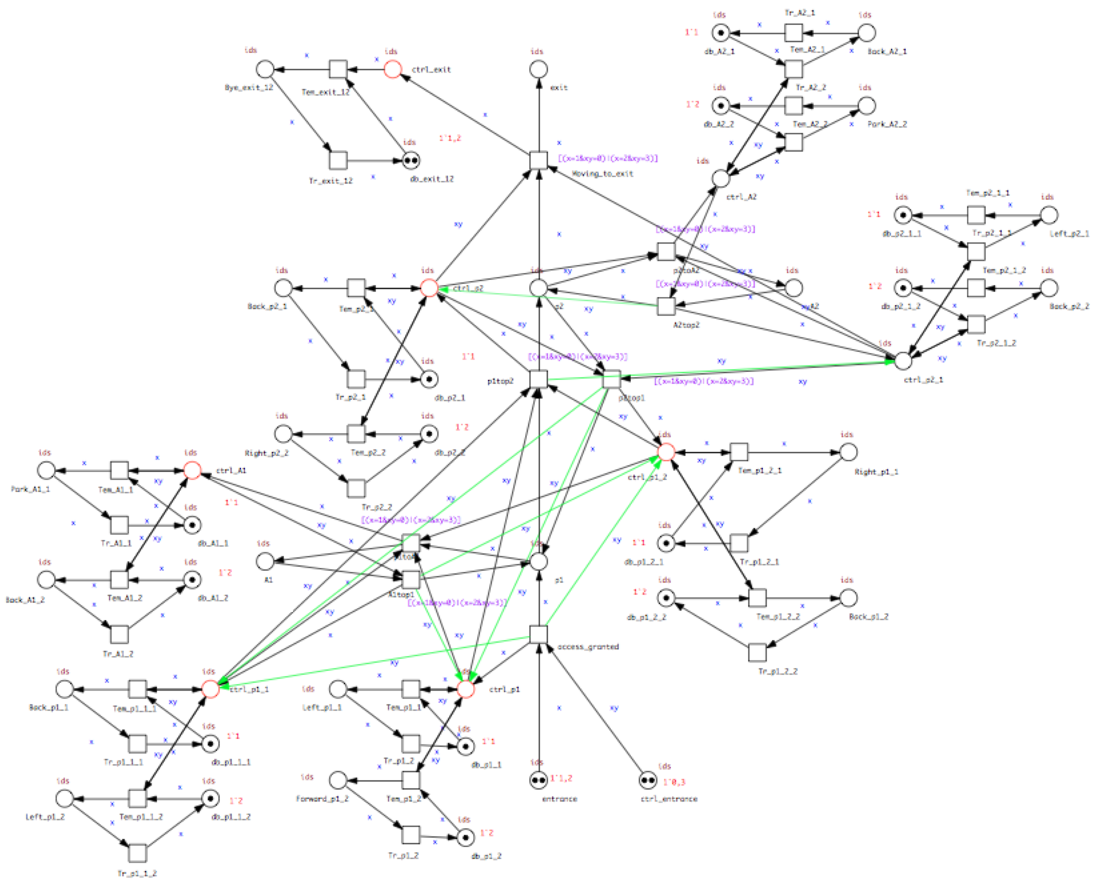


Figure 6.8: The T-APN net of ambient garage in Snoopy.

Now, to make the T-APN net of the ambient garage more realistic, as with the respective APN net in section 4.5.2, we have to bound the step places of the T-APN model. To do so, we propose two different ways, one practical solution that is based on the addition of direct arcs and tokens to the model of the garage and one theoretical solution that is similar to that of section 4.5.2, but adapted to the T-APN class.

Starting with the practical solution of bounding the step places of the model, which represent the physical areas of the garage that are monitored by the sensors, we can bound them by inserting a direct arc with direction from the last step transition of the basic step net that terminates the operation of the system to the root control place of the model and by removing one of the complementary colours of token that reside in the root control place. For example, let's say that we want to bound the step places of the model in Figure 6.8 to only one car per area. To achieve that, firstly we add a direct arc that links the transition $Moving\_to\_exit$ with the root control place *ctrl_entrance* and secondly we remove one of the tokens that resides in the root control place forbidding in that way the temporary access to the respective car. Let's assume that in this case, the car that is temporarily not allowed to access the garage until the allowed car exits from it is that with the ID represented by the complementary token with colour '3', i.e., the car with ID equal to 2. In that way, we allow only one car to move around the garage each time and the next car is allowed to enter the garage only after the first one has already left it. This happens by depositing the appropriate colour of token in the root control place after the execution of the $Moving\_to\_exit$ transition for the first car enabling the access to the garage for the next car. Thus, using this practical solution, we can bound the step places of the T-APN model making it more realistic as regards the cars that could co-exist in the same area but less realistic regarding the free access of some authorised users to the garage as they have to wait till some other users exit from it regardless of the fact that there could be free parking spots in the garage. Finally, it should be noted that this solution fully complies with the definitions of the T-APN class and the construction, expect for that part of the construction that refers to the initial marking of the root net. For that reason, we have to modify the relation that gives the construction of the initial marking of the T-APN net out of the respective initial marking of the APN net by adding that case for the root control place.

On the other hand, the theoretical solution for the bounding of the net places fully complies with the definition of the construction, but it requires the modification of the enabledness rules of the T-APN class. More specifically, that solution follows the same pattern as the bounding solution given in section 4.5.2. Therefore, the additional rule that is needed in order to bound the places is as follows:

$$\forall p \in P, \forall c_i \in C_P(p) : \sum_{i=1}^{n} K((p, c_i)) \leq \ell$$

with $\ell \in \mathbb{N}$ being the upper bound of each net place.

Therefore, taking the total number of coloured tokens that are hosted in every net place and setting it less than or equal to ℓ, we can bound the places of an APN net. For example, in the case of the above model, we could set $\ell = 1$ declaring the upper bound for the total number of tokens per place equal to one. Finally, it should be mentioned that this solution gives a more realistic aspect to the modelling of the garage compared to the practical one as it does not affect the access of the users to that.

6.5.2 Constructed T-APN Model of Ambient Conference Room

Similarly to the construction of the T-APN net of the ambient garage, the construction of the T-APN net of the ambient conference room is related to the APN net presented in Figure 4.16, which consists of forward compositions only. Applying the construction to that APN net, the constructed T-APN net of Figure 6.9 is produced.

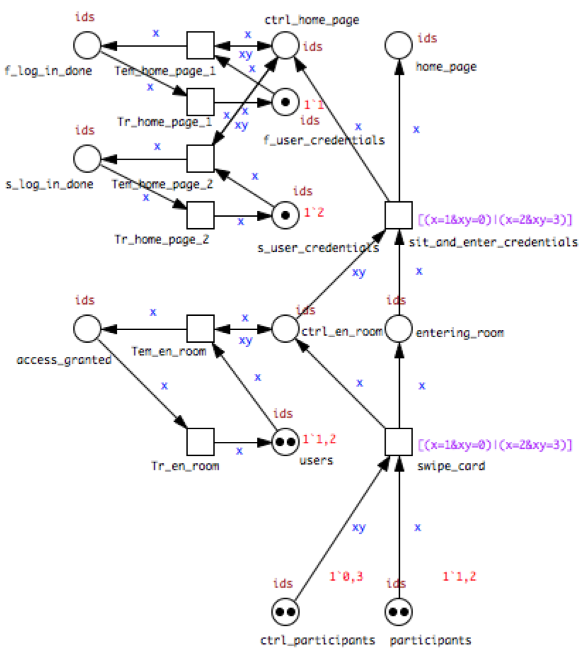


Figure 6.9: Constructed T-APN net associated with APN net of Figure 4.16.

From the construction, it follows that the number of places and transitions of the produced T-APN model is the same as that of places and transitions of the given APN net. Furthermore, it follows that all the pre and post arcs of the APN net are maintained in the T-APN net of the ambient conference room.

As it can be noticed from the comparison of Figures 4.16 and 6.9 that the two nets differ as regards the structuring sets, the additional arcs, the arc inscriptions, the guards of the transitions and the colours of tokens that can reside in the places of these nets.

Starting with the structuring sets, it should be mentioned that the structuring set of identity colours of the T-APN net is the same as that of the APN net. The structuring set of special colours of the T-APN net is given by bijection f, which has as domain the values of the set of identity colours. In the T-APN net of Figure 6.9, x stands for the set of identity colours and xy represents the set of special colours. The structuring set x consists of the elements 1 and 2 which represent either the IDs of the users or the tasks and the roles that are linked to specific users. The set of special colours xy contains the elements 0 and 3, with $f(1) = 0$ and $f(2) = 3$.

The additional arcs of the T-APN net of Figure 6.9 consist of all the pre arcs that replace the respective colour-sensitive inhibitor arcs of the associated APN net and the post arcs between the emptying transitions and the control places of the building blocks. In this example, there are no post arcs that are used for the 'internal' functioning of the net as the T-APN net is associated with an APN net that has no bidirectional steps.

Continuing with the arc inscriptions, it should be mentioned that all the new direct arcs and the replacement arcs takes xy as inscriptions and all the other arcs take x as inscription.

As with the guards of the constructed T-APN net of the ambient garage, we consider two different cases for the guards of constructed T-APN net of the ambient conference room, the guards of T_G and T_{NG} transitions respectively. Regarding the guards of the T_G transitions, the set of *binding pairs* for which the step transitions are enabled is represented in Snoopy [29] by the following function:

$$[(x = 1\&xy = 0) \mid (x = 2\&xy = 3)]$$

On the other hand, the guards of all the other transitions are always true as they belong to the T_{NG} transitions.

Finally, the constructed marking of the T-APN net of the ambient conference room is presented below:

$$M_0 = \{(ctrl\_participants, 0), (ctrl\_participants, 3), (participants, 1), (participants, 2), (users, 1), (users, 2), (f\_user\_credentials, 1), (s\_user\_credentials, 2)\}$$

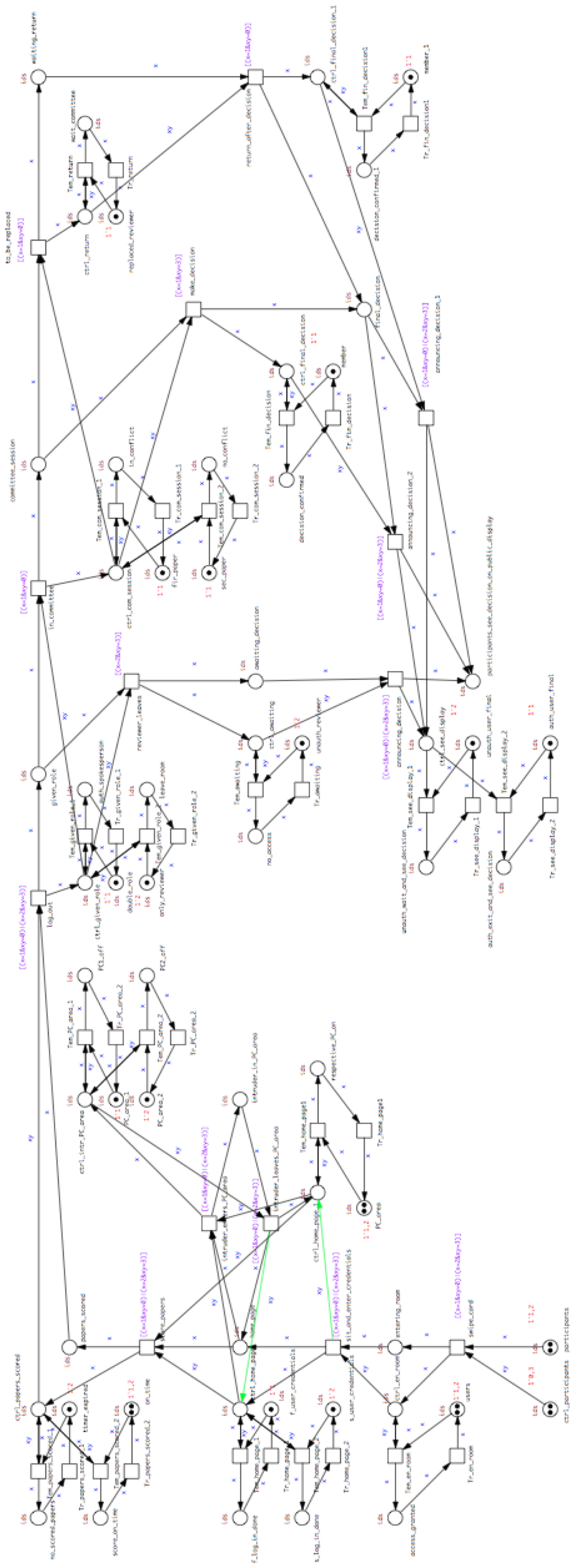


Figure 6.10: The T-APN net of ambient conference room in Snoopy.

where *ctrl_participants* is the only control place that carries special tokens at the initial marking M_0.

As in the case of the T-APN net of the ambient garage, the presence of the structuring set of special colour, additional arcs, arc inscriptions and guards in the T-APN net of the ambient conference room is necessary for the representation of the colour-sensitive inhibitor arcs that are used in the associated APN net in order to keep the functioning of the two nets the same.

Applying the construction for the given APN net of Figure 4.17, the constructed T-APN net of the ambient conference room is produced and is presented in Figure 6.10.

Contrary to the T-APN model of the ambient garage, that of the ambient conference room is not needed to have bounded net places. That derives from the fact that the places in this case do not represent physical spaces in the room, but different roles, tasks or credentials given to particular participants of the conference, as happens with the respective APN model of the conference room, out of which the T-APN model was constructed. For example, there could exist multiple tasks performed by many users at the same time allowing in that sense the presence of several coloured tokens into a net place without being necessary to bound the places using a specific upper bound for them. The only limit that is imposed on these places is that they can host only one token from each of the available colours given by the structuring sets Cl and $\overline{Cl}$. Thus, these places are limited by the cardinality of the union of these two structuring sets. Finally, it should be mentioned that in case that we want to bound the places of the net that represents the conference room for any reason, we could do it in exactly the same way as in section 6.5.1.

6.6 Behavioural Equivalence of APNs and T-APNs

Having described the construction that defines the relations among the elements of the two nets and having provided examples that demonstrate the applicability of the construction to general or particular nets, in this section we prove that the constructed T-APN nets are behaviourally equivalent to the original APN nets, by showing that the concurrent reachability graphs of these nets are isomorphic.

The following definition describes the isomorphism of the transition systems and, by extension, the isomorphism of the concurrent reachability graphs, as the CRGs are step transition systems according to definition 5.

Definition 22 (Transition System Isomorphism). *Let $STS = (Q, \Delta, q_0)$ and $STS' = (Q', \Delta', q_0')$ be two step transition systems over T.*

We say that STS is isomorphic to STS' if and only if there exists a bijection $\psi : Q \to Q'$ satisfying the following conditions:

(i) $\psi\ (q_0) = q'_0$ *and*

(ii) $\forall q_1, q_2 \in Q, \forall U \in 2^{\widetilde{T}} : (q_1, U, q_2) \in \Delta \iff (\psi(q_1), U, \psi(q_2)) \in \Delta'$.

Theorem 1. *Let* $\mathcal{N} = (P, T, Pre, Post, I, Cl, C, K, M_0, G)$ *be a composite step APN net and* $\varphi(\mathcal{N}) = (P', T', Pre', Post', Cl', \overline{Cl}, C_P, C_T, Ins, K', Gd, M'_0, G')$ *be a T-APN net associated with it. Then* $CRG(\mathcal{N})$ *is isomorphic to* $CRG(\varphi(\mathcal{N}))$.

Proof. The proof of the theorem can be found in the Appendix.

□

From the proof of the theorem, it follows that a constructed T-APN net and its associated APN net are behaviourally equivalent as their concurrent reachability graphs are isomorphic.

6.7 Concluding Remarks

In this chapter, we discussed the definition of a construction, which outlines how an APN net can be 'translated' into the respective T-APN net. Afterwards, we demonstrated, through an example, how this construction can be applied to given APN nets in order to produce the respective T-APN net. We also presented an example of the token game of a constructed T-APN net associated with a given APN net that showed empirically that the construction provides nets that are behaviourally equivalent. Furthermore, this chapter dealt with the constructed T-APN models of the case studies, which will be used for the verification. Finally, the chapter ended with the proof of the isomorphism of the concurrent reachability graphs of two nets that are related through a construction.

Consequently, having defined the construction for the nets of these two classes and having proved the behavioural equivalence of these nets through the isomorphism of their concurrent reachability graphs, it can be concluded that the translation of APN nets into T-APN nets can be conducted without any loss of nets' functionality. This fact enables the verification of the T-APN models of the case studies using Charlie [11] getting in that way verification results that are related the examined systems, which were initially modelled as APN models.

Chapter 7

Verification of Ambient Systems Using T-APNs

In the previous chapters, we explained how the modelling of ambient systems can be obtained by applying the step-modelling approach, which is based on the use of APNs. Due to the fact that this approach provides inhibitor APN models that cannot be verified by the majority of the existing verification tools, we described how these models can be translated, through a construction, into behaviourally equivalent non-inhibitor T-APN models that could be used for the verification of ambient systems.

Modelling ambient systems without analysing their models would be of little use, as we would not be able to detect, tackle and improve potentially erroneous states or conditions that could emerge during the operation of these systems. Therefore, within the domain of rigorous system design, verification of systems is an attractive and effective approach of checking the correctness of the examined models. Verification investigates whether the examined models of the systems are correct with respect to the specification.

In this chapter, we present the analysis of ambient systems by examining the correctness of the behaviour of the produced models through the use of the Charlie verification tool [11]. Charlie verification tool supports the analysis of low and high level Petri net models considering model checking, qualitative analysis and quantitative analysis for the verification of these models [11, 28]. For the examination of the correctness of the T-APN models of the chosen case studies, we focus only on the analysis of model checking and qualitative properties.

For the analysis of the T-APN models of the ambient systems, we firstly introduce model checking properties that are related to the basic components and features of the functionality of the ambient system. These properties are distinguished into two categories: the ambient properties and the general properties. All these properties are expressed in Computation Tree Logic (CTL) [16] and are examined using model checking [44].

Additionally to the model checking properties mentioned above, we perform qualitative analysis to the T-APN models considering two categories of properties, the structural properties and the behavioural properties, as they are presented in [11, 28]. Both qualitative categories of properties are 'automatically' checked with respect to the produced reachability graph and the initial marking of the inserted T-APN models. For the qualitative analysis of the T-APN models, we present only those properties that can provide useful conclusions about the interactive aspects and the general functioning of ambient systems.

Further verification analysis is conducted on the T-APN models by examining properties that are related to the examination of the produced reachability graphs of the models and the path analysis of those reachability graphs. Finally, we examine a few fault-tolerance properties that consider the faulty behaviour of the systems.

The examination of the T-APN models against all the above properties results in useful outcomes about the ambient systems' operation. These outcomes become apparent by interpreting the verification results of the properties into meaningful expressions related to the features and components of these systems. The T-APN models of the case studies that are used as input to Charlie verification tool for the analysis of the ambient systems, are shown in Figures 6.8 and 6.10. As has already been mentioned, these T-APN models have been developed using Snoopy modelling tool [29].

Through the verification process of the T-APN models, we try to examine several system properties and give answers to important general questions that are related to the functioning, the components and the interactive nature of the ambient systems. These questions are answered in depth through the examination of the different categories of properties mentioned above. A list of representative questions that are answered through the analysis of the properties of these models is as follows:

(i) Model checking analysis:

- Does the environment always detect the users?
- Do the users always have the initiative regardless of the system's advice?
- Can a user complete his tasks by ignoring the advices of the system?
- Is the information (advice) always available to the users?
- Can an important information leak to unauthorised users during the operation of the system?

- Does the environment always interact with the users and vice versa?
- Is the system context-aware?

(*ii*) Qualitative analysis:

- Is the capacity of the system finite?
- Does the system terminate?
- Does the system's operation involves any deadlock cases?

(*iii*) Path analysis:

- What is the cost for a user that ignores the system's advices compared to a user that always follows them?
- Can a user reach his goal without repeating at least one of his actions when he has already ignored one of the advices?

(*iv*) Fault-tolerance analysis:

- Can a user reach his goal after having taken a wrong advice from the system?
- Can the system provide the user with a correct advice after having given a wrong one to him for the same action?

7.1 Model Checking

The first categories of properties, i.e., Ambient properties and General properties, will be examined using model checking. These properties are expressed as CTL formulas and refer to key features of the ambient systems that are related to both the architecture and the functioning of theses systems. These CTL formulas can be distinguished into *state* formulas and *path* formulas [16]. The *state* formulas check whether an examined expression is true or false for a particular state of the *reachability graph* of a net. On the contrary, the *path* formulas check whether an expression is true or false along a particular path of the *reachability graph* of a net.

For better comprehension of the following properties, we recall the basic quantifiers and operators that are used for the expression of the CTL formulas [16]. These quantifiers and operators are as follows: **A** quantifier is used to examine if a proposition holds along all the computation paths, **E** quantifier is used to examine if a proposition holds for at least one of the computation paths, **G** operator is used to examine whether the proposition always holds globally, **F** operator is used to examine if the proposition holds at some point in the future, **X** operator is used to examine whether the proposition holds in the very next state and finally **U** operator is used to examine if a proposition holds until another proposition is valid. Combining the quantifiers and the operators of CTL, we get operators like **AG**, **EG**, **AF**, **EF**, **AX**, **EX**, etc., that can examine temporal properties of the examined models. The meaning of these operators derives from the above combination. For example, operator **EF** examines if there exist a path for which the proposition holds at some point in the future, operator **AG** examines if for all the computed path the examined proposition holds globally, etc. Similarly for the explanation of the remaining operators.

7.1.1 Ambient Properties

The category of Ambient properties examines the key features of the ambient systems that are considered in the T-APN models of the chosen case studies, as they were discussed in sections 2.1.1 and 2.2.5. More specifically, the main focus of the following paragraphs is the analysis of properties that are related to the behaviour of the sensors, the displays and the response of the system (advices) with respect to the restriction of the user, the leak of information and the user's behaviour. In the latter case, it is examined how the system responds to the user's initiative to either follow or ignore the given advices.

For the evaluation of the modelled features of the ambient systems, these properties are sorted into three subcategories, which concentrate on different individual aspects of this kind of systems. The first two categories of properties are the ambient guidance properties and the ambient information properties, which are based on the classification of the ambient systems into Ambient Guidance Systems and Ambient Information Systems respectively. The ambient guidance properties examine those properties that provide significant observations mainly for Ambient Guidance Systems, such as the ambient garage. Respectively, the ambient information properties examine those properties that result in significant observations mostly about Ambient Information Systems like the ambient conference room. It should be mentioned that both of these categories of properties can be applied to both case studies but the interpretation of the results will be of lower importance for the system that does not belong to the examined category each time. Contrary to the first two categories, the common ambient properties, examine those properties that provide useful observations about the examined features that are of the same importance in both categories of systems.

For better comprehension of the verification results, the properties are firstly expressed in English and then in Computation Tree Logic (CTL). Finally, the verification results are discussed interpreting the outcome of the examined properties with respect to what it should be expected from the specification.

7.1.1.1 Ambient Guidance Properties of T-APNs

Ambient guidance properties focus on the investigation of two particular features of the ambient systems, the sensors and the user's initiative. The behaviour of the sensors is examined through the analysis of the detection of the users by the system. On the contrary, the analysis of the user's initiative is carried out by examining whether the user can ignore the system's advice and whether ignoring the system's advice can affect the user's goal.

These properties refer mainly to the verification of the ambient garage, since sensors and user's initiative are prominent features of the Ambient Guidance Systems, which influence the functioning of the ambient garage, i.e., its response to the user's actions. On the other hand, the functioning of the ambient conference room, is mainly affected by the information provided and not by the presence of sensors or the initiative of the user. Although, users' initiative is limited and sensors are only used for the detection of either the unauthorised users or the authorised users when they first enter the room, this category of properties can also be used for the verification of the ambient conference room.

7.1.1.1.1 Examining Sensors

The following properties examine the detection of the user by the system and the functioning of the sensors for the ambient garage and the ambient conference room respectively.

The first property about the examination of the sensors is expressed as follows:

"A user is always detected by the system's sensors when he performs an action."

To examine that property, the above expression is translated into the respective CTL propositions for both ambient garage and ambient conference room, which are presented in Figure 7.1.

```
AG(entrance==2 -> AX(p1>0&ctrl_p1>0) &
   (p1==2 -> AX((p2>0&ctrl_p2>0) | (A1>0 & ctrl_A1>0))) &
   (A1==2 -> AX(p1>0&ctrl_p1_1>0)) &
   (p2==2 -> AX((A2>0&ctrl_A2>0) | (p1>0 & ctrl_p1_2>0) | (exit>0 & ctrl_exit>0 ))) &
   (A2==2 -> AX(p2>0&ctrl_p2_1>0)));
```

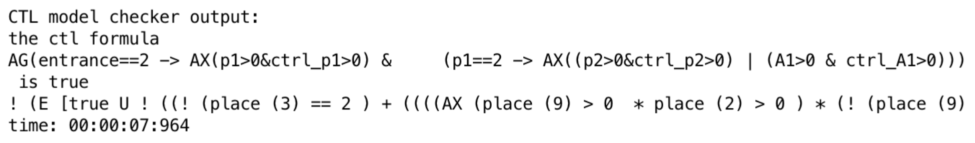
(a) CTL proposition of detection property for ambient garage

```
AG(participants==2 -> AX(entering_room>0&ctrl_en_room>0) &
   (entering_room==2 -> AX(home_page>0&ctrl_home_page>0)) &
   (intruder_in_PC_area==2 -> AX(home_page>0&ctrl_home_page_1>0)) &
   (home_page==2 -> EX(intruder_in_PC_area>0&ctrl_intr_PC_area>0))&
   (committee_session==2 -> EX(waiting_return>0&ctrl_return>0)));
```

(b) CTL proposition of detection property for ambient conference room

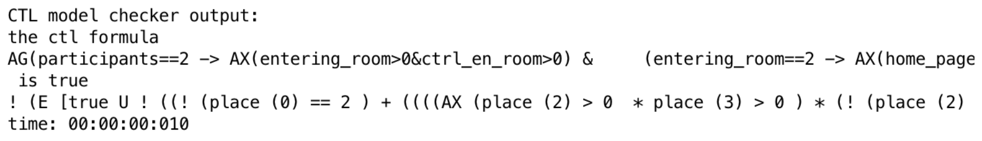
Figure 7.1: CTL propositions of user detection property for both T-APN nets.

These CTL propositions are used as input to Charlie in order to get the model checking result, which is true for both nets as is presented in Figure 7.2.



```
CTL model checker output:
the ctl formula
AG(entrance==2 -> AX(p1>0&ctrl_p1>0) &     (p1==2 -> AX((p2>0&ctrl_p2>0) | (A1>0 & ctrl_A1>0)))
 is true
! (E [true U ! ((! (place (3) == 2 ) + ((((AX (place (9) > 0  * place (2) > 0 ) * (! (place (9)
time: 00:00:07:964
```

(a)Detection property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(participants==2 -> AX(entering_room>0&ctrl_en_room>0) &     (entering_room==2 -> AX(home_page
 is true
! (E [true U ! ((! (place (0) == 2 ) + ((((AX (place (2) > 0  * place (3) > 0 ) * (! (place (2)
time: 00:00:00:010
```

(b) Detection property result for ambient conference room

Figure 7.2: Results of user detection property for both T-APN nets.

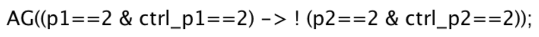
The result of this property being true for the ambient garage ensures that the system is always aware of the user's position in the garage, which means that it can provide the user with the correct advice.

Property being true for the ambient conference room denotes that the system can detect both the authorised and the unauthorised users giving the proper advice each time.

The second property regarding the sensors is related to the detection of the user by two non overlapping sensors. The property is expressed in English as follows:

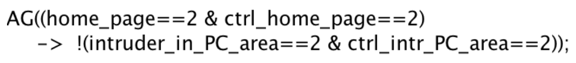
"A user cannot be detected by two different sensors at the same time."

The above expression is translated into the CTL propositions of Figure 7.3 for T-APN nets of the examined case studies.



```
AG((p1==2 & ctrl_p1==2) -> ! (p2==2 & ctrl_p2==2));
```

(a) CTL proposition of sensors overlapping property for ambient garage

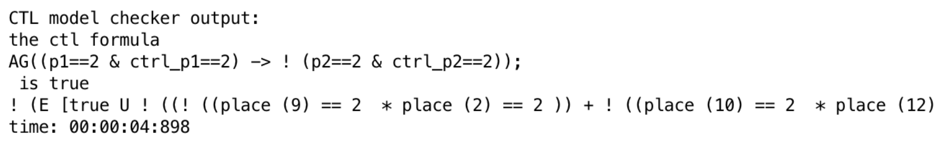


```
AG((home_page==2 & ctrl_home_page==2)
   ->  !(intruder_in_PC_area==2 & ctrl_intr_PC_area==2));
```

(b) CTL proposition of sensors overlapping property for ambient conference room

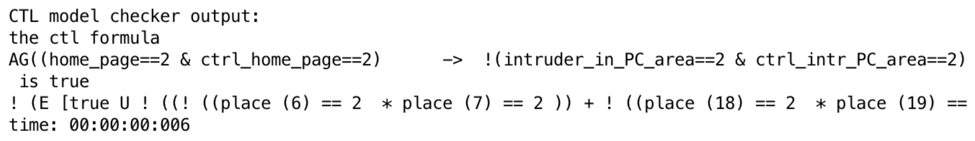
Figure 7.3: CTL proposition of sensors overlapping property for both T-APN nets.

The result of this property is true for both nets as is presented in Figure 7.4.



```
CTL model checker output:
the ctl formula
AG((p1==2 & ctrl_p1==2) -> ! (p2==2 & ctrl_p2==2));
 is true
! (E [true U ! ((! ((place (9) == 2  * place (2) == 2 )) + ! ((place (10) == 2  * place (12)
time: 00:00:04:898
```

(a) Sensors overlapping property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((home_page==2 & ctrl_home_page==2)      ->  !(intruder_in_PC_area==2 & ctrl_intr_PC_area==2)
 is true
! (E [true U ! ((! ((place (6) == 2  * place (7) == 2 )) + ! ((place (18) == 2  * place (19) ==
time: 00:00:00:006
```

(b) Sensors overlapping property result for ambient conference room

Figure 7.4: Results of sensors overlapping property for both T-APN nets.

The result of this property being true for the ambient garage ensures that a user cannot be at two different places at the same as the sensors cannot overlap, which means that the system provides the user with only one advice each time that he is detected by the environment.

Similarly for the ambient conference room, for which the property is also true. This denotes that an intruder cannot be at two different places and as a result only one of the reviewers cannot continue with his current task since the intruder is located in the ‘private’ area of his personal display.

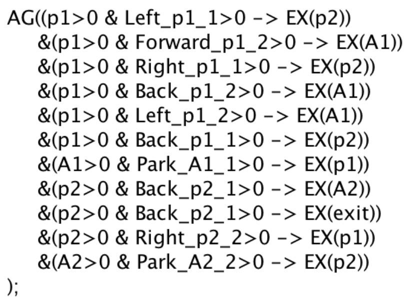
7.1.1.1.2 Examining User’s Initiative

The following properties examine the initiative of the users and how it affects the interaction between them and the environments of the ambient garage and the ambient conference room respectively.

The first property as regards the examination of the user’s initiative is expressed as follows:

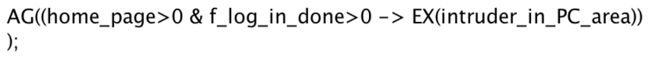
“A user can ignore the system’s advice.”

The CTL proposition for the above property is shown in Figure 7.5 for both the ambient garage and the ambient conference room.



```
AG((p1>0 & Left_p1_1>0 -> EX(p2))
    &(p1>0 & Forward_p1_2>0 -> EX(A1))
    &(p1>0 & Right_p1_1>0 -> EX(p2))
    &(p1>0 & Back_p1_2>0 -> EX(A1))
    &(p1>0 & Left_p1_2>0 -> EX(A1))
    &(p1>0 & Back_p1_1>0 -> EX(p2))
    &(A1>0 & Park_A1_1>0 -> EX(p1))
    &(p2>0 & Back_p2_1>0 -> EX(A2))
    &(p2>0 & Back_p2_1>0 -> EX(exit))
    &(p2>0 & Right_p2_2>0 -> EX(p1))
    &(A2>0 & Park_A2_2>0 -> EX(p2))
);
```

(a) CTL proposition of ignoring advice property for ambient garage

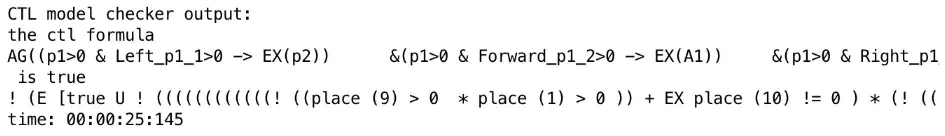


```
AG((home_page>0 & f_log_in_done>0 -> EX(intruder_in_PC_area))
);
```

(b) CTL proposition of ignoring advice property for ambient conference room

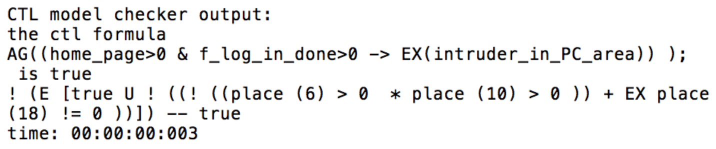
Figure 7.5: CTL proposition of the ignoring advice property for both T-APN nets.

Inserting those CTL propositions into the model checker of Charlie, we get true for both T-APN nets, as is presented in Figure 7.6.



```
CTL model checker output:
the ctl formula
AG((p1>0 & Left_p1_1>0 -> EX(p2))      &(p1>0 & Forward_p1_2>0 -> EX(A1))     &(p1>0 & Right_p1
 is true
! (E [true U ! ((((((((((((! ((place (9) > 0  * place (1) > 0 )) + EX place (10) != 0 ) * (! ((
time: 00:00:25:145
```

(a)Ignoring advice property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((home_page>0 & f_log_in_done>0 -> EX(intruder_in_PC_area)) );
 is true
! (E [true U ! ((! ((place (6) > 0  * place (10) > 0 )) + EX place
(18) != 0 ))]) -- true
time: 00:00:00:003
```

(b)Ignoring advice property result for ambient conference room

Figure 7.6: Results of the ignoring advice property for both T-APN nets.

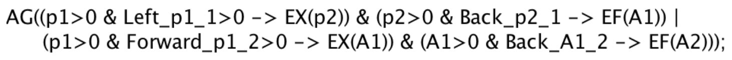
The first property related to the examination of user's initiative is true for the ambient garage indicating that regardless of the advice of the system, a user can head to a different direction from that indicated by the system's advice, which means that the behaviour of the model complies with the specification of the ambient garage.

On the other hand, this property does not provide a significant outcome about the model of the ambient conference room since only the unauthorised user can potentially ignore the system's advice. Property being true can, in some sense, be translated into the arbitrary move of the unauthorised user in the room ignoring the conference procedure and the system's warnings.

The last property that is related to the user's initiative examines the consequence of ignoring the system's advice with respect to the final goal of the user. This property is expressed as follows:

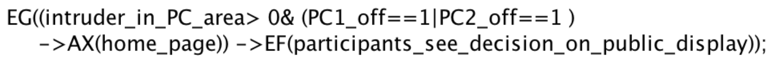
"A user partially ignoring the system's advices can eventually reach his goal."

The above expression is translated into the CTL propositions of Figure 7.7 for T-APN nets of the examined case studies.



```
AG((p1>0 & Left_p1_1>0 -> EX(p2)) & (p2>0 & Back_p2_1 -> EF(A1)) |
    (p1>0 & Forward_p1_2>0 -> EX(A1)) & (A1>0 & Back_A1_2 -> EF(A2)));
```

(a) CTL proposition of reaching goal after ignoring advice property for ambient garage

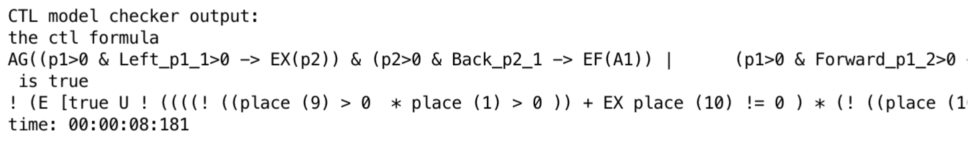


```
EG((intruder_in_PC_area> 0& (PC1_off==1|PC2_off==1 )
   ->AX(home_page)) ->EF(participants_see_decision_on_public_display));
```

(b) CTL proposition of reaching goal after ignoring advice property for ambient conference room

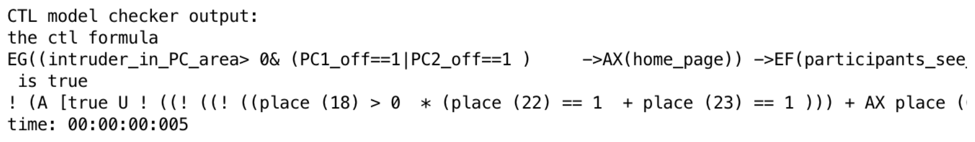
Figure 7.7: CTL propositions of reaching goal after ignoring advice property for both T-APN nets.

The result of this property is true for both nets as is presented in Figure 7.8.



```
CTL model checker output:
the ctl formula
AG((p1>0 & Left_p1_1>0 -> EX(p2)) & (p2>0 & Back_p2_1 -> EF(A1)) |      (p1>0 & Forward_p1_2>0
 is true
! (E [true U ! ((((! ((place (9) > 0  * place (1) > 0 )) + EX place (10) != 0 ) * (! ((place (1
time: 00:00:08:181
```

(a)Reaching goal after ignoring advice property result for ambient garage



```
CTL model checker output:
the ctl formula
EG((intruder_in_PC_area> 0& (PC1_off==1|PC2_off==1 )     ->AX(home_page)) ->EF(participants_see
 is true
! (A [true U ! ((! ((! ((place (18) > 0  * (place (22) == 1  + place (23) == 1 ))) + AX place (
time: 00:00:00:005
```

(b) Reaching goal after ignoring advice property result for ambient conference room

Figure 7.8: Results of reaching goal after ignoring advice property for both T-APN nets.

The result of this property being true for the ambient garage ensures that the user can reach his final goal, which is to park his car, despite the fact that he can ignore some of the system's advices following different directions from those indicated by the garage.

On the other hand, this property cannot be expressed precisely for the ambient conference room, since all the authorised users always follow the advices of the system. The only way to represent this property is by the presence of the unauthorised user in the room. Actually, the result of the property, which is true, reveals that eventually the reviewing procedure of the papers can be continued after the unauthorised user leaves the room resulting in the completion of the conference by releasing the final decision about the papers.

7.1.1.2 Ambient Information Properties of T-APNs

Ambient information properties are related to the investigation of the system components that provide the user with the proper advices (information), i.e., the displays. The behaviour of the displays is examined through the analysis of their activation mode. Furthermore, in this category, the security of the information is examined by showing that no information is leaked to unauthorised users of the system.

These properties refer mostly to the verification of the ambient conference room, since it is considered that only in that system there exist unauthorised users, who can take advantage of vital information. This happens due to the fact that the displays of the ambient garage are public and rarely private. Moreover, leak of information in the ambient garage does not exist, but even if we consider that there is leak of information (advice) from a private display, this incident would be of minor importance as this advice is also presented on the public displays and is visible to all the users of the garage. Finally, examining the activation modes of the display is also not so important for the ambient garage since its public displays are always "on" while the system operates, but it is supposed that they should be "off" when the system terminates or an advice is not available for a user.

Although, displays and leak of information do not significantly affect the functioning of the garage, ambient information properties can be used for verification purposes of the T-APN model of the ambient garage.

7.1.1.2.1 Examining Displays

The following property examines the activation mode of the displays of the system for the ambient garage and the ambient conference room respectively. The property that examines the displays is expressed as follows:

"System's displays are either 'on' or 'off' at any point throughout the operation of the system."

The CTL proposition of the above property is presented in Figure 7.9 for the ambient garage and the ambient conference room respectively.

```
AG((EF(Left_p1_1 == 0) | EF(Left_p1_1 != 0)) &
(EF(Forward_p1_2 == 0) | EF(Forward_p1_2 != 0)) &
(EF(Left_p1_2 == 0) | EF(Left_p1_2 != 0)) &
(EF(Back_p1_1 == 0) | EF(Back_p1_1 != 0)) &
(EF(Park_A1_1 == 0) | EF(Park_A1_1 != 0)) &
(EF(Back_A1_2 == 0) | EF(Back_A1_2 != 0)) &
(EF(Right_p2_2 == 0) | EF(Right_p2_2 != 0)) &
(EF(Back_p2_1 == 0) | EF(Back_p2_1 != 0)) &
(EF(Back_A2_1 == 0) | EF(Back_A2_1 != 0)) &
(EF(Park_A2_2 == 0) | EF(Park_A2_2 != 0)) &
(EF(Left_p2_1 == 0) | EF(Left_p2_1 != 0)) &
(EF(Back_p2_2 == 0) | EF(Back_p2_2 != 0)) &
(EF(Bye_exit_12 == 0) | EF(Bye_exit_12 != 0)));
```

(a) CTL proposition of display mode property for ambient garage

```
AG((EF(access_granted == 0) | EF(access_granted != 0)) &
(EF(f_log_in_done == 0) | EF(f_log_in_done != 0)) &
(EF(s_log_in_done == 0) | EF(s_log_in_done != 0)) &
(EF(PC1_off == 0) | EF(PC1_off != 0)) &
(EF(PC2_off == 0) | EF(PC2_off != 0)) &
(EF(respective_PC_on == 0) | EF(respective_PC_on != 0)) &
(EF(score_on_time == 0) | EF(score_on_time != 0)) &
(EF(no_scored_papers == 0) | EF(no_scored_papers != 0)) &
(EF(in_conflict == 0) | EF(in_conflict != 0)) &
(EF(no_conflict == 0) | EF(no_conflict != 0)) &
(EF(participants_see_decision_on_public_display == 0)
| EF(participants_see_decision_on_public_display != 0)));
```

(b) CTL proposition of display mode property for ambient conference room

Figure 7.9: CTL propositions of display mode property for both T-APN nets.

The result of the model checker for this property is true for both nets, as is shown in Figure 7.10.

```
CTL model checker output:
the ctl formula
AG((EF(Left_p1_1 == 0) | EF(Left_p1_1 != 0)) &(EF(Forward_p1_2 == 0) | EF(Forward_p1_2 != 0)) &
 is true
! (E [true U ! (((((((((((((E [true U place (1) == 0 ] + E [true U place (1) != 0 ]) * (E [tru
time: 00:00:31:998
```

(a)Display mode property result for ambient garage

```
CTL model checker output:
the ctl formula
AG((EF(access_granted == 0) | EF(access_granted != 0)) &(EF(f_log_in_done == 0) | EF(f_log_in_d
 is true
! (E [true U ! (((((((((((E [true U place (4) == 0 ] + E [true U place (4) != 0 ]) * (E [true
time: 00:00:00:016
```

(b) Display mode property result for ambient conference room

Figure 7.10: Results of display mode property for both T-APN nets.

The result of this property is true for the ambient garage, which ensures that the all the displays used in the garage are “on” or “off” at any point of the system’s operation. This means that the advices can appear (or be removed) on (or from) the displays prompting the user to act accordingly.

Property being true for the ambient conference room denotes that if the system detects the presence of unauthorised users, it consequently hides all the private information from them by switch off the displays when is necessary and switching them back on when the unauthorised users leave the restricted area.

7.1.1.2.2 Examining Leak of Information

The following property examines whether information can be leaked to unauthorised users at any stage of the system’s operation.
The property that examines the leak of information is expressed as follows:

“There is no leak of information to unauthorised users.”

The CTL proposition of the above property for the ambient garage and the ambient conference room is presented in Figure 7.11(a) and (b) respectively.

```
AG((db_p1_1 == 0 -> !(Left_p1_1>1)) &
    (db_p1_2 == 0 -> !(Forward_p1_2>1)) &
    (db_p1_1_2 == 0 -> !(Left_p1_2>1)) &
    (db_p1_1_1 == 0 -> !(Back_p1_1>1)) &
    (db_A1_1 == 0 -> !(Park_A1_1>1)) &
    (db_A1_2 == 0 -> !(Back_A1_2>1)) &
    (db_p1_2_2 == 0 -> !(Back_p1_2>1)) &
    (db_p1_2_1 == 0 -> !(Right_p1_1>1)) &
    (db_p2_2 == 0 -> !(Right_p2_2>1)) &
    (db_p2_1 == 0 -> !(Back_p2_1>1)) &
    (db_A2_2 == 0 -> !(Park_A2_2>1)) &
    (db_A2_1 == 0 -> !(Back_A2_1>1)) &
    (db_p2_1_1 == 0 -> !(Left_p2_1>1)) &
    (db_p2_1_1 == 0 -> !(Back_p2_2>1)) &
    (db_exit_12 == 0 -> !(Bye_exit_12>2)));
```

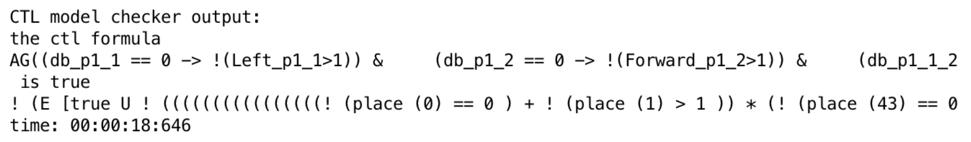
(a) CTL proposition of leak of information property for ambient garage

```
AG((intruder_in_PC_area > 1 -> EX(PC1_off | PC2_off)) &
   (users == 0 -> !(access_granted>2)) &
   (auth_spokesperson == 1 -> EX(committee_session)) &
   (only_reviewer == 0 -> (leave_room ==1))&
   (unauth_reviewer == 0 -> (no_access ==1))&
   (replaced_reviewer ==0 -> (wait_committee))
   );
```

(b) CTL proposition of leak of information property for ambient conference room

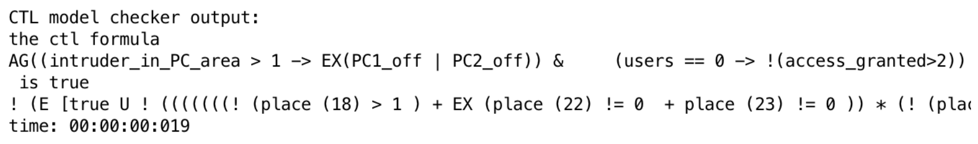
Figure 7.11: CTL proposition of leak of information property for both T-APN nets.

The result of the model checker for this property is true for both nets, as is presented in Figure 7.12.



```
CTL model checker output:
the ctl formula
AG((db_p1_1 == 0 -> !(Left_p1_1>1)) &     (db_p1_2 == 0 -> !(Forward_p1_2>1)) &     (db_p1_1_2
 is true
! (E [true U ! ((((((((((((((((! (place (0) == 0 ) + ! (place (1) > 1 )) * (! (place (43) == 0
time: 00:00:18:646
```

(a)Leak of information property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((intruder_in_PC_area > 1 -> EX(PC1_off | PC2_off)) &     (users == 0 -> !(access_granted>2))
 is true
! (E [true U ! (((((((! (place (18) > 1 ) + EX (place (22) != 0  + place (23) != 0 )) * (! (pla
time: 00:00:00:019
```

(b) Leak of information property result for ambient conference room

Figure 7.12: Results of leak of information property for both T-APN nets.

It can be noticed from Figure 7.12(a) that the property turns true for the ambient garage, which ensures that the information that is provided to the user is not leaked to unauthorised users. This means that if the user gets the information to his private display (i.e PDA, mobile phone, etc.) or the public display, it can be seen only by him or by all the other cars that use the garage. This happens due to the fact the number of users is limited and the system allows only the authorised users to gain access to the garage.

Property being true for the ambient conference room denotes that no vital information is leaked when an intruder appears in a private display area or when the a reviewer is not illegible to participate in the next phase of the conference process or when a user is in conflict with a paper and must exit the room in order to allow the session to continue. Finally, it ensures that access is permitted to a predetermined number of users, who can either hold authorised or unauthorised roles.

7.1.1.3 Common Ambient Properties of T-APNs

Common ambient properties deal with all the actions that can be made in the ambient environments and the way that they affect the interactivity of the systems. More specifically, they examine how the system behaves in accordance with a user's actions and how that user is restricted by the system when is necessary in order to be provided with the appropriate information for each of his actions. Finally, it is examined how the three different kinds of actions, i.e. user's actions, system's actions and internal actions, are related and why.

System's interactivity and user's restriction are prevailing features of the ambient systems and are met in both the Ambient Guidance Systems and the Ambient Information Systems, which means that the properties of this category are very important for the verification of both T-APN nets. Hence, to investigate these features, four different properties that are related to them are described below.

7.1.1.3.1 Notifying the User

The following property questions whether every single action of the user is followed by the respective system's notification. Through this property, we examine the interactivity of the system. For the examination of the user's notification, the following expression is used:

"A user is notified after every single action that he makes."

The CTL proposition of the above property is presented for the ambient garage and the ambient conference room in Figure 7.13(a) and (b) respectively.

```
AG(((p1&ctrl_p1 -> EX(p1&Left_p1_1)) & (p1&Left_p1_1 -> EX(p2&ctrl_p2)))&
    ((p1&ctrl_p1 -> EX(p1&Forward_p1_2)) & (p1&Forward_p1_2 -> EX(p2&ctrl_p2)))&
    ((A1&ctrl_A1 -> EX(A1&Back_A1_2)) & (A1&Back_A1_2 -> EX(p1&ctrl_p1_1)))&
    ((p1&ctrl_p1_1 -> EX(p1&Back_p1_1)) & (p1&Back_p1_1 -> EX(A1&ctrl_A1)))&
    ((p1&ctrl_p1_1 -> EX(p1&Left_p1_2)) & (p1&Left_p1_2 -> EX(p2&ctrl_p2)))&
    ((p2&ctrl_p2 -> EX(p2&Back_p2_1))&(p2&Back_p2_1 -> EX(p1&ctrl_p1_2)))&
    ((p2&ctrl_p2 -> EX(p2&Right_p2_2))&(p2&Right_p2_2 -> EX(A2&ctrl_A2)))&
    ((A2&ctrl_A2 -> EX(A2&Back_A2_1))&(A2&Back_A2_1 -> EX(p2&ctrl_p2_1)))&
    ((p2&ctrl_p2_1 -> EX(p2&Back_p2_1))&(p2&Back_p2_1 -> EX(exit&ctrl_exit))));
```

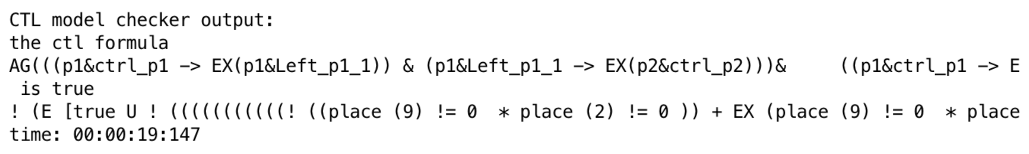
(a) CTL proposition of user's notification property for ambient garage

```
AG(((entering_room & ctrl_en_room -> EX(entering_room & access_granted)) &
  (entering_room & access_granted -> EX(home_page & ctrl_home_page))) &
((home_page & ctrl_home_page & ctrl_home_page_1==0 -> EX(home_page & f_log_in_done)) &
  (home_page & f_log_in_done -> EX(papers_scored & ctrl_papers_scored)))&
((home_page & ctrl_home_page & ctrl_home_page_1==0 -> EX(home_page & s_log_in_done)) &
  (home_page & f_log_in_done -> EX(papers_scored & ctrl_papers_scored)))&
((home_page & ctrl_home_page==0 & ctrl_home_page_1 -> EX(home_page & f_log_in_done)) &
  (home_page & f_log_in_done -> EX(intruder_in_PC_area & ctrl_intr_PC_area)))&
((home_page & ctrl_home_page==0 & ctrl_home_page_1 -> EX(home_page & s_log_in_done)) &
  (home_page & f_log_in_done -> EX(intruder_in_PC_area & ctrl_intr_PC_area)))&
((intruder_in_PC_area ==2 & ctrl_intr_PC_area ==2 & PC1_off ==0 & PC2_off ==0-> EX(intruder_i
  (intruder_in_PC_area ==2 & PC1_off -> EX(home_page & ctrl_home_page_1)))&
((intruder_in_PC_area ==2 & ctrl_intr_PC_area ==2 & PC1_off ==0 & PC2_off ==0-> EX(intruder_i
  (intruder_in_PC_area ==2 & PC2_off -> EX(home_page & ctrl_home_page_1)))&
((home_page ==2 & ctrl_home_page_1 ==2 & respective_PC_on==0 -> EX(home_page & respectiv
  (home_page & respective_PC_on -> EX(papers_scored & ctrl_papers_scored)))&
((committee_session & ctrl_com_session  &  fir_paper>0 & sec_paper >0 -> EX(committee_sessior
  (committee_session & in_conflict -> EX(waiting_return & ctrl_return)))&
((committee_session & ctrl_com_session  &  fir_paper>0 & sec_paper >0 -> EX(committee_sessior
  (committee_session & no_conflict -> EX(final_decision & ctrl_final_decision)))&
((waiting_return & ctrl_return  &  wait_committee==0  -> EX(waiting_return & wait_committee)) &
  (waiting_return & wait_committee -> EX(final_decision & ctrl_final_decision_1)))&
((final_decision & ctrl_final_decision  &  member>0 -> EX(final_decision & decision_confirmed)) &
  (final_decision & decision_confirmed -> EX(participants_see_decision_on_public_display & ctrl_s
((final_decision & ctrl_final_decision_1  &  member_1>0 -> EX(final_decision & decision_confirme
  (final_decision & decision_confirmed_1 -> EX(participants_see_decision_on_public_display & ctrl
((participants_see_decision_on_public_display & ctrl_see_display  &  unauth_user_final>0 &  auth_
  -> EX(participants_see_decision_on_public_display & unauth_wait_and_see_decision)) &
  (participants_see_decision_on_public_display & unauth_wait_and_see_decision
  -> EX(participants_see_decision_on_public_display & unauth_user_final)))&
((participants_see_decision_on_public_display & ctrl_see_display  &  unauth_user_final>0 & auth_
   -> EX(participants_see_decision_on_public_display & auth_exit_and_see_decision)) &
  (participants_see_decision_on_public_display & auth_exit_and_see_decision
   -> EX(participants_see_decision_on_public_display & auth_user_final))));
```

(b) CTL proposition of user's notification property for ambient conference room

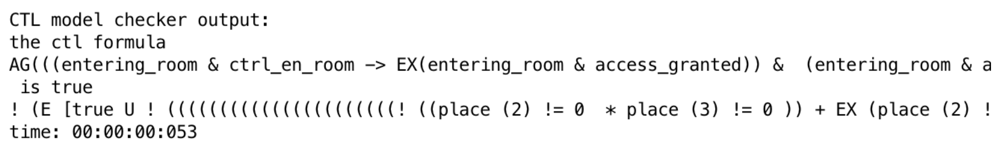
Figure 7.13: CTL propositions of user's notification property for both T-APN nets.

The model checking results of this property for both nets are presented in Figure 7.14.



```
CTL model checker output:
the ctl formula
AG(((p1&ctrl_p1 -> EX(p1&Left_p1_1)) & (p1&Left_p1_1 -> EX(p2&ctrl_p2)))&     ((p1&ctrl_p1 -> E
 is true
! (E [true U ! ((((((((((! ((place (9) != 0  * place (2) != 0 )) + EX (place (9) != 0  * place
time: 00:00:19:147
```

(a) User's notification property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(((entering_room & ctrl_en_room -> EX(entering_room & access_granted)) &  (entering_room & a
 is true
! (E [true U ! ((((((((((((((((((((! ((place (2) != 0  * place (3) != 0 )) + EX (place (2) !
time: 00:00:00:053
```

(b) User's notification property result for ambient conference room

Figure 7.14: Results of user's notification property for both T-APN nets.

The property turns true for both the ambient garage and the ambient conference room, which ensures that an advice (information) is provided to the user after every single action that he makes. After getting the advice, the user can move to his next action receiving again another advice for that. For example, if a car moves from a place to another in the garage, it is always informed about the direction that it should follow at the new place in order to reach to the allocated parking bay.

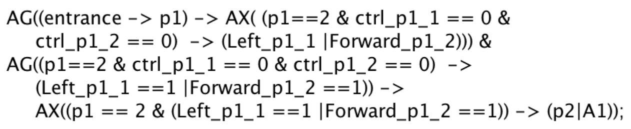
The result of this property also proves that since the user gets an advice after each action, the system interacts with him. This interaction between the system and the user occurs in every single step of the T-APN nets.

7.1.1.3.2 Alternation of Actions

Alternation of actions property also examines the system's interactivity. This takes place here by checking if there exists an alteration between the user's and the system's actions. For the examination of the alteration of the actions, the property is expressed as follows:

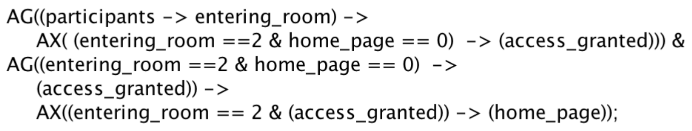
"A system's advice is preceded by a user's action and vice versa."

The CTL propositions of the above property for the T-APN nets of the ambient garage and the ambient conference room are shown in Figure 7.15.



```
AG((entrance -> p1) -> AX( (p1==2 & ctrl_p1_1 == 0 &
    ctrl_p1_2 == 0)  -> (Left_p1_1 |Forward_p1_2))) &
AG((p1==2 & ctrl_p1_1 == 0 & ctrl_p1_2 == 0)  ->
    (Left_p1_1 ==1 |Forward_p1_2 ==1)) ->
    AX((p1 == 2 & (Left_p1_1 ==1 |Forward_p1_2 ==1)) -> (p2|A1));
```

(a) CTL proposition of alteration of actions property for ambient garage

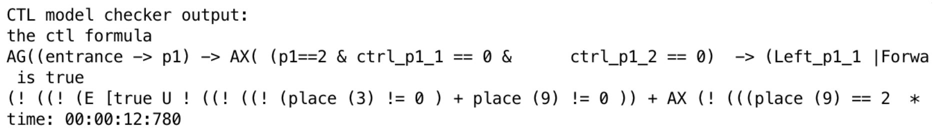


```
AG((participants -> entering_room) ->
    AX( (entering_room ==2 & home_page == 0)  -> (access_granted))) &
AG((entering_room ==2 & home_page == 0)  ->
    (access_granted)) ->
    AX((entering_room == 2 & (access_granted)) -> (home_page));
```

(b) CTL proposition of alteration of actions property for ambient conference room

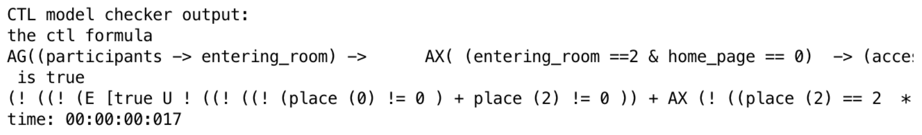
Figure 7.15: CTL propositions of alteration of actions property for both T-APN nets.

These CTL proposition are used as input to the model checker of Charlie in order to get the result for this property. Figure 7.16 presents the output of the examined property for the T-APN models of the ambient garage and the ambient conference room.



```
CTL model checker output:
the ctl formula
AG((entrance -> p1) -> AX( (p1==2 & ctrl_p1_1 == 0 &      ctrl_p1_2 == 0)  -> (Left_p1_1 |Forwa
 is true
(! ((! (E [true U ! ((! ((! (place (3) != 0 ) + place (9) != 0 )) + AX (! (((place (9) == 2  *
time: 00:00:12:780
```

(a) Alternation of actions property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((participants -> entering_room) ->      AX( (entering_room ==2 & home_page == 0)  -> (acce
 is true
(! ((! (E [true U ! ((! ((! (place (0) != 0 ) + place (2) != 0 )) + AX (! ((place (2) == 2  *
time: 00:00:00:017
```

(b) Alternation of actions property result for ambient conference room

Figure 7.16: Results of alternation of actions property for both T-APN nets.

The property being true for both the ambient garage and the ambient conference room, ensures that there is an alternation between the user's actions and those of the system. This means that a user performs an action for which he takes an advice (system's action) and this advice is followed by another action of the user and so on. This property proves once again that the developed T-APN nets comply with both the definition of the ambient systems and the specification of the case studies, since they interact with the users giving advices.

Moreover, the presence of alternation of actions between the users and the environment in the models of the examined ambient systems also reveals that the T-APN nets, in some sense, preserve the functioning of the colour-sensitive inhibitor arcs of the associated APN nets, which have been used in the APN nets of the case studies in order to represent the interaction of the users with the environment in every single action of the first. Finally, the preservation of the functioning of colour-sensitive inhibitor arcs in the T-APN models will be examined and discussed further in the next section.

7.1.1.3.3 Restricting User's Actions

The property of restricting user's actions checks whether the system always provides the user with an advice immediately after every single action that he performs. Furthermore, it ensures that the user cannot proceed to his next action before taking an advice from the system for his current action.

For the examination of the 'restriction' of the user by the system, the property of restricting user's actions is expressed as follows:

"A user cannot perform two consecutive actions without the 'intervention' of the system."

The CTL propositions of the above property that are used for the examination of T-APN models are presented in Figure 7.17.

```
AG((entrance -> p1) -> AX( (p1>0 & ctrl_p1_1==0 & ctrl_p1_2==0)  -> (Left_p1_1 |Forward_p1_2)));

AG((entrance -> p1) -> AX( p1 -> p2));
```

(a) CTL proposition of restricting user's actions property for ambient garage

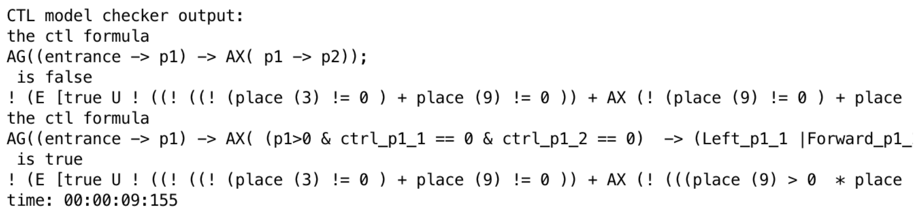
```
AG((participants-> entering_room) -> AX( entering_room>1  -> access_granted));

AG((participants -> entering_room) -> AX( entering_room -> home_page));
```

(b) CTL proposition of restricting user's actions property for ambient conference room

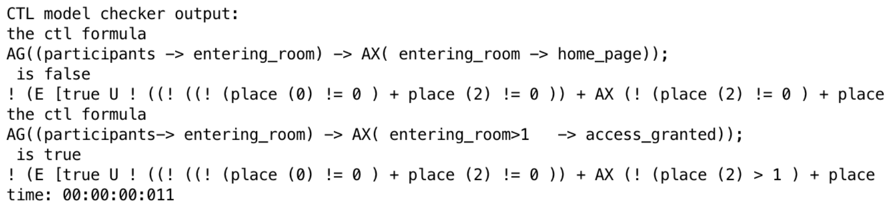
Figure 7.17: CTL propositions of restricting user's actions property for both T-APN nets.

To check the outcome of this property for both T-APN nets, two CTL propositions are used. The first proposition states that a user makes an action and the system always responds to that action with an advice. The second one denotes that a user can perform an action and immediately after the first action he can perform another action consecutively. It is noted that these two propositions contradict each other. This occurs in order to examine whether the environment supports both situations or not. In case that the results are both true or false respectively, then the examined ambient systems malfunction as the users do not interact with the environment implying that either they behave uncontrollably or the system is not in operation. The results of the two CTL propositions of this property are presented in Figure 7.18 for both T-APN models.



```
CTL model checker output:
the ctl formula
AG((entrance -> p1) -> AX( p1 -> p2));
 is false
! (E [true U ! ((! ((! (place (3) != 0 ) + place (9) != 0 )) + AX (! (place (9) != 0 ) + place
the ctl formula
AG((entrance -> p1) -> AX( (p1>0 & ctrl_p1_1 == 0 & ctrl_p1_2 == 0)  -> (Left_p1_1 |Forward_p1_
 is true
! (E [true U ! ((! ((! (place (3) != 0 ) + place (9) != 0 )) + AX (! (((place (9) > 0  * place
time: 00:00:09:155
```

(a) Restricting user's actions property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((participants -> entering_room) -> AX( entering_room -> home_page));
 is false
! (E [true U ! ((! ((! (place (0) != 0 ) + place (2) != 0 )) + AX (! (place (2) != 0 ) + place
the ctl formula
AG((participants-> entering_room) -> AX( entering_room>1   -> access_granted));
 is true
! (E [true U ! ((! ((! (place (0) != 0 ) + place (2) != 0 )) + AX (! (place (2) > 1 ) + place
time: 00:00:00:011
```

(b) Restricting user's actions property result for ambient conference room

Figure 7.18: Results of alternation of actions property for both T-APN nets.

Following the order in which the two CTL propositions of the property are presented in Figure 7.17, their results are true and false respectively, for both the ambient garage and the ambient conference room, indicating firstly that a user's action is always followed by the system's reaction (advice) and secondly that a user cannot execute two consecutive actions.

For the ambient garage, this means that a user cannot park to the allocated parking bay without getting the proper direction (advice) for every single move that he makes. For example, when the user is in place $p1$, he gets the advice go forward or go left depending on where his allocated parking bay is. After that he is able to move to place $p2$. Additionally, the user cannot move from the entrance to the place $p1$ and then immediately to the place $p2$ because he must be informed about the direction that he has to follow.

Similarly for the ambient conference room, where the results of the property denote that a user gets an advice for every task (or action) that he starts or finishes and that a user cannot complete two tasks (or actions) without being advised by the system. For instance, if a participant swipes his card in order to get into the room, then he is informed that the access has been granted. On the other hand, a participant cannot swipe his card to get into the room and start working on the reviewing process of the papers without being allowed to enter the room.

This property ensures that both T-APN nets are modelled in such a way that restrict the user from making his next action before getting an advice from the system. This restriction is applied to the nets for modelling purposes in order to represent that an ambient system works always correctly by providing the proper information to the user immediately after his action. This restriction is consequence of the presence of colour-sensitive inhibitor arcs in the APN models and is passed to the T-APN models through the additional arcs and the presence of guards.

7.1.1.3.4 Examining System's Actions

This property checks the kinds of different actions that may take place within an ambient system during a single interaction between the environment and the users providing useful information about the way that the interaction process is carried out and the sequence of the actions within it. As has been mentioned, these kinds of different actions refer to user's actions, system's actions and actions that are used for the internal functioning of the system.
For the examination of these different actions, the property is expressed as follows:

"Every single user-system interaction consists of user's action, system's response and database refill."

The CTL proposition of the above property for both T-APN models is presented in Figure 7.19.

```
AG((entrance -> AX(p1)) -> EX(Left_p1_1 | Forward_p1_2) -> EX(db_p1_1 | db_p1_2)
& ((p1 -> AX(A1|p2)) -> (EX(Park_A1_1 | Back_A1_2) | EX(Back_p2_1|Right_p2_2))
   -> (EX(db_A1_1 | db_A1_2) | EX(db_p2_1 | db_p2_2)))
& ((p2 -> AX(A2|p1|exit)) -> (EX(Park_A2_2 | Back_A2_1) | EX(Back_p1_2|Right_p1_1) | EX(Bye_exit_12))
   -> (EX(db_A2_2 | db_A2_1) | EX(db_p1_2_2 | db_p1_2_1) | EX(db_exit_12)))
& ((A1 -> AX(p1)) -> EX(Left_p1_2 | Back_p1_1) -> EX(db_p1_1_2 | db_p1_1_1))
& ((A2 -> AX(p2)) -> EX(Left_p2_1 | Back_p2_2) -> EX(db_p2_1_1 | db_p2_1_2)));
```

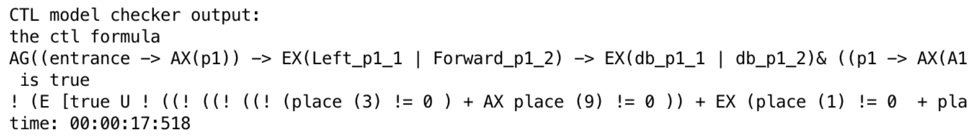
(a) CTL proposition of system's actions property for ambient garage

```
AG((participants -> AF(entering_room)) -> EF(access_granted) -> EF(users))
& ((entering_room -> AF(home_page)) -> (EF(f_log_in_done | s_log_in_done)
   -> EF(f_user_credentials|s_user_credentials))
& ((home_page -> AF(intruder_in_PC_area|papers_scored)) -> (EF(PC1_off | PC2_off) |
  EF(no_scored_papers|score_on_time))  -> (EF(PC_area_1 | PC_area_2) | EF(timer_expired | on_time)))
& ((intruder_in_PC_area -> AF(home_page)) -> EF(respective_PC_on) -> EF(PC_area))
& ((papers_scored -> AF(given_role)) -> EF(auth_spokesperson | leave_room) -> EF(double_role | only_reviewer))
& ((given_role -> AF(committee_session|awaiting_decision)) -> (EF(in_conflict | no_conflict) | EF(no_access))
   -> (EF(fir_paper | sec_paper) | EF(unauth_reviewer)))
& ((committee_session -> AF(final_decision|waiting_return)) -> (EF(decision_confirmed) | EF(wait_committee))
   -> (EF(member) | EF(replaced_reviewer)))
& ((waiting_return -> AF(final_decision)) -> (EF(decision_confirmed_1))
   -> (EF(member_1)))
& ((final_decision -> AF(participants_see_decision_on_public_display)) ->
  (EF(unauth_wait_and_see_decision)|EF(auth_exit_and_see_decision))  -> (EF(unauth_user_final)|EF(auth_user_final))));
```

(b) CTL proposition of system's actions property for ambient conference room

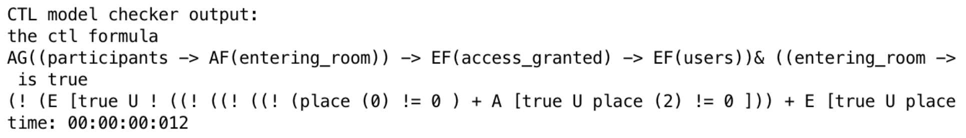
Figure 7.19: CTL propositions of system's actions property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie. The outcome of Charlie model checker for this property is true for both nets as is shown in Figure 7.20.



```
CTL model checker output:
the ctl formula
AG((entrance -> AX(p1)) -> EX(Left_p1_1 | Forward_p1_2) -> EX(db_p1_1 | db_p1_2)& ((p1 -> AX(A1
 is true
! (E [true U ! ((! ((! ((! (place (3) != 0 ) + AX place (9) != 0 )) + EX (place (1) != 0  + pla
time: 00:00:17:518
```

(a) System's actions property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((participants -> AF(entering_room)) -> EF(access_granted) -> EF(users))& ((entering_room ->
 is true
(! (E [true U ! ((! ((! ((! (place (0) != 0 ) + A [true U place (2) != 0 ])) + E [true U place
time: 00:00:00:012
```

(b) System's actions property result for ambient conference room

Figure 7.20: Results of system's actions property for both T-APN nets.

The property being true for both the ambient garage and the ambient conference room, ensures that there is a pattern of actions that is followed when an interaction between a user and the system occurs. This pattern consists of the user's action, which is made first, then the system's response always follows and eventually an action that refills the system's database is carried out, which is characterised as an 'internal' action of the system. This pattern is very strict when the property is examined for T-APN models that consist of only one building block. But, this pattern of actions becomes weaker when it comes to large T-APN nets with several building and tokens (resources). This happens due to the fact that the sequence of enabled transitions changes as regards the 'internal' functioning of the system. The pattern of actions between the user and the system remains always the same, but in large composite T-APN nets after every system's response two actions are enabled, the 'internal' action that refills the database of the system and the next action of the user. Hence, the pattern of actions can differ if the next transition that will fire is that of the user's action. Otherwise, the pattern remains exactly the same in every step. Knowing that there always exists a weak or strict pattern, this makes the examination of the T-APN nets more consistent since all the enabled transitions after the firing of either a user's action, a system's action or an 'internal' action can be predicted.

This means that a user's action always precedes an advice (system's action), an advice always precedes an 'internal' action or a user's new action before another sequence of actions starts. This property also denotes that the interaction between the user and the system is not affected by the internal functioning of the developed T-APN nets for ambient garage and ambient conference room. This happens due to the fact that the user and the system can interact with each other without using internal actions. Internal actions are needed only if one or more states of the T-APN nets are revisited, which means that the database of the system must be refilled with the proper information (resources) about the user's action.

7.1.2 General Properties

In this section, we present the results of the last category of properties that investigates the general functioning of the ambient systems, i.e., the general properties. More specifically, the general properties deal with the analysis of features that are related to the general functioning of the ambient systems such as the completion of user's final goal, the termination of system's operation, the number of available resources, the renewal of database resources, etc.

Contrary to the ambient properties, the general properties are organised into only two subcategories, the general guidance properties and the common general properties. The categorisation of the general properties is conducted using the same criteria as those used for the categorisation of the ambient properties in 7.1.1, but in this case there is no category of general information properties. This occurs because none of the examined functioning features of the ambient systems plays more important role in the Ambient Information Systems than it does in the Ambient Guidance Systems.

The description of the general properties is made in the same way as that of the description of the ambient properties, starting with the expression of the properties in English, then continuing with the properties expressed in Computation Tree Logic (CTL) and finally ending with the interpretation of the results into meaningful outcomes about the examined T-APN models of the chosen case studies.

7.1.2.1 General Guidance Properties of T-APNs

General guidance properties focus on the investigation of two particular features of the ambient systems' functioning, the advices that can be provided by the system throughout its operation and the possibility of returning to a previous state of the system. The first property examines whether the system can provide the user with all the possible advices at some point. In other words, it checks the redundancy of the system's advices. On the other hand, the second property examines whether the user can revisit a state of the system.

These properties apply mostly to the verification of the Ambient Guidance Systems, as their functioning depends both on the advices given to the users with respect to their actions and on the users' ability to redo some actions during the operation of these systems. For example, if at least one of the possible advices of the ambient garage cannot be given to the user, then this could misdirect him resulting in the non completion of his goal. Furthermore, the ability of redoing actions is of high importance for the functioning of the ambient garage as the user should be able to go back to a previously visited place of the garage when the system redirects him after a wrong action. On the other hand, these properties are not so important for the Ambient Information Systems and especially for the ambient conference room, since the users always follow the procedure and the system always provides the proper advices to them. Furthermore, the users of the system are not allowed to redo actions. The ambient conference room permits the return to a previous state, only when an unauthorised user (intruder) enters and then leaves the room representing in that way that the system can go back to a 'normal' condition where the conference process can be continued in order to conclude to a final decision about the reviewed papers.

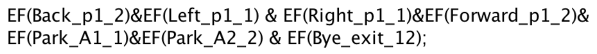
Although, it is known from the modelling of the ambient conference room that it always provides the users with the correct advices and that the user cannot redo certain actions, the general guidance properties can be used to confirm that indeed the system can potentially give all the possible advices to the user and that an unauthorised user eventually leaves the room and the process continues from the point that was 'paused'.

7.1.2.1.1 Examining Redundancy of Advices

This property checks if there exists any advice in the set of all possible advices of the system that is never provided to the user. To examine the redundancy of the system's advices, the property is expressed as follows:

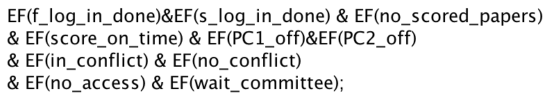
"None of the advices from the set of possible advices is redundant."

The CTL proposition of the above property is presented for both T-APN nets in Figure 7.21.



```
EF(Back_p1_2)&EF(Left_p1_1) & EF(Right_p1_1)&EF(Forward_p1_2)&
EF(Park_A1_1)&EF(Park_A2_2) & EF(Bye_exit_12);
```

(a) CTL proposition of advices redundancy property for ambient garage

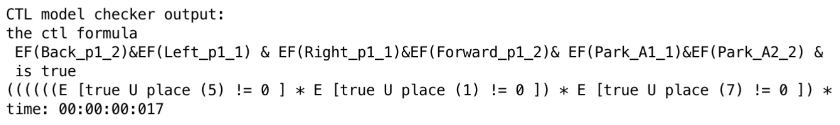


```
EF(f_log_in_done)&EF(s_log_in_done) & EF(no_scored_papers)
& EF(score_on_time) & EF(PC1_off)&EF(PC2_off)
& EF(in_conflict) & EF(no_conflict)
& EF(no_access) & EF(wait_committee);
```

(b) CTL proposition of advices redundancy property for ambient conference room

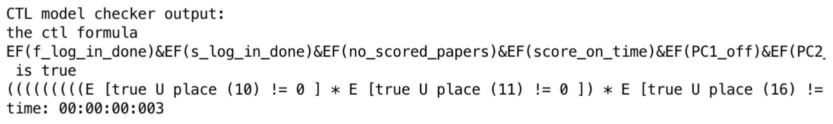
Figure 7.21: CTL propositions of redundancy of advices property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie model checker. The results for both T-APN models are presented in Figure 7.22.



```
CTL model checker output:
the ctl formula
 EF(Back_p1_2)&EF(Left_p1_1) & EF(Right_p1_1)&EF(Forward_p1_2)& EF(Park_A1_1)&EF(Park_A2_2) &
 is true
((((((E [true U place (5) != 0 ] * E [true U place (1) != 0 ]) * E [true U place (7) != 0 ]) *
time: 00:00:00:017
```

(a) Redundancy of advices property result for ambient garage



```
CTL model checker output:
the ctl formula
EF(f_log_in_done)&EF(s_log_in_done)&EF(no_scored_papers)&EF(score_on_time)&EF(PC1_off)&EF(PC2_
 is true
((((((((E [true U place (10) != 0 ] * E [true U place (11) != 0 ]) * E [true U place (16) !=
time: 00:00:00:003
```

(b) Redundancy of advices property result for ambient conference room

Figure 7.22: Results of redundancy of advices property for both T-APN nets.

The property being true for the ambient garage, means that each of the advices might be the right advice for some user at some point of the execution. For example, during the operation of the ambient garage, the advice go left or right might be the proper one for a user with respect to his current position.

On the other hand, the property being true for the ambient conference room confirms that at some point of the system's execution all the tasks or phases of the conference are completed or about to be completed. The completion of the tasks or phases is confirmed by providing the right advices to the users. For example, there exists a state where an advice is given to the user notifying him about a conflict with a paper or another state where an advice is given to notify the user about the papers that have not been reviewed.

Hence, the results of this property show that none of the advices used for the operation of each of the two systems is redundant.

7.1.2.1.2 Returning to Previous States

This property checks whether a state that has already been visited once by a user during the operation of an ambient system, can be revisited at some point in the future. For the examination of the user's return to previous states, the property is expressed as follows:

'The user can return to one of the previously visited states."

The CTL proposition of the above property for the examined T-APN nets is presented in Figure 7.23.

```
AG((p1 -> EX(A1))&(A1 ->EX(p1)))&
AG((p1 -> EX(p2))&(p2 ->EX(p1)))&
AG((p2 -> EX(A2))&(A2 ->EX(p2)));
```

(a) CTL proposition of returning to previous states property for ambient garage

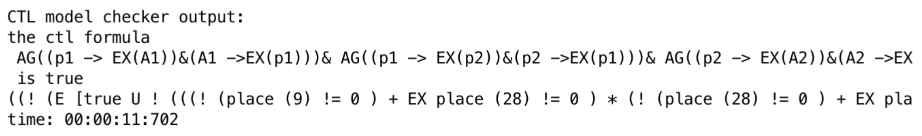
```
AG((home_page -> EX(intruder_in_PC_area))
&(intruder_in_PC_area ->EX(home_page)));
```

(b) CTL proposition of returning to previous states property for ambient conference room

Figure 7.23: CTL propositions of returning to previous states property for both T-APN nets.

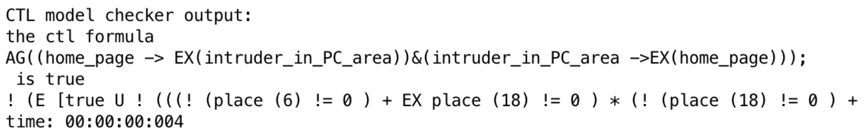
To check this property, the above CTL propositions are examined by Charlie model checker. The results of the property for both T-APN nets are shown in Figure 7.24.

The property being true for the ambient garage means that the users can redo some of their actions throughout the operation of the system. For example, a user can go from place $p1$ to the parking bay $A1$ and go back to $p1$ either because he parked his car to $A1$ and now he wants to leave the garage or because he went to the wrong parking bay ($A1$) and now he wants to correct his mistake by moving back to $p1$ in order to head towards the correct parking bay ($A2$).



```
CTL model checker output:
the ctl formula
 AG((p1 -> EX(A1))&(A1 ->EX(p1)))& AG((p1 -> EX(p2))&(p2 ->EX(p1)))& AG((p2 -> EX(A2))&(A2 ->EX
 is true
((! (E [true U ! (((! (place (9) != 0 ) + EX place (28) != 0 ) * (! (place (28) != 0 ) + EX pla
time: 00:00:11:702
```

(a) Returning to previous states property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((home_page -> EX(intruder_in_PC_area))&(intruder_in_PC_area ->EX(home_page)));
 is true
! (E [true U ! (((! (place (6) != 0 ) + EX place (18) != 0 ) * (! (place (18) != 0 ) +
time: 00:00:00:004
```

(b) Returning to previous states property result for ambient conference room

Figure 7.24: Results of returning to previous states property for both T-APN nets.

On the other hand, the property being true for the ambient conference room implies that the system can return to the state where it was before an unauthorised user enters to a restricted area. For instance, an unauthorised user (intruder) enters the restricted are of the personal display of a reviewer while he is at the home page forcing the system to switch off that display and stop the reviewing process. Then, after the intruder has left the restricted area or the room, the display returns to its previous state, which is the home page allowing the reviewer to continue the process.

The results of this property indicate that some states of the systems can be 're-visited' during the operation of the ambient garage and the ambient conference room respectively.

7.1.2.2 Common General Properties of T-APNs

Common general properties examine those components and features of the ambient systems that are linked to the number of the available resources, the advices provided in relation to the database of the systems, the termination of the systems, the users' goal, the database functioning, etc. All these components and features dominate the functioning of the ambient systems as they define all the essential information needed in order to enable the systems' working.

7.1.2.2.1 Relating Advice With Database Resources

This property checks whether there exists an advice that is not determined with respect to the information that is kept in the database of the examined system. For the investigation of the relation between the advices and the database resources, the property is expressed as follows:

"Advices are 'announced' on the display with respect to what the database indicates about every user."

The CTL propositions of the above property for the examined T-APN models are presented in Figure 7.25.

```
AG(Left_p1_2 ->A(Left_p1_2!=0 U db_p1_1_2==0))&
AG(Back_p1_1 ->A(Back_p1_1!=0 U db_p1_1_1==0))&
AG(Forward_p1_2 ->A(Forward_p1_2!=0 U db_p1_2==0))&
AG(Left_p1_1 ->A(Left_p1_1!=0 U db_p1_1==0))&
AG(Park_A1_1 ->A(Park_A1_1!=0 U db_A1_1==0))&
AG(Back_A1_2 ->A(Back_A1_2!=0 U db_A1_2==0))&
AG(Back_p1_2 ->A(Back_p1_2!=0 U db_p1_2_2==0))&
AG(Right_p1_1 ->A(Right_p1_1!=0 U db_p1_2_1==0))&
AG(Right_p2_2 ->A(Right_p2_2!=0 U db_p2_2==0))&
AG(Back_p2_1 ->A(Back_p2_1!=0 U db_p2_1==0))&
AG(Back_A2_1 ->A(Back_A2_1!=0 U db_A2_1==0))&
AG(Park_A2_2 ->A(Park_A2_2!=0 U db_A2_2==0))&
AG(Back_p2_2 ->A(Back_p2_2!=0 U db_p2_1_2==0))&
AG(Left_p2_1 ->A(Left_p2_1!=0 U db_p2_1_1==0))&
AG(Bye_exit_12 ->A(Bye_exit_12!=0 U db_exit_12<2));
```

(a) CTL proposition of relating advices with database resources property for ambient garage

```
AG(access_granted ->A(access_granted!=0 U users<2))&
AG(f_log_in_done ->A(f_log_in_done!=0 U f_user_credentials==0))&
AG(s_log_in_done ->A(s_log_in_done!=0 U s_user_credentials==0))&
AG(PC1_off ->A(PC1_off!=0 U PC_area_1==0))&
AG(PC2_off ->A(PC2_off!=0 U PC_area_2==0))&
AG(respective_PC_on ->A(respective_PC_on!=0 U PC_area<2))&
AG(auth_spokesperson ->A(auth_spokesperson!=0 U double_role==0))&
AG(leave_room ->A(leave_room!=0 U only_reviewer==0))&
AG(no_access ->A(no_access!=0 U unauth_reviewer==0))&
AG(in_conflict ->A(in_conflict!=0 U fir_paper==0))&
AG(no_conflict ->A(no_conflict!=0 U sec_paper==0))&
AG(wait_committee ->A(wait_committee!=0 U replaced_reviewer==0))&
AG(decision_confirmed ->A(decision_confirmed!=0 U member==0))&
AG(decision_confirmed_1 ->A(decision_confirmed_1!=0 U member_1==0));
```

(b) CTL proposition of relating advices with database resources property for ambient conference room

Figure 7.25: CTL propositions of system's actions property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie model checker. The results of this property are shown in Figure 7.26.

```
CTL model checker output:
the ctl formula
AG(Left_p1_2 ->A(Left_p1_2!=0 U db_p1_1_2==0))&AG(Back_p1_1 ->A(Back_p1_1!=0 U db_p1_1_1==0))
 is true
((((((((((((((! (E [true U ! ((! (place (37) != 0 ) + A [place (37) != 0  U place (38) == 0 ]
time: 00:00:50:488
```

(a) Relating advices with database resources property result for ambient garage

```
CTL model checker output:
the ctl formula
AG(access_granted ->A(access_granted!=0 U users<2))&AG(f_log_in_done ->A(f_log_in_done!=0 U f_u
 is true
(((((((((((((! (E [true U ! ((! (place (4) != 0 ) + A [place (4) != 0  U place (5) < 2 ]))]) *
time: 00:00:00:017
```

(b) Relating advices with database resources property result for ambient conference room

Figure 7.26: Results of relating advices with database resources property for both T-APN nets.

The property being true for both the ambient garage and the ambient conference room, means that all the advices of both systems are determined by the resources kept in the database of the system. For example, during the operation of the ambient garage, every advice can be displayed as long as the respective information about the users is provided by the database of the system. This is depicted in the result of Figure 7.26(a) with an empty database place, which implies that the resources that were available in the database place for a specific action of a user have been consumed enabling the display of the advice on the output devices. Similarly for the ambient conference room, where the property being true verifies that all the advices are both determined with respect to the information about the users that is held into the database of the system and displayed as long as the database permits that.

It should be mentioned that the results of this property indicate that the advices of an ambient system are determined and displayed according to the information of the database of that particular system and according to the action taken by a user making the system context-aware.

7.1.2.2.2 Refilling the Database Resources

This property examines if the information of the database is kept to the system for ever without being removed or changed. This enables the system to provide the user with the same advice (information) whenever he executes the same action in the future. For the examination of this property, the following expression is used:

"The database of the system is 'refilled' after every given advice."

The CTL proposition of the above property for both T-APN models is presented in Figure 7.27.

```
AG((Left_p1_1 -> EX (db_p1_1))&(Forward_p1_2 -> EX (db_p1_2))&
   (Back_p1_1 -> EX (db_p1_1_1))&(Left_p1_2 -> EX (db_p1_1_2))&
   (Park_A1_1 -> EX (db_A1_1))&(Back_A1_2 -> EX (db_A1_2))&
   (Back_p2_1 -> EX (db_p2_1))&(Right_p2_2 -> EX (db_p2_2))&
   (Right_p1_1 -> EX (db_p1_2_1))&(Back_p1_2 -> EX (db_p1_2_2))&
   (Left_p2_1 -> EX (db_p2_1_1))&(Back_p2_2 -> EX (db_p2_1_2))&
   (Back_A2_1 -> EX (db_A2_1))&(Park_A2_2 -> EX (db_A2_2))&
   (Bye_exit_12 -> EX (db_exit_12)));
```

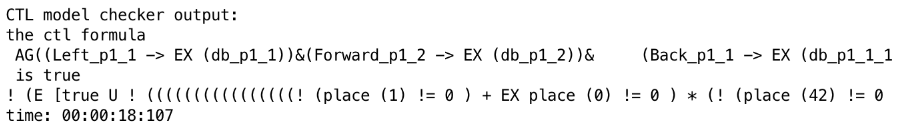
(a) CTL proposition of refilling the database resources property for ambient garage

```
AG((access_granted -> (EX (users)))
  &(f_log_in_done -> (EX (f_user_credentials)))
  &(s_log_in_done -> (EX (s_user_credentials)))
  &(no_scored_papers -> (EX (timer_expired)))
  &(score_on_time -> (EX (on_time)))
  &(respective_PC_on -> (EX (PC_area)))
  &(PC1_off -> (EX (PC_area_1)))
  &(PC2_off -> (EX (PC_area_2)))
  &(auth_spokesperson -> (EX (double_role)))
  &(leave_room -> (EX (only_reviewer)))
  &(no_access -> (EX (unauth_reviewer)))
  &(in_conflict -> (EX (fir_paper)))
  &(no_conflict -> (EX (sec_paper)))
  &(decision_confirmed -> (EX (member)))
  &(decision_confirmed_1 -> (EX (member_1)))
  &(wait_committee -> (EX (replaced_reviewer)))
  &(unauth_wait_and_see_decision -> (EX (unauth_user_final)))
  &(auth_exit_and_see_decision -> (EX (auth_user_final))));
```

(b) CTL proposition of refilling the database resources property for ambient conference room

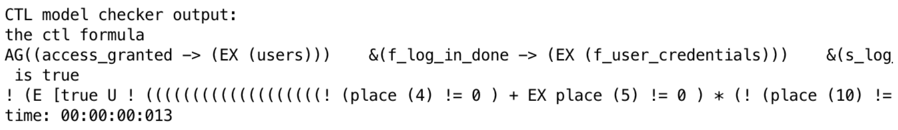
Figure 7.27: CTL propositions of refilling the database resources property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie model checker providing the results of that property for both T-APN models (see Figure 7.28).



```
CTL model checker output:
the ctl formula
 AG((Left_p1_1 -> EX (db_p1_1))&(Forward_p1_2 -> EX (db_p1_2))&    (Back_p1_1 -> EX (db_p1_1_1
 is true
! (E [true U ! (((((((((((((((! (place (1) != 0 ) + EX place (0) != 0 ) * (! (place (42) != 0
time: 00:00:18:107
```

(a) Refilling the database resources property result for ambient garage



```
CTL model checker output:
the ctl formula
AG((access_granted -> (EX (users)))    &(f_log_in_done -> (EX (f_user_credentials)))    &(s_log
 is true
! (E [true U ! ((((((((((((((((((! (place (4) != 0 ) + EX place (5) != 0 ) * (! (place (10) !=
time: 00:00:00:013
```

(b) Refilling the database resources property result for ambient conference room

Figure 7.28: Results of refilling the database resources property for both T-APN nets.

The property being true for both the ambient garage and the ambient conference room, means that after every advice of these systems the database resources kept in the database places of the nets are 'renewed' by executing the internal actions of the systems. For instance, in the ambient garage, after the advice go forward, the respective database place (*db_p*1_2) is refilled with the appropriate resources, which is represented by the presence of tokens in that particular place. Another example of this property, for the ambient conference room, is when the first reviewer has completed the log-in process successfully. After the notification of the user about the successful log-in, the database resources that are related the user's credentials (*f_user_credentials*) are refilled.

7.1.2.2.3 Examining the Quantity of the Resources

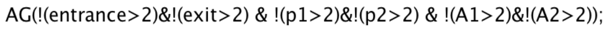
The following two properties deal with the quantity of the system resources by examining the minimum and the maximum number of resources that are allowed throughout the system operation.

The first property examines the maximum number of resources that are permitted during the execution of the examined systems.

For the verification of this property, the following expression is used:

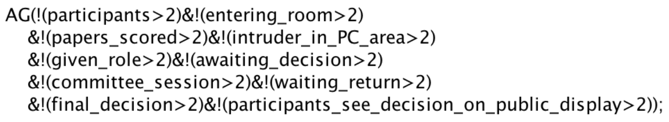
"The number of the resources or agents of the system cannot exceed a certain predefined constant number."

The CTL proposition of the above property for the examined T-APN nets is presented in Figure 7.29.



```
AG(!(entrance>2)&!(exit>2) & !(p1>2)&!(p2>2) & !(A1>2)&!(A2>2));
```

(a) CTL proposition of upper bound of system resources property for ambient garage

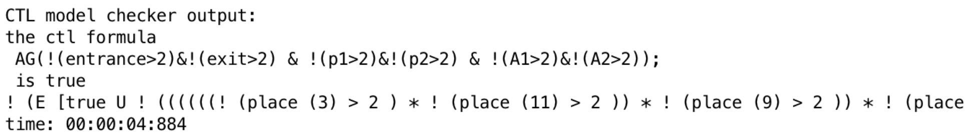


```
AG(!(participants>2)&!(entering_room>2)
   &!(papers_scored>2)&!(intruder_in_PC_area>2)
   &!(given_role>2)&!(awaiting_decision>2)
   &!(committee_session>2)&!(waiting_return>2)
   &!(final_decision>2)&!(participants_see_decision_on_public_display>2));
```

(b) CTL proposition of upper bound of system resources property for ambient conference room

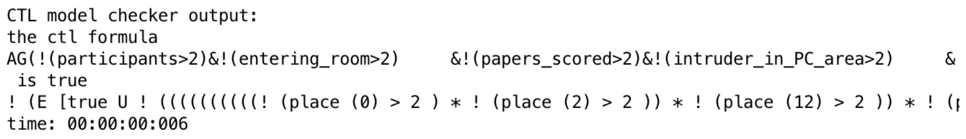
Figure 7.29: CTL propositions of upper bound of system resources property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie model checker resulting in the outcomes shown in Figure 7.30.



```
CTL model checker output:
the ctl formula
 AG(!(entrance>2)&!(exit>2) & !(p1>2)&!(p2>2) & !(A1>2)&!(A2>2));
 is true
! (E [true U ! ((((((! (place (3) > 2 ) * ! (place (11) > 2 )) * ! (place (9) > 2 )) * ! (place
time: 00:00:04:884
```

(a) Upper bound of system resources property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(!(participants>2)&!(entering_room>2)     &!(papers_scored>2)&!(intruder_in_PC_area>2)     &
 is true
! (E [true U ! (((((((((! (place (0) > 2 ) * ! (place (2) > 2 )) * ! (place (12) > 2 )) * ! (
time: 00:00:00:006
```

(b) Upper bound of system resources property result for ambient conference room

Figure 7.30: Results of upper bound of system resources property for both T-APN nets.

The property being true for the ambient garage, means that the number of the users that move around the garage cannot exceed that of two. This happens due to the fact the examined T-APN model of the ambient garage uses two tokens to represent the different users that interact with the system.

The property is also true for the ambient conference room, which indicates that the number of the users, tasks, papers or roles that are used for the representation of the system's functioning cannot exceed that of two. This results from the fact the initial marking of the examined T-APN model of the ambient conference room consists of two tokens to depict the users, the tasks, etc.

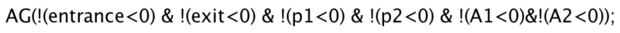
Thus, from what is mentioned above, it can be concluded that the maximum number is equal to two and is determined by the initial marking of the system and the structural boundedness of the examined T-APN nets. Furthermore, that number cannot be increased arbitrarily leading to infinite resources for the system. This CTL property confirms the result of the *k-boundedness* that was presented in section 7.2.2.1.

Contrary to the above property, the following one examines the lower number of resources that are permitted during the execution of the examined systems.

For the verification of this property, the following expression is used:

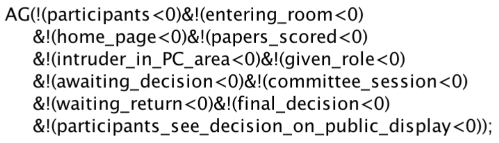
"The number of the resources or agents of the system cannot be depleted throughout the user's actions."

The CTL propositions of the above property for the T-APN models of the ambient garage and the ambient conference room are presented in Figure 7.31.



```
AG(!(entrance<0) & !(exit<0) & !(p1<0) & !(p2<0) & !(A1<0)&!(A2<0));
```

(a) CTL proposition of lower bound of system resources property for ambient garage

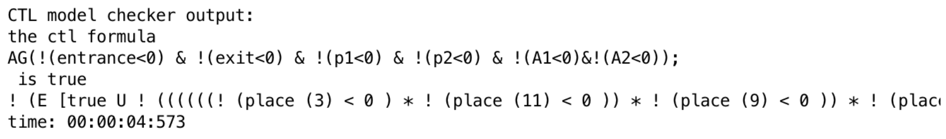


```
AG(!(participants<0)&!(entering_room<0)
    &!(home_page<0)&!(papers_scored<0)
    &!(intruder_in_PC_area<0)&!(given_role<0)
    &!(awaiting_decision<0)&!(committee_session<0)
    &!(waiting_return<0)&!(final_decision<0)
    &!(participants_see_decision_on_public_display<0));
```

(b) CTL proposition of lower bound of system resources property for ambient conference room

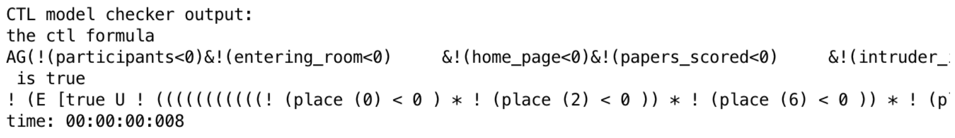
Figure 7.31: CTL propositions of lower bound of system resources property for both T-APN nets.

To check this property, the above CTL proposition are used as input to Charlie model checker. The outcome of the model checking for this property is true for both nets, as is shown in Figure 7.30.



```
CTL model checker output:
the ctl formula
AG(!(entrance<0) & !(exit<0) & !(p1<0) & !(p2<0) & !(A1<0)&!(A2<0));
 is true
! (E [true U ! ((((((! (place (3) < 0 ) * ! (place (11) < 0 )) * ! (place (9) < 0 )) * ! (plac
time: 00:00:04:573
```

(a) Lower bound of system resources property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(!(participants<0)&!(entering_room<0)     &!(home_page<0)&!(papers_scored<0)     &!(intruder_
 is true
! (E [true U ! (((((((((((! (place (0) < 0 ) * ! (place (2) < 0 )) * ! (place (6) < 0 )) * ! (p
time: 00:00:00:008
```

(b) Lower bound of system resources property result for ambient conference room

Figure 7.32: Results of lower bound of system resources property for both T-APN nets.

The property being true for the ambient garage, means that the number of the users that move around the garage cannot be reduced and suddenly be depleted. This happens due to the fact the examined T-APN net of the ambient garage needs a constant number of users to interact with the system in order to work correctly and according to the specification.

The property is also true for the ambient conference room, which indicates that the number of the users, tasks, papers or roles cannot be depleted. Otherwise, there exist no users to review the papers or paper to be reviewed, etc. This results from the fact the initial marking of the T-APN net of the ambient conference room consists of two tokens to depict the users, the tasks, etc.

Hence, from what is mentioned above, it can be concluded that the number of resources or agents is greater than zero for both nets and it is equal to zero only when the resources have been removed from particular places of the net but never from the whole net. Thus, the resources cannot be depleted, which also results from the fact that the property of no output place is false (see 7.2.1.4), which means that there exist no place in both nets that has a post-transition that can consume all the tokens (resources) of the two nets.

7.1.2.2.4 Reaching User's Goal

The following property checks whether the users of the examined systems can potentially achieve their goals.

For the verification of this property, the following expression is used:

"Every user eventually reaches his goal."

The CTL propositions of the above property for both T-APN models are shown in Figure 7.33.

```
EF(A1&Park_A1_1)&EF(A2&Park_A2_2);
```

(a) CTL proposition of reaching goal property for ambient garage

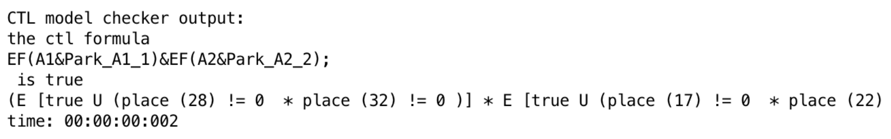
```
EF(final_decision&decision_confirmed)&
EF(final_decision&decision_confirmed_1)&
EF(participants_see_decision_on_public_display&
(unauth_wait_and_see_decision|auth_exit_and_see_decision));
```

(b) CTL proposition of reaching goal property for ambient conference room

Figure 7.33: CTL propositions of reaching goal property for both T-APN nets.

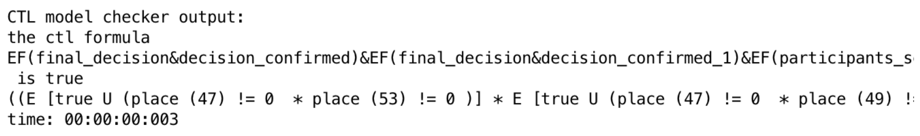
To examine this property, the above CTL propositions are used in Charlie model checker providing the outcome of that property for both models, as is shown in Figure 7.34.

The property being true for the ambient garage, means that every user eventually reaches his goal, which is to park his car to the allocated parking bay. The property is also true for the ambient conference room, which denotes that every user according to his role can either participate in the decision about the reviewed papers or see the final decision at the end of the conference. For example, a user that is member of the committee participates in the final decision if he is not in conflict with the paper. Otherwise, the decision is made by the other members of the committee since he has to exit the room. Finally, a participant who has no eligibility to join the committee for the final decision, he can see that decision when it is released.



```
CTL model checker output:
the ctl formula
EF(A1&Park_A1_1)&EF(A2&Park_A2_2);
 is true
(E [true U (place (28) != 0  * place (32) != 0 )] * E [true U (place (17) != 0  * place (22)
time: 00:00:00:002
```

(a) Reaching goal property result for ambient garage



```
CTL model checker output:
the ctl formula
EF(final_decision&decision_confirmed)&EF(final_decision&decision_confirmed_1)&EF(participants_s
 is true
((E [true U (place (47) != 0  * place (53) != 0 )] * E [true U (place (47) != 0  * place (49) !
time: 00:00:00:003
```

(b) Reaching goal property result for ambient conference room

Figure 7.34: Results of reaching goal property for both T-APN nets.

The result of this property ensures that all the users can reach their goals at some point regardless of whether they are looping for long time. The only way for a user to not reach his goal is by looping for ever, which means that the user does not want to cooperate with the system and that his only intention is to sabotage the system's or the other users' actions. This case is not considered in the modelling of the T-APN models of the ambient garage and the ambient conference room, as it assumed that the users can ignore advices but eventually follow them.

7.1.2.2.5 Reaching Terminal State

The following property checks whether the users can reach the terminal state of the system through at least one sequence of actions (i.e., paths). The terminal state of a system indicates the case where the environment stops to interact with the users after having assisted them to achieve their goals.

For the verification of this property, the following expression is used:

"There always exists a way to get to the terminal state, where the system eventually stops interacting with the users."

The CTL propositions of the above property for both T-APN models are presented in Figure 7.35.

```
AG(EF(exit==2&db_p1_1
   &db_p1_2_2&db_p1_2_1
   &db_p2_1&db_p2_2
   &db_A2_1&db_A2_2
   &db_p2_1_1&db_p2_1_2
   &db_A1_2&db_A1_1
   &db_p1_1_1&db_p1_1_2
   &db_p1_2&db_exit_12==2));
```

(a) CTL proposition of reaching terminal state property for ambient garage

```
AG(EF(participants_see_decision_on_public_display==2
    & users==2 & f_user_credentials & s_user_credentials
    &timer_expired & on_time == 2
    &PC_area == 2 & PC_area_1 & PC_area_2
    &double_role & only_reviewer
    &unauth_reviewer & fir_paper
    &sec_paper & replaced_reviewer
    &member & member_1
    &unauth_user_final & auth_user_final));
```

(b) CTL proposition of reaching terminal state property for ambient conference room

Figure 7.35: CTL propositions of reaching terminal state property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie model checker resulting the outcome of that property for both nets, as is shown in Figure 7.36.

```
CTL model checker output:
the ctl formula
AG(EF(exit==2&db_p1_1    &db_p1_2_2&db_p1_2_1    &db_p2_1&db_p2_2    &db_A2_1&db_A2_2
 is true
! (E [true U ! (E [true U ((((((((((((((place (11) == 2  * place (0) != 0 ) * place (4)
time: 00:00:00:013
```

(a) Reaching terminal state property result for ambient garage

```
CTL model checker output:
the ctl formula
EF(participants_see_decision_on_public_display==2 & users==2  &f_user_credentials & s_user_cred
 is true
E [true U (((((((((((((((((place (54) == 2  * place (5) == 2 ) * place (8) != 0 ) * place (9)
time: 00:00:00:003
```

(b) Reaching terminal state property result for ambient conference room

Figure 7.36: Results of reaching terminal state property for both T-APN nets.

The property being true for both ambient systems means that since all the users can potentially reach the terminal states of the systems, they finally stop to run after having 'served' all the requests of the users.

This result can be translated for the ambient garage into the users' exit from the garage after having parked their cars to the allocated parking bays. Regarding the ambient conference room, the result of the property denotes that finally a decision about the reviewed papers is reached and that it is announced to all the participants after during the last phase of the procedure.

Finally, a useful observation that derives from this property is that eventually both systems terminate resulting in the conclusion that the state space of the models of those systems is finite. This will be discussed further in 7.3 by providing and examining the reachability graphs of the two nets.

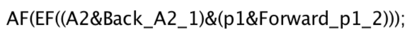
7.1.2.2.6 Interacting With Users

The following property checks whether the examined ambient systems can interact concurrently with users of different privileges and eligibility during their operation.

For the investigation of this property, the following expression is used:

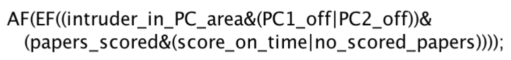
"System should be prepared to interact with both authorised and unauthorised users at any point of its execution."

The CTL proposition of the above property for the examined T-APN models is shown in Figure 7.37.



```
AF(EF((A2&Back_A2_1)&(p1&Forward_p1_2)));
```

(a) CTL proposition of interacting with users property for ambient garage

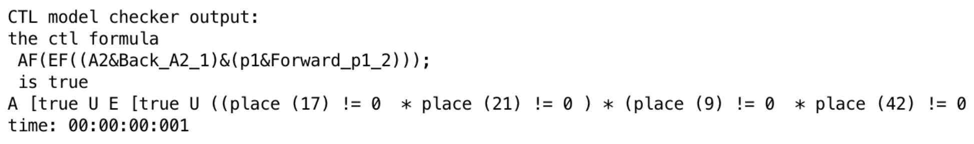


```
AF(EF((intruder_in_PC_area&(PC1_off|PC2_off))&
  (papers_scored&(score_on_time|no_scored_papers))));
```

(b) CTL proposition of interacting with users property for ambient conference room

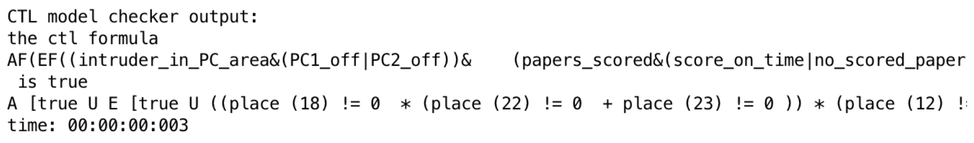
Figure 7.37: CTL propositions of interacting with users property for both T-APN nets.

To examine this property, the above CTL proposition are used as input to Charlie model checker. The outcome of which for this property is shown in Figure 7.38.



```
CTL model checker output:
the ctl formula
 AF(EF((A2&Back_A2_1)&(p1&Forward_p1_2)));
 is true
A [true U E [true U ((place (17) != 0  * place (21) != 0 ) * (place (9) != 0  * place (42) != 0
time: 00:00:00:001
```

(a) Interacting with users property result for ambient garage



```
CTL model checker output:
the ctl formula
AF(EF((intruder_in_PC_area&(PC1_off|PC2_off))&    (papers_scored&(score_on_time|no_scored_paper
 is true
A [true U E [true U ((place (18) != 0  * (place (22) != 0  + place (23) != 0 )) * (place (12) !
time: 00:00:00:003
```

(b) Interacting with users property result for ambient conference room

Figure 7.38: Results of reaching terminal state property for both T-APN nets.

The property being true for both ambient systems, means that these systems can interact concurrently with both authorised and unauthorised users at some point of their execution.

For example, the ambient garage interacts with a car that mistakenly parks to a parking bay that is not allocated to it and another car that at the same time moves to its allocated parking bay by following the advices of the garage. In this example, the first car is considered as the unauthorised user and the latter one as the authorised user. Regarding the ambient conference room, the result denotes that a reviewer can interact with the system by continuing his task while the system prompts an intruder to leave the area of the private display of another reviewer.

The ambient systems being able to interact with both authorised and unauthorised users ensures that these systems can overcome unwanted situations that are created by the unauthorised users and at the same time operates normally helping the authorised users with their goals.

7.1.2.2.7 Dealing With Improper Actions

Dealing with improper actions property checks whether an ambient system can react to improper actions and if it encourages the users to conform to the operating rules.

For the examination of this property, the following expression is used:

"Unauthorised actions are not always followed by appropriate warning advices."

The CTL propositions of the above property for the examined T-APN models are presented in Figure 7.39.

```
AG(A1&Back_A1_2 -> !(EX(p1&ctrl_p1_1)))&
AG(A2&Back_A2_1 -> !(EX(p2&ctrl_p2_1)));

AG((p1 -> A1) -> !(EX(A1&Back_A1_2)))&
AG((p2 ->A2) -> !(EX(A2&Back_p2_2)));
```

(a) CTL proposition of dealing with improper actions property for ambient garage

```
AG(intruder_in_PC_area ==2&(PC1_off | PC2_off )
    -> !(EX(home_page&(PC1_off|PC2_off))));

AG(intruder_in_PC_area ==2 -> !AX(PC1_off | PC2_off ));
```

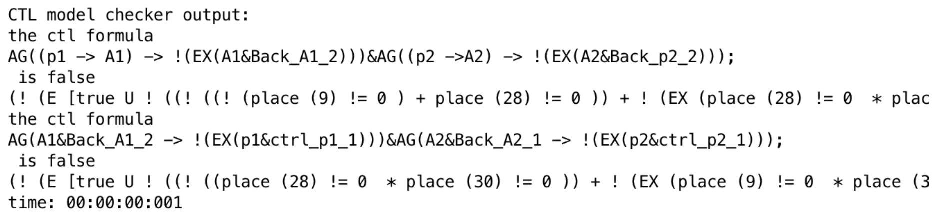
(b) CTL proposition of dealing with improper actions property for ambient conference room

Figure 7.39: CTL propositions of dealing with improper actions property for both T-APN nets.

To check this property, the above CTL propositions are used as input to Charlie model checker resulting in the outcomes shown in Figure 7.40.

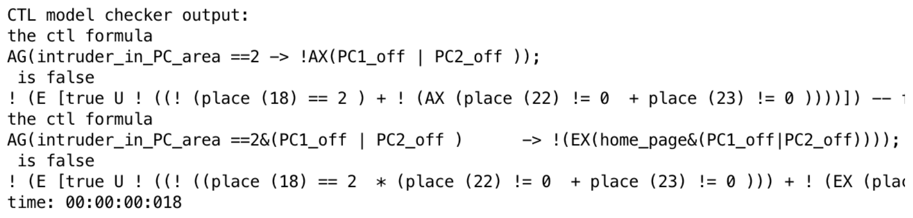
The property being false for both ambient systems denotes that for every improper action that takes place in those systems, there always exists an advice that prompts the user to 'behave' properly by making an action that will get him back to a normal state.

For example, the ambient garage interacts with a user that mistakenly parks his car to the parking bay $A1$ by prompting him to go back and find the correct parking bay. The same happens with another user that parks to the wrong parking bay ($A2$). Once again the user is warned to go back.



```
CTL model checker output:
the ctl formula
AG((p1 -> A1) -> !(EX(A1&Back_A1_2)))&AG((p2 ->A2) -> !(EX(A2&Back_p2_2)));
 is false
(! (E [true U ! ((! ((! (place (9) != 0 ) + place (28) != 0 )) + ! (EX (place (28) != 0  * plac
the ctl formula
AG(A1&Back_A1_2 -> !(EX(p1&ctrl_p1_1)))&AG(A2&Back_A2_1 -> !(EX(p2&ctrl_p2_1)));
 is false
(! (E [true U ! ((! ((place (28) != 0  * place (30) != 0 )) + ! (EX (place (9) != 0  * place (3
time: 00:00:00:001
```

(a) Dealing with improper actions property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(intruder_in_PC_area ==2 -> !AX(PC1_off | PC2_off ));
 is false
! (E [true U ! ((! (place (18) == 2 ) + ! (AX (place (22) != 0  + place (23) != 0 ))))]) -- 
the ctl formula
AG(intruder_in_PC_area ==2&(PC1_off | PC2_off )      -> !(EX(home_page&(PC1_off|PC2_off))));
 is false
! (E [true U ! ((! ((place (18) == 2  * (place (22) != 0  + place (23) != 0 ))) + ! (EX (pla
time: 00:00:00:018
```

(b) Dealing with improper actions property result for ambient conference room

Figure 7.40: Results of dealing with improper actions property for both T-APN nets.

Regarding the ambient conference room, the result denotes that the action of an intruder to get into a restricted area, like the private displays area, is always followed by the response of the system to switch off the display of the respective area forcing the intruder to leave.

At this point, it should be mentioned that the ambient conference room deals with improper actions only when an intruder appears. On the other hand, in the ambient garage, every action that does not comply with the system's advice can be considered as an improper action since the user does not follow the correct direction and the system has to redirect him.

Finally, this property shows that the ambient systems are not only able to interact with unauthorised users but they can also deal with their improper action immediately and successfully by providing the correct advice and prompting the user to follow it.

7.2 Qualitative Analysis

The qualitative analysis examines all those properties of the T-APN models that are related to both the static (structure) and dynamic (behaviour) aspect of the nets, but without considering the time aspect [28].

To present the qualitative properties and the meaning of their verification results, we provide the informal definition of these properties as is described in [11] and we interpret their verification results is into meaningful expressions about the systems' functioning.

7.2.1 Structural Properties

The qualitative analysis of the input models shown in Figures 6.8 and 6.10 starts with the discussion about the structural properties, which refer to the examination of the 'topology' of the examined nets, and in particular to the way under which their nodes[1] are connected. The verification results of the structural properties that are discussed below, were produced by Charlie verification tool [11] and are presented in Figure A1 of the Appendix.

7.2.1.1 No Input Transition (FT0) Property of T-APN nets

No input transition examines whether the transitions of a net have pre-places or not. No input transition property of a net is expressed in [11] as follows:

"There exists no transition without pre-places."

An interpretation of the no input transition property for the ambient systems could denote that the number of the resources or agents of the models are always finite.

Examining the T-APN models of the ambient garage and the ambient conference room against this property, it turns that it is true for both nets. The property being true for the ambient garage implies that the resources/agents are limited, which means that the number of users that can be in the garage is limited and bounded to a given predefined number. Likewise, the property being true for the net of ambient conference room guarantees that the number of users, reviewed papers and roles are limited to a specific number. This means that new resources cannot be arbitrarily added during the operation of the two systems, e.g. new roles.

Generally, the result of this property indicates, for both models, that the capacity of the systems is bounded implying that they can serve up to a specific number of users or deal with a particular workload of tasks.

7.2.1.2 No Output Transition (TF0) Property of T-APN nets

No output transition examines whether the transitions of a net have post-places or not. No output transition property of a net is expressed in [11] as follows:

"There exists no transition without post-places."

An interpretation of the no output transition property for the ambient systems could denote that none of the resources of the models will eventually be depleted.

Examining the T-APN models of the ambient garage and the ambient conference room against this property, it turns that the property is true for both nets. The property being true for the ambient garage implies that a user cannot disappear while moving from place to place, which means that the number of cars in the garage is

[1]Nodes refer to all the places and transitions of a net.

constant throughout the operation of the garage. The property being true for the net of ambient conference room guarantees that the number of users, reviewed papers and roles cannot be changed or reduced during the conference room session. It should be mentioned that the number of tokens in the intermediate states of the two systems can fluctuate for functioning reasons but without being depleted from an action that is taken in these models.

In general, this property indicates that the resources of the systems are always present and available during their operation.

7.2.1.3 No Input Place Property (FP0) of T-APN nets

No input place examines whether the places of a net have pre-transitions or not. No input place property of a net is expressed in [11] as follows:

"There exists no place without pre-transitions."

An interpretation of the no input place property for the ambient systems could be that there exists a limit to the resources or agents of the systems or that the systems have a starting state with limited number of resources or agents.

Examining the T-APN models of the ambient garage and the ambient conference room against this property, it turns that the property is false for both nets. The property being false for the ambient garage implies that there exists a starting point (*entrance*) for the system that defines the number of cars that can use the garage and that this number is always limited throughout the operation of the garage. Similarly for the net of ambient conference room, where the 'negative' result of the property denotes that there exists a starting point (*participants*) for the system that defines the number of authorised users that can participate in the conference and that this number is always limited to an upper bound (initial number of tokens) during the conference room session.

At this point, it should be noted that the limit on the number of tokens (resources) in both T-APN nets is ensured by the non presence of pre-transitions as regards their 'starting' points (root nets). Hence, no additional tokens (resources) can be generated arbitrarily throughout the operation of these nets. Finally, through this property, we could check from another angle the limitation on the resources of the examined systems.

7.2.1.4 No Output Place Property (PF0) of T-APN nets

No output place examines whether the places of a net have post transitions or not. No output place property of a net is expressed in [11] as follows:

"There exists no place without post-transitions."

An interpretation of the no output place property for the ambient systems could be that there exists a state in these systems where all the resources or agents are eventually gathered causing their termination.

Examining the T-APN models of the ambient garage and the ambient conference room against this property, it turns that the property is false for both nets. The property being false for the ambient garage implies that there exists a terminal state (*exit*) in the system where eventually no more actions (transitions) can be executed, as that state has at least one place with no post-transitions. Similarly for the net of ambient conference room, where the result of the property denotes that there exists a terminal state (*participants_see_decision_on_public_display*) for the system where eventually no more actions (transitions) can be executed, since that state has at least one place with no post-transitions according to the result.

The interpretation of the result of the no output place property as terminal state with no further enabled actions to be performed is deduced by the fact that the nets of the examined ambient systems have at least one place with no post-transitions, so all the tokens of each net will eventually end up to that place according to the specific structure of the T-APN nets. Finally, this property can be used to check if there exist places without post-transitions indicating whether there could be a terminal state in the examined model that consists of those places.

7.2.2 Behavioural Properties

The analysis of the examined models continues with the behavioural properties of the nets. These properties focus on the dynamic aspect of the systems by examining the structure of the nets in relation to their initial marking. The verification outputs of the following behavioural properties were produced by Charlie verification tool and are shown in Figure A1 in the Appendix.

7.2.2.1 k-Boundedness (k-B) of T-APN nets

The k-boundedness property checks if all the places of an examined net are bounded up to a number k, which means that the number of resources can not exceed that number k. The property of k-boundedness is expressed, in [11] as follows:

"A Petri net is k-bounded if all its places are k-bounded."

An interpretation of the k-boundedness property for examined ambient systems could be that the number of resources or agents, in these systems, is bounded to a constant number k throughout their operation.

Examining the T-APN models of the ambient garage and the ambient conference room against this property, it turns that the property is true for both nets when k equals 2. The property being true for the ambient garage when k equals 2, implies that the maximum number of users (cars) is equal to two for every state of the system, which means that the maximum number of cars that can be detected by every sensor of the garage is two. Similarly for the net of ambient conference room, where k equals 2 indicates that the maximum number of reviewers, reviewed papers or roles is equal to two during the sessions of the conference. The result of this property also denotes that the resources or tasks of the system are finite and that the conference cannot be suddenly overloaded with new users, papers, etc. In other words, this means that the system has limited capacity and can serve or deal with a specific number of resources or agents.

At this point, it should be mentioned that the examined T-APN nets are 2-bounded due to the fact that we use two tokens of different colours for their dynamic representation. Finally, the k-boundedness of these net results from their structure, which is 'inherited' from the specific structure of the associated composite APN nets.

7.2.2.2 Dynamic Conflict Freeness (DCF) of T-APN nets

This property checks if there exist two or more transitions in a net such that the execution of one automatically excludes the execution of the others. Dynamic conflict freeness of a net is expressed in [11] as follows:

"A net has no dynamic conflicts if no state exists, in which two transitions are enabled, which could disable each other by firing."

An interpretation of this property for the general ambient systems could be that there are no two actions in these systems that exclude each other when they are executed.

Examining the T-APN models of the ambient garage and the ambient conference room against this property, it turns that both nets are not dynamically conflict free. The property being false for the ambient garage implies that the users can have different available options at some points of the garage operation, which options cannot be concurrently executed since these actions share same resources. For example, a user could have the option to turn left or right, but when he chooses one of them, the other becomes unavailable.

Similarly for the net of ambient conference room, which is also not dynamically conflict free. The result of the property indicates that the system provides different options to the users. According to the examined model and its structure the existing conflicts in the conference room represent either the change of roles of the users, the conflict between the roles of the users and the reviewed papers or the presence of an 'intruder' in the room.

It should be noted that this property is related to the static conflict freeness examined above and, according to the rules used in Charlie verifier, it follows that $SCF \Rightarrow DCF$. Finally, the result of the dynamic conflict freeness for both models complies with the specifications of the examined systems as they enable the users to have the choice of different options at particular states of the systems' operation.

7.2.2.3 Dead States (DSt) of T-APN nets

This property checks if there exists a marking in the examined net during which all the transitions are disabled.

"A net has a dead state if no transition can be enabled any more."

An interpretation of this property for general ambient systems could be that there exists a state in the examined systems, during which they terminate either normally or not.

Examining the T-APN models of the ambient garage and the ambient conference room for dead states, it turns that both nets have only one dead state. The fact that the ambient garage has one dead state implies that it stands for the terminal state of the system where all the users (cars) have exited the garage and no more actions are available, either for the users or for the system. Similarly for the net of the ambient conference room, where the only dead state refers to the terminal state of the system, which indicates that the participants have left the room after having seen the final decision about the reviewed papers first. Therefore, the results for both models show that they terminate normally as there exists only one dead state in each model, which corresponds to the terminal state.

The importance of this property lies in the fact that possible erroneous states of the examined models can be detected through it. This happens if the number of dead states is different from what is expected with respect to the specification. Finally, a dead state does not always imply that something goes wrong with the modelling of the ambient systems as it could represent a normal situation, such as their termination.

7.2.2.4 Dead Transition (DTr) of T-APN nets

This property checks whether a transition in a net is enabled at any reachable marking. A dead transition is expressed in [11] as follows:

"There exists no transition that is disabled in all reachable markings."

An interpretation of this property for general ambient systems could be that there exists an action in all states of the system that cannot be executed.

Examining the T-APN models of the ambient garage and the ambient conference room for dead states, it turns that both nets have no dead transitions. The fact that the ambient garage has no dead transitions implies that all the actions of the system are available at least once during its operation, which means that every advice of the system and every action of the users takes places at least one time respectively. Consequently, it can be concluded that no advice or action is unavailable throughout the system operation. Similarly for the ambient conference room, where the non-presence of dead transitions implies that all the advices, tasks and roles are used at least once during the conference session indicating that none of the actions that are related to them is unavailable.

7.3 Further Verification Analysis

Having argued about the outcomes of all the properties of the above categories, the description of some other verification features like the reachability graph and the paths of the examined T-APN nets are presented. All these elements contribute to the thorough and exhaustive verification of the ambient systems, which started with the examination and the discussion of the four different categories of properties above.

7.3.1 Reachability Graph of T-APNs

In this section, the state space of the ambient systems is examined through the reachability graphs of the T-APN nets of the ambient garage and the ambient conference room, which are presented in Figures 7.41 and 7.42 respectively.

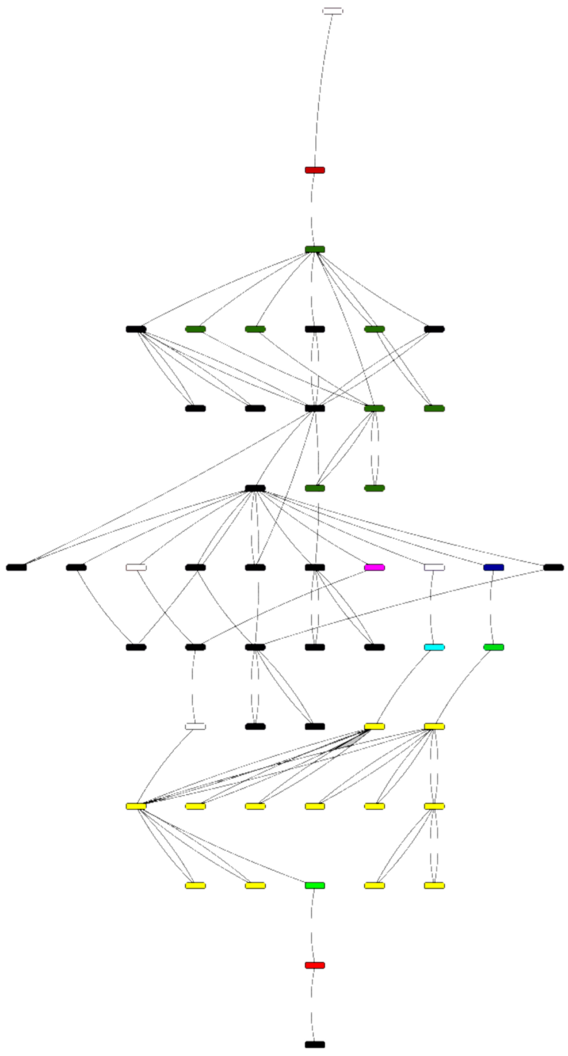


Figure 7.41: Reachability graph of ambient garage produced by Charlie [11].

It should be noted that the reachability graphs have been created considering the maximum concurrency for the examined T-APN nets, but only for visualisation purposes. As has already been mentioned, the verification of all the properties of the models is based on sequential analysis. The reachability graph of the ambient garage, in Figure 7.41, consists of 52 states that represent the different markings that are produced by following the firing rules of the T-APN net and 87 edges that stand for the performed actions (firing transitions). Finally, the reachability graph of the ambient conference room, shown below, contains 82 states and 130 edges.

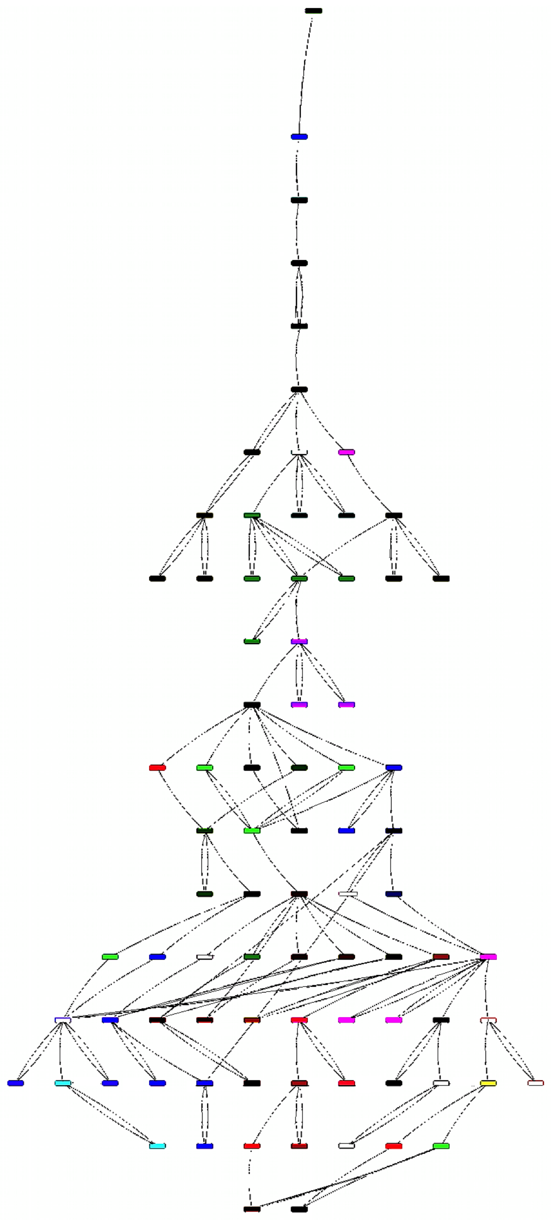


Figure 7.42: Reachability graph of ambient conference room in Charlie [11].

As it can be noticed from the Figures 7.41 and 7.42, both T-APN nets have finite state space, which denotes that the examined nets do not deal with the problem of the state space explosion [18] that is usually met to large systems with intricate behaviour. Furthermore, it can be concluded that behavioural properties like liveness, reversibility and dead states can be examined using the given reachability graphs. For example, the reachability graphs of the two T-APN nets show that these nets are not live since there exists a terminal state in each graph. Moreover, both nets are not reversible as there is no connection (edge) between any state (including the terminal state) and the initial state of the reachability graphs that would make the nets to run eternally. Finally, it can be observed that each graph contains a dead state, which is the terminal one.

7.3.2 Paths of T-APNs

To examine the paths of a T-APN net, the generated reachability graph is considered. Using paths, we can analyse the goals of the users by exploring the sequence of actions that is needed in order to be accomplished. Furthermore, the paths can be used for the examination of the user's behaviour in general by checking whether that user follows or ignores the advices. This is conducted by checking whether there exist repetitions of actions in the examined paths or not. Each time, these paths start from the initial state (initial marking) of the net and end to a state that is related to a specific target which could be any sub-goal or the final goal of the user.

To examine the above cases, two different scenarios are considered. The first scenario demonstrates the case where a user reaches his goal by always following the advices of the system. Evaluating this scenario, it is shown that the shortest way for a user to achieve his goal is by complying with the advices. The second scenario represents the case where a user reaches his goal by partially ignoring the system's advices. Assessing this scenario, it is revealed that ignoring some advices, the user has to 'loop' (revisit a state) at some point if he wants to achieve his goal.

7.3.2.1 Reaching Goal By Following Advices

The path analysis of the first case scenario considers all those cases where a user always follows the advice of the system. The aim of the path analysis, in this case, is to examine whether the paths produced by Charlie are the optimal ones. The path analysis is applied to both the ambient garage and the ambient conference room. For the path analysis, the creation of a filter file that will be used as input to Charlie is required. This file consists of the target state that represents the examined goal.

To perform the paths analysis for the ambient garage in respect of the first case scenario, only the following goal is taken into consideration, as this main goal of the users:

"Each car (user) parks to the allocated parking bay."

The expressions of all the target state used for the path analysis of the above goal are presented in Figure 7.43.

```
Left_p1_1 * A1 * Park_A1_1
```

(a) First user parks to bay $A1$ following the advices

```
Forward_p1_2 * Right_p2_2 * A2 * Park_A2_2
```

(b) Second user parks to bay $A2$ following the advices

Figure 7.43: Expressions of users' goals for the path analysis of the ambient garage.

From the above figure, it can be noticed that two different expression have been created for the path analysis due to the fact that the model of the garage operates with two users. Hence, the paths of both users should be examined.

Firstly, we examine the case where the first user of the garage parks to his allocated parking bay ($A1$) following the given advices of the system. According to Charlie, the length of this path is equal to 4, which is the shortest possible path to the parking bay $A1$, since the path analyser is set to find the shortest path. This outcome can also be confirmed by applying the transition sequence that is produced by the path analyser (see Figure 7.44) to the T-APN net of Figure 6.8.

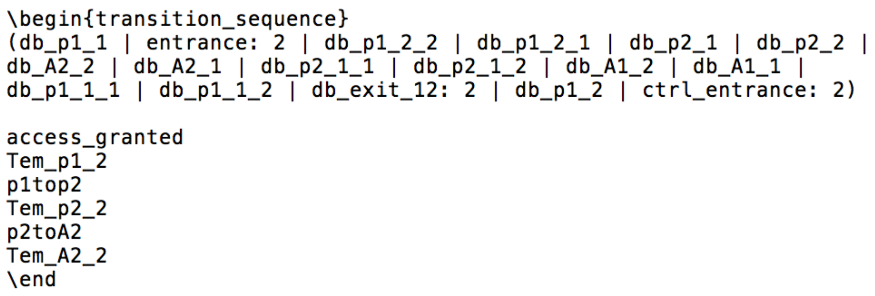
```
\begin{transition_sequence}
(db_p1_1 | entrance: 2 | db_p1_2_2 | db_p1_2_1 | db_p2_1 | db_p2_2 |
db_A2_2 | db_A2_1 | db_p2_1_1 | db_p2_1_2 | db_A1_2 | db_A1_1 |
db_p1_1_1 | db_p1_1_2 | db_exit_12: 2 | db_p1_2 | ctrl_entrance: 2)

access_granted
Tem_p1_1
p1toA1
Tem_A1_1
\end
```

Figure 7.44: Transition sequence of first user parking to bay $A1$ following the advices.

The figure above shows the initial marking of the T-APN net of the ambient garage and the sequence of transitions that are executed in order to reach the target state. The transition sequence consists of four transitions that represent the interaction between the user and the system. The transition sequence contains two actions for each side. The first action of the user is the *access_granted*, where the user moves towards $p1$ after having gained access to the garage. The next action is a system action, the Tem_p1_1. This action results in the advice which notifies the user about his next action. Then, the user takes the advice and moves from $p1$ to the parking bay $A1$, as is shown by the transition $p1toA1$. Finally the system informs the user, executing Tem_A1_1, that he has parked correctly.

Finally, examining this case for the second user, it turns that the length of the shortest path to the parking bay $A2$ is equal to 6. The result of the path analyser is shown in Figure 7.45) and it can be confirmed by applying the produced transition sequence to the T-APN net of the ambient garage, as for the case of the first user.



```
\begin{transition_sequence}
(db_p1_1 | entrance: 2 | db_p1_2_2 | db_p1_2_1 | db_p2_1 | db_p2_2 |
db_A2_2 | db_A2_1 | db_p2_1_1 | db_p2_1_2 | db_A1_2 | db_A1_1 |
db_p1_1_1 | db_p1_1_2 | db_exit_12: 2 | db_p1_2 | ctrl_entrance: 2)

access_granted
Tem_p1_2
p1top2
Tem_p2_2
p2toA2
Tem_A2_2
\end
```

Figure 7.45: Transition sequence of second user parking to bay $A2$ following the advices.

The transition sequence of Figure 7.45 consists of six transitions that represent the interaction between the second user and the system. The transition sequence contains three actions for each side. The first action of the second user, as for the first one, is represented by the execution of the *access_granted* transition, where the user moves towards $p1$ after having gained access to the garage. The next action is a system action, the Tem_p1_2. This action advices the next move of the user to be forward. The user takes the advice and moves from the place $p1$ towards the place $p2$ as is indicated by the transition $p1top2$. Then the system executes the Tem_p2_2 notifying the user to turn right. The user follows the advice once again turning right and moving towards the parking bay $A2$. Finally the system informs the user, executing Tem_A2_2, that he is allowed to park.

Now, for the path analysis of the ambient conference room, the following goals are considered:

(a) *"A user participates in all the phases contributing to the final decision."*

(b) *"A user is replaced in the second phase of the conference procedure."*

(c) *"A user participates only in the first phase."*

The goals mentioned above are all the possible goals for a user that follows the advice of the ambient conference room. To analyse the above goals, the expressions of the target states used are presented in Figure 7.46.

```
access_granted * f_log_in_done * score_on_time * auth_spokesperson *
no_conflict * decision_confirmed * participants_see_decision_on_public_display *
auth_exit_and_see_decision
```

(a) The user participates in all phases

```
access_granted * f_log_in_done * score_on_time * auth_spokesperson *
in_conflict * wait_committee * decision_confirmed_1 *
participants_see_decision_on_public_display * auth_exit_and_see_decision
```

(b) The user is replaced in the second phase

```
access_granted * f_log_in_done * score_on_time * leave_room *
no_access * participants_see_decision_on_public_display * unauth_wait_and_see_decision
```

(c) The user participates only in the first phase

Figure 7.46: Expressions of users' goals for the path analysis of the ambient conference room.

Firstly, we start with case (a), where the user participates in all the phases of the conference following the given advices of the system. According to Charlie, the length of this path is equal to 14, which is the shortest possible path to the final decision if the user is not in conflict with any of the reviewed papers. The correctness of the outcome can be confirmed manually by applying the transition sequence of Figure 7.47) to the T-APN net of Figure 6.10.

```
\begin{transition_sequence}
(participants: 2 | ctrl_participants: 2 | users: 2 | f_user_credentials |
s_user_credentials | timer_expired | on_time: 2 | PC_area_1 | PC_area_2 | PC_area: 2 |
double_role | only_reviewer | unauth_reviewer | fir_paper | sec_paper | replaced_reviewer
| member_1 | member | unauth_user_final | auth_user_final)

swipe_card
Tem_en_room
sit_and_enter_credentials
Tem_home_page_1
score_papers
Tem_papers_scored_2
log_out
Tem_given_role_1
in_committee
Tem_com_session_2
make_decision
Tem_fin_decision
announcing_decision_2
Tem_see_display_2
\end
```

Figure 7.47: Transition sequence of user participating in all the phases of the conference.

The output of Charlie above shows the initial marking of the T-APN net of the Ambient Conference Room and the sequence of transitions that are executed in order to reach the target state. The transition sequence consists of fourteen transitions that represent the interaction between the user and the system. The transition sequence contains seven actions for each side. The first action of the user is to swipe his card (*swipe_card*) to enter into the room. Afterwards, the system checks if the user is eligible for that action and informs the user that the access has been granted (*Tem_en_room*). After having entered the room, the reviewer sits in front of his private display and tries to log in (*sit_and_enter_credentials*). The system notifies the user for his successful log in and allows him to see the home page (*Tem_home_page_*1). Next, the user starts reviewing the papers and gives a score to them (*score_papers*). The system gets that score and sends back an acknowledgement saying that the score has been submitted on time (*Tem_papers_scored_*2). At this point, the reviewer logs out (*log_out*) and waits the system to inform him about whether he should participate in the second phase. The system checks the database and identifies the user as an authorised member of the committee (*Tem_given_role_*1), who can proceed to the next phase. Starting the second phase, the user participates in the committee session as an authorised spokesperson (*in_committee*). During the committee session, the user is informed about possible conflicts with the reviewed papers in order to achieve a unimpeachable decision about those papers. The system checks again its database and notifies that there is no conflict between the user and the papers (*Tem_com_session_*2). By the end of the committee session, a final decision about the reviewed papers is taken (*make_decision*). The system records the outcome of that phase (*Tem_fin_decision*) confirming in this way the final decision. Finally, during the last phase, the committee announces the final decision to all the participants (*announcing_decision_*2) and the system prompts the participants to see the outcome on the public display and then exit the room (*Tem_see_display_*2) closing the conference.

Examining the second case, it turns that the length of the shortest path to the final decision while the user is in conflict with the reviewed papers is equal to 16. The result of the path analyser is shown in Figure 7.48) and it can be applied to the T-APN net of the ambient conference room as for the first case.

The transition sequence of Figure 7.48 consists of sixteen transitions that represent the interaction between the second user and the system. The transition sequence contains eight actions for each side. In this case, the transition sequence is exactly the same as that of the first case till the point where the user is member of the committee and is informed that he is in conflict with at least one of the reviewed papers (*Tem_com_session_*1). Then, the user has to leave the session (*to_be_replaced*) and the system advises him not to return before the final decision about those papers is made (*Tem_return*). Thereafter the user returns to the session (*return_after_decision*) and participates to the final decision about the rest of the papers.

```
\begin{transition_sequence}
(participants: 2 | ctrl_participants: 2 | users: 2 | f_user_credentials |
s_user_credentials | timer_expired | on_time: 2 | PC_area_1 | PC_area_2 | PC_area: 2 |
double_role | only_reviewer | unauth_reviewer | fir_paper | sec_paper | replaced_reviewer
| member_1 | member | unauth_user_final | auth_user_final)

swipe_card
Tem_en_room
sit_and_enter_credentials
Tem_home_page_1
score_papers
Tem_papers_scored_2
log_out
Tem_given_role_1
in_committee
Tem_com_session_1
to_be_replaced
Tem_return
return_after_decision
Tem_fin_decision1
announcing_decision_1
Tem_see_display_2
\end
```

Figure 7.48: Transition sequence of user being replaced during the second phase.

The final decision about all the papers is confirmed by the system ($Tem\_fin\_decision1$) and the decision is released to all the participants by the committee ($announcing\_decision\_1$). Finally, the users are informed that the final decision is available on the public display and that after seeing it they can leave the room ($Tem\_see\_display\_2$).

Finally, examining the last case, the length of the shortest path for a user to the final decision is equal to 12. This happens because the path analysis of this case is examined independently from the all the other users, who could participate to all the phases of the conference. It should be noted that the shortest path of this case would be longer if it was examined in relation to the other users that take part in all the phases of the conference, but this is out of the scope of the path analysis for the ambient conference room The result of the path analyser is shown in Figure 7.49) and it can be applied to the T-APN net of the ambient conference room as with the first two cases.

```
\begin{transition_sequence}
(participants: 2 | ctrl_participants: 2 | users: 2 | f_user_credentials |
s_user_credentials | timer_expired | on_time: 2 | PC_area_1 | PC_area_2 | PC_area: 2 |
double_role | only_reviewer | unauth_reviewer | fir_paper | sec_paper | replaced_reviewer
| member_1 | member | unauth_user_final | auth_user_final)

swipe_card
Tem_en_room
sit_and_enter_credentials
Tem_home_page_1
score_papers
Tem_papers_scored_2
log_out
Tem_given_role_2
reviewer_leaves
Tem_awaiting
announcing_decision
Tem_see_display_1
\end
```

Figure 7.49: Transition sequence of user participating only in the first phase.

The transition sequence of Figure 7.49 consists of twelve transitions that represent the interaction between user and the system. The transition sequence contains six actions for each side. In this case, the transition sequence is exactly the same as those of the first two case till the point where the user log out and the system prompts him to leave ($Tem\_given\_role\_2$). Then, the user leaves ($reviewer\_leaves$) the room and the system informs him that access is not permitted to him any more (no_access) till the end of the conference. Afterwards, when the final decision is released the user can enter the room again ($announcing\_decision$). Finally, after the announcement of the decision the system induces the user not to leave without having seen the final decision ($Tem\_see\_display\_1$), closing in that way the conference.

Having checked the paths produced by the examination of the goals of the users that always comply with the rules, it results that the shortest possible path is always produced for the completion of each of these goals.

7.3.2.2 Reaching Goal By Ignoring Advices

The following paragraphs describe the cases in which the users partially ignore the advices of the ambient systems acting improperly till the point where they start to comply with the advices again in order to achieve their goals. Examining those cases for both T-APN model, we intent to show that the users cannot avoid revisiting some states at some point in order to accomplish their goal. Furthermore, it is proved that the generated paths of the cases are longer than the respective paths of the cases examined in section 7.3.2.1.

To demonstrate the consequences of partially ignoring the advices for the ambient garage, the following scenario is used:

"Each car (user) parks to the allocated parking bay after ignoring advices only once."

The above scenario is applied to the ambient garage for both the first and the second user of the T-APN net, as it happens with the respective scenario in 7.3.2.1.

Using the target states of Figure 7.50 as input to the path analyser of Charlie for the cases of the first and the second user, the length of the two generated paths is 8 and 10 respectively.

```
entrance * ctrl_entrance * Left_p1_1 * A1 * Park_A1_1 * Back_p2_1 * Right_p1_1
```

(a) First user reaches goal ignoring advices

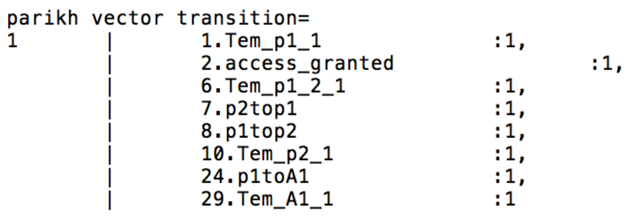
```
entrance * ctrl_entrance * Forward_p1_2 * Right_p2_2 * A2 * Park_A2_2 * Back_A1_2 *
Left_p1_2
```

(b) Second user reaches goal ignoring advices

Figure 7.50: Expressions of users' goals ignoring advices of the ambient garage.

In the first case of Figure 7.50(a), the car enters the garage ($access\_granted$) being advised to turn left (Tem_p1_1). Thereafter, the user instead of following that advice, he gets to $p2$ by going forward ($p1top2$). Then, the system prompts the user to go back (Tem_p2_1) to $p1$ in order to park to the allocated parking bay $A1$. At this point, the user decides to follow the advices till the completion of his goal. Thus, he gets back to $p1$ ($p2top1$), where he is advised to turn right ($Tem\_p1\_2\_1$). Turning right, the user heads towards the parking bay $A1$ ($p1toA1$) achieving his goal. Finally, the system informs the user that he can park to $A1$ (Tem_A1_1).

The result of the path analyser for the path described above is presented in Figure 7.51 as a parikh vector. The parikh vector records how many times every transition is executed during the path [31]. The parikh vector expresses the transition sequence of the path but in unordered way.



```
parikh vector transition=
1       |       1.Tem_p1_1              :1,
        |       2.access_granted                :1,
        |       6.Tem_p1_2_1            :1,
        |       7.p2top1                :1,
        |       8.p1top2                :1,
        |       10.Tem_p2_1             :1,
        |       24.p1toA1               :1,
        |       29.Tem_A1_1             :1
```

Figure 7.51: Parikh vector of the first user achieving goal after having ignored an advice.

Checking the produced parikh vector of the first case, it can be noticed that the user visits place $p1$ twice. This derives from the execution of the transitions $p1top2$ and $p2top1$, which implies that the user is looping. Finally, as regards the length of the path for this case, it turns that is longer than the path of Figure 7.44 since the first is equal to 8 and the latter equal to 4.

In the second case of Figure 7.50(b), the car enters the garage ($access\_granted$) being advised to go forward (Tem_p1_2), the user instead of following that advice, he gets to $A1$ by turning left ($p1toA1$). Then, the system prompts the user to go back to $p1$ (Tem_A1_2) in order to park to the allocated parking bay $A2$. From this point, the user follows all the given advices till the completion of his goal. Thus, he gets back to $p1$ ($A1top1$), where he is advised to turn left ($Tem\_p1\_1\_2$). Turning left, the user heads towards the place $p2$ ($p1top2$), where the system prompts him to turn right (Tem_p2_2) in order to get to the parking bay $A2$. The user turn right following the advice and gets to $A2$ ($p2toA2$) reaching his goal. Finally, the system informs the user that he has parked to $A2$ successfully (Tem_A2_2).

The result of the path analyser for the path described above is presented in Figure 7.52 in the form of parikh vector, as in the first case. Checking the produced parikh vector of the second case, it can be noticed that the user visits place $p1$ twice. This derives from the execution of the transitions $p1toA1$ and $A1top1$, which implies that the user is looping. Finally, the length of the path for this case, it turns to be longer than the path of Figure 7.45 as the first is equal to 10 and the latter equal to 6.

```
parikh vector transition=
1       |       2.access_granted                :1,
        |       8.p1top2                :1,
        |       12.Tem_p2_2             :1,
        |       14.p2toA2               :1,
        |       16.Tem_A2_2             :1,
        |       24.p1toA1               :1,
        |       25.A1top1               :1,
        |       27.Tem_A1_2             :1,
        |       32.Tem_p1_1_2           :1,
        |       36.Tem_p1_2             :1
```

Figure 7.52: Parikh vector of the second user achieving goal after having ignored an advice.

For the examination of the ambient conference room with regard to the length of the produced path after ignoring advices and the looping of the user, only one case is examined. This occurs due to the fact that in the ambient conference room the authorised users always follow the advice and the only situation where a user does not follow the advices of the system is when an intruder is in the room. Thus, using a scenario that consists of an intruder and a user that participates only in the first phase of the conference could be a representative example. As a consequence, the following scenario is used for the path analysis:

"A user that participates only in the first phase is interrupted by an intruder."

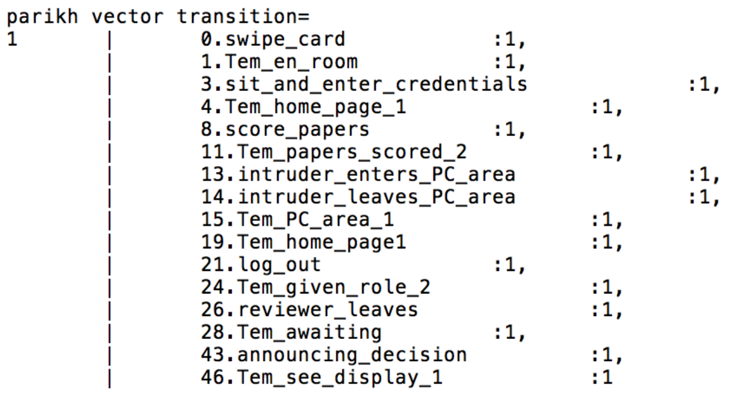
The target state for this scenario is shown in Figure 7.53. The length of the path produced by the path analyser of Charlie for this case is equal to 16.

```
participants * ctrl_participants * access_granted * f_log_in_done * PC1_off *
respective_PC_on * score_on_time * leave_room * no_access *
participants_see_decision_on_public_display * unauth_wait_and_see_decision
```

Figure 7.53: Expression of target state of the scenario for the ambient conference room.

Examining the target state of Figure 7.53, the generated path consists of sixteen actions. The first action of the user is to swipe his card (*swipe_card*) to enter into the room. Afterwards, the system checks if the user is eligible for that action and informs the user that the access has been granted (*Tem_en_room*). After having entered the room, the reviewer sits in front of his private display and tries to log in (*sit_and_enter_credentials*). The system notifies the user for his successful log in and allows him to see the home page (*Tem_home_page_1*). At this point, an intruder appears interrupting the session and ignoring in that way the advice of the system (*intruder_enters_PC_area*). Then the system switches off the private displays and notifies the authorised users about its action (*Tem_PC_area_1*). After a while, the intruder leaves the area of the private displays and the room (*intruder_leaves_PC_area*). The system detects that the intruder has left the area and turns the appropriate private displays back on (*Tem_home_page*1) to the home page, where the session has stopped when the intruder entered the room and the restricted area. From that point, the process is not interrupted any more and the user follows the given advices exactly as it happens with the third case of section 7.3.2.1.

The result of the path analyser for the path described above is presented in Figure 7.54 as a parikh vector.



```
parikh vector transition=
1       |       0.swipe_card            :1,
        |       1.Tem_en_room           :1,
        |       3.sit_and_enter_credentials             :1,
        |       4.Tem_home_page_1               :1,
        |       8.score_papers          :1,
        |       11.Tem_papers_scored_2          :1,
        |       13.intruder_enters_PC_area              :1,
        |       14.intruder_leaves_PC_area              :1,
        |       15.Tem_PC_area_1                :1,
        |       19.Tem_home_page1               :1,
        |       21.log_out              :1,
        |       24.Tem_given_role_2             :1,
        |       26.reviewer_leaves              :1,
        |       28.Tem_awaiting         :1,
        |       43.announcing_decision          :1,
        |       46.Tem_see_display_1            :1
```

Figure 7.54: Parikh vector of user achieving goal after being interrupted by an intruder.

Checking the produced parikh vector of this case, it can be noticed that the state *home_page* is visited twice. This derives from the execution of the transitions *intruder_enters_PC_area* and *intruder_leaves_PC_area*, which start from *home_page* and end to the *home_page* respectively, implying that the user is looping. This can also be verified if we look at the T-APN net of the ambient conference room in Figure 6.10 and check the pre- and post-places of these two transitions of the net. Then, it results that these two transitions create a loop (bidirectional step). Finally, as regards the length of the path for this case, it turns to be longer than the path of Figure 7.49 as the first is equal to 16 and the latter equal to 12.

Having examined the cases where a user always follows the advices or partially ignores the advices of the system, it can be concluded that the shortest path is always provided when the user follows the advice. In all the other cases, the generated paths are not the optimal ones and are not so cost effective for the users since more actions take place and more time is needed for the generation of the path. In conclusion, when a user ignores some advices, he has to loop if he wants to complete his goal.

7.3.3 Fault-Tolerance of T-APNs

In the previous sections of this chapter, we examined all those cases where the systems were responding correctly to the users' actions. On the contrary, in the fault-tolerance analysis, we assume that the systems can provide wrong advices to the users. This could happen by modifying our models by adding some extra tokens to the database places making in that way one of the systems' sensors faulty. To analyse how the behaviour of a defective sensor affects the goals of the users and the functioning of the systems, we examine one property for each case. These properties are expressed as CTL formulas and are examined through the model checker of Charlie verification tool, as with all the other model checking properties[1].

The first property of the fault-tolerance analysis is related to the accomplishment of the user's goal after a wrong advice from the system. This property is expressed as follows:

"A user can reach his final goal even after a wrong advice of the system."

The CTL proposition for the above expression is shown in Figure 7.55 for both the ambient garage and the ambient conference room.

```
AG(p2 == 2 & Back_p2_1 == 2 -> EF(exit & Bye_exit_12));
```

(a) CTL proposition of reaching goal after wrong advice property for ambient garage

```
AG(intruder_in_PC_area == 2 & PC1_off == 2 ->
EF(participants_see_decision_on_public_display & unauth_wait_and_see_decision
& auth_exit_and_see_decision));
```

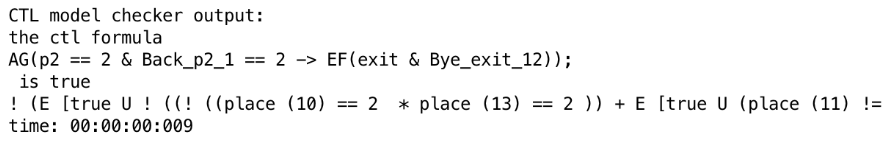
(b) CTL proposition of reaching goal after wrong advice property for ambient conference room

Figure 7.55: CTL propositions of the reaching goal after wrong advice property for both T-APN nets.

Inserting these CTL propositions into the model checker of Charlie, we get the results shown in Figure 7.56.

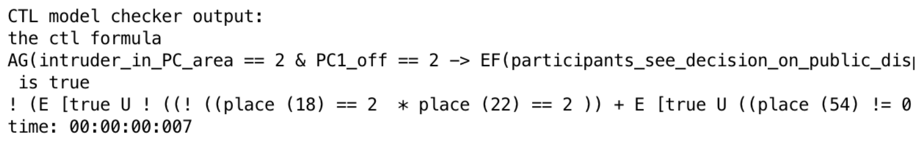
The first property of the fault-tolerance for the ambient systems that is related to the accomplishment of the user's goal after a wrong advice of the systems is true for both T-APN models. This property provides useful information mostly about the ambient garage as its functioning is based on the sensors. On the other hand, the result of this property for the ambient conference room is less important as the system uses sensors only in the areas of the private displays.

[1]Due to the fact that the examination of these properties considers the faulty behaviour of the systems, we present them in a different section and not in the same with all the other model checking properties, which rely on the correct behaviour of the systems.



```
CTL model checker output:
the ctl formula
AG(p2 == 2 & Back_p2_1 == 2 -> EF(exit & Bye_exit_12));
 is true
! (E [true U ! ((! ((place (10) == 2  * place (13) == 2 )) + E [true U (place (11) !=
time: 00:00:00:009
```

(a)Reaching goal after wrong advice property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(intruder_in_PC_area == 2 & PC1_off == 2 -> EF(participants_see_decision_on_public_dis
 is true
! (E [true U ! ((! ((place (18) == 2  * place (22) == 2 )) + E [true U ((place (54) != 0
time: 00:00:00:007
```

(b)Reaching goal after wrong advice property result for ambient conference room

Figure 7.56: Results of Reaching goal after wrong advice property for both T-APN nets.

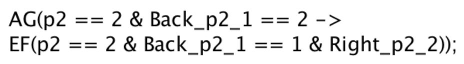
Interpreting the result of the first property for each of the examined ambient systems, it can be concluded that a user can eventually reach his final goal after a wrong advice of the system. Specifically, in the ambient garage, when a user receives a wrong advice about his next action and follows it by mistake, he can accomplish his goal in the future as the system redirects him through the sensors and the output devices that work correctly.

The result of this property for the ambient conference room indicates that despite the fact that a sensor may not work properly prompting the system to provide incorrect advice to the intruder, the authorised users can eventually complete their tasks after the intruder leaves the room or the area of the private displays. This happens due to the fact that the system stops the running session when an unauthorised user enters the room or those areas.

The last property of the fault-tolerance analysis examines whether a defective sensor that caused the wrong advice of the system could work correctly in the future prompting the environment to provide the user with the proper advice for exactly the same action. This property is expressed as follows:

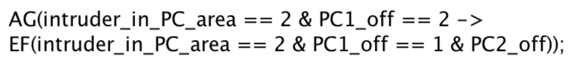
"The system can potentially give a correct advice after a wrong one for the same action of the user."

The above expression is translated into the CTL propositions of Figure 7.57 for the examined T-APN models.



```
AG(p2 == 2 & Back_p2_1 == 2 ->
EF(p2 == 2 & Back_p2_1 == 1 & Right_p2_2));
```

(a) CTL proposition of giving correct advice after a wrong one property for ambient garage

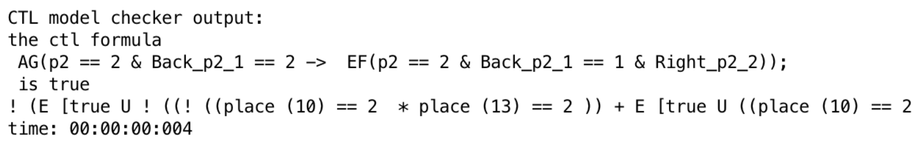


```
AG(intruder_in_PC_area == 2 & PC1_off == 2 ->
EF(intruder_in_PC_area == 2 & PC1_off == 1 & PC2_off));
```

(b) CTL proposition of giving correct advice after a wrong one property for ambient conference room

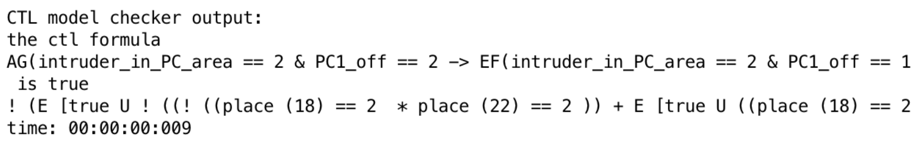
Figure 7.57: CTL propositions of giving correct advice after a wrong one property for both T-APN nets.

Checking these CTL propositions for both T-APN models using the model checker of Charlie, we get the results shown in Figure 7.58.



```
CTL model checker output:
the ctl formula
 AG(p2 == 2 & Back_p2_1 == 2 ->  EF(p2 == 2 & Back_p2_1 == 1 & Right_p2_2));
 is true
! (E [true U ! ((! ((place (10) == 2  * place (13) == 2 )) + E [true U ((place (10) == 2
time: 00:00:00:004
```

(a)Giving correct advice after a wrong one property result for ambient garage



```
CTL model checker output:
the ctl formula
AG(intruder_in_PC_area == 2 & PC1_off == 2 -> EF(intruder_in_PC_area == 2 & PC1_off == 1
 is true
! (E [true U ! ((! ((place (18) == 2  * place (22) == 2 )) + E [true U ((place (18) == 2
time: 00:00:00:009
```

(b)Giving correct advice after a wrong one property result for ambient conference room

Figure 7.58: Results of giving correct advice after a wrong one property for both T-APN nets.

The result of this property being true for the ambient garage means that the environment can provide the user with a correct advice after having given a wrong one. This can happen when the user repeats the same action being detected by the same sensor.

Likewise, the result of this property for the ambient conference room indicates that the environment can provide the intruder with the proper advice after having given the wrong one first. This case takes place only if the intruder repeats the same action while he is in the room or the area of the private displays.

Having analysed these fault-tolerance properties, it can be concluded that the examined T-APN models of the ambient systems support the return of the environment to a reliable and stable condition after its misbehaviour.

7.4 Concluding Remarks

In this chapter, we addressed the verification of the produced T-APN models of the ambient garage and the ambient conference room respectively. The discussion about the verification has been organised into three general areas, qualitative analysis, model checking of the models and further verification analysis.

In the model checking, we discussed all those properties that are related to the key features and the general functioning of the ambient systems, as they were presented in sections 2.1.1 and 2.2.5. From the verification results of these properties, it has been concluded that the produced T-APN models are correct with respect to the specification of the ambient systems. This derives from the fact that the models were verified successfully against properties that were expressing the interactivity of the systems with the users with respect to their detection, notification, initiative, restriction, etc. providing the desired outcomes each time.

In the qualitative analysis, we presented all those properties of the T-APN models that are related to both the static (structure) and dynamic (behaviour) aspect of the nets by examining fundamental properties of Petri nets. Interpreting the verification results of both the structural and the behavioural properties of the examined T-APN models, it has been concluded that the nets of the examined ambient systems do not face any unexpected deadlock case and terminate normally as the only dead state that is detected through the analysis is that of the systems' termination. Furthermore, useful observations has been made regarding the properties that are related to the number of the resources or agents that the systems can deal with. The verification results indicated that the capacity of the systems is limited and that only specific number of resources or agents can be handled during their operation. Moreover, the qualitative analysis showed that all the actions of these systems can potentially take place at least one time during their operation implying in that way that all these action contribute to the functioning of the systems.

In further verification analysis, we examined the state space of T-APN models and the impact of the users behaviour on the completion of their goals. This analysis was conducted through examination of the produced concurrent reachability graphs of the T-APN models and the generation of paths respectively. In the first case, it was shown that the state space of the examined models of the ambient systems is finite, which implies that these models do not face state explosion problem. Moreover, the investigation of the concurrent reachability graphs of the T-APN models confirmed the verification results of properties like, connectedness, liveness and reversibility. In the latter case, we examined the consequences of following and ignoring the advices of the system in regard to the completion of the users' goal by comparing the length of the paths of the two cases. From this comparison, it resulted that the users that always follow the advices of the system reach their goals in less states than the case where they ignore at least one advice. From, the path analysis has been concluded that the system in both T-APN models has been modelled in order to provide always the correct advice and the shortest path to the goal. Finally, we examined two cases where the system provides wrong advices to the users due to a defective sensor of the system. In those two cases, firstly we checked whether the users can reach their final goal after having taken a wrong advice by the system or not and secondly we examined if the defective sensor can provide a correct advice after having given a wrong one first. The verification results of these cases showed that the user can reach his goal regardless of the malfunction in one of the sensors and that the defective sensor can provide correct advices to the user for the same action in the future respectively making in that way the system stable and reliable again.

In conclusion, the verification of the T-APNs showed that the models of the ambient systems that were produced through the step-modelling approach, the application of APNs and their translation into T-APNs are correct with respect to the specification and also comply with the theory of the T-APNs. It could also be deduced that the APN models of the case studies are correct as well since their transformation into T-APNs resulted in the creation of the behaviourally equivalent examined models. The only difference is that some of the qualitative properties might have different verification results due to the slightly different structure of the nets and the presence of the inhibitor arcs.

Chapter 8

Conclusion and Future Work

This thesis addressed the issue of modelling and verification of ambient systems using Petri nets. The main focus of this work was to introduce an efficient modelling approach that would be capable of modelling the interactive behaviour of ambient systems faithfully.

To achieve that, we first examined the available literature about the ambient systems in order to identify all those components and key features that influence the interactivity of these systems. Throughout this research, it was noticed that different definitions, which have common points of reference, have been given to this kind of systems up to date. For that reason, we introduced a generic definition of the ambient systems that incorporates the main points that are mentioned in those definitions. Based on that definition, we created a taxonomy that classifies the general ambient systems into two categories, the Ambient Guidance Systems and the Ambient Information Systems. The taxonomy of the ambient systems into one of the above categories is conducted considering as criteria factors and features that affect the interaction of the environment with the users. Consequently, we chose two representative case studies, one for each category of the taxonomy, in order to analyse the behaviour of these systems through the modelling of the systems' features and components that determine their interactivity in each case.

Thereafter, to model the interactive behaviour of the ambient systems, we defined a subclass of Coloured Petri Nets that provides coloured inhibitor nets, the Ambient Petri Nets. This class was introduced in order to capture and express the structure and the concurrent and interactive behaviour of ambient systems by providing models that reflect the key principles and features of these systems.

Having defined the class of Ambient Petri Nets that would be used for the modelling of ambient systems, we continued with the introduction of the step-modelling approach that is based both on the nature of the ambient systems to interact with the users step by step and on the modularity and compositionality of APNs. In particular, the rationale behind the step-modelling approach is that a small APN net of specific structure (basic step net) can be used as the fundamental building block for the behavioural representation of the ambient systems. This building block captures a single action of a user and the response of the environment to that action. To build the full model of an ambient system out of that building block, we had to glue it several times. Thus, we defined two composition operators: the forward composition operator that extends an APN net by a building block each time that is used and the backward composition operator that creates cyclic APN nets or APN nets with cyclic subnets.

Having applied those compositions and having followed the requirements of the specification, we obtained the APN models of the ambient systems of the chosen case studies, which represent all the potential actions that can be taken either by a user or by the system in order to interact with each other. The main characteristic of these models is the presence of colour-sensitive inhibitor arcs that are used to 'restrict' the users from acting before getting informed by the system. The presence of these inhibitor arcs gives a more realistic aspect of the system interactivity to the models, as every user's action in an ambient system is followed by the immediate response of the environment for that action.

After the completion of the APN models of the case studies, we wanted to verify their correctness using Charlie verification tool. Due to the fact that Charlie and most of the existing verification tools do not support the analysis of inhibitor nets like APNs, we had to transform the APN models into verifiable models with no inhibitor arcs. To tackle this problem, we defined another subclass of Coloured Petri Nets, the T-APN class, which provides non inhibitor nets. Furthermore, we introduced a construction that enables the conversion of the APN nets to the respective T-APN nets. This construction was used to obtain the T-APN models of the case studies out of the APN models that had been developed earlier. But, to use the obtained T-APN models for the verification of the ambient systems, we had to prove first that the T-APN nets produced by the construction are behaviourally equivalent to the associated APN nets. This would be proved by showing that their concurrent reachability graphs are isomorphic. Afterwards, proving that indeed the concurrent reachability graphs of two nets that are associated through the construction are isomorphic, it enabled us to use the obtained T-APN models of the case studies in the verification process.

Finally, the verification of the T-APN models resulted in remarkable observations about the ambient systems. From the observations about the interactive behaviour of these systems, it can be concluded that the ambient environments of both categories always provide the correct advice to the users and that the user is the only one who is able to 'misbehave' during the interaction with the systems. Furthermore, that misbehaviour (or initiative) of a user always has an impact on the completion of his goal. For example, in the ambient garage, this impact could be translated as longer distance to get to the allocated parking bay. Moreover, we extended the verification process by examining the T-APN models against the case where the ambient systems were able to provide wrong advices to the users due to the malfunctioning of a sensor of the systems. This case has been not examined in the above verification process as the users were the only ones who could misbehave. Through the analysis of this case, we observed that the ambient systems can return to a reliable condition after having given a wrong advice helping in that way the users to complete their tasks or goals. Finally, the verification of the T-APN models of the case studies against all the examined properties showed that the models are correct with respect to the specification of the systems.

At this point, we discuss the scalability of the proposed modelling approach with respect to Charlie verification tool. More specifically, we examine how the extension of the developed T-APN models of the case studies could affect the capability of Charlie to cope with the verification of these models when they host a large number of users. To check how Charlie responds to that case, we modified the developed T-APN models gradually in order to represent the interaction of the systems with a different range of users.

We start the discussion about scalability with the model of the ambient garage, which was extended gradually in order to be able to 'serve' up to several different users. The modification of the model took place by adding all those building blocks and tokens needed in order to represent the additional parking spots and users respectively. Then, importing the modified models into Charlie each time, we observed that due to the addition of the building blocks and the tokens, the state space was increased dramatically after a certain number of users resulting in a time consuming verification process that provided reachability graphs of thousand or million states. This fact made the model checking and the path analysis of the different models of the garage very cumbersome as the time that was demanded for the verification of a single property was significant. On the contrary, it should be mentioned that Charlie dealt effectively with the verification of the modified models of the ambient garage when the number of users was up to fifty five providing the respective results within a reasonable time[1]. In this case, the almost immediate collection of the results facilitated and enabled the examination of the system properties as the verification process of the models was cost effective in terms of time.

Now, we continue with the model of the ambient conference room, which was modified for the sake of the scalability discussion, as with the T-APN model of the ambient garage. The main difference between the modification of the T-APN model of the ambient conference room and that of the T-APN model of the ambient garage is that in the first one we did not have to add more building blocks to represent the additional users, tasks or roles needed. All these were incorporated in the modified versions of the model by adding the appropriate number of tokens, database and response places each time. This happens due to the fact that additional tasks or roles can be represented by the already existing building blocks. After the modification of the model, we inserted the modified versions into Charlie for their verification realising that the state space was increased each time as in the case of the ambient garage, but not so significantly.

[1]Note that Charlie provided the verification result of a property within seconds when the users were up to 11. After that value, the result was produced within a couple of minutes. Finally, when the model had more than fifty five users, Charlie needed more than ten minutes to verify each examined property.

Furthermore, verifying the model for different number of tokens, it was observed that Charlie was able to provide the verification results (e.g. model checking and path analysis of the models) effectively and within a reasonable time when the number of these tokens were up to one hundred and forty. On the other hand, the verification of the model for number of tokens greater than that mentioned above was extremely time consuming and totally not cost effective as sometimes Charlie could not even produce the respective reachability graphs or provide the verification results due to problems related to high consumption of memory, which made it crash[1].

From all the above, it can be concluded that the APN class, and by extension the T-APN class accommodate very well the modelling of the interactivity of real ambient systems by producing correct models with nice graphical representation through the use of fundamental building blocks, compositions and construction. Thus, the step-modelling approach that is proposed in this thesis can be used for the faithful modelling and reasoning of general ambient systems. Furthermore, it can be noted that Charlie supports effectively the verification of scalable T-APN models of both Ambient Guidance Systems and Ambient Information Systems, but up to a certain number of users. In more detail, Charlie can analyse models with a small number of users in the case of the first category of ambient systems, but this not the case for the latter category of these systems where Charlie can reliably verify models that host a quite large number of users. Finally, it should be mentioned that the building block of the step-modelling approach can be used for the modelling of ambient systems focusing on different abstraction levels of the interactivity of these systems.

8.1 Future Work

This thesis proposed a modelling approach that is based on the modularity and the compositionality of the nets of the APN class and is used for the modelling of the ambient systems' interactivity. The introduction of this modelling approach brings up new directions for further research. Some of which are briefly described in the following paragraphs.

Firstly, we could examine the applicability of the proposed step-modelling approach to the modelling of other aspects of the ambient systems or even to the modelling of other kinds of systems that may not incorporate interactive behaviour by adjusting the use of the building blocks to the needs or the features of those systems.

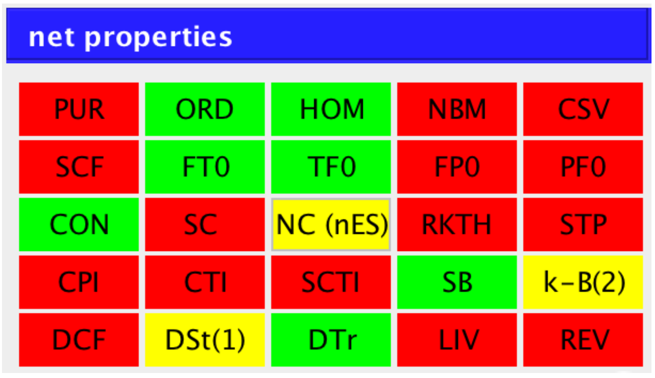
[1]Apart from the size of the examined model, memory consumption problems and time of getting the results depend also on how powerful is the machine on which Charlie operates. For the analysis of the produced T-APN models of this work, a MacBook Pro with 2.4 GHz Intel Core i5 processor and 8 GB 1600 MHz DDR3 memory was used. For example, a machine that has a better specification could enable Charlie to analyse bigger T-APN models.

Another direction that could be investigated is the extension of Charlie verification tool in order to support the verification of inhibitor nets like APNs. Despite the fact that the verification of inhibitor nets is a challenging and demanding task, the development of such an extension would be of great interest and could open new perspectives for the domain of modelling and verification due to the fact that the transformation of inhibitor nets to non-inhibitor nets would be redundant as all the inhibitor nets would be verifiable.

As regards the enhancement of the step-modelling approach, we could consider the addition of the modularity and the compositionality of the T-APNs by introducing the composition and the reverse construction of the T-APN nets that could compose modular T-APN nets and convert the T-APN nets to APN nets respectively. The introduction of these two could strengthen the relation between the two classes. Finally, interesting observations could emerge through the conversion of non-inhibitor nets to inhibitor nets and vice versa.

Appendix

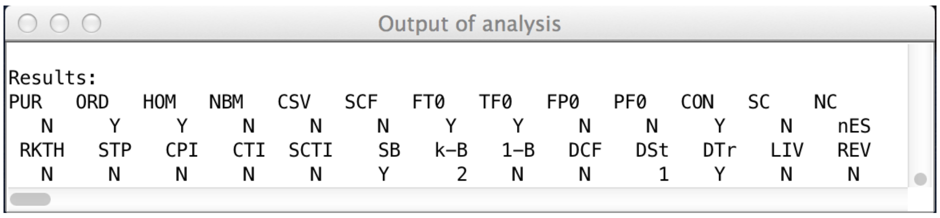
The following figure shows the verification results that are used for the elaboration of this work.



net properties

🟥 PUR	🟩 ORD	🟩 HOM	🟥 NBM	🟥 CSV
🟥 SCF	🟩 FT0	🟩 TF0	🟥 FP0	🟥 PF0
🟩 CON	🟥 SC	🟨 NC (nES)	🟥 RKTH	🟥 STP
🟥 CPI	🟥 CTI	🟥 SCTI	🟩 SB	🟨 k-B(2)
🟥 DCF	🟨 DSt(1)	🟩 DTr	🟥 LIV	🟥 REV

(a) Graphical representation



Output of analysis

```
Results:
PUR   ORD   HOM   NBM   CSV   SCF   FT0   TF0   FP0   PF0   CON   SC    NC
  N     Y     Y     N     N     N     Y     Y     N     N     Y     N   nES
 RKTH   STP   CPI   CTI  SCTI    SB   k-B   1-B   DCF   DSt   DTr   LIV   REV
  N     N     N     N     N     Y     2     N     N     1     Y     N     N
```

(b) Tabular representation

Figure A1: Charlie output representations for structural and behaviour properties of both T-APN nets.

Proving Theorem

The following paragraphs describe the equations and lemmas needed for the proof of theorem 1 in chapter 6.

Equations

For the proof of the isomorphism of the concurrent reachability graphs of $\mathcal{N}$ and $\varphi(\mathcal{N})$, we define the following equations:

$CRG(\mathcal{N}) = ([M_0\rangle_{\mathcal{N}}, \Delta, M_0)$ is a step transition system over T, where $(M_1, U, M_2) \in \Delta$ *iff* $M_1[U\rangle_{\mathcal{N}} M_2$.
$CRG(\varphi(\mathcal{N})) = ([M_0'\rangle_{\varphi(\mathcal{N})}, \Delta', M_0')$ is a step transition system over T, where $(M_1, U, M_2) \in \Delta'$ *iff* $M_1[U\rangle_{\varphi(\mathcal{N})} M_2$.

First, we define a mapping $\psi : [M_0\rangle_{\mathcal{N}} \to [M_0'\rangle_{\varphi(\mathcal{N})}$ as follows:

$$\psi(M) = \sum_{(p,g)\in\widehat{P}} \psi(M)((p,g)) \cdot m_{(p,g)} \tag{1}$$

where $\sum$ denotes the multiset sum, $m_{(p,g)}$ is a multiset over $\widehat{P}$ containing only one element (p,g) (see definition 1) and

$$\psi(M)((p,g)) = \begin{cases} M((p,g)) & \text{if } p \in P \setminus \{p_c^0\} \wedge g \in Cl_{ID}, \\ 1 - M((p,g)) & \text{if } p = p_c^0 \wedge g \in Cl_{ID}, \\ 1 - M((p, f^{-1}(g))) & \text{if } \big(p \in (P_C \cap P_I) \wedge (\exists_{t\in T_{STEP}\setminus\{\epsilon\}} : \\ & Post(t)(p) = 0 \wedge Post'(t)(p) = 1 \wedge \\ & M((p_t, f^{-1}(g))) > 0) \wedge g \in Cl_{SPEC}\big) \\ & \vee\big(p \in (P_C \cap P_I) \wedge (\exists_{t'\in T_{EM}} : \\ & Post(t')(p) = 0 \wedge Post'(t')(p) = 1) \wedge \\ & (\exists_{t\in T_{STEP}\setminus\{\epsilon\}} : p = cp^t(p_t) \wedge \\ & M((p_t, f^{-1}(g))) > 0) \wedge g \in Cl_{SPEC}\big) \\ & \vee\big(p = p_c^0 \wedge g \in Cl_{SPEC}\big), \\ 0 & otherwise. \end{cases} \tag{2}$$

Transition Neighbourhoods of APNs

Definition 23. *Neighbourhood of a transition of a composite step APN net $\mathcal{N}$ is called the set of all the places of the net $\mathcal{N}$ that are directly connected to that transition.*

From $(\star\star\star)$ (see p. 95), it follows that every transition of a composite step APN net $\mathcal{N}$ belongs to either emptying, retrieve or step transitions.

Lemma 4. *Every emptying transition of a composite step APN net $\mathcal{N}$ has a set of pre- and post-places that consists of a control place and a database place and one response place of the same basic step net of $\mathcal{N}$ respectively.*

Proof. From basic step net and composite step nets (definitions 6 and 13 respectively), it follows that $\forall t \in T_{EM}$ we have got $Pre(t) = \{p_c, p_d\}$ and $Post(t) = \{p_r\}$. $\square$

An example of the above case is transition t_5 and places p_7, p_8 and p_{10} of Figure 6.1.

Lemma 5. *Every retrieve transition of a composite step APN net $\mathcal{N}$ has a set of pre- and post-places that consists of a response place and a database place of the same basic step net of $\mathcal{N}$ respectively.*

Proof. From basic step net and composite step nets (definitions 6 and 13 respectively), it follows that $\forall t \in T_R$ we have got $Pre(t) = \{p_r\}$ and $Post(t) = \{p_d\}$. $\square$

An example of that case is transition t_6 and places p_{10} and p_8 of Figure 6.1.

Lemma 6. *Every step transition of a composite step APN net $\mathcal{N}$ has a set of pre- and post-places that consists of a step (starting) place and a step (finishing) place and a control place of the same basic step net of $\mathcal{N}$ respectively. In addition, every step transition has a special set of 'pre-places' that consists of all the possible inhibitor places of that transition.*

Proof. From basic step net and composite step nets (definitions 6 and 13 respectively), it follows that $\forall t \in T_{STEP}$ we have got $Pre(t) = \{p_s\}$ and $Post(t) = \{p_f, p_c\}$ with p_s and p_f being the starting and finishing step places of the basic step net where the step transition belongs to. Furthermore, from definitions 8 and 12 (compositions), it turns that inhibitor arcs are added connecting the step transition with one or more inhibitor places depending on the compositions that have been applied. $\square$

For example, in Figure 6.1, p_2 is the only pre-place, p_3 and p_{14} are the inhibitor places and p_6 and p_7 are the post-places of transition t_4 respectively.

Transition Neighbourhoods of T-APNs

From $(\star\star\star)$ and definition 21(ii) it follows that all the transitions of $\varphi(\mathcal{N})$ also belong to either the retrieve, the emptying or the step transitions.

Lemma 7. *Every emptying transition of a T-APN net $\varphi(\mathcal{N})$ that is associated with a composite step APN net $\mathcal{N}$ has a set of pre- and post-places that consists of the control place and a database place and the control place and a response place of that transition respectively.*

Proof. From basic step net (definition 6), composite step nets (definition 13), construction (definition 21) and structural equivalence of the two nets, i.e., $\mathcal{N}$ and $\varphi(\mathcal{N})$, it follows that an emptying transition has as pre-places the control place and a database place of the respective basic step net, as it happens in $\mathcal{N}$. Regarding the post-places of that emptying transition, it follows that they consist of the control place and a response place of the same basic step net. □

For example, p_7, p_8 and p_7, p_{10} are the pre- and post-places of the emptying transition t_5 respectively, as is shown in Figure 6.2.

Lemma 8. *Every retrieve transition of a T-APN net $\varphi(\mathcal{N})$ that is associated with a composite step APN net $\mathcal{N}$ has a set of pre- and post-places that consists of a response place and a database place of that transition respectively.*

Proof. From basic step net (definition 6), composite step nets (definition 13), construction (definition 21) and structural equivalence of the two nets, i.e., $\mathcal{N}$ and $\varphi(\mathcal{N})$, it follows that a retrieve transition has as only pre-place a response place of the associated basic step net of $\mathcal{N}$. Regarding the post-places of that emptying transition, it follows that they consist of only one database place, which is associated to the respective database place of $\mathcal{N}$. □

An example of the above case is retrieve transition t_6 and places p_{10} and p_8 of Figure 6.2.

Definition 24. *The starting place of a bidirectional step* (p_{s_b}) *is defined as the step place of the bidirectional step that has the most control places in total before and after the application of the backward composition.*

Lemma 9. *Every step transition of* $\varphi(\mathcal{N})$ *has a set of pre-places that consists of a step place and one or more possible 'control places' (inhibitor places in* $\mathcal{N}$*) of that transition, depending on whether that transition and its pre step place belong to a bidirectional step or not. Furthermore, the set of post-places of that step transition contains a step place and one or more control places depending on whether the post step place of that transition belongs to a bidirectional step or not, regardless of the transition belonging to that bidirectional step or not.*

Proof. For the proof of the pre-places of a step transition, we examine two cases. Thus, from definitions 12, 21 and ($\star\star$), we get the following:

First pre case: If the pre step place of the step transition is a starting place of a bidirectional step (p_{s_b})[1] and the step transition belongs to a bidirectional step, then the set of pre-places includes more than one control place. For example, in Figure 6.2, p_3 and p_{14} are the two pre control places of transitions t_4. The bidirectional step in this example includes transitions t_4 and t_9, and the pre step place in question is p_2.

Second pre case: If the pre step place of the step transition is not a starting place of a bidirectional step (p_{s_b}) and the step transition either belongs to a bidirectional step or not, then the set of pre-places includes only one control place. For

[1]The starting place (p_s) of a composite step net should not be confused with the starting place of a bidirectional step. However, the starting place (p_s) of one of the composite step nets which participate in the backward composition always coincides with the starting place of the bidirectional step.

instance, in Figure 6.2, p_0 and p_7 are the only pre control places of transitions t_1 and t_9 respectively. The bidirectional step of this example is the same as that of the first pre case, and the pre-places in question are p_1 and p_6 respectively.

For the proof of the post-places of a step transition, two cases are also examined. From definitions 6, 8, 12 and 21, we get the following:

First post case: If the post step place of the step transition is a starting place of a bidirectional step (p_{s_b}) and the transition either belongs to a bidirectional step or not, then the set of post-places includes more than one control place. For example, in Figure 6.2, p_3 and p_{14} are the two post control places of transitions t_9 and t_1 respectively. The bidirectional step in this example includes transitions t_4 and t_9, and the post step place in question is p_2.

Second post case: If the post step place of the step transition is not a starting place of a bidirectional step (p_{s_b}) and the transition either belongs to a bidirectional step or not, then the set of post-places includes only one control place. For instance, in Figure 6.2, p_7 is the only post control place of transition t_4. The bidirectional step of this example is the same as that of the first case, and the post-place in question is p_6.

□

Theorem Proof

In this section, we provide the proof of theorem 1 in chapter 6.

Proof. We need to prove that:

(i) ψ is a bijection,

(ii) $\psi(M_0) = M'_0$,

(iii) $\forall M_1, M_2 \in [M_0\rangle_{\mathcal{N}}, \forall U \in 2^{\widetilde{T}} : M_1[U\rangle_{\mathcal{N}} M_2 \Longleftrightarrow \psi(M_1)[U\rangle_{\varphi(\mathcal{N})}\psi(M_2)$.

To show the above, it is enough to prove the following:

(A) $\psi(M_0) = M_0'$,

(B) $\forall M_1, M_2 \in [M_0\rangle_{\mathcal{N}}, \forall U \in 2^{\widetilde{T}} : M_1[U\rangle_{\mathcal{N}} M_2 \Rightarrow \psi(M_1)[U\rangle_{\varphi(\mathcal{N})}\psi(M_2)$.

(C) $\forall M_1, M_2 \in [M_0\rangle_{\mathcal{N}}, \forall U \in 2^{\widetilde{T}} : \psi(M_1)[U\rangle_{\varphi(\mathcal{N})}\psi(M_2) \Rightarrow M_1[U\rangle_{\mathcal{N}} M_2$.

Point (A) follows from the construction (definition 21(xi)).

Now we prove point (B):

Let $U = \{(t_1, c_1), \dots, (t_n, c_n)\} \in 2^{\widetilde{T}}$. Let's assume that we can go from M_1 to M_2 through U in the CRG of the composite step APN net $\mathcal{N}$. So U is enabled at M_1 and M_2 is reachable from M_1 by executing U, which is denoted by $M_1[U\rangle_{\mathcal{N}} M_2$.

U is enabled at M_1 for the composite step APN net means that $\widetilde{Pre}(U) \leq M_1$ and $M_1 + \widetilde{Post}(U) \leq K$ hold. $\widetilde{Pre}(U) \leq M_1$ implies that the pre-places of the transitions of the enabled step U host adequate number and proper colour of tokens for this step to fire. Furthermore, from $M_1 + \widetilde{Post}(U) \leq K$, it follows that the number of tokens of the post-places of the transitions of the enabled step U will not exceed their capacities when depositing tokens after the firing of U.

From APN theory we know that an enabled step U is a set of enabled structured transitions and that a composite step APN net has a specific structure regarding the connection of the places and the transitions (see definition 6). From these two, we could say that every transition of the enabled step U has specific pre-places, which host adequate number and proper colour of tokens for the firing of that particular transition since it belongs to the enabled step U. Thus, from lemmas 4, 5 and 6, it follows that every transition has a specific neighbourhood of pre-, post-places and inhibitor places.

Applying definition 21, we construct a T-APN net $\varphi(\mathcal{N})$ that is associated with the composite step APN net $\mathcal{N}$ and these two nets are *structurally equivalent*, which means

> that $\mathcal{N}$ and $\varphi(\mathcal{N})$ are almost the same, and the only structural difference lies in the additional direct arcs of $\varphi(\mathcal{N})$ to compensate the lack of colour-sensitive inhibitor arcs. By the construction, for every place and transition of $\mathcal{N}$ there exists the respective place and transition in $\varphi(\mathcal{N})$ (*i.e.* $P' = P$ and $T' = T$). Furthermore, for every pre and post direct arc of $\mathcal{N}$, there exists a respective direct arc in the same direction in $\varphi(\mathcal{N})$. Moreover, every inhibitor arc of $\mathcal{N}$ is replaced by a direct arc and new direct arcs are also added in $\varphi(\mathcal{N})$ through the construction (definition 21(iii)).

The two nets being structurally equivalent results in the conclusion that the pre- and post-places of the transitions of step U in $\varphi(\mathcal{N})$ consist of the related pre- and post-places of the respective transitions of the enabled step U in $\mathcal{N}$ plus the 'new' pre- and post-places that are 'created' by the replacement of the inhibitor arcs and the addition of the new direct arcs (definition 21(iii)).

The neighbourhoods of the transitions of $\varphi(\mathcal{N})$ with respect to the neighbourhoods of the respective transitions of $\mathcal{N}$ are presented in lemmas 7, 8 and 9 and depend on which set the examined transitions of $\varphi(\mathcal{N})$ belong to.

From the theories of both APN and T-APN nets it follows that nets $\mathcal{N}$ and $\varphi(\mathcal{N})$ are colour safe, which means that they allow at most one token of a given colour per place. Also, from the construction it follows that the transitions of the two nets work under the same colour mode, i.e. $C_T(t) = C(t)$ (definition 21(vii)) and that all the places of $\varphi(\mathcal{N})$ retain the same colours of tokens as those of the respective places of $\mathcal{N}$ except for the control places that can carry both identity and special colours of tokens, as described in definition 21(vi).

Finally from the construction it follows that the capacities of the associated places of the two nets are the same for tokens with identity colours and 1 for tokens with special colours (definition 21(ix)). This means that the maximum number of identity tokens that could reside in the places of $\varphi(\mathcal{N})$ is the same as that in $\mathcal{N}$ and that the number of special tokens that could reside in the appropriate places is 1 per colour per place. It also follows from the construction that the guard function used on the T_G transitions of the T-APN net depends on the set of *binding pairs* which is built through the bijective function f (definition 21(x)). This means that the guard evaluates to true for the pairs of colours that belong to the set of *binding pairs.* If tokens of these pairs of colours reside in the pre-places of a transition of $\varphi(\mathcal{N})$, then they can be consumed as a pair (one coloured token from each pre-place). On the contrary, the guard of a transition that belongs to the T_{NG} transitions is always true, regardless of the colours of tokens that reside in the pre-places of the transition (definition 21(x)).

After this preliminary discussion, we have to show that for the associated marking of M_1, i.e. $\psi(M_1)$, the step U is enabled in $\varphi(\mathcal{N})$ and that the reachable marking after the firing of U is the associated marking of M_2, the $\psi(M_2)$.

To prove that U is enabled at $\psi(M_1)$ for the T-APN net $\varphi(\mathcal{N})$, we have to examine whether the following conditions hold:

- $\widehat{Pre}(U) \leq \psi(M_1)$
- $\psi(M_1) + \widehat{Post}(U) \leq K'$
- $\forall\ (t_i, c_i) \in U,\ with\ i = 1, \ldots, n,\ Gd((t_i, c_i)) = true.$

We now investigate whether the above conditions hold for a singleton structured transition $(t, c) \in U$. As has already been mentioned, the structured transition (t, c) can be either a retrieve, emptying or step transition (see $(\star\star\star)$). Consequently, we examine these three different cases for the enabledness of (t, c).

Case 1: The structured transition (t, c) is a step transition, i.e., $t \in T_{STEP}$ in $\mathcal{N}$.

We know from our assumption that step U is enabled at M_1. This means that all the transitions of U are enabled at M_1, which in its turn implies that all the step transitions of U are enabled at M_1 as well and that each of the pre-places and the inhibitor places of these transitions carry the appropriate colour and number of tokens. We also know from the above discussion that $\mathcal{N}$ has a specific structure since it is a composite step APN net and that every step transition of a composite step APN net has specific pre-places and at least one inhibitor place. As (t, c) is enabled in $\mathcal{N}$, this means that the pre step place of t carries a token of colour $g \in Cl_{ID}$, which should coincide with the colour mode of t, i.e. $g = c$ as $\widetilde{Pre}(U) \leq M_1$ in $\mathcal{N}$. The number of tokens in the step place derives from the fact that t is enabled, so the step place should have at least one token and the fact that APN nets are colour safe nets, which means that only one token of a specific colour can be hosted in each place of the net. Since the transition is enabled, consequently we know that the inhibitor places of t hold no tokens of colour c $((U \times M) \cap \widetilde{I} = \varnothing)$.

Now, we consider the constructed T-APN net $\varphi(\mathcal{N})$ that is associated with $\mathcal{N}$. From what has been mentioned above, we know that $\varphi(\mathcal{N})$ is structurally equivalent to $\mathcal{N}$, so for every place, transition and direct arc of $\mathcal{N}$ there exists a respective one in $\varphi(\mathcal{N})$. Furthermore, we know that every inhibitor arc is replaced by a direct arc converting the inhibitor place to a control pre-place (definition 21(iii)). Replacing the inhibitor arcs by direct ones implies that in order to fire the associated transition t of $\varphi(\mathcal{N})$ in mode c, a token must be consumed from both the associated places of the inhibitor places and the associated place of the step place of $\mathcal{N}$. The associated places of the inhibitor places of $\mathcal{N}$ are structured using both identity and special colours and all the other places of $\varphi(\mathcal{N})$ are structured in the same way as in $\mathcal{N}$ according to the construction (definition 21$(viii), (xi)$) and the definition of the $\widehat{Post}$ mapping (see p. 87). In addition, all the new direct arcs in $\varphi(\mathcal{N})$, and only them, carry special colours and are connected to control places (definition 21$(viii)$). As $c \in Cl_{ID}$, we know that $f(c) \in Cl_{SPEC}$, which implies that the associated places of the inhibitor ones can potentially take a token of colour $f(c)$ in $\varphi(\mathcal{N})$ (definition of $\widehat{Post}$ mapping and definition 21$(viii), (xi)$). Furthermore, if $a = (p, t)$ is a replacement arc in $\varphi(\mathcal{N})$ of an inhibitor arc that was used in $\mathcal{N}$ to control the behaviour of (t, c), then we have, from the construction (definition 21$(viii)$ and (x)) that $f(c) \in Ins(a)$ and $(c, f(c)) \in B_t$ (and in consequence $Gd((t, c)) = true$).

Now, in order to prove the first point for the transition of (t, c) in $\varphi(\mathcal{N})$, we have to calculate the marking $\psi(M_1)$ for the pre-places of (t, c) in $\varphi(\mathcal{N})$ using the fact that (t, c) is enabled at M_1 in $\mathcal{N}$, and check if the calculated marking is enough for the firing of the structured transition (t, c) in $\varphi(\mathcal{N})$. To calculate the marking $\psi(M_1)$, we use (1) and (2). In this case, from (1), the first and third point of (2) and the neighbourhoods of the step transitions (lemmas 6 and 9), it follows that we retain the token of colour c for the pre step place of (t, c) in $\varphi(\mathcal{N})$ and we get a token of colour $f(c)$ for every associated place of the inhibitor places in $\varphi(\mathcal{N})$ instead of being empty of tokens of colour c as it happens with the inhibitor places of (t, c) in $\mathcal{N}$. In addition, as is said in the preliminary discussion, arcs joining pre-places of t with t have appropriate inscriptions that enable them to carry the tokens from all the pre-places: the arcs that are not changed by the transformation are inscribed with identity colours, while the arcs that replace the inhibitor arcs are inscribed with special colours (definition 21$(viii)$). Hence, from (†) (see p. 86), we have $\widehat{Pre}(t, c) \leq \psi(M_1)$, for $t \in T_{STEP}$.

For the proof of the second point, we use the differences between the neighbourhoods of (t, c) in $\mathcal{N}$ and (t, c) in $\varphi(\mathcal{N})$ (lemmas 6 and 9). Since (t, c) in $\varphi(\mathcal{N})$ is the associated step transition of (t, c) in $\mathcal{N}$ (definition 21(ii),(vii)), we know that the post-places of (t, c) in $\varphi(\mathcal{N})$ are the 'same' as for (t, c) in $\mathcal{N}$ with possibly some additional post-places (e.g. transition t_9 and places p_2, p_{14} in Figure 6.1 and transition t_9 and places p_2, p_{14} and p_3 in Figure 6.2 respectively). So, we investigate the marking of the post-places of (t, c) in $\varphi(\mathcal{N})$ in relation to the marking of the respective places in $\mathcal{N}$, which means that the marking of these places should be examined both before and after the firing of the respective transitions. As (t, c) in $\mathcal{N}$ is enabled at M_1, it follows that $M_1 + \widetilde{Post}(U) \leq K$ is satisfied. This implies that the capacity of the post-places of (t, c) in $\mathcal{N}$ is not violated after the firing of the transition in mode c and that these post-places do not host any token of colour c before the firing since $\mathcal{N}$ is colour safe. Now, using the marking relation for the 'same' post-places of (t, c) in $\varphi(\mathcal{N})$ that consist of a step place and a control place, it turns from the first point of (2) that these post-places do not carry any token of colour c (as $c \in Cl_{ID}$) and none of the post-places is the root control place (p_c^0) of the 'starting place' of $\varphi(\mathcal{N})$ (e.g. post-place p_2 and p_{14} of transition t_9 in Figure 6.2). Regarding the additional post-places of (t, c) in $\varphi(\mathcal{N})$, we know from definition 21$(iii), (viii)$ that they are linked with the transition through 'new' direct post arcs that carry special tokens. This means that after the firing of (t, c) in $\varphi(\mathcal{N})$, they should host at least one token of colour $f(c)$ according to (††) (assuming enabledness). But, to ensure that the capacity of the additional post-places will not be violated after the firing of (t, c) in $\varphi(\mathcal{N})$, we examine whether their marking already includes tokens of colour $f(c)$ before the firing of the transition. From the neighbourhood of (t, c) in $\varphi(\mathcal{N})$ and definition 21(iii), it follows that these additional post-places are control places (e.g. post-place p_3 of transition t_9 in Figure 6.2). We also know from the neighbourhood of (t, c) in $\mathcal{N}$ (lemma 6) and definition 21(iii) that the respective places in $\mathcal{N}$ are neither pre-places nor post-places of (t, c) in $\mathcal{N}$ as there are no arcs between the places and the transition (e.g. place p_3 and transition t_9 in Figure 6.1). From the root net (definition 7), the compositions of the composite APN net $\mathcal{N}$ (definitions 8 and 12), the colour safeness of $\mathcal{N}$ and the special structure of the composite step nets, it follows that only one step place and its control inhibitor place of the same composite step net, out of all the step places and control inhibitor places of $\mathcal{N}$ can host a token of colour c each time. This results from the fact the different colours of tokens of $\mathcal{N}$ are used as unique IDs. This implies that these places, in $\mathcal{N}$, do not carry any token of colour c as (t, c) is enabled in mode c at M_1, so the token of colour c is occupied by pre step place of the enabled step transition of $\mathcal{N}$. Consequently, from (2) and the proof of the first point for (t, c) in $\varphi(\mathcal{N})$ $(\widehat{Pre}(t, c) \leq \psi(M_1))$, it follows that the additional post control places do not host any tokens of colour c or $f(c)$ before the firing of the transition. This can be explained by the structural equivalence of the two nets and by showing how the colour safeness of $\mathcal{N}$ is 'passed' to the net $\varphi(\mathcal{N})$. From the fact that $\mathcal{N}$ is colour-safe, which implies that at most one token of a given colour is allowed per place and the fact that the marking relation (2) gives always 0 or 1 for the colours of tokens of $\varphi(\mathcal{N})$, considering as input of that function the marking of the respective places of $\mathcal{N}$, it results that $\varphi(\mathcal{N})$ also allows at most one token of a given colour (identity or special) per place. Thus $\varphi(\mathcal{N})$ is colour-safe. Having shown that none of the post-places of (t, c) in $\varphi(\mathcal{N})$ carries tokens of either colour c of $f(c)$ before the firing of the transition, we continue with the examination of the marking of the post-places after the firing of (t, c) in $\varphi(\mathcal{N})$. Knowing that (t, c) in $\mathcal{N}$ has adequate resources for its firing as $\widetilde{Pre}(U) \leq M_1$ and according to

$(*)$ (see p. 38), it follows that a token of colour c is consumed from the pre step place of the transition and that the inhibitor place of (t, c) in $\mathcal{N}$ has no token of colour c. Having also proved, in the previous point, that all the pre-places of (t, c) in $\varphi(\mathcal{N})$ have adequate resources for the firing of the transition, from the first and third point of (2) and (†) (see p. 86), it follows that a token of colour c is consumed from the pre step place of (t, c) in $\varphi(\mathcal{N})$ and that a token of colour $f(c)$ is consumed from the control places (associated inhibitor places in $\mathcal{N}$) for the firing of (t, c) in $\varphi(\mathcal{N})$ respectively (assuming enabledness). After the firing of (t, c) in $\mathcal{N}$, a token of colour c is deposited to every post-place of the transition ($w.r.t$ $(**)$ in p. 38). From the first point of (2) and (††) (see p. 87), it follows that if (t, c) in $\varphi(\mathcal{N})$ fires, the 'same' post-places of the transition can host tokens of colour c. On the contrary, the additional post control places of (t, c) in $\varphi(\mathcal{N})$ can host tokens of colour $f(c)$ (point three of (2) and (††)). Due to the fact that all the above post-place of (t, c) in $\varphi(\mathcal{N})$ had no tokens of colour c or $f(c)$ before the firing of the transition and the fact that $\varphi(\mathcal{N})$ 'inherited' the colour safeness from $\mathcal{N}$, it follows that the firing of (t, c) deposits only one token of colour c into each of the 'same' post-places and only one token of colour $f(c)$ into each of the additional post-places of $\varphi(\mathcal{N})$ ($w.r.t$ (††) in p. 87). Moreover, from definition 21(ix), it follows that the capacities of the associated places are linked. So, according to definition 21(ix), the capacity of the 'same' post-places of $\varphi(\mathcal{N})$ is equal to the capacity of the respective places of $\mathcal{N}$ and the capacity of the additional post-places the equal to 1 per colour per place. Consequently, in both cases the capacity of the post-places of (t, c) in $\varphi(\mathcal{N})$ is not violated. In the first case, this derives from the fact that the capacity of the respective places of $\mathcal{N}$ is not violated as (t, c) is enabled at M_1, the fact that the 'same' post-places have the same capacities in both nets and the fact that the firing of (t, c) in $\varphi(\mathcal{N})$

deposits only one token of colour c in the post-place of the transition. On the other hand, in the second case the capacity of the additional post-places of (t, c) in $\varphi(\mathcal{N})$ is not violated since it is equal to 1 for each place and according to (††) only one token of colour $f(c)$ is deposited to the additional post-places. Hence, from all the above, it follows that the second point is satisfied for $t \in T_{STEP}$.

Finally, for the third point, the guard of (t, c) in $\varphi(\mathcal{N})$ evaluates to true since the tokens that are deposited into the pre-places belong to the set of *binding pairs* and the colours of these tokens comply with the arc inscriptions as was described in the previous paragraphs.

Case 2: The structured transition (t, c) is a retrieve transition, i.e., $t \in T_R$ in $\mathcal{N}$.

From definition 21(ii),(vii), it follows that (t, c) in $\varphi(\mathcal{N})$ is a retrieve transition that works in mode c, as it happens with (t, c) in $\mathcal{N}$. From the neighbourhoods of (t, c) in $\mathcal{N}$ and (t, c) in $\varphi(\mathcal{N})$ (lemmas 5 and 8 respectively) and the structural equivalence of the two nets, it follows that these two transitions have the 'same' pre- and post-places, i.e a response place and a database place respectively. We know that (t, c) in $\mathcal{N}$ is enabled at M_1, so its pre response place carries a token of colour c, as $g = c$ from the enabledness of (t, c) in $\mathcal{N}$. From the first point of (2), it follows that the only pre response place of (t, c) in $\varphi(\mathcal{N})$ hosts a token of colour $c \in Cl_{ID}$ as well. Hence, from (†) and definition 21$(iii), (viii)$, we have that the first point of the enabledness is satisfied for $t \in T_R$ in $\varphi(\mathcal{N})$.

For the second point, we first check the marking of the post-places of (t, c) in $\varphi(\mathcal{N})$ in relation to the marking of the respective places in $\mathcal{N}$ before the firing of (t, c) in $\varphi(\mathcal{N})$. Afterwards, we examine whether a possible firing of (t, c) in $\varphi(\mathcal{N})$ violates the capacity of its post-places after depositing the respective tokens into these places. As was mentioned earlier, (t, c) in $\mathcal{N}$ and (t, c) in $\varphi(\mathcal{N})$ have the 'same' neighbourhoods (lemmas 5 and 8) and, by extension, the 'same' post-places, i.e., database places. From definition 21(vi), it follows that the post database place of (t, c) in $\varphi(\mathcal{N})$ is structured with the same colours of tokens as those of the respective database place in $\mathcal{N}$. Checking the marking of that post-place before the firing of (t, c) in $\varphi(\mathcal{N})$, from the first point of (2), it follows that it has no tokens of colour c as the respective database place of (t, c) in $\mathcal{N}$ holds no token of colour c. At this point, we examine the marking of the database place of (t, c) in $\varphi(\mathcal{N})$ after the firing of the transition. From the enabledness of (t, c) in $\mathcal{N}$ and $(*)$, it follows that a tokens of colour c is consumed from the pre-place of the transition during its execution. For (t, c) in $\varphi(\mathcal{N})$, from the proof of the first point, (†) and definition 21$(viii)$, it follows that a token of colour c is consumed from the pre response place of the transition during its firing (assuming enabledness). From $(**)$ and colour safeness of $\mathcal{N}$, we know that after the firing of (t, c) in $\mathcal{N}$, a token of colour c is deposited to the post database place. In addition, from (††), it follows that after the firing of (t, c) in $\varphi(\mathcal{N})$ at least one token of colour c is deposited in the post database place of the transition. But, from the first point of (2) and the fact that $\varphi(\mathcal{N})$ 'inherits' the colour safeness from $\mathcal{N}$ through (2) (as was discussed in ***Case 1***), it turns that only one token of colour c is deposited into the post-place of (t, c) in $\varphi(\mathcal{N})$. Furthermore, we know that $M_1 + \widetilde{Post}(U) \leq K$ is satisfied since (t, c) is enabled in $\mathcal{N}$, so capacity is not violated. Moreover, from definition 21(ix) and colour safeness of $\varphi(\mathcal{N})$ and as $c \in Cl_{ID}$, it follows that since the capacities of the associated database places are equal and the post database place of (t, c) in $\varphi(\mathcal{N})$ hosts only one token of colour c, the capacity of the database place is not violated proving in that way that the second point is also satisfied for $t \in T_R$ in $\varphi(\mathcal{N})$.

Finally, for the third point of the enabledness, it follows from the guards of T-APNs (definition 5.1) and definition 21$(ii),(x)$ that every retrieve transition of $\varphi(\mathcal{N})$ belongs to the set of transitions T_{NG}, which implies that the guard is always true for (t,c) in $\varphi(\mathcal{N})$. Therefore, the retrieve transition (t,c) in $\varphi(\mathcal{N})$ is enabled as the third point is also satisfied.

Case 3: The structured transition (t,c) is an emptying transition, i.e., $t \in T_{EM}$ in $\mathcal{N}$.

From definition 21$(ii),(vii)$, it follows that (t,c) in $\varphi(\mathcal{N})$ is an emptying transition that works in mode c, as it happens with (t,c) in $\mathcal{N}$. From the neighbourhoods of (t,c) in $\mathcal{N}$ and (t,c) in $\varphi(\mathcal{N})$ (lemmas 4 and 7), it follows that these two transitions have the 'same' pre-places, i.e control place and database place, but slightly different post-places. Actually, they have the 'same' post response place, but the pre control place of (t,c) in $\varphi(\mathcal{N})$ is also post-place of the transition. We know that (t,c) is enabled at M_1 in $\mathcal{N}$, so the pre database and control places of (t,c) in $\mathcal{N}$ carry a token of colour c each, with $g = c$. From the first point of (2), it follows that the pre database and control places of (t,c) in $\varphi(\mathcal{N})$ also host a token of colour c each, since $c \in Cl_{ID}$. Hence, from (†) and definition 21$(iii),(viii)$, it follows that the first point of the enabledness is satisfied for the structured transition (t,c) in $\varphi(\mathcal{N})$.

For the second point, we first check the marking of the post-places of (t,c) in $\varphi(\mathcal{N})$ in relation to the marking of the respective places in $\mathcal{N}$ before the firing of (t,c) in $\varphi(\mathcal{N})$. Afterwards, we examine whether a possible firing of (t,c) in $\varphi(\mathcal{N})$ violates the capacity of its post-places after depositing the respective tokens into these places. As was mentioned above, (t,c) in $\mathcal{N}$ and (t,c) in $\varphi(\mathcal{N})$ have 'same' post-places (i.e., response places), but (t,c) in $\varphi(\mathcal{N})$ has an additional post-place (i.e., the pre control place) as is described in the proof of the first point. Regarding the 'same' post-places of the two nets, from definition 21(vi), it follows that the post response place of (t, c) in $\varphi(\mathcal{N})$ is structured with the same colours of tokens as those of the respective response place in $\mathcal{N}$. Therefore, checking the marking of that post-place before the firing of (t, c) in $\varphi(\mathcal{N})$, from the first point of (2), it follows that it has no tokens of colour c since the respective response place of (t, c) in $\mathcal{N}$ holds no token of colour c. Examining now the marking of the additional post place before the firing of the transition, from the first point of (2), it follows that the marking of the additional post control place before the firing of (t, c) in $\varphi(\mathcal{N})$ contains no tokens of colour $f(c)$ as that place is treated as pre-place of the transition, which already holds a token of colour $c \in Cl_{ID}$. At this point, we examine the marking of the 'same' post-place of (t, c) in $\varphi(\mathcal{N})$ after the firing of the transition. From the enabledness of (t, c) in $\mathcal{N}$ and $(*)$, it follows that a tokens of colour c is consumed from the pre-place of the transition during its execution. Similarly for (t, c) in $\varphi(\mathcal{N})$, from the proof of the first point, $(\dagger)$ and definition 21$(viii)$, it follows that a token of colour c is consumed from both the control place and database place of the transition during its firing (assuming enabledness). From $(**)$ and colour safeness of $\mathcal{N}$, we know that after the firing of (t, c) in $\mathcal{N}$, a token of colour c is deposited to the post response place. In addition, from $(\dagger\dagger)$, it follows that after the firing of (t, c) in $\varphi(\mathcal{N})$ at least one token of colour c and $f(c)$ is deposited in the post response place and the post control place of the transition respectively. But, from the first and third point of (2) and the fact that $\varphi(\mathcal{N})$ 'inherits' the colour safeness from $\mathcal{N}$ through (2), it turns that only one token of colour c and $f(c)$ is deposited in the respective post-places of (t, c) in $\varphi(\mathcal{N})$. Furthermore, we know that $M_1 + \widetilde{Post}(U) \leq K$ is satisfied since (t, c) is enabled in $\mathcal{N}$, so capacity is not violated. Moreover, as $c \in Cl_{ID}$, from definition 21(ix) and colour safeness of $\varphi(\mathcal{N})$, it follows that since the capacities of the associated response places are the same and the post response place of (t, c) in $\varphi(\mathcal{N})$ hosts only one token of colour c, the capacity of the response place is not violated. Moreover, from definition 21(ix) and colour safeness of $\varphi(\mathcal{N})$, it follows that the capacity of the post control place of (t, c) in $\varphi(\mathcal{N})$ is equal to 1 for every special colour and that this place hosts only one token of colour $f(c)$ after the firing of the transition, which means that the capacity is not violated. Therefore, it has been shown that the second point is also satisfied for $t \in T_{EM}$ in $\varphi(\mathcal{N})$.

Finally, as (t, c) in $\varphi(\mathcal{N})$ is an emptying transition, from the guards of T-APNs (definition 5.1) and definition 21(ii), (x), it follows that every emptying transition of $\varphi(\mathcal{N})$ belongs to the set of transitions T_{NG}, which implies that the guard of (t, c) is always true. Consequently, the emptying transition (t, c) in $\varphi(\mathcal{N})$ is enabled since the third point is always satisfied.

From what is mentioned in the three above cases, it can be concluded that the associated transition of (t, c) in $\mathcal{N}$, which is enabled at M_1, is also enabled at the associated marking $\psi(M_1)$ in $\varphi(\mathcal{N})$. From the enabledness and firing of step U in $\mathcal{N}$, it follows that the transitions of U do not consume shared resources and do not deposit tokens of the same colours to the same post-places violating the capacity and the colour safeness of $\mathcal{N}$. Thus, from the assumption that U is enabled at M_1 in $\mathcal{N}$, it can be deduced that the transitions of that step can be executed both individually and concurrently as all the enabledness conditions of U in $\mathcal{N}$ are satisfied. Based on that, it has been shown that a transition of step U in $\varphi(\mathcal{N})$ is enabled and can be executed at $\psi(M_1)$ individually, as is described by the three above cases. As is known, from the construction (definition 21(ii),(vii)) and, by extension, from the structural equivalence, the step U in $\varphi(\mathcal{N})$ consists of the respective transitions of U in $\mathcal{N}$, which implies that these transitions have the same or similar neighbourhoods to those in $\mathcal{N}$ and work under the same colour modes. At this point, it is examined whether these transitions can be executed concurrently considering the fact that this can be done in $\mathcal{N}$.

To prove the concurrent firing of the transitions of U in $\varphi(\mathcal{N})$, we have to show that these transitions are not in conflict (i.e., share common resources) and that there is no violation of the capacity after depositing of tokens into the post-places of the transitions.

We start the proof of the concurrent execution of the transitions of U in $\varphi(\mathcal{N})$ by examining the resources of the pre-places of these transitions. From the specific structure of the composite step nets and the compositions of the APNs (see definitions 8 and 12), it follows that two or more transitions can be in conflict (see definition 4) only when they share a pre step place and its respective inhibitor places or a control place. Hence, the transitions of U in $\mathcal{N}$ that can be in conflict, sharing resources, are only the step transitions and the emptying transitions of the step U. This means that there exist two cases for all these places: either they are shared by transitions of U in $\mathcal{N}$ or not. As all the transitions of U in $\mathcal{N}$ can fire concurrently, this implies that either they do not share pre-places and inhibitor places or they have places in common but without sharing the resources of those places (i.e., tokens). For example, two transitions that share pre-places, inhibitor places or both can fire concurrently if they fire in different colour modes (i.e., consume different colours of tokens or expect that tokens of certain colours are not present there).

Not sharing pre-places: Now, let's examine the case where the transitions of U in $\mathcal{N}$ do not share pre-places or inhibitor places. From the construction (definition 21) and the structural equivalence of the nets, it results that the transitions of U in $\varphi(\mathcal{N})$ host the adequate resources and can fire concurrently as they are not in conflict with each other (assuming enabledness). In fact, the concurrent firing of these transitions, in this case, is based on the individual firing that was examined in the three above cases.

Sharing pre-places: In the second case, the transitions of U in $\mathcal{N}$ fire concurrently while sharing pre-places, inhibitor places or both, which implies that these transitions do not consume tokens of the same colours. This results from the fact that $\mathcal{N}$ is colour safe, which means that only one token of each colour can reside in each of the shared pre-places or can be consumed during the firing of the transitions. From the colour safeness of $\mathcal{N}$, it follows that all the transitions of U that fire concurrently and share common places work under different colour modes and consume the tokens that correspond to their colour modes when they fire.

From the neighbourhoods of transitions and points one and three of (2), it follows that for every empty inhibitor place, the respective control place in $\varphi(\mathcal{N})$ hosts the appropriate colour and number of tokens (i.e., one token of each permitted special colour) and that for every shared pre-place in $\mathcal{N}$, the respective shared pre-place in $\varphi(\mathcal{N})$ carries the appropriate colour and number of tokens. Actually, this derives from the colour safeness of $\mathcal{N}$, which is passed to $\varphi(\mathcal{N})$ through definition 21(xi) and (2) as they define that only zero or one token of each (identity or special) colour can be hosted in the places of $\varphi(\mathcal{N})$. Hence, from the structural equivalence, (2) and (†), it turns that the common pre-places of the transitions of U in $\varphi(\mathcal{N})$ carry the adequate number and the proper colour of tokens (i.e., only one token of each colour) and that the respective transitions are enabled under different colour modes.

Having shown that the pre-places of the transitions of U in $\varphi(\mathcal{N})$ carry the adequate resources in order to be enabled for concurrent firing, we examine whether that concurrent firing violates the capacity after depositing of tokens into the post-places of the transitions. Regarding the depositing of tokens to the post-places after the concurrent firing of the transitions of U in $\mathcal{N}$, it follows that the following cases should be examined: the post-places are shared by the transition of U in $\mathcal{N}$ or not.

Not sharing post-places: In this case, the transitions of step U in $\mathcal{N}$ do not share post-places. Thus, after the firing, they deposit the tokens to separate post-places without violating the capacity restriction as U is enabled at M_1 in $\mathcal{N}$ (e.g. transitions t_3 and t_9 and transitions t_2 and t_9 in Figure 6.1). From definitions 8, 12 and 21 and structural equivalence, it follows that since the transitions of U in $\mathcal{N}$ do not share any post-place, then the associated transitions of U in $\varphi(\mathcal{N})$ will not share any old post-places[1] as well, but they might share 'new' post-places[2] (e.g. transitions t_3 and t_9 with no shared post-places (either old post-places or 'new' post-places) and transitions t_2 and t_9 with a shared 'new' post-place in Figure 6.2 respectively). From definition 21(*iii*) and structural equivalence, it follows that the only post-places of the transitions of U in $\varphi(\mathcal{N})$ that might be shared when the associated transitions of U in $\mathcal{N}$ do not share any post-place are the 'new' post-places, namely the 'new' post control places of the transitions. This implies that there exist two cases for the 'new' post-places of the transitions of U in $\varphi(\mathcal{N})$, either they are shared or not.

Not sharing 'new' post control places: If the 'new' post control places are not shared, this means that none of the post-places of the transitions of U in $\varphi(\mathcal{N})$ is shared. Therefore, from (2), the fact at all the transitions of U in $\varphi(\mathcal{N})$ are enabled and can fire individually at $\psi(M_1)$ (as was proved in the

[1]Old post-places refer to the associated post places of $\varphi(\mathcal{N})$ that are 'inherited' from $\mathcal{N}$ though the construction.

[2]'New' post-places refer to the additional post-places that derive from the construction.

three different cases of transitions described earlier) and (††), it follows that the transitions of U in $\varphi(\mathcal{N})$ can fire concurrently depositing the consumed tokens to the respective post-places without violating the capacity (e.g. concurrent firing of transition t_3 and t_9 in Figure 6.2).

Sharing 'new' post control places: If the 'new' post control places are shared by the transitions of U in $\varphi(\mathcal{N})$, we have to examine if the capacity of these shared places is violated when the transitions that share those places fire concurrently (assuming enabledness). From the compositions of $\mathcal{N}$ (definitions 8 and 12) and the construction (definition 21), it follows that the situation of sharing 'new' post control places and not sharing old post-places takes place between step transitions (T_{STEP}) and emptying transitions (T_{EM}) in $\varphi(\mathcal{N})$ (e.g. transitions t_5 and t_7 and transitions t_2 and t_9 in Figure 6.2 respectively).

From definitions 8 and 12, the specific structure of composite step net and definition 21(*iii*), it follows that a step transition and an emptying transition share a 'new' post control place in $\varphi(\mathcal{N})$ only if the post step place of the step transition is the same with the step place that is 'monitored' by the 'new' post control place of the emptying transition with the step place having at least one post-transition (e.g. p_2 is post step place of step transition t_9, is 'monitored' by the 'new' post-place p_3 of emptying transition t_2 and has a post-transition (t_4)). Furthermore, from specific structure of composite step net and definition 21(*iii*), it also follows that the emptying transitions of the same composite step net can share a 'new' post control place (e.g emptying transitions t_5 and t_7 share the 'new' post control place p_7 in Figure 6.2).

From the specific structure of composite step nets, the colour safeness of $\mathcal{N}$ and the presence of the colour-sensitive inhibitor arcs that derives from the application of definitions 8 and 12, it follows that the step transitions and emptying transition in $\mathcal{N}$ cannot fire in the same colour mode concurrently. It also follows, that the emptying transition of the same composite step net in $\mathcal{N}$ cannot fire under the same colour mode concurrently (e.g. assuming enabledness, transitions t_5 and t_7 in Figure 6.1 fire concurrently only under different colour modes).

It can also be concluded that since the transitions in $\mathcal{N}$ fire in different colour mode and the 'new' post-places in $\varphi(\mathcal{N})$ (definition 21(iii)) are not post-places of the fired transitions in $\mathcal{N}$, from definition 21(vii) and (2), it follows that they will not carry any tokens of colours that are the same with those of the colour modes of the fired transitions in $\varphi(\mathcal{N})$ as the transitions of U in $\varphi(\mathcal{N})$ have the same colour modes with the associated transition of U in $\mathcal{N}$. Consequently, from the enabledness of U in $\mathcal{N}$, the structural equivalence and the fact that the colour safeness of $\mathcal{N}$ is 'passed' through definition 21(xi) and (2) to $\varphi(\mathcal{N})$, it follows that since the step and emptying transition of U in $\mathcal{N}$ can fire concurrently under different colour modes, then the respective step and emptying transitions in $\varphi(\mathcal{N})$ can also fire concurrently in the same colour modes (definition 21(vii)) depositing (through (††)) tokens of different colours in the shared 'new' post control places, without violating their capacities.

Thus, since the capacity is not violated for both the old post-places that are not shared and the 'new' post control places that are shared, it turns that in this case the second condition of the enabledness is satisfied for the concurrent firing of the transition of U in $\varphi(\mathcal{N})$.

Sharing post-places: In this case, the transitions of U in $\mathcal{N}$ share the post-places. According to definitions 8, 12 and the specific structure of the composite step nets, the only post-places that can be shared by transitions in $\mathcal{N}$ are the step places (e.g. place p_2 is shared by transitions t_1 and t_9 in Figure 6.1). From the structural equivalence and definition 21(iii), it follows that it turns that the post-places that might be shared by the transitions of U in $\varphi(\mathcal{N})$ during their concurrent firing are the post step places and the 'new' post control places (e.g. transitions t_1 and t_9 share the post-places p_2, p_3 and p_{14} in Figure 6.2). The case of the shared 'new' post control places has been examined above, so the only case that should be examined is that of the shared post step places.

As U is enabled at M_1 in $\mathcal{N}$ and $\mathcal{N}$ is *colour-safe*, it turns that the capacity is not violated after the concurrent firing of the transitions of U in $\mathcal{N}$, which means that the transitions that share post step places deposit tokens of different colours into them (w.r.t $(**)$). From the neighbourhoods of transitions of U of both nets and the structural equivalence of the nets, it follows that the post step places are shared by the associated step transitions of $\mathcal{N}$ and $\varphi(\mathcal{N})$ (e.g. post step place place p_2 is shared by transitions t_1 and t_9 in Figures 6.1 and 6.2 respectively). Since the shared post step places have the same pre-transitions in both nets, from definition 21(vii), it follows that these pre-transitions fire under the same colour modes in both nets. Furthermore, from structural equivalence, definition 21$(vi), (xi)$ and (2), it can be concluded that the shared post step places in $\varphi(\mathcal{N})$ hold no tokens of colours that are the same with the colours of the colour modes of their pre-transitions as happens with the associated post step places in $\mathcal{N}$. Hence, from definition 21$(vii), (ix)$, the structural equivalence of the nets, $(\dagger\dagger)$ and (2), it follows that after the firing of the step transitions of U the shared post step places of these transitions of the step U in $\varphi(\mathcal{N})$ host tokens of different colours and the number of these coloured tokens is equal to 1 per colour per place, satisfying in that way that the capacity of the shared post step places is not violated. So, $\psi(M_1) + \widehat{Post}(U) \leq K'$ holds as it is satisfied for both the shared post step places and the shared 'new' post control places.

As a consequence, from what is mentioned in the above cases, it turns that the transitions of U in $\varphi(\mathcal{N})$ can be executed concurrently. This follows from the fact that there exist the adequate resources for their firing and that the capacity is not violated after the firing of U in $\varphi(\mathcal{N})$.

Now, the reachable markings of the two nets are calculated using the formulas $M_2 = M_1 - \widetilde{Pre}(U) + \widetilde{Post}(U)$ and $\psi(M_2) = \psi(M_1) - \widehat{Pre}(U) + \widehat{Post}(U)$ respectively. Calculating locally the marking M_2 after the firing of a (t, c) in $\mathcal{N}$, it follows that the pre-places of t have no tokens of colour c and that the post-places will host tokens of colour c. Now, calculating the marking $\psi(M_2)$ after the firing of the respective (t, c) in $\varphi(\mathcal{N})$, the post-places will host either a token of colour c or $f(c)$ depending on the occasion. At this point, we examine how M_2 and $\psi(M_2)$ are related with respect to the conditions of marking relation mentioned in (2). We expect the post step, response and database places of both nets to get a token of colour c after the firing of the transitions of the respective enabled steps and indeed they get a token of colour c according to the post firing rules of the two nets ((∗∗) and (††)). The post control places, in $\mathcal{N}$, take the colour c and the respective ones, in $\varphi(\mathcal{N})$, take either the same colour as that of the inhibitor places or the complementary colour of c, the $f(c)$, depending on which of the step transitions has fired in each case. But in every case it can be noticed that the markings M_2 and $\psi(M_2)$ are related locally by either definition 21(xi) or (2) and this relation can be generalised using (1). Thereafter, it can be concluded that $\psi(M_2)$ is the associated reachable marking of the reachable marking M_2 of $\mathcal{N}$ after the firing of the associated transitions of the two nets.

Point (C) can be proved in a similar way.

$\square$

References

[1] M. G. Ames and A. K Dey. *Description of design dimensions and evaluation for Ambient Displays.* Computer Science Division, University of California, 2002.

[2] K Androutsopoulos. *Verification of reactive system specifications using model checking.* PhD thesis, Kings College London, 2004.

[3] N. A. Anisimov and A. A. Kovalenko. Asynchronous composition of petri nets via places. *Proceedings of the 2nd AP Ershov International Memorial Conference on Perspectives of System Informatics-96 (PSI96)*, pages 214–219, 1996.

[4] J. C. Augusto. Ambient intelligence: The confluence of ubiquitous/pervasive computing and artificial intelligence. In *Intelligent Computing Everywhere*, pages 213–234. Springer London, 2007.

[5] J. C. Augusto. Ambient intelligence: Basic concepts and applications. In *Software and Data Technologies*, volume 10 of *Communications in Computer and Information Science*, pages 16–26. Springer Berlin Heidelberg, 2008.

[6] J. C. Augusto. Past, present and future of ambient intelligence and smart environments. In *Agents and Artificial Intelligence*, volume 67 of *Communications in Computer and Information Science*, pages 3–15. Springer Berlin Heidelberg, 2010.

[7] A.A. Basten. *In Terms of Nets: System Design with Petri Nets and Process Algebra.* Eindhoven University of Technology, 1998.

[8] E. Best, R. Devillers, and M. Koutny. *Petri Net Algebra.* Springer-Verlag New York, Inc., 2001.

[9] J. Billington. Extending coloured petri nets. Technical report, Computer Laboratory, University of Cambridge, 1988.

[10] D. Bjørner, C. B. Jones, M. Mac an Airchinnigh, and E. J. Neuhold. VDM '87: VDM - A formal method at work. In *Proceedings of the VDM-Europe Symposium*, 1987.

[11] M. A. Blätke, M. Heiner, and W. Marwan. Tutorial - Petri Nets in Systems Biology. Technical report, Otto von Guericke University Magdeburg, Magdeburg Centre for Systems Biology, 2011.

[12] N. Busi. Analysis issues in petri nets with inhibitor arcs. *Theoretical Computer Science*, 275(12):127 – 177, 2002.

[13] J. Campos and J. Merseguer. On the integration of uml and petri nets in software development. In *Petri Nets and Other Models of Concurrency - ICATPN 2006*, volume 4024 of *Lecture Notes in Computer Science*, pages 19–36. Springer Berlin Heidelberg, 2006.

[14] L. Cardelli and A. D. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177 – 213, 2000.

[15] K. M. Chandy, B. Go, S. Mitra, C. Pilotto, and J. White. Verification of distributed systems with localglobal predicates. *Formal Aspects of Computing*, 23(5):649–679, 2011.

[16] E. Clarke, O. Grumberg, and D. Long. Verification tools for finite-state concurrent systems. In *A Decade of Concurrency Reflections and Perspectives*, volume 803 of *Lecture Notes in Computer Science*, pages 124–175. Springer Berlin Heidelberg, 1994.

[17] E. M. Clarke. The birth of model checking. In *25 Years of Model Checking*, volume 5000 of *Lecture Notes in Computer Science*, pages 1–26. Springer Berlin Heidelberg, 2008.

[18] E. M. Clarke, W. Klieber, M. Novacek, and P. Zuliani. Model checking and the state explosion problem. In *Tools for Practical Software Verification*, volume 7682 of *Lecture Notes in Computer Science*, pages 1–30. Springer Berlin Heidelberg, 2012.

[19] T. Coe, T. Mathisen, C. Moler, and V. Pratt. Computational aspects of the pentium affair. *Computational Science Engineering, IEEE*, 2(1):18–30, 1995.

[20] P. Cousot and R. Cousot. A gentle introduction to formal verification of computer systems by abstract interpretation. In *Logics and Languages for Reliability and Security*, NATO Science Series III: Computer and Systems Sciences, pages 1–29. IOS Press, 2010.

[21] F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva, and F. B. Vernadat. *Practice of Petri nets in manufacturing*. Chapman & Hall, 1993.

[22] P.-E. Dubreuil. Public display system. Technical report, School of Computing Science, Newcastle University, 2007.

[23] A. El Fallah Seghrouchni, K. Breitman, N. Sabouret, M. Endler, Y. Charif, and J.-P. Briot. Ambient intelligence applications: Introducing the campus framework. In *13th IEEE International Conference on Engineering of Complex Computer Systems, 2008. ICECCS 2008.*, pages 165–174, 2008.

[24] Cl. Girault and R. Valk. *Petri Nets for System Engineering: A Guide to Modeling, Verification, and Applications*. Springer-Verlag New York, Inc., 2001.

[25] M. Gonalves and J. M. Fernandes. Guidelines for modelling reactive systems with coloured petri nets. In *Model-Based Methodologies for Pervasive and Embedded Software*, volume 7706 of *Lecture Notes in Computer Science*, pages 126–137. Springer Berlin Heidelberg, 2013.

[26] M. D. Harrison and M. Massink. Modelling interactive experience, function and performance in ubiquitous systems. *Electronic Notes in Theoretical Computer Science*, 261(0):23 – 42, 2010.

[27] M. D. Harrison, C. Kray, Z. Sun, and H. Zhang. Factoring user experience into the design of ambient and mobile systems. In *Engineering Interactive Systems*, volume 4940 of *Lecture Notes in Computer Science*, pages 243–259. Springer Berlin Heidelberg, 2008.

[28] M. Heiner, D. Gilbert, and R. Donaldson. Petri nets for systems and synthetic biology. In *Formal Methods for Computational Systems Biology*, volume 5016 of *Lecture Notes in Computer Science*, pages 215–264. Springer Berlin Heidelberg, 2008.

[29] M. Heiner, M. Herajy, F. Liu, C. Rohr, and M. Schwarick. Snoopy a unifying petri net tool. In *Application and Theory of Petri Nets*, volume 7347 of *Lecture Notes in Computer Science*, pages 398–407. Springer Berlin Heidelberg, 2012.

[30] A. Iliasov, A. Romanovsky, B. Arief, L. Laibinis, and E. Troubitsyna. On rigorous design and implementation of fault tolerant ambient systems. In *10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, 2007. ISORC '07.*, pages 141–145, 2007.

[31] M. V. Iordache and P. J. Antsaklis. Synthesis of supervisors enforcing general linear vector constraints in petri nets. In *Proceedings of the American Control Conference*, pages 154–159, 2002.

[32] R. Janicki and M. Koutny. Semantics of inhibitor nets. *Information and Computation*, 123(1):1 – 16, 1995.

[33] K. Jensen. *Coloured Petri nets: basic concepts, analysis methods and practical use*, volume 1. Springer Science & Business Media, 1997.

[34] K. Jensen and L. M. Kristensen. *Coloured Petri nets: modelling and validation of concurrent systems.* Springer Science & Business Media, 2009.

[35] G. Juhás, R. Lorenz, and S. Mauser. Complete process semantics for inhibitor nets. In *Proceedings of the 28th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency*, ICATPN'07, pages 184–203, 2007.

[36] A. Konios. Ambient systems and taxonomy approaches. Technical report, School of Computing Science, Newcastle University, 2011.

[37] A. Konios. Modelling ambient systems with petri nets. In *Proceedings of 13th International Conference on Application of Concurrency to System Design, PhD session, (ACSD 2013)*, pages 247–251, 2013.

[38] A. Konios and M. Pietkiewicz-Koutny. Modelling ambient systems with coloured petri nets. Technical report, School of Computing Science, Newcastle University, 2013.

[39] A. Konios, K. Konstantopoulos, and D. Salogiannis. Virtual reality application. Department of Informatics, University of Piraeus, 2005.

[40] C. Kray, P. Olivier, A. W. Guo, P. Singh, H. N. Ha, and P. Blythe. Taming context: A key challenge in evaluating the usability of ubiquitous systems. *Ubiquitous Systems Evaluation 2007 (USE07)-Workshop at Ubicomp 2007*, pages 1–13, 2007.

[41] B. B. Kristensen, D. C. M. May, and P. Nowack. Collaboration and modeling in ambient systems: Vision, concepts and experiments. In *Proceedings of 44th Hawaii International International Conference on Systems Science (HICSS-44 2011)*, pages 1–6, 2011.

[42] N. G. Leveson. *Safeware: System Safety and Computers.* ACM, 1995.

[43] T. Matthews, T. Rattenbury, S. Carter, A. Dey, and J. Mankoff. A peripheral display toolkit. 2003.

[44] S. Merz. Model checking: A tutorial overview. In *Modeling and Verification of Parallel Processes*, pages 3–38, 2001.

[45] A. Mokhov. *Conditional Partial Order Graphs.* PhD thesis, Newcastle University, 2009.

[46] M. Mukund. Petri nets and step transition systems. *International Journal of Foundations of Computer Science*, 3:443–478, 1992.

[47] M. Nielsen and V. Sassone. Petri nets and other models of concurrency. In *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the Volumes Are Based on the Advanced Course on Petri Nets*, pages 587–642. Springer-Verlag, 1998.

[48] M. Nikolaidou and D. Anagnostopoulos. A distributed system simulation modelling approach. *Simulation Modelling Practice and Theory*, 11(3-4):251–267, 2003.

[49] C. A. Petri. *Kommunikation mit Automaten.* PhD thesis, Bonn: Institut für Instrumentelle Mathematik, 1962.

[50] A. Petrou. Exploring privacy and security in ubiquitous systems. Master's thesis, School of Computing Science, Newcastle University, 2010.

[51] M. Pietkiewicz-Koutny. The synthesis problem for elementary net systems with inhibitor arcs. *Fundam. Inf.*, 40(2,3):251–283, 1999.

[52] N. Plat, J. van Katwijk, and H. Toetenel. Application and benefits of formal methods in software development. *Software Engineering Journal*, 7(5):335–346, 1992.

[53] Z. Pousman and J. Stasko. A taxonomy of ambient information systems: Four patterns of design. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '06, pages 67–74, 2006.

[54] J.M. Proth and X. Xie. *Petri nets: a tool for design and management of manufacturing systems.* Series in intelligent control and intelligent automation. Wiley, 1996.

[55] A. V. Ratzer, L. Wells, H. M. Lassen, M. Laursen, J. F. Qvortrup, M. S. Stissing, M. Westergaard, S. Christensen, and K. Jensen. Cpn tools for editing, simulating, and analysing coloured petri nets. In *Proceedings of the 24th International Conference on Applications and Theory of Petri Nets*, ICATPN'03, pages 450–462. Springer-Verlag, 2003.

[56] W. Reisig. *Petri Nets: An Introduction.* Springer-Verlag New York, Inc., 1985.

[57] W. Reisig. Simple composition of nets. In *Proceedings of 30th International Conference on Applications and Theory of Petri Nets, PETRI NETS 2009, 2009.*, pages 23–42, 2009.

[58] A. Ruß, W. Hesse, and D. Müller. Ambient information systems - do they open a new quality of is? In *Proceedings of the Third AIS SIGSAND European Symposium on Analysis, Design, Use and Societal Impact of Information Systems. SIGSAND-EUROPE 2008.*, pages 93–108, 2008.

[59] V. Rybakov. Linear temporal logic with until and next, logical consecutions. *Annals of Pure and Applied Logic*, 155(1):32 – 45, 2008.

[60] D. Sangiorgi and D. Walker. *PI-Calculus: A Theory of Mobile Processes.* Cambridge University Press, 2001.

[61] J. L. Silva, O. R. Ribeiro, J. M. Fernandes, J. C. Campos, and M. D. Harrison. The apex framework: Prototyping of ubiquitous environments based on petri nets. In *Human-Centred Software Engineering*, volume 6409 of *Lecture Notes in Computer Science*, pages 6–21. Springer Berlin Heidelberg, 2010.

[62] D. Sokolov. *Automated Synthesis of Asynchronous Circuits Using Direct Mapping for Control and Data Paths.* PhD thesis, Newcastle University, 2006.

[63] M. Tomitsch, A. Lehner, and T. Grechenig. Towards a taxonomy for ambient information systems. In *Proceedings of the Workshop for Ambient In Systems at the 5th International Conference on Pervasive Computing (PERVASIVE 2007)*, pages 42–47, 2007.

[64] W.M.P. van der Aalst. Pi calculus versus petri nets: Let us eat "humble pie" rather than further inflate the "pi hype", 2003.

[65] S. Van der Vlugt, H. C. M. Kleijn, and M. Koutny. Coverability and inhibitor arcs: an example. Technical report, School of Computing Science, Newcastle University, 2011.

[66] J. Wang. *Petri Nets for Dynamic Event-Driven System Modeling*, pages 24.1–24.17. Chapman and Hall/CRC, 2007.

[67] R. Want and T. Pering. System challenges for ubiquitous pervasive computing. In *Proceedings of 27th International Conference on Software Engineering, 2005. ICSE 2005.*, pages 9–14, 2005.

[68] J. Woodcock, P. Gorm Larsen, J. Bicarregui, and J. Fitzgerald. Formal methods: Practice and experience. *ACM Comput. Surv.*, 41(4):1–36, 2009.