# Modelling and Validating the Multi-agent System Behaviour for a Washing Machine Production Line

**Paulo Leitão[1,2], Nelson Rodrigues[1]**

[1] Polytechnic Institute of Bragança, Campus Sta Apolónia, Apartado 1134, 5301-857 Bragança, Portugal
{pleitao, nrodrigues}@ipb.pt
[2] Artificial Intelligence and Computer Science Laboratory, Rua Campo Alegre 102, 4169-007 Porto, Portugal

*Abstract-* **This paper describes the formal modelling and validation of the behaviour of a multi-agent system that integrates the production and quality control processes in a washing machine production line. The modelling, analysis and validation process uses the Petri nets formalism that provides a rigorous and formal language based on its powerful mathematical foundation, supporting the complete verification of the system correctness during the design phase and before to proceed to the deployment phase. The behaviour models of each agent belonging to the system architecture are edited, analysed and simulated in the PnDK framework.**

## I. INTRODUCTION

The current trend in the manufacturing domain is to develop modular, flexible and reconfigurable systems that address the strong requirements imposed to manufacturing companies in terms of cost, quality, customization and responsiveness [1]. The EU FP7 GRACE (inteGration of pRocess and quAlity Control using multi-agEnt technology) project is aligned with this challenge and intends to develop a modular, intelligent and distributed manufacturing control system, using the multi-agent systems (MAS) paradigm and integrating the production and quality control processes [2]. In fact, the multi-agent systems [3] is a suitable approach to face the described challenge by providing an alternative way to design control systems based on the decentralization of the control over distributed, autonomous and cooperative entities, the agents.

The formal modelling and validation of the structural and behavioural specifications of the agents in a multi-agent system solution, and the interaction of these agents, aiming to reach the global manufacturing control system, assumes a critical aspect. The validation phase is crucial to guarantee the correctness of the designed model, guaranteeing that the model represents correctly the specifications of the real system.

In this work, the structure and dynamic behaviour of each individual agent, belonging to the GRACE multi-agent system, is modelled by using the Petri nets formalism [4-6] to understand and synthesise the system specifications and to validate the correctness of those models. The Petri nets formalism, based on a powerful mathematical foundation, is adequate to model and analyse the behaviour of complex event-driven systems, characterised as being concurrent, asynchronous, stochastic and with high distribution degree, such as the GRACE multi-agent system is. In particular, the validation can use the functional analysis and linear algebra provided by Petri nets to simulate the designed model, verifying the system behaviour over the time. Note that other available languages to model the system behaviour, e.g., UML (Unified Modelling Language), lacks in the formal validation (including the simulation) of the correctness of these models.

This paper describes the formal specification of the structural and behavioural aspects of the distributed agents composing the GRACE multi-agent system that integrates the production and quality control processes in production lines [2], using the Petri nets formalism. This formal specification involves, in a first step, the modelling of the agents' behaviour and in a second step their formal analysis, validation and simulation, by using the PnDK framework.

The rest of the paper is organized as follows: section 2 describes the behavioural models of each agent by using the Petri nets formalism. Section 3 introduces the PnDK tool used for the edition, analysis, validation and simulation of the behavioural models. Section 4 describes the quantitative analysis and Section 5 the qualitative analysis performed as part of the formal validation process. At last, Section 5 rounds up the paper with the conclusions.

## II. MODELLING THE BEHAVIOUR MODELS OF THE GRACE AGENTS

The designed GRACE multi-agent system architecture distributes the manufacturing control functions by several agents. In such distributed environment, the proposed architecture identifies several types of agents, according to the process to control and to their specialization:

- Product Type Agents (PTA).
- Product Agents (PA).
- Resource Agents (RA)
- Independent Meta Agents (IMA).

The next sections detail the behaviour models by using the Petri nets formalism for the identified types of agents. Each Petri net model contains several timed transitions representing complex functions that can be exploded by a more detailed sub-Petri net model. This allows a top-down methodology to model the logic control structure of the agent behaviour, refining step by step some timed transitions to

978-1-4673-0157-2/12/$31.00 ©2012 IEEE

include enough system operation details for hardware implementation purpose. Additionally, the extended Petri nets formalism used in this work considers tokens containing information to compress the model when the system grows.

### *A. Behaviour Model of the Product Type Agent*

The PTA agents represent the catalogue of products of a plant/production line, being its Petri net behaviour model illustrated in Fig. 1.

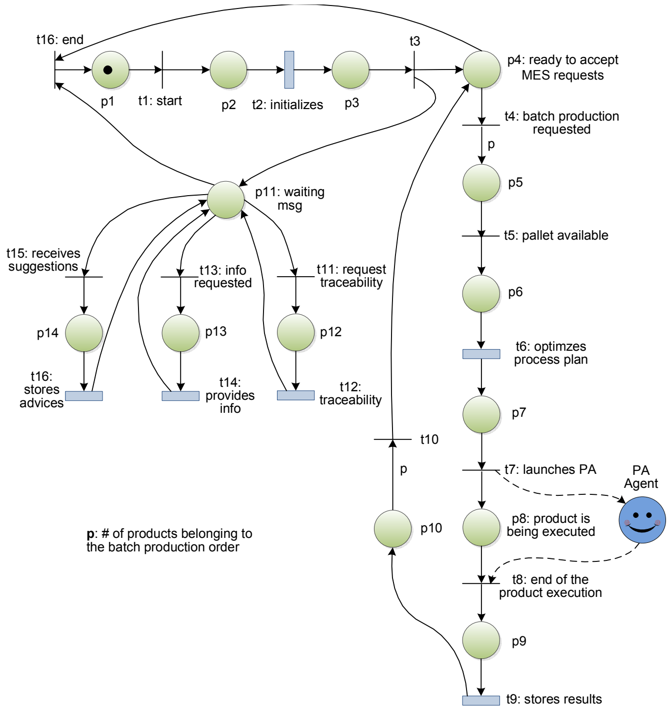



Fig. 1. Petri net Model of the PTA Agent.

Briefly, after the initialization procedure, represented by the transition $t_2$ and comprising the loading of the agent profile, the connection to the local database and the registration of the agent skills in the multi-agent system directory facilitator (DF), the PTA agent enters in a state (i.e. $p_4$) waiting for a production order to be executed. When the request occurs, coming from the ERP/MES system, the PTA agent launches PA agents according to the availability of the pallets in the production line. Note that it is possible to have several product instances belonging to the same PTA being processed simultaneously in the production line, in this case *p*. At the end of each production execution, the PTA agent stores the production data for future processing (e.g., historical information to be used for traceability purposes).

In parallel, the agent is continuously waiting for requests from other agents related to information about the current status of the production order execution and traceability of previous products execution. Additionally, it can also receive suggestions for local optimization provided by the IMA agents, which are stored in the local database and will be used in future process plan optimization.

### *B. Behaviour Model of the Product Agent*

A PA agent is launched to manage the production of a product instance, being provided with a proper process plan and the identifier of the associated pallet. The Petri net behaviour model for the PA agent is illustrated in the Fig. 2.

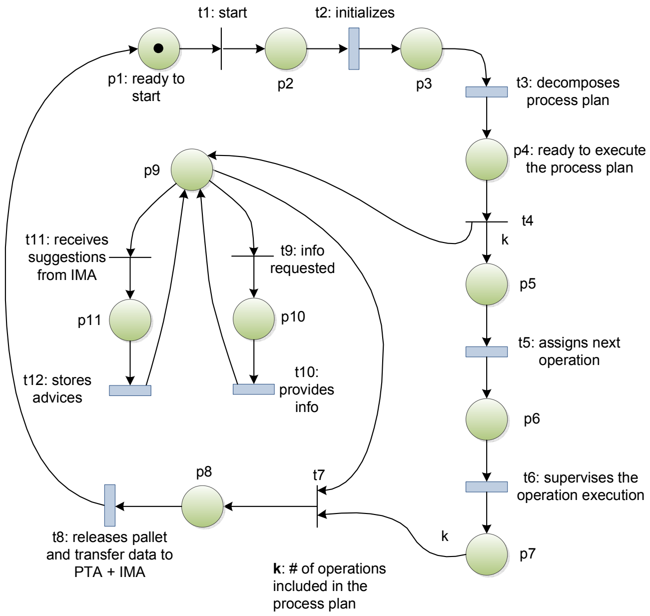



Fig. 2. Petri net Model of the PA Agent.

After the initialization procedure, the PA agent starts the production process according to the process plan, interacting with RA and IMA agents. During this phase, two sub-behaviours are launched, namely executing the process plan ($p_5$) and waiting for queries and suggestions ($p_9$), being handled asynchronously in parallel, so that the execution of one process doesn't block the execution of the other process.

In the execution of the process plan, the agent initially decomposes the process plan into *k* operations and makes an initial assignment to the RA agents (the complexity of this function is strongly dependent of the production type: less complexity in rigid production lines and high complexity in flexible systems).

The agent starts managing the execution of the *k* operations taking into consideration the precedences among operations and the current situation of the production line. For this purpose, at the end of the execution of one operation, the next operation to be executed is selected and assigned by the function represented by the transition $t_5$. The assignment of the next operation includes the selection of the resource that will execute the operation and the configuration of the operation parameters (i.e. the adaptation of the operation parameters taking into consideration the current production situation and the historical information of the product execution, e.g., provided by previous quality control operations). After the assignment of the next operation, the PA agent triggers the execution of the activity “supervises the operation execution” represented by the transition $t_6$.

At the end of the execution of all operations belonging to the process plan, the PA executes a set of actions, represented by the transition $t_8$, related to release the pallet to be used in upcoming products in the production line and to transfer the relevant information of the product execution to the associated PTA and IMA agents. These agents will use this information for posterior analysis to perform adaptation in the future process plans.

During the execution of the process plan, the monitoring requests about the state of the product execution, requested by other agents, are handled in parallel and not affecting the process execution. Also the suggestions sent by IMA agents are stored in the local database for posterior usage.

### *C. Behaviour Model of the Resource Agent*

A RA agent is associated to each resource of the production line, namely machines, quality control stations and operators. The Petri net behaviour model for the RA agent is illustrated in the Fig. 3.

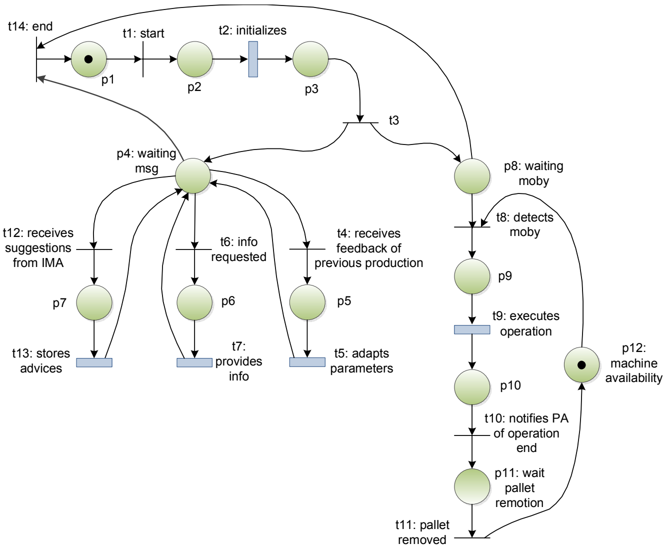



Fig. 3. Petri net Model of the RA Agent.

As the previous agents, after the initialization phase, the RA agent enters in two sub-behaviours, running in parallel, related to managing the operation execution and waiting for monitoring requests or feedback suggestions.

When a pallet arrives to the workstation, detected by the RFiD (Radio-Frequency Identification) reader, represented by the transition $t_8$, the agent starts the execution of the operation, represented by the timed transition $t_9$. In the execution of operation activity, the RA agent may adjust the operation parameters according to its local knowledge, i.e. based on the knowledge gathered and generated by operations previously executed, and then triggers the execution of the operation by sending a command to the resource (physical equipment or operator). At the end of performing the processing or testing operation, the gathered data is analysed, and the results are sent in a XML file to the PA and IMA agents.

In parallel, the agent may receive information of the execution of the previous products, from the quality control agents, which will support the improvement of its performance in the future (e.g., by using self-learning procedures in the activity "adapts parameters" represented by the timed transition $t_5$). Also, the suggestions sent by IMA agents are stored in the local database for posterior usage.

### *D. Behaviour Model of the Interdependent Meta Agent*

The Petri net behaviour model for the IMA agent is illustrated in the Fig. 4, comprising the behaviours that provide the global supervisory control, optimized planning and decision-making mechanisms. This agent acts at strategic level, taking advantage of its global perspective to provide global optimization over the production system (seen as advices from the other entities of the system).

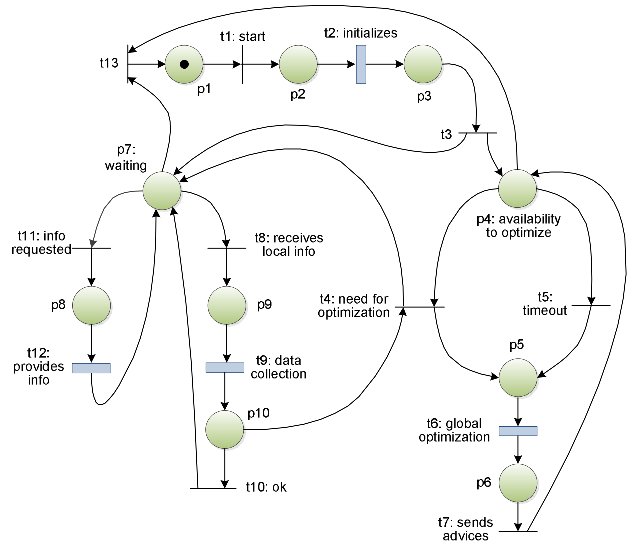



Fig. 4. Petri net Model of the IMA Agent.

This model contains two sub-behaviours running in parallel: one related to data collection and another to data processing.

In the first one, represented by having a token in the place $p_7$, the agent is waiting for feedback from individual PA and RA agents, to execute the data collection and aggregation. The data collection will be used to feed the execution of the global optimization, and may include a pre-analysis of data that allows to detect shortly any major deviation in the production.

The second sub-behaviour is related to perform the detection of patterns and trends aiming the global optimization. For this purpose, a token in the place $p_4$ represents the availability to start the execution of a global optimization process. The modularity of the agent behaviour allows the easy plug-and-play of different global optimization algorithms, e.g., neural networks, data mining or statistical analysis.

The expected outputs of the global optimization are mainly related to warnings for the maintenance department about the need to perform recovery operations or improvements on the processing/testing machines (related to the improvement of the process), and suggestions to the other agents aiming their adaptation/optimization according to the past knowledge and current situation (related to the improvement of the product).

## III. Edition of the Petri nets models

The validation of the GRACE multi-agent system, through the analysis and simulation of the behavioural models of the individual agents, was performed using the Petri nets Development toolKit (PnDK) software tool [7]. This software tool, developed in C++ programming language, allows the edition, analysis and simulation of generalised and temporised Petri nets models. The structural analysis is based on the matrix representation of the Petri net, i.e. the incidence matrix, and the quantitative analysis is performed by means of the simulation of the temporized Petri net models.

In this paper, the edition, analysis and simulation of the GRACE multi-agent system is illustrated by the analysis of the behavioural model of the RA agent. Fig. 5 illustrates the Petri net behavioural model of the RA agent, edited in the PnDK tool.

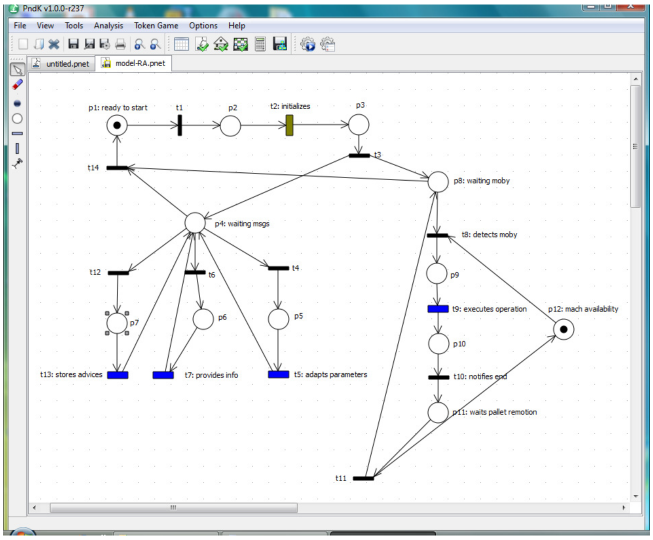



Fig. 5. Edition of the RA Agent Model.

The validation of the Petri net models can be performed through two distinct ways: qualitative analysis, related to the structural and behavioural validation, and quantitative analysis, related to the simulation. Next sections will describe these two types of validation.

## IV. Qualitative Analysis

The qualitative analysis allows the verification of the structural and behavioural properties of the model, extracting conclusions about the operation of the system, such as the existence of deadlocks, the bounded capacity of resources, and the existence of structural and behavioural conflicts in the system [8]. In this work, the behavioural properties were analysed using linear algebra methods, provided by the PnDK software tool.

### A. *Validation of the behavioural models*

The results from the analysis of the behavioural properties of the RA behavioural model are illustrated in Fig. 6.

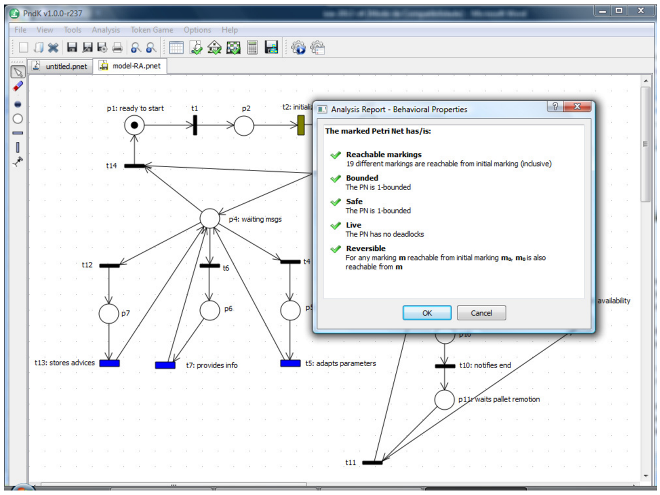
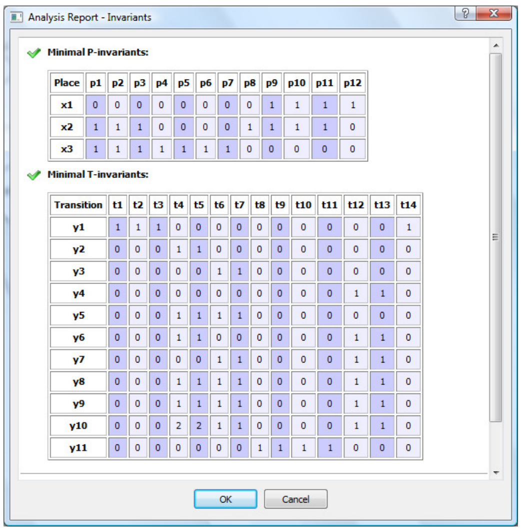



Fig. 6. Behavioural Analysis of the Model of the RA Agent.

From this analysis it is possible to verify that the model is:

- *Bounded*, i.e. the maximum number of tokens in a place is 1, which means that the resource can only execute one operation at time.
- *Reversible*, i.e. the initial marking is reachable from all reachable markings, guaranteeing that the model can reinitialise from itself.
- *Absence of deadlocks*, i.e. for each reachable marking *m* there is at least one transition that can fire to reach another marking.

More information related to the model can be extracted from the analysis of the P- and T- invariants, illustrated in Fig. 7, extracted from the incidence matrix and using linear algebra methods.


Analysis Report - Invariants

Minimal P-invariants:

| Place | p1 | p2 | p3 | p4 | p5 | p6 | p7 | p8 | p9 | p10 | p11 | p12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| x1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| x2 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| x3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Minimal T-invariants:

| Transition | t1 | t2 | t3 | t4 | t5 | t6 | t7 | t8 | t9 | t10 | t11 | t12 | t13 | t14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| y1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| y2 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| y3 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| y4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| y5 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| y6 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| y7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| y8 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| y9 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| y10 | 0 | 0 | 0 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| y11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

OK Cancel

Fig. 7. P- and T-invariants for the Model of the RA Agent.

The analysis of the P-invariants constitution allows confirming mutual exclusion relationships among places, functions and resources involved in the structure and behaviour of the agent. For example, analysing the P-invariant $x_1 = \{p_9, p_{10}, p_{11}, p_{12}\}$, it is possible to confirm that during the execution of one operation, only one of the places referred in the P-invariant can be marked at any time, i.e. there is a mutual exclusion relationship among those places representing different stages in the operation execution.

The T-invariants represent the several sequences of operation, i.e. the work cycles, exhibited by the behaviour model. From the analysis of the T- invariants it is possible to verify the existence of 5 invariants (note that the invariants that are linear dependent are not considered), which have the following physical meaning:

- $y_1=\{t_1, t_2\ t_3, t_{14}\}$, representing the cycle of initialization and conclusion of the agent behaviour.
- $y_2=\{t_{14}, t_5\}$, representing the work cycle that adapts the operation parameters according to the feedback of previous production execution.
- $y_3=\{t_6, t_7\}$, representing the work cycle that provides information to the requested agents.
- $y_4=\{t_{12}, t_{13}\}$, representing the work cycle that stores the advices/suggestions sent by the IMA agents.
- $y_{11}=\{t_8, t_9, t_{10}, t_{11}\}$, representing the work cycle related to the execution of an operation.

The observation of these properties allows to conclude about the structural and behavioural correctness of the model.

### B. *Validation of the behavioural sub-models*

The model for the RA agent comprises the refinement of some timed transitions, following a top-down approach, to include enough system operation details. Namely, it includes the sub-Petri net models "*initializes*", "*executesOperation*", "*providesInfo*", "*storesAdvices*" and "*adapts parameters*".

The formal validation of the large model requires the analysis of the sub-Petri nets and the application of the theorems established by [9] and latter generalised by [10] about the preservation of boundedness and liveness properties. This theorem points out that using stepwise refinement it is not necessary to perform the analysis of the detailed and large Petri net, because all of its properties can be deduced from the analysis of the initial Petri net and each one of the sub Petri nets.

The edition and behavioural analysis of the sub-Petri net models belonging to the RA model are then performed to analyse the systems properties. As an example, the edition and analysis of the sub-Petri net model "*executesOperation*" is illustrated in Fig. 8.

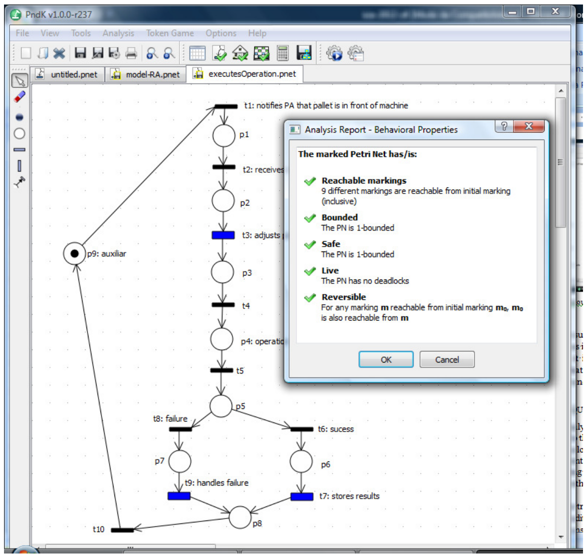


Fig. 8. Edition and behavioural analysis of the sub-model "executesOperation".

Since all these four sub-Petri net models are bounded and absent of deadlocks, as illustrated for the sub-Petri net model "*executesOperation*", it is possible to conclude, according to the Vallete theorem [9], that the larger Petri net model for the RA agent is also bounded and absent of deadlocks.

## V. QUANTITATIVE ANALYSIS

The quantitative analysis requires the introduction of time parameter associated to the transitions. The editor draws these transitions with blue colour (represented by a solid rectangle) and they are representing time-consuming activities. The transitions representing logic and non-time consuming activities are drawn with black colour (represented by a thin rectangle).

In the RA model, the transitions $t_1$, $t_3$, $t_4$, $t_6$, $t_8$, $t_{10}$, $t_{11}$ and $t_{12}$ represent logical conditions, being estimated 1 t.u. per transition. The transitions $t_2$, $t_5$, $t_7$, $t_{13}$ represent computational activities, being estimated 2 t.u. per transition. The transition $t_9$ represents the activity related to the operation execution in the machine, being considered that this activity takes 10 t.u..

The simulation of the Petri net behavioural models uses the token-game provided by the Petri nets formalism, showing the evolution of the tokens over the places and over the time. Fig. 9 shows a state of this evolution progress, where the place $p_4$ is marked, meaning that is waiting for messages, and the place $p_9$ is marked, being the timed transition $t_9$ enable and ready to be fired.

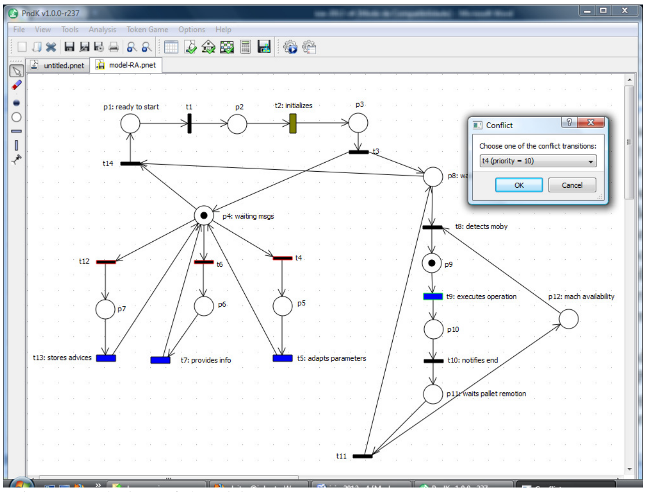


Fig. 9. Evolution of the Firing Process.

The complete information about the timed evolution of the agent behaviour can be summarized with a Gantt diagram, which is illustrated in Fig. 10, reflecting the temporal sequence of the system functioning.

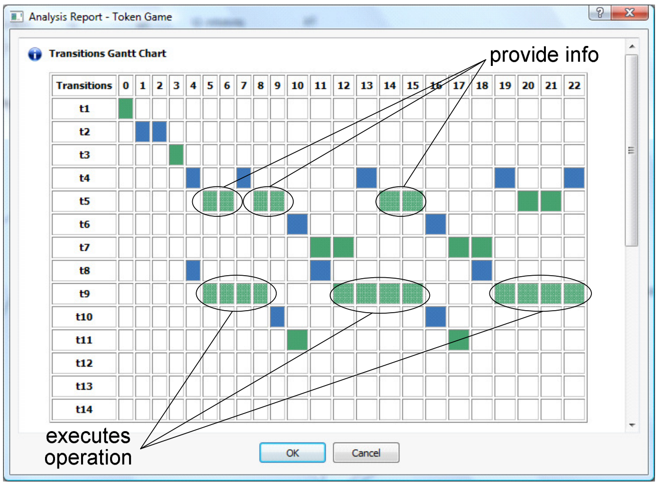


Fig. 10. Gantt Diagram for the Performance Analysis.

The analysis of this Gantt diagram allows to discover important characteristics of the agent behaviour, namely cyclic evolution, existence of bottlenecks, mutual exclusion activities or conflicts. Based on the information provided by the simulation, optimization strategies were proposed and online verified, contributing for the improvement of the final correctness and system performance.

The procedure for qualitative and quantitative analysis, described previously for the RA agent, has been repeated for the other Petri net models of the individual GRACE agents, allowing the validation of the structural and behavioural specifications of the GRACE multi-agent system. At the end, the GRACE multi-agent system is ready for being implemented and deployed using a proper multi-agent system development framework, e.g., the Java Agent Development Framework (JADE) platform [11].

## VI. Conclusions

This paper presents the modelling and validation of the behaviour of individual agents belonging to the GRACE multi-agent system that integrates the production and quality control processes in a washing machine production line. The behaviour models were designed using the Petri nets formalism that provides a formal language to model and validate complex systems behaviour based on its powerful mathematical foundation.

The structural and behavioural validation, and the posterior simulation of the Petri net models, allowed the complete specification of the correct behaviour for the GRACE multi-agent system. After adjusting and exploring different strategies during the simulation phase, the GRACE multi-agent system is ready for implementation using a proper multi-agent system development framework.

Future work is related to the specification of the interaction patterns among the Petri net models and lately to its implementation using a proper multi-agent system development framework.

## Acknowledgment

This work has been partly financed by the EU Commission, within the research contract GRACE coordinated by Univ. Politecnica delle Marche and having partners SINTEF, AEA srl, Instituto Politecnico de Bragança, Whirlpool Europe srl, Siemens AG.

## References

[1] H. ElMaraghy, “Flexible and Reconfigurable Manufacturing Systems Paradigms”, *International Journal of Flexible Manufacturing Systems*, 17, 2006, pp. 261-271.

[2] P. Castellini, C. Cristalli, M. Foehr, P. Leitão, N. Paone, I. Schjolberg, J. Tjønnås, C. Turrin, T. Wagner, “Towards the Integration of Process and Quality Control using Multi-agent Technology”, Proceedings of the 37th Annual Conference of the IEEE Industrial Electronics Society, Melbourne, Australia, 7-10 November, 2011, pp. 374-379.

[3] M. Wooldridge, *An Introduction to Multi-Agent Systems*. John Wiley & Sons, 2002.

[4] T. Murata, “Petri Nets: Properties, Analysis and Applications”, *IEEE*, 77(4), 1989, pp 541-580.

[5] R. Zurawski, M. Zhou, “Petri Nets and Industrial Applications: A Tutorial”, *IEEE Transactions on Industrial Electronics*, 41(6), 1994, pp. 567-583.

[6] A. Desrochers and R. Al-Jaar, *Applications of Petri Nets in Manufacturing Systems-Modeling, Control and Performance Analysis*, 1995, IEEE Press.

[7] J.M. Mendes, A. Bepperling, J. Pinto, P. Leitão, F. Restivo and A.W. Colombo, “Software Methodologies for the Engineering of Service-oriented Industrial Automation: The Continuum Project”, Proceedings of the 33rd Annual IEEE International Conference on Computer Software and Applications (COMPSAC’09), Seattle, Washington, 20-24 July, 2009, pp. 452-459.

[8] K. Feldmann, C. Schnur, and A. Colombo, “Modularised, Distributed Real-Time Control of Flexible Production Cells, Using Petri Nets”, Control Engineering Practice, 1996, pp. 1067-1078.

[9] R. Vallete, “Analysis of Petri nets by Stepwise Refinements”, *Journal of Computer and Systems Science*, 18, 1979, pp. 35-46.

[10] I. Suzuki and T. Murata, “A Method for Stepwise Refinements and Abstractions of Petri nets”, *Journal of Computer and Systems Science*, 27, 1983, pp. 51-76.

[11] F. Bellifemine, G. Caire, D. Greenwood, *Developing Multi-Agent Systems with JADE*. Wiley, 2007.