2018 26th Mediterranean Conference on Control and Automation (MED)
Zadar, Croatia, June 19-22, 2018

Modeling Task Dependencies in Project Management using Petri nets with arc extensions

George J. Tsinarakis

Abstract— **Alternative ways of modeling the four possible types of task dependencies in Project Management are introduced. The approach presented makes possible modeling and simulation of the execution procedure of complicated projects that can be analyzed in interconnected tasks, using Petri Nets and their extensions. The proposed method allows the calculation of the overall duration, as well as other useful measures according to the task durations and the interconnections between them. The advanced modeling power of Petri Nets makes possible the inclusion of random events (following a distribution or not) that in real world may disturb the project execution, as well as may change its duration and may lead to more accurate calculations compared to the traditionally used Project management tools and methods such as Network and Node diagrams. This can be used for more efficient project planning as well as for designing necessary actions to correct real time behavior of Project execution.**

I. Introduction

Project Management (PM) is a rather young field in Management Science that receives increased importance nowadays. In the first steps of this field only people of Academy and Engineers especially coming from the sectors of Construction and Defense were using the tools and methods of PM to optimize their scheduling, objectives (usually project duration) and improve the allocation of their resources, external activity planning and coordination and communication with external entities [1]. However this has changed significantly the last thirty years when more sectors such as Information Theory, Energy, Pharmacy and many more adopted the Project Approach. Currently, more than 20% of the global economic activity takes place as projects, and in some emerging economies it exceeds even 30 %. World Bank data indicate that 22 % of the world's $55 trillion Gross Domestic Product (GDP) is gross capital formation, which is almost entirely project-based [2].

According to [3] "A project is an organization of people dedicated to a specific purpose or objective. Projects generally involve large, expensive, or high risk undertakings which have to be completed by a certain date, for a certain amount of money, with some expected level of performance. The result is a unique product, service or generally result. At a minimum, all projects need to have well defined objectives and sufficient resources to carry out all the required tasks." Project management concerns the planning, organizing, directing, and controlling of company resources for a relatively short term objective that has been established to complete specific goals and objectives [4].

George J. Tsinarakis is with the Production Engineering and Management School, Technical University of Crete, GR-73100 (+306946298642; e-mail: tsinar@dpem.tuc.gr)

[5] summarizes the five typical characteristics of most projects: i) A start and a finish ii) A time frame for completion iii) An involvement of several people on an ad-hoc basis iv) A limited set of resources and v) A sequencing of activities and phases.

A classical Discrete Event tool, Petri nets (PNs) and some extensions with certain features and advanced modeling capabilities are used. In literature PNs are used by many authors for modeling and study of real world systems of increased complexity from different fields including production, industry, business, biology, computer science and more [6], [7]. In addition, the problems modeled and studied using Petri nets vary from strictly theoretical to practical.

PN use in Project Management literature is rather short. In [15] a reconfiguration method for project management systems to incorporate resource constraints for planning and scheduling is presented. In [16] a formal model based on Petri nets is designed to show impacts of resource assignment strategies on project schedules.

The rest of the paper is organized as follows: In Section II Ordinary Petri Nets, Timed Petri-Nets and TPN with arc extensions are introduced shortly. Section III describes the different types of task dependencies that are possible in Project Management, while in Section IV different versions of the Petri-Net models of these alternative task interrelation types are introduced and explained analytically. Section V presents a case study of a simple Project and how the proposed models are used to construct the overall model and finally Section VI, concludes the manuscript.

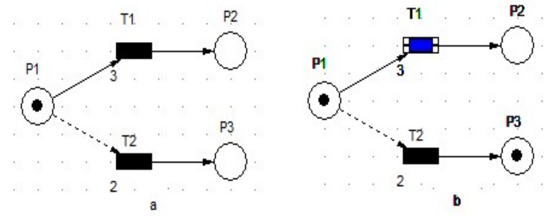
II. Timed Petri nets with arc extensions

In this part the fundamental theory of Timed Petri Nets and Timed Petri Nets with arc extensions is presented. These concepts are described analytically in [8] The initial formalism introduced were the Ordinary Petri Nets (OPN), that are bipartite directed graphs formally defined as the five-tuple: $PN= \{P, T, I, O, m_0\}$, where $P= \{p_1, p_2 ... p_{np}\}$ is a finite set of places, $T= \{t_1, t_2, ..., t_{nt}\}$ is a finite set of transitions, $P \cup T=V$, where V is the set of vertices and $P \cap T = \emptyset$. I represents an input function and O an output function and m_0 is PN initial token distribution referred to literature as marking. In a Petri net graph, places represent resources and control conditions; transitions represent events and arcs direct connection, access rights or logical connection between places and transitions. PN properties (reachability, coverability, safeness, k-boundedness, conflicts, liveness, reversibility, persistency, deadlock-freeness, P- and T-invariants) capture precedence relations and structural interactions between system components

T-timed PNs arise from the corresponding Ordinary PNs by associating each transition t_i a firing delay (constant, following a given distribution or random according to the actions modeled) and are defined as $TPN= \{P, T, I, O, m_0, D\}$ with D representing time delay that is a function from the set of non-negative real numbers $\{0, \mathbb{R}^+\}$. Arc extensions increase the modeling power of Petri nets and allow modeling of more sophisticated concepts with generally simpler net structures (consisting of less places and transitions than the respective ordinary PN models). Arc extensions are critical for modeling of complicated cases and relations, including that of control actions embedded in PN structure. In general, 3 types of arcs are used, the standard arcs (→), inhibitors arcs that are represented by arcs whose end is marked with a small circle (─O) and activator arcs that are drawn as dashed vectors [9], [10], [11].

An input place p to a transition t through an inhibitor arc of weight w, t can fire only as long as the number of tokens residing in p remains less than w. On the contrary, if p is connected to t through a test arc with weight w, t can fire if the marking of p is greater than or equal to w. During the firing process, the marking of p remains unchanged from firing of transitions that p is input place connected through test and inhibitor arcs (no matter if these arcs are input or output ones) since no tokens are removed from input places through these transition firings as shown in Fig. 1.

Figure 1. Marking of the PN model before (a) and after (b) T_2 firing (connected through activator arc to p_1) after passing of two time units.

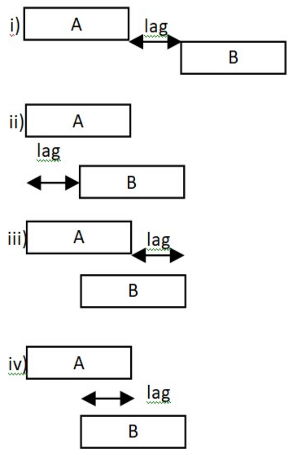


III. TASK DEPENDENCIES IN PROJECT MANAGEMENT

A project consists of a well-defined collection of simpler jobs, called tasks and ordinarily culminates in the creation of an end product or service. The main object of PM is to define a preferred sequence of execution for the project's tasks that will be called schedule. The goal of the methods and tools used is to define the optimal with respect to a given objective (that most of the times is the overall Project duration) and under certain constraints (in resources, money etc.) solution. There will always be certain amounts of uncertainty associated with projects and this uncertainty represents risk—an ever-present threat to the ability to make definitive plans and predict outcomes with high levels of confidence [12].

Another type of constraint regards the dependencies between the tasks composing a project. There are four main types of such task interrelations that are used to construct network and node diagrams that allow the application of scheduling methods as Critical Path Method (CPM). In all the cases, the dependencies describe the relationships between two (or more) tasks, such that one is the independent task and the other is the dependent one [13]. Each task may proceed from multiple other tasks and also may depend from and follow multiple ones. The alternative types of dependencies are: i) Finish to Start, where task A must finish for at least d time units (length of lag) before task B can start. This type of dependency is the most common in practice, ii) Start to Start where task B cannot start before task A has already started for d time units, iii) Finish to Finish in which task A has to finish at least d time units before task B can finish and iv) Start to Finish where task B cannot finish before task A has started for d or more time units. This is the rarest dependency and many authors do not consider it in practice. Graphical representation of the possible task dependencies is shown in Fig. 2. When there is no lag between the two tasks the length of the respective arrow as well as d is set equal to zero.

Figure 2. Alternative types of task dependencies.



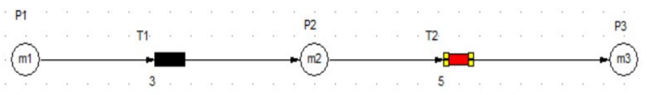
IV. PETRI NET MODELING OF ALTERNATIVE TASK DEPENDENCIES

In this part of the paper the Petri net models of the four possible types of dependencies are presented. All models are generally live, deadlock free and all the places used in them have maximum capacity equal to one (since each task is considered not to be repeated). All models are built, tested and extensively simulated using Visual Object Net Software Package [14]

A. Finish to Start Models

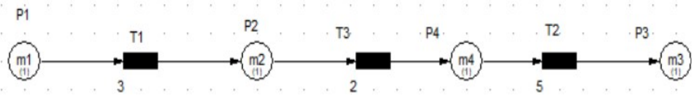
The simplest Petri net model is the one describing Finish to Start relation. In the simplest case, that there is no lag, the finish of Task A will be the start of Task B (if other conditions that are embedded in the PN model as input places to the T_1 transition do not inhibit the start of the second task, such as a certain resource availability). In this case the model consists of 3 places and 2 transitions as shown in Fig. 3. T_1 and T_2 describe task A and B implementation respectively and their duration is equal to the duration of the tasks. Unitary capacity places P_1 P_2 and P_3 represent ready to start task A, finished task A and at the same time ready to start task B and finished task B. One token initially resides in P_1. The proposed model in fact represents serial execution of the events describing the two tasks A and B.

Figure 3. Finish to Start with no lag Petri net model.



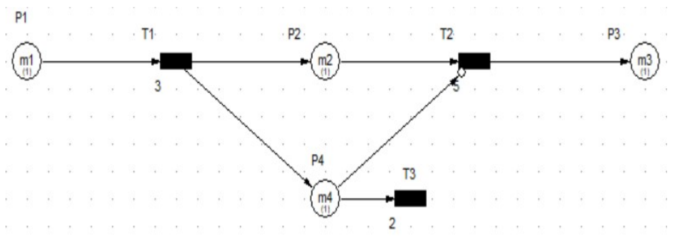
When a time lag exists between finish of task A and start of task B, the model has to become more complicated. Two alternative implementations of this concept are presented in Figs 4 and 5. According to the first one, a transition and a place are added after P_2 (that now represents completed task A only) and before T_2. T_3 represents the lag duration that has to pass after the execution of task A before the implementation of Task B can start and P_4 that task B is ready to start.

Figure 4. Finish to Start with lag Timed Petri net model.



In the second version, represented at Fig. 5, Timed Petri nets with arc extensions are used to model the lag duration. In particular, in the initial model of Fig. 3, a new place and a new transition are added with a completely different mission during model execution compared to the respective nodes of the model of Fig. 4. In this case the extra place P_4 is connected in parallel to P_2 and takes input from T_1. Its output is through an inhibitor arc to T_2 and through a normal arc to the new transition T_3 (that represents pass of time equal to lag duration d). Through this structure, when Task A finishes, a token will be moved in P_2 as before and another token will reside in P_4. As long as this token remains in P_4, T_2 cannot become enabled because of the presence of the inhibitor arc connecting them. After d time units, the enabled transition T_3 will fire and the token of P_4 will be “consumed”. This will enable T_2 that will fire after time period equal to its duration and token will move to P_3, representing the execution of task B. The model of Fig. 5, is more efficient compared to that of Fig. 4 as its main part represents the tasks and their implementation while the extra place, transition and arcs are mostly a control structure and can be combined with extra control actions that may be necessary during the implementation of the model of a complicated project. The nodes (places and transitions) complexities of models of Figs 4 and 5 are exactly the same.

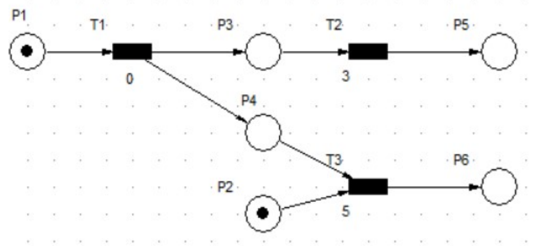
Figure 5. Finish to Start with lag Timed Petri net with arc extensions model.



B. Start to Start models

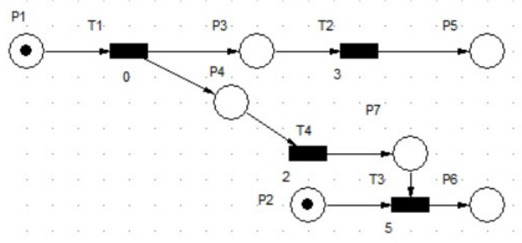
Initially the Start to Start model with no time lag between Tasks A and B is implemented and presented in Fig. 6. In this, execution of Task B can begin just after the beginning of the execution of Task A. For this reason, a trick is used and the implementation of task A is modeled by two consecutive events, represented by T_1 and T_2. T_1 represents beginning of Task A and for this reason has duration equal to 0 (immediate transition) and sends a token in P_4 that is an input place to T_3. T_2 represents implementation of Task A and its duration is the duration of Task A while T_3 implementation of Task B. P_1 stands for task A ready to start, P_2 for task B ready to start, P_3 for task A already started, P_4 is a control place that ensures the validity of task A start condition, P_5 represents task A completion and P_6 the respective for task B. In order to enable T_3, T_1 should have already fired since in any other case P_4 will remain empty. For the execution of process A to start a token should be present in P_1 while the respective stands for task B and P_2.

Figure 6. Start to Start with no lag Timed Petri net model.



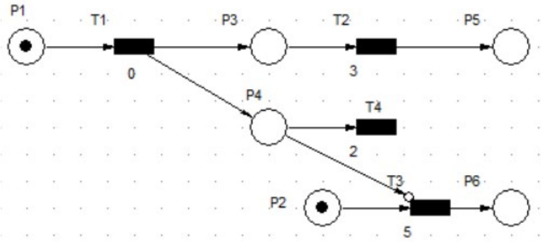
The Timed Petri net model of the Start to Start task dependency with a lag is presented in Fig. 7. The main structure of the model is similar to that of the respective PN model of Fig. 6. The main difference is that P_4 is not directly connected to T_3 anymore. P_4 is now the input place to T_4 that represents the minimum lag duration between tasks A and B. After d time units T_4 fires and sends a token to the new place P_7 that is the input place of T_3 representing validity of the Start to Start with given minimum lag condition.

Figure 7. Start to Start with lag Timed Petri net model.



An alternative way of modeling Start to Start task dependency with lag between tasks is presented in Fig 8. In this an inhibitor arc is used to connect P_4 with T_3. As soon as T_1 fires, a token stays in P_4 for d time units, making not possible the enabling of T_3. After the pass of d time units T_4 fires, token leaves from P_4 and T_3 is enabled. This model is simpler compared to that of Fig. 7 since it has one less place and one less arc. The pair of nodes P_4 and T_4 are used only for control purposes and do not represent a task implementation or state.

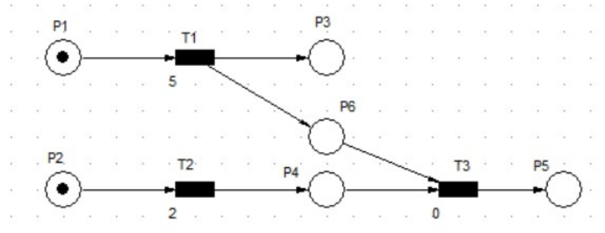
Figure 8. Start to Start with lag Timed Petri net with arc extensions model.



C. Finish to Finish models

In this Section the Finish to Finish task dependency models are presented, starting with the representation of the case when there is no lag between the two tasks. In order to implement this model a trick similar to the one used in the previous models will be applied. In particular, the execution of task B is represented from two consecutive events, the one referring to task implementation and the second, a type of dummy event that has zero duration describes task B completion. This completion will be possible only if task A is already completed. The Petri net model is shown in Fig. 9.

Figure 9. Finish to Finish with no lag Timed Petri net model.



In this P_1 and P_2 represent tasks A and B ready to start, P_3 and P_5 tasks completed, P_4 that execution of task B has finished and if the condition finish to finish that is represented by P_6 is valid it can be completed. T_1 and T_2 represent execution of tasks A and B and immediate transition T_3 completion of task B. It must be noted, that if task A has no other tasks following it, the model of Fig. 9 can be significantly simplified. In particular the use of P_6 is not necessary and T_3 can be enabled and fired by connecting P_3 to T_3 through an activator arc. The respective model is presented in Fig. 10.

Figure 10. Finish to Finish with no lag Timed Petri net model where task A has no other tasks following.

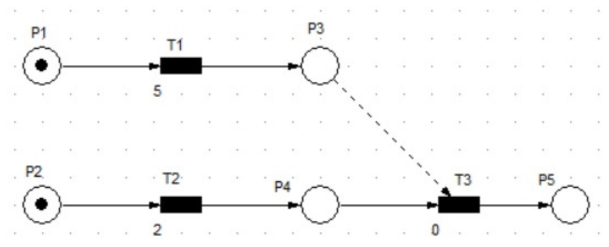
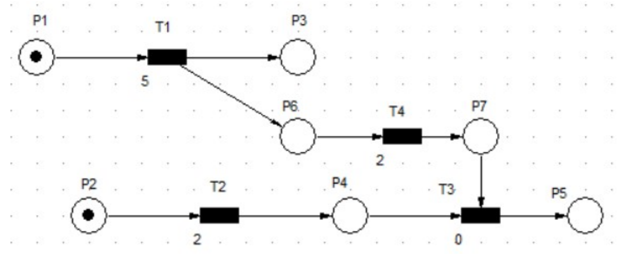
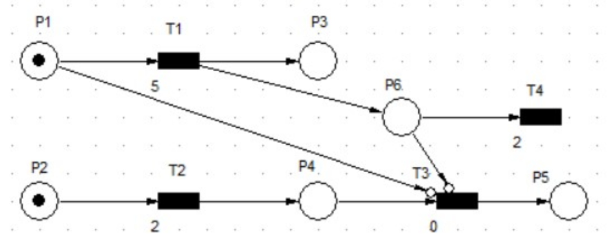


Figure 11. Finish to Finish with lag Timed Petri net model.



The Timed Petri net model of the Finish to Finish dependency with a lag is presented in Fig. 11. The main structure of the model is similar to the respective PN model of Fig 9. The major difference is that P_6 is not directly connected to T_3 anymore. P_7 is now the input place to T_4 that represents the minimum lag duration between the completion of tasks A and B. After d time units T_4 fires and sends token to the new place P_7 that is the input place of T_3 representing validity of the Finish to Finish dependency with given minimum lag condition.

Figure 12. Finish to Finish with lag Timed Petri net model with arc extensions.

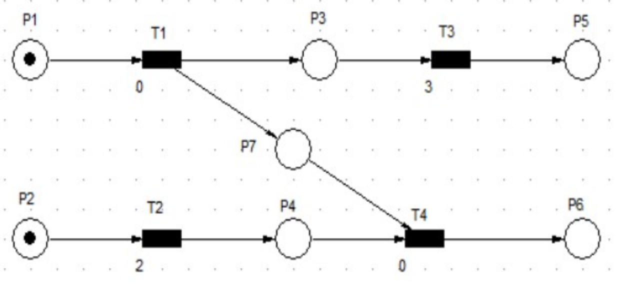


An alternative way of modeling Finish to finish task dependency with lag between tasks is presented in Fig. 12. In this two inhibitor arcs are used to connect P_1 and P_6 with T_3. The necessity of using two inhibitor arcs and not just one has to do with the durations of the different tasks. If the duration of T_2 firing is less than the respective one of T_1 and no inhibitor arc from P_1 exists, T_3 can fire (since it is immediate transition) before the finish of task A. In case that T_1 has a shorter duration that T_2, the inhibitor arc from P_1 to T_3 can be omitted. Such an event would violate the dependencies between tasks. With the given net structure, as soon as T_1 fires, a token remains in P_6 for d time units, making not possible the enabling of T_3. After the pass of d time units T_4 fires, token leaves from P_6 and T_3 is enabled. This model is simpler compared to that of Fig. 11 since it has one less place and one less arc. The pair of nodes P_6 and T_4 are used only for control purposes in this model.

D. Start to Finish models

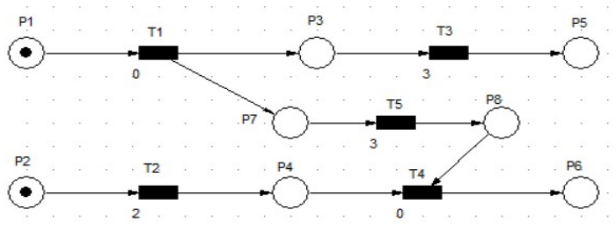
In this Section the start to finish models are presented, starting with the model when there is no lag between the two tasks. This practically means that Task B cannot be completed if Task A has not started. In this case both tasks have to be represented by two consecutive events. In both cases, the one event is dummy and has duration zero but is necessary in order to define the control actions and conditions. In particular, for task A, first event of zero duration will be start of the task and the second one execution of Task A while in the case of task B first event will represent implementation of the task (with duration equal to tasks B duration) and the second one whose duration is zero tasks B completion, that would be possible only after the finish of the first event of task A. The respective model is presented in Fig. 13.

Figure 13. Start to Finish with no lag Timed Petri net model.



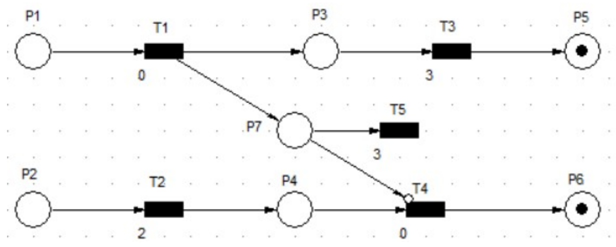
The Timed Petri net model of the Finish to Finish dependency with a lag is presented in Fig. 14. The main structure of the model is only partially different from the respective PN model of Fig. 13. The main difference is that P_7 is not connected anymore directly to T_4 but a new pair of nodes has been added between them. P_7 is now the input place to T_5 that represents the minimum waiting time duration after the start of task A and before the completion of task B is possible. After d time units T_5 fires and sends one token to the new place P_8 that is the input place of T_4 representing validity of the Start to Finish dependency with given minimum lag.

Figure 14. Start to Finish with lag Timed Petri net model.



An alternative way of representation of Start to Finish task dependency with lag between tasks using Petri nets with arc extensions is presented in Fig. 15. In this an inhibitor arc is used to connect P_7 with T_4, making impossible the enabling of T_4 for d time units. After this time, T_5 fires and P_7 becomes empty again. The meaning and the use of all the rest places and transitions of the model is exactly the same with the respective ones of the model of Fig. 14. This allows T_4 to become enabled and fire at once since it is an immediate transition. This model is simpler compared to that of Fig. 14 since it has one less place and one less arc.

Figure 15. Start to Finish with lag Timed Petri net model with arc extensions.



E. General conclusions

The models of the four types of dependencies between tasks in Project Management have been introduced, including the possibility of lag between them or not. The proposed models are as general as possible. In practice, according to specific features of the Project under study (relative durations of connected tasks, multiple dependencies of a task with others) it is often possible to simplify the overall model compared to the respective ones of the tasks composing it, as certain parts of the individual models are repeated. In addition external constraints as resource sharing have to be modeled and may increase the duration of the simulation.

V. Case Study

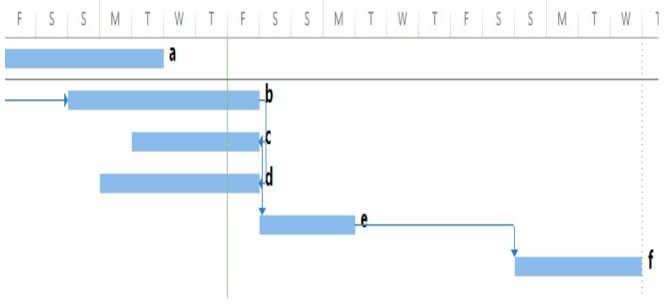
In this section, the proposed methodology is applied in a relatively simple example of Project, in order to show the applicability of the method. A Project consisting of 6 tasks is considered. Table I shows the types and the rest features describing the dependences between the tasks of the Project as well as tasks durations in days. From this, it is obvious that task A is the initial task of the Project (has no preceding tasks) and that tasks D and F are final tasks (are not followed by any other).

TABLE I. Task Precedence Matrix

Task name	Predecessor	Dependency	Duration
A	-		5
B	A	SS+2Days	6
C	B	FF	4
D	C	FF	5
E	C	FS	3
F	E	FS+5Days	4

The Gantt chart created with respect to the features described above is shown in Fig. 16. In this only the durations of the tasks are considered and not the days of the week and feast or holidays that will increase the overall period that is necessary for the execution of the Project. As it is obvious from this, the overall duration for implementation of the Project is 20 days.

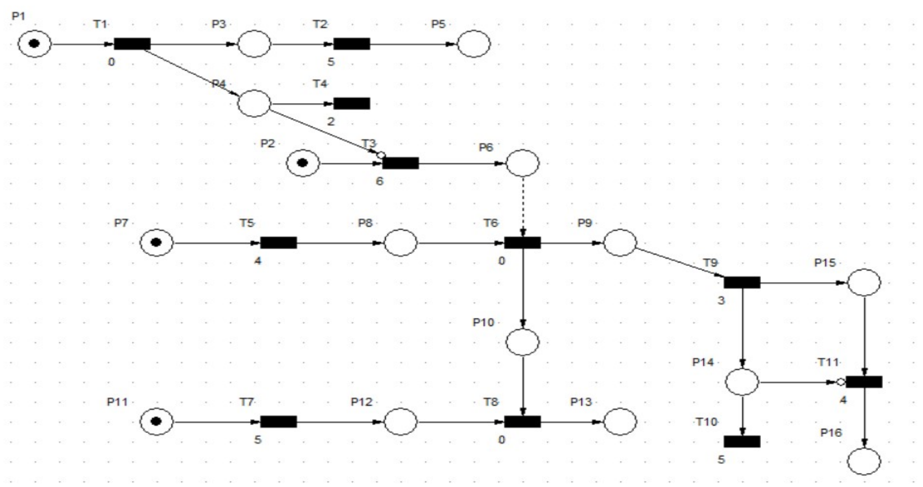
Figure 16. Gantt chart of the tasks of the Project under study.



The overall systems model is built using the appropriate models of the task dependencies from the ones presented above. It consists from 16 places and 11 transitions. These numbers are considerably lower than the expected by using the individual models, as certain branches were repeated and could be deleted in order to simplify the overall model. In particular, if the 5 individual models were used according to the precedence matrix (1 SS with lag, 2 FF, 1FS and 1 FS with lag), the sum of the nodes would be 24 places and 15 transitions. As it is obvious the reduction of the overall nodes used is about 30%. The time delays of the transitions are the ones of the lags and task durations as explained during individual models presentation and as they arise from Table 1 of task precedence. All the other transitions are immediate.

In the overall model of Fig. 16, simulation is terminated when a token reaches P_{16}. In the initial marking, tokens are placed in P_1, P_2, P_7 and P_{11} in order to start the simulation. With these parameters, simulation is terminated in 20 time units, representing days. This shows that the results obtained are correct since they agree with the ones of the Gantt chart of Fig. 16. Changes of parameters such as task durations and lags between tasks where considered, do not cause a change in the net structure as structure shows the dependencies between tasks and the transition parameters show the task durations and lags.

Figure 17. Petri net model of the overall Project according to the durations and dependencies of Table I.



VI. Conclusions

A modular methodology for representing task dependencies using Petri nets with arc extensions is presented. The proposed procedure can be applied for assembling the network in any generic Project Management process of any complexity and structure, following a number of well-defined steps. The major field of interest is Project Management that receives much attention nowadays. The main ambition of the author is to provide an alternative, considerably different from the popular network and node diagrams that can be used for calculation of the overall duration of complicated Projects. The modeling power of Petri nets makes possible the inclusion of random events in the model as well as the update of nets execution according to real time data that may be available from sensors, employees, general information and other sources. The proposed methodology follows well-defined rules and assumptions and may be adjusted to the demands of any system or activity of similar behavior by defining the appropriate fundamental systems Petri net models.

For this reason, Petri net models of the four possible types of task dependencies are defined and analytically presented and explained. Then a case study Project is considered and its overall model is built by connecting in an appropriate way the individual models of dependencies that are described in Precedence Matrix. The overall model is simpler than the individual models used to construct it, since certain parts are repeated and excluded and its simulation is terminated very quickly. All the models implemented are deadlock free as long as simulation is not terminated and the maximum number of tokens found in their places is one. Possible changes of arithmetic parameters do not cause changes in the structure of the overall model. Changes of the considered task dependencies however change the structure of the model.

Upcoming steps of this research concern inclusion of the models of the resources used in certain tasks in the overall model. This will produce more complicated models and will give more realistic results for the study of real world problems. In addition the use of colored Petri nets will be studied since the structure of the overall model may be simpler as certain parts of it that have similar structure may be represented by tokens with different colors and may transfer more useful types of data.

References

[1] W. Herroelen, and R. Leus, *Project scheduling under uncertainty: Survey and research potentials*, European Journal of Operational Research, vol. 165, pp. 289-306, 2005.

[2] J. R. Turner, F. Anbari and C. Bredillet, *Perspectives on research in project management: the nine schools*, Glob Bus Perspect, vol. 1, pp. 3–28

[3] G. J. Tuman, *Development and implementation of effective project management information and control systems*, in Cleland, D.I. & King, W.R. (eds.) Project management handbook. New York: Van Nostrand Reinhold Co., 1983, pp.495-532.

[4] H. Kerzner, Project management, *A systems approach to planning, scheduling and controlling*. New York: John Wiley and Sons, 2003.

[5] G. P. Prabhakar, "Projects and Their Management: A Literature Review", *International Journal of Business and Management*, vol. 3, 2009.

[6] L.E. Holloway, B.H. Krogh and A. Giua, "A Survey of Petri Net Methods for Controlled Discrete Event Systems," *Discrete Event Dynamic Systems*, vol. 7, pp. 151-190, 1997.

[7] R. Zurawski and M. C. Zhou, "Petri nets and Industrial Applicatrions: A Tutorial", *IEEE Transactions on Industrial Electronics*, vol. 41, No. 6, December 1994.

[8] T. Murata, Petri Nets: Properties, analysis, and applications, *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[9] R. David and H. Alla, *Petri Nets &Grafcet – Tools for Modeling Discrete Event Systems*, Prentice Hall, 1992.

[10] J. Wang, *Timed Petri Nets: Theory and Application*. Norwell, MA: Kluwer, 1998.

[11] K. Vrontakis, A. Kampianakis, and G. Tsinarakis, "A Petri net based methodology for modelling, analysis, demand forecast and optimal planning of batch production systems", *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 9-12 Oct. pp.1135 – 1141, 2016.

[12] G. Heerkens, *Project Management*, McGraw Hill, 2001.

[13] M. W. Newell and M. N. Grashina, *The Project Management Question and Answer Book*, Amacom, 2003.

[14] R. Drath, Visual Object Net software package, v.2.07 a 2002. Available: http://www.ParamSoft.de/

[15] M. Haji and H. Darabi, "Petri net based supervisory control reconfiguration of project management systems", *IEEE International Conference on Automation Science and Engineering (CASE)*,. p. 460-465, 2007.

[16] Y. L. Chen, P. Y. Hsu and Y. B. Chang, "A Petri Net Approach to Support Resource Assignment in Project Management," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 38, no. 3, pp. 564-574, 2008.