# THE DESIGN OF AN ONLINE PETRI NET BASED TOKEN (LUDO) GAME

Bakpo, F.S.
Department of Computer Science
University of Nigeria, Nsukka
E-mail: fbakpo@yahoo.com

**ABSTRACT**

The behavior of Petri nets with exponentially distributed firing times can be represented by labeled directed "slate" graphs in which labels describe the probabilities of transition firings (displacement of tokens) between vertices of the graph. The interactive firing of transitions in subsequent markings is known *as token game.* This development is analogous to that of games of strategy, in which each player is assigned a set of possible strategies (actions or moves) and each possible combination of strategies, one for each player, produces an outcome. That is, player's fortunes are intertwined and determined by chance events. The design of an interactive Petri net-based token (ludo) game is presented in which transition firing is determined by dice cast. Programming is considered and developed in Visual Basic 6.0 programming language, which is an object-oriented, event-driven and visual programming environment.

## 1.Introduction

There is something in the air of a gambling activity that always invigorates every type of persons from small to big. This is the element of chance or chanced event, that is, the possibility that luck will convert a pocketful of money into a mountain of wealth. A game thus consists of a set of players P = {Pi}, i ≥2, a set of actions A = {Ai}, i ≥1 for each player and a set of loss function L = {Li}, i ≥I. Game theory is about how multi-players or participants make decisions with or without the intervention of nature to pursue conflicting interests. It provides methods for identifying optimal strategies and predicting the outcome of strategic interactions [1,2,4]. The first comprehensive formulation of game theory started in 1944 with the publication of the book "Theory of Games and Economic Behavior" by famous mathematician John von Neumann and eminent economist Oskar Morgensten[2,4] Game theory has proven to be successful in many areas such as political sciences, military strategy, business, law, economics, biology, computer science etc. However, there have only been a handful of researchers applying Petri net theory to token (1udo) game. Infact, the only notable work in this direction was the one borne in the work at Kelvin Mcleish [8] which merely used the exponential transition firing to illustrate a Petri net token game animation. This in effect is not a real life game. Consequently, it seems proper to say that until now, there was no real life game devised on Petri net-based frameworks. Nearly all games require seeing patterns, making plans, searching combinations, judging alternative moves [1,11], etc. Petri net offers a viable modeling tool for implementation of the above game policy. The concept of Petri nets has its origin in Carl Adam Petri's dissertation "Kommunikation mit Automaton", submitted in 1962 to the Faculty of Mathematics and Physics at the Technische Universitat Darmstadt, Germany [9] Since then, the use and study of Petri net theory have increased tremendously. For a review of the history of Petri nets and an extensive bibliography, the reader is referred to [6.7.12.13]. A Petri net has been described in [8.13] as a convenient graphical and mathematical modeling tool allowing for easy representation of concurrency, synchronization and conflict among parts of the modeled system. As a graphical tool, Petri nets can be used as a visual communication tool similar to flowcharts, block diagrams etc. As a mathematical tool, it is possible to set up state equations and other mathematical models governing the behavior of a system. Areas of application of Petri nets include distributed data based system, communication protocols, performance evaluations of complex systems, workflow managements, etc.

## 2. Formal Definitions of Petri Nets

There are many equivalent formal definitions of Petri nets [7.12.13]. The following one will suffice for the purposes of this paper.

### 2.1 Marked Petri Nets

A Petri net is a triple N = (P, T, A) where:
P is a finite, nonempty set of places;
T is a finite, nonempty set of transitions;
A is a set of directed arcs, which connect places with transitions and transitions with places.
$\forall(t \in T)\exists(Pi, Pj \in P)(Pi, t) \in A\hat{}(t, Pj) \epsilon A.$

A place P is an input (or an output) place of a transition t iff there exists an arc (p, t) (or (t, p), respectively) in the set A. The sets of all input and output places of a transition t are denoted by Inp (t) and Out (t), respectively. Similarly the sets of input and output transitions of a place P are denoted by Inp (p) and Out (p). A place P is shared iff it is an input place for more than one transition. A net is conflict-free iff it does not contain shared places. Only

conflict-free Petri nets are considered in this paper.

A marked Petri net is a pair M = (N, m) where: N is a Petri net N = (P, T, A), m is an initial marking function which assigns a nonnegative integer number of tokens to each place of the net; m: P→(0.1). Figure I (a) depicts a marked Petri net[3].

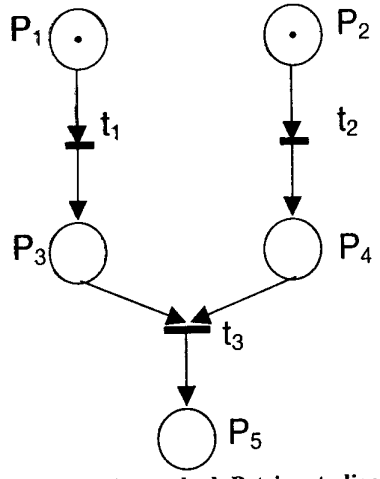


**Fig. 1(a): A marked Petri net diagram (before firing)**

It consists of places (circle), transitions (bar) and directed arcs (flow relations) that connect them. Input arcs connect places with transitions, while output arcs start at a transition and end at a place. A place may contain zero or more markings also called tokens or weights. The current state of a modeled system (the marking) is given by the number (and type) of tokens in each place. Transitions are active components and are used in modeling activities (events) which can occur (the transition fires), thus changing the state of the system (the marking of the Petri net). Transitions may fire only if they are enabled. When a transition is fired, it takes as many tokens from each of its input places as specified by the weight of the corresponding input arc and deposits as many tokens to each of its output places as indicated by the respective output arc weight. In other words, the number of tokens removed/added depends on the cardinality of each arc. For example, in figure I, places PI, P2 initially hold one marker each, as follows:

Before firing: $P_1$, $P_2$, $p_3$, $p_4$ ,$p_5$,= 1,1, 0, 0, 0.

During firing: $P_1$, $P_3$,: =$P1^{-1}$, $P_3$+1, if $P_1$, >0

$P_2$, $P_4$: =$P_2$-1, $P_4$+1, if $P_2$>0

$P_3$ $P_4$, $p_5$: =$P_3$-1, $P_4$-1, $P_5$+1, if $P_3$,>0 & $p_4$.>0

After firing: $P_3$, $P_2$, $P_3$, $P_4$, $p_5$: =0, 0, 0, 0, 1.

The resulting marking M' for the Petri net in figure 1 is 0, 0, 0, 0, 0, 1 (see figure 2).

The state of a Petri net is its marking. By firing a transition, the net changes state. A sequence of transition firings σ = $t_{j0}$, $t_{j1}$, $t_{j2}$, …$t_{jk}$ defines a sequence of markings reachable from the initial marking $M_o$ (i.e., *Mo→Ms* ). We call σa firing sequence, and a vector v (σ)ε N [T], a firing vector ofσ if the k-th component of v (σ) is equal to the number of occurrences of transition $t_k$ in σ.In this paper, each reachability set is referred to as a round of the game. A marked net M is bounded if a positive integer k exists such that each marking in the set M *(M)* assigns at most k tokens to each place of the net: $\exists(k > 0)\forall(m \in M(M)\forall(P \in P)m(p) < K$.If a marked net M is bounded, its reachability set M *(M)* is finite. This is similar to a win or a draw in the game. Only bounded Petri nets are considered in this paper.

An enable function of a marking m in a net N is any function e (m): T→7(0, 1, ... , 6). A random number choosing from dice casting determines this function

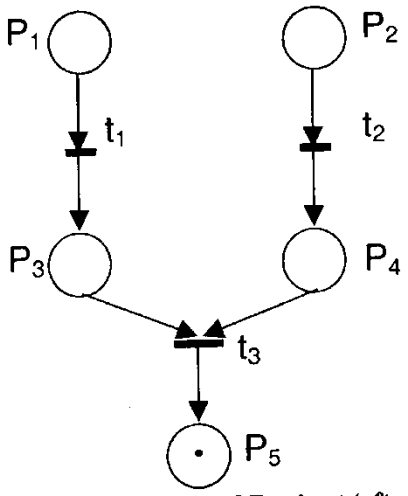


**Figure 1(b): A marked Petri net (after firing)**

## 2.2 Timed Petri Nets

In a timed Petri net, each transition t takes a positive time to fire. When a transition t is enabled, a firing can be initiated by removing a token from each of t's input places. This token remains in the transition t for the "firing time", and then the firing terminates by adding a token to each of t's output places. Each of the firings is initiated in the same instant of time in which it is enabled. For conflict-free nets all enabled transitions immediately initiate their firings since each marking of a net uniquely determines the enable function. The operation of a timed Petri net can thus be considered as taking place in "real time", and it is assumed that it starts at the time r=0. At this moment the firings indicated by the enable function e (m) are initiated and the tokens are removed from the corresponding input places. Then after the time determined by the smallest "firing time" of the transitions which initiated firings, the tokens are deposited in appropriate output places creating a new

marking, a new set of enabled transitions, and so on. The actual firing times of transitions can be described in several ways. In D-time Petri nets[7] they are deterministic (or constant) i.e., there is a positive (rational) number assigned to each transition of a net. In M-timed Petri nets or stochastic Petri nets[13], the firing times are exponentially distributed random variables, and the corresponding firing rates are assigned to transitions of a net. The memory less property of exponential distributions is also a key factor in the analysis of M-timed Petri nets. An M-timed Petri net T is defined in 13 as a pair T = (M, r) where:

M is a marked Petri net; M = (N,m), N= (P, T, A), r is a firing rate function which assigns a positive real number r(t) to each transition t of the net. r: $T \rightarrow R^t$ and $R^t$ denotes the set of positive real numbers. The firing time of a transition t is a random variable *v* (t) with the distribution function: Prob ( v(t) > x ) = $e^{-r(t)x}, x > 0$

The memory less property of exponential distributions means that if the duration *v* of a certain activity (e.g., the firing time) is distributed exponentially with parameter *r*, and if that activity is observed at time *y* after its beginning, then the remaining duration of the activity is independent of *y and* is also distributed exponentially with parameter r: Prob (v >y + x | v>y) = Prob (v>x) = $e^{-r.x}$

The exponential distribution is the only distribution with that property. Also, if *v* and *w* are the durations of two independent simultaneous activities a and b, distributed exponentially with parameter q and r, respectively, then the time interval *u* until the first completion of an activity (a or b) is distributed exponentially with parameter (q+r), and the probability that the activity 'a' will complete first is equal to q/(q+r), while the same probability for the activity 'b' is equal to r/(q+r). These results can be generalized in an obvious way to any number of activities. We called either of this result- a **winning probability**.

## 3. Petri Net Based Token Game Design

### 3.1 The Concept of Petri Net Token Game

As discussed in section 2, all Petri net models consist of two parts:

(i) The net structure that represents the static part of the system and;

(ii) A marking that represents the overall state of the structure.

The number of tokens (markings) at a place represents the local state of the place so that the marking of the net represents the overall state of the modeled system. The flow of tokens and the firing of transitions then model the dynamic behaviour of the system. Generally, transition firing involves the following steps:

(i) A transition is said to be enabled if each input place has at least as many tokens as the weight of the arc connecting them.

(ii) Enabled transition may be fired by removing from each input place the number of tokens equal to the weight of the arc connecting them.

(iii) When the transition is fired, tokens will be added to the output places connected to the transition.

The number of tokens to be added to each output place is equal to the weight of the arc joining them.

In this paper, transitions are related to external conditions, e.g casting of dice. This, chance event determines whether they may fire (and for, how long) or not when enabled. In other words, casting of dice triggers transition firings. The above-described mechanism is called "firing rule" or informally, "token game". For our purpose, this concept is applied to a ludo game that consists of multi-persons (two players) each of which has a **home place combined End place**. The motivation behind using Petri nets for the design of token (Ludo) game is stimulated by its modeling power, a graphical and mathematical arsenal supporting the analysis of a theoretical token game.

### 3.2 Functional Description of the Petri Net based Token Game

The basic elements or features of a Petri net theory that are harnessed for the purpose or this design include places (explicit fixed-points) transitions (represented by bars), directed arcs, and colored tokens. The token (ludo) game is a two-player game organized to take either of the following modes:

(i) Computer versus a single player, playing in-turn.

(ii) A single player versus oneself, playing in-turn.

(iii) Two players: player A versus player B, playing in-turn.

At start-up each player is prompted to choose a colored token. For instance, player A may choose red token while player B - green or any other. The token (Judo) game is captured in figure 2.

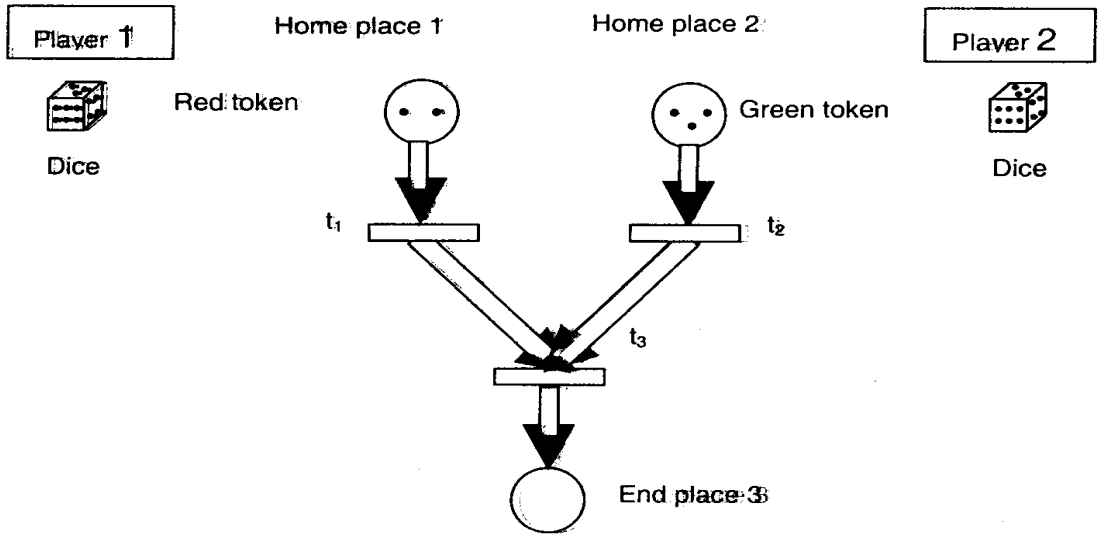


**Fig. 2: the Petri net token game configuration**

A Petri net-based game of two players: player 1(manipulating token l') and player 2 (token 2) consist of a series of rounds. Each player of the game-takes turn to cast his dice and each is equally likely to win. In earn turn, a move of token I is triggered by a number (n) shown by the dice cast at random, with $l \leq n \leq 6$, that is, n is a number shown on a face of a six-sided cast-dice. This number n corresponds to the weight of its arc and by it the: token (s) move automatically along a directed arc downward to a new place (or point). A possible (equivalent) move of player2 (token2) is also triggered' by a number n on his dice cast at random ($I \leq n \leq 6$), and is either <, =,or> that of player 1. This must be consistent with all constraints induced in previous turns, player 1wins the game, if his total set of moves (points) exceeds that of player 2, in which case, taken 1 consequently, arrived first at the End place. The following informal statement is observed for this finite game of two players1 and 2.that a player 1 is in a position where he/she wins against 2 is equivalent to that 1 has a move prior to and after which the game is not lost, but after which every move of 2 lead to a position where (again) 1 win against 2. In other words: a set position with winning strategy is a fixed point of some transformation of sets of position[11].The game may be declared draw, if the total set of points of 1 equals that of 2, in which case, a fresh round of the game could be initiated by the players. In this same vein, the game can also be played as many rounds as possible. Figure 3 depicts a typical moment of the game.

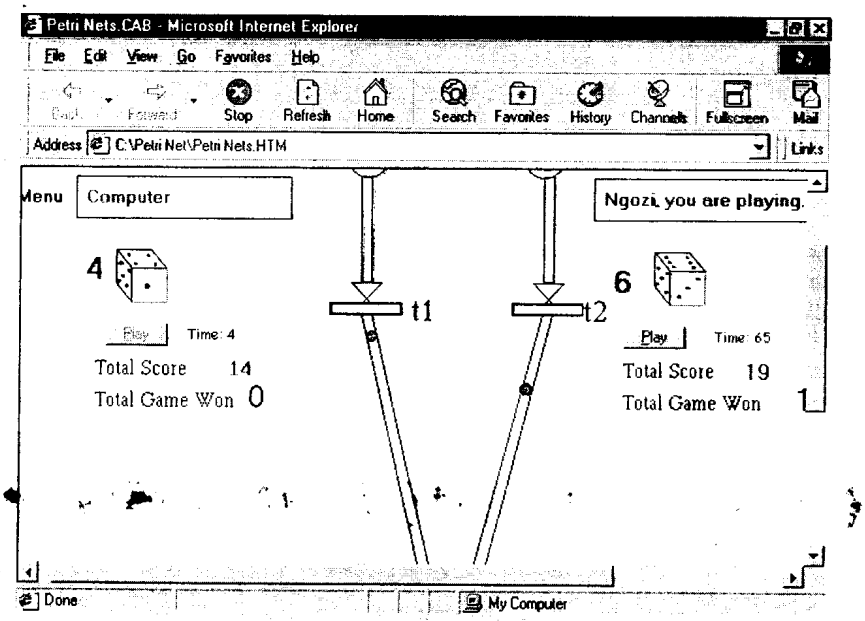


**Fig.3 A typical moment in the Petri net based game**

The above game policy conforms to the popular "game theory Assumptions" stated in [1] as follows:

1. Players or parties are fully rational.
   - Players attempt to maximize their utility/ outcome.
   - Players will accept the highest payoffs.
   - Players will only accept solutions that are at or greater than their security levels (resistance points).
   - Players know the "rules of the game".
   - Players assume other parties to be fully rational.
2. The number of players in fixed and known to all parties.
3. Communication is limited, highly controlled, or not relevant to the conflict negotiation interaction.
4. A decision must be possible that is maximally efficient, i.e, intersects with the solution set at a point that maximizes each party's own interests
5. **Implementation**

An important issue for user support in a new design is adaptation. Adaptation is an attempt to bridge the gulfs between the system and the user, either from the user's side or from the system's side [5,10]. In the former, the user's behavior (possibly problematic) is corrected to fit that required by the system through training, additional documentation, etc. In the later, the system is modified so as to adapt to the user's needs and requirements. In this view, the system is the one at fault and the user is behaving sensibly. We are interested in adaptation of the second form because of our interest to accommodate user's preferences and user's preferences and user's growth. User comforts and user preferences address the issue of enabling users to do things in ways that they prefer and are comfortable with. Techniques like user profiles, aliases, macros, and shell command scripts allow the user to customize and extend the basic repertorire of interactions available to him. An equally relevant consideration is a facility that enables a user to configure workspaces and to switch between them adeptly. Windows (i.e. partitioning the display into a number of virtual screens) are extremely useful and powerful for this purpose, and are considered topmost in this paper. They currently provide the best kind of support for concurrent activities. A starting point in developing the pertri net- based (ludo) game is the functional decomposition of the overall system. The design of the game is based on the following model/object decomposition (see figure 4)

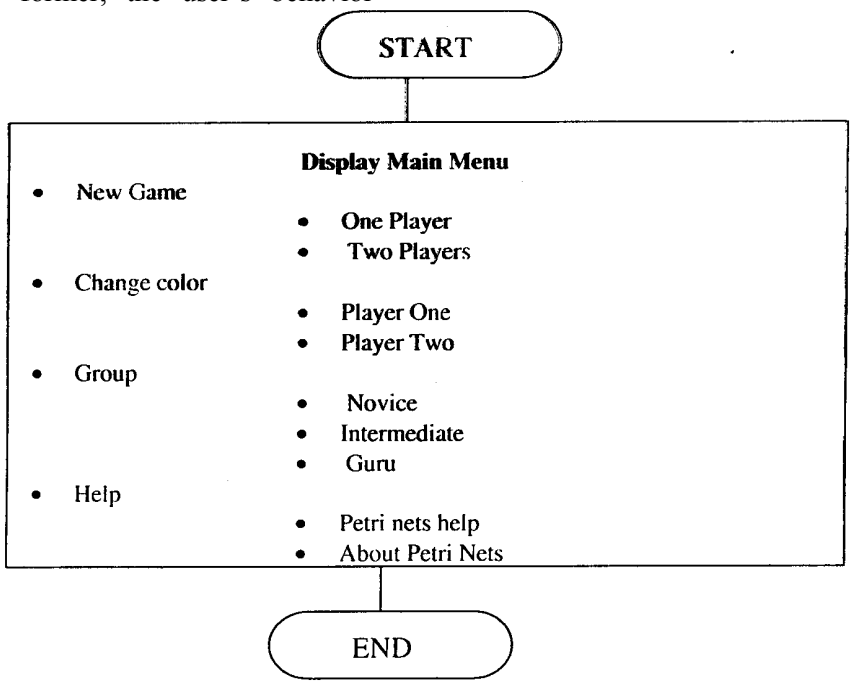


**Fig. 4: The Game Objects Design**

To play the token game, the user clicks on the menu icon and a pop down menu is displayed giving the players the following choices:

(i) New Game (One Player/Two player) - possibility to play with the computer or with a partner.

(ii) Change color (Player one/ Player Two) - Possibility to change colors of the tokens.

(iii) Group (Novice/Intermediate/ Guru) - Possibility for most experienced players.

Help *(Petri Nets Help/about Petri net)* - Provision of helpinformation on petri nets. 1chose to implement this game in visual basic 6.0 language to take advantage of its diversity of random number generation, OOP, visual/event-driven and a shorter coding-debugging life cycle. The methods of Petri nets that constitute the game, such as dice casts (), Transition firing (), token movement (), place sets (), in Turn (), e.tc are component that perform a specific programming task. These components are defined as separate objects. A fragment of the programming implementation follows (see figure 5):

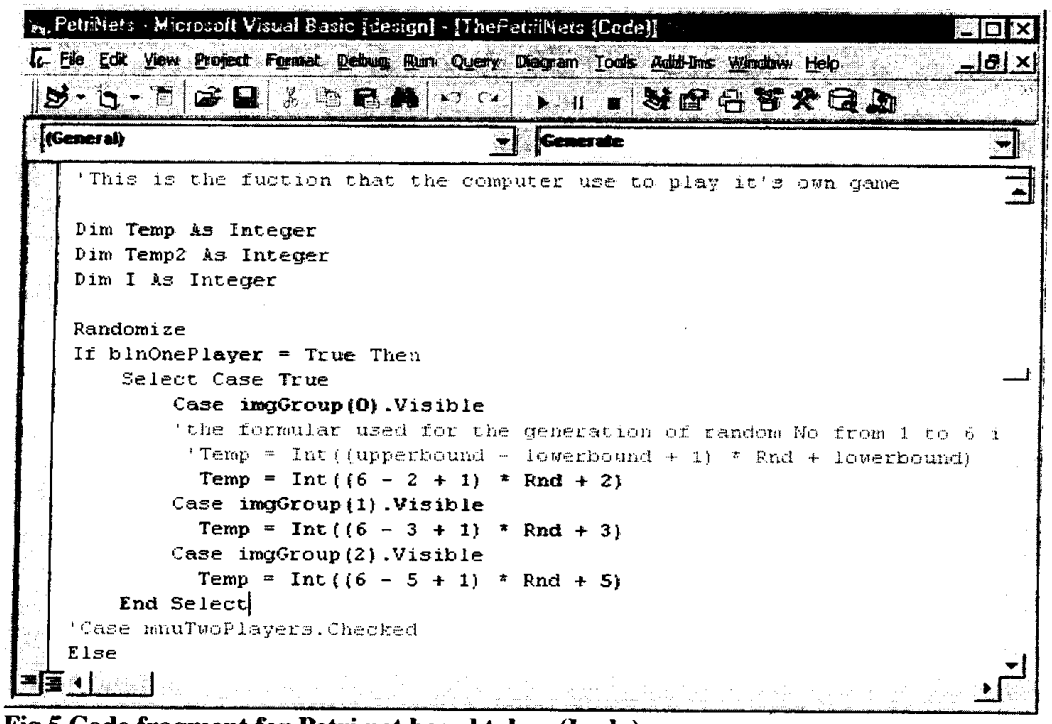


**Fig.5 Code fragment for Petri net based token (Ludo) game**

The Petri net based ludo game have been played severally in the Department of Computer Science at University of Nigeria, Nsukka with both staff and students finding it very interesting and exciting. With its online facility, home users worldwide could be motivated to engage with PC via the game, just like video films engaged people with watching TV.

## 5. Conclusion

An informal real-life computer (ludo) game based on the framework of Petri nets is presented and implemented in a programming language VB 6.0, that is object-oriented, event-driven and offered suitable visual tools. In order to make the game available worldwide, I have prototyped it as a web- based. From my experience, I believe that the framework is useful and promising. However, the idea of designing a real-life game based on Petri a net technique is still new, and I am still exploring its capabilities and limitations. This work promotes the current understanding of the theory of Petri nets.

## References

1. Adam, B(2003)Game theory and Business Strategy" Online:

2. Adam, B (1992) "Knowledge and Equilibrium in Games" *Journal of Economic Perspectives 6. 83-101*

3. Bakpo, F.S. (2003) "Software Design Modelling with Functional Petri Nets" Nigeria *Journal o] Technology [NIJOTECH*] *University of' Nigeria, Nsukka, (Accepted for Publication)*

4. Banerji, R (1987) "Game Playing: In Encyclopedia of Artificial Intelligence" Vol l , ed. Shapiro, Stuart C, 312-319. New York: Wiley & Sons.

5. Deitel, H. M., Deitel, P. J and Nieto. T. R (1999) Visual Basic 6 How To Program" *Prentice Hall. Upper saddle River New Jersey. 07458*

6. Erden, Z., Erkmen, A.M and Erden, A (1998). "A Petri Net-Based Design Network with Applications to Mechatronic Systems" *SDPS Transactions, Journal of Integrated Design*

*and Process Science. Vol.* 2, *No* 3. *pp* 32 - 48.

*7.* Guroglu, S., (1999). "Implementation of an Algorithm for a Petri Net Based Design Inference Network". *M .Sc. Thesis. METU,TURKEY*

8. Kelvin, M (2003)"petriNets" http://www diaimi.au. dk/Petri Nets/tools/.

9. Petri, C A (1962) "Kommunikation mit Autornaten" *Ph D thesis institut fur instrumentelle Mathematik, Bonn*

*10.* Podlin, S. J. and Palmer P (2000) "Hands on Visual Basic 6"*Tech Publications pte Lid. Singapore*

*11.* Shilov, N. V and Yi, K (2003)" Engaging Students with Theory Through ACM Collegiate programming .contests" *Communications of the ACM*, 45, 98- *101*

*12.* Van der Aalst, W.M. P. (2002)" The Application of Petri Nets to Workflow Management" On Internet: h*ttp://psim.tm.tue.nl/staff/wvdaalst/Publicalionls /p53,pdf*

*13.* Zuberek, W. M (1985)"Performance Evaluation of Concurrent Systems using Timed petri Nets." *Proceedings of the* 1985 *ACM Computer Science Conference Agenda* for *Computing Research. P326-329.*