We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

State of the Art in Interactive Storytelling Technology: An Approach Based on Petri Nets

Hussein Karam Hussein Abd El-Sattar

Additional information is available at the end of the chapter

1. Introduction

Interactive Storytelling is one of most promising technologies for the development of new media and new forms of digital entertainment. Interactive Storytelling can be defined as the endeavour to develop new media in which the presentation of a narrative, and its evolution can be influenced, in real-time, by the user. In traditional forms of storytelling, a storyteller would present the scenario of a story to the audiences in a predefined way (also known as a plot), which limited the variation in character interactions and context. The story refers to the succession of actions that happen in the world represented by the narrative. The origin of computerized storytellers can be traced to the "story-generation system TALE-SPIN" (Meehan, J., 1981) which is the most popular generator of short tales. A scene of the storyline is composed of beats, where the term beat refers to the smallest unit of action that has its own complete shape. A beat is a dramatic action that occurs in a scene to achieve a narrative goal. It consists of

i.   Pre-conditions, a list of predicates that need to be true for the beat to be selected;
ii.   Post-conditions; a list of predicates that will be true as a consequence of firing the beat;
iii.   Success and failure conditions, and
iv.   A joint plan to be executed by the actors.

The growing interest in Interactive Storytelling as a Research topic and as a potential technology derives from the current competition between traditional and interactive digital media. Interactive Storytelling is of interest to broadcasters and computer game producers alike. In the former case, Interactive Storytelling will bring interactivity into traditional media, potentially revolutionizing the entertainment experience. In the latter, it would improve the narrative and "aesthetic" value of computer games, with the potential to develop new game genres and attract a wider audience. The term Interactive Storytelling is sometimes used in a broader sense in the field of new media, to reflect the fact that some

interactive systems, like computer games, can be designed to reflect an underlying narrative, which is conveyed via the gameplay, often enhanced by animation cut scenes.

On the other hand, several years of research are establishing Petri nets (PN) (Tado, M., 1990; Peterson, J. L., 1997) and its extension High-level PN as a process modelling formalism for a variety of application domains. A high level PN is a Petri net extended with colour, time, and hierarchy (Jenson, K., 1997). The power of Petri nets lies in its formal semantics and non semantics properties. This chapter addresses a new application domain of PN for modeling game systems and game workflow control in the context of workflow management concept and game rules principles. It presents state-of-the-art results with respect to interactive storytelling and PN, and highlights some petri-net-based workflow tools for game design. This is done by proposing an integrated framework for deeply combining interactivity and narrative in computer games which is achieved by:

i. composing the game rules in the game's workflow environment by different triggers and effects, and

ii. separating the rules of the game environment into two parts: controllable rules, and uncontrollable rules. The controllable rules are the rules in which its template is directly related to the goal of the game, mainly as a feedback within the rule effects. The uncontrollable rules are the rules in which its template is independent from the game goal. The rule is then characterized by a trigger based on the computer game's input, and an effect targeting only the game elements.

Evaluation and performance results supported by some case study called crazy ball 2 are also demonstrated. Crazy ball 2 is a platform-type genre, much like the worldwide-known game Mario and the Konami's Castlevania series.

In the reminder of this chapter we will offer insights into how workflow management concepts can be jointly utilized with Petri nets (PN) for modeling game systems and game workflow control. To do this, we first introduces the problem statatment, prior research and objectives in section 2. Section 3, briefly summarizes the background and the basic terminology and notions that will be used throughout this chapter. Section 4 introduces the methodology used. Evaluation and practical performance results are discussed in Section 5. Sections 6 concludes this chapter and outlines some directions for future work.

2. Prior research, problem statement, and objectives

2.1. Prior research

Building interactive storytelling systems is gained a growing attention among a growing number of researchers from a huge diversity of discipline and orgins of expertise. As a result, many different approaches which differ on various dimensions and sometimes overlap are developed under the following four and some other research directions:

i. Generative computer graphics, animated storytelling for film production;
ii. Human-computer interaction (HCI);

iii. Computer game design, and

iv. Artificial intelligence.

Story plot and character(s) are the two most important element of a story. As a consequence, several approaches, at different levels of complexity, have been developed for representing plots in games, monitoring the course of the story and controlling stories in games and storytelling applications. These techniques are summarized as follows:

i.   Planning techniques (Riedl, M. & Stern, A., 2006);

ii. Beat approach (Brom, et. al., 2007);

iii. Finite-state machines (Sheldon, L., 2004), and

iv. Petri nets (PN) (Delmas, G. et. al, 2007; Brom, C. & Abonyi, A. 2006; Karam, H., 2010).

Several years of research are establishing Petri nets (PN) and its extension High-level PN as a process modelling formalism for a variety of application domains, for instance: network protocols, logistics, scientific workflows and gaming theory. Their power lies in their formal semantics and non semantics properties. The main contribution of this chapter is to show how workflow management concepts can be jointly utilized with Petri nets (PN) for modeling game systems and game workflow control. This is done by composing the game rules in the game's workflow environment by different triggers and effects. The idea is derived from the study of PN, game theory, workflow management, story writting, AI, and cinematography in interactive storytelling. In this contribution, interactive storytelling is viewed as a hybrid form of game design and cinematic storytelling for entertainment applications among two skills: artistic and technical. Evaluation and performance results in terms of some case study called crazy ball 2 are also demonstrated. Crazy ball 2 is a platform-type genre, much like the worldwide-known game Mario and the Konami's Castlevania series.

2.2. Problem statement and objectives

Recently, interactive storytelling has become a major issue in video games development. Several categories of video games arose to either historical, editorial or narrative criteria. Within the field of Interactive Storytelling, interactive drama is a computer-based fiction where a user chooses most of the actions for the main character in a story. Interactive drama is the ultimate challenge of digital entertainment because it involves both the dynamic generation of narrative events and the integration of user inputs within the generation. This is a hard challenge, because it involves both the dynamic generation of narrative events and the integration of user inputs within the generation. Moreover, both Storytelling unfolding and player's interaction can't take place at the same time. The first relates to game designer's control of the game he/she has created as the second relates to player's control on the game he/she has bought. Each interactive drama needs a model of narrative. The challenge of interactive drama is to find a model suited to the interactive nature of computers. Interactive drama architecture has several key components: the environment, the player, the user, the writer, and the director. For successful interactive drama architecture, three requirements are necessary (Magerko, B. & Laird, J. 2003):

i.   the balance between writer flexibility vs. user flexibility;

ii.   temporal variability (i.e. allowing time to be the key variable for the flexibility in an interactive experience), and

iii. transparency (How do we encourage the User to follow a particular destiny without having him feel forced into it?).

Moreover, it is necessary to overcome two obstacles (Szilas, N., 2005): technical problem and conceptual problem. This chapter attempts to solve these problems by proposing an integrated framework for deeply combining interactivity and narrative in computer games. The approach is based on separating the actions of the system into two parts, the controllable and the controllable actions. The controllable actions are controllable by the system we model. The system can choose which and when to execute controllable actions. The controllable actions are not controllable by the system, but can occur whenever they are enabled. In this contribution, interactive drama is viewed as a hybrid form of game design and cinematic storytelling for entertainment applications among two skills: artistic and technical. The idea is derived from the study of interactive drama, Petri nets (PN), narrative structures in computer games and game workflow activity process. The main advantages of using PN are that it copes well with branching stories and can evolve in parallel in large virtual world. The proposed idea is supported by some case study called Crazy ball 2. Crazy ball 2 is a platform-type genre, much like the worldwide-known game Mario and the Konami's Castlevania series. It possesses many universally-shared game features such as Hit Points (HP), Game Over, Enemies, and Bosses. One of the advantages of the proposed computer game is the inclusion of the "Freestyle combat system", which allows the user to completely control the attacks of the player using a mouse. Moreover, it possesses a feature which is called an "even game", in which the game challenge level matches the skill of the human player.

3. Fundamentals and basic notions

3.1. Petri Nets

Petri nets are a "pinball game" for mathematicians (Tado, M., 1990; Peterson, J. L., 1997). It is a particular kind of directed graph, together with an initial state called the initial marking M_0. The underlying graph G of a PN is a directed, weighted, bipartite graph consisting of two kinds of nodes, called places (P) and transitions (T), where arcs are either from a place to a transition or from a transition to a place. N=(T,P,F) is called a net, iff

$$\begin{cases} (i)P \cap T = \emptyset, and \\ \\ (ii)F \subseteq (P \times T) \cup (T \times P) \ is \ a \ binary \ relation \end{cases} \tag{1}$$

Let N be a net and let $x \in (P \cup T)$, then

$$\begin{cases} {}^{\cdot}x = \{y | yFx\} \ is \ called \ the \ set \ of \ pre-conditions \\ x^{\cdot} = \{y | xFy\} is \ called \ the \ set \ of \ post-conditions \end{cases} \tag{2}$$

Formally, a Petri nets $PN = (N, M_0)$ consists of a structure N and an initial marking M_0, where:

i.   N=(P,T,F,W) is a Petri nets structure,

ii. $P = \{p_1, p_2, \cdots, p_m\}$ is a finite set of m places,

iii. $T = \{t_1, t_2, \cdots, t_n\}$ is a finite set of n transitions,

iv. $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs with $P \cap T = \phi, P \cup T \neq \phi$.

Arcs are labeled with their weights (positive integers), where a k-weighted arc is interpreted as a set of k parallel arcs. Labels for unitary weight are usually omitted.

$$W : F \rightarrow \{1,2,3,\cdots\}$$

is a mapping which associates to each arc of the net its weight,

$$M_0 : P \rightarrow \{1,2,3,\cdots\}$$

is the initial marking representing the initial state of PN. The state of PN is described by means of the concepts of marking. A marking is a transition function that assigns to each place a nonnegative integer called a token. A token is the main information unit and a primitive concept of PN like places and transitions. If a sufficient number of tokens are contained in specific places, an action is triggered. After firing an action, the tokens that helped to fire this action are removed, and some new tokens are generated. Which tokens fire which action and which action generates tokens to which places is specified by the transition function. In graphical representation, places, transitions, tokens and transition function are represented by circles, rectangles or bars, pellets and arrows respectively. Table 1, illustrates such representations and its interpretations of a PN model. Figure 1 presents an example of a PN model with graphical and mathematical notations.

In modeling using PN, we regard the places as conditions, the transitions as events or actions, and a marking as triggers. A trigger can be associated with an action (action trigger) or a place (place trigger). A transition has a certain number of input and output places representing the pre-conditions and post-conditions of the event, respectively. The presence of a token in a place is interpreted as holding the truth of the condition associated with the place. In another interpretation, "k" tokens are put in a place to indicate that "k" data items or resources are available. Some typical interpretations of transitions and their inputs and outputs places are shown in Table 2.

PN Model	Graphical representations	Interpretations
Places	○	States/conditions
Transitions	▭	Events/actions
Tokens	●	Marks/States
Transition function	→	Triggers

Table 1. Graphical Representations and Interpretations of a PN Model.

$P=\{P1,P2,P3,P4\},$
$T=\{t1,t2\},$
$F=\{(P1,t1), (P2,t2), (P3,t2),$
$(t1,P4),(t2,P4)\},$
$W(P1,t1)=2,$
$W(P2,t2)= W(P3,t2)=$
$W(t1,P4)=W(t2,P4)=1,$
$M_0 = [1 \quad 1 \quad 1 \quad 0]^T.$
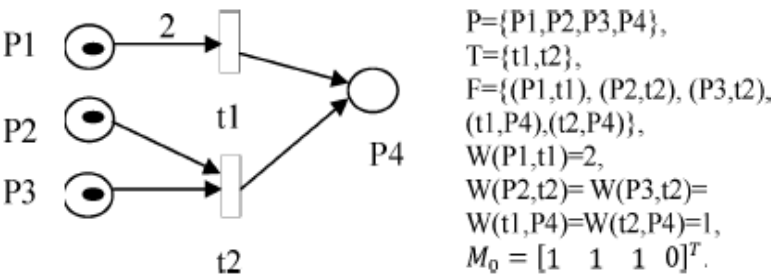
**Figure 1.** An illustration example of a PN model with graphical notation (left), and mathematical notation (right).

Input Places	Transition	Output Places
Pre-conditions	Event	Post-conditions
Input data	Computation step	Output data
Conditions	Clause in logic	Conclusion(s)
Resources needed	Task/job/behavior	Resources released

Table 2. Some typical interpretations of transitions and places.

The dynamics of the net is described by moving tokens among places according to the transition firing rules or enabling test in marking. A transition t is said to be enabled if each input place P of t is marked with at least W (P, t) tokens, where W (P, t) is the weight of the arc from P to t. Once enabled, a transition will fire when its associated event occurs. A firing of an enabled transition t removes W (P, t) tokens from each input place P of t, and adds W (t, P) tokens to each output place P of t, where W (t, P) is the weight of the arc from t to P. Figure 2 demonstrates an example for the transition firing rules of a PN model shown in Figure 1. In the PN of Figure 2, only transition t2 is enabled; t1 is not enabled because it would require two tokens in P1 to fire, since W (P1, t1) =2. When t2 is fired, the tokens in P2 and P3 are removed and P4 receives one token (see Figure 2).
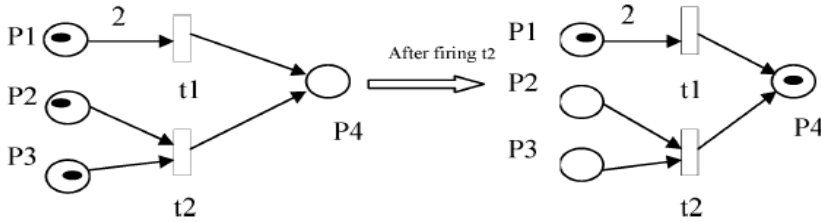

Figure 2. PN marking example before and after firing the enabled transition t2.

### 3.2. Workflow management concept

Workflows are gaining a lot of interest both in the business and scientific environments for automating the execution of complex IT processes. The term workflow management (WM) refers to the domain which focuses on the logistics of business processes (Van der Aalst, W., 1998). The goal of workflow management is to handle cases as efficiently and effectively as possible by the right resource (e.g. actor, person, participant, or application) at the right time. Workflows are case-based, i.e. every piece of work is executed for a specific case. The

key concept of WM is task. A task is a piece of work to be done by one or more resources in a pre-determined time interval. A task item is a task that needs to be executed to handle a specific case. Task items are executed by resources. Synonyms for resource are actor or participant. A work item which is being executed by a specific resource is called an activity, which is an actual performance of a work item. The activity contains the rule entities to describe the conditions under which the activity may be fired. A workflow procedure defines a partial ordering of tasks to handle cases of a specific type. A workflow process definition comprises a workflow procedure, a set of resources and a strategy to map task items to resources. Figure 3 shows that a workflow has three dimensions: case, resource and process definition dimension. A number of dots shown represent either a work item (case+task) or an activity (case+task+resource). Note that, work items link cases and tasks. Activities link cases, tasks, and resources
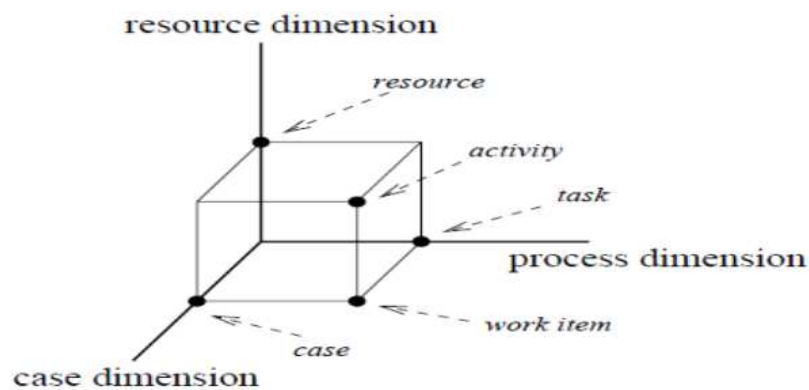

Figure 3. A three dimensional view of a workflow

A "workflow process definition" dimension specifies how the cases are routed along the tasks that need to be executed. Routing of cases describes the lifecycle of a case, i.e., which tasks need to be performed and in which order. Four types of routed are identified as shown in Figure 4: sequential (tasks are executed sequentially), parallel (task B and C are executed in parallel), conditional (either task B or C is executed), and iteration (multiple B's). These four routed types outlined the currently accepted narrative structures used in the creation of computer games (Mark Reidl, O., Michael R., 2005). For instance, linear narrative which corresponds to the sequential routing is a traditional form of narrative in which a sequence
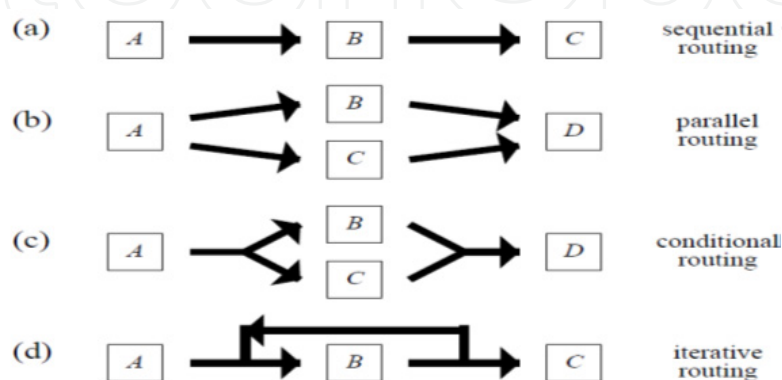
Figure 4. Four routing constructions demonstration.

(a) Linear Narrative Progression   (b) Branching Narrative Progression
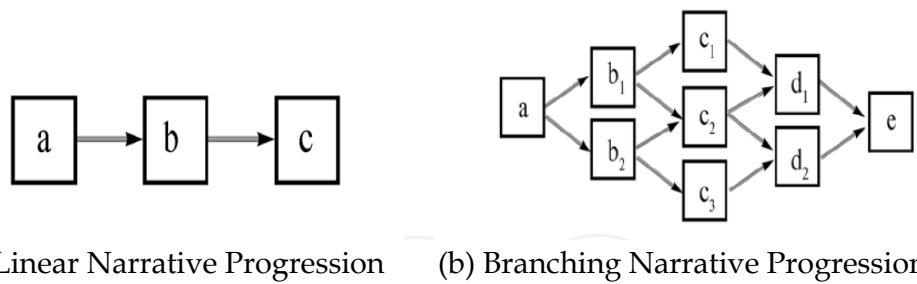
Figure 5. An illustration for both linear and branching narrative progressions.

of events is narrated from beginning to ending without variation or possibility of a user altering the way in which the story unfolds or ends. Some computer games use branching narrative in which there are many points in the story at where some action or decision made by the user alters the way in which a narrative unfolds or ends. Both narratives are shown in Figures 5(a)-(b).

Since Petri Nets (PN) are a process modeling technique, then modeling a "workflow process definition" onto PN is straightforward: tasks, conditions, and cases are modeled by transitions, conditions and tokens respectively. Figure 6 shows how all these concepts can be mapped onto the Petri net. Table 3 shows the mapping between the workflow process definitions and PN terms.
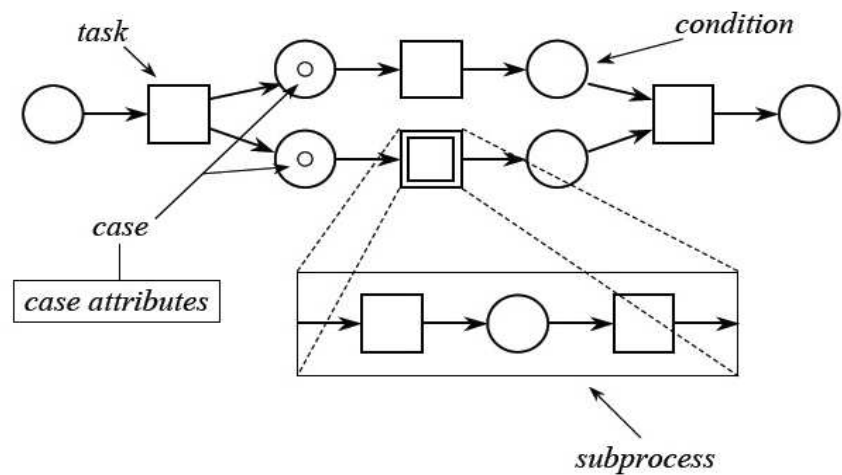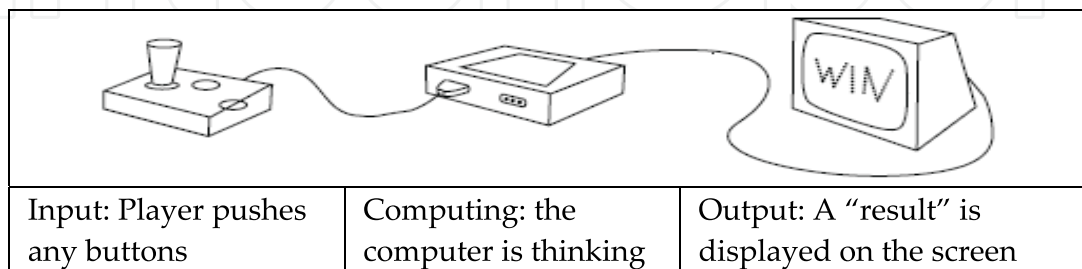

Figure 6. Mapping a process definition onto Petri nets.

Workflow Perspective	PN terms
Task execution	One or more transition
Work item	Transition being enabled
Activity	Firing of a transition
Data that flows between tasks/case attributes	Tokens
Conditions	Places

Table 3. Mapping Workflow process onto Petri nets (PN) terms.

4. Methodology

Computer games are defined as an activity with some rules engaged in for an outcome. Key components of games are goals, rules, challenge, and interactivity. Games are regarded in terms of three levels of temporal design, from simulation at the lowest time scale, through the design of game moves above the simulation level, to the structure of specific narrative patterns at the highest level. The methodology in our approach is based on the study of workflow process under these levels for the structural parts of a computer games and is shown in Figure 7.

| Input: Player pushes any buttons | Computing: the computer is thinking | Output: A "result" is displayed on the screen |

Figure 7. The structural parts of computer games.

The diagram in Figure 7 is composed of three parts: the "Input" peripheral devices allowing the user to enter choices. These choices are then evaluated by the rules of the "Computing" part, in order to produce a "result". This result is finally communicated to the player through the "Output" device. Computer game development is a tremendous task that requires different roles for the storyteller and audience, and a lot of resources including: storytelling elements and elemental interplay. Storytelling elements includes: narrative, plot, and story. The interplay of elements in the game environment consists of: structure, theme and metaphor. The structure describes the influence of plot on narrative. Theme is the expression of the story within the plot. Metaphor is the means by which story is encoded within the narrative. From the technical point of view, game can be seen as a client-server application connected with some data. From the conceptual point of view for game representation and according to (Salen, K., Zimmerman, E., 2003), computer games are defined as an activity with some rules engaged in for an outcome. Therefore, there is a need to develop an integrated framework for deeply combining interactivity and narrative. As a sort for achieving this goal and under the study of workflow management concept, Figure 8 shows a design of a workflow activity process for the conceptual representation of computer games.

Figure 8 shows how the conceptual representation of a game can be modeled as a series of processes, where each process consists of some activities. Activity is a description of a piece of work that forms one logical step within a process. An activity has one or multiple roles, zero or multiple rules, zero or multiple event and one or more multiple data. Partial order relationships between activities (e.g. sequential, parallel, alternative and looped) are notated with the arc, which links the activity entity to itself. The activity also contains the rule entities to describe the conditions under which the activity may be fired. The rule is used to
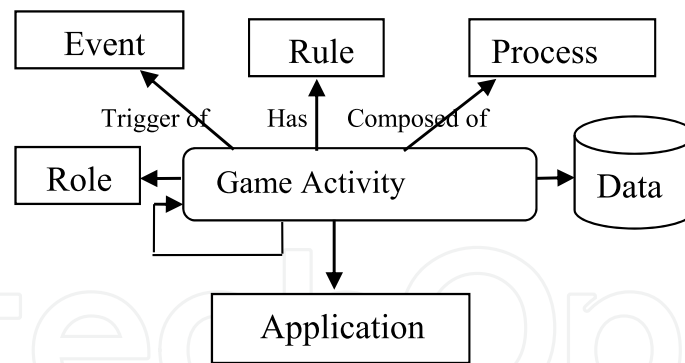
Figure 8. Game workflow activity process

describe logic conditions for executing activity and always related to case "data". The "application" entity extracts software tools, by which the role completes the activity, and the "data" depicts the input and output data of the activity. Since the game engine we are going to propose in this paper executes workflows represented in terms of PN. The classical structure of a PN model is formally defined by a set of places, a set of transitions and a set of arcs connecting places to transitions and vice versa. For the purpose of this chapter we extend the classical PN with features that make them more suitable for workflow representation given in terms of High level Petri nets (HLPN) workflow (Jenson, K., 1997). An HLPN is normally represented using a graphical form which allows visualization of system dynamics (flows of data and control). By using HLPN formalism it is possible to

i.   formally represent the workflow state (and its evolution);
ii. formally describe both the control and the data flow, and
iii. deal with dynamic workflows.

An HLPN can be defined by the tuple: HLPN= (P, T, Type, Pre, Post, E, G, M_0), where

- P is a finite set of elements called places
- T is a finite set of elements called transitions
- "Type" is a function that assigns a type to places and expressions;
- $\Pr e \subseteq (P \times T)$ is the subset of arcs from places to transitions and $Post \subseteq (T \times P)$ is the set of arcs from transition to places
- E is an expression. It is defined from $(\Pr e \cup Post)$ into expressions. Expressions may comprise constants, variables (e.g. m,n) and functions (e.g. g(x)) which are types;
- G is the guard function, a Boolean expression inscribing a transition $(t \in T)$ (where Type(G(t))=Boolean);
- M_0 is the initial marking: a multi-set of tokens associated with the places

Now let us analyze the game workflow activity shown in Figure 8 in terms of HLPN and game components. Key components of games are goals, rules, challenge, and interactivity. Games takes place in a virtual universe and it is composed by several elements, where these elements are submitted to "rules", in accordance to the game. Since a workflow system is a reactive system, i.e. it is triggered by the environment. This means that, enabling of a task doesn't imply that the task will be directly executed. Therefore, the objective is to analyze

the workflow management process of games rules and an attempt to classify them in terms of the game elements constitutes the game environment. By focusing on games as rules we means looking at games as reactive systems, both in the sense that the rules are inner structures that constitute the games and also in the sense that the rules schemas are analytic tools that mathematically dissects games. As a result, we found it is necessary for game developers to distinguish between the enabling of a task and the execution of a task. Since the enabling of a task does not imply that the task will be executed immediately, it is important to have this distinction. In order to have this distinction, we should consider triggering of tasks in the game's flow environment. A trigger is an external condition which leads to the execution of an enabled task. One way to perform this is to separate the rules of the game environment based on whether its template is direct or indirect related to the goal of the game into two parts: controllable rules, and uncontrollable rules. Controllable rules are the rules in which its template is directly related to the goal of the game, mainly as a feedback within the rule effects. In this case, the rule is characterized by a trigger based on the state of the game elements, and an effect linked to the computer game's output. The uncontrollable rules are the rules in which its template is independent from the game goal. The rule is then characterized by a trigger based on the computer game's input, and an effect targeting only the game elements. An example for the template is the one given by: "if player element collides with a hostile element, then there is a negative feedback towards the player element". The real power of separating these rules illustrates how the separation of them allow us to explicitly specify what parts of the model comprise the system and what parts comprise the environment. For this purpose we used four types of tasks:

i.   automatic: a task is triggered the moment it is enabled (i.e. No trigger is required);
ii. user: a task is triggered by participant/actor;
iii. Message: an external event is required, and
iv. time: the task requires a time trigger.

An awareness of a narrative's time line is an important element to focus on when creating the environment for an audience's imagination. The triggering concept can be modeled in terms of HLPN for user task "A" as shown in Figure 9.
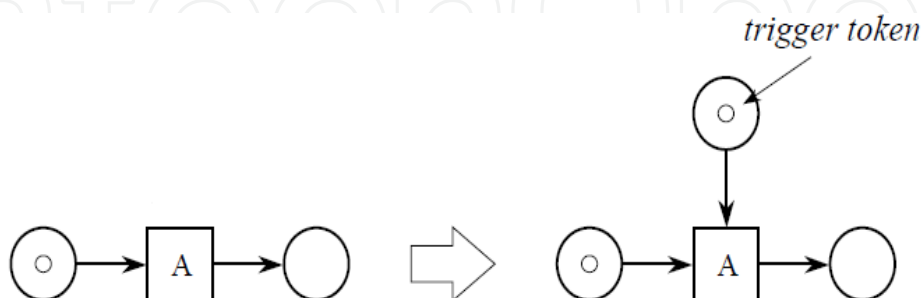
Figure 9. The triggering concept for user task called "A" workflow activity process

Since game is defined as an activity with some rules engaged in for an outcome. So, game activity can be represented now as: activity = task + case + (resource) + (trigger), where

trigger is one of the previous mentioned tasks. This activity represents the actual execution of a task for a specific case. Now let us analyze the effects of trigger concept and game rules (controllable/uncontrollable) applied to the proposed case study game "crazy ball 2". Games takes place in a virtual universe and it is composed by several elements, where these elements are submitted to "rules", in accordance to the game. For example, table 4, shows some of the universe "elements" applied for the case study game "crazy ball 2" and its uses. These elements are submitted to different rules, for instance, "if the red ball element jumps towards the player, then it explodes on touch". By analyzing this rule, we realized that it is composed of two parts: (i) the "trigger": "if the red ball element jumps towards the player,", and (ii) the "effect(s)": "then it explodes on touch". In the same logic, the goal of a game can also be described and controlled by its rules.

Game Elements	Uses
	Used in the combat system to control the player using mouse and also used for the render that made the Game Over screen
	Used as a model for Bad Ball and Red Ball enemies.
	Used as the door object in the game's stage, it has an animation that allows the grey stone part to slide open upwards.
	Used as the model for the player's hit points (HP). Eight heart graphics models are used to display player's HP on the screen.
	Used as the Information Signs' object. The question mark has a rotation animation that is constantly active in the game.
	Used for setting the sword swing power (force) display
	Player control
	Game envirnoment

Table 4. Some game elements for the Crazy ball 2 game and its uses.

Now, the question is how can we construct game rules that include game goals? The answer to this question is derived from the study of both HLPN and game workflow activity process previously mentioned. As a result, we defined the "game elements" as "a canvas of rules", a diagram to follow in order to build a rule or a group of rules in a computer games. This is done by composing the game rules by different triggers and effects as shown in Figure 10. The game workflow activity engine forr crazy ball 2 is shown in Figure 11.
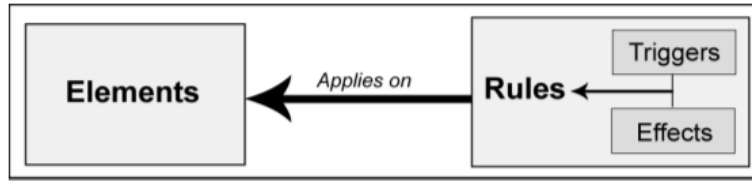
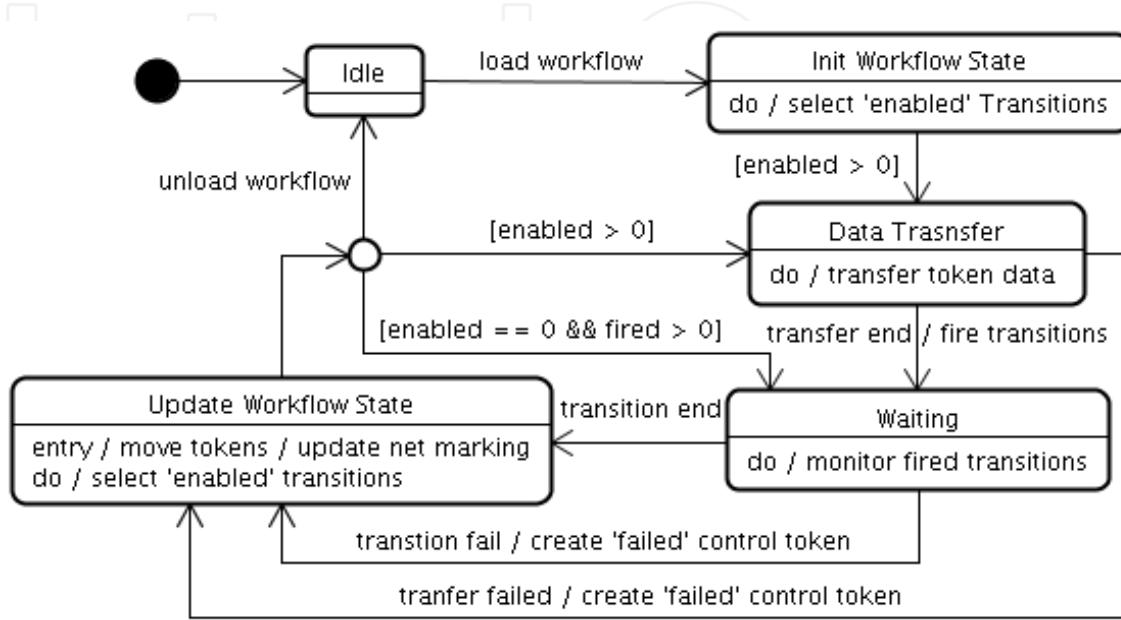**Figure 10.** Game rules as a triggers and effects.

Figure 11. Game workflow activity engine

The following flowchart summarizes the idea, where T is a set of transition instances in the given HLPN, $T_{disabled}$ is a subset of T with elements having status disabled, and $T_{unknowns}$ is a subset of T with elements having status unknown

Step 1. (Initialization)

- $T_{disabled} \leftarrow \phi$
- $T_{unknowns} \leftarrow T$ and $count \leftarrow 0$

Step 2. (Loop)

- While ($T_{unknowns} \neq \phi$) do
  - $t_{candidate} \leftarrow$ (random element from $T_{unknowns}$)
  - Status $\leftarrow$ (try finding an enabled binding for $t_{candidate}$, make it occur and return occurred. Otherwise return enabling status)
- if (status = disabled) Then (move $T_{candidate}$ from $T_{unknowns}$ to $T_{disabled}$)
- else if status=occurred Then ($count \leftarrow count + 1$)
- $T_{dependents} \leftarrow (\{T_{candidate}\}^{\bullet})^{\bullet}$
- Move ($T_{dependents} \cap T_{disabled}$) from $T_{disabled}$ to $T_{unknowns}$
- End if ;
- End while

5. Experimental results and discussion

In the implementation of the Crazy ball 2 case study game, five game rules in terms of five modules are used namely: "Enemy Create", "Enemy Active", "Enemy Move", "Enemy effect", and "Enemy Attack". The "Enemy Create" module is basically used to create the desired enemy with some collection of predefined attributes and values. The "Enemy Move" module is used to move the enemy object based on its Move ID parameter. For instance, in the case of the "Bad Ball", enemy model, "Enemy Move" module will make it gain positive or negative speed based on the position of the player object. The "Enemy effect" module is used to make the object emit special effects or change colors based on its Effect ID parameter. The "Enemy Attack" module will make the enemy engages or do damage based on its Attack ID parameter. Finally, the "Enemy Active" module enables the enemy to receive collision and physics calculations. Figure 12 shows some screen shoots for the crazy ball 2 game environment with some enemies challenging modules.
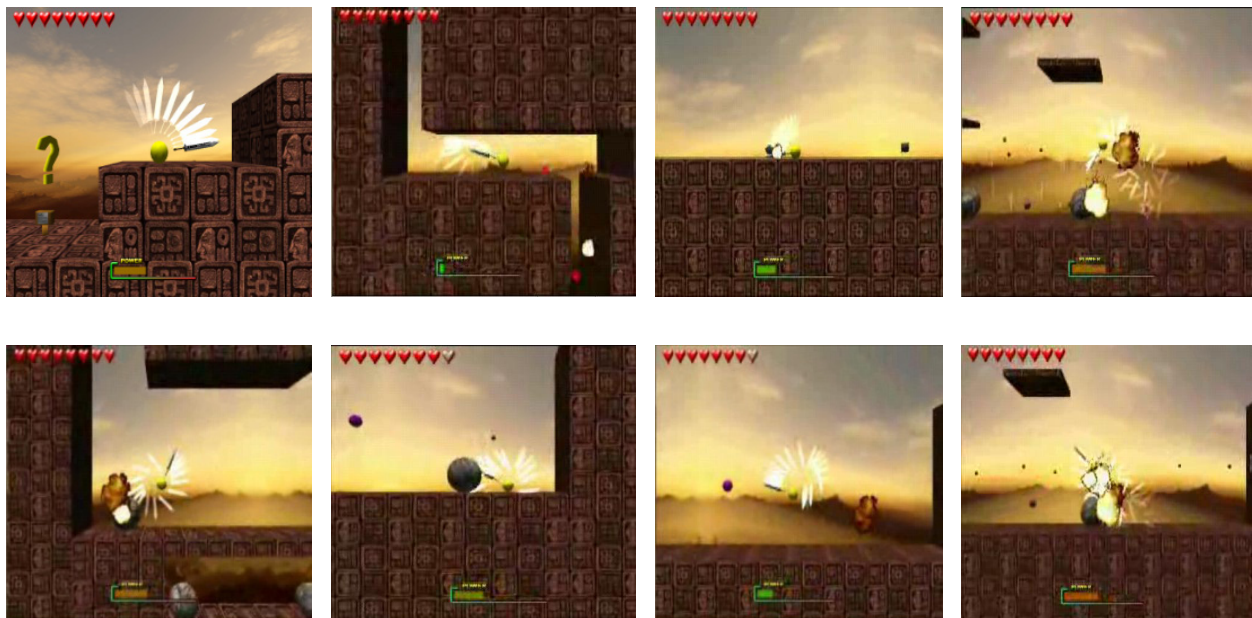
Figure 12. Screenshots for Crazy ball 2 game environment with some enemies challenging modules.

6. Conclusion and future work

This chapter attempts to overcome some problems which are encountered in the interactive drama systems as well as storytelling applications. This is achieved by proposing an integrated framework for deeply combining interactivity and narrative in computer games workflow. The idea is derived from the study of interactive drama, Petri nets (PN), narrative structures in computer games and game workflow activity process. The main contribution of this paper is to show how workflow management concepts can be jointly utilized with Petri nets (PN) for modeling game systems and game workflow control. The main advantages of using PN are that it copes well with branching stories and can evolve in

parallel in large virtual world. The proposed idea is supported by some case study called Crazy ball 2. Further research effort is still needed for establishing more relationship between the game workflow activity process and other entertainment game applications with different graphics aspects, interfaces and contents.

Author details

Hussein Karam Hussein Abd El-Sattar
Ain Shams University, Faculty of Science, Mathematics & Computer Science Dept., Abbassia, Cairo, Egypt
Al-Yamamah University, CCIS, Riyadh, KSA

7. References

Brom, C.; Abonyi, A. (2006). Petri nets for game plot, *In Proc. of AISB*, Vol. 3, pp. 3-13, AISB press, 2006

Brom, C.; Sisler, V.; and Holan, T. (2007). Story manager in Europe 2045 uses Petri nets, Proceedings of the 4th International Conf. on Virtual Storytelling, LNCS 4871, pp. 38-50, Springer-verlag, 2007

Delmas, G.; Champagnat, R.; and Augeraud, M., (2007). Plot monitoring for interactive narrative games, *Proc. of ACE07*, Austria, pp. 17-20, 2007

Jenson, K. (1997). *Coloured Petri nets-based concepts, analysis methods and practical use* (2nd Edition), Springer-Verlage, New York, 1997

Karam, H., (2010). A new plot/character-based interactive system for story-based virtual reality applications, *International Journal of image and graphics*, Vol. 10, No. 1, 2010

Magerko, B.; Laird, J. (2003). Building interactive drama architecture, ACM SIGCHI International Conf. on Advances in Computer Entertainment technology, ACE 2003

Mark Reidl, O., Michael R. (2005). From linear story generation to branching story graphs, *American Association for AI*, 2005

Meehan, J. (1981). TALE_SPIN, Shank, R. C. and Riesbeck C. K. (Eds.), *Inside Computer Understanding: Five programs plus Miniatures* (Erlbaum, Hillsdale NJ), pp.197

Peterson, J. L. (1997). Petri Nets. ACM Computing Sueveys, Vol 9, No. 3, 1977

Riedl , M.; Stern, A., (2006). Believable agent and intelligent story adaptation for interactive storytelling, In Proc. of TIDSE, LNCS 4326, Springer-verlage, pp. 1-12, 2006

Salen, K., Zimmerman, E. (2003). *The rules of play*, MIT Press, 2003

Sheldon, L., (2004). *Character development and storytelling, Thompson course Technology, 2004.*

Szilas, N. (2005). The future of interactive drama, Proc. of IE2005, the Second Australian Conference on Interactive Entertainment, ISBN: 0-9751533-2-3, 2005

Tado, M. (1990). Petri Nets: Properties, Analysis and Applications. Proc. IEEE, 77, 1990

Van der Aalst, W. (1998). The application of petri nets to workflow management, the journal
    of Circuits, Systems and Computers 8, (1), pp. 21-66, 1998.