BRIDGING THE GAP:
Transforming Knowledge into Action through Gaming and Simulation

Proceedings of the 35th Conference of the International Simulation and Gaming Association, Munich, 2004

Edited by Thomas Eberle, Willy Christian Kriz, Matthias Puschert & Fabian Glötzner
Towards a Choreography of Gaming

PIETER VAN DER HIJDEN

1. Choreography

1.1. Dance

Choreography is a formalised method of dance notation. It describes everything that is not spontaneous in a dance performance. Rudolf Laban developed such a standard notation method in 1928. Using his Labanotation, choreographers (i.e. the creators of a dance) can express their intentions and communicate them to related professionals. Compared to an informal narrative describing a dance, choreography is more rigorous and less confusing.

1.2. E-Commerce

In recent years the term choreography has been applied to electronic commerce. In this context it describes the ordering and sequencing of business activities within electronic collaboration.

In normal daily life, we have the impression that our behaviour is spontaneous and not directed by any script. In fact we have been programmed by rules and conventions and by many mini scripts. For example, when we buy food at a market, we follow an implicit script. But when we shop in a supermarket, our script will be slightly different.

As soon as computers became used for commercial activities, it became necessary to make commercial scripts explicit. How do you find a business partner? How do you negotiate the business rules to be applied? How are you actually applying these rules? Specialists developed a standard notation for such processes, thereby inventing electronic commerce choreography.
1.3. Gaming
Gamers do not use choreography. They describe the content, system and process of their games mainly in verbal form. It is also good practice to visualise the content by means of a diagram. However, it is left entirely to the game designers (i.e. the creators of a game) to decide how the narrative and the diagram are written or drawn. A choreography of gaming could improve the quality of our games and the efficiency of the game development process.

1.4. This paper
In this paper we “reverse engineer” various existing games in search for elements of a gaming choreography. We set our first steps towards such a choreography and present a game design workshop to apply them.

2. Current practice
We begin our presentation with some “reversed engineering” of two existing non-computer games. The first is Pachisi, an entertainment game. The second is Hex, a management game.

2.1. Example: Pachisi
Pachisi is a well-known board game that originated in India during the sixth century. The aim of each player is to move his four counters around the board and back to the centre before his opponents.

When we visualise player activity during a Pachisi session, we see a sequence, repeated over and over again in a fixed order. Were this visualisation to be a musical score, we would hear a univocal melody built up from four tones that constantly repeat. The players do not interact with each other; they interact with the game board. Pachisi has a simple structure. Its temporal behaviour is largely mechanical and there is no need for a facilitator.
Figure 3: Pachisi - temporal behaviour

Suppose we were designing a game like Pachisi. How could we write down in an exact and efficient manner the way in which the game should be run? A long narrative describing all the repetitions would not be appropriate here. But a flow chart such as computer programmers use could provide a compact notation. By introducing a loop, we would only once have to specify actions that will be performed many times during the run.

We could even go further, and consider all four players and their turns during one micro-cycle as a form of repetition. This form of repetition may also be represented by a loop in our diagram. In this way the Pachisi game can be described by a double loop.

Figure 4: Pachisi - flow chart (double loop)
2.2. Example: Hex

Hex, by Richard Duke, is a board game on decision making and resource allocation. The participants play the roles of 12 Chiefs of local settlements, 3 Regional Administrators, and one National Leader. The aim is to develop the settlements, the regions and the simulated country as a whole.

Figure 5: Hex Game Board

Visualising the temporal behaviour of a Hex session is quite different from the Pachisi example. Again we see micro-cycles built up from smaller steps. However, we now see various player activities taking place at the same time, with compulsory as well as optional actions. Nevertheless, the time base of a Hex session has a strong hierarchical structure. The game proceeds from year to year. Each year consists of a fixed sequence of steps. The transition to the next step is a moment of synchronisation.

The musical score of a Hex session would sound like a symphony, with bursts of activity following well-defined patterns and a well-defined rhythm. A Game Overall Director is needed to orchestrate this performance.

When we “reverse engineer” the Hex game, we can describe the way it works via a flow chart. This one is more complex than the Pachisi flow chart, in that it contains parallel activities.

2.3. Conclusion

Pachisi is an entertainment game that has existed for centuries, whereas Hex could be seen as an archetype for many modern management games. Despite their differences, they both use a hierarchical time structure that is identical for all player roles.

Richard Duke introduced the terms macro cycle and micro cycle to indicate the temporal character of games. Although he has recently renamed them ‘primary phases’ and ‘steps of play’, their essence remains the same. The
macro cycle indicates the major steps to be taken during a game run. One of these is the real kernel of the game. It consists of a sequence of micro-cycles, possibly further subdivided into a fixed sequence of smaller steps. In fact, the whole game session is subdivided into a hierarchy of smaller pieces.

Flowcharting might provide an adequate form of notation for the temporal behaviour of these games. With basic constructs like concatenation, selection and iteration, supplemented by parallelism and nesting, flowcharting could be a candidate for a choreography of gaming.

3. Upcoming practice

A hierarchical time structure has proven quite adequate in many successful games, old and new. However, in certain cases it might be too limited. The real world is not a well-organised clockwork mechanism, but a complex whole of loosely connected actors following their own scripts and interacting in positive and negative ways. Each actor might follow a hierarchical time schedule, triggered by certain events (e.g. major life events, annual holiday, monthly salary payment, weekly car washing, daily meals, hourly breaks etc.). However, the whole (all the actors together) is not synchronised. Of course, the actors are not completely free. There are rules, sequences and conditions. Nevertheless, an actor might be involved in various chains of activities at the same moment. At any given time, an actor may have the chance to choose between various alternative actions. When considering a choreography of gaming, we need to take into account games that have not been based on hierarchical time structures. Pacman is a good example of such a game.

3.1. Example: Pacman

Pacman was originally an Arcade game and only later became a computer game. It has been around for more than twenty years. Although computer games have gone through an enormous evolution since its creation, Pacman could still be considered an archetype. Its simple form illustrates the essence of games that do not follow a hierarchical time structure. In computer jargon, Pacman is object oriented rather than procedure oriented. The Pacman player walks through a maze while eating pills. He is free to choose his way. However, the maze is also populated by four ghosts with varying levels of intelligence, some more aggressive than others. They are chasing the player. Only when the player eats a power pill, are the roles reversed for a while.
Pacman is a one player game. The four ghost roles are played by the computer. Suppose, however, that these four roles were also played by humans. Could we still visualise player activity in the same way as we did above? The answer is no. Player behaviour is now less predictable. All the players are active at the same time and are free to choose their own way through the maze. No time structure has been imposed. A visualisation of the temporal behaviour of the game could only be a description of a sample session, not a prescription for all sessions. Considered as a musical score, a Pacman session would sound more like traffic noise than a piece of music.

When designing a game like Pacman, a flowcharting technique that describes the whole game as one big procedure has limited value. Fortunately, object oriented computer programs have their own range of concepts and types of diagrams. An object oriented computer program is based on objects. Objects are artefacts that have certain characteristics (data) and can perform certain actions (functions). Objects are instances of classes and classes could be ordered hierarchically, where lower classes can inherit the repertoire of functions and characteristics of higher ones. In a running object oriented program all objects behave like independent and parallel running subprograms.

In the Pacman game, the pacman and the four ghosts are objects. They each have certain characteristics, such as their appearance and their actual position in the maze. They all have a repertoire of possible functions, like eating pills or eating Pacman. All moving objects in Pacman might form a separate class. They share certain characteristics and functions, although each descendant (Pacman versus the ghosts) has other characteristics and functions of its own.

Figure 6 - Pacman Screen
3.2. Conclusion

Our treatment of the Pacman game illustrates the existence of a category of games without universal hierarchical time structure. In these games the players and their activities are only loosely connected. They even could avoid each other. In the Pacman game they have to! We will call this category object oriented games. Some of them are computer-based, some are not. Object oriented games like Pacman are not isolated counter-examples to the procedure oriented games with their hierarchical time structure. They deserve their own position in the gaming spectrum, both on principle and for practical reasons. In some cases, the way in which object oriented games reflect reality might be preferred on principle. The practical reason is that the absence of a pyramid of synchronisation points facilitates the construction of asynchronous games, i.e. games where not all players have to be active at the same time. This category of games may well be growing in the near future, especially in the field of collaborative electronic learning. Flowcharting may provide an adequate method of notation for hierarchically time structured games. For object oriented games it does not. With flowcharting alone, a gaming choreography will be incomplete.

4. First steps towards a gaming choreography

Our first steps towards a choreography of gaming focuses on the design phase of a new game. This implies that the definition phase and the analysis phase of the game development process have been finished and their results are available in whatever form (narrative, diagrams).

The design phase is not usually a purely analytical and intellectual activity that can be executed by a single person. Most of the time it will be a collective activity, i.e. a number of meetings or a workshop. We have developed the framework for such a workshop that will result in a set of documents and diagrams. These will form our first steps towards a gaming choreography.

The agenda of our workshop contains the following activities: inventory of actors, analysis of actors, and analysis of processes.

4.1. Inventory of actors

The team of game designers knows the purpose of the game to be designed and has the results of the analysis phase at its disposal. In our game design workshop they will individually prepare a list of all possible actors. Questions to help identify such actors are:

Which organisations, groups, individuals (internally or externally) are important for reaching the internal goal of the game? Which could help, which could hinder?

For which organisations, groups, or individuals (internally or externally) will the internal goal of the game have a positive or negative impact?
The game designers will then collectively consolidate their lists and restrict the consolidated list to the most relevant actors (preferably not more than 7-9). It strengthens the robustness of the design and makes the rest of the development more efficient if actors could be grouped into classes and classes-of-classes.

At this point it might be relevant to already determine how these actors will be implemented in the game. There are four options: played by members of the game’s target audience (i.e. the students), played by someone else (a facilitator, a tutor, an expert), simulated by computer software (i.e. software agents or mathematical models), or carried out directly on the game board (either physically or virtually).

4.2. Analysis of actors

Our game design workshop proceeds with the analysis of actors. The game designers will distribute the actors that appear on the list amongst themselves. One game designer or a small team will be responsible for each actor. The designers analyse “their” actor by finding answers to the following questions:

What are the basic characteristics of this actor? What is different from the other actors?

In which terms can we specify the status or position of this actor at any moment during the game? What does this actor know about the status or position of other actors at any moment during the game?

What is the status or position of this actor at the start of the game?

From the perspective of this actor, what might be the most positive outcome of the game?

From the perspective of this actor, what might be the most negative outcome of the game?

The game designers write the conclusions of their analysis on coloured memo stickers and stick them on a large matrix form, the actor matrix. This form lists all the actors involved, both horizontally and vertically. The results are then collectively discussed and changed where appropriate.

The relations between the actors will then be analysed. This activity is again executed by single participants or small teams. The questions to be answered are now:

Which repertoire of activities is available for this actor with respect to each of the other actors? Which are required to help the game proceed? Which are optional?

Which influences are felt from each of the other actors?
Again the results are written on memo stickers, placed on the matrix form, cross-checked and discussed collectively.

The ultimate outcome of this activity will be a completed, checked and agreed upon actor matrix. This is the first step towards our gaming choreography.

<table>
<thead>
<tr>
<th>Actor</th>
<th>Characteristics</th>
<th>Start Position</th>
<th>Possible Actions</th>
<th>Final Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td></td>
<td>Manager, Politician, Activist</td>
<td>Press, Facilitator</td>
<td>Neg., Pos.</td>
</tr>
<tr>
<td>Politician</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Activist</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Press</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Facilitator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7: Actor Matrix

4.3. Analysis of processes

The participants will now collectively discuss how the relations between the actors as specified in the actor matrix will be transformed into processes (chains of activities) during the game. This is a highly heuristic process. Some activities have to be combined to form more complex collaborations. Other activities might be seen as too complex and have to be recomposed. This discussion might have some retrospective impact on the actor matrix.

The main result of the current analysis, however, is a so-called activity diagram (or a set of such diagrams). An activity diagram shows the dynamics of interactions between actors. It is an existing diagram type belonging to the Unified Modelling Language, a widely accepted range of concepts and diagrams used by computer programmers.

An activity diagram is a form displaying vertical swim lanes. Each actor has a swim lane of its own. In our workshop we use a large copy of such a form. The processes can be visualised by memo stickers and arrows.

This approach is useful for hierarchically time structured games as well as for object oriented games. In the first case you see one overlapping procedure. In the second you see isolated processes triggered by some internal or external event. This is the second step towards our gaming choreography.

Figure 8: Activity Diagram
Afterthoughts

The language we use influences our thinking. An informal and narrative method of describing games will therefore be reflected in the games we develop. This could be a hindrance. New types of games might be possible if we could use a more expressive language to describe them. Further, information and communications technology is being increasingly used to establish well-defined communication patterns between gaming participants. This requires better defined descriptions of the activities and interactions in the game. Finally, electronic learning is developing towards game-like group activities, but so far a formal method of game description has been lacking. It is worthwhile, therefore, to investigate the potential of gaming choreography. In this paper we have introduced the theme and contributed the first steps.

References

Duke, R. Hex Fact Sheet; Richard Duke; http://www.sofos.nl/products/hexagon%20factsheet.htm
Grunfeld, F.V. (1982). Games of the world; how to make them, how to play them, how they came to be. Switzerland, Swiss Committee for Unicef.

Author

VAN DER HIJDEN, Pieter, MSc
independent management consultant specialising in e-government, e-learning, management games and XML
Sofos Consultancy, P. O. Box 94874, 1090 GW Amsterdam, The Netherlands
Email: pvdh@sofos.nl; www.sofos.nl