TRAINING PROJECT REPORT NO 4

COLAB: Designing a R & D Management Game

R E Thomas
J D Walsh

(Reproduced with permission from European Training 1, 2, Summer 1972)

Central Training Section
Science Research Council
State House, High Holborn
London WC1R 4TA

October 1972

This article describes the design and development of a management game called "COLAB". It was developed for use as part of a general management training course. Originally it was designed and run with manual control but lately it has been computerized.

The intention of the article in recounting the stages in the design and development of COLAB, is to help and encourage would-be game designers by describing some of the decisions to be made and problems which arise in the process. One factor which became apparent in the later stages was that separation into design and computerization stages was not the best way to do it.

The game is now in use and the article concludes with a brief evaluation of its use todate.

Games v Simulations

The bibliography* on the design and use of management games provides little help to the would-be designer of such an exercise. He is led to believe either that they are so easily obtainable, or so mathematically complex, that it would not be worth writing one for his own use. It is true that the availability of these exercises has increased in recent years, that a greater proportion of the material now available is of Cisatlantic origin, and that where no suitable example exists, consultants can purpose-design one; but these should not be reasons enough to dissuade any reasonably numerate training officer from writing his own.

COLAB was designed because nothing similar was known to exist, and part of the purpose of this article is to demonstrate the ease with which a game can in fact be constructed. The mathematical analyses of the theory of management games also throw up an apparent reason, where none exists, for the "amateur" not to proceed; the theoretical treatment has, for instance, resulted in the word "simulation" being used to describe a management game.

In fact, games differ from real life in that they define its parameters in terms so simplified as to be quite different in nature. To provide a framework for controlling the game, its writer has to make numerous assumptions, omissions and approximations; perhaps the most flagrant of these is that he has to assume that human behaviour is predictable and controllable by a small number of variables. How easy life would be if, for instance, the people one worked with were as rational as COLAB's employees! COLAB also reveals many more glimpses of Utopia: the Inland Revenue doesn't bother them, the employees are never ill, the equipment is no sooner off the delivery van than it's working, etc., etc.

But the biggest difference is that a simulation is the result of trying to build, for whatever reasons, a model that replicates some phenomenon, whereas a game is a teaching device; games are - or should be - fun to play, and are the management training equivalent of the discovery method

* See, for example, BIM Library Bibliography: Business Games, September 1968.

of the schoolroom. COLAB is a game of at least average complexity, yet its design called as much for the exercise of the imagination as for the use of mathematical skills. The only simulation involved in the whole process was the modelling of the control of the game in the form of a program.

Saying that games are fun to play probably sums up all the reasons that contribute to their merit as a teaching device. They provide or promote involvement, excitement, group loyalties, social interaction and perhaps even self-analysis. They provide a practical exercise in decision-making (individually or in groups), to a time schedule and to a background of desirable results against which progress can be measured. They allow practice in the interpretation of the feedback from past decisions, and, for those whose work experience has been limited to one or a few of the functions of management, they can give an introduction to the interaction of the many areas in which successful activity is required to keep an organization afloat. Computerized games also impart a familiarization with the use of terminals that is a very useful bonus to any training.

The Choice of Variables

Having said that to call a game a simulation is the wrong basis from which to start designing, there is nevertheless a sound psychological reason that the variables used in the game should be labelled in terms that are understandable to, and appear relevant to the jobs of, the players. COLAB is meant for use in a part of the public sector where successful financial performance cannot be measured by the usual values or ratios of profit, turnover, sales, etc. To have used one of the readily available manufacturing-cum-marketing games might well have frustrated players' involvement because of its evident irrelevance to their work. COLAB, therefore, is set in a research laboratory that already has a handsome income in the form of investment interest.

Having decided the setting for the game, the designer next needs to ask himself whether or not it should be interactive. In an interactive game, the results of the decisions taken by any one team of players are influenced by the simultaneous decisions of their competitors. Such a game is more difficult to design and very much harder to control in

operation. Its merit is that true competition is seen to exist and for that reason COLAB was made interactive.

The next step is to decide what the variables and the parameters of the game are. The variables are those quantities that can be changed either by the game's controller in setting and using his parameters, or by the players in making their decisions. In COLAB, the first group of these includes the interest rate paid on the investment fund, the growth rate of that sector of private industry from which most contract income originates, and the level of annual governmental grant; the second group includes salary levels, contract tender prices, numbers of recruitments and redundancies, and capital equipment purchases. The variables used in COLAB are shown in Table 1. The number of player-controlled variables is a guide to the length of time that each cycle of the game will take in playing.

A. Altered by players	B. Controlled
Salary levels of the three types of staff	(i) Pre-set
Recruitments	General levels of economic activitity
Redundancies	Contracts available
Staff deployment	(ii) Controlled by parameters
Welfare costs	Contracts awarded
Equipment purchases	Resignations
Tender prices	Government and industry grants
	Other incomes
	Running costs
	Equipment depreciation

TABLE 1: COLAB variables.

The Basis of Control

The parameters define the way in which the variables interact; it is here that the designer can make the game as complex as he wishes, though it is likely that too great a complexity will cancel itself out in terms of effect, and that a simpler relationship would have sufficed. Certainly, complex parameters result in difficulty in controlling the game manually because of the additional calculation required. An example of one of the most complex parameters used in COLAB is the relationship between contract tendering price (a decision by the players) and the actual award of contracts (a function of the controller). A contract is awarded to the team with the lowest tender figure after bid prices have

been "corrected". The extent of the "correction" is a function of each team's capital equipment inventory value, its speed in completing previous contracts and its reputation; its reputation depends in turn on the current average salary level of its research workers and the percentage of them employed on non-commercial work. There is a self-cancelling element here; the more researchers there are on non-commercial work the higher is the reputation of the laboratory, but it is then slower at finishing contract work.

The choice of variables is a fairly easy task, though here again it will be important to remember that it is a game and not a simulation that is being written, and the original list of possible factors may well need pruning to provide a manageable task. The same is even more true of parameters; they should be fairly simple: resitance to the temptation of making them too "life-like" at this stage will be repaid later when controlling the game manually or when writing the control program.

Another decision required when formulating parameters is whether or not to introduce an element of randomness into the control. An interactive game already produces feedback to the players that appears to contain some randomness because they are ignorant of the actions of competitors. Given this feature there seems to be no reason at all, for instance, to base one of the parameters on a random number table: the trainees might just as well be playing "Monopoly". Indeed, except in the simplest non-interactive games, it is likely that the activities of four or five teams or players will produce enough "noise" between their output (decisions) and the feedback from the control, that they will actually need some extra information to aid their interpretation of the results. In COLAB, which is played in "quarters", each team has to publish "annually" a report on the previous year's activities and the current values of some of its variables. This is made available to the other teams.

In deciding his parameters, ie how one variable is to change with another or others, the designer has also provided the controls for managing the game when it is played. These may be in the form of graphs, tables or nomograms and it is very likely that when testing the game he will find some short cuts that make manual control easier or quicker.

Documentation

Before a test-run, the designer must also write the "story" and specimen documents. The documents are of two sorts: those that teams use internally as records of their decisions or to help their calculations, and those that form the interface between players and controller, passing one way with, to take a COLAB example, tender prices, and returning with the results of the bids made. The interface forms should be kept as few as possible for obvious reasons. The manual version of COLAB uses eight forms, as shown in Table II. Copies of all these forms can be included with the "story" in a manual that is issued to players before the start of the game.

Form	Type	Contents
C	Interface	Contract bids and results; other information from control
CC	Internal	Contracts current and state of completion
CF	Internal	Contracts finished
F	Internal	Financial summary and equipment inventory
P	Internal	Personnel cost and deployment
R	Internal	Staff movements in and out
SA	External	Annual report
SQ	Interface	Quarterly summary of data

TABLE II: COLAB forms

The "story" sets the scene for playing the game. In COLAB, it deals with the history of the laboratory, its sources of income, and the personnel arrangements. It lists the variables and their inter-dependencies, without actually quantifying the parameters. It gives a number of rules that in effect set limits on the values of the decisions that teams can make. Some of these rules may be for ease of control, eg that salaries can be varied only in multiples of £100; others are arbitrary limitations that may involve extra control work in checking them or may require an "auditor" to visit teams as the game proceeds. The manual also gives a set of data applicable to all teams at the start of the game, it introduces the purposes of the forms to be used, and finally it sets out the procedural arrangements for playing the game.

The material produced by this stage comprises the controls (graphs, tables and nomograms), the "story" (including rules and "year-zero data") and specimen forms.

Testing

The game must now be tested and it is at this point that the effort required begins to be significant. All the work described so far can be done by one person in odd moments and in the evenings; testing entails enlisting the help of at least as many colleagues as the game is to have teams, and using them for probably a whole day. They will have to be willing, also, to pre-read the draft "story" and familiarize themselves with the forms. They will get none of the benefits, moreover, that ultimate players of the game can expect, for the purpose of testing is to verify whether the size of the tasks imposed on players and controller are reasonable, and to see if the control parameters, pre-set levels and "year-zero data" give satisfactory results numerically. These results cannot be assessed until the test run has lasted for about as many cycles as it is intended to run the game in practice.

The other piece of useful information that a test run provides is an indication of the factors that it will be possible to measure that will reflect the overall performance of the teams. When COLAB is played, each team is required at the end of the game to report on its own performance in terms of how it organized itself and what its policies were. These beliefs can then be compared with a number of performance indicators that the controller calculates based on performance data that he has received. These indicators allow teams to be ranked on scales (see Table III) and provide contrasts both with teams' stated policies and with their account of their own organizations.

Indicator	Scale
1	Big - small
2	Good employers - bad employers
3	Industry-orientated - research-orientated
4	Well equipped - poorly equipped
5	Cautious - careless
6	Rich - poor
7	Well set-up - shaky
8	Good salesmen - poor salesmen
9	Acute - ill-informed
10	Learning from mistakes - acumen decreasing
11	Generous - mean
12	Hoarders - spendthrifts
13	Hard-working - slackers
14	Effective - ineffective
15	Financially successful - financially unsuccessful

TABLE III: COLAB indicators

Computerizing COLAB

All that has been said so far relates to games whether they are destined for manual or computer control, though the form of the parameters used can be influenced by which sort is intended. It is likely that the average training officer will get as far as testing and perhaps even manual operation before he feels able to call on programming resources, in order to relieve him of the onerous mental arithmetic involved in manual control. This is not, however, the ideal method, and unless the designer of the game is himself a systems analyst it is preferable to form this liaison earlier. In either case, the programming will, like the manual test-run, probably throw up the need for some further design modifications.

In spite of the case made so far for simplicity in design, it was found that when COLAB was played manually, the time taken to perform the control function was sufficiently great, and the tasks involved sufficiently mentally fatiguing, to warrant the introduction of computer aid. Connected to the Science Research Council's Atlas computer, among other things, are a number of typewriter-like terminals, driven by an operating system, which allow users to run jobs and look at the output. Of course, this is a general system, used by many people in and outside the Atlas Laboratory to do all kinds of different tasks, but it was possible to include COLAB in this environment. It was decided that the computerized COLAB would carry out as many of the control functions as possible, bearing in mind that some form of over-ride would always be needed. The system was made to look as similar to the manual control system as possible, so that, if the computer by some unlikely chance were to break down in the middle of a game, it would be possible to continue under the old manual scheme without interruption. Some of the tasks of the players were also made automatic, including the preparation and typing of forms C and SQ, and the keeping of records of contracts completed. However, this last item is hidden from the players, and, in general, it is wise for them to continue to fill in the forms, not least because it enables them to clarify their ideas.

Each "quarter" can be split into four separate actions. Firstly, control types in details of the contracts on offer and then informs the players that they may proceed. It is essential to have some mechanism by which

control and players may keep in step, and the Atlas system provided such a method. One of the commands allows the terminal to be used like a TELEX, and enables one terminal to send messages to another. In this way, the progress of the various sections can be monitored. When they receive the message, the players type the form C held by the computer, and make bids for the contracts. These bids come in the form of total salary cost for the job, assuming that all workers of each type earn an average salary and that this will not change over the period of the contract, and a percentage overhead. Thus, players will also be asked for their salary scales and welfare costs now, data usually found on form P. Finally, the players send messages to control to say that bids have been made. When control has received all these messages, the awarding of contracts can be done automatically. As was stated earlier, various "correction" factors are involved in producing a modified bid price for each team, depending on its record to date. However, it is possible for control to over-ride the machine and award any contract to whomsoever he wishes (a simulation, perhaps, of the "old boy" network). When the players discover who has been awarded what, they are asked for all the rest of the variable data, such as staff allocations to contracts for this quarter, recruitment/redundancy plans, and any new items of equipment purchased. The computer then produces form SQ for them, together with a balance sheet for this "quarter", showing the current state of their investment fund. This should agree with the calculations they have done on their forms (in practice this sometimes happens). Control can now look at the players' forms SQ, to see how they are progressing, and proceed to the next "quarter". As before, form SA is prepared by hand "annually", so that the ideas behind the policies of each team can be assessed.

The programs were written in FORTRAN, hopefully so that they can, with minimum effort, be transferred to another machine. Care was taken to make the use of the terminal as easy as possible and to prevent input errors from causing havoc. Thus free-format input, all integer, was used throughout. The multi-access system allows sets of commands to be strung together to make one "pseudo" command, and much use has been made of this facility to minimize the amount of typing required from a player.

A number of problems did arise in the implementation, however. Ideally, COLAB should work in an interactive environment. All data should be requested by cue words, checked for accuracy, and any errors immediately reported so that corrective action can be taken. The Atlas multi-access system does not allow this, all input having to be typed in before the program can start. The cueing of input data is done, however, by inserting messages in the "pseudo" commands. Any data errors cause the program to abort with a suitable message, and stops the data from the current run being saved in the data file (all information is kept in a data file for use from one run to the next). A user can then correct the data and re-run the job. If, however, a player inputs correct-looking data about which he subsequently has second thoughts, control must take a hand to reset the data file. Such action, however, can be penalised by the manual over-ride adversely affecting the award of contracts to the erring player.

Another problem, common to many multi-access systems, is the provision of a time-out. In order to share facilities evenly, a multi-access system will detect when someone has been inactive at his console for some time (depending on which system is being discussed), and will disconnect the line. Reconnection may entail anything from the pressing of one key to the typing of several lines of commands. In a business game environment, unless players can be persuaded to log on and off many times during the game, this is likely to occur very frequently. In this case it was decided to remove the time-out feature for the duration of the game.

The Structure of Computer Control

As was said earlier, it is better to call in a systems analyst at the design stage of a game that is destined for computer control. Trying to reproduce a previously manual system entails the inclusion of some awkward features. Principal among these are look-up tables for various values which can change from "quarter" to "quarter". These can be quite short, as that for the change in interest rate, or relatively very large, such as the tables for providing the number of resignations each "quarter", which depend in a non-uniform manner on the salary of the staff, the number of staff and how many are doing research. Another difficult feature is the digitizing of a graph which has been sketched roughly by hand to yield the "right sort of result".

Obviously, graphs can be approximated and tables can be packed to conserve space, but it should be possible to provide acceptable results from functions, which would have the added advantage of being open-ended (to allow COLAB to run for an extra "year" would entail adding more entries to the ends of a number of tables). In spite of the undesirability of using random numbers in a general way to control the game, the judicious use of bounded numbers in certain contexts could be of service here. However, the game has been written making great use of subroutines, so it is an easy task to alter the control methods for each variable in turn without upsetting the general system.

The Benefits of Computer Control

Speed and the familiarization with computers have already been mentioned as advantages of computerizing a management game. Another is the checking out of the logic behind the game. When putting such a game into a form suitable for programming, particularly when the subjects of data checking and error recovery are foremost considerations, many "grey" areas of the game are brought into the open and cleared up. For example, what should be done when a player overspends his capital? At what point in the "quarter" (beginning or end) do such items as new salary scales take effect? All these must be catered for by the program somehow whereas using manual control it is possible ot ignore them at the design stage and do nothing until a problem arises in practice, when the credibility of the whole exercise could be badly affected.

Finally, once the game has been computerized, it is possible to try adding some more variables without the worry of having to perform many more computations by hand. A well-designed game should be capable of extension in this way. However, even if no developments of this sort are attempted, an undeniable long-term training advantage of having computerized the control of the game will be that all managers who play it are being encouraged to use computers, understand them, and even to trust them.

Further Development

The flexibility resulting from the use of subroutines allows the designer to adjust the control mechanisms in the light of experience. One such adjustment currently (April 1972) being made is the inclusion of penalty-

clause payments for contracts that are not completed within a given time. There is a temptation, that should be resisted, for later adjustments to software to be in fact additions, and once again the motivation will be an attempt to approximate more closely to life-like conditions. The results, however, could be an increased load of calculation and anlysis on the players, with a consequent reduction in the time available to them to examine their working methods and interactions. Playing the game would probably also be slowed down.

The chimera of perfection will, however, tempt the designer who remains involved in the use of a game to adjust its mechanics as time goes on. So that although COLAB is now available for use, it is also likely to undergo continuous development. Its availability is further modified by having been computerized: the manual version could have been run anywhere by anyone suitably instructed, but the portability of the computerized version is limited by hardware availability and software compatibility.

Evaluation of COLAB

People who have played COLAB so far as part of a general-management training course have been asked to evaluate it. On five-point symmetrical scales, with 5 as best marking, the overall results to date are: for job relevance, 3.6; for intrinsic interest, 4.6; for quality of presentation, 3.8. The relevance markings are disappointingly low bearing in mind that the game was purpose-designed; this must spring from the fact that looked at from a "real-life" viewpoint, any game looks threadbare, and adds to the earlier point about the difference between a game and a simulation. As between evaluations of manually controlled and computerized runs, the only possibly significant difference has been in the markings for interest, where the computer (4.8) scored higher than the manual control (4.5).

The qualitative results from the game are more interesting. No attempt has been made to measure, for instance, any changes in attitude that playing the game may have brought about, but the final reporting session produces two sorts of observable reaction on the part of the players.

The first of these stems from a residue of corporate spirit within teams that causes them to want to have "won". Although, as shown in Table III,

no overall "winner" emerges, it is usually fairly evident that some teams have done better than others. The rank positions on each of the scales in Table III are obtained by calculating simple performance ratios, and it is important to state that all rankings are relative to the performance of the other players on one occasion, so that coming first in indicator 1, for instance, does not given an absolute measure of size. Furthermore, while some indicators give a scale that can be thought of as good/bad, others are symmetrical and best performers on them are ranked in the middle. At the time of writing, consideration is being given to computerizing the calculations necessary for the presentation of these final results.

Teams are prepared for the final session by being asked to summarize their activities. To avoid comments on the mechanics and logistics of playing the game it is important to brief them beforehand, and in COLAB this is done by issuing a note listing the sort of questions to which answers are wanted: what were the major policy decisions, how were they taken, were most members satisfied with the way they were taken, were there any changes in policy or decision-making method, etc? The second reaction from players occurs when their own accounts of their activities are compared with their actual performances, since this is rarely an equation. The confrontation between belief and fact then occurring is a powerful means of changing attitude. This situation could provide a useful starting point for some other part of a training course.