Chapter 8: Design for Quality

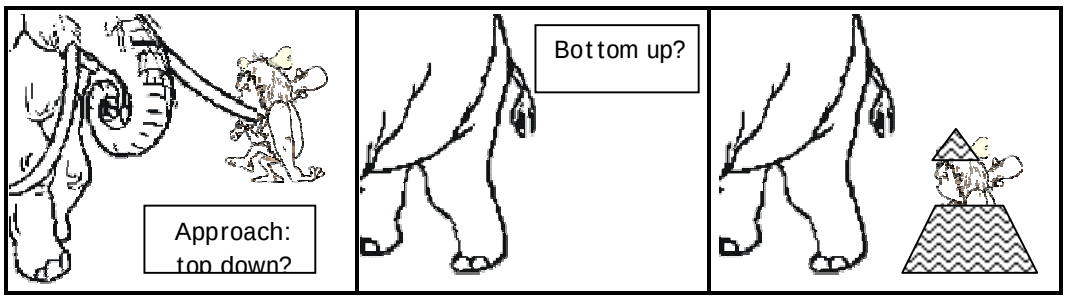
Contents

Introduction

"Everything in a comic has to work – words, pictures and timing – or else it fails" (Sabin, 1996). Likewise, for business simulations, the model, the interactions (decisions and results) and how the experience evolves (timing) have to work in unison – or the simulation does not deliver engaging and effective learning. This chapter explores the aspects of simulation design that impact quality. It covers the design *methodology* that I use (The Rock Pool Method) and design *soundness* in terms of *structural soundness* (simulation composition), *logic soundness* (ensuring the model is sound), *language soundness, software soundness* (sources of software errors), *testing soundness* (final testing and piloting) and, finally, *learning soundness* (validating that the finished design delivers learning effectively, efficiently and consistently).

The Rock Pool Method

This section explores software design methodologies in the context of business simulation design and describes a methodology - the rock pool method that provides structure while maintaining creative freedom. It is based on a paper that won the 2005 ABSEL best paper award.



Cartoon 8.01: Design Methodology - the Approach: Top Down or Bottom Up

Cartoon 8.1 illustrates two approaches to simulation design – top down or bottom up – where bottom up starts by modelling a business in all its detail and top down starts by defining learning objectives and constraints before deciding what is needed and then only building in what is required to deliver learning. The Rock Pool Method is a top down approach – without the bottom up problem experienced by our hunter-gatherer!

Developing computer simulations for management development and business education present particular software design problems. On one hand, for computer software development there is a need for a rigorous, structured approach. But, equally, creating simulation models that deliver learning in an effective, efficient and consistent way is a creative process – the Rock Pool method combines both.

The "rock pool" metaphor was chosen because of the way I, as a child, explored the rock pools on a beach after the tide had receded - moving from rock pool to rock pool and searching the rocks in each for *critters*.

Each rock pool represents a stage in the design process. Within each rock pool there are several design elements (the rocks) but these are not processed in a predefined order and are revisited several times. Although moving between rock pools is systematic, the order the rocks are explored within a rock pool depends on the simulation and the movement between rocks is based on creative needs.

Cartoon 8.02: Rock Pool

Software Design Methodologies

General software design methodologies divide into traditional, rigorous methodologies and the more recent "lightweight" or agile methodologies.

Traditional (Highsmith 2002) or "heavyweight" software design methodologies are rigorously structured. They are exemplified by the *waterfall method* (Brooks, 1995). These methods involve working sequentially through a series of stages at the end of each the stage is approved before moving on to the next stage. The *Stairway Methodology* (Allwood et al, 2001) is a rigorous software method (RSM) for business simulation design. The Stairway Methodology has five main stages (definition, formulation, evaluation, modification and completion) and within each of these there are several steps that are cycled through. For simulations, where design is a creative process, I believe that it is not possible to define everything beforehand as, in my experience, ideas develop as the design evolves.

More recently agile (lightweight) methodologies (Poppendieck, 2003) have appeared such as Extreme Programming (XP) (Beck 2000) and Feature Driven Development (FDD) (DeMarco, 1999) where the design process is not rigorously structured. Rather, it is a feedback process (Anderson, 2003) where the working software is delivered in stages and where the software *evolves* through frequent refactoring (rework and redesign based on software usage experience to simplify the software). It is argued that this approach is better at dealing with the complex adaptive system of software development (Anderson, 2003). In the context of gaming and simulation design the process is suggested as a spiralling cycle of *goal setting, action through technique* and *interpretation of results* (Bizzocchi, 2003). For simulations, Jones (1998) states *"authorship is not sequential. Teachers love aims, but authors love ideas much more. To stipulate that an author should start then follow it by devising parameters is, to my mind, unrealistic and inefficient".* But, as this statement is made in the context of students designing and running their own simulations, it is likely that the feedback process inherent in lightweight methodologies is necessary for these naïve designers to learn about the simulation design process. However, the creative nature of simulation design means that the design evolves and at first glance a lightweight method would seem to be appropriate.

What is the best methodology? Anderson (2003) suggests that the choice of software methodology depends on the maturity of the application and whether the project must be delivered holistically or can be delivered incrementally. Anderson shows these in a two by two grid (Figure 8.01)

	Immature	Mature
Holistic Delivery	Agile Method	RSM Method
Partitioned Delivery	Lightweight Agile Method	Either Agile or RSM

Figure 8.01: Maturity/Delivery Matrix

In the Maturity/Delivery Matrix, the upper right-hand corner represents conventional (mature) data processing applications that must be delivered as a whole. Anderson suggests that rigorous software methodologies (traditional, heavyweight methods) are best for these. By their nature, business simulations are immature products where each new simulation may require considerable innovation. (Where innovation is necessary because of continuing technological change, changes in the use of technology for education, changing business processes and structures and widening application of simulation). In terms of the delivery dimension, usually it is not possible to deliver a new simulation in stages (partitioned delivery). Thus computer business simulations fall into the top left-hand corner of the matrix, where Anderson suggests that agile methods are most appropriate as these utilize feedback to handle the immaturity of problem specification.

The Rock Pool Method is a middle road approach where, at a macro level the design process is a rigorous software methodology but at a micro level it is agile. Thus it combines both heavy and lightweight methodology using a rigorous structure where the simulation is developed through several major stages (rock pools) but within each rock pool development is unstructured and agile. Experience developing several simulations suggests that the agile, lightweight approach within a rock pool is appropriate to the immature (creative) nature of simulation design. (The rigorous structure imposed by the rock pools ensures that a sound simulation is developed *and* delivered on time.)

Thus the *Rock Pool Design Method* metaphorically proposes a design process that involves moving progressively and sequentially between rock pools with each rock pool representing a major design stage. Within each rock pool, there are several design elements, each represented metaphorically by a rock. Reflecting the intrinsically creative nature of computer simulation design, at an individual rock pool stage design is not a sequential process and does not have a defined starting point. Rather, depending on the simulation, its purpose and the designer, a rock pool development stage can start with any rock. As the development progresses within a rock pool, the designer moves between rocks revisiting each several times. When a rock pool's design tasks are complete, the designer moves on to the next rock pool.

Defined stages - the rock pools

The rock pool method involves progressively moving from one to the next of the following rock pools:

- **Needs Definition**
- **Simulation Specification**
- **Simulation Design**
- **Simulator Development**
- **Simulation Validation**
- **Finalise Design**

And, these define the structured and sequential elements of the method.

1. The first rock pool - needs definition
The "need definition" rock pool consists of four elements:

a. **Specifying the Target Audience(s)**
b. **Defining Learning Objectives**
c. **Settling Duration**
d. **Defining Manner of Use**

a. Specifying the Target Audience(s)
This involves defining who will participate in the simulation, who will run it and the type of organization that will use it and authorize its use. Randolph and Posner (1979) emphasize the identification of student skills and motivation, instructor skills and values and the institutional and professional pressures/concerns as factors affecting the effective design on learning situations and this suggests that the specification of these factors is a necessary part of defining needs and, certainly, this is my experience.

For learners the following are important:

Range of prior knowledge and experience
Diversity of prior knowledge and experience
Maturity and expectations

The range of prior knowledge and experience provides a basis to assess the need for learner and tutoring support. **The diversity of knowledge and experience** shows the extent to which learners can share knowledge and act as a learning resource. **Maturity and expectations** indicates the extent to which learners can handle the pressures of the activity, ambiguity and uncertainty. Thus range and diversity of prior knowledge and experience define cognitive (content) support needs and maturity and expectations define affective (engagement) support needs,

For the Strategic Exploration of Entrepreneurial Directions (SEED) simulation, the target audiences were university students, business people who are considering starting their own business and as a role reversal exercise for bankers and tax officers. As the prime student group would be studying science, technology and medicine rather than business the simulation would need to build in comprehensive knowledge support and help them handle the ambiguity and uncertainty associated with entrepreneurship. Further, their lack of world experience may make them uncomfortable with this form of learning (Knowles, 1998).

For the trainers/academics (tutors) running the simulation there are several issues:

Knowledge of the subject being taught
Familiarity with the use of simulations
Familiarity with technology

The **level of subject knowledge** defines the extent to which knowledge support must be encapsulated in the simulation and the extent to which the trainer must be automated out of the process. DISTRAIN and Modern Banking provide a contrast. DISTRAIN was to be used by experienced business people with deep business knowledge and so required minimal tutor support. In contrast, business experience and knowledge of the people using Modern Banking was not definable and so the tutor support had to be significant. **Familiarity with using simulation and technology** defines the extent to which the tutor will be comfortable with the activity, the need for training and parallel running (where the tutor runs the simulation in partnered with an experienced trainer). The simplicity of Product Launch and the experience of the trainer who would use Prospector meant that this was not an issue. In contrast, although DISTRAIN was to be used by experienced business people, they did not necessarily know how to use a simulation and this meant that a train-the-trainer workshop was required.

b. Defining Learning Objectives
This is more than just defining learning (knowledge acquisition or even skill development) needs. I use (Hall, 1996) a five dimensional model:

Knowledge Exploration
Skill Development & Practice
Motivation/Behavioural Needs
Assessment
Enhancing Learning

For the SEED simulation, the knowledge to be explored was that associated with planning an entrepreneurial start-up company (the entrepreneurial ideal) and developing entrepreneurial skills (analysis, decision-making, etc.). Motivational needs included *"enhancing the entrepreneurial culture"* and providing a learning activity that was engaging and fun. Although, the simulation would not be formally examined, on an informal basis learners should be able to assess their current levels of business knowledge and use this to decide personal development plans. (These are also discussed in Chapter 2 (Learning and Simulation).

c. Settling Duration
This involves deciding the amount of time that can be budgeted for the simulation. After deliberation, the duration for the SEED simulation was to be one day (six hours). As complexity is highly correlated with duration (Hall & Cox, 1994) this would limit the number of decisions and the size of the simulation model. In general I find duration a major constraint and a major problem area. It is a major problem area as clients ask me to pack too much into the simulation. As DISTRAIN was to be created by customising an existing simulation (adding decisions and reports) without lengthening the simulation, this was a concern.

d. Defining Manner of Use
This involves defining the way that the simulation would be used (as described in Chapter 4 – Design for Process).

For the SEED simulation, for university students, it would be run as a series of stand-alone seminars. But, for prospective entrepreneurs, reflecting their situation it might be run as a distance/spare time learning activity or as a course finale. In contrast, DISTRAIN and Modern Banking were designed to be used as Stand Alone seminars, Product Launch to reinforce a topic, provide a break or as an ice breaker. Prospector was to be a course theme.

Development sequence
These elements are not progressed in any predefined sequence nor are there any definite starting points. Rather, different simulation designs start from different starting points and progress, recursively, between the other rocks in this rock pool.

The SEED simulation's starting point was specifying the Target Audiences - university students and business people whom are considering starting their own business. But the development of other simulations involved other starting points. The design of Prospector had its starting point as the Definition of Learning Objectives. Product Launch (1977) and to an extent Modern Banking had Duration as the starting point. The Challenge Series (Management (1986), Retail (1987) and Service (1989)) were developed for use as part of an international promotional contest and so had Manner of Use as their starting points.

Generally, after defining the first *rock* to visit the others are visited immediately. For instance, for the Challenge Series, immediately after defining the Manner of Use, the duration was settled. This stage involves moving between and revisiting elements. For the SEED simulation, the duration was initially envisaged as one and a half days but later was reduced to one day (6 hours). This required revisiting the learning objectives (but equally might have meant redefining the manner of use). While defining needs, account

was taken that the SEED simulation might be used with prospective entrepreneurs, on an E-learning basis, and, perhaps as a management contest.

Stage outcomes

At the end of this stage there is a need to examine how the elements relate to each other and resolve any conflicts. For instance, the target audience coupled with learning objectives for the SEED simulation meant that its scope would stretch the knowledge base of the learners. Because of this, it would be necessary to build in significant learner support (in the form of business advice). Further, the short duration (a day) coupled with the Learning Objectives would cause a major development problem. And, this was amplified by the fact that the simulation would be run in a single, undivided session. (If the simulation had be spread, in separate sessions over several weeks, the learners would have the opportunity to reflect and, possibly, budget extra time.)

2. The second rock pool - simulation specification

This rock pool consists of these elements:

a. Define issues
b. Decide simulator type
c. Decide delivery mode
d. Decide version(s)
e. Decide business scenario

a. Defining issues

This involves translating the learning objectives into business-oriented issues that are appropriate to the target audience. For the SEED simulation the issues included market selection, pricing, promotion, working capital and venture funding. The issues define the discussion areas for the team and hence the areas where *deep cognitive processing* occurs. When defining issues I find it useful to look at these in the contest of the industry and the client company.

b. Deciding simulator type

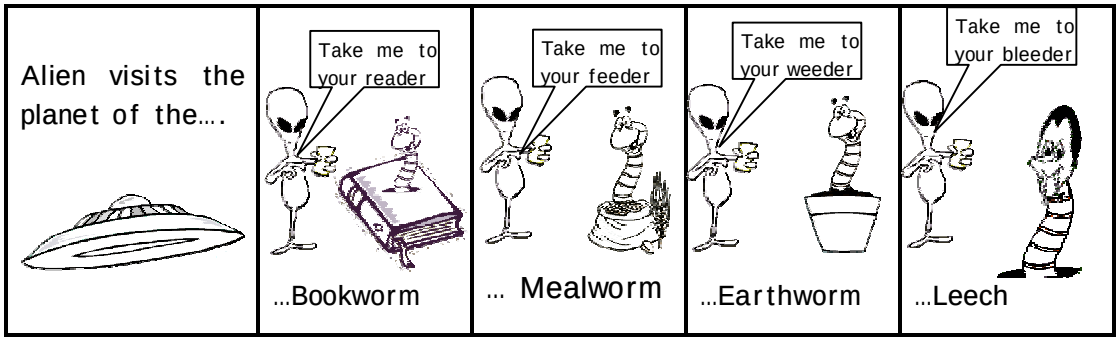
This involves deciding the structural aspects of the simulation as described in Chapter 1 (Types of Simulation). Initially, it was felt that the SEED simulation was a planning simulation where learners would utilise a what-if model to determine the *best* plan. But beside the need to perform What-If analyses, it was necessary for the learners to experience the planning process and especially the time taken to develop a business plan. This meant that learners were only allowed to make a limited number of analyses before a simulated month passed. In turn, this meant that learners had to balance the risk of an incomplete and inadequate plan against being *"fast to market"* - a situation that was magnified by the seasonal nature of the market and the need to become established before the seasonal peak.

c. Decide Delivery mode

This involves who uses the simulator – specifically whether they are *Tutor-Mediated* or *Direct Use* (Chapter 1). As the SEED simulation addresses planning issues it is non-competitive between teams and it is fitting for each team to make direct use of the simulator. Als the teams can work asynchronously, this shortens the simulation's duration. However, set against this is the fact that Direct Use of the simulation can change team behaviour (Coote, 1985) and the software must be designed to minimize this risk (Hall, 1995b). In contrast, Tutor Mediated delivery processes decisions synchronously but as the learners are divorced from the computer they focus on learning rather than the technology

d. Decide Version(s)

This involves deciding the versions the simulation will have where the range of use, learning objectives and audiences help define the decisions and reports provided by the simulation and how they differ between versions.



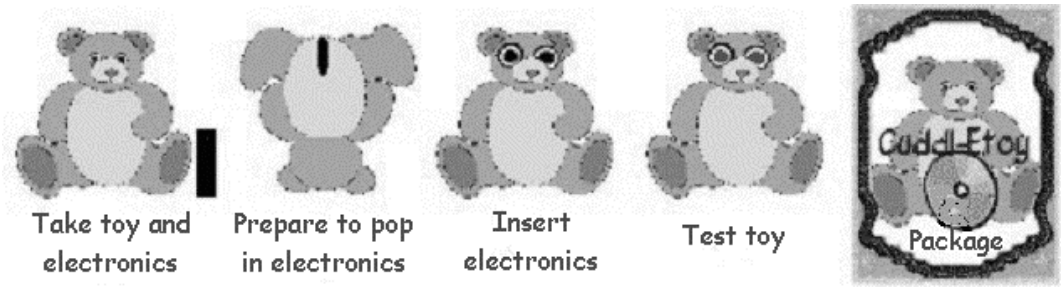
Cartoon 8.03: Versions for Different Audiences

For SEED, the different target audiences and manner of use conjoined and so it was decided to create a single version. (If different uses were desirable then several versions would have to be developed ranging from basic (for the contest) through more complex versions with additional decisions and reports.) However, it may be that at a future time, versions will be created with different tax structures and in different languages. However, contrasting with this there are multiple versions of Product Launch. To address different learning needs there are several versions including a basic one (Classic) and one where new reports ere introduced period-by-period (Progressive). To address different audiences besides Classic and Progressive (for company training) there are a schools and university version. Next to address business terminology needs both the Classic and Progressive versions are available in UK English and American English and to deal with language differences it is possible to have French, German, Spanish etc. versions. To deal with geographic differences besides SouperHot (self-eating soup) there is HotTea (pronounced hotty – a self-heating tea!)

f. Deciding the business scenario

This involves deciding the industry to be simulated (Teach, 1990). Often the type of organization that will be using the simulation predefines this and this was the case for Modern Banking. DISTRAIN and Prospector.

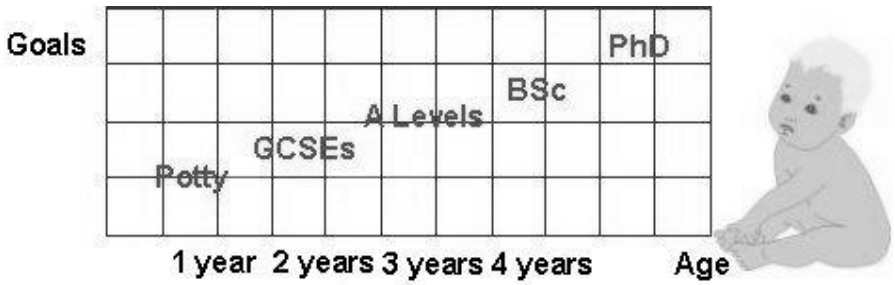
But, for the SEED simulation, although the target audience did not predefine the business scenario, there were several factors that made the choice important. First, because of the age of the target learners, it was necessary for the simulation to be engaging and the product or service should be *"real".* But because of its entrepreneurial nature the product could not be an existing product. It had to be a product that might exist because of market needs but does not because of technological or economic factors. Taking into account prior knowledge & experience, it needed to be a relatively simple business in marketing, operational and financial terms. That is to say the market structure and marketing mix should be simple and manufacture should consist of a single process with few component materials.



Cartoon 8.04: Simple Product to make

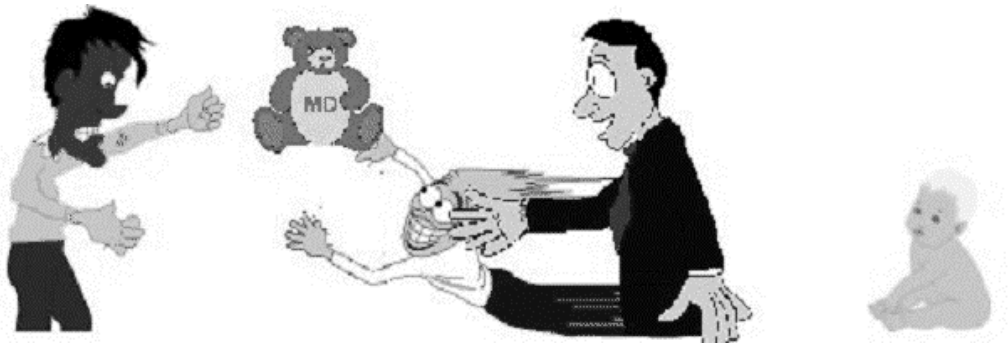
Also, it should take into account the scientific and engineering background of the learners and the product should be high tech and thoroughly modern! The product chosen was a soft toy (like a teddy bear) incorporating electronics to link it to a normal home PC and

allow it to communicate with a small child (aged between two and five). It would be used as a companion for the child and, via the PC, provide early learning. Reflecting its high-tech aspect the product is the Cuddl-Etoy! (For my American readers, GCSEs are exams taken at 16 and A Levels are exams taken immediately before university.)



Cartoon 8.05: Early Learning

The scenario explained that the Cuddle-Etoy was developed after a failed University project (where the toy was developed as a collaborative project involving maths, engineering and medical students as an aid to diagnosis for very young children). A project that failed because of difficulty with sterilisation and the screaming fits when the child had to give up the toy after the diagnosis session!



Cartoon 8.06: Failed research project.

At the second pilot we illustrated the product with several real cuddly toys. Immediately on arrival, two female students annexed these, but since they were destined for a member of the academic staff they had to be retrieved.

As part of scenario choice, it is necessary to determine the extent to which it will be simplified and stylised (Chapter 3 – The Art of Simulation Design) and the extent to which is replicates a specific industry (Chapter 6 – Design for Value).

Development sequence

Again, there is no defined starting "rock" in this rock pool. For the SEED simulation the starting point was defining the scenario. In contrast, the starting point for the development of Prospector was to define the issues that had to be considered when searching for and identifying *profitable* projects and the scenario (project details) was of minor importance. Equally, manner of use may mean that the starting point is deciding the simulator type. For instance, as the Challenge Series (1986, 1987, & 1989) was developed as part of a management contest the simulation had to be competitive with the decisions of one team interacting with the others and where business issues secondary and the business scenarios of no importance.

Stage outcomes

At the end of this stage the key structure of the simulation is defined and specified. This may be used to search for and select a suitable *existing* simulation or as the basis of the development of a new simulation (as was the case with SEED, Modern Banking,

Prospector and Product Launch) or one that could be customised (as was the case with DISTRAIN and the FMCG version of SMITE).

3. The third rock pool - simulation design
This rock pool consists of these elements:

a. Decide decisions
b. Decide results
c. Create models linking decisions & results
d. Create validation & quality assurance support
e. Develop preliminary documentation

Besides these, there is the need to design or have access to software routines to manage the user interface - decision entry, report preparation and display, on-line help system etc. However, for SEED a *software shell* was used to provide this functionality.

a. Deciding decisions
This is a creative process starting from the issues list (2a) and learning objectives (1b). For each issue the designer must ponder the cognitive processing required to ensure that the learners fully think through the issues and adequately explore the knowledge needs. For example for the SEED simulation and reflecting its entrepreneurial background, decisions separated into three key areas – Marketing, Finance and Resourcing.

Besides considering the decision areas, it is necessary to think of their *granularity* and *ambiguity* (Chapter 3, Art of Simulation Design).

b. Decide Results
This involves determining what results are produced by the simulation model divided into the categories described in Chapter 6 (Craft Elements of Design).

c. Create models linking decisions & results
This involves developing the logic and algorithms that link the decisions and results and defining the data associated with these. At this stage the model algorithms are researched and described and if necessary *prototyped* on a spreadsheet.

d. Create validation & quality assurance support
This involves creating routines and reports that reveal explicitly how the models work. With SEED, translating marketplace decisions into sales demand was very complex with multiple interacting factors that affected demand. Thus, as part of the validation and quality assurance process the individual market responses were captured and were made available in a report. A second example was the reconciliation of creditor changes.

f. Developing preliminary documentation
This involves recording and describing the decisions, results and models. This may take place before, concurrently or after particular decisions, results and models have been designed. The documentation separates into three parts - the learners' brief, the trainer's manual and online help. If a predefined *shell* program had not been used a fourth document describing the technical aspects of the software would need to be developed. If appropriate, for inexperienced trainers and academics, it may be necessary to produce a document discussing the classroom aspects of running the simulation. Again, for SEED this document existed.

Learners' Brief
Trainer's Manual
Online Help
(Using the Software)
(Using Simulations)

In parallel to the development of the online help system the *learners' brief* and the *trainer's manual* were drafted. The learner's brief is the document provided to the learners to read and become familiar with before starting to use the simulation. The trainers'

manual is for the trainers to help them understand the simulation and the learning derived from it. Additionally, at this stage, the trainer's manual serves to document the design and both are drafts and focus mainly on the decisions, results and the model.

Development sequence

In this rock pool, reflecting the level of creativity, movement between the rocks (elements) was very complex as the decisions; results and models refined and refactored. For instance, the first design work involved the marketplace models. But the complexity and dynamics of these meant that these were also the last to be completed. Effectively, this pool does not comprise a few large rocks but consists of several piles of weed-encrusted stones with a large number of agile and elusive creatures darting between them! For the SEED simulation the starting point for a cycle tended to be defining a decision or group of decisions, deciding on how they interacted and produced results and then defining the result set. However, in contrast, for the Financial Analysis planning simulation the starting point was to define what results were to be produced and only then the decisions that impacted these. For the DISTRAIN simulation, the issues (2a) were used to design the models and then the results and decisions were determined.

Stage outcomes

At the end of this stage an *alpha test* version of the simulation exists ready to be refined in the next stage into the *beta test* version. At this stage, the documentation's purpose is to support the designer rather than for use by the learners and the tutors. Thus, later, it must be modified and refined to be of use to the learners and the tutors.

4. The fourth rock pool - simulator development

This rock pool consists of these elements to bring the simulation to the beta test stage

a. Test models
b. Calibrate models
c. Ramp workload
d. Create learning & tutoring support
e. Refine documentation

a. Testing models

This involves ensuring that algorithmic logic and program code are correct. If logic (the algorithms) are incorrect the simulation will lack face validity (Woolfe, 1989) and this will affect learning (Teach, 1990). And if the simulation is incorrectly coded results will be calculated incorrectly and, even worst, the software may *crash.* Biggs and Halpen (2004) arguing a counter view about the *utility* of BUGS (defined by them as Basic Unplanned Glitch Situations) provide a taxonomy of problem areas.

The quality and validation support (3c, above) help with this process but for a complex simulation it is often necessary to develop a series of spreadsheets to reconcile results. And, for financial validity, the Balance Sheet must balance and this tests the accounting models.

b. Calibrating models

This involves ensuring the realism of outcomes and balancing results. In particular the balance between the ability to generate profits (profitability) and cash flow (survival) is crucial. For SEED it was important that no particular plan was obviously the best. This was both to ensure adequate discussion and to ensure that the teams had to face the problem of deciding when they should choose which plan they were to go with. For DISTRAIN a key requirement was that simulated companies would perform similarly to *real-world* distributors. Because in the *real-world* it was difficult for a distribution company to be successful but easy for it to be unsuccessful, DISTRAIN was calibrated so that it was easier to be successful. For Banking Challenge, a key calibration issue was to ensure that learners always had to manage Capital Adequacy (liquidity and toxic loans). For Product Launch, the simulation was calibrated so that by the end of the simulation (period 8) it was possible but very, very difficult to make a cumulative profit of one million.

c. Ramping workload
This involves increasing the number of decision areas or reports produced as the simulation progresses thus reducing the risk of role overload (French, 1972) in the early stages of the simulation. Teach (1990) discusses the same problem in terms of "analysis paralysis". And Hall (2004) suggests that by ramping the complexity as the simulation progresses not only is the risk of role overload diminished but learning is enhanced as the additional reports and decisions "stimulate discussion and provide the opportunities for additional cognitive development". For the original pilot of the SEED simulation workload was not ramped and, as described later, this caused role overload.

d. Creating learning & tutoring support
This involves defining reports, help texts and decision screens that provide information to reconcile the results, advice & explanations, knowledge support (for tasks) and identification of strengths, weaknesses and possible problems.

The standard *shell* used for SEED provided a full *online help system* with using the software, the current task, definitions of terminology and an online manual. Software help already existed but help with the current task, definitions and the online manual had to be developed as a context sensitive, hypertext database. Much of the data in the database could be copied from the draft learners' brief and background notes. But, reflecting the difference between print media and screen display, the data was edited and abstracted.

e. Refining documentation
This involves taking the draft documentation created in stage 3d (earlier) and editing it to improve clarity, remove omissions and incorporate the parameters determined during calibration (4b). And, now the major development work is complete, it is appropriate to develop a Power Point brief.

Development sequence
This rock pool is closely associated with the simulation design rock pool and generally involves some movement between the two. For instance as the models are calibrated they may need to be modified. As with the previous stages, there was movement between the *rocks* in this rock pool. In particular, creation of learning and tutoring support was closely linked with refining the documentation. As the models were tested it becomes apparent where both the online help system and the printed documentation needed to be clarified and expanded. Finally, testing and calibration are linked.

Stage outcomes
At the end of this stage the simulation has reached the *beta test* stage and is ready for piloting (testing with *real* people) and the authentication of learning.

5. The fifth rock pool - simulation validation
This rock pool consists of these elements:

a. Pilot simulation
b. Refine and modify the simulation
c. Refine and modify documentation
d. Authentication

a. Pilot simulation
This involving testing the simulation with a group of *forgiving* learners or trainers. The SEED simulation had two main pilots with several groups of students. The SEED simulation had two main pilots with several groups of students. Piloting not only tested the robustness of the models and code and how the simulation delivered learning but also tested *process aspects* - workload, behaviour and usability.

Although it was realistic to allow learners access to the full set of decisions from the start, on the first pilot this caused role overload (French, 1972). As this risk was considered when settling duration (1c) and again when ramping complexity (4c) we were ready to introduce the decisions in stages and this was done for second pilot.

Checking the *behavioural* aspects of the direct use of the simulation by the learners (Coote, 1985) was identified as a possible problem earlier when deciding the delivery mode (2c), but it did not cause problems.

b. Refine and modify the simulation
This involves taking feedback from the pilot and correcting logic and code errors. For SEED the major change was to phase the introduction of decisions and results over the first four simulated months (Figure 8.02).

Month	Planning Options
January	Business Research & Policy Advice
February	As January plus Marketing decisions
March	As February plus Resourcing & Working Capital decisions
April onwards	All decisions

Figure 8.02: Phased introduction of decisions and results

c. Refine and modify documentation
This involves checking the completeness of *all* documentation (learners' brief, trainer's manual and the online help). For the learners' brief this involves improving clarity and this may mean adding to the manual or subtracting from it. Experience suggests that the design of the learners' brief is a balance between completeness and length. For, if the brief is too long it will not be read! Because there is an online help system there is the choice between adding to the paper document or to the online help or both. I remember with embarrassment my using the word *expatriot* rather than *expatriate* when writing the participants' brief for my INTEX (Industrialising Nation Simulation) (in mitigation several dozen people read the brief before the error was discovered).

e. Authentication
This involves obtaining the views of the learners, subject matter experts (SMEs), academics and trainers and teachers. For SEED feedback from the learners was through a questionnaire.

Development sequence
Unlike the previous rock pools where there are no defined sequences, this stage involves a repeating cycle of piloting and modification until the simulation authentically meets the needs and issues defined in the first and second "rock pools".

Stage outcomes
At the end of this stage a fully operating simulation exists and all that remains is *packaging* and *dissemination*.

6. The sixth rock pool - finalisation
This rock pool consists of these elements:

a. Finalise documentation (learner & tutoring)
b. Finalise tutoring support
c. Dissemination

a. Finalise documentation (learner & tutoring)
This involves editing the documentation to improve accessibility (readability, spelling, layout etc.) and incorporate the changes made during the validation stage.

b. Finalise tutoring support
This involves the final completion of the software. For instance, after the pilot of the DISTRAIN simulation, the reports for the trainers and the learners were modified to match the requirements of the people who were to run the simulation.

c. Dissemination
To organizations, to trainers, to students (learners) and via the academic community involves press releases, printed and electronic promotional materials (such as a website) etc.

Development sequence
Again there is no defined sequence between these tasks and very often the finalized documentation and tutoring support use the same textual and graphic data in a different form. Further, for SEED some of the graphics from the Power Point briefing and the online help system were used on the website.

Stage outcomes
A complete simulation package.

Summary
Anderson (2003) uses the Lapre and Van Wassenhove Project Matrix (2002) to suggests that the choice of appropriate software methodology depends on the degree of *Operational Learning* (experience "know-how") and degree of *Conceptual Learning* (cause and effect understanding) - each of these ranks from low to high (Figure 8.03).

		Operational Learning	
		Low	High
Conceptual Learning	Low	fire fighting	artisan skills
	High	unvalidated theories	operationally valid theories

Figure 8.03: Lapre and Van Wassenhove Project Matrix

In the context of computer simulation design, the Operational Learning dimension relates to design experience. Here low Operational Learning is where the designer or designers have no or little experience of simulation design and high Operational Learning is where the designer has developed several simulations and has researched simulation design. And, the Conceptual Learning dimension relates to the *difficulty* of the current design where Design Difficulty is a combination of simulation complexity and novelty (Chapter 6 – Design for Value). The measurement of simulation complexity has been explored in several papers (Wolfe 1978; Pray and Gold 1982; Gold and Pray 1984) that link it to the number of decisions and software size (model size). Simulation *novelty* depends on the degree of innovation in simulation structure and the situation (industry) modelled. Two simulations illustrate this. In model terms and industry modelled, Prospector is simple but the structure of the simulation is innovative. In contrast, Foundation Challenge has intermediate model complexity and a standard structure, but as it models a not-for-profit organization it was an innovation for the me. These two examples also illustrate that the novelty level depends on the experience and knowledge of the designer and hence the amount of conceptual learning required. For the SEED simulation, the model was complex and the structure necessary to enable a complex simulation to have a short duration had high novelty.

The choice of the appropriate simulation software methodology depends on the complexity & novelty of the simulation and the experience of the developers as shown in Figure 8.04.

		Design Experience	
		Low	High
Design Difficulty (novelty/complexity)	Low		Stairway Method
	High	Lightweight Method	Rock Pool Method

Figure 8.04: The Project Matrix (modified)

I suggest that for simple simulations involving a few decisions and a simple simulation model a rigorous method such as the Stairway method is appropriate and desirable. Because one aspect of design difficulty is due to the *novelty* of the design to the designer, where the designers have no design experience the simulation defaults to highly novel. For this situation, a lightweight method is the most appropriate as it allows the designers to learn experientially as the simulation is created. For the situation where the simulation is novel or complex but the designer has considerable design experience the Rock Pool Method balances the rigor of structure with the creativity and flexibility necessary to learn to deal with the novelty or complexity of the simulation.

Design Example: Modern Banking

This section reprises the report on the design of the Modern Banking simulation for the National Bank of Serbia.

Background Information

This report describes the development of a Computer Business Simulation addressing the roles and businesses of the different players in the Serbian Financial Markets with a focus on the mutual understanding of the National Bank and Commercial Banks.

Outcomes

A banking simulation was developed to meet the requirements of the National Bank of Serbia to be used to train their staff and those of the commercial banks that they regulate. The simulation designed was to allow junior management and graduate employees of both the National Bank of Serbia and the commercial banks of Serbia gain a basic understanding of the working of a bank. It was of intermediate complexity and lasts a day. The simulation was designed for use by trainers from the National and Commercial banks. It runs on standard microcomputers running any version of Windows from Windows 98 through Windows XP. Besides developing the simulation software a learners' manual was written together with a trainers manual.

The simulation was successfully piloted with a group of eight trainers who then attended a Train the Trainer course. Following this, two of the trainers ran the simulation with a group of National and commercial bank staff.

The simulation software and documentation was supplied on a CD ROM for the Academy of Banking and Finance to use, copy and distribute to trainers.

Schedule

The table (Figure 8.05) summarises the work schedule for the project (note it includes 6 days travelling between London and Belgrade).

Dates	Days	Cum.	Work
17/07/05 – 23/07/05	7 days (Serbia)	7	Basic specification, initial information, began work on the learner's brief, background notes, databases and simulation models.
27/07/05 – 02/09/05	10 days (Office)	17	Complete first draft of learners' manual and database and began construction of simulation models and trainer's manual.
04/09/05 – 16/09/05	13 days (Serbia)	30	Worked on the simulation model and trainers' manual. Discussed design and obtained financial data.
17/09/05 – 26/09/05	5 days (Office)	35	Completed simulation model and trainers' manual
27/09/05 – 04/10/05	3 days (Serbia)	38	Piloted simulation with trainers and ran train the trainer course. (2 days + 1 day travel)
	3 days (Serbia)	41	Modified and re-calibrated the simulation as a result of the pilot, discussions with the trainers and staff of the Academy of Banking and Finance.
	2 day (Serbia)	43	Live simulation run (with JH shadowing trainers) (1 day + 1 day travel).

Figure 8.05: Work Schedule – Modern Banking

Design Stages

The design process was based on Jeremy Hall's Rock Pool Methodology (appendix 1). This section details the design process.

Needs Definition (17/07/05 - 23/07/05)
Simulation Specification (17/07/05 - 23/07/05)
Simulation Design (24/07/05 - 09/09/05)
Simulator Development (10/09/05 - 26/09/05)
Simulation Validation (27/09/05 - 04/10/05)
Finalise Design (04/10/05)

1 Needs Definition

This involved defining the requirements in terms of:

a. **Learning Purposes**
b. **Target Audience(s)**
c. **Duration**
d. **Manner of Use**

This information was obtained in discussions with Mr. Rautenberg (President of the Managerial Board of the Fund "Academy of Banking and Finance") and others at the National Bank of Serbia during the week of July 18th 2005.

a. Learning Purposes was to provide a basic understanding of the operation of a bank – specifically:

Finance – Understand basic financial concepts and reports (Income Statement, Balance Sheet & Cash Flow), measures of performance (profitability, liquidity and capital adequacy) and how managerial actions impact these. (The simulation will use the International Financial Reporting Standards as appropriate to banks as published up to March 2004.) But will not include foreign currency issues.

Marketing – Understand the basic marketing mix (price, promotion, 'place and product) and how these interact and affect business success.

Operations – Resourcing management, forecasting, quality & productivity Improvement and the impact of these on marketing and financial results.

Although providing learning at an *appreciation* level the design facilitates expansion to cover strategic issues (for a more senior group of learners). But this work was not part of the current project.

Besides the knowledge exploration dimension the simulation will involve learners working in small teams (of four or five learners) to develop **team working**, **negotiating** and **business presentation** skills. (Team working is seen as a major development benefit for staff at the National Bank.) Where used with staff from one bank, the simulation will encourage team-building across functions. Where used with a mixed group of National Bank and Commercial Bank staff, the simulation will facilitate mutual understanding.

b. Target Audience(s) separate into two parts - the people who participate in the simulation and the trainers who run it

Learners are junior management and graduate employees of both the National Bank of Serbia and the commercial banks of Serbia. For the first group (National Bank staff), the simulation provides an appreciation of the working of a commercial bank. For the second group (commercial bank staff), the simulation allows them to explore and understand the wider aspects of how their bank operates.

Trainers: The simulation will be run by National and Commercial banking who have an understanding of the banking business and are reasonably experienced computer users but who do not necessary have experience using simulation for business training. This necessitated a one-day train the trainer course following the pilot run of the simulation. The train the trainer course covered using simulations to train business people, using the software and the background to the simulation models.

c. Duration for the target learners the simulation lasts a day. (Although, for more senior managers, it can be run in an accelerated manner in just over half a day.)

d. Manner of Use as a stand-alone seminar or, spread over several weeks where the decisions are made in *spare-time*. Where the spare-time version might be run as an intra or inter bank contest on an annual basis. However, the ultimate design expanded this to include use as a course finale, spread over a course, to enliven a business conference and with minor modification as part of an assessment/development centre.

2 Simulation Specification

This involves specifying the simulation in terms of:

a. Business Scenario
b. Issues
c. Simulation Type
d. Delivery Mode
e. Version(s)

And the design issues associated with these.

a. Business Scenario was a commercial bank selling to two markets - retail and corporate. For each of these there are two deposit products (demand and long term) and two loan products (short and long). There are two types of operational resources (junior and senior staff). The bank promotes to new and existing clients through separate expenditures to the two markets and through a sales force that serves the corporate market. Interest rates are set separately for the two market sectors' term deposits, short and long term loans. All financial measures are in Euros.

b. The Issues facing the learners are as follows (and detailed in Appendix 2):

Banking Appreciation
Banking Objectives & Measures
Financial Appreciation
Marketing - Pricing & Promotion
Product Mix & Contribution
Forecasting & Control
Client Management & Policy
Business Improvement Policy & Impact
Team Working
(Business Presentation)

c. Simulation Type is a *total enterprise simulation* at a business appreciation level involving learners in running a complete business rather than the detailed operation of part of a business (Functional Simulation) or the exploration of a particular business concept or process.)

e. Delivery Mode is by the trainer (*tutor mediated*) where two to eight teams (of four or five bankers) interact in the same marketplaces in direct competition. Where decisions are taken from teams and processed by the trainer using a standard microcomputer (Window 98 through Windows XP operating systems) and printer (monochrome laser or ink jet). This not only allows teams to interact in the same marketplaces but minimises the need for hardware.

f. Versions: The simulation was developed for use in the classroom or as a contest. It is possible to develop an extended version that can be used by more senior management but this was not part of the current work.

Design Issues

The Banking Simulation is of intermediate complexity and is not especially novel. (Except that the simulation that some National and commercial bank staff have been exposed to in the past have been based on a generic manufacturing industry scenario (rather than a commercial banking scenario) and a *very* complex banking simulation (lasting five days)). The banking scenario had issues in terms of the accounting system; financial and marketing models and ensuring that these did not result in a simulation that *overwhelmed* the target learners.

The form of the simulation meant that some tutoring materials existed together with the major part of the computer software (simulation platform or *shell*). The availability of a simulation *shell* played a key part in enabling the simulation to be developed in the time available as the existing shell provided 92% of the software leaving only 8% (the simulation model to be developed) (see Appendix 3). (The remaining development needs are detailed in Appendix 4.)

3. Simulation Design

This involved:

Deciding decisions
Deciding results
Creating models linking decisions & results
Creating validation & quality assurance support
Developing preliminary documentation

This was started during the trip to Belgrade (July 17th - 23rd) and continued off site and during the second trip to Belgrade (September 4th – 16th).

a. Decisions consisted of:

- **Interest Rates (Term Deposits, Short & Long Term Loans)**
- **Operating Staff Numbers (Junior & Senior)**
- **Promotion (Retail & Corporate Sectors)**
- **Account Manager Numbers**
- **Quality & Productivity Improvement**
- **Loans to and from Other Banks**

These were chosen to provide a basic mix between functions (finance, marketing and operations) as explored with the Academy of Banking and Finance, the National Bank and commercial banks.

b. Results consisted of financial, operational, marketing and performance measurement reports. (Including reports for the teams, the trainer and to help check design validity, the reporting *pack* consists of 137 reports).

Financial reports are based on the International Financial Reporting Standards for Banks modified to suit the National Bank of Serbia and simplified because of the level of learner knowledge and the simulation's duration.

Operational reports provide information about staffing, their utilisation, skills and productivity and the impacts of this on costs, clients, loan defaults and business size.

Marketing reports provide information about the mix of business, customer levels and types.

Performance reports provide information about bank performance (profitability, liquidity, risk etc.) and information about the quality of decision-making.

Additionally, reports are produced that help support the trainer during the simulation and help validate the simulation models.

c. Models parallel the reports consisting of financial, operational, marketing and performance measurement models and link the decisions to the results.

d. Validation & Quality Assurance Support involves creating reports and algorithms that validate the simulation and ensure that it performs in a robust and stable manner.

e. Preliminary Documentation involved creating:

The Learners Manual
The Tutor's Manual
On-Line Help System
Decision Forms (and work sheets)
Simulation Briefing Slides

The **Learners Manual** explained the activity, the bank and includes three year's trading history and a discussion of liquidity constraints and measures.

The **Tutor's Manual** consisted of Background Notes (describing the simulation), Running Simulations and Using the Simulator (Running Simulations and Using the Simulator already existed).

The simulation software incorporated an extensive context sensitive hypertext **help system.**

The **Decision Form** is complete and validated at the pilot and full run but time constraints meant that only a single worksheet was completed.

The **Simulation Briefing Slides** provide a quick introduction to the simulation and were validated both at the pilot run and the full run.

4. Simulation Development
This consists of these elements and describes work done during the second trip to Serbia (September 4th to 16th) and completed following this.

Test models
Calibrate models
Create learning & tutoring support
Refine documentation

a. Test models involved ensuring that algorithmic logic and program code was correct.

b. Calibrate models involved ensuring the realism of outcomes and balancing results. In particular the balance between the ability to generate profits (profitability) and cash flow (survival) is crucial. During the design it involved running the model repeatedly to *tune* the parameters. Following the pilot run the calibration was refined further following the *live* run and completed so as to emphasise the issues identified by the Academy of Banking and Finance.

c. Create learning & tutoring support involved defining reports, help texts and decision screens that provide information to reconcile the results, provide advice & explanations, knowledge support (for tasks) and identification of strengths, weaknesses and possible problems. This was completed based on feedback from the pilot run. Three groups of reports were developed to help the trainer *manage* the learning process. These were the **Tutor's Audit** where teams are contrasted and compared to allow the trainer to identify learning and support needs and discuss team results during the simulation review. **Team Commentaries** provide detailed individual team information that can either be used as additional feedback to a team or as information for assessors on an assessment centre. **Reconciliations** that allow the trainer to respond quickly and authoritatively to requests for help with the accounting and operational calculations.

d. Refine documentation involved taking the draft documentation created in stage 3d and editing it to improve clarity, remove omissions and incorporate the parameters determined during calibration (4b). This was done following the pilot run, the train the trainer course and the full run.

5. Simulation Validation
This involved the following:

Discussions with National and commercial banks
Piloting the simulation with the trainers
Running the simulation with *real* learners

a. Discussions were ongoing during the design process and in particular during the second visit (September 4th - 16th).

b. The **simulation pilot** not only provided a test of the simulation but also allowed the trainers who will use it gain knowledge of the simulation. Feedback was obtained during the pilot and the train-the-trainer session following the pilot. This feedback was refined further in discussions with Mr Rautenberg and ABF staff. This lead to modifying the responses of the simulation to better explore the issues, modifying the reports produced and producing a series of *comments* to be feedback to learners to stimulate thought and discussion.

c. Running with *real* learners provided the trainers with experience and rigorously test the simulation.

6. Finalise Design
This involved providing a final report on the simulation together with updates following the first real run. The finalised design and a full set of documentation is provided to the Academy of Banking and Finance on a CD ROM that the Academy can duplicate and provide to the trainers for their use.

Work Summary

Work	Actual	Note
Simulation Model	940 Algorithms	Created during project
Simulation "shell"	9250 Algorithms	Pre-existing
Reporting Pack	137 Reports	Created during project
On-Line Help System	552 Help Pages	Created during project
Learners' Manual	11 A4 pages	Created during project
Decision Form	1 A4 page	Created during project
Work Sheets	1 A4 page	Created during project
Background Notes	18 A4 pages	Created during project
Running Simulation	19 A4 pages	Pre-existing
Using the Simulator	11 A4 pages	Pre-existing
Briefing Slides	20 slides	Created during project

Jeremy J. S. B. Hall, 6th October 2005

Appendices

Simulation Design Methodology

The simulation design uses the *Rock Pool Method*™ and Jeremy Hall's existing business simulation shells. The Rock Pool Methodology won the best paper award at the Association of Business Simulation and Experiential Learning conference in Orlando 2005. The simulation shell architecture won an Innovation Award in 2002 because it incorporated leading edge features and, at the same time, reduced simulation development time by 80%. The features and knowledge incorporated in Jeremy Hall's design led to a UK National Training Award in 2003.

Issues Addressed by the Simulation

Banking Appreciation

The simulation involves running a complete bank with decisions covering marketing (promotion, selling, sector and product focus), finance (interest rates, funding sources and mix) and operations (operating staff, quality and productivity improvement). It should help learners appreciate the general operation of a complete bank.

Banking Objectives & Measures

During the simulation, learners will be concerned with creating a profitable, growing bank that survives. They will develop an understanding of the bank's purpose, how this is influenced by their decisions and how the objectives and measures interact.

Learners without business experience may be especially challenged by the need to deal with ambiguity and make decisions without perfect information and, where, some information is provided as *qualitative comments* about the bank.

Financial Appreciation

The simulation introduces learners to financial fundamentals (Income Statement, Balance Sheet, cash flow and key measures). Optionally, using the work sheets provided, learners can test their understanding by, manually, preparing the financial reports

Marketing

Teams decide promotion for each sector. Besides considering how clients will respond, they must consider the effect of the competitors' actions, the financial and operational consequences of their decisions and servicing both existing and new clients.

An important aspect is how *press reports* may affect the corporate image of the bank.

Optionally, using the tutor's audit, the trainer running the simulation can provide *market research* to help teams identify the reasons for their results.

Product Mix & Contribution

Although only selling into two sectors (retail and commercial banking) each of these involve four product areas: Demand Deposits, Long Term Deposits, Short Term Loans

and Long Term Loans. Learners will need to decide how each of these contributes to the bank's profitability, growth and cash flow.

Forecasting & Control
With decisions covering staffing and funding, learners must forecast staffing and funding needs. This must take into account the uncertainties of the impact of their marketing decisions, competitive actions, the marketplace and a changing economy.

Client Management & Policy
Learners need to decide promotion, staffing and quality to ensure that they keep existing clients and acquire new clients in the right numbers.

Business Improvement Policy & Impact
With decisions about quality and productivity improvement, learners must balance the cost of this with the benefits obtained in the marketplace, staffing needs and costs.

Team Working
With learners working in small teams, they have the opportunity to share experience and knowledge, present and promote different viewpoints and develop their people skills.

(Business Presentation)
Optionally, at the end of the simulation, teams can make a *board presentation* covering objectives, strategies, process, the future (of the simulated business) and learning.

Available Materials
The following materials existed and so did not have to developed as part of the project:

Tutor Mediated Shell
Running Simulations
Using the Simulation Software

Tutor Mediated Shell provides the standard software to manage the simulation, for decision entry, reporting and help. For this simulation, pre-existing software accounted for 92% of the software. This meant that the development of the banking simulation model only accounted for the remaining 8% of software – a very significant saving in development time and cost. (Comparative metrics suggest that simulations of similar complexity where a *shell* is not used would take 300 or more days to create.)

Running Simulations is a chapter extracted from Jeremy Hall's book (Simulation: Virtual Business Experience) and describes the issues associated with using simulations to train business people. It covers pre-course planning, deciding team formation, managing the learning process and reviewing the simulation.

Using the Simulation Software describes how to use the simulation software on a standard PC with printer. It covers installing the simulation and running it on the computer. As a standard *software shell* was used this document existed.

Development Tasks
Learner's Manual – describes the simulation and provides three year's trading history.

Support Materials – decision forms, Power Point briefing slides and worksheet.

Background Notes – provide information for the trainer about the simulation, timetables, issues, its models and the reports produced.

Databases – consist of the data and parameters that drive the simulation model and the structures of the decision entry templates and the business reports produced. (Business reports separate into the reports for each bank showing their business results, market research reports, reconciliations (explaining how accounting figures were arrived at), a tutor's audit (explaining and highlighting differences between team performance) and team commentaries (detailing team performance).

Simulation Model – computes the impact of decisions, produces results, reconciliation and performance evaluation data. Besides totally new algorithms the model incorporated thinking and algorithms from several existing simulations (Business Focus and Profess – both designed for Barclays Bank and Service Challenge – a generic service industry simulation).

Tutor Support System – provides the reports and help screens that support the trainers. It helps them answer questions, identify learning needs & opportunities, coach & challenge the learners. Because of the range of trainers who will use the simulation, this was extensive and incorporated a context sensitive help system consisted of some 550 pages.

Train the Trainer - this allows the people who will run the simulation learn about it, how to use the simulator (software) and how to use simulation for business training. A Train the Trainer Course was run following the simulation pilot on September 29th 2005.

Structural Soundness

Just as good composition separates the good painter or photographer from the amateur, the structural elements of the business simulation are an indicator of quality and separate the good simulation from the bad one. There is a need to understand common composition/structural errors and make sure they do not occur in a simulation. These structural problems should be born in mind throughout the design and are vital when validating the finished simulation.

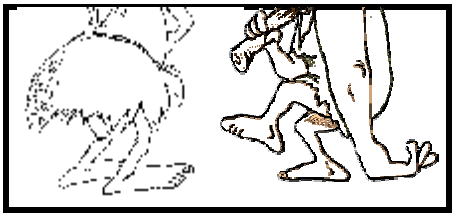
This section uses the composition knowledge base for photographs and paintings to reveal and explore simulation structural problems in terms of:

- **Lack of Focus**
- **Poor Framing**
- **No Background (no depth of field)**
- **No Foreground**
- **Lack of Perspective**
- **Lack of Contrast**
- **Too much Contrast**
- **Tangency**
- **Clutter**

Lack of Focus

This means that the simulation is not aligned with learning/development needs and may be confusing or overly complex. As a result, the simulation may be seen as irrelevant and waste learner's time. Consequentially, design must start with a definition of learning needs and continue with a definition of the issues that need to be addressed. Further, throughout the design it must be repeatedly checked against these needs and issues. Also it is important not to introduce elements that are not relevant.

Cartoon 8.07: Lack of Focus



Cartoon 8.08: Poor Framing

Poor Framing

This means that inappropriate elements are included and necessary elements are excluded. Where inappropriate elements are included, learners will waste time learning things that are not appropriate or already know. Where necessary elements are excluded, learners lose the opportunity to learn, may see the simulation as irrelevant and a waste of time.

No Background
This means that all aspects of the simulation are important and learners will not have to prioritise and consider the relative importance of different decisions and reports. So here all decisions are important with high ambiguity and there are few or no decisions that are unimportant and relatively unambiguous. Likewise, all issues are highly important. As part of the solution, you should consider when the important decisions, results and issues are introduced and not introduce them all during the first period.

No Foreground
Here everything in the simulation is of minor/minimal importance and so it does not matter what the learners do! Thus the simulation will lack challenge, be disengaging and be seen as a waste of time. The simulation has decisions that have minimal impact on results, are unambiguous and there are no results that are important or key.

Lack of Perspective
This means that the simulation has an inappropriate and confusing mix of operational, tactical and strategic interactions. For instance, if a simulation's purpose is to explore business strategy, it is inappropriate to include operational decisions such as material purchases and it may be inappropriate to include tactical decisions such as staff numbers or detailed scheduling. To overcome this you should consider the range of decisions and if it is appropriate to *intelligently* automate decisions (as was done with SEED's production scheduling decisions).

Lack of Contrast
This means that it is difficult to identify cause and effect. Specifically, the impact of a decision is minimal and insufficient to allow the learners to understand the impact of their decisions and this is made worse if decisions and results are too ambiguous.

Too Much Contrast
The other extreme, Too Much Contrast means that the outcome of a decision is too obvious and, as a result, learners do not think deeply enough and spend enough time discussing it. Where *lack of contrast* or *too much contrast* is identified during design as a possible risk then this must be checked when the simulation is piloted.

Tangency
Tangency for graphic novels *"is a bad arrangement of the elements in a picture that … creates confusion by making unrelated objects seem to be connected"* (Gertler and Lieber, 2004). In the context of business simulations, *tangency* occurs where decisions and results interact in a way that disguises, confuses or implies untrue cause-effect relationships.

Cartoon 8.09: Whose mother-in-law?

When designing Cartoon 8.9 I drew on British comedy tradition where the *husband's* mother-in-law is the butt of the joke. But later I realised that because I did not use bubbles to identify the speaker the first panel could be interpreted in two ways – the cave wife nagging the cave man (my intention) or the cave man talking to the cave wife. Thus the interpretation of the first panel involved prior knowledge about the British comedy tradition. However, the second panel, with the thought bubbles from the cave man eliminates this ambiguity. Identifying and correcting *tangency* (excessive ambiguity) requires judgment as some ambiguity is required to ensure deep thought.

For Tutor-Mediated simulations the Tutor's Audit and Team Commentaries reveal cause and effect. For both Tutor-Mediated and Direct Use simulations *comments* can help untangle cause and effect relationships.

The SMITE simulation presented very considerable problems as sales performance was influenced by the sales person and the sales area – each in several ways (Figure 8.06. Beyond this, factors such as morale are impacted by training, time spent with a manager, salary, prospecting, numbers of customers, travel and skills. To expose the reasons for good or bad sales performance, the simulation makes extensive use of comments from individual sales people, comments from "head office" and feedback from the trainer based on the Tutor's Audit and Team Commentaries.

Sales Person
Personality (Effort)
Morale
Customer Knowledge
Selling Skills
Merchandising Skills

Sales Area
Sales Potential
Number of Purchase Points
Number of Outlets
Degree of Competition
Gross Margin
Working Capital Needs
Travel Time & Distance

Figure 8.06: Sales Influences

Clutter

This means that there are too many decisions, results and issues for the learners to consider. Additionally, this is made worse if the decisions have significant interactions and are ambiguous. Also, if there are too many results this may lead to *"analysis paralysis"* (Teach, 1990) where the learners are overloaded with detail. To overcome this you should consider the relative *granularity* of results and using *comments.* I had this problem with the SEED simulation. Initially, all decisions could be made from the first simulated period. But I found during the first pilot that this was too confusing and I then introduced the decisions in stages. However, this was at the expense of the learners facing the real world issue of prioritising the work they needed to do when planning the new enterprise. In other words, by defining the planning sequence, I reduced clutter but at the expense of learning about an entrepreneurial planning issue.

Logic Soundness

Over the years, generally from bitter experience, I have built up several approaches that are designed to ensure the quality of the simulation model and reduce the likelihood of (often-catastrophic) logic bugs. These approaches are thus:

Employ double entry bookkeeping
Check model dynamics
Reconcile interim calculations

Employ double entry bookkeeping involves modelling all the financial accounting transactions. If these are correct the Balance Sheet balances. If they are not there is a logic error. For the Modern Banking simulation, this was a vital step as there were several components that were a cost (in the Profit and Loss Account) that did not involve cash expenditures. I have found that except for my simplest simulations, this approach has caught logic errors on a regular basis. (Also, as checking whether the Balance Sheet is embedded in the accountant's psyche, if you do not do this, then any accountants in the learning group will bitch loud and long about this problem and their perception that all the models are wrong – be warned.)

Check Dynamics means that besides checking how the model processes data for a single period you muse check to see how it dynamically responds to decisions over time. Let me explain with two examples - the first from DISTRAIN and the second Modern Banking. For DISTRAIN the simulation had to replicate the issues facing a distributor of electrical equipment and three interacting factors interacted. Low profit margins, inventory levels and long term availability meant that if a company got a reputation for not being able to supply sales demand it was impossible to recover. Although this is realistic it is not acceptable with a learning simulation as it would cause disaffection. With Modern Banking, a major issue is *capital adequacy* – the ability of a bank to handle loan defaults and repay deposits on demand. If the bank has an inadequate level of capital then the

Central Bank will force the bank to take very high cost loans and not grow its loans from customers. This can lead to the bank being insufficiently profitable to regain its adequacy. Both of these problems were identified during the design process and the simulation dynamics checked to prevent the teams getting into a position from where it would be impossible to recover

A second dimension to dynamic problems is the ability of participants to understand the dynamics. Product Launch exemplifies this. Originally, the price changes were modelled thus – price increases were recognised immediately and decreased sales but price decreases were not recognised immediately and so there was a delay before the full sales increase was achieved. This was realistic but participants did not recognise the impact and the delay in response to price decreases was removed.

Reconcile Calculations means that you show the results of interim calculations. Most of these will in the Reconciliations Reports. (The Reconciliation Reports are provided to help explain how the simulation calculated results if the participants need to know.) Figure 8.07 shows a reconciliation report for the DISTRAIN simulation. Its Profit and Loss Account (Income Statement) showed a single result (Operating Expenses) that was made up of several costs. The way this was calculated was shown in a reconciliation report and each of these costs had its own reconciliation report.

Marketing Cost	1000
Sales Force Cost	250
Delivery Cost	1250
Warehouse Cost	5500
Additional Expenses	700
Volume Discount	0
Payment Discount	0
Improvement Expenses	0
Market Research	0
Operating Expenses	8700

Figure 8.07: Reconciling Expenses

Besides reconciling accounting and operational (white box) calculations, there is a need to show how the "black box" model works. It may be necessary to provide other reports to help with the "calibration" of the model and to check dynamics.

Language Soundness

As described in Chapter 6 (Craft Elements of Design), I design my simulations using Visual Basic rather than using a spreadsheet. Beyond legacy and familiarity issues there are issues about suitability for modelling and errors. At first glance, as spreadsheets are designed to facilitate model construction, they would seem to be ideal. But as explained in this section, I do not believe that this is true. But because of the widespread use of simulations, I now explain why spreadsheets are inappropriate for simulations for use for business learning and where their use is appropriate.

Spreadsheets are Inappropriate

Although I use spreadsheets for prototyping and business planning, I feel that they are inappropriate for building simulations that are robust, easy and quick to use, easy and quick to design and can be customised and updated. There are several reasons for this including the difficulty of documenting the model algorithms, the lack of easy separation of the data and the model, the difficulty of separating the standard components and a lack of structure. This, I believe, leads to a less reliable, flexible and robust simulation that takes **significantly longer to design**.

Recently, I had the opportunity to compare one of my simulations with a similar simulation based on a spreadsheet. The spreadsheet simulation consisted of several dozen linked spreadsheets. Despite the model being of similar complexity and the simulation having a similar duration, compared to my simulation, it had taken ten times as long to develop.

The ***error potential*** and ***auditing problems*** with spreadsheets have been recognised for a considerable time. Research by Coopers and Lybrand (Freeman, 1996) suggested that 90% of spreadsheets with more than 150 rows (logic steps) had at least one significant error. As a simulation of intermediate complexity will be typically six times more complex and assuming the same rate of significant error, the chance of an error free spreadsheet

simulation model is virtually zero. Rajalingham et al (2000) cite a KMPG study showed a 92% error rate. A recent article entitled "Beware the Perils of Spreadsheets" in the Financial Times (Cook, 2006) suggests that "between 78 and 97 per cent of spreadsheets contain *"serious material errors".* And then lists the cost of these to companies. Panko (2005) reviewed studies of spreadsheet errors and found 94% of the spreadsheets in the studies had errors. Panko then cites discussions with experienced spreadsheet auditors who stated, *"that they had **never** seen a **major** spreadsheet that was free of errors"* (bold mine).

The ***structural*** issues associated with spreadsheets are discussed by Howard (1999) who suggests that despite spreadsheets allowing non-programmers to build models they can result in *"spreadsheets that are inflexible and difficult to maintain"* and he also stated, *"it is difficult to look at spreadsheet outputs in different ways"*. (As described earlier, a simulation's reports are key and numerous. This is illustrated by Modern Banking's 137 different reports and DISTRAIN's 209 different reports.) A white paper from Cognos (2006) suggests *"Spreadsheets offer flexibility, but without structure"* and, based on this, suggests that they are difficult to update (or customise). In this context the report says *"making such changes in a large, complex spreadsheet requires both an inordinate amount of time and great care".* The white paper also lists problems with usability, reporting capability, relevance and credibility. But it accepts that *"spreadsheets have proved a useful productivity tool"*. It is perhaps this last statement that highlights the problem with spreadsheets. They are very, very useful to prepare *small, personal* models that are not used by third parties in a time-critical situation. But simulations are not small models and they will be used by third parties (trainers). As discussed in Chapter 6 (Design for Value) speed and easy of use is critical for efficient and effective learning.

Finally, when creating a simulation model using a high level language, it is possible to *step* through the logic checking the algorithms step-by-step and this simplifies and speeds checking.

Spreadsheets are appropriate
Although spreadsheets are inappropriate for the simulation, they are incredibly useful to:

Check calculations
Calibrate the Models
Analyse History
Prototype
Investigate problems

However, deliberately, these are very small, simple spreadsheets ranging in size from a few calculations to rarely more than two-dozen calculations.

Checking calculations involves building a series of *small* spreadsheet models in parallel with the main simulation model to check calculations. Generally, I find it useful to replicate a simulation's Profit and Loss (Income Statement), Balance Sheet and Cash Flow calculations. Depending on the simulation, I use spreadsheets to check other calculations (such as manufacturing or materials flow). Figure 8.08 shows a list of some of the spreadsheet models I created to check SEED's calculations.

Innovation Diffusion
Price/Margin Sensitivity
Production Cost/Investment
Production Volume/Cost Analysis
Selling Method Analysis
Bank Financing
Inventory/Sales Tax
Product Depth
Outlet Penetration Factors

Figure 8.08: Calculation check models

Calibration involves adjusting parameters to ensure that the simulation has an appropriate economic response. Here spreadsheet models are used to determine how sales, profit, cash flow etc are likely to change during the simulation. (as explored in the Price/Margin Sensitivity Model (Figure 8.08).)

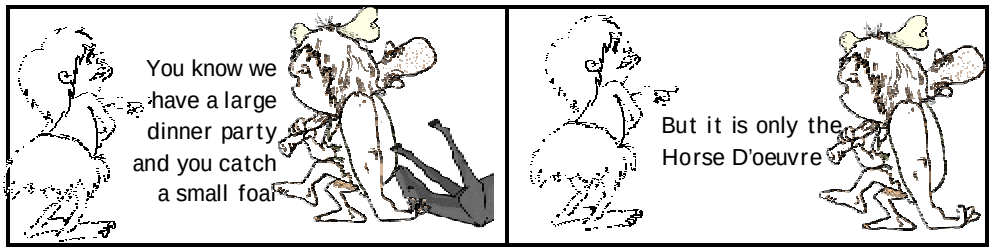
Analysing History is appropriate where the learner's brief includes several periods' past history. Here one or more spreadsheet is used to *analyse* trading history to help with a SWOT Analysis and hence identify what the learners should be able to perceive from an analysis of the history in the Learner's Manual.

Prototype using a spreadsheet involves building a series of very small models to be used to design complex algorithms. When the true model is created it can be checked back against the spreadsheet model. This approach uses the rapid prototyping benefits of spreadsheets. Prototyping was key to the design of SEED and to a lesser extent DISTRAIN.

Investigating problems is required when the simulation model is not behaving correctly. When creating Modern Banking, I had major problems with cash flow – I just could not get the Balance Sheet to balance. It took me well over a day to solve the problem. As part of the solution, I created a spreadsheet model that replicated the Profit and Loss and Balance Sheet calculations the main model. Eventually, I discovered the problem was caused by the way I handled impaired loans.

Summary

Consequentially (as illustrated in Cartoon 8.10) I feel that spreadsheets are ideal for small models used by the designer but not appropriate for large models that are used by third parties and where processing speed is crucial.



Cartoon 8.10: Language must be appropriate

Software Soundness

Having developed many simulations over the years I have made many errors that this section attempts to list, describe and where appropriate suggest solutions. After considerable devious creativity I arrived at the mnemonic *LOVE AN ODD FAILURE* to identify error types thus:

Language Errors
Overflow Errors
Variable Errors
Endless Loop Errors
Array Errors
Needs Change
Order Errors
Duplicate Calculation Errors
Data Errors
Formula Errors
Accounting Errors
Initialisation Errors
Logic Errors
Unexpected Decision Combination Errors
Rush
Extreme Decisions Errors

Language Errors

These occur when an unrecognised language statement is used (during design) or where the language cannot process the statement (when running the simulation). Generally, during design, unrecognisable statements are trapped by the development system or when the simulation is run initially. But some errors of this type that are not trapped by the development system but may be found when the simulation is run initially are incorrectly terminated or nested conditionals and loops and missing parts of loop and conditional statements (Figure 8.09). The practice of indenting logic within conditionals and loops helps overcome this problem as it eases finding missing terminations and incorrect nesting.

```
For Company = 1 to AllCompanies ‘loop through all competitors
 UnitSales(Company) = 0 ‘initialise company sales
For Product = 1 to AllProducts ‘loop through all products
UnitSales(Company) = UnitSales(Company) + ProductSales(Company, Product)
MarketSales = MarketSales + UnitSales(Company) ‘accumulate sales
Next Company
```

Figure 8.09: Language Error: Missing loop end statement

Overflow Errors

This occurs when a calculation produces a number that is too large. Usually an overflow error is caused by dividing by zero (Figure 8.10a, b and c)

```
InventoryTurn = UnitSales / UnitInventory ‘fails when there is no inventory
```

Figure 8.10a: Divide by Zero Error

```
If UnitInventory = 0 Then ‘check for zero inventory
   InventoryTurn = 999 ‘set to very high number
Else ‘inventory is greater than zero
   InventoryTurn = UnitSales / UnitInventory ‘OK
End If
```

Figure 8.10b: Divide by Zero Trap

```
InventoryTurn = UnitSales /(UnitInventory + .0001) ‘protects against no inventory
```

Figure 8.10c: Alternate Divide by Zero Trap

Figure 8.9c simplifies logic at the expense of a slight inaccuracy and is only usable where the divisor can never be negative.

A second cause is with a high level language where an integer variable is used inappropriately. For example, although it may be wholly appropriate to use integer variables for production and inventories, this may cause a problem (Figure 8.11a).

```
Production = 25000 ‘production decision can be large
PreviousInventory = 10000 ‘high previous inventory
UnitInventory = PreviousInventory + Production ‘result too large
```

Figure 8.11a: Integer Overflow (inappropriate integer use)

A second example of integer overflow may occur due to calculation sequence (Figure 8.11b). Here, although the result is 25, while calculating market share the company sales are multiplied by 100. This leads to an interim value of 35000 that is too large. The Figure 8.11b problem would be eliminated by placing brackets around the CompanySales / MarketSales calculation and calculate this before multiplying CompanySales by 100. (This is a *feature* of Visual Basic 6 – I have not tried to replicate it in Visual Basic.net.)

```
CompanySales = 3500 ‘unit sales made by the company
MarketSales = 14000 ‘unit sales total for all companies
MarketShare = 100 * CompanySales / MarketSales  ‘percentage market share
```

Figure 8.11b: Integer Overflow (calculation sequence)

Variable Name Errors

These occur when a variable name is mis-spelt (Figure 8.12). Here, the variable name UnitInventory in the conditional statement is mistyped with the *e* and the *n* transposed. (Commonly, high-level languages can force you to *declare* all your variables and although with Visual Basic declaring variables is optional, I always declare my variables to protect against mis-spelt variable names.

```
If UnitInvnetory = 0 Then ‘check for zero inventory
    InventoryTurn = 999 ‘set to very high number
Else ‘inventory is greater than zero
    InventoryTurn = UnitSales / UnitInventory ‘OK
End If
```

Figure 8.12: Miss-spelt Variable Name Error

To help readability, I use the PascalCase naming style for variables, routines and functions. The PascalCase naming style uses as identifiers of two or more words each start with a capital (Balena, 2002).

After finding an error, I use *Find* to see if I have made the same mistake elsewhere.

Endless Loop Errors

These occur when a loop is never exited because either the exit conditions are never met or there is a zero step size. For example in the Prospector simulation the learners *search* for project opportunities that met the criteria that they defined. This involved randomly generating project opportunities until one met the defined criteria (Figure 8.13a). Unfortunately, if the criteria decided were too tight a project opportunity would never be found. To overcome this problem the logic was changed to that in Figure 8.13b. As each opportunity is tested against the criteria, the number of opportunities tested is incremented and if this number is significant then the loop is exited and the learners informed that *“No match found – criteria too exacting”*.

```
Do
    GetOpportunity(OpportunityParameters()) ‘randomly generate a project opportunity
    CriteriaMatch = Function (DefinedCriteria(),OpportunityParameters()) ‘check for match
Until CriteriaMatch = “Y”  ‘exit when match
```

Figure 8.13a: Endless Loop Error

```
OpportunityNumber = 0 ‘intialise number of opportunities tested
Do
    GetOpportunity(OpportunityParameters()) ‘randomly generate a project opportunity
    CriteriaMatch = Function (DefinedCriteria(),OpportunityParameters()) ‘check for match
    OpportunityNumber = OpportunityNumber + 1 ‘increment opportunities tested
Until CriteriaMatch = “Y”  or OpportunityNumber > 10  ‘exit when match or no match likely

If CriteriaMatch = “N”
    DisplayComment(“No match found – criteria too exacting”) ‘feedback to learners
End If
```

Figure 8.13b: Endless Loop Trap

Array Errors
These occur when there is an attempt to access an illegal or incorrect array element. In terms of spreadsheets this is where an element in a formula points to the wrong cell. Figure 8.14 illustrates the situation where the Product variable is used in the UnitSales array rather than the Company variable. (Compare Figure 8.13 with Figure 8.8). Note, where there are more products than companies, the software may *crash* with a *Subscript out of bounds error.* Even if this does not happen the UnitSales calculation is wrong!

```
For Company = 1 to AllCompanies ‘loop through all competitors
   UnitSales(Company) = 0 ‘initialise company sales
   For Product = 1 to AllProducts ‘loop through all products
       UnitSales(Product) = UnitSales(Product) + ProductSales(Company, Product)
    Next Product
MarketSales = MarketSales + UnitSales(Company) ‘accumulate sales
Next Company
```

Figure 8.14: Incorrect array element referenced

As before, if you have made this type of mistake once it is possible that you have made it elsewhere and so check.

Needs Change Errors (design creep)
These occur when the client changes their needs during design and this requires the models to be changed or extended. A second situation is where after using a simulation for a period of time, there is a need to update or extend it. Because of the creative nature of the simulation design process some design creep is inevitable. But having said that, changing the design and in particular the models once they have been designed can lead to problems as the change can interact with existing models. Updating or creating a new version later can also introduce errors, especially where the model is not well documented. Recently, I revisited my UMIX simulation to update it. This simulation was created in 1990 in BASIC for the MSDOS operating system. Because of this there were virtually no comments in the code and the simulation did not have online help explaining variables. As a result I had to spend considerable time delving into the models to determine how variables were calculated and creating additional reports that explained and reconcile calculations. Turning this into a benefit – revisiting this simulation after nearly twenty years meant that I examined the simulation from the user’s viewpoint rather than the designer’s. Thus it showed me where the documentation was incomplete and needed to be added to.

Order Errors
These occur when calculations are performed in the wrong order. For example (Figure 8.15a) MarketShare for a company is calculated before calculating the total sales for the market (MarketSales).

```
MarketShare = 100 * CompanySales / MarketSales  ‘percentage market share
…..
…..
MarketSales = 0 ‘initialise total sales in the market
For Company = 1 to AllCompanies ‘loop through all competitors
     MarketSales = MarketSales + UnitSales(Company) ‘accumulate sales
Next Company
```

Figure 8.15a: Calculation Sequence Error

A second problem is where the arithmetic operations in an individual calculation are done in the wrong order (Figure 8.15b). Here, if you were using a calculator the answer would be correct. But as programming languages perform divisions and multiplications before subtractions (or additions), OperatingExpenses will be divided by SalesRevenue and

multiplied by 100 before being subtracted from GrossProfit. However, as OperatingProfit should be subtracted from GrossProfit before dividing and multiplying, this sequence can be enforced by using brackets (Figure 8.15c)

```
ProfitToSales = GrossProfit – OperatingExpenses / SalesRevenue * 100
```

Figure 8.15b: Arithmetic Sequence Error

```
ProfitToSales = (GrossProfit – OperatingExpenses) / SalesRevenue * 100
```

Figure 8.15c: Using brackets to define Arithmetic Sequence

As arithmetic sequence problems tend to occur where the calculation is complex, I make generous use of brackets to clarify the calculation sequence and, often, break the calculation into several steps.

Duplicate Calculation Errors

These occur when a calculation is done several times. For example (Figure 8.16), the accumulation of TotalSales is done twice and because there are multiple calculations (indicated by ….) between the two the duplication is not apparent.

```
For Product = 1 to AllProducts ‘loop through all products
   TotalSales = TotalSales + UnitSales(Product) ‘accumulate unit sales
  …..
  …..
  TotalSales = TotalSales + UnitSales(Product) ‘accumulate unit sales
Next Product
```

Figure 8.16: Duplicate Calculation Error

Data Errors

These occur when the wrong value is used for a variable. This is a particular problem where data is assigned in program code rather than using data files. (The values assigned in this manner are known as *Magic Numbers* and are seen as poor practice (Martin, 2009).) This is a problem that is compounded where the same datum is assigned in several places in program code as it is then necessary to change all occurrences. As you will need to change data when *calibrating* the simulation, when updating the simulation later, when creating new versions or when changing currencies data errors are a risk. In the context of currency, whenever possible I used a *universal* currency the “Account Unit” (or AU). Not only does this mean that the simulation can be used around the world but there is no need to update the simulation to take into account inflation. (The only exception is North America where I am generally forced to use dollars!)

Formula Errors

These occur when an algorithm is theoretically wrong (Figure 8.17a) or incorrect (Figure 8.17b).

```
SalesDemand = NominalSales * (PriceDecision / NominalPrice)
```

Figure 8.17a: Formula Error (wrong algorithm)

Figure 8.16a shows SalesDemand increasing as price increases (rather than reducing).

```
ClosingInventory = OpeningInventory + Production – SalesDemand
```

Figure 8.17b: Formula Error (unreasonable result)

Figure 8.16b when SalesDemand is greater than OpeningInventory + Production this leads to negative ClosingInventory (usually incorrect). (This may seem obvious, but there was a classic situation in the early days of Corporate Modelling where a planning model

was built where inventory was allowed to be negative and this led to a totally bogus business plan because actual sales were overstated substantially!)

To minimise the chance of formula errors it is useful to provide a *reconciliation* report to show how the result is calculated. Figure 8.17c shows an example of such a report but with an additional line showing Available Inventory to help clarify the relationship between Closing Inventory and Sales Demand.

Inventory Movement				
	Premium	Standard	Economy	Total
Opening Inventory	11606	0	6445	18051
Inventory Purchases	14000	19000	14000	47000
Available Inventory	25606	19000	20445	65051
Sales Demand	24472	16548	18941	59961
Closing Inventory	1134	2452	1504	5090

Figure 8.17c: Calculation Reconciliation

Accounting Errors

These occur when accounting calculations are incorrect. In particular a value is incorrectly assigned to a cost, cash expenditure, asset or liability. Such problems are characterised by the Balance Sheet not balancing. For example, a few years back I was asked to move an existing simulation (designed by an economist) from one platform to another. When I checked whether the Balance Sheet balanced I discovered that it did not. On investigation I found that the cost model and the cash flow models were designed by two different people and some costs were included that did not have matching cash payments.

I always use the Balance Sheet and double entry book keeping to validate the accounting models where the simulation involves costs, cash flow, assets and liabilities (as was the case with Modern Banking. DISTRAIN and SEED). This was vital for the Modern Banking because the novelty (for me) of banking accounting meant that initially, the Balance Sheet did not balance because I had modelled Loan Impairment incorrectly.

Initialisation Errors

These occur when a variable is not initialised properly. In Figure 8.18, we are accumulating sales across several product lines and if the TotalSales is not initialised to zero before doing this the TotalSales figure 8.will be wrong.

```
For Product = 1 to AllProducts ‘loop through all products
    TotalSales = TotalSales + UnitSales(Product) ‘accumulate unit sales
Next Product
```

Figure 8.18: Initialisation Error

To help minimise the chance of initialisation errors, run the simulation for several periods (to check for period start initialisation) and reconcile (as in Figure 8.15c).

Logic Errors

These occur when a conditional statement is incorrect because the wrong conditional is used or because of rounding problems (due to the way binary handles decimal numbers). In Figure 8.19a the conditional should be less than or equal (<=) rather than greater than or equal (>=).

```
If UnitInventory >= 0 Then ‘check for zero inventory
    InventoryTurn = 999 ‘set to very high number
Else ‘inventory is greater than zero
    InventoryTurn = UnitSales / UnitInventory ‘OK
End If
```

Figure 8.19a: Conditional Logic Error: Wrong Conditional

Figure 8.19b shows calculations in Microsoft® Works Version 7.0. Column C is calculated by subtracting column B from column A. Column D is the same number typed in and column E is column D subtracted from column C. (However, in Microsoft® Excel 2000 all elements in column E are zero. Because of this a conditional based on two numbers equalling zero can produce the wrong action (as illustrated in Figure 8.19c where with Microsoft® Visual Basic 6.0 the program will process the *not equal logic!*)

	A	B	C	D	E
1	525	0	525	525	0
2	0	1200	-1200	-1200	0
3	2750	6807.47	–4057.47	–4057.47	4.54747E–13
4	0	154.21	–154.21	–154.21	0
5	2435	1425.62	1009.38	1009.38	–1.13687E–13
6	650	468.84	181.16	181.16	–2.84217E–14

Figure 8.19b: Problem with decimal fractions (Microsoft® Works Version 7.0)

```
If (2750 – 6807.47) = –4057.47 Then ‘check for equality
        ‘logic when equal
Else ‘not equal
        ‘logic when not equal
End If
```

Figure 8.19c: Problem with decimal fractions (Microsoft® Visual Basic 6)

If this is a problem with language that you use or if you are concerned that it might cause the wrong code to be executed, you can use the ABS or Absolute function to change a negative number into a positive number (and leave positive numbers as they are). This is illustrated in Figure 8.19d, where when the difference between two numbers is approximately zero (less than .000001 in the example) it is assumed that the two numbers are the same.

```
If ABS((2750 – 6807.47) – (–4057.47)) <.000001 then
        ‘logic when equal
Else
        ‘logic when not equal”
End If
```

Figure 8.19d: Decimal Fraction Trap

Whenever there are conditional checks involving decimal numbers take care!

Unexpected Decision Combination Errors occur when one decision interacts with another or others. For example, I had one group of learners who felt that it was appropriate to exit one of two markets (half the business). In business terms this was very, very foolish as it meant that the business was now too small to cover fixed costs and make profits. Unfortunately, the combination of no production, no sales and no inventory caused the software to crash. Biggs and Halpin (2004) cite another case where if *all* three production constraints were exceeded there was no limit to the production level when there should be. **Always assume that *some* learners will make *silly* decisions.**

Rush Errors

These occur when the design is rushed because of inappropriately short design times. Regularly I have found that working long hours developing a new simulation is a recipe for disaster. This is because if I continuing working when I am tired I begin to make more and more errors. Errors ranging from the ones described here to deleting complete programs or files. Under these conditions, regularly backing up work is absolutely necessary as is stopping working when you are tired. For example, many, many years ago I was writing a sales forecasting simulation on an Apple II microcomputer. Having

written it, instead of saving, I ran the simulation – only to discover that Apple II's graphic memory was in the middle of program memory. As a consequence, I wiped out a large part of the software that I had just written (and I have avoided using Apples ever since).

Extreme Decisions Errors
These occur when a very large or very small decision causes the model to produce silly results. A (possibly apocryphal) situation is where a simulation accepted a price of one million and still sold one unit resulting in an exceptional profit. A situation that occurred regularly with one of my clients was that if they found a product range or market sector slightly difficult to manage, they withdrew from it even though this was foolish. These premature withdrawals caused the simulation to crash until I recognised this as a client characteristic. (Interestingly, in their real business the senior management did the same thing and this resulted in them completely destroying shareholder value!)

Soundness Testing

Key to determining *design soundness* is testing it with client and the pilot(s). These provide *stress* tests of the simulation models, their calibration, the learners' documentation, the Tutor Support System and simulation use. If you have sufficient time then a three-stage process consisting of Alpha Test, Beta Test and then Piloting is desirable. Fripp (1993, p81) suggests a seven-stage process, but I never found it possible to spend such a length of time. Actually, because of time-restrictions, I have often had to eliminate the Alpha Test and run the Beta Test as part of a Train-the-Trainer Course.

General Issues
Simulation software has all the problems associated with software design plus more! This is because of the range of input values and options (decisions) and processing complexity (with multiple paths through the model) means that it is unlikely to be possible to test all combinations. Testing is further compounded if the simulation is stochastic and/or the simulation is interactive between teams.

With normal software it is possible to define data entry possibilities accurately. But, with simulations, learners must be given freedom to make a wide range of options for each decision variable. Also, the decisions may interact with each other and with the current business situation. This means that even though the simulation has comprehensive decision screening, it is difficult to predict all eventualities. A similar situation occurs with logic processing where the processing sequence is complex and depends on the decisions and the current business situation.

Alpha Test
At this stage the *basic* simulation model will be completed but not fully *stress* tested. Although the first draft of the Learners' Manual complete, the Tutor and Learner Support Systems and the Tutor's Manual will be incomplete.

This test will be concerned with testing the simulation software:

for software bugs
for logic errors
identifying errors and lack of clarity in the documentation
tutor/learner support needs

And

adjusting the simulation's calibration
checking and adjusting period-to-period progressions
checking the workload

Software bugs
These are the problems that stop the simulation working completely because the computer cannot process the logic. These have been discussed earlier – "LOVE AN ODD ERROR" and most should have been identified during testing.

Logic errors
These are problems with the calculations and occur when an algorithm or formula is entered incorrectly or where the wrong variables are used. To help reveal these errors where the logic is complex, I build in extra *design* reports. This was particularly relevant for SEED where the taxation model was particularly complex and these special design reports helped me correct several logic errors.

Documentation problems
These range from simple typing errors to where the learners' manual, trainers' manual and online documentation are unclear, incomplete or too long. The Learners' Manual should be provided to the testers in electronic form so that they can make corrections directly.

Support Needs can only be fully identified after experience with actual use. This is because of the diversity of backgrounds for both the learners *and* the trainers. (In the latter context, I have one long-term user who has his own version with Tutor Support reports designed specifically for him.)

Recalibration/Modification
This may be necessary even though the behaviour of the simulation (calibration) was done during design. This is because calibration involves balancing challenge and difficulty *throughout* the simulation. For example, the Alpha Test version of the DISTRAIN simulation assumed that the customers of Electrical Distributors would require inventory to be available *off-the-shelf* and, if there were repeated inventory shortages, the company would get a reputation for poor customer service. In turn, this meant that this reputation would depress sales and reduce profits. But, with low profit margins it became impossible to generate enough cash to fund the necessary inventory levels – a situation that led to terminal decline (and very disengaged learners). Because of this, the customer service aspects of the simulation were removed.

Adjust Progressions
This involves changing the timing when new decisions and results are introduced. This was done for the SEED simulation. Initially (the alpha test version) of SEED involved the learners making decisions about all the planning activities. The rational was that in the *real world* the budding entrepreneur would have to prioritise planning decisions as it would be wrong to do some before others. (For example it would be wrong to decide price before researching markets.) Unfortunately, although real, this was too difficult and the simulation was modified to introduce planning decisions stage-by-stage.

Check the Workload
This involves seeing the amount of time to run the simulations and make decisions is reasonable (neither too much or too little). I remember sitting in on a simulation where the designers had obviously not checked this and as a consequence allowed much too little time for decision-making. As they were forced to extend the time available repeatedly, the learners became more and more frustrated and disaffected. A factor here is where a simulation that has been used spread over a course is then used in a single session. This is because between the two there is a significant change in the amount of time to reflect and the need for rapid decision processing.

It may help if some of the people doing the Alpha Test have no business knowledge as they will make *daft* decisions and this is a good way of testing the simulation logic. (I remember well the pilot of the Benson & Hedges Management Challenge in 1984, where an executive from the client's advertising company increased advertising spend 100 times – the model responded appropriately but the extra sales were much, much less than those needed to cover the extra costs – the client was very amused but the advertising executive was never amused no matter how many times over the ensuing months we reminded him.)

Beta Test/Piloting
This stage continues with the testing but the software bugs and the logic errors should be virtually but probably not totally eliminated. Thus the emphasis moves from testing the software to improving the documentation and support need and, getting the *calibration* right.

The people doing the Beta Test should be business trainers and it can be useful to combine this with the Train-the-Trainer course or if pressed for time with the pilot. For DISTRAIN simulation, the first day of the Train-the-Trainer course involved the trainers running the simulated business with the key part of the second day discussing the Learner' Manual and the Tutor Support System.

Piloting Process
This section covers selection of the learners for the pilot, timetable issues, tutoring and the software.

Pilot Learners need to understand and be happy that there may problems during the run. An ideal audience for the pilot are the people who will run the simulation. Not only will this ensure that they have a deep understanding of the simulation from the learners' viewpoint but can provide useful feedback for the post pilot modification.

Pilot Timetable consists of the normal simulation timetable (preparation, simulation and review) plus an extended the review to discuss where the simulation needs to be modified and, possibly, a train-the-trainer session.

For Banking Challenge and DISTRAIN (both one day simulations), the pilot lasted two days. On the first day the learners experienced the simulation and on the second day there was a combined pilot review and train-the-trainer session.

Tutoring consists of the normal work supporting the learners plus recording modification needs and, possibly, software modification (see The Software).

Post Pilot Shadowing involves the designer *shadowing* the trainers for the first live run. (With very complex simulations it is advisable to shadow several live runs.) Hopefully, there will be no bugs and the documentation will complete and the designer's role is to help answer any questions asked by the trainers running the simulation. His or her role is not to actually run the simulation. I was able to shadow the initial runs of both DISTRAIN and Modern Banking.

Summary
The outcomes of the Alpha and Beta Testing and the Pilot are:

Documentation Changes
Software Errors Corrections
Model Corrections
Model Changes
Model Recalibration
Timing Changes
Report and Decision Introduction Changes
Tutor/Learner Support System Enhancement

Changing the Documentation
This takes into account feedback from the tester, trainers and learners and observations during the pilot. Besides correcting typing, spelling and grammatical errors, it involves cross checking the information provided in the Learners' Manual with the simulation's data and rewording, adding to and removing from the manual to clarify and simplify. (I remember a Freudian slip with the Benson & Hedges Management Challenge where I referred in the Learners' Manual to ex-patriot managers rather than expatriate managers.)

Correcting Software Errors
These are the errors that cause the simulation to crash and should have been caught during the Alpha Test. Recently a simulation that had been used many hundred times over a twenty-one *year* period crashed because of a particular combination of decisions. (The fact that the decisions were particularly inappropriate and showed a lack of basic business knowledge is immaterial.)

Correcting the Models
This involves correcting incorrect algorithms and variable use. (Identification of these errors will be facilitated using the Reconciliations reports and special *Design Validation* reports.)

Changing the Models
This involves modifying, removing or adding algorithms. This may be necessary to provide extra information to help the trainer manage learning, adjust the way the simulation works or simplify it. As, the way a result is calculated may not be sufficiently transparent, it may be necessary to provide an addition Reconciliation Report.

Recalibrating the Model
This involves changing the response of the model to decisions and changing the economic situation over time. With Modern Banking, after the pilot the trainers felt that the simulated banks were too liquid and Capital Adequacy (solvency) issues would not be sufficiently stressed and I had to reduce liquidity.

Changing Timing
This involves checking that the amount of time allocated to the simulation and decision-making is neither too short nor too long. If there is a problem then one can either adjust the timetable or change the report and decision introduction (see next).

Changing Report and Decision Introduction
This involves delaying or advancing when these are introduced into the simulation. Changing report and decision timing many be necessary to adjust workload for individual periods and to ensure that the focus for a period is absolutely clear.

Enhancing Support Systems
This involves adding reports and explanations. The need for this becomes clear during use as the trainer running the test or the pilot observes and records questions that are being asked.

Practicalities
Build a list of things to be done as the test or pilot proceeds. Figure 8.20 shows a partial list for a recent simulation. The list includes adding help and comments, changing models and adjusting the calibration, Etc.

For the pilot, where the simulation is spread over a course, it may be possible to make some changes as the simulation progresses. But this must be done with great care.

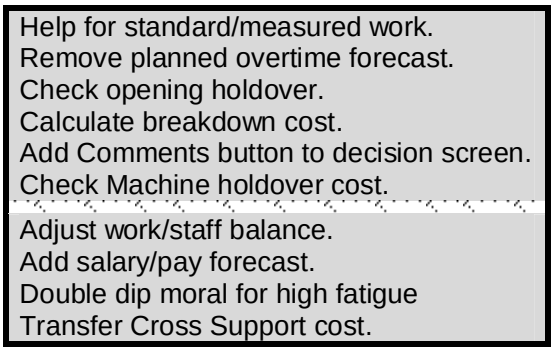


Figure 8.20: Example of to do list

Besides checking the *cognitive processing aspects* as the simulation progresses, check the *affective behaviour* of the learners.

Learning Soundness (Validity)

Ultimately the success of your design will depend on the extent to which it delivers learning and this sections explores this in terms of:

Business Impact
Perceptions – learners and trainers
Theoretical Issues

Business Impact

Arguably, all learning interventions must have an impact on the business and, in the best of all possible worlds it would be possible to measure this impact. Kirkpatrick (1998) suggests that training should be evaluated at four levels (Figure 8.21).

Level	Metric
1. Reaction	The extent to which learners enjoy the learning activity
2. Learning	The extent to which learners have acquired the learning.
3. Behaviour	The extent to which learners apply what they have learned.
4. Results	The extent to which targeted outcomes occur.

Figure 8.21: Kirkpatrick's Four Levels

But, in practice, for simulations, measuring above level 1 is very difficult. This is because of the nature of learning from business simulations and their use by adult learners. As described in Chapter 1 (Learning and Simulation) learning from a simulation as a strategic investment and as such it is difficult, perhaps impossible, to accurately access ROI. Secondly, as adult learners, each simulation participants differ in terms of perceived and actual learning needs.

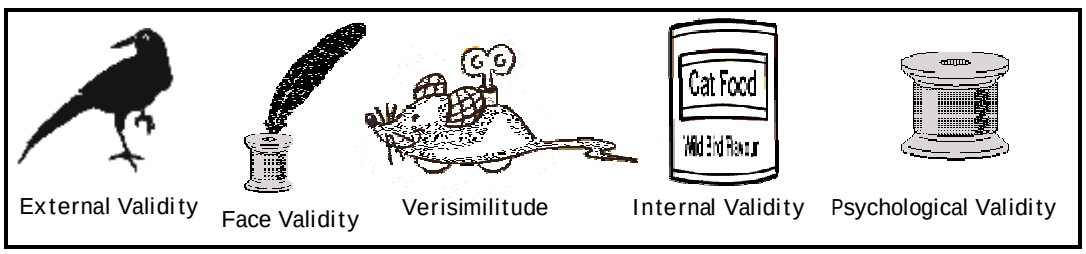
Perceptions – learners and trainers

Because of the difficulties measuring the ultimate, long-term impact of participating in a business simulation, I feel that the best evaluation of it's learning soundness is from the perceptions of learners *and* trainers directly after the event. Where the simulation is used by business people, evaluation of the course by the delegates is usual and because these are experienced and have their own agenda is a valid way of evaluating learning. Having said that, the excitement engendered by the simulation's competition can distort the response (positively and negatively). A second, source of evaluation are the views of the trainer using the simulation. Again, it is normal for the business trainer to have had significant business and management experience. Thus he can evaluate the simulations outcome appropriately.

Theoretical Issues

The academic literature on simulations discuss design soundness in terms of *validity* thus:

External Validity
Face Validity
Verisimilitude
Internal Validity
Algorithmic Validity
Psychological Fidelity



Cartoon 8.11: Validity Viewpoints

External Validity
This is *"the degree to which the game corresponds with the real life situation"* (Norris, 1986). Where one is investigating a real situation (as is the case for management science/operational research models) external validity is vital. But, the purpose of a business simulation is to develop managers and train businesspeople and so an exact replication of the real world is counterproductive because the real world is complex and a replica will not be sufficiently *focused.*

Face Validity
This "*pertains to whether the test 'looks valid' to the examinees who take it, the administrative personnel who decide to use it, and other untrained observers*" (Anastasi, 1988). Therefore it is the reality as perceived by the learners and as such it is key to their engagement. But beyond this is reality as perceived by the trainers who use the simulation (administrative personnel) and the people who sponsored development (other untrained observers).

Verisimilitude
This, like Face Validity, is *"the ability of the model to give the appearance of reality"* (Barton, 1980). Feinstein and Cannon (2001) call this fidelity *("the level of realism that a simulation presents to the learner"*). Therefore it is the reality as perceived by the learners and as such it is key to their engagement.

Internal Validity
This is the extent to which cause and effect are truly linked and hence validates the links between decisions and results. However, as discussed in previous chapters, for learning to occur there needs to be some ambiguity and, it is often desirable to adjust the internal validity to ensure engagement and learning. Thus, with DISTRAIN, it is possible to be significantly more profitable than in the real world. For Product Launch, price cuts have an immediate impact whereas in actuality, price cuts would have a delayed impact as the knowledge of the cut spreads through the market – a distortion to enable learners to identify the impact of their price decision and hence ensure learning.

Algorithmic Validity
This is the extent to which *"the algorithm used in the simulation really model the phenomena it is supposed to represent"* (Feinstein and Cannon, 2001). Initially, like motherhood and apple pie, this would seem to be a *good thing.* But, there are considerations in terms of simplification, stylisation, limited duration, perceptions and the extent to which invalid algorithms affect learning. For example, the simulation with the accounting errors (described previously) was in use for more than a decade and used by literately thousands of students, yet (apparently) it did not reduce learning.

Psychological Fidelity
This *"concerns the extent to which the training environment prompts the essential underlying psychological processes relevant to key performance in the real-world setting"* (Kozlowski and DeShon, 2004). As such it is vital as it ensures that the learners exercise the thinking that is necessary for them to develop business wisdom.

Summary
Soundness and validity must to based on *purpose.* I started my modelling career building models for business *planning* and operations *research* where the purposes were to be able for forecast and budget accurately and to accurately represent the situation modelled. For operations research simulations *external, internal* and *algorithmic validity* are crucial. But, simulations for business learning are very different and *psychological validity* and *verisimilitude* are crucial. In other words, the cognitive purpose is to get the learners to think through the management of their simulated business and through this deep thought develop business understanding and wisdom – develop the meta-skills that enable the learners to handle the real world. And, the affective purpose is for the learners

to believe that the business situation that they are managing *appears* to be real and hence be engaged by it.

It is likely that it is easier to design a simulation that has good algorithmic validity than it is to design one that has good psychological validity that provides good learning.