

THE **e**LEARNING DEVELOPERS' JOURNAL™

Strategies and Techniques for Designers, Developers, and Managers of eLearning

JOURNAL™

THIS WEEK — DEVELOPMENT TECHNIQUES

Developing e-Learning Simulations With Tools You Already Know

BY MIKE SMIALEK

Study after study has shown that learning through simulation is one of the best ways to learn and master new skills. Airlines use flight simulators to give pilots hands-on experience; NASA uses simulators to prepare astronauts and ground control for missions; and successful companies like GE use simulation to teach financial management skills to employees. Why, then, isn't simulation used more pervasively throughout corporate and academic education?

The reason is **cost**. Good simulations are notoriously expensive to build. This makes sense because of the complexity involved. If you are just trying to teach a few facts or a simple decision process, then simulation is certainly overkill. But if you need to impart an accurate mental model of a sophisticated system or process, you want simulation. By definition these situations are inherently complex. (See Sidebar 1 on page 2, "The Importance of Mental Models.")

Typically a simulation must let the learner make multiple, interrelated decisions over multiple periods of time, resulting in thousands or millions of possible learner paths and distinct outcomes. These simulation models must be accurate and realis-

tic, even under extreme states, to keep the learner from developing an inaccurate mental model.

Furthermore, sophisticated decisioning simulations require rich and complete feedback capability to keep the learner from getting stuck. The feedback must consider all user decisions, outcomes, and key performance indicators, in order to provide useful support and guidance to the learner.

The development of complex simulations with rich feedback can be daunting. The sheer volume of code required to handle so many combinations of user states and feedback can be immense. But with the right tools you can not only dramatically

Continued on next page

Simulations can be very expensive to build due to the time it takes using traditional e-Learning tools (not to mention the learning curve required). This article will introduce you to capabilities of a tool that you probably already use — Excel — that is also an excellent simulation development tool. Here's a step-by-step process for quickly and easily creating rich simulations for a fraction of the cost you'd expect.

A publication of



SIDEBAR 1: The Importance of Mental Models

If you see a rock hurtling toward your head, you duck. This is because as a young child you developed a mental model of the causes and effects of object collisions. You learned that objects in motion tend to stay in motion, and that objects colliding with your head can cause pain or death. You use this mental model of direct cause and effect to avoid the undesirable outcome of being hit in the head. You recognize that an ambient condition is about to cause you harm, and you use the variables that you have control over (in this case your head position) to affect the outcome.

As we grow into adults, most of the cause and effect relationships we must deal with in life and on the job are not as direct. Managing for desirable outcomes involves multiple, interacting variables, often with degrees of uncertainty and variability. But the utility of an accurate mental model is exactly the same as in the collision avoidance example: it allows you to take actions that result in desirable outcomes.

With an accurate mental model, you can understand the conditions over which you have no control (ambient conditions), and synthesize an action plan that is likely to result in desirable outcomes because you can predict how your actions will influence the outcomes. For a pilot, this might mean successfully recovering from an engine fire. In a complex business environment, this might involve entering a new market, anticipating competitor response, or hedging against potential economic risks.

Simulation lets the learner develop a mental model in less time and with less risk than it would take to develop the mental model through on-the-job training.

reduce development cost, you can even make development downright fun. This article will take you step-by-step through the process of creating a simulation, and teach you how to do it with a minimal learning curve using tools you're probably already familiar with. (See Sidebar 2, "Where Simulation Fits.")

Overview

The intent of this article is to demonstrate one way to efficiently develop a simulation. For the purpose of clarity, we will

use a reference simulation (described later) as a basis for the examples.

I will touch briefly on design, but focus mainly on implementation. Efficient simulation development revolves around the ability to isolate the main feature areas of the application, separating them into three tasks: Creation of the simulation model, development and testing of support and guidance features, and finally development of the user interface.

In explaining these areas, I will delve into a subset of the reference simulation, going into more detail and providing full source code.

Reference simulation: "BreakEven"

For the purpose of providing context, we will use an existing simulation as a reference. (See Figure 1 on page 3.) BreakEven is available online at <http://www.knowledgedynamics.com/demos/Breakeven/index.htm>. This simulation teaches the basics of a break-even analysis by putting the learner in the role of a senior finance professional. The learner must plan the launch of a new product line, making key decisions in the manufacturing and marketing of the new product. The learner must decide what price to charge for each unit, and how much marketing

investment to make. The learner must also make several production decisions. The learner's business goal is to break even within two years. In pursuing this goal, the learner can rely on feedback from three personalities: the CFO, the VP of Manufacturing, and the VP of Marketing. BreakEven is relatively simple by simulation standards, but is still sophisticated.

The target audience is new finance professionals. The simulation will help them make better financial decisions and it will give them insight into the due diligence required in evaluating business development opportunities.

If you're not familiar with the concept of Break Even Analysis that's OK, because you can learn it by playing the simulation! The gist of breaking even is that it's not enough to just sell items for more than they cost to produce, you must also recover the initial investment that was made in the production facilities.

From a development perspective, the important observations to make about the BreakEven simulation are:

- The learner makes multiple, interrelated decisions.
- All decisions flow all the way through the simulation to the outcomes.
- The 'break even formula' and 'cost classification' decisions are right or wrong individually, but the 'production decisions' must all be considered together to determine success or failure.
- The outcomes and Key Performance Indicators (KPIs) react immediately to changes by the learner.
- The feedback from the intelligent personalities (CFO, VP of Manufacturing, and VP of Marketing) adjusts to the learner's performance and path through the simulation, so each learner has a unique experience.

No installation or special software is required to run BreakEven. It runs in browsers with Java 1.1 or higher and JavaScript. In addition, if the learner is unfamiliar with the breakeven formula, or with the concepts of cost classification and KPI's, these are explained in the "references" pulldown menu in the simulation.

A note about design

No matter how good your simulation technology is, you cannot succeed without good design. I will not go into simulation design in great depth, because it is subjective and the process can be very different for different topics, but here is a summary of the main elements.

SIDEBAR 2: Where Simulation Fits

Simulation is best used when you need to impart to the learner an accurate mental model of a sophisticated system or process. In these cases only practice will help the learner become proficient.

But it is important to note that in most cases, simulations cannot stand on their own. Simulations are tools that let learners *practice new skills*. But simulation is usually not the best medium for *introducing new skills*. Most simulations require an introductory presentation of materials that orients the learner to the concepts and to the simulation story. Simulations also require reference materials that the learner can access for support while taking the simulation. The introductory materials and reference materials are often very much like traditional CBT, consisting of Flash movies, video clips, and web pages.

The business goal of training. In my opinion, *the only goal of corporate training is to change the behavior of employees in a way that improves business performance.* All simulation design should be governed by specific performance improvement goals. In the case of BreakEven, the business performance goal is to improve financial decision-making by giving new employees a deep understanding of break-even analysis.

Concepts. The first step in designing a simulation is to identify the comprehension and skill you want to convey to the learner. In BreakEven, the basic concepts are the break-even formula and classification of costs as fixed or variable. The primary skill we are trying to impart to the learner is the ability to perform a break-even analysis. We want the learner to internalize an accurate mental model of the interrelation between fixed costs, variable costs, sales volumes, and profitability.

Decisions. The next step in designing a simulation is to identify decisions and activities that allow the learner to practice the skills and processes that are germane to the concepts. In BreakEven, the learner must fill in the break-even formula, classify costs as fixed or variable, and choose production methods and pricing parameters.

Background story. The concepts and decisions must be wrapped in a story that engages the learner and provides job-related context for the learning. The learner should be given a goal, and resources and constraints with which to achieve the goal. Although some designers contend that the story and simulation should mimic the real world as closely as possible, we have found that in most cases it's more effective to use an approximation of reality that isolates and emphasizes the key learnings. In BreakEven, the learner is put into a role with much more responsibility than a new finance professional would likely be given. The decisions are simplified to illustrate the underlying financial principles.

Common mistakes. It is important that the simulation lets learners make mistakes and learn from them. This is especially true for more common mistakes made by beginners. In BreakEven, the learner can devise a plan that appears to be successful, but ultimately fails because there is not enough Production Capacity to meet Projected Demand.

Simulation interaction styles. There are several different styles of simulation. Depending on your subject matter, you may have several options:

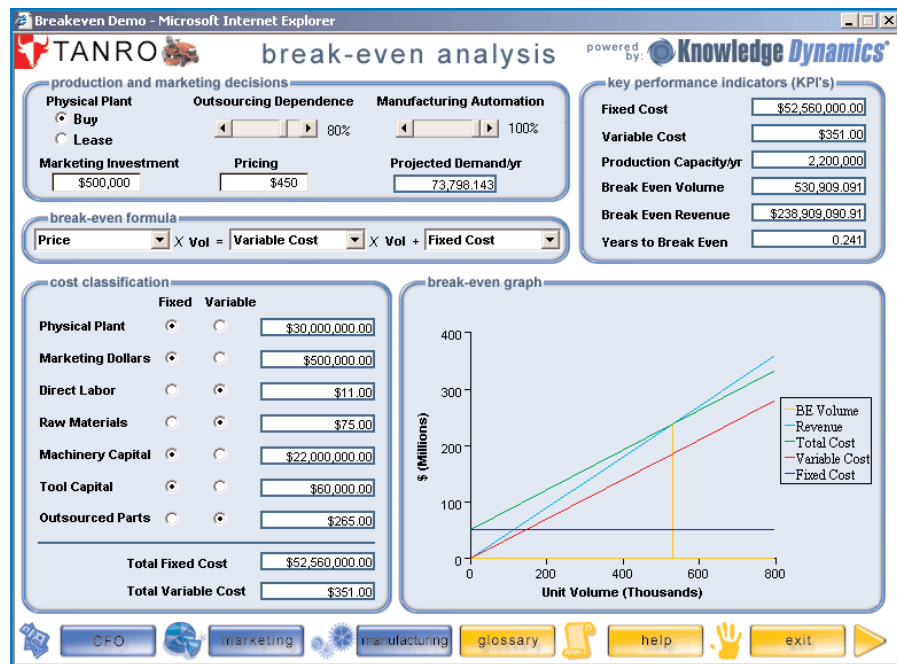


FIGURE 1 The BreakEven simulation places the learner in the role of a senior finance professional.

- **Artifact Based** — BreakEven is an artifact-based simulation. The learner must build or craft a deliverable, iterating over it and improving it until it meets some acceptance criteria. In BreakEven the learner's key deliverable is a product manufacturing and marketing plan that is financially sound.

- **Time Based or Management** — In a time-based simulation, the learner has to make decisions over multiple periods of time. The learner makes several decisions and then advances time on a turn-by-turn basis.

- **Conversation** — These simulations mimic a conversation between the learner and one or more simulated agents. In these simulations, the learner must decide what question to ask next or how to respond to questions from the simulated agents. They are commonly used to teach soft skills such as how to negotiate, sell products, or handle customer service.

- **Continuous Simulation** — Continuous simulations are usually used to teach someone how to manage a physical system, for example how to control the cooling system of a nuclear power plant. SimCity is a continuous simulation.

- **Sensory Immersive** — These simulations teach not just decision skills, but also psycho-motor skills. Commercial airline pilots and some military personnel train with sensory immersive simulations. These simulations can be so accurate that

participants forget they are in a simulation, giving them a taste of some of the stresses they may face in the real work environment.

- **Multi-player** — Any of the foregoing simulation styles can be used in a multi-player simulation. Multi-player simulations are usually much more expensive to build because they require a server-side application, database, and communication and synchronization support.

Although this article specifically discusses Artifact-Based decision simulation, all of the design and development techniques are equally well suited to the other interaction styles.

Medium. Lastly, a designer must take into account the delivery medium for the simulation. Delivering simulations to dial-up users will have different design criteria than a simulation for broadband or CD-ROM distribution. Factors such as computer speed and operating system differences might affect the use of sound, video, animation, and graphics.

The development of the BreakEven simulation

From years of experience developing simulations, we have found that developing them with the traditional eLearning toolset (i.e. Flash, DreamWeaver, Authorware, etc) is prohibitively expensive. Out of necessity, we have had to evolve methods that reduce development cost and risk. Efficient

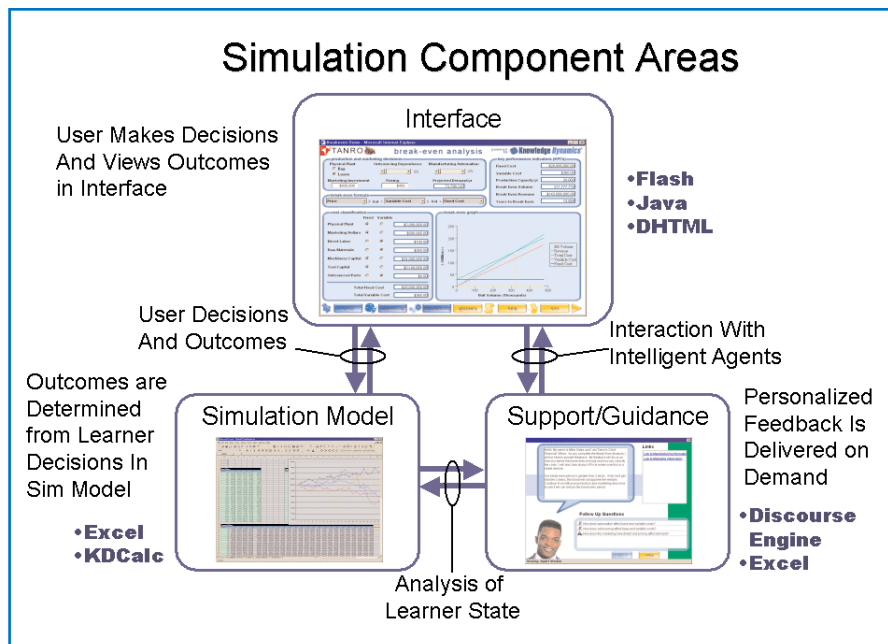


FIGURE 2 Simulation component areas provide the foundation for efficient development.

simulation development is easier when the tasks are broken into three distinct areas:

- **User Interface** — The learner is presented with User Interface (UI) elements for making decisions, seeing the resulting outcomes and navigating and accessing the features of the simulation.
- **Underlying Simulation Model** — The simulation model receives the learner's inputs from the UI and generates the resulting outcomes for display in the UI.
- **Remediation** — The feedback analyzes the learner's decisions, outcomes, and KPIs, to deliver tailored feedback and guidance.

This breakdown extends to the development tasks and to the application components. (See Figure 2.) Each of these three areas communicates with the others only in well-defined ways, allowing them to be developed independently and in parallel. This borrows from a software development technique known as *component-oriented development*.

The simulation model

After the design is completed, the first of the three areas to focus on is the underlying simulation model. The first step in building the simulation model is to list all of the decisions that the learner will have to make and all of the outcomes that will result. For BreakEven these are:

Learner input decisions:

- Fill out the three parts of the Break Even Formula

- Classify each production cost as Fixed or Variable
- Determine whether to Buy or Lease the Production Facility
- Determine how much to charge for each unit (Pricing)
- Determine how much to invest in Marketing
- Determine how much to rely on Outsourcing
- Determine how much to rely on Automated Machinery

Outcomes that result from learner input decisions:

- Projected Demand
- Production Capacity
- Total Fixed Cost
- Total Variable Cost
- Break Even Revenue
- Break Even Volume
- Break Even Date

The next step is implementing the simulation model that turns the learner's decisions into a set of outcomes. After much experimentation, we have found that the easiest way, by far, to develop simulation models is with Microsoft Excel. We have found developing with Excel to be about 20 times more efficient than hand-coding simulation logic in Java, JavaScript, Flash, or some other development tool.

What about delivery?

If you are delivering your simulation to clients that are sure to have Excel

installed, then one option for delivery is to build the rest of your application in Visual Basic (VB) or Visual Basic for Applications (VBA). VB is the most popular development tool for building Windows applications, and it integrates well with Excel using Microsoft's Component Object Model (COM) technology. VBA, on the other hand, is built into Excel and it enables you to ship the whole simulation as a single Excel file. They both use similar syntax, they are both well documented, and skilled developers are plentiful.

But you can't always count on integrating with Excel in your end-users' environment. So we used a product called KDCalc, which is an Excel-compatible spreadsheet engine for Java and .Net. This approach allows you to develop simulations for web-based as well as stand-alone delivery. (Editor's Note: KDCalc is one of many spreadsheet engines available that perform these functions; you may wish to search for them using your favorite Internet search engine and the search string "spreadsheet engine.")

Projected Demand mini simulation

To simplify the discussion of how simulations are developed in Excel, we will focus only on a subset of the BreakEven simulation. I have also provided a smaller Projected Demand mini simulation with full source code. It can be viewed and downloaded at <http://www.knowledgedynamics.com/demos/ELDJ/index.htm>. (See Figure 3 on page 5.) You may find it helpful to try out this mini-simulation before proceeding.

In BreakEven, the projected demand is calculated directly from the learner's price and marketing investment decisions. The mini-simulation focuses specifically on this relationship. To model the relationship in Excel we identify one cell for each of these variables. We use Excel's naming feature to name these cells 'Learner_Price_Value', 'Learner_MktInv_Value', and 'Projected_Demand'. (See Figure 4 on page 5 for how the model looks in Excel). The named cells and ranges are indicated in the callouts in the figure. Color coding has been added to improve readability. (See Sidebar 3 "Using Excel's Naming Feature" on page 6 for more information if you are not familiar with the use of named ranges).

Although the spreadsheet in the figure appears complicated at first glance, it is actually quite simple. Only three of the cells have formulas. The rest of the cells just hold numbers and text for documenta-

tion. You can download the spreadsheet and see it embedded in a DHTML page at: <http://www.knowledgedynamics.com/demos/ELDJ/index.htm>

The learner input for Price is a continuous value (ie the learner can type in any number that she wants). But for the BreakEven simulation, we want to group the values into discrete buckets or levels. This will facilitate both computation and selection of feedback text. To accomplish this we identified five different Pricing Levels as part of the simulated marketing research data. The Pricing Levels are:

1. Very Low
2. Low
3. Medium
4. High
5. Very High

Now we use Excel's built-in 'LOOKUP' function to translate the continuous Price value into a discrete Price Level. We do this in a cell named 'Learner_Price_Level'. Its formula is:

```
Learner_Price_Level =
LOOKUP(Learner_Price_Value,
Normalized_Price_Values,
Normalized_Price_Levels)
```

In English, the cell formula says "look up the Learner_Price_Value in the range named Normalized_Price_Values, and return the corresponding Level from the range named Normalized_Price_Levels." (Normalized pricing information is also part of the simulated marketing research data.)

So, a Learner_Price_Value of \$425 corresponds to a Learner_Price_Level of 3. With the standard behavior of the 'LOOKUP' function, all prices greater than or equal to \$400 and less than \$430 correspond to a Learner_Price_Level of 3. Likewise, all prices greater than or equal to \$430 and less than \$500 correspond to a

Learner_Price_Level of 4. The great thing about this is that you can have as many levels as you want, and it is very easy to adjust the thresholds by simply typing in different numbers.

We use the same method to calculate the Learner_MktInv_Level (for the learner input "Marketing Investment"). A Learner_MktInv_Value of \$700,000 corresponds to a Learner_MktInv_Level of 4.

Now that we have the two Levels, we can use them to determine the Projected Demand. To do this we use Excel's built-in 'OFFSET' function. The cell formula for the 'Projected_Demand' cell is:

```
Projected_Demand =
OFFSET(Projected_Demand_Matrix,
Learner_Price_Level,
Learner_MktInv_Level)
```

With a Learner_Price_Level of 3 and a

Learner_MktInv_Level of 4, this returns the value in the 3rd row, 4th column of the Projected_Demand_Matrix. This is 253,750 units.

In other words, this cell formula says "Start at the top-left corner of the Projected_Demand_Matrix and offset down Learner_Price_Level rows and over Learner_MktInv_Level columns.

Again, you can have as many rows and columns in the matrix as you like. You can tweak the Level thresholds and Matrix values until they give realistic, desirable behavior. This is a particular *design pattern* in Excel. You use it when you need to derive one value (the output) as a function of two other values (the inputs) and the relationship is *nonlinear* (there is not a mathematical relationship between the inputs). If the relationship between the

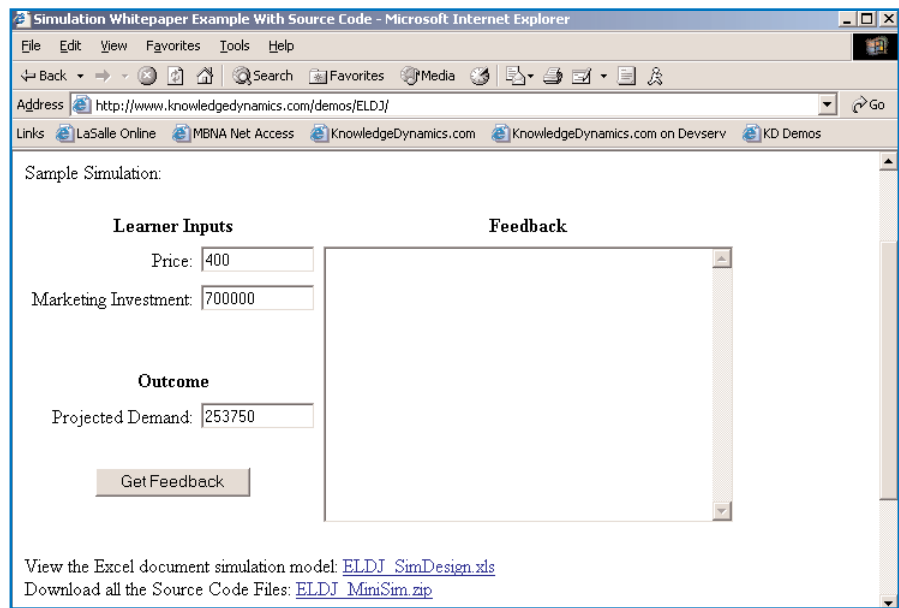


FIGURE 3 The Projected Demand mini-simulation is a subset of BreakEven.

Calculating Projected Demand from Simulated Marketing Research Data

Price	Value	Learner_Price_Level
3	400	

Learner_Price_Level = LOOKUP(Learner_Price_Value, Normalized_Price_Values, Normalized_Price_Levels)

Marketing Investment	Value	Learner_MktInv_Level
4	700,000	

Learner_MktInv_Level = LOOKUP(Learner_MktInv_Value, Normalized_MktInv_Values, Normalized_MktInv_Levels)

Projected Demand	Value
	253750

Projected_Demand = OFFSET(ProjectedDemandMatrix, Learner_Price_Level, Learner_MktInv_Level)

Price Normalizer	Level	Price
1	100	
2	300	
3	400	
4	430	
5	500	

Normalized_Price_Values
Normalized_Price_Levels

Marketing Normalizer	Level	Investment
1	1	
2	400,000	
3	500,000	
4	700,000	
5	1,000,000	

Normalized_MktInv_Values
Normalized_MktInv_Levels

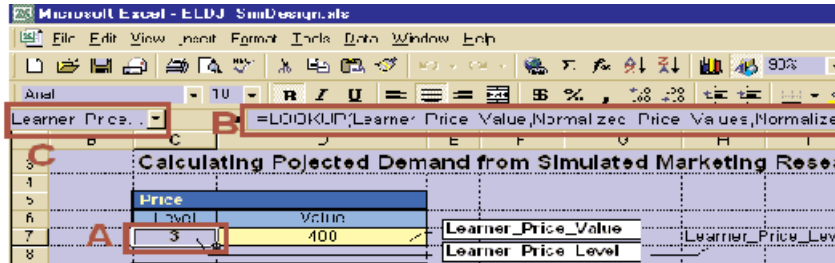
Projected Demand Matrix	Price	Marketing Investment
1	1	175,000
1	2	437,500
1	3	546,875
1	4	656,250
1	5	875,000
2	1	43,750
2	2	109,375
2	3	262,500
2	4	328,125
2	5	437,500
3	1	32,156
3	2	87,500
3	3	218,750
3	4	253,750
3	5	293,125
4	1	21,875
4	2	43,750
4	3	75,264
4	4	109,000
4	5	132,157
5	1	10,937
5	2	21,875
5	3	27,084
5	4	30,137
5	5	32,650

Projected_Demand_Matrix

FIGURE 4 This spreadsheet model calculates ProjectedDemand in BreakEven

SIDEBAR 3: Using Excel's Naming Feature

One of Excel's most valuable features is the ability to name individual cells or ranges of cells. This allows you to define cell formulas in terms of names rather than the less readable row-column format. In the figure below, cell C7 is highlighted (A). The cell's formula appears in the Formula Bar (B), and its name appears in the Name Box (C).



You can see that the cell formula is defined in terms of other named cells and ranges.

To name a cell, click on the cell in Excel, then type the name in the Name Box (C). Be sure to hit <Enter> or the Name will not be saved. You can name ranges of cells the same way — just drag-select a range of cells and type the name in the Name Box. To edit or delete the name, use the 'Insert | Name | Define...' menu in the menu bar.

inputs and outputs is linear, you can use a single cell formula.

You can easily extend this design pattern. You could replace the raw numbers in the Projected Demand Matrix with cell formulas, to simulate changing economic conditions. There are several such design patterns that are useful for building simulations in Excel. Simulation models are just aggregations of a few basic design patterns.

Building and testing the Projected-Demand calculation takes a non-programmer only a few minutes in Excel. Contrast that with the cost and effort of hand-coding it in something like Java or ActionScript. And how would you test it? In a programming language you would have to build a user interface before you could test it at all, and that effort would be a throwaway. In Excel you can test it right away.

This Excel-centric approach provides a lot of other benefits as well. It enables Subject Matter Experts (SMEs) to participate extensively in model development because many SMEs are already familiar with Excel. Another big plus is that it eliminates a layer of communication between the SMEs and the developers. You can build and test a simulation iteratively, without needing to have the whole thing built before you can start to test. You can also implement generic, automated testing in Excel (a topic for another article).

Excel tips

Most business people are familiar with Excel and have no trouble building spread-

sheets that perform mathematical operations. There are some Excel functions that might not be part of your vocabulary, but are essential for developing simulations. These are: IF, OR, AND, LOOKUP, OFFSET, COUNTIF, SUMIF. It is worth taking a few minutes to experiment with these functions if you're not familiar with them. You'll find that they significantly extend the power of Excel and make it easy and fun to use Excel for developing simulations. In fact, the whole of BreakEven is built using only these functions plus the basic math operations (addition, multiplication, etc).

As you can see, leveraging Excel as a simulation development tool dramatically reduces the effort and skill level required to develop sophisticated simulations. But more importantly, by building on a few basic patterns in Excel, you can build incredibly dynamic simulations, limited only by your own creativity.

Remediation

Feedback and guidance are absolutely essential for an effective simulation. Learners are more likely to have difficulties and need help in a simulation than in other forms of eLearning, because simulation is more complex. In some simulations it may

be the case that humans provide the feedback, but this is hardly cost effective for broad application in the business world — so automated feedback is required.

In traditional Web-Based Training (WBT), the most common exercise interface is a multiple-choice quiz: A question, followed by three to five possible responses. Here, the automated feedback usually consists of a sentence for each response indicating why it is right or wrong. This works in a quiz format where the questions are each independent, but it fails in a simulation, where the learner's responses are interdependent and must be diagnosed together.

The BreakEven simulation uses a proprietary technology called the Discourse Engine to assess the learner's performance and generate feedback tailored to each learner's needs. The Discourse Engine allows you to create multiple feedback personalities and teach them about the simulation model. They can each react differently to the learner's actions and change their mood over time. But even without the Discourse Engine, it is possible to create feedback that is almost as rich, using just the basic features of Excel. Cell formulas with built-in Excel functions can analyze the learner's decisions and outcomes to dynamically generate feedback. *(Editor's Note: As the author says, feedback can be created using Excel alone, without Discourse Engine. If you are considering the value of tailored feedback, you can locate additional proprietary development resources with an internet search on the string "rules based AI engine".)*

Figure 5 shows a template for making a feedback rule in Excel. Part (A) is the Feedback Rule Label. This is purely for documentation so you can identify the rule at a glance. We've seen cell labels before in the simulation model. They are used as visual markers in the spreadsheet, but they are not used in the calculations. Part (B) is the Feedback Text. This may be plain text, or it may be a cell formula that calculates the text from the learner's decisions. Part (C) is the Feedback Rule. It can be any cell formula that diagnoses strengths and weaknesses in the learner's decisions.

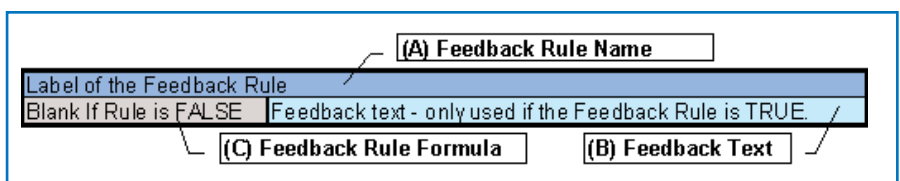


FIGURE 5 A feedback rule template in Excel can dynamically generate feedback text.

Figure 6 shows all the rules in the simple example. Let's examine the 'Price is too high' feedback. The cell formula for the Feedback Text is:

```
= "Your price of $" & Learner_Price_Value & " is a little high. I think ..."
```

This formula dynamically includes the learner's Price in the feedback. Now look at the Feedback Rule Formula. Its cell formula is:

```
= IF(Learner_Price_Level>4, E42, "")
```

This says "If the Learner_Price_Level is more than 4, then return the Feedback Text from cell E42, otherwise, return an empty string." So whenever the learner sets the Price at \$500 or above, this feedback will be delivered.

Note that there are three feedback rules that look at the learner's decisions. These are 'Price too low,' 'Price too high,' and 'Marketing Investment too low.' There is also a rule that looks at the outcome: 'Projected Demand too low.' In practice, there can be any number of Feedback Rules that examine any combination of learner decisions and outcomes to deliver targeted feedback to the learner.

The 'Looks Good' feedback rule praises the learner for developing a good marketing plan. Its cell formula is:

```
=IF( AND(Learner_Price_Level>2, Learner_Price_Level<5, Learner_MktInv_Level>2, Projected_Demand>110000) ,E51, "")
```

This says "If the Price Level is more than 2 and less than 5, and the Marketing Level is more than 2, and the Projected Demand is more than 110,000 units, then deliver the 'Looks Good' feedback text, otherwise don't."

The last feedback rule is the 'Nothing Interesting' rule. Just in case no other feedback rule delivers feedback, you want to make sure you at least say something to the learner. In this case the cell formula is:

```
=IF(AND(D39="", D42="", D45="", D48="", D51=""), E54, "")
```

This says "If all of the other feedback rules delivered no feedback, then deliver the 'Nothing Interesting' feedback text from cell E54, otherwise don't".

Finally, you need some way to put all the feedback together. At the top of Figure 6 is the 'Total Feedback' cell. This cell is named 'Feedback' and this is where the UI will look when the learner asks for feedback. Its cell formula is:

```
=D39 & D42 & D45 & D48 & D51 & D54
```

It simply appends all the available feedback from all the feedback rules. In this small example it is OK to just append all the feedback together, because you can anticipate every possible combination of the feedback. In practice you will usually want some additional cells for determining which feedback rules to use if many are available.

So let's look at some interesting feedback. If the learner enters a Price of \$350 and a MarketingInvestment of \$700,000, this results in a ProjectedDemand of 328,125 units and the following feedback:

"Your price of \$350 is kind of low. Although you will sell a lot of mowers I think you will have trouble making a profit with such a low price."

If the learner enters a Price of \$500 and a MarketingInvestment of \$400,000, this results in a ProjectedDemand of 21,875 units and the following feedback:

"Your price of \$500 is a little high. I think you could drive higher demand with a lower price. Your projected demand is not very high. Try adjusting your Price and Marketing Investment to drive higher demand."

Note that this feedback includes text from two feedback rules. With some more creative cell formulas, you can make your simulation deliver a comprehensive learner assessment consisting of multiple different pieces of text. You can also make intelligent use of conjunctions and do other linguistic tricks. For example, it is not too difficult to set up the second piece of text to say "Also, your projected demand is not very high..." when it follows another piece of text.

So, you can see that developing rich, dynamic learner assessment and feedback in Excel is not much different from developing the simulation model.

Calculating Feedback in a Spreadsheet	
Total Feedback:	
Your marketing plan looks good. \$400 is a reasonable price, and with a \$700000 marketing budget we will achieve great market penetration.	
Feedback Rules:	
Label of the Feedback Rule	Feedback Color Key Template
Blank if Rule is FALSE	Feedback text - only used if the Feedback Rule is TRUE.
Price is too low	Your price of \$400 is kind of low. Although you will sell a lot of mowers I think you will have trouble making a profit with such a low price.
Price is too high	Your price of \$400 is a little high. I think you could drive higher demand with a lower price.
Marketing Investment Too Low	\$700000 is not a substantial marketing investment. You need to get the word out if you want to sell enough mowers to break even.
Projected Demand too low	Your projected demand is not very high. Try adjusting your Price and Marketing Investment to drive higher demand.
Looks Good	Your marketing plan looks good. \$400 is a reasonable price, and with a \$700000 marketing budget we will achieve great market penetration.
Nothing interesting to say (no other feedback rules from above fired, but I should still say something)	
I don't have anything interesting to tell you.	

FIGURE 6 Feedback in Excel may be calculated according to rules in the spreadsheet.

SIDEBAR 4: Crash Course in Event Driven Programming

Way back in the good ol' days (the early 80's, before the Macintosh debuted), most programs were single-purpose batch programs. They would load a file, do something to it, write the results to another file, and then end. These programs were started with a *command line* that indicated which file to load and which file to write the results to. The programs were written in a style called *procedural programming*, in which the whole program consists of a series of steps executed in order.

With the dawn of Graphical User Interfaces came the need for *event driven programming*. When a user of an application interacts with a UI, something called *events* are generated. For example, when you click a button on a UI in Windows it generates three events: 'MouseDown', 'MouseUp, and 'ButtonClicked'. Even when you are doing nothing, 'IDLE' events are being generated. In fact, at any given moment in Windows or MacOS, hundred of events are fired every second. A programmer can write *handler* functions to handle any or all of these events. The handler functions are each still procedural, but they are triggered in response to events, allowing the user to access the program's features in any order. So when you click the 'Save' button in Excel, it calls the 'OnSave' handler function, which saves your spreadsheet to disk.

SIDEBAR 5: User Interface Code

The general pattern in pseudo code is:

```
SomeEventHandler()
{
    Get_All_The_User_Inputs_From_The_UI()
    Pass_Them_Into_The_Simulation_Model()
    Pull_All_The_Results_Out_Of_The_Simulation_Model()
    Update_The_UI_With_The_New_Results()
}
```

This pattern holds true whether you are developing a UI in DHTML, Flash, or Java. In the ProjectedDemand mini sim, the event handler that triggers the process is 'OnBlur'. When the user changes an input value and then clicks outside of the input control, the input control *loses focus*, hence the name OnBlur. The syntax for doing this in DHTML/JavaScript follows:

```
. . .
/**
 * Populates the interface with values from the simulation model.
 */
function populateInterface()
{
    // This function not only refreshes the learner's outcomes, but the input
    // decisions as well. This is because when the web page
    // is first loaded, the default decision values can come from the spreadsheet.
    // Notice that the values are pulled from the spreadsheet by Name.
    priceTxt.value = KDCalcEngine.getNumberFromNamedCell("Learner_Price_Value");
    marketTxt.value = KDCalcEngine.getNumberFromNamedCell("Learner_MktInv_Value");
    projDemandTxt.value = KDCalcEngine.getNumberFromNamedCell("Projected_Demand");
}

/**
 * Called from onBlur of the input text boxes. Pushes changed values and
 * re-populates the interface.
 */
function doRecalc(component)
{
    // Push the learner's decisions into the simulation model
    KDCalcEngine.setNumberInNamedCell("Learner_Price_Value", priceTxt.value);
    KDCalcEngine.setNumberInNamedCell("Learner_MktInv_Value", marketTxt.value);

    // Call the populateInterface function to refresh the values in the UI
    populateInterface();
}
. . .
```

Full source code for the ProjectedDemand mini sim is at
<http://www.knowledgedynamics.com/demos/ELDJ/index.htm>.

User interface

After the underlying Simulation Model and Remediation features have been built, the core functionality is essentially done. The next step is to wrap the simulation with a user interface. With the Excel-based approach used for the Simulation Model and the Feedback, the only place where it's actually necessary to write code is in developing the user interface (UI).

Fortunately, there are many capable tools available for building UIs in web-deployable simulations. Some of our favorites are Flash, DreamWeaver, and HomeSite from Macromedia, Visual Café from TogetherSoft, and Visual Studio from Microsoft.

The development of simulation UIs is similar to traditional UI development, so I will only focus on aspects that are unique to simulations.

A simulation user interface should:

1. accept decisions from the learner,
2. pass them into the simulation model,
3. retrieve results/outcomes from the simulation model,
4. display them to the learner,
5. allow the learner to navigate through the simulation and advance time, where applicable, and
6. display feedback to the learner, where applicable.

Different kinds of simulations will do the above things differently. In BreakEven, the outcomes are recalculated instantly whenever the learner makes a change to any decision. So, when the learner slides the ManufacturingAutomation slider in the BreakEven UI, the graph and the numerical outcomes are all recalculated and redisplayed. Other simulations will have the learner make several decisions, but not update the outcomes until the user pushes a 'Submit' button or triggers some other event. Yet other simulations recalculate continuously, so the speed with which the learner makes decisions is a factor.

If you've never programmed before, see Sidebar 4 on page 7, "Crash Course in Event Driven Programming."

In BreakEven, every UI control that takes a user input has an "OnChange" event handler. Whenever the user changes a value in any of the input controls the OnChange event handler triggers a process that gets all the user inputs from the UI, passes them into the model, gets the results out of the model, and updates the UI with the new results. The syntax for doing this in DHTML/JavaScript is shown in Sidebar 5, "User Interface Code."

Now you should take note of another important benefit of the Excel-based approach: The person(s) developing the UI can work independently of and in parallel with the person(s) developing the model and feedback. This is possible because the UI communicates with the simulation model only via the four named cells in the spreadsheet: Learner_Price_Value, Learner_MktInv_Value, Projected_Demand, and Feedback. Of course these are the inputs and outputs of the simulation model, and the feedback. As long as this list of inputs and outputs of the model doesn't change, then the Excel parts remain independent of the UI parts. The spreadsheet can be updated and enhanced without having to change the UI, and vice versa.

So you can see that simulation UI development is just like any other UI development. Short, simple code is used to capture the learner's input decisions, pass them into the model, and display the resulting outcomes. The heavy lifting is done in the underlying simulation model that was easily crafted in Excel.


Summary

Hopefully it is clear that good eLearning simulations don't have to break the bank. By splitting the development into the three areas of simulation model, feedback, and UI, you can develop simulations systematically, using the best resources and the most appropriate tools for each part.

By leveraging Excel as your simulation development environment you can:

- Dramatically reduce the effort and skill level required to create simulations,
- Directly involve SMEs in the simulation development process,
- Eliminate a layer of communication between SMEs and developers,
- Integrate high-quality learner assessment and feedback,
- De-couple the simulation model from the User Interface to parallelize processes, and
- Employ automated testing procedures in Excel.

Finally, you can leverage widely available tools and general-purpose development techniques to package the simulations in media-rich interfaces for web-based or stand-alone deployment.

(Editor's note: Readers who know of additional spreadsheet engines, or artificial intelligence toolkits which they have successfully used in creating simulations may add them to the Guild Resources.) 

AUTHOR CONTACT

Mike Smialek is Co-Founder and CEO of Knowledge Dynamics, Inc., a provider of simulation technologies and custom services to the corporate and academic markets. Mr. Smialek has over 10 years of experience in artificial intelligence and simulation technologies. Prior to founding Knowledge Dynamics, he was an executive of Accenture's eLearning Simulation practice, helping develop custom simulations for global clients such as General Electric and Pratt & Whitney. Technology developed by Mr. Smialek has been used on over \$100M worth of custom simulation engagements for Accenture clients. Mike can be reached at MikeS@KnowledgeDynamics.com.

ONLINE DISCUSSIONS

Extend your learning beyond the printed page! If you are looking for more information on this topic, if you have questions about an article, or if you disagree with a viewpoint stated in this article, then join the online discussions and extend your learning.

Follow these easy steps to participate:

1. Go to www.eLearningGuild.com
2. Click on the Online Discussion button on the main menu.
3. Using the pull down menu, select the Online Discussion: Journal Topics
4. Select this article from the Subject list.
5. Click on ADD A NEW MESSAGE.
6. Enter your message. It will be posted as soon as you hit the Submit button on the form.

Get Published in...

THE LEARNING DEVELOPERS' JOURNAL

This publication is by the people, for the people. That means it's written by YOU the readers and members of **The eLearning Guild!** We encourage you to submit articles for publication in the **Journal**.

Even if you have not been published before, we encourage you to submit a query if you have a great idea, technique, case study or practice to share with your peers in the e-Learning community. If your topic idea for an article is selected by the editors, you will be asked to submit a complete article on that topic. Don't worry if you have limited experience writing for publication. Our team of editors will work with you to polish your article and get it ready for publication in the **Journal**.

By sharing your expertise with the readers of the **Journal**, you not only add to the collective knowledge of the e-Learning community, you also gain the recognition of your peers in the industry and your organization.

How to Submit a Query

If you have an idea for an article, please submit your article idea by:

- **Sending an email to** Bill Brandon at bill@eLearningGuild.com.
- **Include the following information in your query email message:**

- 1: The title of the article.
- 2: What will the article be about? What is the issue/problem that will be addressed?
- 3: Why is this issue important to the reader? Industry?
- 4: Why are you the one to tell this story?
- 5: List your contact information (name, job title, company, phone, email). This information should be for the WRITER of the article. NO agents please.

- **Limit the information above to approximately one page.**

If the topic appears to be of interest, we will ask you to submit an article. Refer to www.eLearningGuild.com for more details.

Publisher David Holcombe

Editorial Director Heidi Fisk

Editor Bill Brandon

Copy Editor Charles Holcombe

Design Director Nancy Marland

The eLearning Guild™ Advisory Board

Ruth Clark, Conrad Gottfredson, John Hartnett,
Bill Horton, Kevin Moore, Eric Parks,
Marc Rosenberg, Allison Rossett

Copyright 2002. **The eLearning Developers' Journal™**.
Compilation copyright by The eLearning Guild 2002. All
rights reserved. Please contact **The eLearning Guild** for
reprint permission.

The eLearning Developers' Journal is published weekly
by **The eLearning Guild**, 525 College Avenue, Suite
215, Santa Rosa, CA 95404. Phone: 707.566.8990.

The eLearning Guild is an operating unit of Focuszone
Media, Inc., 1030 Beatrice Street, Eagan, MN 55121.

The Journal is distributed to all **Guild** members free of
charge. To join the **Guild** go to www.eLearningGuild.com.

The eLearning Developers' Journal™ is
designed to serve the industry as a catalyst for
innovation and as a vehicle for the dissemina-
tion of new and practical strategies and tech-
niques for e-Learning designers, developers and
managers. The **Journal** is not intended to be
the definitive authority. Rather, it is intended
to be a medium through which e-Learning practi-
tioners can share their knowledge, expertise
and experience with others for the general
betterment of the industry.

As in any profession, there are many differ-
ent perspectives about the best strategies,
techniques and tools one can employ to accom-
plish a specific objective. This **Journal** will share
these different perspectives and does not posi-
tion any one as "the right way," but rather we
position each article as "one of the right ways"
for accomplishing a goal. We assume that
readers will evaluate the merits of each article
and use the ideas they contain in a manner
appropriate for their specific situation. We
encourage discussion and debate about articles
and provide an online SIG Talk™ discussion
board for each article.

The articles contained in the **Journal** are all
written by people who are actively engaged in
this profession at one level or another — not
by paid journalists or writers. Submissions are
always welcome at any time as are suggestions
for articles and future topics. To learn more
about how to submit articles and/or ideas,
please refer to the directions on this page or
visit www.eLearningGuild.com.

About the Guild



The eLearning Guild™ is
a Community of Practice
for designers, developers,
and managers of e-Learn-

ing. Through this member-driven community,
we provide high-quality learning opportunities,
networking services, resources, and publica-
tions. Community members represent a diverse
group of instructional designers, content devel-
opers, web developers, project managers, con-
tractors, consultants, and managers and direc-
tors of training and learning services — all of
whom share a common interest in e-Learning
design, development, and management.

The eLearning Developers' Journal™

The Guild publishes the only online "e-Journal"
in the e-Learning industry that is focused on
delivering real world "how to make it happen in
your organization" information. The Journal is
published weekly and features articles written
by both industry experts and members who
work every day in environments just like yours.
As an active member, you will have unlimited
access to the Journal archive.

Guild Research

The Guild has an ongoing industry research
service that conducts surveys on 20 topics
each year. These topics are identified by the
Research Advisory Committee. The data collect-
ed is available for all members.

Resources, Resources, Resources

The Guild hosts the e-Learning industries most
comprehensive resource knowledge database.

Currently there are over 1,600 resources avail-
able. Members have access to all of these
resources and they can also post resources at
any time!

People Connecting With People

The Guild provides a variety of online member
networking tools including online discussion
boards, and the Needs & Leads™ bulletin
board. These services enable members to
discuss topics of importance, to ask others to
help them find information they need, and to
provide leads to other members.

It's About Leadership

The Guild draws leadership from an amazing
Advisory Board made up of individuals who pro-
vide insight and guidance to help ensure that
the Guild serves its constituency well. We are
honored to have their active engagement and
participation. The Guild has also established
three committees made up of active members
who help steer its editorial, events program and
research efforts.

Discounts, Discounts, Discounts

Guild members receive discounts on all Guild
conferences and on other selected products
and services. Your Guild membership will save
you 20% off the list price of Guild events!

Membership is Completely FREE!

Yes, FREE! All you are required to do is com-
plete a membership profile form and you will
have access to everything listed above... and
MORE! Join today at www.eLearningGuild.com!

Become a member today — FREE! Join online at www.eLearningGuild.com.

THANK YOU TO THESE GUILD ENTERPRISE SPONSORS

CLARK Training & Consulting (CTC) is a global leader in instructional design offer-
ing both training and consulting services. Our award-winning seminars are based
on the latest research in instructional psychology and human performance
improvement.



www.clarktraining.com
Contact: Kimberly Perkins
602-230-9190

Cyclone Interactive is an interactive media and web development firm creating
online, CD and presentation solutions for a wide range of clients and industries.
www.cycloneinteractive.com



Contact: Earl Dimaculangan
earl@cycloneinteractive.com
617.350.8834

Spectra Interactive Learning is a unique, full service, e-Learning consulting compa-
ny — growing and expanding in North America and Europe to meet the growing
need for expertise in e-Learning strategy development, instructional design and
program implementation.



www.spectrainteractive.com
Contact: Brenda Pfaus, President
bpfaus@spectrainteractive.com
Ottawa, Canada (613) 230-9978

To learn how to become a **Guild** Enterprise Sponsor, please contact David Holcombe
at dh@eLearningGuild.com or call 707.566.8990.