PART
One

First Steps

Chapter 1: Introduction

This chapter introduces the reader to the organization of the book and a broad survey of the types of subjects that we will cover. The book is designed to present the reader with a problem in large model development. We define the target audience and state the purpose of the book. This purpose is threefold:

1. Improve your software skills in Excel, Visual Basic for Applications (VBA), Access, PowerPoint, and Outlook.
2. Expand your model design skills by addressing the issues in the planning and execution of large models.
3. Expand your finance knowledge by presenting other asset types and a multiple class liabilities structure.

We will close by discussing how the chapters are structured and the general content and function of each of the major chapter sections.

Chapter 2: The Existing Model

This chapter reviews the existing Structuring Model from which we will launch our later development efforts.

Chapter 3: Conventions and Advice

This chapter is divided into two sections. The first deals with a series of practices that we encourage you to adhere to as you develop the VBA code, build Access data bases, and employ Outlook and PowerPoint in the new models.

The second section of the chapter consists of a few pieces of commonsense advice that many people have found useful and that we hope will save you time and trouble in the future.

Chapter 4: Segregation of the Existing Model's Functionality

In this chapter we examine the advantages of splitting the existing model into two different models. The first of these models will focus exclusively on the activities of modeling the assets of the model. These activities relate to the:

- Screening and selection of collateral.
- Generation of collateral cash flows under various default and prepayment assumptions.

The second of the models will deal exclusively with the application of the output of the asset model (the collateral cash flows) to the liabilities structure. The liabilities structure receives these cash flows as inputs and then applies them to the payment rules of the deal to retire the notes of the structure.

Chapter 5: Building the Base Asset Model

In this chapter we will start with the original model of Chapter 2 and proceed to remove all Excel/VBA related to the liabilities functions. When we are complete we will have produced a model exclusively focused on processing the asset side of the deal.

Chapter 6: Building the Base Liabilities Model

In this chapter we will perform the reverse of the processes we engaged in Chapter 5. Here we will remove all of the asset-focused functionality of the model, leaving us with a liabilities-only model.

Chapter 7: Establishing the New Model Environment

Before we begin to develop the two successor models to those that we created in Chapters 5 and 6, we will lay out the new modeling environment within which this development will take place. This chapter contains a schema for a set of new directories into which we will place the various models, data files, assumption files, databases, report template files, and model inputs and outputs that will become the different parts of the model environment.

A Note to Readers

The activities in Chapters 5 and 6 are designed to serve these purposes:

- Familiarize the reader with working on a model of intermediate size that they did not write themselves. In doing so this process requires them to read the model, understand its basic structure, and add or remove code to achieve the desired ending configuration.

- Acquaint the reader with the types of subroutines and functions that perform the core functionalities of menu management, error checking, data management, collateral selection, and cash flow generation and reporting.
- Introduce the reader to the systemic organization of a large model by applying a specific approach to creating and naming VBA modules to hold the various model elements.

Everyone will, at some point in their career, have to undertake a task similar to the activity covered in this chapter. We hope you will not need to do this more than once or twice. This is because while instructive the first time you have to perform these tasks, you will learn almost everything you need to know by doing this once and only once.

Future repetition of these activities carries a dramatically diminishing incremental value.

Therefore, if in your experience you have already walked the walk, you need not do it again here. Jump from Chapter 4 to Chapter 7 with my blessing. For those of you that have not had experience with this process, studying Chapters 5 and 6 will make it significantly quicker and less painful when you are called on to do so.

CHAPTER 1

Introduction

OVERVIEW

The purpose of this chapter is to give you a view from 100,000 feet in the air as to the goals of the book, how I will present the material, the general organization of that material, and, along the way, a helping of model-building philosophy.

I will also give you a punch list as to what skills we expect you to have. If you do not have these skills, you may need to reconsider buying this book as it will not address a number of fundamental VBA or VBE skills, and that might end up making the work more frustrating than fun. "Fun" seems to be a neglected word when it comes to technical books. It seems that everyone has to be very matter of fact, declamatory, and serious. I think this is overdoing it. I really do not know anyone who is a competent model builder who does not think that a good portion of the intellectual and computational challenges of building sophisticated models is not fun. There may be elements of frustration (see the story at the end of this chapter about a FORTRAN compiler). There may be boredom, or pressure (some of it extreme), but in the end the feelings of accomplishment and validation are worth the effort. In total, the experience is a positive one, and I certainly think fun too.

In this vein, the voice of this book will be a conversational one. We would like you to think of it as an extended chat along the lines of Franklin Delano Roosevelt's series of radio broadcasts that become known as the "Fireside Chats." An even better analogy is sitting with a somewhat more experienced coworker on a joint project. The mentor genuinely wants to share his experiences. He hopes that you can avoid the traps and travails he himself had encountered working his way from novice to journeyman and finally to mastery of these specific skills.

WHY WAS THIS BOOK WRITTEN?

This book is written to address a need for an intermediate-level guide to model building in the primary context of the Excel/VBA environment but with incremental inclusions of other Microsoft products to enhance productivity and performance. There are any numbers of books that deal with Excel spreadsheet development. There are also many that teach the elements of and how to program in the VBA language. The same can be said for Access, PowerPoint, and Outlook. There are also books that will show you how to develop single-purpose financial applications in these languages.

Where there is a dearth of information is how to build models that combine the collective functionality of a number of Microsoft products. Each of these products is employed to optimize its particular strengths in the finished model and to bring the synergy of this combination to bear on the problems the model is intended to address. Most programming or model examples covered in other books are very narrow, single-purpose programs of limited sophistication and scope. They almost always are presented as a start-from-scratch effort. This is a low-probability scenario for a new analyst or associate joining a department with a developed software infrastructure. A person in that situation will need to assess what has already been developed and will face time, cost, and perhaps even personal pressures to make do with what is already in place.

This book seeks to set out a representative real-world development challenge that calls for the analyst to have or acquire an intermediate-level mastery of both model design and development and of software that can adequately address a more robust set of requirements.

Addresses a Need for Large Program Programming

The key element in the aforementioned approach is the use of a large application. Many books shy away from this particular class of problem. Some of this avoidance is legitimate. The book in question may wish to cover a variety of other subjects, or the authors or publishers may feel that a large application consumes too much space and that its value is too limited. Other authors approach the subject but limit themselves to the use of Excel spreadsheets with minimal or no use of the other available software products. Several others deal with large program development but do so in a fragmentary manner. In these works only portions of the corpus of the application code is available. The code presented tends to be either mostly unavailable to the reader or unavailable in a sufficiently critical mass to provide a firm understanding of the practical challenges and full scope of such a development effort.

An earlier book[1] by Preinitz, addressed this issue by developing a rudimentary moderate-size model that was used to securitize a portfolio of small business loans. This model was developed, using VBA, from the starting base of an Excel spreadsheet application.

The finished model application was comprised of several parts. The first was an Excel file, composed of a single spreadsheet. The contents of the spreadsheet represented all the information regarding a portfolio of small business loans that we would need to conduct a cash flow analysis. This Excel file contained the financial and demographic information on approximately 2,000 individual loans. A model was developed to transfer the information from this collateral portfolio file, determine through a user-described selection process the collateral that was usable in the deal, purge the ineligible collateral, and then amortize the eligible collateral under a variety of economic conditions. The model then applied the resultant cash flows to a liability structure. The results of the collateral selection process, the cash flow generation processes, and the liabilities waterfall performance were all captured and displayed in a series of reports. These reports were produced using preconfigured Excel spreadsheets that served as template report files. The model execution options,

file selection, and reporting options were specified and controlled by user entries to a series of menus.

This model, which will become that starting point for the case study exercise in this book, is reviewed in Chapter 2, "The Existing Model."

A Real-World Problem

The issue most beginning developers face is that, like it or not, they usually end up working with someone else's code. This is especially true of those areas that use proprietary applications. These applications are more likely than not the backbones of the analytical framework of the business units that have developed them.

In this environment the situation most commonly faced by the modeler is to produce a series of modifications to the existing model. The first assignment may be something small: adding additional fields to a report, changing the format of an existing report, adding a relatively straightforward calculation to an easily identifiable place in the program. Just as often, however, these changes are not simple. This is especially true when the prior person jealously guarded the code as a form of job security. Depending on the timing and reasons for the person's departure, the severity of the resultant crisis may range from a merely annoying level all the way up to a drop-dead, heart-attack level. Now the lucky man or woman has to read and understand the application immediately, under economic and time pressures. Having the experience of picking up a moderately large application, reading it, understanding it, and then significantly modifying it is under these circumstances a critical job skill.

Thus you can see that aside from whatever other information or skills you acquire from this work, being prepared for such an eventuality can be a real career lifesaver. Not only can it save you immense amount of pain and suffering, but it can be the springboard to better things. There is nothing so valuable as to be viewed as the solution to a crisis, and the more severe and hopeless the crisis initially appears, the better.

Steps in Model Evolution

The subject of Darwinian evolution is still held in contention by a regrettably significant number of people. One form of evolution that cannot be disputed, however, is the evolutionary pathway of models. Not all models pass through the stages detailed below. Many models never evolve past simple self-contained Excel spreadsheets (nor do they need to). Some rise to a higher level of sophistication and become stable. Some models relentlessly evolve, becoming ever more intricate, complex, or massive, or, in rare cases, all three.

You are a person who will, it is highly probable (possibly sooner rather than later), be called upon to work on the dreaded someone else's model (SEM). This SEM should be approached prudently. One of the first things you should be able to assess is where it is on the evolutionary scale of model development. Still floating aimlessly in the sea, has it just developed into a multi-cellular structure? Has it just crawled onto the land? Is it, even now, sitting in the cockpit of an F-22 jet fighter, zooming down on you with all guns blazing? It is good to have the answer.

Evolutionary Hierarchy Although we can always argue how many angels can dance on the head of a pin, in a nutshell, there are roughly nine stages of model evolution:

Level 1. A single-purpose Excel spreadsheet application, an equivalent of an electronic scratchpad.

Level 2. An Excel spreadsheet with recorded macros that address repetitive or onerous tasks. The macros serve as labor-saving devices and perform tasks such as data importation or report formatting.

Level 3. An Excel spreadsheet with VBA subroutines and functions. The blocks of VBA code may or may not be organized into task-specific VBA modules. They are, however, purpose designed and are significantly more complex and powerful than recorded code. All output is contained within the model.

Level 4. An Excel/VBA model that uses menus and external files to input data and template files to output the results. The main computational and analytical burden of the model is now preponderantly seated in the VBA code. The VBA code is interpreted, not compiled.

Level 5. A Level 4 model with Access to address more onerous data management issues. The model may or may not make use of online data sources, such as Bloomberg, or private client Web sites to import data into the model.

Level 6. A Level 4 or 5 model with a more sophisticated and/or intensive user interface. This usually occurs when the model has reached a stable state of development, usually correlated to the maturation or decline of its product line.

Level 7. A Level 4, 5, or 6 model with compiled VBA code in separate libraries linked to the model. This VBA-compiled code addresses those functionalities of the model earmarked by timing software that measure the highest-use portions of the program. Object-oriented programming may now be introduced or may have already appeared as early as Level 4.

Level 8. A Level 7 model with a preponderance of compiled code replaces almost all of the VBA modules. This compiled code is most often C++, Java, or many other general-purpose programming languages based on the particular suitability of the product and the expertise or organizational platforms already in place.

Level 9. The final extension of a Level 8 model, which uses compiled code only. The role of Excel is restricted to the user interface, and VBA has been completely eliminated from the model.

Generally the trend is that the higher the evolutionary level, the greater the support requirements of the model. A good rule of thumb is a factor of 2 for each level over Level 4. This is because as the models grow in size and sophistication, they may require extensive hardware support as well as increasingly burdensome software management. In addition, models with lives of five years or more are usually deemed mission critical to the business unit and are therefore subject to such activities as periodic audits, release control procedures, third-party validation, and valuation methodology reviews.

Where We Are Now The model that we will inherit and be asked to improve is a solid Level 4 model. We will review its characteristics in Chapter 2. (The model and its supporting files will be available on the Web site.)

Where We Will Be at the End of the Book When the smoke clears at the end of the book, we will be as close to Level 7 as we can get without using another compiled language to augment the calculation subroutines. Before we get too far ahead of ourselves, let us examine the fundamental work to be done.

The first thing that we are going to do is break down the existing model into its two natural functionalities. The first functional portion that we will segregate from the initial model will be the code that handles the assets and generates cash flows from them. Upon completion we will call this the Base Asset Model (BAM). The second functional portion of the original model that we will segregate will focus exclusively on calculation the performance of the liabilities when the cash flow streams produced by the BAM are applied to them. We will call this the Base Liabilities Model (BLM). Once this segregation has been accomplished, we will separately develop each of these models from its current Level 4 form to a Level 6 configuration. This will be enough work to keep everyone, especially you, gentle readers, more than busy.

WHO IS THE TARGET AUDIENCE?

In order of immediacy, this book is aimed at these people:

- Those working in the financial industry, especially at the levels of vice president and below, who want or need to develop their modeling skills beyond a fundamental competency.
- Members of regulatory, governmental, or audit functions seeking to be able to investigate and understand intermediate-level models and modern modeling techniques.
- Intermediate-level managers who supervise these people but have little or no knowledge of, or experience with, modeling. They want to know what is possible, how difficult the task is, and how to go about accomplishing it.
- Anyone wishing to develop intermediate-level modeling skills using a combination of Microsoft application software.
- Students in graduate or undergraduate programs wishing to enter the financial community in the near future and interested in trading or banking positions requiring analytical skills.
- Anyone who wants to have the ability to intensively explore problems that require a quick, fluid medium of analysis. This includes anyone in the financial world dealing in risk and especially those who seek to measure it and evaluate risk through modeling.
- Anyone in any commercial, nonprofit, or military function who needs a tool to assist in dynamic problem solving.
- Anyone who is a regular Excel/VBA user and wants to significantly expand the power of the applications by integrating other Microsoft products in the models.

- Anyone who thinks he or she knows and is good at Excel, VBA, Access, or PowerPoint. I am certain that you will find many useful techniques here that you may have overlooked.

WHAT IS THE PURPOSE OF THE BOOK?

The thrust of the book is to expand your skills and knowledge in three distinct but interrelated areas:

1. Your software and application skills
2. Your modeling skills
3. Your financial knowledge

EXPANDING YOUR SOFTWARE SKILLS

The expansion of your software skills will fall mainly along the lines of the five Microsoft products we will be working with:

1. **Improving your Excel,** especially as it relates to UserForms and the user interface portions of the model.
2. **Improving your VBA techniques.** This will span the subjects of supporting the UserForm menu additions, to financial amortization calculation of the collateral, to reporting techniques.
3. **Expanding your Access knowledge** by building databases to support the model by storing, retrieving, managing, and outputting data and results.
4. **Expanding your PowerPoint skills** by learning to harness the combination of Excel, VBA, and PowerPoint to improve model reporting activities.
5. **Expand your Outlook skills** by giving your model the ability to communicate with you and the rest of the world.

Excel Skills

Two of the areas that you will expand your Excel knowledge are UserForms embedded in the menus of the model and working with Excel graphs/charts for augmenting reports.

Employing UserForms in Menus You will learn how to enhance the appearance and functionality of the model by the employment of UserForms in the menus. These forms will allow us to develop a significantly more dynamic model interface. They will also allow us to use a series of lists to control and direct the operation sequence of the model. In addition, they will serve to decrease the chance of model operator errors by limiting the possible choices to valid and meaningful ones.

Improving Reports with Graphs and Charts We will make aggressive use of charts and graphics in both the presentation and working-level reports produced by the model.

VBA Skills

This book also seeks to expand your mastery of the various uses of VBA in model building.

The Real World: New Models from Old Models As discussed above, one of the most important lessons you will learn is how to work with someone else's model. You will be presented with a model of moderate complexity and size, a Level 4 model, and will have to significantly alter it to create the desired model, a Level 6 model. You will start by learning to identify and understand the significant functionalities of the initial model. You will then split this model into two self-contained entities along these functionalities. In the process you will either retain or discard portions the existing code. You will, however, try to preserve as much of that code as possible to accelerate and streamline the production of the follow-on application.

We cannot overemphasize the importance of having experience in this type of activity at least once early in your career. You will find over and over again that it is more often the rule than the exception in model building.

Expanding the Scope and Variety of Inputs We will significantly expand the scope of data inputs and the ability of the user to specify what selection actions the model is to perform on this information. This includes the ability to dynamically build selection and sorting criteria conditions and report formatting options. The size of the collateral portfolios will be significantly larger and more complex. The number of data per loan will increase by a factor of 2 and the number of initial loans in the portfolio by even more than that over the requirements of the base model.

Access Skills

The Access database product will be employed as a part of both the models we will build in the later chapters of this book. Access will serve to more efficiently facilitate the movement of significantly larger amounts of data than the existing first model we will encounter in Chapters 4, 5, and 6. These databases will hold loan level data for the collateral portfolio, collateral selection criteria sets, prepayment, default, market value decline, recovery lag period curves, the portfolio and sub-portfolio collateral cash flows, and the performance metrics of the liability tranches.

PowerPoint Skills

PowerPoint will be used to produce specialized reports to be used in regulatory, rating or investor presentations. We will use VBA and Excel to automatically populate and format PowerPoint template files. This will result in PowerPoint presentation reports without any other need to transcribe, copy, or import the data manually.

Outlook Skills

Outlook will be employed to improve the messaging capabilities of the program, allowing the model to communicate directly to the users of the model. This communication can take several forms, such as calculation progress, data import facilitation,

or any other significant events that occur in the course of the operation of the model. It can also be used to create automated report distribution systems that can dramatically reduce the time and effort of disseminating the output of the model to critical parties.

EXPANDING YOUR MODEL DESIGN SKILLS

This book will seek to integrate all of the above items into an improved model engineering expertise. We will show you how to think of a model as more than a specific single program entity.

It Is a Modeling System, Not a Model

As the model is bifurcated into the Collateral Cash Flow Generator (CCFG) and the Liabilities Waterfall Model (LWM), we will make use of an ever-increasing number of Microsoft products to build a modeling system versus a single stand-alone model. The integration of these various products in such a model, along with its template files, databases, and associated programs, now begins to resemble a system more than an individual model. The flexibility of this approach will give you experience in thinking in a more general and fluid manner about how to combine various program and products to achieve results you desire.

Improving Reporting Flexibility Models are worthless if you cannot effectively communicate their results in a manner that is intelligible to the intended audience. Keeping this critical fact in mind, we will look at improving your modeling knowledge by significantly overhauling both the content and the appearance of the model's report package. We will target specific report packages to specific audiences. On one end of the spectrum, we will have the basic working reports that are designed to support the day-to-day work on the deal. On the opposite end, we will have reports that are narrowly targeted to the particular requirements of a specific audience.

More Complex Report Formats The complexity of the report packages will increase to encompass the increasing amounts of information available to the model. This information will take the form of both collateral data analysis,the composition of geographically adjusted prepayment and default assumptions,and the performance of the various components of the liability structure.

Tiered and Specialized Reporting In addition to the increase in complexity noted above, we will tier our reports depending on the degree of detail and the level of complexity appropriate to the target audience. This specialization will be reflected in report packages designed for both internal and external target audiences.

EXPANDING YOUR FINANCE KNOWLEDGE

The financial knowledge, like the model itself, is divided between the asset side and the liability side of the deal. A broader understanding of the behaviors of collateral

is key as these assets are the preponderant source of cash flows entering the deal. As you expand your knowledge of various liability structures you will improve your understanding of how structured finance attempts to segregate and identify risk.

On the Asset Side

The concepts we will cover regarding the asset side of the model fall into the four broad concepts discussed below.

Using Residential Mortgages in the Structure Residential mortgages have been selected as the asset collateral because of their ongoing relevance in the current economic crisis. They have another advantage. The securitization of mortgages has long been an important part of the financial landscape. They are a data-rich asset class and because of this fact are excellent training tools. They will present you with a variety of risk issues, the conceptual knowledge of which is immediately transferable to practically any other asset producing a stream of payments. There is a saying, "If you can do mortgages, you have a good foundation for everything else."

Different Amortization Patterns The plethora of mortgage products that have evolved in the last 30 years is truly astounding. We can expect to encounter any number of very interesting amortization patterns, including standard adjustable rate mortgages, hybrid adjustable rate mortgages, and both fixed and adjustable rate balloon mortgages.

Pool Level Prepayment and Default Assumptions The current trend in prepayment and default analysis, especially given the wide range of economic conditions nationwide, is to employ geographically specific prepayment and loss curves. This analysis may also be augmented by an analysis of the particular demographic, credit, property, loan type, and servicing capacity for each individual loan in the portfolio.

Loan Level Market Value Decline and Recovery Lag Period Assumptions Market value decline (MVD) is the amount of value loss experienced since the purchase of the property by the current owner until its liquidation sale. This is a factor that is strongly determined by local real estate market conditions and therefore modeled with geographically distinct assumptions. It is the prime determinant in the amount of recoveries generated by a liquidation sale. The recovery lag period (RLP) is the amount of time since the initial foreclosure to the receipt of the recovery amount.

On the Liability Side

Two additional concepts will be introduced when we develop the LWM. The first is the use of multiple bond classes. The second is the use of other mechanisms to preserve the creditworthiness of the liabilities structure.

Multiple Tranche Structure We will introduce the concept of the tranche. The term "tranche" means "slice" in French. Bond tranches allow the deal structure to

custom-design liabilities of differing risk characteristics, weighted average life, and duration to meet specific investor requirements.

The liabilities structure that we will model in the LWM will consist of two bond classes. The first bond class, the A class, will consist of two tranches of bonds, the A-1 and A-2 tranches. The second bond class, the B class, will consist of four bond tranches, B-1 to B-4. There are very important distinctions between these two bond classes. The class A bond classes will have seniority over the class B bond classes. This "seniority" will take the form of preferential treatment in regard to the receipt of the cash flows available from the collateral payments. If an interest rate swap is included in the liabilities structure, the class A bonds will be first in line to receive payments from that source as well. In addition to the priority of payment privilege the class A bonds enjoy, they will also enjoy an enhanced immunity to principal losses. In certain prepayment and default conditions there may be insufficient collateral cash flows to fully retire the principal balances of all tranches of the bond classes. In this case the class B tranches will absorb the losses until the point of their complete devaluation, thus protecting the class A bonds to the extent they can.

Both tranches of the class A bonds will also solely receive principal payments for a certain initial period of the deal. No principal will be retired from any of the class B bond tranches during this initial lockout period.

Additional Structural Enhancements Additional structural enhancements will also be employed in the LWM. One of the most common is an interest rate swap (IRS). This type of derivative is used to negate to a significant (but not perfect) extent the fluctuations between fixed and floating rate interest cash flows. In the case of the model, an IRS may be purchased and included in the deal if we are faced with the task of immunizing the deal from the effects of a potential fixed/floating interest rate match. The collateral of the mortgage portfolio that supports the deal could be comprised of fixed-rate mortgages while the class A and B bonds may have floating-rate coupon payments. If the dollar weighted fixed rate collateral coupon rates always exceed the floating rate debt service payments of the bond classes, things are fine. If, however, the reverse becomes the case, over time the deal structure could find itself in dire straits. All the collateral interest cash flows would be paid out in the form of debt service on the bonds and still not be sufficient to meet the expenses. An IRS allows two parties to agree that one will pay the other a fixed rate of interest while the other pays a floating rate of interest based on a predefined principal balance schedules. This allows, for a fee, the "swapping" of a fixed-rate interest flow for a floating-rate interest flow.

In addition to an IRS, the deal will also employ overcollateralization as a credit enhancement for the bond classes.

ORGANIZED TO TEACH

A certain amount of thought has gone into the organization of this book, its chapters, and its accompanying Web site. An understanding of some of the common features in each chapter will help frame your understanding of the contents more clearly. These common features are designed to present the initial concepts to the

reader, reinforce them with the body of the chapter content, and recapitulate the material in a summary section at the end of the chapter. This approach is based on an old saw about effective instruction. The steps of effective teachings are as follows:

1. Tell the students what you are going to tell them.
2. Tell them what you told them you would tell them.
3. Tell them what you have told them.

CHAPTER ORGANIZATION

Each chapter contains a number of common section headers. These sections, such as "Overview," "Deliverables," "Under Construction," and "On the Web Site," are prospective in nature. They have been included with the intention of focusing your attention on specific upcoming events.

Overview Section

This section gives a one- to two-paragraph introduction about what major subjects to expect in the chapter. It is meant to alert you to the broad concepts and, it is hoped, whet your appetite for the upcoming material. If you are looking through the book after having read it once, this section will concisely inform you if you are looking in the correct chapter for that piece of, as Poe put it so succinctly, "forgotten lore."

Deliverables Section

This section, usually in the form of a list or two, is designed to delineate the specific concepts, skills, or facts that you need to learn from the chapter. This serves as a sort of a punch list in waiting. Keep your eyes glued for these items and be ready to jump on them when they appear later in the chapter.

Under Construction Section

This section lists any changes to the model environment, the model, or any of its supporting template files or databases that we will be making in the chapter. Anytime you see items in this section, the model or its environment will be undergoing significant change. The section will list the names of the specific files or directory elements, the types of changes we will be making, what the form of these changes will be, and why we will be making them. If we are going to be adding significant amounts of VBA code or any new VBA modules to house them, they will be listed here. The section is designed to give you a view as to the incremental changes to the model or models that we will be working on in the chapter.

Chapter Material

This section contains the particular subject and its supporting concepts. We have tried to make extensive use of both diagrams and code samples. We are willing to bet that there are very few books concerning modeling and VBA that have anywhere near this amount of code exhibits or, in many cases, entire modules and subroutines.

The reasoning behind this approach is twofold. The first is that while many may argue that programming is an art, just as many will argue that it is a skill that can be learned by following specific examples. Literally, the more well-organized, well-documented, and well-implemented VBA code you see, the higher the chances are that your code will grow to look more and more like the code in the examples. It is only natural for you to adopt and employ the techniques as you see them presented (at least in the beginning), then to go off on a tangent. You will find that if you follow this book's examples, you will be able to develop better code faster. You will also have a *much* easier time when the day comes that you need to look at your code again after a substantial time away from it. There is no worse feeling in the world than picking up a piece of code that you wrote three years back and not being able to understand what you did or why you did it.

The second reason is that a gradual incremental approach is the best way to approach these subjects. Start with simple and straightforward examples and move on from there. The approach that we find infuriating is to see a nine-step process presented in a book in the following manner:

Step 1 is fully explained.

Step 2 is fully explained.

Steps 3 through 8: "These steps are left to the reader."

Step 9 is the solution.

In most of these cases I immediately ask myself why I bought the book. If I already knew how to do it on my own, I wouldn't need the book in the first place. I promise to never leave you hanging for those interim steps. That is why you will see 20, 25, 30, or more exhibits in some of the more detailed and technical of the chapters.

These issues aside, you almost always find that the first section of the chapter after the "Under Construction" section contains a brief outline of the chapter material to follow. You will know the subject material portion of the chapter has been concluded when you hit the "Deliverables Checklist" section.

On the Web Site Section

This section describes the contents of the section of the Web site that pertains to the material in this chapter. Some chapters will have nothing while others may have many, many files. This section will give you a brief description of each piece of the material and its relevance to the chapter contents. Usually a large "Under Construction" section is highly correlated to a large "On the Web Site" section.

ACCOMPANYING WEB SITE

This book comes with a Web site www.wiley.com/go/modeling. On this Web site you will find a variety of material. This material can take the form of models in progress, data files, report template files, Web site chapter files, and last but not least Web chapters.

Web Site Organization

The Web site is organized on a chapter-by-chapter basis with a section for each chapter that contains material. Placeholder notations for chapters without Web site content simply state: There is no material for this chapter.

Content

The content of the Web site falls broadly into five types of files.

1. **Blurbs.** Blurbs are sections of descriptive text that describe the Web site content for the chapter. They are meant to be a reminder of where we are in the book, what are the current concepts, and how the files or other material on the Web site directly relate to the book. There is generally a comment or explanation for each file, placing it in the chapter context.
2. **Comment files.** Comment files for each chapter provide a guide for downloading the software and any modifications to the model environment that have to be made based on the activities in the chapter. Each of these files is divided into four sections. These sections give an overview of the material, the specific file-by-file descriptions, what you need to do with the files, and any changes to the environment needed to accommodate their use. *It is imperative that you read this file before downloading any web site files containing code*. This file contains critical instructions that are placed there to make your life easier and to minimize the chance of errors in running the models.
3. **Chapter files.** These files are essentially the same as the chapters in the book. Often they are organized along the same lines, having an "Overview," "Deliverables," and other sections you will become familiar with in the chapters of the book. These chapters contain a wide range of explanatory text and examples. All in all they amount to several hundred pages of additional material that the authors thought you might need to look at. Why were these chapters not included in the book? Much of the material is subject matter that we thought you should know but could not be sure you would know. We have provided it to provide a crib if you need to check on a particular subject. Some of the material is extensions of the material in the book that was just too lengthy to include.
4. **Code files.** These files are the model, data, database, report template, or any other application files featured in the associated chapter.
5. **Errata files.** As time progresses, there is a high probability that errors will be discovered in the text of the book or in the code. In this event, I will post errata files before sitting down to a tasty meal of crow.

LEARNING THE "HARD" WAY

I believe that there are two ways to learn how to program—the smart way and the hard way. I hope I am providing you with a valuable tool to learn the smart way. A lot of thought and a lot of work has been put into the book and the Web site. They are designed to work in tandem. The more you use them in this manner, by downloading the models, data, and reports from the Web site and combining this material with the contents of the book, the smarter you will be.

Or you can do it the hard way. . .

"The FORTRAN Compiler Is Broken!"

There is one final thought we would like to leave with you before you leave the Web site section. You can approach the material on the Web site in a number of different ways. On one extreme you can ignore it entirely. On the other you can download the material and painstakingly replicate each of the exhibited code and design changes. Which approach do you think will give you the greater understanding of the principles and practices we are trying to communicate to you?

In 1977 I was enrolled in an MBA program that required all its candidates to take a course entitled "Fundamentals of Business Systems." The purpose of the course was to expose the MBA generalist to the problems faced by the computer specialist. The professor wanted the students to take one additional but voluntary step: to write and compile a simple program. The reward was a significant credit toward the final class grade that all but guaranteed passing the course. I thought I would give it a try. The language was FORTRAN, now nearly extinct in the United States. I bought a basic guide to programming in FORTRAN, studied it closely for about three days, and wrote my first program. The program consisted of 20 statements inside a simple loop. The program was intended to calculate various wage and tax amounts for a six-person employee payroll. Using a keypunch machine, I entered the statements on a set of Hollerith cards and ran them through the card reader. I expected to receive back a report with all the sums in nice neat columns, accurately calculated. What I got back from the FORTRAN compiler instead was an error listing 26 items long. Dumbfounded, I surveyed the, to me, almost unintelligible messages, immediately noting that there were more errors on the listing than I had statements in the program. Coming out of my stupor, I realized the extreme seriousness of the situation. There were hundreds if not thousands of students' work at risk, and it seemed that I was the only one aware of the scope of the impending catastrophe. I immediately made a beeline for the small office containing the three graduate Computer Science students who were on duty to help the uninitiated heathens such as myself. Not wishing to cause a panic, I shut the door before informing them of my terrifying discovery.

"The FORTRAN compiler is broken!" I said.

"Really?" they replied, staring at each other in amazement.

"Yes. Really! I just wanted to bring this to your attention before things got much worse!" I said with a note of panic seeping into my voice.

Needless to say, after their rather embarrassing hilarity subsided, they pointed out to me that what I had mistaken as evidence of a broken compiler was, in fact, an error in my program of my own making. It was in fact what is known as a cascading

error. This was a type of FOTRAN syntax error that triggered the compiler to misinterpret many if not all of the succeeding program statements following the location of the error and report additional errors that were not actually present. Six sets of changes, many new Hollerith cards and four hours later I had a working program.

The moral is: The compiler isn't broken, you moron! You can read about model development all day long, but until you get down and dirty in the code yourself, you aren't going to learn a thing.

NOTE

1. William Preinitz, *A Fast-Track to Structured Finance Modeling, Monitoring, and Valuation: Jump Start VBA* (Hoboken, NJ: John Wiley & Sons, 2009).