ANALYZING SYSTEMS

JAMES A. KOWAL

ANALYZING SYSTEMS

JAMES A. KOWAL

AT&T Bell Laboratories, Inc.
Naperville, Illinois

North Central College
Naperville, Illinois

PRENTICE HALL, Englewood Cliffs, New Jersey 07632

FOREWORD

Some time ago, I attended a conference on software development techniques. Many of the talks at the conference were far from lustrous in quality, and I noticed that several attendees were taking opportunities for a quiet snooze at the back of the room.

After lunch on the second day, I chose to attend yet another talk on structured analysis, the only alternative was "Dump Debugging Techniques for Large FORTRAN Programs," and I settled in for a postprandial nap. Hardly had I fallen asleep when I began to dream that I was listening to a speaker who was fresh, clear, interesting, and informative. I woke up with a start only to find that the speaker really was fresh, clear, interesting, and informative. That speaker was Jim Kowal, whose book you have in your hands right now.

But why, you ask, do we *need* another book on structured analysis, albeit one from a man who is fresh, clear, interesting and informative? After all, there are so many classics in this field, including Gane & Sarson, 1977; DeMarco, 1978; Weinberg, 1978; Flavin, 1981; McMenamin & Palmer, 1984; and Ward & Mellor, 1985.

Well, it's true that these books are classics, but they're classics that tend to emphasize some particular aspect of structured analysis. For instance, DeMarco, 1978, is about the basics of structured analysis; Flavin, 1981, is about information modeling; McMenamin & Palmer, 1984, is about the essential analysis model; and Ward & Mellor, 1985, draws many of its ideas from the so-called real-time world.

The beauty of Jim's book, on the other hand, is that it addresses *all* of the areas that a modern systems analyst needs to understand in order to specify a system. It also shows how these areas fit together. So read the structured analysis classics, by all means — especially if you wish to specialize in a particular area. But I believe that you'll gain a much broader view of structured analysis if you first read Jim's book before you tackle the others.

Meilir Page-Jones,
Wayland Systems, Seattle

PREFACE

In preparing a college curriculum in the field of computer science, the question often raised is: What are the most important things we could teach young people to best prepare them to be valuable, contributing professionals in business and industry four years from now? As part of my responsibilities at AT&T, I frequently interview and hire computer science majors as software professionals. I asked myself: What could we have taught these people four years ago that would have made them valuable to their employers today? Operating systems have been changing rapidly over the years from single-user systems to multiprogramming environments, virtual operating systems, virtual machines, UNIX®, etc. Traditional programming techniques and languages such as PL/1 and COBOL are losing popularity. Today, employers in business-applications development are looking for people with knowledge in fourth-generation nonprocedural languages and do not consider COBOL or PL/1 valuable language skills. Similarly, every 5 to 7 years, the industry has provided hardware with an order of magnitude improvement in capability, speed, and capacity. Management information systems were previously built using hierarchical data-base systems such as IBM's IMS. Today, such systems are being implemented using relational data-base management systems because of their ability to service unanticipated queries. So then, what should we teach that would make a person a valuable professional in the 1990s and beyond?

In the 1990s, as is true today, the greatest demand will be for people who are *problem solvers*. We need to teach young people how to solve problems found in business and manufacturing environments.

Traditionally, the systems analyst fulfills the prominent role of a problem solver in an organization. The systems analyst is responsible for determining what must be accomplished and how to carry on the business of the enterprise. When existing operations do not meet the expectations of management, an analyst is called on to define the source of the problem and to offer alternative solutions. To be a valuable asset to an organization, the systems analyst must define the problem and provide alternative solutions.

The first step in good problem solving is to be able to define the problem, hence structured analysis. With the methods of structured analysis, we can approach problems by immediately working to define what currently exists, how it works, and why there's a problem with the current way of doing things.

Data flow diagrams, data dictionaries, mini-specifications, E-R diagrams, input/output processing charts, event diagrams, event tables, state-transition diagrams

and tables, and so forth are tools to begin the problem-solving activity. There are tools that deal with processes as well as data, content as well as structure, relationships among processes, and relationships among data elements. The tools provide ways of separating the *how it is done* from the *what is being done.* These tools also provide a means of modeling alternative solutions very economically. Models are most useful when they can be verified by the end user prior to implementation. User verification and acceptance are critical to the success of a system because it is the users who must carry out the new ideas if the problems plaguing the current system are to be overcome.

To understand the problem that the analyst faces, it is helpful to do a physical analysis. During the physical analysis, the analyst constructs a model that depicts the physical flow of data and the processes that transform the data into information. The model should be easy to understand in order to be verified by those working within the system. The model needs to be complete and detailed enough to identify problem areas and/or inherent limitations in the current way of doing things.

Processes within a system serve basically two types of functions: those that administer to the system (infrastructure) and are usually related to a specific type of technology, and those that perform the basic data transformations to fulfill the needs of the business (essence).

To understand which functions administer to the system and which carry out the necessary business functions, the analyst performs a logical analysis of the system, which contains three steps:

1. **Identify the processes and data** that serve the *system* from those processes and data that serve the *owner(s)* of the system. Processes serve the owners of a system when they carry out the basic functions and objectives of the system.
2. **Reorganize all data elements** by logically associating them to the main *objects* the data describes.
3. **Reorganize all processes** by logically associating them to the *events* to which the system responds.

Upon completion of the above three steps, the logical analysis of the system is concluded and a logical abstraction of the system can be created. A logical, stimulus/response view of a system is provided by an event diagram, which consists of all the essential processes and data organized around the events to which they respond.

By using a structured approach to system analysis, we can identify the parts of the existing system that do not meet the needs of the users. The analyst can then propose alternatives and model the necessary changes. The scope of change can be in three areas: the technology or administrative processes of the system, the basic business functions of the system, or in both. Using the event model, allows the scope and impact of any proposed change(s) to be easily assessed.

The purpose of this text is to bring together the tools and methods used successfully by analysts in modeling the physical, logical, and data aspects of a system. We can then propose changes to the model and review the proposed changes with the user to evaluate their effectiveness.

The following tools and methods are presented in the text:

- **Context diagram:** The highest-level diagram of a system that shows the boundaries of the system, all of the input and output data flows, and the origin and destination of the data.
- **Data flow diagram:** A graphic illustration that shows the data flow among the processes and data storage areas.
- **Data dictionary:** A directory that contains the definitions of the data elements and data storage areas.
- **Mini-specification:** The statement of policy that governs the transformations of the input data into the output data.
- **Pseudocode/structured English:** The detailed statements of logic that define how the input data is transformed into the output data.
- **Action diagram:** A method of illustrating the structure of the logic specified in a mini-spec.
- **Decision table:** A means of indicating the appropriate actions to be taken when various conditions exist.
- **Decision tree:** A graphic illustration of the appropriate actions to be taken when various criteria are met.
- **State-transition diagram:** A chart that illustrates the resulting change of state during the processing of a finite-state process.
- **State-transition table:** A table that lists the resulting change of state during the processing of a finite-state process.
- **Time-dependency chart:** A chart that illustrates the relationships among processes with respect to time.
- **Input/output processing chart:** A cross-reference of all the inputs, outputs, and data stores of the system and the processes that are associated with them.
- **Essential data flow diagram:** A data flow diagram that contains only the essential activities, data flows, and memories of the system.
- **Normalization and object analysis:** Processes that quantify and organize data logically according to the inherent structure of this data.
- **Entity-relationship (E-R) diagrams:** A graphic illustration of the relationships among the data used by the system.

- **Event diagrams:** A graphic illustration of the relationships among the data and the independent actions inside or outside the system that prompt the system to respond.
- **Event tables:** A list of all the input and output data and the events that cause activity within the system.

Finally, in Chapter 12, a basic system architecture is proposed that permits the analyst to take full advantage of both the process- and data-oriented methods of structured analysis. The advantages of the process-oriented methods are realized when successive decomposition and traditional structured specification tools are applied to capturing input data and transforming them into stored information. The advantages of the data-oriented methods are realized when the methods are applied to stored information, so that the users are able to apply fourth-generation languages and satisfy their own needs without any special development effort.

This text is intended for a college-level course in systems analysis. It provides the definitions and theory that permit one to create a model of a system solution that is complete, unambiguous, nonredundant, maintainable, and simple enough so that it can be verified by a nontechnical user community. The only prerequisite is basic knowledge of computer fundamentals, applications software, and the experience of at least one high-level programming language.

Technology will continue to change, providing even greater advances in hardware and software. Methods will improve, providing a means of more quickly defining and resolving the problems and not merely solving the symptoms. My efforts here present the tools and methods that have been used successfully to solve some of the problems in information systems and other software developments. These tools and methods are valuable assets in our struggle with the most vulnerable part of any system, the frailty of human communications.

J. A. Kowal

ABOUT THE AUTHOR

James A. Kowal obtained a Bachelor of Science degree in mathematics at Northern Illinois University in 1969 and a Master of Science degree in Computer Science/Industrial Engineering at Purdue University in 1976. He holds a Certified Data Processor (CDP) certificate from the Institute of Certification of Computer Professionals which he earned in 1977. He is a member of ACM and IEEE.

Mr. Kowal began his career with AT&T (formerly the Western Electric Company) at Bell Laboratories, Naperville, Illinois, in 1969 as an Information Systems Designer in the 1A ESS™ Switching System's Standards Engineering organization. As an Information Systems Staff Member, he was responsible for developing software that provided engineering and manufacturing information to AT&T factories manufacturing switching systems. In 1973, he became a systems programmer responsible for a corporate nationwide telecommunications word processing and text message system at the Warrenville Data Center, Warrenville, Illinois. In 1977, he was promoted to Department Chief of the On-line Telecommunications and Data Base Systems organization, where he was responsible for all on-line and data-base systems services for the AT&T locations in 36 states. In 1979, he was transferred to AT&T's Western Electric manufacturing plant in Lisle, Illinois, where as Department Chief of Information Systems Development he developed and supported business application software and managed data-center operations. Recently, as the Planning and Engineering Manager of the new AT&T Advanced Communications Package he directed the planning, engineering and development being done at the AT&T Network Software Center, Lisle, Illinois. Currently, he is responsible for the Centrex Applications Processor Systems Engineering group at the AT&T Bell Laboratories, Naperville, Illinois.

Mr. Kowal began teaching in 1969 at Moraine Community College and later at the College of Dupage. In 1978, he joined North Central College and taught in an MBA program in cooperation with the Illinois Institute of Technology, Chicago, Illinois. In 1980, he became a charter member of the Computer Science department at North Central College, where he is now an adjunct professor of Computer Science. He was instrumental in developing the undergraduate computer science curriculum and was involved in the development of a new program leading to a Master of Computer Science degree at North Central College, Naperville, Illinois.

Since 1970, Mr. Kowal has made his home in Naperville, Illinois, with his wife and three children. He is a musician and is actively involved in community, educational, and professional activities.

ACKNOWLEDGMENTS

Human beings do their best work by iteration and this book was no exception. I wish to thank all of the CSC 250 students and my colleagues at North Central College for their patience and tolerance in putting up with all of the early versions of the manuscript. Also, I'm grateful to the many individuals who provided suggestions, comments, and corrections to the manuscript. I'm especially grateful to my family, who gave up their husband and father for those many days, nights, and weekends so that I could complete this work.

CONTENTS

PART I PHYSICAL SPECIFICATIONS

PART I

Physical Specifications

This part presents the tools and techniques for preparing a structured specification of an existing system. The system is modeled in its entirety, including all manual and automated activities. While the model is being constructed, the users, that is, those who work within the system, must be able to verify the accuracy of the model. Thus, the notation must be complete and precise in order to accurately represent the system; yet, it must be simple enough so that nontechnical users can understand and verify the correctness of the model.