Mapping Workflows and Managing Knowledge: Simply, Sensibly, Flexibly, and without Software

Chapter 3

John L. Kmetz, MBA, DBA

President, Transition Assistance Associates;
Associate Professor of Management,
Department of Business Administration; and
Faculty Director, Advanced Project Management Program,
University of Delaware
Newark, DE 19716
USA

This chapter is excerpted from my book, *Mapping Workflows and Managing Knowledge: Simply, Sensibly, Flexibly, and without Software*. Newark, DE: Transition Assistance Associates, 2010. ISBN 978-1460-935170.

This is provided free of charge as a service to my students. It is copyrighted material, and any distribution, reproduction, or use by any other individuals, institutions, or for other purposes without written permission is strictly prohibited.

Copyright © 2010, John L. Kmetz

Chapter 3. Workflow Mapping Fundamentals

return to Contents

Introduction

This chapter provides a standardized Workflow Mapping and Analysis (WFMA) technique for all types of users, ranging from those who need to study and understand complicated problems bound up in a flow of work, to those needing to capture and codify the knowledge used in those processes, to those needing only to diagram a well-understood procedure for purposes of training new employees or documenting a process for an ISO 9000 or other quality certification. In most cases, the applications will have much in common with each other, and most users will find that this chapter applies to them rather broadly. In reality, most WFMA projects contain some elements of routine diagraming and some elements of diagnosis and discovery.

WFMA meets these objectives by showing how the "stuff" of any organization's work, whether tangibles liked manufactured materials, or intangibles like information or changes in patients' states of health, moves through that organization. In today's economy, the "material" that most organizations work on is information, and for most of the time I will also refer to this as material, even though it is not. Any of the things that organizations do to accomplish their objectives involve both the processing of their materials and the processing of information about what to do with them.

While WFMA is an increasingly important tool in Six Sigma programs and other aspects of quality management and performance improvement, there is no standard format for creating workflow maps (Harris, 1999); not uncommonly, these are referred to as "flowcharts" or "flow diagrams," "process maps," "business process maps," etc.

I use the term "workflow mapping" to differentiate what is being done here from these other methods, and to reinforce the idea that what we are creating with this method really is a "map," more often through uncharted territory than we might imagine. In discussing the unknown aspects of processes, the unintended outcomes of processes and the like in Chapter 2, we should expect to find these and other aspects of processes whenever we begin a mapping project. I always encourage anyone beginning such a project to adopt a mind set of "assume nothing."

Hasn't this all been done before?

Before we go on, we should answer a good question—hasn't this type of book already been done, or isn't this material all out there on the Web? I should

point out that to those who have done some web-surfing to examine materials on flow mapping, business process mapping (BPM), business process reengineering (BPR), knowledge management (KM) and related topics, that it may seem rather strange to propose that we should start with another clean slate.

There is no question that a lot has been accomplished to bring workflow mapping to the level of management awareness it deserves, and that much effort has been expended to create a body of tools and techniques to enable managers and analysts to do it. In particular, organizations like the Workflow Mapping Consortium (www.wfmc.org) have been a great benefit to the field. Founded in 1993, the WfMC is a global nonprofit organization dedicated to promoting awareness and uniform standards for workflow mapping and analysis, and publishes an annual handbook. It comprises over 300 member organizations and has developed a standard method for graphing and modeling workflows (XPDL), which enables companies and organizations to use information technology in standardized ways and with maximum interoperability between systems and locations. Many members and nonmembers of WfMC have made major contributions to this field.

In March, 2010, IBM Corporation and Lombardi LLC held a webinar to discuss "lessons learned from the first decade of BPM" (Diaz and Rudden, 2010). This was promoted by another web-based organization, the Business Process Management Institute (www.BPMInstitute.org), also an excellent resource for process mapping information. However, one might ask what WfMC believed it self to be doing if it had been founded eight years earlier than the beginning of that "first decade." While I like much of what WfMC and BPMInstitute have to offer, and am especially glad for the support they provide to an important function like process mapping, these two organizations also illustrate the diversity, and sometimes the confusion, that surrounds it.

So why do we need anything more? The short answer to this question, at least for now, is that while there is a great deal of material on the Web or published elsewhere, much of it excellent, it is unfortunately true that confusion and complexity are big problems with this body of work. I will expand on my reasons for this a bit more in Appendix 2, where I list *some* WFMA software (and explain why I don't list some others), and Chapter 5, where I discuss the broader issue of KM.

Much of what I find on the Web suffers from several problems. The biggest problem is that in order to use the tools and methods found there, the user is going to have to invest considerable time and energy into learning a fairly complex approach to process mapping before they can begin mapping their own processes. These mapping efforts become major projects, with all that a major project entails

by way of planning a major program to detail what must be done, gain support for doing it, and the like.

In making that investment, those users will have to learn a number of conventions and rules (sometimes a large number), and then fit their processes into a predefined method that is usually unique to the specific methodology being used. Increasingly, these "BPM" approaches are associated with software applications and improvement of them, where "process" effectively means "automated process," despite occasional nods in the direction of non-automated methods (e.g., van der Aalst *et al.*, 2002). Along with the growth in sophistication that has clearly accompanied these methods over the past few years, there is also a growing dependence on the use of information technology; I am a big fan of IT, but there are cases where the software and hardware can get in the way of things we need to do quickly and simply. They approaches are also often costly. Of course, companies can always hire consultants to do this for them.

A second problem is that many of these methods are overprescribed in at least one of two ways. First, they either want to make the user apply a relatively rigid set of methods to workflow mapping, often involving special software and predefined approaches to mapping and storing data. This requires learning the vocabulary of that approach, software applications, and basically how to fit everything we need to learn about a process into that framework. In a word, these all are, or become, *specialist* approaches—they ultimately require the level of mastery only a specialist can provide through careful study if they are to be applied correctly.

Specialist methods inherently limit accessability to process mapping and the interpretation of data for many who might benefit greatly from access to these. An individual in a large company in need of help with an inefficient process may often find that a specialist approach to deal with the problem must go through a review and approval procedure that may delay needed fixes for far too long; similarly, the small firm with a limited budget may not be able to afford the fix, even if it could be done quickly. Complexity in process-mapping approaches does not serve these potential clients well.

The second type of overprescription is often that the providers of mapping software and techniques have attempted to catalog processes and constrain the user to applying predefined selections from that catalog, regardless of fit. The logic behind this creation of catalogs is directly derived from the idea of workflow mapping—capture the information and steps involved in a process which creates system outcomes by drawing a picture of it. From there, it is easy to extend the logic a bit and conclude that there must be a finite number of small but recurring processes, and so creating an exhaustive catalog of them would permit anyone to

"assemble" a complex process from the contents of the catalog, with a bit of software support.

The idea is so appealing that in the early 1990's the Massachusetts Institute of Technology (MIT) undertook exactly such a project to create and commercialize a master repository of over 5,000 business processes (Carr, 1999a, 1999b; Malone *et al.*, 1999, 2003). With this repository and its accompanying software, it was thought, any and every organization could map its processes directly by simply examining the company, matching or building its processes from the repository, and then assembling them into a master map. The project was commercialized and turned over to the Phios Corporation (phios.com) in 1996.

The project appears to have floundered. On visiting the company web page, one finds a few entries from the late 1990's, 2003, and 2006, but little evidence of continuous or more recent activity. The list of clients is relatively short. MIT has little to say about it, but in 2006, Phios announced that it had turned its processes over to MIT in an open-source format. Neither of the links to the MIT open-source materials currently work.

What happened? My guess, in two words, is "the inevitable." For reasons discussed in Chapter 2, where we glimpsed the potential variety of processes, workflow processes simply cannot be predefined in the way the MIT project assumed. In addition, given the diversity of companies and organizations, the speed and extent of change and evolution in them over time, and the difficulties in collecting valid information on much of what they do in their processes, assembling a comprehensive catalog would have been nearly impossible. In short, the MIT project was doomed from the outset. It made the fatal assumption that processes are fixed entities which, once specified, will be found in the same form no matter who does them; *i.e.,* they all fit into a KK cell in Figure 1.1. I think that is most unlikely, and moreover, even if it were true, the variety of organizations out there will find many ways to link small processes in their systems that were not envisioned when the repository was created. People will find unique and creative ways to use tools and processes that were simply never intended by their originators. The MIT software is certain to have major problems in adapting to these variations, and I would guess it cannot. Further, exception handling is always a major challenge, and as we will see is often the most difficult part of a process to map and understand. Exceptions, by definition, cannot be predicted, and their resolution cannot be predetermined. In contrast, APQC (2010) may have had more success with their Process Classification Framework (PCF), begun in 1992, in part because it is a classification scheme, not an exhaustive inventory.

In many ways, what seems to have happened to the MIT project is that it ran headlong into the information issues Chapter 1 discusses, and in working

organizations, those obstacles are not theoretical abstractions. The imperfections of information, the adaptability of systems, and the creativity of people always combine to create truly unique entities to which "one size fits all" solutions will never apply. In a complex world, we have to come up with our own answers to the challenges of life. Even in a specialized and delimited field such as software development for "knowledge engineering" (Montero and Scott, 1997), characteristics of the "domain" or general environment of a business are so diverse and difficult to capture that creation of a meaningful knowledge base and the collective definition of what its content should be is a major challenge.

The issues I have with many existing systems have long been recognized by others, as summarized by Alonso and his colleagues (1997) in the abstract to their analysis of automated workflow systems:

> Workflow systems hold the promise of facilitating the everyday operation of many enterprises and work environments. As a result, many commercial workflow management systems have been developed. These systems, although useful, do not scale well, have limited fault-tolerance, and are inflexible in terms of interoperating with other workflow systems. In this paper, we discuss the limitations of contemporary workflow management systems, and then elaborate on various directions for research and potential future extensions to the design and modeling of workflow management systems.

I believe these challenges continue to explain why major vendors of BPM systems all seem to agree that many such software projects fail, wholly or in part (Diaz and Rudden, 2009; Haugh, 2010), and that the quest for the successful IT-based system that will solve these problems also continues (Diaz and Rudden, 2009; Haugh, 2010; Ketabchi, 2010).

For all these reasons, the workflow mapping method in this book is deliberately the opposite of these overprescribed approaches—after learning the simple symbols and rules in this chapter, anyone can quickly begin to model and map workflow processes. The method is better described as "disciplined" rather than "standardized," and while it is likely there will be much variation in the way different users might describe the same process, this method allows for multiple "right answers." My approach is *nonspecialist,* and can quickly be learned and applied by nearly anyone. Most importantly, it can also be understood by anyone who is not a trained user in just a few minutes. Nevertheless, for those who are considering the next step into an IT-intensive system of workflow modeling, they will find that my method is quite consistent with the basic approach used by WfMC.

Workflow Mapping and Analysis

What is Workflow Mapping and Analysis?

As I have pointed out earlier, I differentiate WFMA from IT-dependent mapping methods. WFMA is a graphic method of completely describing the material and information flows necessary to accomplish one or more specific objectives of work, in their correct sequence, in a single job, a process, an organizational unit, or an entire organization. As we will see very shortly, WFMA is designed around *simplicity* and the application of a *discipline*, rather than software or elaborate methodologies. It certainly may use software support, but no specific program or package is required.

A workflow map shows exactly how all materials and information are combined in the correct sequence to accomplish a known end purpose. Unlike a system diagram (Figure 2.2), which shows the overall structure of the system, the workflow map shows the linear sequence of operations on material and information to achieve the system objective. A system diagram and a workflow map for the processes in that system are quite different, and it is often a useful exercise in gaining understanding of a system to diagram both of these. Our present concern, however, is the workflow map.

Workflow maps have a number of important properties. First, they are *graphic*—they show workflow processes visually, diagraming them as a flow of activities and information. Second, the method here is *singular*, meaning that it uses one specific symbol set for all workflows or processes being mapped, and that these are used with a specific set of rules. Third, these rules constitute a *discipline*, and this is as much part of workflow mapping as the symbols. Fourth, they are *scalable*, in that they can encompass all parts of a process at whatever level of detail is selected by the user. Fifth, they are *robust*—the mapping method we will see below can be applied to any flow of work in virtually any kind of organization. Sixth, they are *verifiable*—maps can describe existing processes as they are, and any map can be audited or checked against the actual flow of materials and information and the behavior of jobholders to determine its accuracy. Finally, these properties make WFMA an important precursor for *process improvement* and workflow measurement—all workflow activities, flows, and decisions can be measured in a variety of ways that support improving processes.

Workflow mapping creates a “static model” of a process within a business system. When a workflow map is created and verified the user has a graphic depiction of everything needed to do the work the process includes and identify all information needed to perform the process and handle exceptions in the defined system. If a manufacturing process or an insurance claim process is mapped,

people can “walk through” the process as if they were on the plant floor or at the claim site.

There are two major components of Workflow Mapping and Analysis—the symbols used to graph the workflow maps, and the discipline that governs the methods and conventions used to create the maps. Both are important.

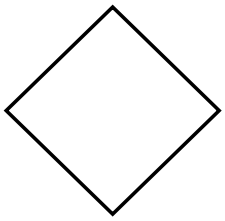
Part 1. The WFMA symbols

In Figure 3.1, the reader will find the basic symbol set used for all workflow mapping and analysis in my method. The usual reaction to first seeing these is, “Is that all there is”? or “Big deal”! That is entirely by design. One of the hard lessons learned in developing this system in my NAVAIR studies is that the symbols have, in fact, very little to do with WFMA; they are a means to identify and order the universal elements comprising any flow of work, but the real objective is to understand the information content of the map, and that has much more to do with the words and expressions in the symbols, along with the relationships between them, than the symbols themselves. The symbols are important, as we will see, and must be used in a consistent way. But they cannot be, and should not be, the dominant part of the mapping process. This is one key element of the designed simplicity of this method.

The graphic devices in Figure 3.1 are a very basic set of flowcharting symbols which have been adopted as the entire set for creation of workflow maps. No other symbols are needed, and as we will see, no others are desired. There is nothing unique or original about these, and many different flowcharting programs and systems use them as well. Why these were selected is simply that through multiple trials of mapping workflows, validating these maps with operators on the site, seeking diagrams that could communicate clearly with other analysts, managers, and specialists in the Navy and elsewhere, and which would do all the things I needed. These five survived a Darwinian selection process. They work. They have been used in every conceivable circumstance and are so robust that any organizational process can be mapped using them.

The first is the simple **rectangle** (and this must always be a rectangle—a square is not an acceptable substitute). It can show a huge variety of events and operations, including processes, activities, and location; it can be used to designate delays or holds (the absence of activity over time), such has waiting for paint to dry. It can be used for materials and information. It is often referred to as a *process block*.

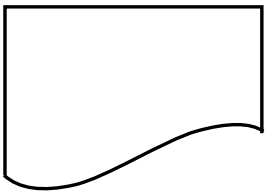
RECTANGLE:
- Process, Activity
- Location
- Delay, Hold
- Both material and information symbol
-



DIAMOND:
- Decision, Branch
- Two *mutually exclusive* exit paths only (yes/no, true/false), *always labeled*
- Primarily information symbol
-

CIRCLE:
- Used in matched pairs or groups to connect parts of flow or continue across pages (“A” connects to “A,” etc.)
- Used singly for “start” and “stop” terminals (traditional oval variation OK for these uses)

ARROW:
- Material or information flow in direction of arrowhead
- Always single direction and single arrow
- Material and information symbol
- Labels may be attached for clarity



DOCUMENT:
- Paper input or output
- Used for required documentation in or out
- Material (“quasi-material”) and information symbol

Figure 3.1 The WFMA symbol set

Source: John L. Kmetz, *The Information Processing Theory of Organization: Managing technology accession in complex systems* (Aldershot, England: Ashgate, 1998): Adapted from Figure A8.1.1, p. 378.

The second is the **diamond**. This is simply a small square rotated 45 degrees, and represents a decision or a branching point in a flow. It is because this is a rotated small square that the rectangle must never be replaced with a square—these symbols must be kept visually distinctive. The diamond is the only symbol that has two exit paths, which must be *mutually exclusive*. It is primarily an information symbol, although it can be used to show how physical items are parsed or divided.

Third, there is the **circle**. This can be used individually as a terminal to designate "start" and "stop" points, but is most commonly used as a continuation where matched pairs enable a large, complex map to be arranged in sections with connectors for clarity. A connector labeled "A" at the end of an arrow designates that another "A" at the head of an arrow leading into the workflow is where the flow continues. These may also be used to indicate where different parts of a path might lead back to a single re-entry point in a flow. (Since many programming purists over the years have harassed me on the use of connectors for "start" and "stop" events, I have allowed the small circle to be stretched into an oval, without repercussions for those who do this, but an oval is not really necessary.)

The fourth symbol is the **arrow**. This is a single-headed arrow to be used individually between other symbols. Pairs of arrows from any source other than a decision diamond are not allowed, nor is a double head, i.e., an arrow with opposing heads on each end.

Finally, there is the **document**. This is a bit of an anachronism. The truth is that most of what is shown using a document symbol can be represented just as effectively with a rectangle, but I have found that there are often circumstances where the need to show that a document was produced, or a rule referred to in a process, or a report generated, and the like, were very important. For that reason, it survives.

All of these symbols are associated with brief descriptive information. Process blocks (rectangles) usually contain *names* and *statements*. Diamonds usually contain *conditions* in the form of *questions* or *tests*, and these should be unambiguous. Connectors tell us what connects to what, or where we start and stop workflows. Arrows are often used without descriptions, but paths from diamonds should *always* be labeled. Finally, documents should have a description of either what was done (like a process block), or what the document is, and sometimes both.

The selection of these symbols was rather Darwinian, as I suggested earlier. What I mean by this is that they are the "survivors" of a number of different candidates that I tried over time. There are several related reasons for their

survival, and for my insistence that they be used only as shown here. First, they are all visually distinctive—if drawn as shown, there is no possibility for any of them to be confused with any other symbol. Second, they can be learned very quickly, without requiring a user to first learn either a complex set of symbols or the mastery of any particular software. Third, they are robust, such that they can be applied to any flow of work. Finally, they are comprehensive—they are able to capture all activities and information in a workflow or process. Together, they effectively establish a standardized graphic vocabulary for mapping workflows in organizations.

Part 2. WFMA Discipline—the Rules for Symbol Use

In Chapter 2 we saw that *processes are a combination of actions and information.* However simple the symbols are, do not be deceived by that simplicity! After many years of experience with workflow mapping, there are three issues that always confront new users of WFMA: (1) Mapping the actions in a workflow is by far the easiest task, since these are the things that actually move through the process. This can create a challenge to new mappers, because they tend to focus too exclusively on the actions, and have difficulty with the information. (2) Seeing and capturing *information* in a workflow map is one of the most challenging parts of the job. We run into both explicit and tacit knowledge everywhere, and capturing that is always a far bigger challenge than capturing the primary flow of materials. The tacit information is tricky because it is characterized by "four I's"—it is individualized, internalized, invisible, and idiosyncratic (often to the company or organization). (3) Part of the mapping process is to create a document that is clear and orderly on the page, so that it is as informative as possible to its users. This will inevitably take more than one draft to get into the final form we want. Like any other skill, *practice* at workflow mapping is necessary, and makes it easier every time we do it.

There are a number of rules for symbol use, which are discussed in the next section. I refer to these as a *discipline*— "rules" rather than "suggestions" or some gentler term—because I have found that violating them, even slightly, leads to still more violations, and that these progressively destroy the standardization and simplicity which makes this method of WFMA robust and effective.

The rules are the syntax for constructing workflow maps. When applied correctly, they make it possible for the symbol set to quickly retreat into the perceptual background of any WFMA user. *The objective of such a simple set is to make it disappear in use—if more than two minutes is needed to completely learn the symbols, they are too complex*. Literally, if one flips through a five-slide show, with one symbol and the bullet points explaining it as shown in Figure 3.1, an

audience with handouts should need no more than 25 seconds to understand each symbol. They will need practice using them, of course, but accurate recognition should be achieved quickly.

Having said this, it is still true that the basic WFMA methods here can be learned quickly and efficiently. By keeping both the symbols unchanged and the discipline consistent, anyone—jobholder, analyst, consultant, or manager—will be able to create and interpret workflow maps in one morning.

The "consistent way" of symbol use discussed in the preceding section is the most critical part of WFMA, and what I am referring to is the *discipline* that must be used to apply WFMA successfully. This discipline must be learned and ingrained at the outset, and required in all workflow maps created with the symbols. This discipline is necessary to *preserve the simplicity* of the WFMA system, and thereby its power to describe a vast array of different kinds of workflows. At the same time, users should recognize that this discipline does not result in a single "correct" map for a given workflow or process; in fact, it makes it more likely that it will result in multiple possible "correct" maps, none of which will necessarily be the same.

How can this be so? I will deal with this issue in more detail in Chapter 5, but for the moment the answer is that any workflow map is a joint product of the flow of work being mapped or described, and the knowledge and perceptual processes of the mapper, whether this is a jobholder or a specialist. There is no way to isolate these or their effects on each other, and so every map produced will be a function of what work is done interacting with the eye of the beholder. This is also why a disciplined approach to mapping is required. There are seven basic components of the WFMA discipline.

1. Keep It Simple, Stupid (KISS). The basic WFMA symbol set is based on a sound psychological principle that favors fewer rather than more symbols (Miller, 1956). Busy people have a hard time keeping large numbers of information "chunks" (including symbols) straight in their minds, and overload is never far away.

Many people suggest that a method that makes workflow mapping easy is to simply write down, in step-by-step fashion, what is done in a process; some even contend that this is more than enough to provide the information a map would yield. I disagree with both of these arguments. Very few processes are so completely linear as to allow a clear, well-organized written description. What do you do, for example, each time you come to a branch or decision? How do you keep track of where you have been when you're on page 26 and don't quite remember which set of branches you took to get there? When you have to depart from a main process to deal with exceptions, how do you track through frequently complex exception-

handling procedures and remember what was happening when you return to the main flow? Diagraming the workflow seems preferable to verbal descriptions for all of these reasons, and more.

2. Use the symbols consistently and correctly. A major benefit of simple symbols is that they have highly specific meanings. Using them as intended prevents confusion; be especially careful not to violate these rules unless there are truly compelling circumstances:

- Use the symbols only for the purposes shown. For example, never show decisions in rectangular process blocks (a fairly common error, actually).

- Use brief, clear text within the symbols (as much as possible) to communicate what is happening at each step. If additional annotation is needed, this can be added in other locations on the page or through notes—this will be discussed in more detail below.

- Use only single arrows, always with the arrowhead shown to designate direction. Two-way or double-headed arrows are ***never*** allowed.

- Use only ***two mutually exclusive*** exit paths from a decision diamond (unless there is truly no other choice, and that is extremely rare). Mutually exclusive paths mean that there can be no confusion over what each path means or why one would be selected rather than the other. This is an extremely important rule, and in some cases may make a workflow map segment larger than multiple exit paths would suggest.

  Here are two examples: if we apply a first coat of paint to a wall and want to recoat it, we might show this as a decision ("first coat dry?") with "Yes" and "No" exit paths. However, which one applies to "touch dry"? We might instead use three paths labeled "fully dry," "touch dry," and "wet." To make sure that we do not mistakenly apply the second coat too quickly, we may need to add one or more additional conditions (decision diamonds) to the diagram to be unambiguous. Alternatively, we might ask in a first diamond whether the paint is "touch dry," and in a second diamond whether at least eight hours have passed since the last of this coat was applied (surface-dry paint might still wrinkle if recoated too soon). A "yes" to both of these might actually make "fully dry" redundant. Even a

simple matter like this may be represented by more than one “correct” map.

In a second case example, credit-card customers were being reviewed at random, and for those with some questions about their credit histories, a possible outcome of the review was to limit their account or suspend it entirely; most, however, were at least renewed and frequently offered a higher line of credit. The student with whom I had this argument made a logical point—there could be five outcomes from a single review decision. These were (1) raise the limit; (2) renew the account at present limit; (3) renew the account with lower limit; (4) suspend the account for a limited time; and (5) close the account. The student argued that all five exits paths could come from a single “Account review outcome?” diamond, and that there was no need to make this more complicated by the addition of a stepwise series of two-outcome decisions. (I insisted on a progressive series of decision diamonds with yes-no exit paths.)

In both cases, it sounds logical to use more than two exit paths from a decision, since this summarizes the actual outcome states; however, I disagree with the association of all possible outcomes with a single decision. My contention is that if we do not insist on mutually exclusive paths at all branches in the workflow map, it is inevitable that ambiguous conditions for branching will creep into the map, and what seems absolutely clear to the person(s) creating the map will be completely confusing to other users. My personal experience with this issue bears this out—in another credit-card case, one student argued for three decision exit paths, labeled “Yes,” “No,” and “Maybe.” I asked the student to define “Maybe,” and of course this was a nested set of further questions and conditions that had to be detailed in order to understand the full process. This confusion never arises when decisions or branching conditions are mapped in mutually exclusive terms. Moreover, the consistent use of two paths becomes habituated for both mappers and users, so that when a decision diamond is encountered, there will always be two paths to evaluate—no more, no less.

The most important reason for consistent use of two mutually exclusive paths, however, is that these require the creators of the map to fully articulate the logic for the branching that occurs at that point. This is often harder to do than we might think, because much of what gets done within many jobs is a process of evolutionary change and internalized (“tacit”) learning. Having to spell out all that

is done, and why, involves more head-scratching than is immediately apparent, but the payoff is that a great deal of what we will later discuss as *tacit knowledge* is uncovered this way. I will have more to say on this subject in Chapter 5.

- Use only one *exit* arrow from process block rectangles; however, there may be multiple arrows *entering* a process block. There are many cases and situations where multiple paths and subroutines in a workflow can converge on a single process or activity; *in no case, however, does a rectangular process block have more than one exit arrow*.

- Put brief text *labels* on exit paths from a decision diamond. This enables the user to follow the map clearly, and helps keep the logic of the map visible and accessible to the user. Arrows between other symbols, and those entering decision diamonds, are simply flow paths and usually are not labeled; there are cases, however, where a label adds valuable information, and if so, then label the arrow.

- Avoid complex backward flows. These are acceptable on small areas of the map, but can be confusing on large areas. The problem is typically that in some part of the diagram, flow arrows will have to cross or "jump" each other. Use connector circles instead to break the map into coordinated sections and pieces which can be followed more easily.

- Be careful of creating endless loops. This can happen for several reasons, but is often because of the incorrect use of double (or double-headed) arrows in a flow, or ambiguous criteria for a decision or branch. In more complex situations an endless loop may be created by a condition that sends one far upstream in a workflow, and you get caught in a loop without realizing it. Validation (see below) will catch this.

- Avoid making a single page too dense, whether with large numbers of symbols and arrows, too much text, small symbols and fonts, or a combination of these. In an earlier lifetime in the printing industry, we were always concerned about the amount of "white space" on a page, meaning that there was enough unprinted area to make the printed parts accessible to the reader and the page visually appealing. While one always has to struggle with the tradeoff between detail and content per page versus overall document size, adequate white space

is beneficial to workflow maps, as it is to most documents. Connectors help solve this problem.

- When using connectors, there is usually only one start location in a workflow map, but there may be multiple stopping points. These *terminal* points should be identified clearly. It is also best to match *continuation* connectors clearly, so that a point where a loop or another part of a process may rejoin an activity flow from several other locations cannot be confused—all "C" connectors that are exits from part of a flow should connect to only the single correct "C" reentry point, and so for all connectors.

3. A "correct" process map captures an unbroken sequence of processes and branches from beginning to end. In short, from the start of any process map, it must be possible to follow the sequence of activities to any end point(s) with no gaps in the sequence and with no stopping point(s) other than completion of the work cycle. In most cases, we want to map the full "work cycle" in the system, showing the input (or how we get it), the transformation process (probably in some detail), and identify the output (or the stopping point).

The stopping point in the workflow being mapped may not always be the end of the full cycle as it exists in the business relationship. For example, it is common in many maps to trace the workflow to the customer, and sometimes into the customer's application of the output. However, if the map you choose to create is only for one department of several on that workflow, the "end point" you select is the point at which the department completes its part of the work cycle. The same could be true for the "starting point"—it may be where the work in this department begins.

When the symbol set is properly used, there will be no places where one creates a document or comes to a process and stops; there will be no gaps; all branches from diamonds will flow to a completion point or will merge back into the flow. It bears repeating that no two analysts will be likely to do the "correct" map the same way.

4. Start by creating *actual* maps of the workflow, not the maps that "should be"! A sure way to waste lots of time with WFMA (or any process mapping effort) is to let people diagram the way (they think) work *should be* done when the mapper needs to know how it is *actually done now*. Most WFMA projects start with the intention of description and diagnosis, and often are intended to support process improvement. Validation, the next step, will help ensure that the map really reflects what is being done now, and that is the point of departure for any application of the map. Mapping what should be done is almost always an

exercise in fiction writing, to some extent. Fiction writing is *not* the right idea for WFMA, no matter how good the final story.

This should not be interpreted as meaning that a normative, "should be" map can never be drawn—a normative map might serve as a valuable straw man for discussion of how one or more jobs might be redesigned, for example. The problem is that when this approach is used for many existing jobs and workflows, the version of reality that is reflected in the map will be an idealized view of it, and often a view held by one or two passionate individuals alone. In mapping the workflows on seven aircraft carriers and seven shore sites in my NAVAIR studies, I found that not a single one was the same; nevertheless, I had three experienced chief petty officers provide me with "ideal" workflow and organizational maps for different sites. None of those matched, either.

5. Validate the map. When a map has been done, trace it through to the final details and verify the map against the workflow. If the map doesn't completely match the process, it is not a valid map, and you don't know what is being done in this case. Revise the map and try again. Seldom does a first draft not need revision.

This is a task that often falls on the manager or originator of the mapping process, and sometimes requires the support of higher management (such as the process owner, to be discussed in a later chapter). In simple terms, that person often has to be a pest and do a "walk through" of the map with the person(s) who created it. At every step, that "pest" needs to be prepared to ask questions, clarify terms, correct some of the mapping discipline, add or subtract steps, and the like. Typically, a revised map will be needed, and this may happen several times until the map is fully validated. This is a high value-added step, and if it is not done there are many things that can go wrong with workflow mapping, producing maps that are ambiguous, incorrect, and misleading.

In the majority of cases, it is most important to get a fully validated map. This means that there will be "little things" that are done that don't match the general flow, i.e., exceptions, and many of these aren't "little things." It is not at all uncommon for the major part of the work done in a process to be the smaller part of a map, while exceptions and deviations from the norm eat up lots of time and energy to diagram, often mimicking the reality of actually doing them. Getting this information will take persistence, but this is where process variation lives, and it is often where the richest opportunities for process improvement are found as well.

Rules 4 and 5 are parts of the mapping process where information flows become paramount. This is true for several reasons. First, most people are familiar with the routine process that works correctly, so it is easy to map it and describe its

details. This also seems to be the most important part of the process to map, since it is where most companies add value that customers are willing to buy; it indeed seems logical to pay the most attention to this part of the process.

However, these routine material flows are not the part of the process that necessarily requires the most attention from workflow staff; that distinction is usually associated with the exceptions we referred to above, and these are often complicated with respect to both the sources and causes of them, as well as from the work needed to correct them and get the process or product back on track.

In validating a workflow map, the majority of what will be added to the map will be examples of information-processing activity undertaken solely to deal with exceptions, and much of this may be for specific types of exceptions. These graphics may well be larger than that for the entire routine workflow. Generating them often seems to be more time-consuming than justifiable when it is first done, given that exceptions are not where value is perceived to be added—people often resist doing this part of the process, for that reason. The tacit nature of much of this information also makes it difficult to describe and to map.

My experience has been that creating both a complete workflow map—one that attends to all the details as they now exist—and validating that it is in fact the actual process that is followed both when thing go right and when they do not, is the most challenging part of WFMA. This is true largely because the analyst must dig out all of the detail that has to be acquired, processed, understood, and put into the map in such a way that both the nature of the exception and the process to handle it are clear and fully communicated. For experienced personnel in many organizations, this is an area where deep internalization of information occurs over the course of many exceptions, and this is the “tacit knowledge” in the workflow. This can be discovered and mapped, but it is typically much easier to carry out this process than it is to fully describe the activity, the information required, and the outcome it should generate.

6. What happens before, and what happens next? If a map can’t account for where something came from before it got to this location in a process, or where it goes next, is it really connected to the rest of the organization; is it really connected to critical suppliers; is it really connected to customers? *Again, organizations are systems.* Valid workflow maps must show what the system is actually doing, and without a valid map meaningful process improvement or other change is most unlikely.

This may be related to map validation in that the “before” and “next” may refer to exception-handling processes. In many cases, companies may find it helpful to start with mapping the normal, routine, successful workflow and get that

correct (this is good practice for mappers, too), before going into mapping the handling of exceptions.

7. Make WFMA your own, but limit "customization." Every organization needs to make a few changes to the basic symbols to account for things unique to that organization. There will also be some need for an organizational standard format. Do these things, but only as much as really necessary. Using "swimlanes" or color to designate where parts of the process are done are fine, for example. *Do not* add more symbols (or "modifications") to the symbol set.

This rule is especially likely to be violated by those with flowcharting, data flow diagraming, or similar kinds of experience. The temptation to add a shadow to a rectangle after doing it for years is hard to resist; so is using a round-shouldered as opposed to a right-angled rectangle, or double vertical walls. But doing this now requires the user to learn, and to carefully observe, these subtle differences while also attending to the logic of the flow and the information contained in the symbols; this introduces noise and inevitable confusion to the system, and these are never good things.

Like the symbol set, these seven rules for use of them are also simple. Following them consistently is often neither simple nor easy. The discipline needed to make WFMA effective is to apply these rules strictly, and not to allow variations to creep into either the maps or the procedures that produce them. If this is not done, any moderately large firm, or firm with geographically separated offices, will inevitably begin to develop divergent and incompatible WFMA techniques, and the power of WFMA will be lost.

Basic mapping conventions

Figures 3.2 to 3.4 illustrate the major conventions for constructing workflow maps.

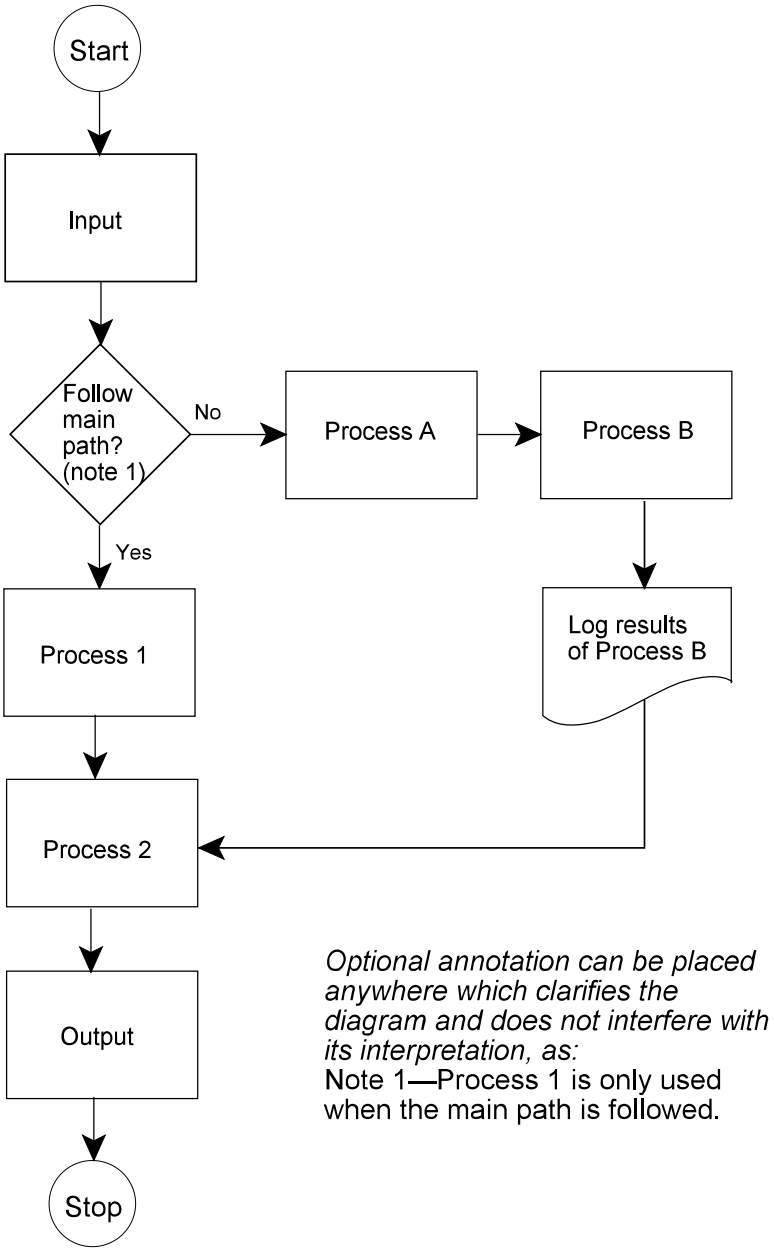


Figure 3.2 Single-cycle process flow

Figure 3.2 illustrates the principal relationships between diagram elements as they might be assembled in nearly any workflow map. Annotations of various parts of the process can be added if they add clarity, as shown, and this will be discussed momentarily.

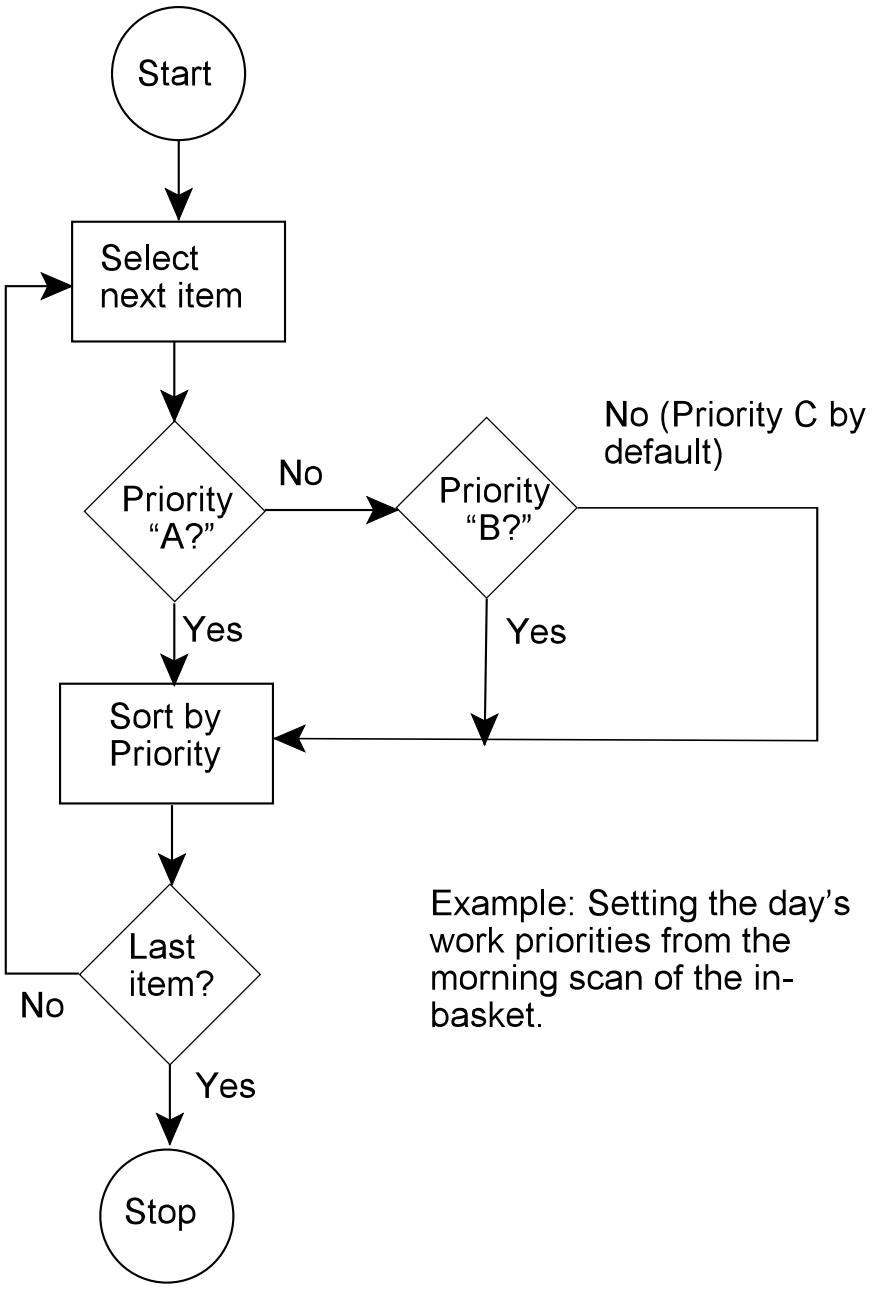


Figure 3.3 Multiple-cycle ("looping") process flow

Figure 3.3 illustrates a repetitive loop in a diagram. While I generally discourage backward flow loops in diagrams, this is a case where it clearly makes more sense to keep the diagram on one page rather than force some kind of

stylistic compromise to avoid such a loop. Proliferation of such backward loops, however, can reduce the clarity of a map, and one excellent application of connectors is to reduce the extent to which backward (and especially crossing) flows are used in a map.

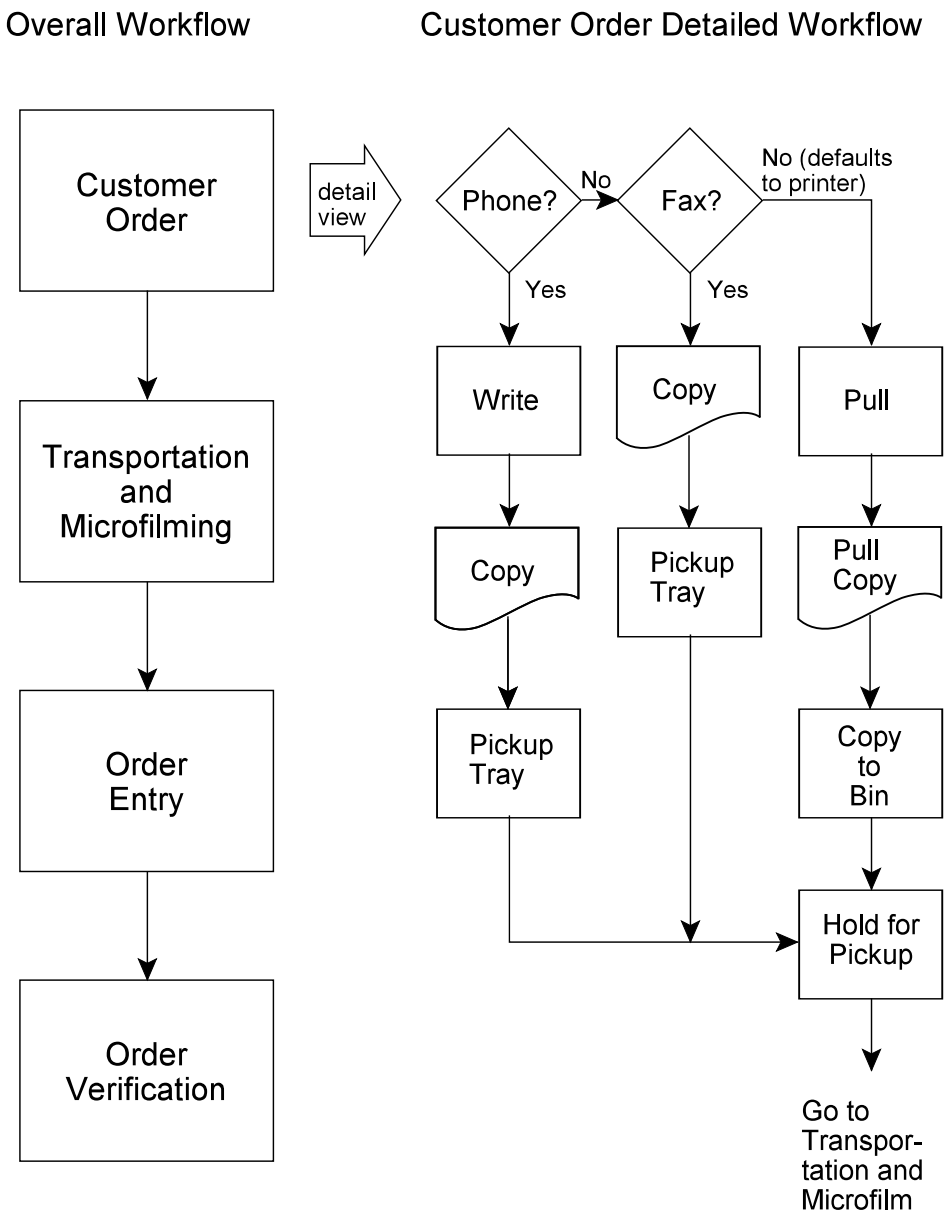


Figure 3.4 Two levels of detail in WFMA ("drill-down" or "layer")

Figure 3.4 illustrates how processes can be viewed at differing levels of detail. On the left side of the figure the four high-level steps necessary for completion of a transaction in this financial institution are shown. The right side shows the first of these four steps, the taking of the customer order, in more detail. Here we can see the three different ways that orders can be received (telephone, fax, or via dedicated computer link), and the additional activities needed to prepare them for the next major part of the process.

Using annotation in WFMA diagrams

The general rule for the use of text in WFMA is that we want to keep as much of the necessary text and information about a process flow inside the process symbols as possible, or closely associated with them. This is part of the WFMA discipline and a payoff of it—when the user follows the diagram, there will be an automatic association between the symbols and the information in them. Together, these fully define the workflow we are interested in mapping.

Occasions arise where annotation apart from the symbols is necessary, and there is a wide variety of such circumstances. In making a workflow map for training new personnel, we might note to the trainee that things done in one step may be referred to another office under some conditions; that there may be supplier or other external relationships that could affect a part of the flow; that random samples of work may be taken for quality assurance; and the like.

Whatever the situation requiring annotation, there are several rules of thumb that should be followed to the greatest extent possible:

1. Try to keep annotation within the area of the relevant symbols. This simply associates the annotation with the process steps it refers to, and prevents the user from having to go somewhere else in the document to get the desired information.

2. Do not mix annotation directly with process information in the process symbols. Annotation implies supplemental information, and it is best to keep it that way. When annotation is used, it should be apparent that it is annotation.

3. Avoid using straight arrows; connective symbols for annotation should be avoided. I generally discourage the use of symbols to associate comments and annotation with the elements of the map—introducing the famous engineer's "curving arrow" (see the "system boundary" in Figure 1.4) may be appropriate in some cases, but too many of them

clutter the diagram and make misinterpretation of it more likely rather than less. My preference is to use relatively standard typographic tools such as asterisks, parenthetic note references, footnotes, and endnotes to associate supplemental information with the diagram. Footnotes are often a very effective way to make this association, and are automatically fit onto a page by word processors. This is a convenient association for making printed copies of a map, and is also useful for electronic documents, since relatively large displays with high resolution have become common.

4. Use annotation appropriate to the level of detail in the workflow map. Annotation is sometimes used to address matters at levels of detail much higher or lower than the level of the map, and this can be confusing and distracting. If one is looking at the high-level (left side) view of Figure 3.4, for example, it might be appropriate to note why the sequence is what it was at that time. (This is an historical case where time is money, and these are high-value transactions; thus, nothing was done until the order was logged and officially in the system. Needless to say, this was automated a long time ago.) It would not be appropriate to explain at the high level that pulling a copy from the computer printer meant separating five-part forms with carbon sheets between them (as was the case then); this would be appropriate to the detail view on the right side.

   Cases like these are where the layering capabilities of many drawing and workflow-mapping software packages can be used to excellent advantage. Figure 3.4 would be separated into two layers (the high level and detail views), and each could be annotated as needed. A user wanting to have easy access to information at different levels would only need to select the desired layer to see the alternate view.

5. Hyperlinks. Hyperlinks, the familiar blue-colored underlined connections we use to browse the Internet and link documents, are easily created with nearly every major software package, and can be a very useful form of annotation. As an example, suppose that a particular step in the administration of an experimental drug to a patient requires the consultation of an attending physician; specific attention to several important patient-health details as well as other matters; and that a form must be signed by the appropriate physician upon giving a dose. Trying to annotate this much supporting information at a single point in a workflow map might be dysfunctional given the length of the procedures and forms to be completed.

> However, this requirement could be met with annotation such as, "Only attending physicians are allowed to give this drug and they must document their actions every time." The hyperlink associated with "document" may connect to a lengthy and complicated research protocol; the workflow map, on the other hand, can immediately and clearly go on its way.

Somewhat more time is needed to learn the rules, but these should also quickly become part of the disciplined approach to WFMA. Learning the rules is best done with hands-on practice, beginning with something relatively simple and then progressing to a more complex process. *Nearly anyone should become an "expert" in WFMA diagraming in no more than a few sessions.* Validating the map and doing the rest of what WFMA entails takes a bit more time, but the basics should be successfully acquired very quickly.

Again, what I learned in my early experiences with WFMA is that the information conveyed by the map, through the brief descriptors used in conjunction with the arrangement of symbols that graphically describes how something gets done, is the only thing that matters. If the mapping graphics and rules get in the way, they obscure what they should reveal, and they distract from the analysis rather than support it. In no case is this a good thing, and in situations where there may be political or other opponents of what is being done with WFMA, these distractions provide ammunition for them to discredit the entire program and its objectives.

Types of workflow maps

What types of workflow maps we can produce with the WFMA symbol set and rules? In one sense, there is only one kind of map, because the disciplined use of the symbol set works the same way no matter how it is applied. Different kinds of maps can be produced, however, depending on the focus of the WFMA user. There are basically two questions to consider—do we want to focus on a process or another kind of activity, and do we want to be comprehensive or map a segment of that activity? First, maps can be constructed with a focus on *tasks* or *jobs*, as well as processes. That is, we can create workflow maps of individual *tasks* within a larger job; or we can map the entire *job*, which will always have at least one task, but nearly always has more. We might want to map the work done in or through a *group*, whether the group is a team, part of a unit, or a whole unit, and could also include work done by contractors and others who contribute to the output but are not members of the organization. At the upper extreme of group size, we can even discuss mapping an entire organization. No matter what the scale, the objective of this type of map is to capture the job content and knowledge

in the relevant system, where that system is not a "workflow" as we have been using that term. This kind of application has been used by several clients to create graphic job descriptions and supporting training documents, among other things.

Second, we can map specific types of *workflows*, without respect to whether the map captures the entire content of any particular job on the flow. This is the most common type of map and implicitly has been the type which has been the focus of our considerations thus far. To treat a patient in an emergency room requires the services of a receptionist (if the patient is self-transported), an orderly, one or more nurses, and an ER physician, at the minimum. Some or all of these people may have things to do that go beyond the emergency patient-care workflow, but our concern here is with the handling of the patient. The workflow map for a cardiac emergency will not be the same as for a splinter in the left big toe. Such workflow maps are typically not exhaustive descriptions of the individual jobs involved in them, either.

We could also map the flow of *materials* or specific items of *information*, without initial regard to how these are used. The latter types of flows are often useful for investigatory purposes such as process improvement, security, waste reduction, cost control, and the like; in some cases, we just want to see what happens to a piece of paper or a bit of information for one reason or another.

These categories are not necessarily mutually exclusive. It is entirely possible for a process to be contained within a single job. The main point to be aware of in considering types of workflow maps is that whether they are comprehensive of either jobs or workflows is a matter of how WFMA is applied. Many workflows only require a segment of the individual jobs they pass through, and conversely, many jobs contribute partially to several workflows. WFMA is flexible enough to accommodate all of these, but it is necessary for users to be clear about the content they want in a workflow map.

Flow relationships and flow diagrams. In Chapter 2, we discussed convergence and synchronicity in flow relationships. These distinctions in the nature and timing of workflow elements do not necessarily require separate types of maps, but these may be disaggregated into distinctive parts. If we are mapping the *process* of "cash management," meaning depositing and disbursing cash along with periodic reconciliation of the cash account, this will be several flows of activities with related exception-handling processes, much like any other. These would indicate that one person prepares and deposits cash receipts, and another will be the one to write checks and disburse cash. At certain dates or on certain random intervals, reviewers who are not part of this regular workflow will verify the overall transactions and the resulting balances (i.e., an audit). The overall process is continuous and cyclical, even though it is done by a group of individuals with

responsibility for discrete activities in the process. Divergence is a matter of where the activities that make up the workflow are done, and timing is a matter of when these activities get done; but all of them are part of one process—cash management.

Much of the purpose of WFMA is to describe, and in some cases to discover, exactly this kind of deeper understanding. The understanding begins with a *valid* map of what is actually being done, and this is sometimes a very challenging map to create, for reasons discussed above. Having this map, however, enables properties like the functional or dysfunctional nature of the workflow to be seen as well as matters of synchronization or convergence of actions and decisions.

Workflow Mapping versus Flowcharting

Most methods of charting or mapping workflows are derived from standard flowcharting symbols, as mine has been, but there are no truly standardized methods for mapping (Harris, 1999:155). While well-known flowcharting symbols are used to create workflow maps, workflow mapping is *not* flowcharting, although the outward similarity at first and the frequent use of that term outwardly might make it seem so. There are a number of important differences between WFMA and flowcharting as the latter is applied in programming and information technology environments:

1. Only a limited set of flowcharting symbols and rules are used for workflow mapping, and these are distinctive and easy to learn and apply. This is quite important for workflow maps to be useful to those not trained in flowcharting, and makes the symbols a means to the desired end, which is understanding the workflow. Most flowcharting symbols, in fact, are far too specialized for workflow mapping.
2. Workflow mapping works by capturing and describing an actual process used to accomplish a purpose, whatever steps may be involved. Flowcharting works from a logical program structure and follows rules similar to computer language. In that sense, workflow maps are *descriptive*, where flowcharts are *prescriptive*.

3. Workflow maps can vary enormously; it is unlikely that two different workflow mappers would create the same map for any purpose or process; therefore, many different versions of a workflow map can be "correct." Flowcharting will produce a "right" answer that can vary only to the extent the programming language allows. Unlike a workflow map, it is likely that two skilled programmers would solve a programming problem in relatively similar ways.

4. Levels of detail and presentation format can vary enormously in workflow maps, whereas flowcharts must provide sufficient detail to write code, and must be consistent with the rules and syntax the computer language requires. Workflow maps can and should be tailored to the level of detail most useful and informative to the user.

5. Workflow mapping is an iterative process of discovery and investigation, not of application of externally-determined coding rules; the objective of workflow mapping is to develop a valid map of a process. The objective of flowcharting is to develop an efficient, correct answer to a programming problem, and by definition that means it has to run.

Who Prepares the Map?

One last question often posed in WFMA concerns who should prepare the map. The best approach is to have the person who does the job prepare the map. Prior training in the symbol set and its use is strongly advised, but this is not a time-consuming requirement—having trained hundreds of people to create workflow maps, my experience is that 75 percent of trainees will learn to use the symbols correctly on the first or second try, and nearly everyone else gets it with a third try. This is especially likely if several non-threatening exercises are done as part of the training, and groups of trainees serve as "consultants" to each other. The "consulting" usually consists of simple tasks like helping people to remember to label exit paths from decisions, not mix diamonds and rectangles (a common problem at first), and putting arrowheads on arrows. Clarifying what is meant by the content of the diagram is also important—it is quite common for statements to be put in decision diamond when a question is actually needed, for example. I have provided several exercises at the end of the chapter to assist this part of the process, and it is easy to develop similar exercises that may have specific appeal or relevance to other groups.

While there may be some circumstances where the workflow map should be prepared by a supervisor or a specialist, my experience is that these are rare, and generally not beneficial. First, it takes a great deal of time and interaction between two people for the second one to understand and correctly describe the other's work. Second, there are many perceived reasons why the jobholder may not want to be fully forthcoming to someone else—lack of trust in that person, a feeling of being examined or watched closely (perhaps to be caught in mistakes), or a sense of pressure just to get done, from either or both people. Most importantly, the internalized tacit knowledge on the part of the jobholder may be harder to discover and understand if interpreted by another. These factors interact to make validation

harder rather than easier, because the onus of correct explanation of a flow of work has been shifted from the person who does it to the person mapping it.

A supervisor or internal consultant, on the other hand, should be involved in validation of the map. Inevitably, that which is crystal-clear to me when I describe it in familiar terms is absolute gibberish to someone else; steps that are assumed to be known from birth will be absolute unknowns to someone else; common language and acronyms that "everyone" knows will be misinterpreted; and so on. All of these issues must be resolved before a map is considered valid.

There are many situations where the objective of WFMA is to develop a map for a process that is performed by a group. An alternative to building the process map from individual ones may be for the group to do the map collectively. For WFMA projects like this, the common "sticky note" can be an engaging way to get everyone in the group involved. The common rectangular and square notes provide the two workhorse symbols (the process block and decision diamond); the other three can easily be drawn on these two. These can be put on a wall or whiteboard in a public area (the coffee pool) with an invitation to everyone to change or contribute to the diagram as one sees fit. Not only does this create a great deal of interest in the mapping process, it also creates "buy-in" for the final product.

As with any job, skilled specialists in WFMA diagraming will likely be able to prepare maps more quickly and easily than individual jobholders, and larger organizations may want to have one or two of these around. Specialists selected and trained for their interpersonal skills, for example, are often very effective at getting through the validation of a map with less stress than the supervisor or an outsider. They can take a first "rough cut" jobholder diagram and do the editing and revision these typically need very easily. As they learn the nature of work done in a group or unit, they can become very effective at asking good questions about exception handling and parts of the job where tacit knowledge is important. They can play a valuable role, but they take time and practice to develop. If specialists are used, my recommendation is that they become very good at workflow mapping and have excellent interpersonal skills.

Summary

This chapter, in many ways, is the major "nuts and bolts" of the WFMA method I have developed. By design, the method is simple and disciplined; however, "simple" is often misperceived as "simplistic," and nothing could be further from the truth—"simple" is "powerful," but only so long as the simplicity is preserved.

The idea behind this method is straightforward—both mappers and users can quickly learn the symbols and rules for their use, so that either can create, validate, modify, or question any map produced with this system. The maps can be consolidated and aggregated easily, whether in soft (electronic) or hard copy. Equally important, the maps can be changed quickly when processes change. As we will see in the next chapter, the maps create a basis for measurement of process outcomes, and can be linked to important company and organizational metrics. Finally, the method is highly robust, and can be applied to any organization, no matter what it does.

Exercises

Unlike other chapters in this book, it seemed that several practice exercises would be helpful in this one, since one of the principal objectives is to learn how to use the WFMA symbol set and its related discipline. I have included two groups of short exercises here to allow readers to apply what they have read in this chapter to different kinds of processes. The first two are exercises in doing personal things we all have to do, and which most of us probably do "on autopilot." But try to be as complete in your mapping of both processes as you can.

The second set are situations more typical of what we might find in many workplace and office settings. Both of these have objectives relevant to the success of the hypothetical company in which they are done, and are not as likely to be personalized as the first group.

A hint—in both exercise sets, try to account for the parts of the processes which are purely information as well as those which may be movement of physical materials, and don't be surprised if the information flows are bigger and more involved than those tracking the material.

Individual Exercise Group I

On a separate piece of paper, draw a workflow diagram of:

1. how you mow the lawn, or (for those who don't have one), how you clean your residence; and

2. how you do your laundry.

Individual Exercise Group II

On a separate piece of paper, diagram the workflows for the following two problems.

1. Your job is to periodically review customer credit card accounts in the bank where you work. Periodically, you retrieve accounts, review them, and must decide to either renew them, suspend them, or terminate them. If you suspend an account, you must change its status so that it should not be used, and then notify the customer of that change. If you terminate the account, you must notify merchants that the account is terminated, and then notify the customer. Naturally, if the account is renewed, you want to notify the customer of that, too.

2. You work in a mail-order catalog house where you receive orders from your customers by telephone. If a customer calls, you need to know if this is an existing customer with whom you have done business in the past; previous customers have a PIN. If so, you retrieve a customer number from your terminal, take the order and complete billing information, transmit the order to shipping, and then mail the documentation to the customer. If this is a new customer, and someone who is using a credit card, you must obtain the credit card data before completing the billing information. As with existing customers, you then transmit the order to shipping, and then mail the documentation to the customer. If the customer doesn't have a credit card (and this sometimes happens), he or she will have to go through a credit review process that you pass on to another department; if approved for credit, billing information will be completed there and the order finished as with any other customer.