Case Management: Contrasting Production vs. Adaptive

Keith D. Swenson

Fujitsu America, Inc & WfMC Chairman

1. Introduction

While participating in discussions of case management, and while reviewing the submissions to the Adaptive Case Management awards, I see two distinct approaches to case management -- one approach called **Adaptive Case Management** (ACM), and a different approach which meets an entirely different need which we should call **Production Case Management** (PCM). This chapter explains the difference, and how these fit into a spectrum of process technologies.

It is important to distinguish these two. If you need ACM, and you acquire PCM you will be very disappointed, and vice versa. Both are useful in relatively unpredictable situations, but there is one huge distinction that makes all the difference: PCM requires and can benefit from a software developer, while ACM must not. Put another way, PCM must separate the development time from the run time, while ACM must unify these. This may not seem like a big difference, but it affects the very root of what knowledge workers can do, and how they achieve results.

2. PCM In A Nutshell

Production Case Management (PCM) is an approach to supporting knowledge workers which is programmed by specially-trained technical people (programmers) to produce a case management application. That application is deployed for use by knowledge workers to get their work done. The application offers collections of operations that the knowledge worker can select to use or not use depending on the specific needs of the case.

A PCM application is used when there is a certain amount of unpredictability in the work, and a healthy amount of flexibility is needed, but necessary actions are regular enough, or the volume of work large enough, to make identifying and codifying regular patterns valuable. A worker using PCM will be involved in selecting the actions toward the outcome of a particular case, but will not be responsible for the kinds of actions that might be available.

3. ACM In A Nutshell

Adaptive Case Management (ACM) is an approach to support knowledge workers who need the most flexibility to handle their cases. ACM allows the knowledge workers themselves to create and modify all aspects of a case at any time. There is no distinction between design time and run time: the designing and running are done at the same time by the same people.

This approach is used by knowledge workers who have unique expertise in an area. They don't have a lot of time to transfer this specific knowledge to a programmer, and it is too expensive to hire a programmer for one-off cases. ACM offers a Do-It-Yourself (DIY) approach to process programming. The worker using ACM is responsible not only for the outcome of a case, but also for how the handling of that kind of case improves over time.

4. SEVEN DOMAINS OF PREDICTABILITY

Process technology can be spread across a spectrum to support work with varying degrees of predictability. At one end you have entirely predictable processes which need to be done exactly the same way every time, and that way of doing it will be the same over the course of many years. The other end of the spectrum is complete unpredictability where there is no way to know from moment to moment what will have to be done next.

Predictability and repeatability go hand-in-hand. Any work which is repeated the same way thousands of times, is predictable by definition. Work that is not done the same way every time, that is frequently repeated, is consequently less predictable.

The approach to developing any system will depend on how much change the system will have to respond to over time. Extremely predictable, stable environments can benefit from powerful but inflexible approaches. As the anticipated amount of change rises, it becomes more important to use a technique which offers greater flexibility. More flexible approaches have less precision to exactly match the needs of the situation. The approach depends entirely only on the amount of need to respond to change.

Most job situations lie between the extremes of completely predictable/repeatable and not predictable/repeatable. We can break the field into seven domains according to the technology that might be used to support workers:

1. Traditional Application Programming—If work is very predictable and stable over time, one can use traditional development techniques (e.g. using any third generation language like Java, VB, PHP, etc.) to create a supporting application. The cost of development might be high, but the benefit of having very precise control of the capabilities, will yield efficiencies that over a large number of cases will pay back the up-front costs.

2. Process-Driven Server Integration (PDSI)—Integration patterns between systems can be quite stable in the short term: months or years. Still the systems being integrated do change, and the business needs change as well. PDSI usually incorporates a development approach where the key interaction points are depicted on a process diagram is a technique that helps deal with the kinds of change that IT departments experience.

3. Human Process Management (HPM)—Human business processes involving forms routed through a set of people, may also be well-understood routine processes for people in the short term that can be fully designed, but like PDSI there are changes in the environment, and having a process diagram as the backbone of this application helps cope with this kind of change. Furthermore, humans have greater need for flexibility than servers when they are part of a business process. For example, humans change duties much more frequently than servers, and are not identical replaceable units. Thus HPM has higher requirements for coping with change, and capabilities for addressing this.

4. Production Case Management—PCM is designed to handle situations where there is so much variation between individual cases that it is not possible to set out a single fixed process, and yet there is still a well known set of actions that can be taken. The knowledge worker is actively involved in deciding the course of events for a case, but the range of actions and options is bounded and can be specified in advance.

5. Adaptive Case Management—Cases vary so much that knowledge workers are constantly striving for innovative approaches to meet the needs of new cases. The knowledge worker is involved not only in the case, and picking a predefined action, but is actually helping invent the actions that can be taken. However, there is still enough predictability that a given knowledge worker may want to re-use a process from before, and may want to share and discuss process plans with others.

6. Social Business Software (SBS)—This is collaborative software without a fixed plan. There may be representations of tasks, but they are strictly created on the fly and discarded after use.

7. Email, Telephone, Texting—There is no process at all, no permanent structures, simply communication. This is the default that many current processes are forced to use, but this approach puts the greatest burden on the user, and yields the least amount of analytic data to monitor and improve processes.

It should be immediately apparent that each domain of predictability is distinguished by the amount of investment that must be made up front—the amount of preparation that must be done before you start working. Traditional programming requires a large development project and is useful only once the entire project is completed, and the fully-tested software is installed. At the other end of the spectrum is email, or phone calls, which need no preparation at all, and can be used immediately without delay. PCM and ACM lie between these extremes; both require a certain amount of upfront investment, but still leaves quite a bit of flexibility to the case manager.

If there is investment up front, then dealing with change means some kind of cycle of improvement. PDSI and HPM have fully defined processes which are regularly updated in what we call the BPM lifecycle. PCM has a less fully-defined process model; the case manager picks from a menu of actions in order to meet the needs of the case. The possible actions are predefined and can change only with the involvement of a developer. ACM takes this a step further by allowing the case manager to not only choose the actions, but to create entirely new actions on the fly as needed. You can think of all of these as having a cycle of improvement that get progressively smaller and faster across the spectrum, and when we get all the way to ACM, the entire improvement cycle is performed by a knowledge worker, making the cycle of improvement fast and effective.

5. WHAT IS ADAPTIVE?

To understand the difference between PCM and ACM, you need to understand what we mean by *adaptive*. Whenever you hear about an adaptive system, you should think about muscles. If you want to increase the size or strength of a muscle, you exercise it. The use of a muscle triggers a response to build the muscle. Conversely, lack of use causes muscle atrophy.

Adaptiveness is not simply the capability to increase or decrease muscle size. Instead it is more about the ability of the muscle to self-modify to fit the situation; the ability to sense a need, and to respond to it in a kind of feedback loop. Organizations are naturally adaptive if you have experienced people in management and they are getting accurate information about the situation.

Homeostasis is the idea that an adaptive system responds to external changes in such as way as to keep certain aspects constant. Your body maintains a constant temperature by various mechanisms that respond to temperature change. A retail store that can detect the increasing popularity of an item will order larger quantities in order to keep the item available for sale.

6. EXAMPLES: ADAPTIVE SYSTEM

Human Body—Not just the human body, but all life forms have aspects of adaptiveness. Your DNA specifies how to form all the various muscles, but it does not include complete specifics on size. Instead a feedback loop is used to find the optimal size. There is no need to predict ahead of time exactly the amount of muscle needed. Each muscle is built with the ability to measure the amount of use and respond by growing, or shrinking, appropriately. This simple mechanism eliminates any necessity to predict up front precisely how much is needed.

Adaptive systems optimize themselves. Take, for example, someone who suffers the tragedy of losing the use of their legs. The muscles that are no longer used will reduce in size, while the muscles in the arms will increase to accommodate the increased use.

There are many such systems in the body. The skin responds to light exposure by varying the amount of melanin at the points that received the exposure. This saves one from having to figure out in advance which parts of the skin should be more and less pigmented. Body temperature is maintained at a homeostatic constant through a number of mechanisms including sweat glands and shivering.

The concept of "practice" pertains exclusively to adaptive mechanisms. Want to learn the piano? Then sit down at the keys and practice, practice, practice. Want to learn to play tennis? Start playing hitting the ball and practice the right moves. Practice only works because the system is adaptive.

The Brain—this complex adaptive system allows a child to learn the language that they hear spoken around them by trying and practicing to improve skill. We study subjects in order to learn them. The concept of "learning" is again something that refers only to an adaptive system.

Ecosystems—The diversity of different organisms that thrive in differing conditions form an adaptive network, each organism dominating different aspects of the ecosystem when the conditions permit. The forest as a whole is extremely robust due to the adaptive nature of the biodiversity.

7. ENTERPRISE AS AN ADAPTIVE SYSTEM

Human organizations have always been naturally adaptive. The day-to-day decisions are decentralized and delegated to front-line workers. Different divisions compete for scarce resources, and good management will shift resources as needed. There may be a centralized view and control at a very high level, but generally this is very much abstracted away from the details of day-to-day operations. Various parts of the organization are sensing and responding to their situation. There is a nested, recursive aspect of this, so that as you get to smaller parts of the organization, the sensing and responding are more finely tuned and detailed.

Organizations are constantly changing and responding to that change. When a person leaves a position, the jobs of dozens of others will change. When an individual is promoted, many people will change their own behavior in response.

Yet the organization is stable. Adaptiveness does not cause constant fluctuation in the organization as a whole. In fact, it is well known that it is incredibly difficult to change an organization once it is in place. Adaptiveness presents a kind of homeostasis that allows an organization to keeps its character and form over the years even though people within the organization are constantly coming and going.

An adaptive system is one that effectively senses what is needed, and automatically responds. The knowledge worker is part of that sensing and responding. Without the ability for the knowledge worker to self-modify the system, to adapt to the situation, then the ability to sense and respond at a system level is lost.

Adaptive systems have evolved to conquer complexity. System thinking is the approach to try to understand how things influence one another within a whole. System theory attempts to understand self-regulating systems, which is achieved through some form of feedback.

When we talk about a case management system being adaptive, the complete system includes the case managers as well. Humans are not excluded from the feedback loop. We talk about a good ACM system facilitating what the professional wants (needs) to do. Professionals (case managers) play active roles in adapting the system to their needs. We can think of this as being self-modification because there is no need for a software developer or process analyst: the professional can adapt the system as needed to meet the constantly changing requirements. For example, when a doctor gets the idea for a new treatment plan, they can institute that new plan without involving a software expert.

There is a feedback loop within a PCM system but it is slower, requiring many weeks or months to get a new release out. This much slower cycle means that a PCM system cannot adapt as quickly as an ACM system.

8. COMPARE AND CONTRAST

There are some strong similarities between PCM and ACM:

1) At run time the most important concept is that of a "case" which is primarily a folder to collect all the information around the case, accepting essentially any format of document
2) Knowledge workers use their own expertise to control the advancement of the case from state to state.
3) The resulting case folder represents a system of record for the work that was done.

The biggest distinction between PCM and ACM is that PCM is not adaptive, and this can be seen by three factors:

1) The programmers uses formalism such as modeling or programming to put the application together
2) It uses a standard application development lifecycle: the application is constructed, then tested, then deployed to non-programmers
3) After deployment there is limited ability for the case manager to alter the structure of the application itself.

ACM is used for what I would call innovating knowledge workers: inventors, creative people, executives, managers, innovators, business entrepreneurs, media producers, doctors, lawyers, etc. These are people who really do need to decide their course of action every day, and the course of action might be to do things that have never been done before. A board of directors does not have a menu of options to pick from when it comes to actions to take. Someone responsible for the merger of two companies will not have a system with all possible actions pre-programmed. A doctor responsible for the survival of a patient may prescribe radical and untested treatment if it seems like the only option.

PCM is for environments where the number of knowledge workers is high, and the courses of action on a given case are sufficiently predictable to justify the cost of developing a dedicated application, though not entirely predictable. . Also, those knowledge workers are less responsible for evolution of work to fit new contingencies. For example, while a doctor might be in a position to prescribe a radical treatment, there are many others who work in a health care facility who should not have that flexibility. The routine care of a patient may still be too unpredictable for a fixed process, may still require the judgment of a nurse or clinician, and still the options available may be restricted to a set of known actions.

9. WHEN TO USE PCM

PCM is used when the number of knowledge workers doing the same job is large, and the domain relatively well known, but the process is not entirely predictable. Because the PCM application is developed by programmers, it can make use of more traditional mechanisms for data integration: structured information can be read from some sources, transformed, and written to other destinations. The sources and destinations can be web services or applications with an API. Like a typical development model, once the application is coded, the design rationale behind a particular transformation is not included in the final produced code, because it is not needed.

There are a lot of service businesses which can make use of PCM. You might use PCM for telecom repairmen. These people need to visit the site, determine what the problem is, and then prescribe a resolution from a menu of well-known operations. It is hard to represent what these people do as a traditional HPM process because it is not predictable enough for a pre-defined fixed process. The process unfolds at run time because the first resolution might not work, and that tells the repairman more about the situation, possibly leading to further action. Yet the repairman is not in a position to invent entirely new procedures. The phone/TV system is big and complex, and therefore the repairman's options are necessarily restricted to operations that are well-known not to cause a problem with the operation of the network.

Another example is auto service: the car is brought in, there is a set of things to examine. There are decisions about what to repair or replace. Maybe parts have to be ordered. Maybe components need to be sent to a shop for specialized repair. Perhaps the car needs to stay the night. With luck it all ends with the car being driven home.

PCM is most useful when the number of nearly identical offices is large, and the knowledge worker is a professional but not necessarily an owner of the process, and the process itself is not predictable enough to specify every step in advance.

In this year's ACM Awards, three submissions were awarded a Special Honorary Award for Production Case Management. The case studies were interesting and in their way solved important problems, but they got low marks based on the ACM awarding criteria. It is not that the cases were bad, or that the systems were poorly implemented, but simply that the cases were not "adaptive." We felt that readers would still benefit from these cases, and include them in this book in order to highlight the differences between Production and Adaptive Case Management. The cases are:

- JM Family Enterprise—an interesting interactive records management system that allows flexibility with less prescriptive processes, providing automation of support and reporting processes. It showed an advantage over

BPM for customer support scenario, but no real need to customize on the fly.

- Touchstone Health—a system that can handle a large number (1500) of appeals and grievance cases per month, and showed a documented savings over their earlier, manual system. The high number of cases and inability to customize individual cases both point to PCM.
- New York State Department of Financial Services—another high volume case management system with a very interesting, documented 40 perecent reduction in processing time for cases. Like most high volume situations, there is little support for, or need for, customization of individual cases.

All three were quantitative examples of how automation can benefit an organization, but the focus was on automation of the interactions for a large number of workers. Even though a developer was needed to prepare the applications, the cost of this overhead is less than the benefits conferred by efficiencies when the number of cases and the number of workers is high.

10. CAN ONE SYSTEM BE BOTH?

The smart money is betting that in the next year, many systems will be presented as supporting both ACM and PCM. Is this possible? Can a vehicle be both a car and an airplane? Yes it can, but it is neither a good car, nor a good airplane. Can a vehicle be both a car and a boat? Yes it can, but it will be neither a good car nor a good boat. Because cars, boats, and airplanes have distinct needs and requirements, there is no chance to meet perfectly multiple needs at the same time.

To be adaptive, the system must be programmed by the knowledge worker, which necessarily means that traditional programming skill must not be required. Involving a developer would be a barrier to getting the job done.

The Do-It-Yourself (DIY) aspect of ACM puts some significant constraints on capabilities. Consider other DIY products you might see in a store. They must be designed to be put together with no particular skill. An amateur DIY kit will always be more limited than the parts that a professional will use. The "professional" products that require a professional for installation can be higher quality for the same price, because they don't need to be designed to be fool-proof.

There are amateur (DIY) kits, and there are professional kits, but one kit is never *both* amateur and professional. A real artist will never use a paint-by-number kit, and a paint-by-number kit will never produce real art.

I have argued that using a two-dimensional graphical process diagram of any sort (i.e. BPMN) is a tool that professionals can use. However, such a language does not provide the kind of fool-proofness that a DIY process system needs. It is easy to drop a few shapes on the canvas and have something that is *syntactically incorrect*. The syntax rules are hidden, and must be learned independently. The knowledge worker is too busy with their profession to learn the intricate details of the syntax of BPMN.

For an adaptive system, the right format for a process for a knowledge worker is a simple list of goals. Anyone can add a goal to the list, and there is no possibility to invalidate the syntax. There are no hidden rules to violate.

It seems overly glorified to talk about a simple list of goals as a *process*. However, this simple approach allows a case manager to complete cases effectively, and repeatedly in the face of changing demands of the situation.

A programmer, designing a PCM process for an organization, will tend to want to use many sophisticated capabilities to do things, such as determining when a task is complete. Or in hiding particular potential tasks until prerequisite tasks have been completed. All of these are effective in making the PCM application effective in the organization and, because PCM is used in high-volume situations, any small increase in effectiveness is multiplied by the number of cases.

Can't you have both? Have the programmer make the basic process, and then allow the case manager to modify it? This is not possible because in the act of designing a program, the programmer makes many assumptions which are not apparent in the resulting program. It is dangerous to make even a small change in such a program without careful study of all the assumption behind the program.

While a BPMN diagram is powerful for a process expert to use, it does not make clear the underlying assumptions that went into that program, nor does it provide a way to safeguard those important assumptions, while allowing other things to be changed in controllable ways. Consider process variables: they are not visible in the diagram, nor is it clear exactly all the ways that a process variable is used. Changing the way a variable is manipulated in one part of the diagram might have dangerous consequences somewhere else. Only after careful understanding of the complete diagram can one make a safe change to it, and such effort is not something that a case manager can put into it.

My conclusion, which undoubtedly will be debated, is that if you offer a powerful language for a process expert to make a PCM application, you will eliminate the adaptive capability, and it cannot simultaneously be an ACM system. The two kinds of technology are distinct.

11. REFLECTION

Two approaches to knowledge work imply that there are two kinds of knowledge workers distinguished by “responsibility.”

Knowledge worker for hire—someone is trained in a specific field, and learns to be an expert, but has little or no ownership of the overall process. A car mechanic must make accurate suggestions on how to repair the car, but does not take responsibility for the repair shop business, and must work within the constraints set by others.

Knowledge worker with responsibility—someone who can plan and be responsible for the course of events. This is defined by Peter Drucker as someone “knowing more about their job than anyone else in the organization.” These are the workers who handle the wicked problems and have to think outside of the box, e.g. a board member or someone responsible for mergers and acquisitions.

I am sure there are many more distinctions between types of workers, but for now this seems to be a determinant for whether you use PCM or ACM.

12. SUMMARY

At least two distinct forms of case management have been identified: ACM and PCM. They are similar: both having case folders, goals, and case history. But PCM appears as an application designed and implemented by a programmer and deployed to a large number of users who face very similar situations. ACM is for less predictable work situations where knowledge workers need a greater amount of flexibility to alter any aspect of the case at any time. The primary difference is that PCM needs the support of a software developer, while ACM requires case managers to do it themselves without specialized training.