Position: BPMN is incompatible with ACM

Keith D Swenson

Fujitsu America, 1250 E Arques Ave, Sunnyvale, California
kswenson@us.fujitsu.com

Abstract. The role of two-dimensional process graphing in Adaptive Case Management (ACM) is examined. Three design criteria are identified for ACM that were never considered for BPMN. The question for discussion is whether these requirements eliminate all kinds of flow-chart type languages from consideration for use as a process modeling language for users of ACM.

1 Introduction

Within our current technology setting, essentially all discussions of business process make the implicit assumption that Business Process Modeling Notation (BPMN) will be the graphical language for expressing the business process. For the purpose of discussion, consider this decidedly radical proposition: "Any work support system that depends upon processes designed with BPMN (or BPMN-like languages) cannot be considered an ACM system."

This statement is intentionally bold in order to question our basic assumptions about process modeling for ACM where all modeling is done by the end user: the knowledge worker. Is it reasonable to expect case managers to ever have the skills to use a two dimensional BPMN-like process language? If not, does this lack of skill in itself become a barrier to effective use? The question is not how BPMN might have to be changed to suit case managers. What this is proposing is that whenever a system depends upon a two-dimensional design format for describing the processes, that dependence itself makes it unsuited for use as an ACM System (ACMS). The proposition is that an ACMS absolutely must not have BPMN, in order to be considered an ACMS where planning must be performed by end users while working.

2 Setting

In the early days of business processes management – we called it business process *reengineering* back then – I published a number of papers about graphical business process definition languages which were designed to be used by business people directly[1,2]. This visual language is surprisingly similar to BPMN with some superficial differences. The main elements were tasks, represented as ellipses instead of the rounded rectangles that BPMN uses today. Transitions between activities were arrows. Specific nodes performed branching and joining. Start and end nodes were hexagons instead of circles in BPMN. Most notably, events were represented as small circle on the edge of an activity. Those diagrams can be isomorphically translated into a BPMN diagram today, with equivalent readability / expressibility.

At Fujitsu, I was able to produce a system that supported the graphical definition of business processes, and the enactment of them.[3] The most important aspect of that system was the ability to change the process at any time by anybody. Any business user of the system can retrieve and edit a business process at any point in time. Simply point the browser at the server, to retrieve your task list, and if desired you can also edit any process as well.

We found over 15 years that business users essentially *never* design a graphical business process. Processes are always designed by some sort of developer. Those process enthusiasts who train themselves on process technology do not represent the typical knowledge worker. Even managers who design detailed interaction patterns for their team, will rarely actually draw a picture of the process themselves. Instead they hire process specialists who will draw the diagrams for them. Even office workers who are comfortable reading a process diagrams and who use them for training about a given process will rarely actually draw the process.

My conclusions (given here without any evidence) are that:

- Drawing a diagram requires a kind of abstract thinking about the process that a business user is not comfortable with. Instead, they want to define activities in terms of responsibilities on people, and not in terms of a flow of a token through a series of tasks.
- Drawing a diagram actually involves some programmer-like skills. For example, a branch node requires variables, which need to be set up in advance so that the branch condition can test those values. Separating the information into discrete units in a format that can be tested is a developer skill unfamiliar to business users.
- Modifying someone else's diagram is particularly difficult because all of the assumptions that went into drawing the diagram are not present in the diagram. A simple example is that one node may initialize a variable, and another may use that variable, so switching the order of these nodes would break the functioning of the diagram. There is no way to indicate in the diagram might or might not be safe. Like most programming languages, it is designed to embody a set of assumptions, and not to necessarily express those assumptions. We can say that the original rules and assumptions are *hidden* by the resulting process diagram.

3 Relevance to ACM

A graphical process definition plays different roles in different styles of process technology. For example, in Process Driven Server Integration (PDSI) – a name for the kind of BPM which is commonly associated with a Service Oriented Architecture (SOA) environment – the process diagram is part of the programming process, or it may simply be in the design spec and completely replaced by the implementation. This programming and designing of the server integration is done by programmers, and not business users, so use of BPMN is not impeded.

For Human Process Management (HPM) – a type of BPM targeted at modeling routine human activities – it is a process specialist, or possibly a programmer, who designs a process which is used by the process participants. Typically in HPM the end user does not have to design processes.

The same is true for Production Case Management (PCM) where once again you have specialists who define the interaction patterns, which are developed and deployed as a finished application to the production users.

ACM however is different from these all of these because there is no distinct design phase. Designing and performing the work are done at the same time by the end user. There is no distinction between the users and the developers. The business users themselves must design the process. Even professionally designed templates must be modifiable by the case managers to fit the needs of each case, and so must not have hidden assumptions.

4 Use of the Language

BPMN-like languages were originally designed for an expert to express something very precisely. The expression will take into account many understood rules, but may not express those rules directly. The developer says *what* to do, and not *why* to do it.

This can be a large investment of time. If the process is executed many times, then a large up-front cost of developing a process can be compensated by a modest increased efficiency of the process. The investment that most styles of process technology (PDSI, HPM, PCM) require is not a particular problem.

For ACM the process is designed by the case manager as they do the work, and usually just for the benefit of that one case. If things work out well, that process may become a template and reused many times, but each case manager must justify the effort of creating the diagram in terms of the case they are currently working on. From this see criteria 1:

- Ability to design a basic process quickly with very little investment by the user is far more important than the ability to define a precise process which uses more time an attention from the business user.

In ACM, process change is an everyday activity. If the ACMS were to require skill in BPMN to change the process, then the case managers without that skill will be prevented from using ACMS as intended. This brings us to design criteria 2:

- Process design must not require a skill beyond what business users possess.

Even if the business user is an expert in BPMN, they still may find it difficult to change the process because of the hidden assumptions problem. BPMN-like languages are designed to express a final process, and not all the steps and decisions along the way to develop the process. This brings us to the design criteria 3:

- For a process definition to be modified by business users there must be no hidden assumptions.

The third criteria deserves means either that the process diagram must include a lot of additional information to express all the decisions behind the way the process is designed, or it means that there is a limitation on how complicated the process can be.

5 A Process Language Designed for Change

A case manager needs to be able to change a process quickly and effectively every time without error. This means that all possible changes need to be valid changes. If the case manager needs to switch the order of two activities for a good reason, there should be no possibility that such a switch can cause a failure of the process to execute. A language that is designed for change will allow (almost) any change, and will prompt for the resolution of any problems that arise because of the change.

Gödel’s Theorem might imply that it is impossible for a developer to express all of the reasons for designing a process in a particular way. Even if it is possible, we can be certain that it is tedious and far beyond the skills of a business user.

For this reason, I think that the right process definition language for ACM will be one that appears, compared to BPMN, to be very simple. It will appear to be very loose, flexible, and unstructured. However, it will be such that every business user can read and understand without special skills, and it will be one that can be changed in any way, without causing inconsistency. Some will question the utility of such an approach, but this is a case of “less is more”.

6 Summary

The goal here is not to state any conclusions in this paper, only to raise questions for discussion. I hope that the results of the discussion can produce a set of actions items that can resolve the issue. What experiment or measurement is necessary to determine if flowcharts can be effectively used by business users? What experiment would show that checklists function more effectively? How can we determine the likelihood that business users will learn enough formal process modeling skills to be effective at ACM?

7 References

1. Swenson Keith D.: A Visual Language to Describe Collaborative Work, Proceedings of the International Workshop for Visual Languages (1993)
2. Swenson, Keith D.: Visual Support for Reengineering Work Processes, Proceedings of the Conference on Organizational Computing Systems (1993)
3. Swenson, Keith D., et.al.: A business process environment supporting collaborative planning, Journal of Collaborative Computing, Chapman & Hall (1994)
4. Swenson, Keith D.: Workflow for the Information Worker. Workflow Handbook 2001, Layna Fisher (ed), Future Strategies (2001)