

Event-based Workflow and the Management Interface

Jeffrey V. Nickerson
Stevens Institute of Technology
jnickerson@stevens-tech.edu

Abstract

Long transactions cause pragmatic problems for workflow systems – as the transaction is moving, so is the surrounding world. We look at three scenarios in which external events affect the performance and public perception of a system. Then we discuss management user interfaces. First, we consider an intervention interface. Second, we argue that a context monitor, manned by humans rather than machines, can provide a way for the overall system to be more responsive to what is happening in the surrounding world, by making use of our cognitive capacity to grasp and react to unforeseen events. Third, we propose a planning interface, which can be to design how managers will respond to certain foreseeable situations. The relationship of these interfaces to the overall problem of enterprise integration is described, and some generalizations are made about how we interact with software.

1. Introduction

Different from most software systems, workflow involves long transactions that can take place over days or weeks. These long transactions present a challenge – while the transaction is taking place, assumptions made early in the processing become untrue as the world around the system changes.

In discussing this problem we use the convention of a sequence diagram, which shows message communication between major actors and programs across a horizontal dimension, and shows the sequence of this communication using the vertical dimension [1]. A more common representation used in workflow papers is the Petri Net, which, as Wil van der Aalst points out, can show decision points as well as sequence [2]. For us, the sequence diagram more clearly shows the distinction between human actors and machine processes, a distinction that is central to our concern.

In understanding sequence diagrams, it is useful to imagine a situation in which we model all people or computer processes that could possibly be involved in the

scenario. To do so will result in a very long, horizontal diagram, figure 1, in which much of the communication is happening in parallel. From this diagram, we extract the interactions between the smaller number of people and programs that constitute the system we are analyzing, shown boxed on the right.

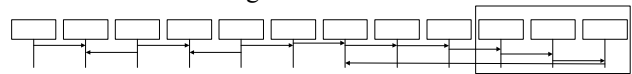


Figure 1.

The rest of this diagram represents what is external to the system – the system context. Some of the boxes talk directly to the system – these represent the people and systems that would show up on a conventional context diagram. But most of the other boxes do not, and are never discussed in a systems analysis. We are interested in situations where the surrounding actions, the boxes not directly attached to the system, do effect what should happen within the system. We present several scenarios, inspired by recent public events, which pose problems to the typical workflow system.

2. The First Scenario

We look first at a scenario in which an outside event, if noticed, would make the internal flow of a case futile.

Individual A applies to the US Immigration and Naturalization Services for permission to study under a student visa. The application takes many months to process, and Individual A dies during this time period.

Common sense suggests that the application should not continue to go through the approval process – that it makes no sense to grant a dead person a visa. As we consider the scenario, it becomes clear that it may not be easy to implement common sense. In many workflow systems, there is a set of preconditions before a case is started, and once these preconditions have been met, there is no going backwards to reconsider them. So if the applicant was alive at the beginning of the case, there would be no further check.

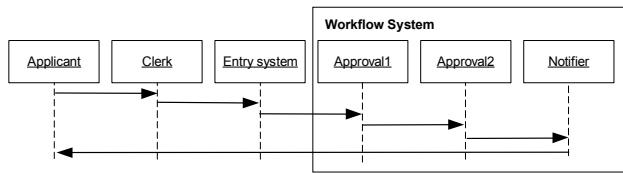


Figure 2.

To make matters more complicated, workflow systems are often separated from the original entry of the information— the data entry systems, which are specific to a specific application domain, often submit the information into the system, and the workflow system takes over to manage the process. This differentiation of process management from application domain knowledge is considered a virtue of the workflow approach.

An applicant seeking approval may be seen as external to the workflow system. In figure 2, the applicant is three steps removed from the system. The applicant presented identification to a clerk, who keyed or scanned information into a data entry system, which initiated the workflow case. The person who applied was obviously alive. And the assumption is that the person continues to be alive. The assumption is really more general – that whatever was true at the beginning holds for the rest of the processing.

In order to come up with a design that would handle this scenario, we can posit a database that contains all the information on the initial conditions of the case. We will discuss later why the creation of this database is itself problematic – for now, we assume it is possible.

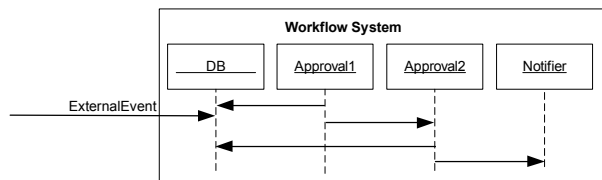


Figure 3.

At the end of the approval chain, right before the application is approved or rejected, the preconditions can be rechecked, as shown in figure 3. This technique is analogous to concurrency techniques used in data caching – before the cached information is written back, a check is made to see if something changed in the meantime [3].

While this technique is simple, it is not fully satisfying. For if some precondition is violated early on in a lengthy approval process, the rest of the processing will be in vain. Our common sense instinct is that the system should stop processing the application as soon as the violation becomes known.

In order to do this, the system should have the capacity to respond to events that happened outside the normal set

of process management activities. We can build into the system a technical mechanism for distributing events, and we address this now, following with a discussion of the problem of discovering the events in the first place.

Event-based technologies such as publish/subscribe are designed to handle arbitrary events coming from outside of the primary system, so appear to be appropriate in a design to solve this scenario. For different reasons, other researchers have reached the conclusion that event-based infrastructure should underpin workflow management systems. Two recent papers describe workflow systems that utilize the publish/subscribe paradigm [4, 5], originally discussed in [6]. Publish/subscribe technology provides a method for applications to publish messages that are received only by those who actively subscribe to a category of messages.

Figure 4 shows how publish/subscribe looks in a sequence diagram. The broker takes subscription requests, and on receipt of a new published event, distributes the events to all subscribing processes.

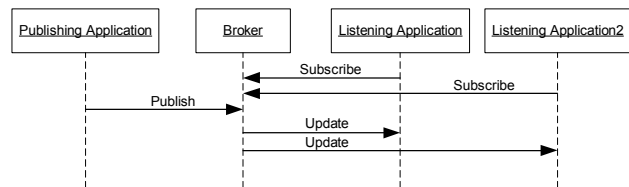


Figure 4.

The mechanism of publish/subscribe can be used to create loosely-coupled system – the publisher does not need to know who is listening, and the listener does not need to know the physical location of the publishing source. While these architectures are normally used within an enterprise, there are efforts to extend the paradigm across organizations, and there is a new standard which will encourage this [7] [8].

Publish/subscribe is a facilitating infrastructure technology that can make possible the integration of external events from a technical perspective. The larger problem is noticing the events in the first place.

We look now at how an event such as a death can be discovered. It is unlikely that there is a field in the workflow-related database which indicates whether a particular individual is alive or dead. If there were such a field, it is unlikely that it would be filled in at the start of a workflow instance – the assumption would be that the person was alive. We might imagine a system which monitors for death events. Deaths of individuals are reflected in death certificates, which are stored in civil databases. However, in the case of a visa application, the applicant is likely to be from another country, so a death is most likely recorded in a foreign database.

In other words, finding the source database of an event such as a death is in itself a difficult problem.

For the moment, we will assume that a database event can be found, and that an event publisher can be constructed that on the death of an individual, publishes that message to whoever is interested. Then there must a listener for the event. Ideally, we would probably opt for a system which automatically detected external events and followed a set of rules for these events. If an individual has died while their case is active, then the system would receive the event and automatically divert the case.

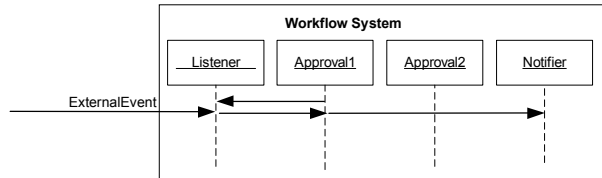


Figure 5.

Yet, in order to accomplish this, we would have to anticipate that this particular event might happen, and specifically program a response. This is certainly possible – now that the scenario has been explained, an analyst could see how to design against the scenario. Figure 5 shows a generalization, in which an approver subscribes to events from a listener, and then is interrupted if the event occurs, forwarding the uncompleted case to notifier. But it is unlikely that an analyst would have anticipated the event we have been describing. For what we are doing is considering how to respond when the presupposition for an implicit precondition of a transaction has proven false. The implicit precondition is that the person we are approving is alive at the time of application. The presupposition is that the person stays alive through the length of the long transaction.

Even supposing our analyst really did zero on this particular problem, there are a myriad of other similar problems. An applicant can have applied multiple times, or apply simultaneously in multiple countries, or be arrested, or be drafted into the military. Those familiar with artificial intelligence literature will recognize this problem as being related to the frame problem [9, 10]. Whenever we find ourselves in the situation of having to enumerate all the possible ways the world can change, we are in trouble. This is especially true if we need to invoke common-sense reasoning (such as dead people don't need visas) as part of the process. The complexity of representing common-sense knowledge has kept AI researchers busy for decades [11], and is still a research problem. More specifically, there appears to be no formal method for enumerating a complete set of external events that might have a substantial impact on a workflow in process.

At this point in the analysis, we have identified that publish/subscribe technology could possibly help, but we are discouraged by how complex it would be to create a solution for what appears to be a simple common-sense problem. The second scenario, by making circumstances more drastic, will suggest another approach.

3. The Second Scenario

Individual A applies to the US Immigration and Naturalization Services for permission to study under a student visa. The application takes many months to process, and Individual A dies in a very public manner during this time period – reports of A.s death are discussed in international media for months.

In the first scenario, a common sense approach said that the event of someone's death should be noticed, and the application not approved. Yet a system that failed to catch the death event probably wouldn't be perceived as seriously flawed. For this second scenario, any failure to catch the death will be seen as a major flaw, as the publicity over A.s death will be assumed to be known to anyone touching the record of A.

In response to public indignation, the argument might be made that the system was fully automated. Yet that argument leads to the question of why an automated system took months to process an application

Our perception is that people are aware of their context, and pay attention to the news. We may understand that computers do not comprehend the news, but we also believe that computers work quickly. Our expectation is that a system with long transactions will include people who will respond to events in the outside world that effect cases currently being processed.

At the end of the first scenario, we faced the discouraging conclusion that it might be very difficult to solve such a scenario in a general way. For even if we could look for death events, we might forget to look for arrest events or passport-losing events or a large number of other events that an individual possessing common sense would unconsciously presuppose about the visa application process.

In this second scenario, there is another way to handle it – we rely on the humans in the system to intervene. In other words, in the case of a very public event, we might expect a manager in the INS to actively locate cases related to public events, and make conscious decisions about those cases. In the same way as we expect a corporation to respond quickly to public events that affect its customers, we expect a government agency to do the same, if for no other reason than to show they are in control of their systems. This intervention can be accomplished without much technology, but it requires

process. Someone needs to have the job of paying attention to the world. This person needs a way to intervene, which might be as simple as an interface to find and change records.

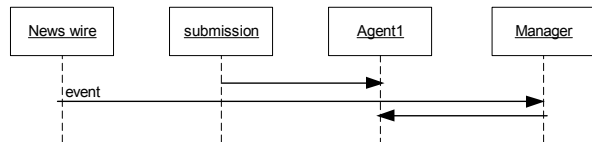


Figure 6.

The advantage of this design, as shown in figure 6, is that it can be accomplished through a process guideline - that managers intervene when outside events have implications for the systems they are in charge of. Even without an explicit real-time feed, a manager who read the morning paper could look to see if a high-profile person has an active case in the system. The intervention process can be further decomposed. First, there is the issue of assignment - some managers might have expertise surrounding only certain types of events. Second, once an event is detected, information might need to be verified. For example, one reviewer observed that events detected through the press might be either incorrect or ambiguous. Third, intervention might need authorization or review, as the power of intervention might be abused to either deny a valid request or approve an invalid one.

All systems implicitly permit intervention - but most do not explicitly recognize it as an interface. We are arguing here that intervention be an integrated, explicit aspect of workflow.

Technically, the publish/subscribe mechanism we discussed in relation to the first scenario will also serve in this scenario. The bus takes advantage of our human ability to understand messages from surrounding contexts. A publish/subscribe system in which the subscriber is a human can work without an explicit ontological description of each possible event - instead, it might be enough that a new piece of information on A. has updated another federal database - then it is a human's turn to figure out what to do with it.

The human ability to understand contextual information and assess its importance is very high, and the architecture we are suggesting here takes advantage of the human's ability to interpret. There is a tradeoff. To fully automate a system that would respond to arbitrary external events, one would need to predefine the events and the responses. If we are willing to leave humans in the system, we can let them handle the recognition. A fully automated system is labor-intensive in analysis, while a semi-automated system is labor-intensive in production. The semi-automated system will be less brittle in the face of new event types.

Our approach here is consistent with Rosen's observation that that closed systems will never be able to mimic what open ones do, and our automated systems are essentially closed systems [12]. Yet we are also saying that automation is useful - we don't always need to take advantage of a human interpreter. Process management which has many tasks automated is enhanced if we can utilize human intelligence for the difficult job of recognizing a broad range of types of external events. We leave open the possibility that automated tools can at least augment our ability to interpret events, as is suggested in ontology-related AI research [13].

Now we turn to a related scenario, in which the focus is the performance of the workflow system itself.

4. The Third Scenario

Applications for student visas are taking 6 to 12 months to be approved, and everyone understands these delays exist in the system. So an individual A, who has an expired tourist visa and a student visa under application, presents the student visa application in lieu of a student visa, and gains re-entry to the country after leaving it, against the intent of the law.

This scenario is more complex than the previous ones, in that it involves the recognition not of a particular anomaly, but of an overall degradation. An intervention won't work at an individual task level, but must be undertaken at a process level.

When a system has deteriorated, and a large community understands this, then, in order to complete business, people may, and often do, work together to circumvent the system. When long transactions pass a certain threshold, the system is perceived as being ridiculous by its users, and the system loses authority.

Determining ahead of time at what point the delays in a system become unbearable is not easy. Also difficult is figuring out the appropriate action to take, as a large and slow administrative system may not be easy to fix while it is being used.

The wisdom we have acquired on large business systems says that the system must be owned - that without ownership, there will be no attention [14]. Also, that the system needs to be measured against a set of target metrics - so that a deviation in performance is noticed and acted on. Target metrics are especially important, as economists have pointed out our propensity to defer those things that are painful to do. Without a plan well ahead of time as to what the response will be to certain scenarios, the realization of a system problem will most likely result in procrastination [15].

Finally, feedback is needed from outside the system – the dissatisfaction of the greater community surrounding a system won't be sensed unless feedback is actively sought and acted upon. Systems themselves rarely make it possible to comment on the system, so some form of regular survey is often needed to check outside perceptions. As well as a mechanism to respond to unsolicited feedback, which in our terminology would take the form of external events. More concretely, in scenario three, if a process for intercepting and acting on feedback learned of a pattern of system circumvention, a manager might be convinced take action. So the design we described in response to the second scenario, of having a manager responsible for reacting to external events, also applies to this third scenario.

There are additional things that need to be done to design against scenario three. Goals need to be set for the performance of the system. A monitoring system needs to be put in place to compare actual performance against target performance. Finally, there needs to be a process to estimate and implement changes to the ongoing system. We show this process in the following diagram, figure 7. The manager who owns the system runs an interface to determine target metrics, which are loaded into the workflow monitor. The monitor reads all internal system events, and notifies the owner when threshold values are approached.

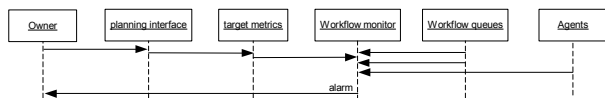


Figure 7.

Looking back at the three scenarios, we can see that solving this third scenario would make the first scenario less likely – the shorter the delays in the system, the less chance that preconditions on an application will be undone.

5. User Interfaces for Event-Based Workflow

In analyzing the first scenario, we established that an event-based architecture would be needed, but showed that a fully automated architecture would be difficult to build. In analyzing the second scenario, we pointed out that external events, while hard for a computer to understand, are not hard for humans to understand. A human, equipped with a way of receiving external events and a way of intervening in the system, could handle the problem. In the third scenario, we pointed out that the managerial task will involve monitoring of internal events such as the performance of the system – but that this interface needs an accompanying interface, a planning interface, so that actual results can be compared against projected results.

Here we propose a model architecture incorporating all these interfaces. First, we look at the status quo – the Workflow Management Coalition Reference Model for interfaces [16], shown in Figure 8.

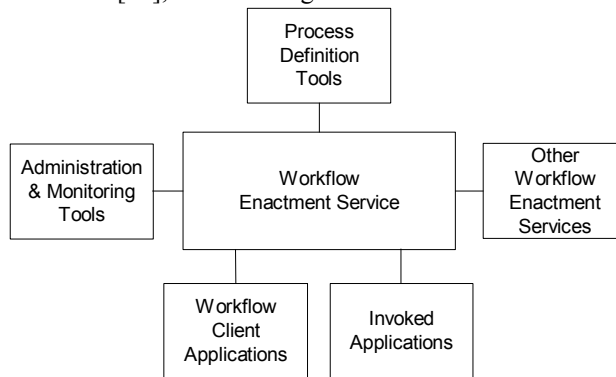


Figure 8.

This model differentiates the workflow client from the invoked applications. It bundles together many different functions under the category of *Administration & Monitoring Tools*. One could keep the model and differentiate the administration and monitoring tools into the interfaces discussed here. Changes from the outside world might be handled at a layer outside of the system.

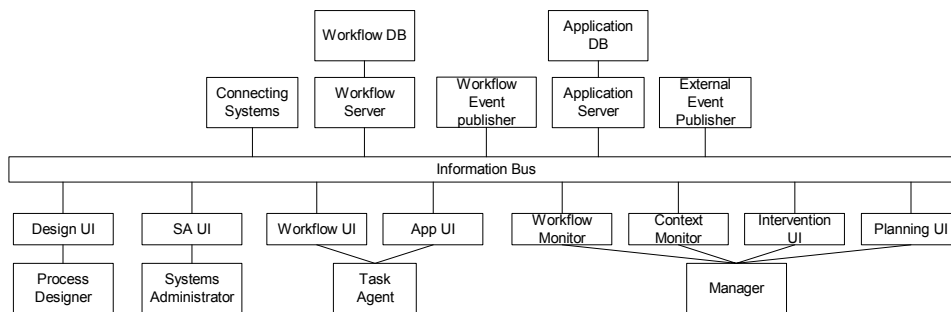


Figure 9.

We believe there is a greater opportunity in redesigning the interface to feature the information bus rather than the enactment service, as in figure 9. This diagram provides a way of thinking about workflow that differentiates many different functions, and provides for continuing differentiation.

In the new model, a software bus infrastructure links together the different internal and external components. Along the top of the information bus the server systems are shown, both the workflow server components proper as well as other systems. Connecting systems, such as data entry systems, can be attached to the bus as the vehicle for integrating new cases into the workflow. The workflow server encompasses the workflow engine, with its accompanying database. The workflow event publisher sends out information for the workflow monitor to process and display, such as additions or deletions from queues. The application server and database also connect to the information bus. An external event publisher is responsible for distributing events from external systems, to be displayed in the context monitor. Along the bottom are a series of interfaces, and the people who will use the interfaces.

Table 1. Comparing the models

| Interface | WfMC Reference Model Correspondence | Description |
|--------------------------------------|---|---|
| Design User Interface | Process Definition Tools | An interface for the design of new process flows |
| Systems Administrator User Interface | Administration and Monitoring Tools | Internally-related matters such as user authorization |
| Workflow User Interface | Workflow Client Applications | For use by the agent pulling a new piece of work off a queue or routing a piece of work to the next step. |
| Application User Interface | Invoked Applications | The application used as part of accomplishing a work task |
| Workflow Monitor | One part of Administration & Monitoring Tools | A look at the state of the system overall, including queue lengths |
| Context Monitor | None | A look at external events that may impact the system |
| Intervention User Interface | Related to Administration Tools | A way for a manager to change something – to pull back cases, or stop the flow |
| Planning User Interface | None | An interface for anticipating possible events and setting up monitoring |

In table 1, we detail the individual components and their correspondences with the Workflow Management

Coalition reference model. We differentiate two monitors – one that receives internal events, and one that receives external events. An interface that allows for interventions is included. Finally, we distinguish a planning UI, which helps determine what events need to be monitored, inside and out.

5. The Management User Interfaces

We describe several of the interfaces in more detail.

5.1 The Intervention Interface

Any administrative interface can be used to make interventions. Yet there are changes to a system, such as switching the primary and the backup machine, that, while they require special privileges, are not public acts. In response to the second scenario, the systems manager may want to pull a set of records off the system – this is an act that should be public. By creating a separate interface, a metaphoric spotlight is shown on such activities, and a system of checks and balances can be put in place so that interventions are public and, most likely, infrequent. All systems effectively have a meta level, in that they can be unplugged or modified. Here we make the meta-level an explicit part of the model.

5.2 The Context Monitor

The intention of this interface is to increase situational awareness for the owner of the system. Just as a good driver will pay attention to cars way ahead or behind, so a systems manager will pay attention to what is going on in connecting systems, in the local environment, and in the world.

Contextual information comes through our senses, and not necessarily through the computer on our desk, and many researchers are looking at how aspects of our environment provide us contextual information. For example, recent research suggests that interfaces which play off our sense of the periphery function well in keeping us aware without interrupting us [17].

5.3 The Planning Interface

It has been observed that changing a workflow system is another form of workflow [18]. In thinking about change at the system level, it is important to recognize that change is happening to a hybrid of people and machines, as shown in figure 10. If we represent a workflow as a set of states, in circles, and a set of activities, in squares, overseen by people, as drawn here,

then changing the system can be seen as a task, overseen by a manager, who moves the entire system, including the people, from an old to a new state.

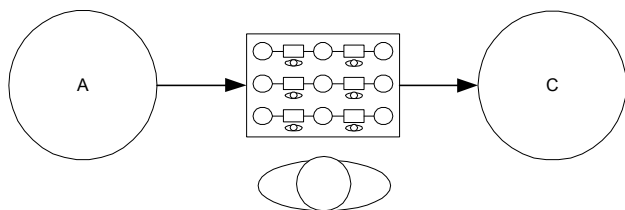


Figure 10.

So the planning interface has an organization flavor. The basic functions that a planning interface might provide include, for example, the setting of budgets – if there is no budget to maintain a system, then deterioration can be predicted with certainty. Related to this, a planning interface provides the manager a way to set the metrics that will be monitored and the threshold values of these metrics that might call for action.

In a more sophisticated mode, scenarios are created and simulations are run to determine alternative strategies. For example, the interface might provide a way of considering the effects that delays have on the enforceability of regulation.

Whereas the workflow and external event monitors prompt reactions, the planning interface are designed to reduce the reactive aspect of running the system, by either preventing the rise of problems, or catching them through alerts so they can be solved before they becomes intractable. Ultimately, the planning software should become a facilitator of anticipation – reacting is the mode we are used to, and anticipation is its opposite [19].

6. Relationship to enterprise integration

Workflow systems were created to perform a very specific piece of integration. The essential concept is to factor out the actual application and focus on the process side of things – the routing. Yet as we have seen, workflow cannot be entirely isolated from issues of data or issues of organization. We believe workflow interfaces play a privileged role in the overall problem of business integration, and here we look at a recent model of enterprise integration, Table 2, explained in [20], and apply it to our discussion. This integration model points out that integration happens at both the machine and human level, and that the mechanisms for integration are both technical and organizational.

In our discussion of the first two scenarios, it became apparent there is a tradeoff – a fully automated system demands a level of analysis that is not within our grasp. Instead, the use of a decision maker who can respond to

messages from the surrounding world is more likely to succeed. Integrating involves working at many different levels, and there are capability tradeoffs – computers don't get tired, but they can't interpret new types of information. How much a workflow system can be automated may be a function of how open it needs to be to external events.

Table 2.

| | Resource/ Integration Need | Examples of Integration Mechanisms | Enabling environment /Infrastructure | |
|----------------------------|---|--|--|-------------------------|
| Organizational Integration | Organizational Units (Functions/Departments) | E-mail, collaborative software, lateral teams ----- Top Management Strategy, budgets, performance metrics | Organization policies/ structure | |
| | Decision Makers | Email, collaborative software, knowledge management systems ----- Face-to-face meetings, job design, performance metrics | | |
| | Business Processes (both internal & external to the firm) | Workflow, Collaborative Systems, SCM, CRM, Web Services ----- Process owners, teams, performance metrics, service level agreements | Standards | Systems Architecture |
| System Integration | Applications | Inter-process communication, RPC, Messaging, ERP, Web Services | Networks | |
| | Data | Data Dictionaries Databases, XML | Platforms | |

The interaction between the layer of business processes and the layer of decision-makers is the workflow user interface. We pointed out that there are really multiple interfaces, some geared toward the performance of tasks, and others geared toward the performance of the overall system. Managing the overall system is a task that demands organizational integration, as workflow systems are composed of both people and machines. And in managing an overall system, strategy, budgets, and performance metrics are as important as the length of queues.

We see that all the techniques of integration come into play in the planning, design and implementation of workflow. At one level, workflow can factor out certain well-understood aspects of business processes. At another level, the system itself needs to be part of an overall integration strategy that takes into account levels ranging from data to organization.

7. Related Work

The scenarios described here are simplifications of recent real-life scenarios involving the United States

Immigration and Naturalization Service, which are well-documented in both the press and in congressional testimony [21]. Flaws in a workflow system became national news because of external events, and the INS is radically reorganizing as a result.

Wargitsch et al. [22] discusses the issue of handling workflow exceptions using organizational memory information systems. In an organizational memory model, there is not only a loop for handling the individual tasks, but a loop for supervising the overall system. This approach could be applied to the third scenario, in that an overall system deterioration could be caught by the supervisory loop.

In terms of the architecture, ever since [6], system designers have been finding new applications for the publish/subscribe paradigm, and many commercial manifestations of it exist, as well as a standard [8]. Closer to the subject of this paper, Moro and Viroli [5] describe a publish/subscribe workflow architecture. They propose an *observation interface*, in which a system makes its state available for inspection to those registering. They make the point that system events should be translated into more understandable observation events before publishing, although their focus is on computer, rather than human, parsing of the events.

A recent paper by Cugola, Nitto and Fuggetta describes a workflow system based on an event-based paradigm [4]. The paper cites other related efforts by the same team [23-25], and describes in detail both commercial and research implementations of event-based workflow.

Other work has also looked at publish/subscribe in relationship to integration [26, 27]. More formal looks at workflow and the problem of long transactions provide a syntactic and semantic perspective [3, 28, 29].

On attempts to describe the context of a system, Abecker et al [13] details an approach to defining ontologies. Earlier, we pointed out that long transactions can be considered from the perspective of the frame problem [9, 10]. In related distributed computing work, software programmers have been trying to create what they describe as context-aware programs – a recent article critiques some of their efforts [30].

Rosen [12] makes the point forcefully that natural systems, including humans, can do a lot more than machines – most importantly, living things can anticipate. Nadin takes this argument further [19] and defines anticipation as the “human sense of context”.

Enterprise integration as it relates to workflow is discussed in [20]; a more general look at coordination science provides perspective on business as well as technology integration [31].

8. Conclusions

While workflow systems can separate process management from applications, they can't separate themselves from the surrounding context. The longer a transaction runs, the more likely that some change in the surrounding world will introduce transaction-affecting information. The challenge is to build a system that incorporates our human ability to respond to this information. At the organizational level, we need to design structures which assign responsibility to this task. At the technical level, we need a mechanism for the distribution of new event-based information. For this, the an information bus architecture will work.

The management interface for workflow, which appears in the Workflow Management Coalition reference document as a combination of administrative and monitoring procedures, needs more distinctions. Monitoring of both events internal to the system as well as external is called for. And a planning process is needed to make monitoring worthwhile – in other words, scenarios need to be generated, and the monitoring directed toward highlighting those phenomena or points in time that have been anticipated to call for further action.

The monitoring of context is what our senses are designed to do. A workflow system deals with distributed activity, and can be difficult to visualize. Events that affect the workflow are likely to be large and unexpected. The first step is to realize such monitoring is important. The second is to plan what to monitor. The third is to integrate the planning and monitoring into the organization.

Workflow systems provoke research in issues such as human decision-making and the use of context. Because transactions are long, changes in the environment can effect the transactions as they happen, as in the scenarios we discussed. And the techniques that can be applied to solving this problem for workflow will also work for other transaction systems. Workflow, by showing how external events influence a running system, may push us to design interfaces that draw on our human ability to recognize significant changes in our environment.

9. References

- [1] J. Rumbaugh, I. Jacobson, and G. Booch, *The unified modeling language reference manual*. Reading, Mass.: Addison-Wesley, 1999.
- [2] W. v. Aalst and K. v. Hee, *Workflow Management: Models, Methods, and Systems*. Cambridge, Mass: MIT, 2002.
- [3] N. S. Barghouti and G. E. Kaiser, "Concurrency Control in Advanced Database Applications," *ACM Computing Surveys*, vol. 23, pp. 260-317, 1991.
- [4] G. Cugola, E. Di Nitto, and A. Fuggetta, "The JEDI event-based infrastructure and its application to the development of the OPSS WFMS," *Software Engineering, IEEE Transactions on*, vol. 27, pp. 827 -850, 2001.
- [5] G. Moro and M. Viroli, "Enabling business cooperation using a publish-subscribe architecture aware of transactions," presented at The 34th Hawaii International Conference on System Sciences, 2001.
- [6] B. Oki, M. Pfluegl, A. Siegel, and D. Skeen, "The information bus-- an architecture for extensible distributed systems.," presented at Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles, 1993.
- [7] F. Casati and A. Discenza, "Supporting Workflow Cooperation Within and Across Organizations," presented at SAC'00, Como, Italy, 2000.
- [8] B. R. Monson-Haefel and D. Chappell, *Java Message Service*: O'Reilly, 2000.
- [9] J. McCarthy and P. J. Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence," in *Machine Intelligence 4*, B. Meltzer and D. Michie, Eds. Edinburgh: Edinburgh University Press., 1969, pp. 463-502.
- [10] L. Morgenstern, "The Problem with Solutions to the Frame Problem," in *The Robot's Dilemma Revisited: The Frame Problem in Artificial Intelligence*, K. M. Ford and Z. Pylyshyn, Eds. Norwood, New Jersey: Ablex Publishing Co, 1996, pp. 99-133.
- [11] D. Lenat and R. Guha, *Building Large Knowledge-based Systems: Representation and Interface in the Cyc Project*. Reading, MA.: Addison-Wesley, 1990.
- [12] R. Rosen, *Anticipatory Systems*. New York: Pergamon Press, 1985.
- [13] A. Abecker, A. Bernardi, K. Hinkelmann, O. Kuhn, and M. Sintek, "Context-aware, proactive delivery of task-specific information: the KnowMore project," *Information Systems Frontiers*, vol. 2, 2000.
- [14] J. Luftman, "Applying the Strategic Alignment Model," in *Competing in the Information Age: Strategic Alignment in Practice*, J. Luftman, Ed. New York: Oxford University Press, 1996.
- [15] T. O'Donoghue and M. Rabin, "Choice and Procrastination," *Quarterly Journal of Economics*, vol. 116, pp. 121-160, 2001.
- [16] D. Hollingsworth, "The Workflow Reference Model," Workflow Management Coalition, 1995.
- [17] J. Cadiz, G. Danielle, Venolia, G. Jancke, and A. Gupta, "Sideshow: Providing Peripheral Awareness of Important Information," 2001.
- [18] C. Ellis and K. Keddera, "A Workflow Change is a Workflow," in *Business Process Modeling*, vol. 1806, W. v. d. Aalst, J. Desel, and A. Oberweis, Eds.: Springer, 2000, pp. 201-217.
- [19] M. Nadin, "Anticipation--A Spooky Computation, the Third International Conference on Computing Anticipatory Systems (CASYS 99), HEC, Liege, Belgium, August 9-14, 1999," presented at Third International Conference on Computing Anticipatory Systems (CASYS 99), Belgium, 1999.
- [20] E. T. Stohr and J. V. Nickerson, "Enterprise Integration: Methods and Direction," in *Competing in the Information Age: Align in the Sand*, J. Luftman, Ed. New York: Oxford University Press, 2002.
- [21] J. Ziglar, "Testimony of the INS Commissioner James Ziglar, Commissioner, Immigration and Naturalization Services, Before the House Committee on the Judiciary Subcommittee on Immigration and Claims," 2001.
- [22] C. Wargitsch, T. Wewers, and F. Theisinger, "An organizational-memory-based approach for an evolutionary workflow management system-concepts and implementation," presented at The 31st Hawaii International Conference on System Sciences, 1998.
- [23] G. Cugola, "Tolerating deviations in process support systems via flexible enactment of process models," *Software Engineering, IEEE Transactions on*, vol. 24, pp. 982 -1001, 1998.
- [24] G. Cugola, E. Di Nitto, and A. Fuggetta, "Exploiting an event-based infrastructure to develop complex distributed systems," presented at The 1998 International Conference on Software Engineering, Kyoto, Japan, 1998.
- [25] G. Cugola and C. Ghezzi, "Design and implementation of PROSYT: a distributed process support system," presented at IEEE 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999. (WET ICE '99) Proceedings., California, USA, 1999.

[26] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," EPFL DSC ID:2000104, 2001.

[27] P. Eugster, R. Guerraoui, and C. Damm, "On Objects and Events," presented at OOPSLA, 2001.

[28] W. M. P. v. d. Aalst, "Exterminating the Dynamic Change Bug: A Concrete Approach to Support Workflow Change," *Information Systems Frontiers*, vol. 3, pp. 297-317, 2001.

[29] M. Weske, "Formal foundation and conceptual design of dynamic adaptations in a workflow management system," presented at The 34th Hawaii International Conference on System Science, 2001.

[30] T. Erickson, "Some Problems with the Notion of Context-Aware Computing," *Communications of the ACM*, vol. 45, pp. 102-103, 2002.

[31] T. W. Malone, K. Crowston, J. Lee, B. Pentland, C. Dellarocas, G. Wyner, J. Quimby, C. S. Osborn, A. Bernstein, G. Herman, M. Klein, and E. O'Donnell, "Tools for inventing organizations: Toward a handbook of organizational processes," *Management Science*, vol. 45, pp. 425-443, 1999.