MANAGING E-MARKET TRANSACTION PROCESSES
Exploring the limits of process management with a multiagent system

John Debenham
University of Technology, Sydney, Australia

Keywords: Multiagent systems, process management, datamining, electronic market transaction management

Abstract: Knowledge-driven processes are business processes whose execution is determined by the prior knowledge of the agents involved and by the knowledge that emerges during a process instance. They are characteristic of emergent business processes. The amount of process knowledge that is relevant to a knowledge-driven process can be enormous and may include common sense knowledge. If a process' knowledge can not be represented feasibly then that process can not be managed; although its execution may be partially supported. In an e-market domain, the majority of transactions, including requests for advice and information, are knowledge-driven processes for which the knowledge base is the Internet, andso representing the knowledge is not an issue. These processes are managed by a multiagent system that manages the extraction of knowledge from this base using a suite of data mining bots.

1 INTRODUCTION

In an experimental e-market, *transactions* (Debenham, 2001) include: trading orders to buy and sell in an e-exchange, single-issue and multi-issue negotiations between two parties, requests for information extracted from market data *as well as* from news feeds and other Internet data. This e-market is used at UTS for research and teaching. In it *every* market transaction is managed as a business process (Fisher, 2000). To achieve this, suitable process management machinery has been developed. To investigate what is 'suitable'the essential features of these transactions are related to two classes of process that are at the 'high end' of process management feasibility (van der Aalst et al., 2001). The two classes are goal-driven processes (Sec. 2) and knowledge-driven processes (Sec. 3). The term "business process management" is generally used to refer to the simpler class of workflow processes (Fisher, 2000), although there notable exceptions using multiagent systems (Jennings et al., 2000). Sec. 4 describes the relationship between the transactions themselves and the contextual information extracted from the Internet and from market data. Sec. 5 discusses the single-issue and multi-issue negotiation transactions. e-market transactions are described in Sec. 6.

2 GOAL-DRIVEN PROCESSES

A *goal-driven process* has a process goal, and achievement of that goal signals the termination of the process. The process goal may have various decompositions into possibly conditional sequences of sub-goals where these sub-goals are associated with (atomic) activities and so with atomic tasks. Some of these sequences of tasks may work better than others, and there may be no way of knowing which is which (van der Aalst et al., 2001). A task for an activity may fail outright, or may fail to achieve its goal in time. In other words, a central issue in managing goal-driven processes is the management of task failure. Hybrid multiagent architectures whose deliberative reasoning mechanism is based on "succeed/fail/abort plans" (Rao et al., 1995) are well suited to the management of goal-driven processes. Goal-driven processes are a more powerful concept than production workflows (or, called "activity-driven processes" in (Debenham, 2000)). *Activity driven-processes* are associated with possibly conditional sequences of activities where performing that sequence is assumed to "work always".

Following (Rao et al., 1995) a plan for a goal-directed process can not necessarily be relied upon to achieve its goal even if all of the sub-goals on the

Debenham J. (2004).
MANAGING E-MARKET TRANSACTION PROCESSES - Exploring the limits of process management with a multiagent system.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 322-330
DOI: 10.5220/0002597203220330

chosen path through that plan have been achieved. The *success condition* (SC), described in (Debenham, 2000), is a procedure whose goal is to determine whether a plan's goal has been achieved. The final sub-goal on *every* path through a plan is the plan's success condition. The success condition is a procedure; the execution of that procedure may succeed(3), fail (7) or abort (**A**). If the execution of the success condition fails then the overall success of the plan is unknown (**?**). So the four possible plan exits resulting from an attempt to execute such a plan are as shown in Fig. 1. A *plan body* is represented as a directed AND/OR graph, or state-transition diagram, in which some of the nodes are labelled with sub-goals.

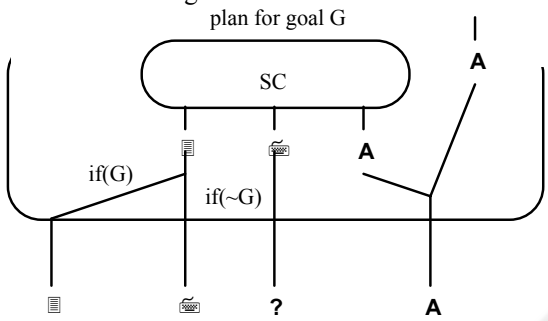

Figure 1: The four plan exits

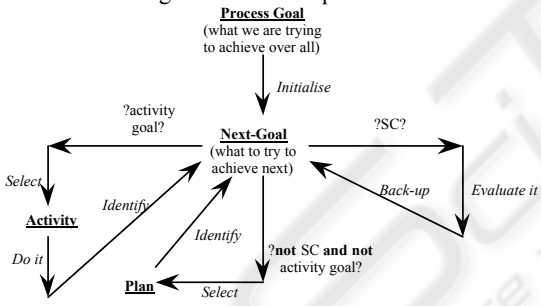

Figure 2: A simplified view of goal-driven process management

The management of goal-driven processes is shown in a simplified form in Fig. 2. There, starting with the overall process goal, repeated decomposition of plans and goals is performed until either the next goal is a success condition or is an *activity goal*—ie: a goal for which there is a hard-coded procedure. Fig. 2 is simplified because it does not show what happens if the success condition returns fail "7", or what happens if a plan is aborted. Further it does not show the mechanism for selecting plans for goals. For goal-driven processes there is, in general, no *ex ante* 'best' choice of plan.

3 KNOWLEDGE-DRIVEN PROCESSES

A second class of process, whose management has received little attention, is called knowledge-driven processes. *Process knowledge* is all the knowledge that is relevant to a process instance. It includes common-sense knowledge, knowledge that was available when an instance is created, and knowledge acquired during the time that that instance exists. A *knowledge-driven process* may have a process goal, but the goal may be vague and may mutate. In so far as the process goal gives direction to goal-driven—and activity-driven—processes, the process knowledge gives direction to knowledge-driven processes. The body of process knowledge is typically large and continually growing—for example, it may include common sense knowledge—and so knowledge driven processes are seldom considered as candidates for process management. They are typically supported, rather than managed, by CSCW systems. But, even complex knowledge-driven processes are "not all bad"—they typically have goal-driven sub-processes that may be handled as described above. *Knowledge-base processes* are a special type of knowledge-driven process for which the process knowledge *can* be represented and accessed by a process management system. This proves to be a useful concept for managing e-market transactions.

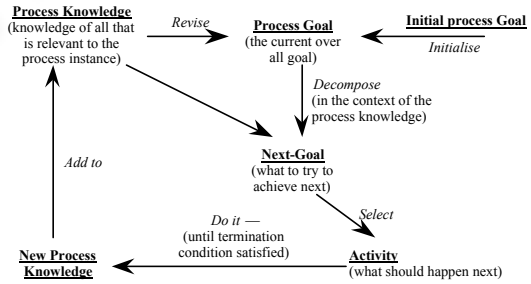

Figure 3: A simplified view of knowledge-driven process management

The management of goal-driven and knowledge driven processes are radically different. Goal-driven processes may be managed by a goal/plan decomposition process (see Fig. 2), and knowledge-driven processes are managed by continually reviewing the growing corpus of process knowledge—this is illustrated in Fig. 3. That Figure is deceptively simple in that the business of managing the process knowledge and of revising the process-goal andnext-goal in the light of that growing body of knowledge is far from trivial in

even simple examples. In general this problem will be intractable. But in some cases, including the majority of e-market transactions, smart tools may be used to do this. This is discussed in the next section.

4 E-MARKET TRANSACTIONS AND CONTEXTUAL INFORMATION

E-market transactions include: trading orders to buy and sell in an e-exchange (single-issue and multi-issue negotiations as described above), requests for market data *as well as* requests for information extracted from news feeds and other Internet data. In an experimental e-market, *all* e-market transactions are managed as constrained knowledge-driven processes.

Sec. 5.1 discusses single-issue one-to-one negotiation. Single-issue negotiation also takes place in exchanges, for example a 'buy' trading order to "buy a chair and a desk for less than $100". This is represented (see Fig. 4) as a naive plan with goal [G, c] = [desk and chair have been purchased, cost < 100]. This plan has sub-goal SG_1 = 'chair and desk selected', $[SG_2, c_2]$ = [chair purchased, cost < 30], $[SG_3, c_3]$ = [desk purchased, cost < 50], and $[SG_4, c_4]$ = [desk and chair delivered, cost < 20]. The management of this purchase order is represented as a plan whose goals have monetary constraints.

An example of a request for information is "find out all you can about ABC Corp within five minutes". This triggers a process to locate, extract, validate, condense and combine information from the Internet. The location and extraction tasks are achieved by data/text mining bots that are described in [7]. A handcrafted plausible inference network combines contradictory information. The use of belief nets that can be trained "off line" is very tempting and is currently being investigated [8]. The data/text mining bots produce output in the form [data, belief]—ie: some data and a measure of the belief held in the validity of that data. A request for information is first represented as a goal/constraint pair: [find_info_about('ABC Corp'): time_upper_limit = now + 5mins]. Given a goal/constraint pair, a plan (see Fig. 4) is selected for it—the mechanism for selecting a plan is described in Sec. 6. A *plan* for a goal/constraint pair is a possibly conditional state-chart of sub-goals over which constraints are distributed as described in Sec. 6. For a 'find_info_about'process, the plan uses a Dempster-Shafer network (see Fig. 5) to combine results $[D_i, b_i]$, in the form [data, belief], extracted from the Internet by a suite of data/text mining bots. The network actually does more than combine information. If the level of belief, b_R, in a result, R, derived by the network is below a set threshold then a 'reverse calculation' identifies 'inputs' whose belief levels are responsible for the low level of belief in R. Then further data/text mining is initiated in an attempt to raise this level of belief at least for future calculations if not for the present calculation.

A three-year research project commencing in 2002 at UTS, is investigating the mechanisms required to support the evolutionary process in e-markets (Debenham, 2001). It is presently funded by four Australian Research Council Grants; awarded variously to the author and to Dr Simeon Simoff:
`http://www-staff.it.uts.edu.au/~emrktest/eMarket/`
Market evolution is linked to innovation and entrepreneurship (in its technical, economic sense). Present plans for the three year project are: (1) to build an e-marketplace trader's workbench that, in principle, enables a trader to operate without external information, (2) to assist a trader to identify arbitrage opportunities triggered by the occurrence of rare events, (3) to assist a trader to identify innovative forms of trade, and, possibly, (4) to understand somethingof the evolutionary process itself.

5 NEGOTIATION TRANSACTIONS

Negotiation is a process whereby two or more agents reach an agreement on a set of issues. One-to-one negotiation, in which there are just two negotiating agents, is sometimes called *bargaining*, or informally "haggling" or "dickering". An *issue* is any good or service that one agent can provide to another, including money. The *issue set* is the range of possible issues that may be considered during a negotiation. An issue set may be *fixed*; for example, in a single-issue negotiation where the only issue on which agreement is sought is an amount of money. An *open* issue set may contain *any* issue. In a*limited* issue set, the issues that may be included in an offer is limited to those chosen

from a set agreed to by the negotiating agents. An issue—for example, "period of warranty"—is normally associated with some value—for example, "two years". An *offer* consists of a particular set of issues chosen from the issue set, together with values for those issues. During a negotiation with an open or limited issue set the collection of issues in an offer may mutate although in practice it tends to be moderately stable.

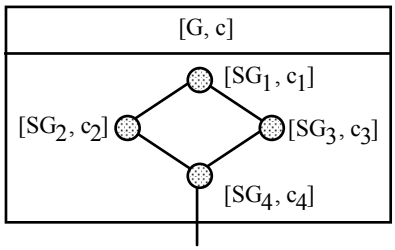

Figure 4: A plan for goal [G,c]

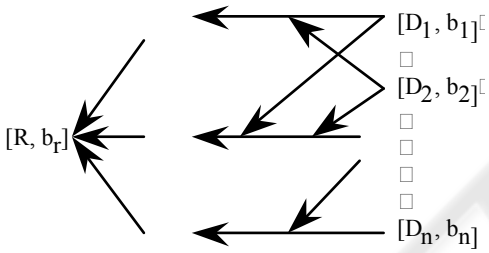

Figure 5: Belief network combines information

A *negotiation mechanism* specifies how a negotiation may proceed; they are sometimes called "interaction protocols" in multiagent systems work (Weiss, 1999). Two forms of negotiation mechanism have received a considerable amount of attention in the economics literature, and in e-Markets research. First, single issue negotiation for one or moreitems being offered either to a set of buyers or to a set of sellers; see, for example, the extensive work on forward and reverse auction mechanisms (Klemperer, 2000), and second, one-to-one negotiation, or bargaining, mechanisms.

Management of the negotiation process in an e-Market—both for negotiation through e-exchanges and through single- and multi-issue one-to-one negotiation—includes a continual investigation of the negotiation *context* as well as the construction, evaluation and revision of offers. For example, the bone fide of the opponent may require verification, the quality of the goods should be confirmed, alternatives should be investigated, and so on. In the experiments described here this information is assumed to be available on the Internet. A good e-market negotiator should conduct these contextual investigations as an integral part of the negotiation process (Debenham et al., 2002). "Good negotiators, therefore, undertake integrated processes of knowledge acquisition combining sources of knowledge obtained at and away from the *negotiation table*. They learn in order to plan and plan in order to learn" (Watkins et al., 2002). In the management of the negotiation process described here, the information and the offers develop in tandem; they both feed off each other. The term "e-marketplace" is used here to acknowledge this duality between offers and contextual information. An *e-Marketplace* is a market in which trading can be conducted over the Internet, and for which sufficient information to trade "well" is available over the Internet. This information may be derived from on-line market data, for mining historic market data, from text-mining news feeds and so on. The Sydney Stock Exchange is an example of an e-marketplace.

5.1 Single-issue negotiation

Single-issue negotiation is the most common form of negotiation, in particular where the issue is price. The number of issues in any form of negotiation, including single-issue, can be increased if one of the negotiating parties offers "kick backs". For example, an offer of two free bottles of wine for every dozen bought *provided that* you have spent more than $500 with that merchant in the previous twelve months. This sort of offer changes what was initially a single-issue negotiation to a multi-issue negotiation. In this Section it isassumed that the negotiation is strictly single-issue and that both parties understand the meaning of the issue. For example, such an issue could be an amount of money. It is argued that single-issue negotiation is appropriately managed as a knowledge-driven process. From a process management perspective this is interesting because the management of "real life"knowledge-driven processes is usually unfeasible.

Two important classes of bargaining mechanisms are alternating offers mechanisms and single-round, "one-hit" mechanisms that may be used when the agents have determined private valuations in advance. For example, (Myerson et al., 1983) shows that a one-hit"split the difference between bid and ask" mechanism should be preferred by both buyer and seller to *any other* mechanism *ex ante*—that is, before their private

valuations are actually determined. Alternatively, an agent's valuations may be refined as the negotiation proceeds—in which case an alternating offers mechanism is which information is tabled as appropriate—this is the focus here.

The negotiation protocol used is a time-constrained, unbounded alternating offers protocol (Kraus et al., 2001). In this protocol two bargaining agents exchange offers until either one agent accepts an offer from the other agent, one agent rejects an offer and withdraws without penalty, or one agent exceeds an agreed time constraint on making an offer. So negotiation using this protocol could, in principle, proceed indefinitely—hence the description *unbounded*.

Consider a transaction to purchase something. Suppose that this transaction can be appropriately managed by: identifying a need, selecting a good to satisfy that need, choosing a supplier for that good and negotiating terms for that good from that supplier. A sequential procedure basedon this would not be appropriate for purchasing all classes of goods; it could, however, be suitable for purchasing a technical book. The appropriateness of this "purchasing procedure" is not of concern here. Suppose further that we wish to select the "most appropriate" good, to choose the "best supplier" and to negotiate "acceptable" terms. In an *e-marketplace* sufficient information to trade successfully in this sense is assumed to be available on the Internet.

The use of "software bots" to assist the buying process by extracting contextual information from the Internet is commonplace. For many classes of goods, bots that do some of this work are freely available: http://www.botspot.com/ — viz: the sections "Shopping Bots" and "Commerce Bots". The entire problem considered here lies beyond the capacity of most off-the-shelf bots that at best recommend rather than decide. Although the use of demographic data, collaborative filtering, clustering, or previously expressed user preferences can deliver reasonable performance in choosing the "most appropriate" good for the user. With current technology it could be reasonable to give the authority to a bot to select, order and pay for paper stock for photocopiers, but many would be reluctant to permit a bot to select, order and pay for a book on Bayesian Nets, for example.

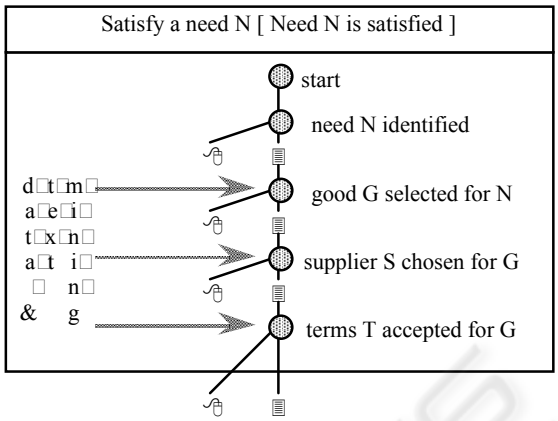

Figure 6: A goal-driven plan to satisfy a need based on "succeed / fail" plans

In this project the contextual information from the Internet is first extracted by a range of data and text mining bots mostly written by undergraduates at UTS. Some of these bots read reviews of products in an attempt to determine the comparative inherent quality of a good as well as its basic attributes. This in general leads to a collection of contradictory evidence that is combined to give coherent advice. The approach taken to plausible inference is described in Sec. 4

A simple, double (ie: succeed / fail) branching plan to manage a "purchase something" transaction is shown in Fig. 6. That plan treats the process sequentially in that, for example, once the good is selected then its appropriateness is not reconsidered. This may be appropriate when the whole transaction can be resolved quickly, but could otherwise lead to poor decision making. That plan relies on information from data and text mining bots to support the decision making in the achievement of three of its sub-goals. [Plans for those three sub-goals are not shown here.] Despite the vital role of the bots, that plan manages the transaction as a goal-driven process. The management of the same transaction as a knowledge-driven process is described below. This is achieved by feeding the contextual information into the reactive "abort" conditions in the plans.

To simplify the following discussion the operation of the data and text mining bots is hidden in the following predicates: INeed(N) that means: "I need an N", Satisfy(N, G) that means: "good G is the most appropriate good that satisfies need N". Calculation of values to satisfy these predicates may take some time.

Fig. 7 shows one of a sequence of linked plans for the "purchase something"transaction. In that Figure "d t m" denotes information that is acquired

from the Internet and databases by data and text mining bots and combined into coherent advice by a plausible inference network. That plan is more intricate than the previous goal-directed version. Even so this sequence is flawed in that the process may now continue indefinitely; this difficulty is addressed by constraints in Sec. 6 below. They include reactive abort triggers that redirect the course of the transaction if any prior decision ceases to be valid. The direction, and possible redirection, of this transaction is governed entirely by the contextual information received, and repeatedly reconfirmed, from the data and text mining bots. This plan is useful but is not particularly noteworthy in itself. What is of note, from a process management perspective, is that this is a fully managed knowledge driven process, despite its presentation in a goal / plan framework. It is a knowledge-base driven process where the knowledge base is the Internet and market data, the query mechanism is the bots, and the reactive 'abort' exists are used to modify the direction of the process when necessary.

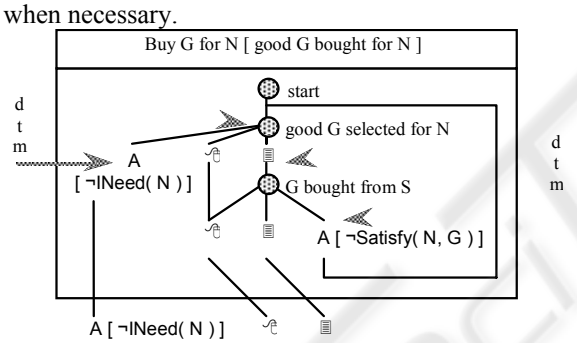

Figure 7: Knowledge-driven plans to satisfy a need based on "succeed / fail / abort" plans

5.2 Multi-issue negotiation

The discussion in Sec. 5.1 on the management of single-issue negotiation is rather simplified in that it assumes that the single-issue in which the offers are expressed is unambiguously understandable. In practice negotiation is more complicated than this. For example, when the issue is money then an amount expressed in dollars is readily understood, but when and where the payment has to be made may not be. If the issue set contains things like an "unconditional warranty" then it would be prudent to clarify quite what this really means. So the feature of multi-issue negotiation that is explored here is the opponent as a source of information, used to clarify the meaning of an offer or otherwise.

A negotiation process with fairly minimal functionality is shown in Fig. 8. There the process is triggered by the arrival of an offer from the opponent. This is analysed to ensure that the meaning is clear—to uncover the "fine print"— and to detect any inconsistencies. Then the offer is evaluated to determine what it is "worth"—this can lead to acceptance or outright rejection, or to the development of a counter offer. Another context for the generation of the counter offer is the history of offers received in this negotiation—this enables an assessment to be made of"where are the opponent is at". Eg: "is she about to give in?" The process illustrated in Fig. 8 can seen as an attempt to satisfy the high level goal "attempt to negotiate a satisfactory outcome". But the direction that the process takes is determined by the flow of information—from the offer itself, the data and text mining bots, the opponent and from the growing history of offers. So this is a knowledge-driven process. At present the evaluation function is available from the bots as described above. At the time of writing the rest of the machinery is not yet available, but plans are to achieve this by the end of 2002. There is much to be done, for example the detection of inconsistencies in an offer is not trivial even if the terms of the offer are represented in Horn clause logic.

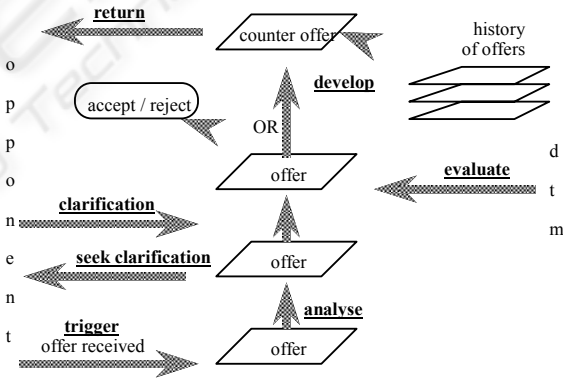

Figure 8: High-level view of the negotiation process

6 TRANSACTION CONSTRAINTS

All e-marketplace transactions are assumed to be constrained by time constraints and possibly by cost constraints or success constraints. *Time constraints* may be the maximum (or minimum) time by which a deal must be struck and/or by which the goods should be delivered. The *cost constraints* could be constraints on the cost of the transaction, the cost of the goods or a combination of the two. *Success constraints* may be constraints on the outcome of the

deal; for example, "I must have a car for the weekend, get the best deal you can".

The e-marketplace transaction management system attempts manage transactions to "deliver the best it can whilst satisfying the constraints". To do this it selects plans to achieve goals on the basis of expected time and cost estimates. Further, if actual performance differs significantly from these estimates then estimates for subsequent plans are adjusted leading, possibly, to a revised plan. This will occur if network performance is unexpectedly degraded, for example. To derive time and cost estimates for each plan it gathers performance measurements on each plan and sub-system, such as an information gathering bot, and maintains running estimates of future expected performance. It then adjusts these estimates when measurements are observed outside expected limits. For example, if the network is slow when gathering data from New York, then time estimates for extracting data from London may be adjusted to some extent.

Time and cost performance measurements are made for each plan and for each atomic sub-system whenever it is used. These measurements enable the transaction management system to choose a plan for a goal (G in Fig. 4) and to determine the constraints ($\{c_1,..,c_4\}$ in Fig. 5) for each sub-goal in that plan. A plan's *performance estimate* is the expected time "t" and cost "c" to satisfy the plan's goal. These estimates will be calculated from performance estimates for each atomic sub-system. The parameters t and c are assumed to be normally distributed—this is a wild assumption—but it provides a framework for identifying measurements that abnormal. Given a parameter, p, that is assumed to be normally distributed, an estimate, μ_p, for the mean of p is revised on the basis of the i'th observation ob_i to $\mu_{p_{new}} = (1 - \alpha) \_ ob_i + \alpha \_ \mu_{p_{old}}$ which, given a starting value $\mu_{p_{initial}}$, and some constant α, $0 < \alpha < 1$, approximates the geometric mean $\f(\O(\Sigma_{i=1}^{n}) \alpha^{n-i} \_ ob_i, \O(\Sigma_{i=1}^{n}) \alpha^{n-i})$ of the set of observations $\{ob_i\}$ where i = n is the most recent observation. In the same way, an estimate, σ_p, for $\r(\f(2,\pi))$ times the standard deviation of p is revised on the basis of the i'th observation ob_i to $\sigma_{p_{new}} = (1 - \alpha) \_ | ob_i - \mu_{p_{old}} | + \alpha \_ \sigma_{p_{old}}$ which, given a starting value $\sigma_{p_{initial}}$, and some constant α, $0 < \alpha < 1$, approximates the geometric mean $\f(\O(\Sigma_{i=1}^{n}) \alpha^{n-i} \_ | ob_i - \mu_p |, \O(\Sigma_{i=1}^{n}) \alpha^{n-i})$. The constant α is chosen on the basis of the stability of the observations. For example, if $\alpha = 0.85$ then "everything more than twenty trials ago" contributes less than 5% to the weighted mean.

Given a transaction and its constraints (expressed in terms of t and s), the transaction management system makes two decisions. First it selects a feasible plan for that transaction's goal. Second it determines the constraints on each sub-goal in that plan. Then further plans are selected for those sub-goals, and so on. Each time a plan for goal G is used measurements are made of t and c for each sub-goal in that plan. Further each of those sub-goals may be invoked by other plans. So the estimates of the mean and standard deviation of t and c for those sub-goals may be expected to be more accurate than the estimates for goal G. So each time a plan is considered, the t and c estimates for its goal are re-computed from those on the estimated costliest path through the plan.

Plan A for goal [G, c] is *feasible* if $c > \mu_A + \kappa \_ \sigma_A$, where c is expressed in terms of t and c, μ and σ are expressed likewise, and κ is a constant usually > 1. If $c < \mu_A - \kappa \_ \sigma_A$ then the plan is not expected to achieve its goal within constraint c. This enables the constraints to be relaxed on each sub-goal so that the estimated costliest path through the plan satisfies c. If a sub-goal SG_i of plan P for goal Gis not achieved within its constraint c_i then *first* another plan is sought for SG_i and for any other as-yet-unsatisfied 'down stream' sub-goals, for which an allocation of constraints in P is feasible, and *second* the whole plan P fails and another plan is sought for G with tighter constraints than c.

Given a goal G with constraints c the transaction management system first identifies a set of feasible plans for G. Then from this set the system selects a plan for a given goal G using the stochastic strategy: the probability that a plan is selected is the probability that that plan is the "best" plan. This strategy has been found to work well for managing high level processes [6]. Here *best* may mean "the most likely to satisfy the constraints on G" or some other criterion such as "the plan likely to deliver the best quality advice" as discussed below. Given two plans A and B for the same goal G, if the constraint on G is represented by a parameter p (in

terms of t and c) that is assumed to be normally distributed then the probability that plan A is "better than" plan B is the probability that $(p_A - p_B) > 0$. Using elementary statistics, an estimate for this probability is given by the area under the normal distribution with:

mean = $\mu_A - \mu_B$ [where μ_A and μ_B are estimates of the means of p_A and p_B]

standard deviation = $\r(\f(\pi,2) \_ (\sigma\O(^2,_A) + \sigma\O(^2,_B)))$ [where σ_A and σ_B are estimates of $\r(\f(2,\pi))$ times the standard deviations of p_A and p_B] for $x > 0$. This method may be extended to estimate the probability that one plan is better than a number of other plans.

The measurement of the quality, q, of work in any business process is seldom available to the management system except through subjective assessment. This issue complicates the "optimal" management of all business processes. In the experimental e-market some information sources may reasonably be given quality estimates. For example, subjective estimates of the mean and variance may be attached to text by a particular journalist in a news feed. If these estimates are available then the notion of "best" may be extended to include quality. However, if "best" is to mean some combination of q, c and t then these parameters may need to be measured in the same units, such as some monetary value.

The adjustment of estimates in the light of measurements that fall outside expected ranges is achieved using the geometric weighted mean method used to estimate s and t. These *multipliers* υ_{ij} mean: if measurement m_i of service i lies outside the expected range then multiply the estimate for service j by $\F(m_i ,\mu_i) \_ \upsilon_{ij}$. This is crude but in a sense these multipliers are no cruder than the estimates that they are adjusting. What is known is the network topology and so too potential causal links between components' performance. The use of some form of belief net [8] is appealing in that the learning mechanism has a scientific basis, although here the nets will need to represent conditional t and c estimates rather than conditional probabilities. This is presently being investigated.

7 CONCLUSION

Two classes of business process are goal-driven processes and knowledge-driven processes. Goal-driven processes may be managed but they are inherently unpredictable. The management of knowledge-driven processes that involve human agents is seldom feasible due to the size of the process knowledge base. A significant class of knowledge-driven processes is e-market transactions in which the process knowledge base is the Internet. These are managed using a multiagent system that is supported by a suite of data and text mining bots whose output is combined using a belief network. The proactive component of these agents is specified by plans. For goal-driven processes the proactive 'succeed' exit leads the way, and for knowledge-driven processes the reactive 'abort' exit is used to determine the process' direction as knowledge is revealed.

REFERENCES

Debenham, JK. Supporting the actors in an electronic market place. In proceedings Twenty First International Conference on Knowledge Based Systems and Applied Artificial Intelligence, ES'2001: Applications and Innovations in Expert Systems IX, Cambridge UK, December 2001, pp29-42.

Fischer, L. (Ed). Workflow Handbook 2001. Future Strategies, 2000.

van der Aalst, W. & van Hee, K. Workflow Management: Models, Methods, and Systems. MIT Press (2001).

Jennings, N.R., Faratin, P., Norman, T.J., O'Brien, P. and Odgers, B. (2000) Autonomous Agents for Business Process Management. Int. Journal of Applied Artificial Intelligence 14 (2) 145-189.

Rao, A.S. and Georgeff, M.P. "BDI Agents: From Theory to Practice", in proceedings First International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, USA, pp 312—319.

Debenham, JK. Supporting knowledge-driven processes in a multiagent process management system. In proceedings Twentieth International Conference on Knowledge Based Systems and Applied Artificial Intelligence, ES'2000: Research and Development in Intelligent Systems XV, Cambridge UK, December 2000, pp273-286.

Debenham, JK and Simoff, S. Investigating the Evolution of Electronic Markets. In proceedings Sixth International Conference on Cooperative Information Systems, CoopIS 2001, Trento, Italy, September 5-7, 2001, pp344-355.

Cowell, RG, Dawid, AP, Lauritzen, SL and Spiegelhater, DJ. Probabilistic Networks and Expert Systems. Springer-Verlag, (1999)

Weiss, G. (ed) (1999). Multi-Agent Systems. The MIT Press: Cambridge, MA.

Klemperer, P. (Ed). The Economic Theory Of Auctions. Edward Elgar Publishing (2000).

Debenham, JK and Simoff, S. Designing a Curious Negotiator. In proceedings Third International Workshop on Negotiations in electronic markets - beyond price discovery - e-Negotiations 2002, September 2002, Aix-en-Provence, France.

Watkins, M. Breakthrough Business Negotiation—A Toolbox for Managers. Jossey-Bass, 2002.

Myerson, R. & Satterthwaite, M. Efficient Mechanisms for Bilateral Trading. Journal of Economic Theory, 29, 1–21, April 1983.

Kraus, S. Strategic Negotiation in Multiagent Environments. MIT Press, 2001.