A Multiagent System Manages Collaboration in Emergent Processes

John Debenham
Faculty of Information Technology,
University of Technology, Sydney, NSW Australia
debenham@it.uts.edu.au

Abstract

Emergent processes are non-routine, collaborative business processes whose execution is guided by the knowledge that emerges during a process instance. They may involve informal interaction, and so there is a limit to the extent to which the processes can be "managed". The collaboration *however can be managed. Managing collaboration needs an intelligent agent that is guided not by a process goal, but by observing the performance of the other agents. Each agent has* process knowledge *— that is information either generated by the individual users or is extracted from the environment, and* performance knowledge *— that describes how the other agents, together with their 'owners', perform — including how reliable they are. The integrity of the information derived from past observations decays in time, and so they have an inference mechanism that can cope with information of decaying integrity. An agent is described that achieves this by using ideas from information theory.*

1. Introduction

Emergent processes are business processes that are not predefined and are *ad hoc*. These processes typically take place at the higher levels of organisations [9], and are distinct from production workflows [5]. Emergent processes are opportunistic in nature whereas production workflows are routine [3]. How an emergent process will terminate may not be known until the process is well advanced. The tasks involved in an emergent process are typically not predefined and emerge as the process develops. Those tasks may be carried out by collaborative groups as well as by individuals [15] and may involve informal meetings, business lunches and so on. Further, the goal of an emergent process instance may mutate as the instance matures. So unlike "lower-order" processes, the goal of an emergent process instance may not be used as a focus for the process management system. An emergent process may have a fixed goal such as "maximize profits" — but it is unlikely that a process management agent, or a human agent, will have an executable plan to achieve such a goal.

The term "business process management" [14] is generally used to refer to the simpler class of workflow processes [5], although there are notable exceptions [11]. From the management perspective, emergent processes are "knowledge-driven". A *knowledge-driven process* is guided by its "process knowledge" and "performance knowledge". *Process knowledge* is information either generated by the individual users or is extracted from the environment, and includes background information. *Performance knowledge* describes how the other agents together with their 'owners' perform, including how reliable they are.

In so far as the process goal gives direction to goal-driven processes, the continually evolving process knowledge gives direction to knowledge-driven processes. So plan-based agent architectures such as BDI [17] are not directly suitable. To manage knowledge-driven process with such an architecture would require machinery to manage the mutations of the process goal. The agent architecture described here manages the collaboration in the processes, and not the processes themselves. This architecture is based on information theory.

2. Process Management

Following [5] a *business process* is "a set of one or more linked procedures or activities which collectively realise a business objective or policy goal, normally within the context of an organisational structure defining functional roles and relationships". Implicit in this definition is the idea that a process may be repeatedly decomposed into linked sub-processes until those sub-processes are activities which are atomic pieces of work. [viz [5] "An *activity* is a description of a piece of work that forms one logical step within a process."].

A particular process is called a (process) *instance*. An instance may require that certain things should be done; such

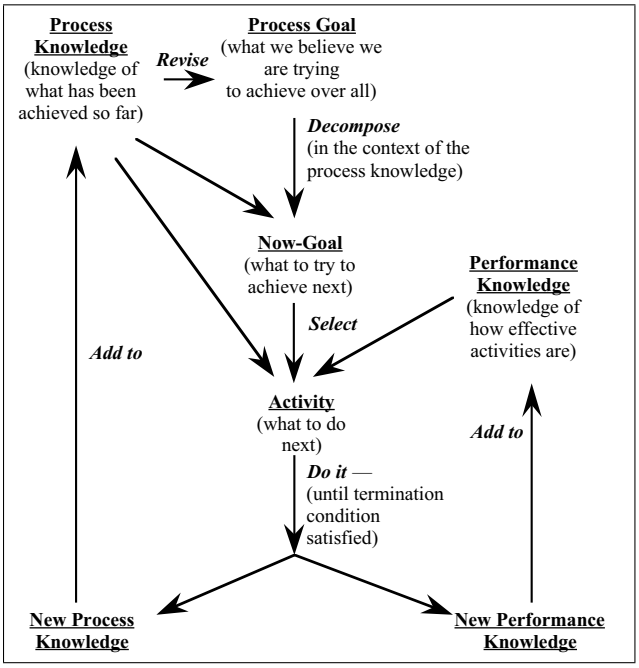

Figure 1. Knowledge-driven processes.

things are called tasks. Three classes of business process are defined in terms of their management properties.

- A *task-driven process* has a unique decomposition into a — possibly conditional — sequence of activities. Each of these activities has a goal and is associated with a task that "always" achieves this goal.
- A *goal-driven process* has a process goal, and achievement of that goal is the termination condition for the process. The process goal may have various decompositions into sequences of sub-goals where these sub-goals are associated with (atomic) activities and so with tasks. Some of these sequences of tasks may work better than others, and there may be no way of knowing which is which [15]. The possibility of task failure is a feature of goal-driven processes.
- A *knowledge-driven process* may have a process goal, but the goal may be vague and may mutate [3]. Mutations are determined by the knowledge generated during the process. At each stage in the performance of a knowledge-driven process, the "next goal" is identified using the process knowledge. So in so far as the process goal gives direction to goal-driven — and task-driven — processes, the process knowledge gives direction to knowledge-driven processes. A simplified view of knowledge-driven process management is shown in Fig. 1.

The complete representation, never mind the maintenance, of the process knowledge may be complex. However, in a process management system, or in an Electronic Institution [4], much of that knowledge may be readily available. Performance knowledge is not difficult to capture, represent and maintain. For example, measurements of how long another agent took to complete a sub-process, and measurements of how reliable the other agents are.

3. Emergent Process Management System

In the system described here each human player is assisted by an agent. As emergent processes may involve informal interaction between players, there is a limit to the extent to which the processes themselves can be "managed". This is in contrast to task-driven processes, or production workflow, where a management system prescribes *what should happen next* — agents are "asked" to do things and are "expected" to comply. For emergent process the *collaboration* can be managed. The questions that an emergent process agent considers include: "who to ask to assist", "who can I rely on", "who works well with who", "who do I want to build a relationship with". The answers to these questions are inferred by observing the dynamics of the collaboration between the agents. So an agent for emergent process management needs to be able to observe and evaluate the collaboration — what appears to work and what does not — and has to make sense out of this diverse information.

The act of an agent *joining* a real or virtual group for some purpose is fundamental to collaboration. Another act is one agent *delegating* responsibility for a sub-process to another. The product of group activity, or process delegation, is some information being generated, and so to the act of *information being passed* from one agent to another. The system aims to implement these three types of act intelligently. It consists of the set of agents $\{X_i\}_{i=0}^{n}$ — the description following is written from the point of view of agent X_0 that interacts with the other n agents. In the text, the agent X_ω is "an other" agent — i.e. $\omega \neq 0$.

The agent architecture extends the agent described in [2]. It is driven by the contents of a knowledge base that represents the agent's world model represented in probabilistic first-order logic. The system attempts to manage the collaboration using the information that is generated both by and because of it. To achieve this, it draws on ideas from information theory. As with the agent described in [2], X_0 makes assumptions about: the way in which the integrity of information will decay, and some of the preferences that its collaborators may have for some agreements over others. It also assumes that unknown probabilities can be inferred using *maximum entropy inference* [12], *ME*, which is based on random worlds [7]. The maximum entropy probability distribution is "the least biased estimate possible on the given

information; i.e. it is maximally noncommittal with regard to missing information" [10]. In the absence of knowledge about the other agents' allegiances, X_0 assumes that the "maximally noncommittal" model is the correct model on which to base its reasoning.

X_0 decides what to do — such as what message to send — on the basis of its past observations, the current integrity of which is expressed as degrees of belief. X_0 uses this information to calculate, and continually revise, probability distributions for that which it does not know. One such distribution, over the set of all possible actions, expresses X_0's belief in the suitability to herself of performing that action. Other distributions attempt to predict the behavior of another agent — such as what agreements she might accept. X_0 is purely concerned with the other agents' behaviors — what they actually do — and not with assumptions about their motivations. This somewhat detached stance is appropriate for emergent process management in which each agent represents the interests of it owner, whilst at the same time attempting to achieve social goals.

4. Emergent Process Agent X_0

X_0 operates in an information-rich environment that includes the Internet. One source of X_0's information is the signals received from X_ω. These include proposals from X_ω to X_0, the acceptance or rejection by X_ω of X_0's proposals, and information that X_ω sends to X_0. Incoming information is augmented by X_0 with sentence probabilities that represent the strength of her belief in its truth. If X_ω refused to assist X_0 two days ago then what is X_0's belief now in the proposition that X_ω will assist her now? Perhaps it is around 0.1. For simplicity, a linear model is used to model the integrity decay of these beliefs, and when the probability of a decaying belief approaches its maximum entropy value the belief is discarded.

4.1. Interaction Protocol

An *agreement* is a pair of commitments $\delta_{X_0:X_\omega}(x_0, x_\omega)$ between an agent X_0 and another agent X_ω, where x_0 is X_0's commitment and x_ω is X_ω's commitment. $\mathcal{A} = \{\delta_i\}_{i=1}^{D}$ is the agreement set — ie: the set of all possible agreements. If the context is clear then the subscript "$X_0 : X_\omega$" is omitted. These commitments may involve multiple issues — not simply a single issue such as time to complete a task. The set of *terms*, $\mathcal{T}$, is the set of all possible commitments that could occur in an agreement $a \in \mathcal{A}$.

An agent may have a real-valued *utility* function: $\mathbf{U} : \mathcal{T} \rightarrow \Re$, that induces an ordering on $\mathcal{T}$. For such an agent, for any agreement $\delta = (x_0, x_\omega)$ the expression $\mathbf{U}(x_\omega) - \mathbf{U}(x_0)$ is called the *surplus* of δ, and is denoted by $\mathbf{L}(\delta)$ where $\mathbf{L} : \mathcal{T} \times \mathcal{T} \rightarrow \Re$. For example, the values of the function $\mathbf{U}$ may expressed in units of time. It may not be possible to specify the utility function either precisely or with certainty. This is addressed in Sec. 5.

The agents communicate using sentences in a first-order language $\mathcal{C}$: $\mathrm{Delegate}(\cdot)$, $\mathrm{Join}(\cdot)$, $\mathrm{Accept}(\cdot)$, $\mathrm{Reject}(\cdot)$, $\mathrm{Inform}(\cdot)$ and $\mathrm{Quit}(\cdot)$. $\mathrm{Delegate}((X_0, \rho), (X_\omega, G_i))$ means "X_0 proposes to recompense X_ω with ρ if X_ω agrees to take responsibility for an individual goal G_i". $\mathrm{Join}((X_0, \rho), (X_\omega, G_i))$ means "X_0 proposes to recompense X_ω with ρ if X_ω agrees to contribute to cooperative goal G_j". $\mathrm{Accept}(\delta)$ means "the sender accepts your proposed agreement δ". $\mathrm{Reject}(\delta)$ means "the sender rejects your proposed agreement δ". $\mathrm{Inform}((X_0, \mathcal{I}_k), X_\omega)$ means "X_0 offers information $\mathcal{I}_k$ to X_ω". $\mathrm{Quit}(\cdot)$ means "the sender quits — the interaction ends". So for these predicates, and in this discussion, an agreement δ has the form $((X_0, \rho), (X_\omega, G_i))$.

The communication predicates described in the previous paragraph introduce a number of concepts. In the interest of brevity these are only described here informally. The notion of one agent recompensing another [i.e. ρ] refers to both the informal "thanks, I owe you one", and to the formal "take the rest of the day off", or some sum of money. An *individual goal* has the form of: information $\mathcal{I}_k$ will be sent to agent X_r by time t. An *cooperative goal* has the form of: the assembly of information $\mathcal{I}_k$ will be coordinated by agent X_0 by time t. The expression of the information requires some *ontology* — that is not described here.

4.2. Agent Architecture

X_0 uses the language $\mathcal{C}$ for external communication, and the language $\mathcal{L}$ for internal representation. One predicate in $\mathcal{L}$ is: $\mathrm{Acc_d}(((X_0, \rho), (X_\omega, G_i)))$. The proposition $(\mathrm{Acc_d}(\delta) \mid \mathcal{I}_t)$ means: "X_0 will be comfortable accepting the delegation agreement δ with agent X_ω given that X_0 knows information $\mathcal{I}_t$ at time t". The idea is that X_0 will accept delegation agreement δ if $\mathbf{P}(\mathrm{Acc_d}(\delta) \mid \mathcal{I}_t) \geq \alpha$ for some threshold constant α. The precise meaning that X_0 gives to $\mathrm{Acc_d}(\delta)$ is described in Sec. 5. Similarly $\mathrm{Acc_j}$ for $\mathrm{Join}(\cdot)$ agreements. The probability distribution $\mathbf{P}(\mathrm{Agg_d}((X_0, \rho), (X_\omega, G_i)))$ is agent X_0's estimate of the probability that agent X_ω will agree to the $\mathrm{Delegate}$ agreement δ [or $\mathrm{Agg_j}(\cdot)$ for $\mathrm{Join}(\cdot)$ agreements] — it is estimated in Sec. 6.

Each incoming message M from source S received at time t is time-stamped and source-stamped, $M_{[S,t]}$, and placed in an *in box*, $\mathcal{X}$, as it arrives. X_0 has an *information repository* $\mathcal{I}$, a *knowledge base* $\mathcal{K}$ and a *belief set* $\mathcal{B}$. Each of these three sets contains statements in a first-order language $\mathcal{L}$. $\mathcal{I}$ contains statements in $\mathcal{L}$ together with sentence probability functions of time. $\mathcal{I}_t$ is the state of $\mathcal{I}$ at time t and may be inconsistent. At some particular time t, $\mathcal{K}_t$ contains statements that X_0 believes are true at time t, such as

$\forall x(\mathrm{Accept}(x) \leftrightarrow \neg\mathrm{Reject}(x))$. The belief set $\mathcal{B}_t = \{\beta_i\}$ contains statements that are each qualified with a *given sentence probability*, $\mathbf{B}(\beta_i)$, that represents X_0's belief in the truth of the statement at time t. The distinction between the knowledge base $\mathcal{K}$ and the belief set $\mathcal{B}$ is simply that $\mathcal{K}$ contains unqualified statements and $\mathcal{B}$ contains statements that are qualified with sentence probabilities. $\mathcal{K}$ and $\mathcal{B}$ play different roles in the method described in Sec. 4.3; $\mathcal{K}_t \cup \mathcal{B}_t$ is required by that method to be consistent.

X_0's actions are determined by its "strategy". A *strategy* is a function $\mathbf{S} : \mathcal{K} \times \mathcal{B} \rightarrow \mathcal{A}$ where $\mathcal{A}$ is the set of actions. At certain distinct times the function $\mathbf{S}$ is applied to $\mathcal{K}$ and $\mathcal{B}$ and the agent does something. The set of actions, $\mathcal{A}$, includes sending $\mathrm{Delegate}(\cdot)$, $\mathrm{Join}(\cdot)$, $\mathrm{Accept}(\cdot)$, $\mathrm{Reject}(\cdot)$, $\mathrm{Inform}(\cdot)$ and $\mathrm{Quit}(\cdot)$ messages to X_ω. The way in which $\mathbf{S}$ works is described in Secs. 6. Two "instants of time" before the $\mathbf{S}$ function is activated, an "import function" and a "revision function" are activated. The import function $\mathbf{I} : (\mathcal{X} \times \mathcal{I}_{t^-}) \rightarrow \mathcal{I}_t$ clears the in-box, using its "import rules". An import rule takes a message M, written in language $\mathcal{C}$, and from it derives sentences written in language $\mathcal{L}$ to which it attaches decay functions, and adds these sentences together with their decay functions to $\mathcal{I}_{t^-}$ to form $\mathcal{I}_t$. These decay functions are functions of the message type, the time the message arrived and the source from which it came — an illustration is given below. An *import rule* has the form: $\mathbf{P}(S \mid M_{[X_\omega,t]}) = f(M, X_\omega, t) \in [0, 1]$, where S is a statement, M is a message and f is the decay function. Then the belief revision function $\mathbf{R} : \mathcal{I}_{t^-} \rightarrow (\mathcal{I}_t \times \mathcal{K}_t \times \mathcal{B}_t)$ deletes any statements in $\mathcal{I}_{t^-}$ whose sentence probability functions have a value that is ≈ 0.5 at time t. From the remaining statements $\mathbf{R}$ selects a consistent set of statements, instantiates their sentence probability functions to time t, and places the unqualified statements from that set in $\mathcal{K}_t$ — the qualified statements, together with their sentence probabilities, are placed in $\mathcal{B}_t$.

X_0 uses three things to construct proposals: an estimate of the likelihood that X_ω will accept any agreement [Sec. 6], an estimate of the likelihood that X_0 will, in hindsight, feel comfortable accepting any particular agreement [Sec. 5], and an estimate of when X_ω may quit and leave the interaction — see [2].

4.3. Inference

X_0 employs maximum entropy inference and minimum relative entropy inference to derive expectations of future performance from prior, sparse observations. Let $\mathcal{G}$ be the set of all positive ground literals that can be constructed using the symbols in $\mathcal{L}$. A *possible world* is a valuation function $\mathcal{V} : \mathcal{G} \rightarrow \{\top, \bot\}$. $\mathcal{V}$ denotes the set of all possible worlds, and $\mathcal{V}_\mathcal{K}$ denotes the set of possible worlds that are consistent with a knowledge base $\mathcal{K}$ [7].

A *random world* for $\mathcal{K}$ is a probability distribution $\mathcal{W}_\mathcal{K} = (p_i)$ over $\mathcal{V}_\mathcal{K} = (\mathcal{V}_i)$, where $\mathcal{W}_\mathcal{K}$ expresses an agent's degree of belief that each of the possible worlds is the actual world. The *derived sentence probability* of any $\sigma \in \mathcal{L}$, *with respect to* a random world $\mathcal{W}_\mathcal{K}$ is $(\forall \sigma \in \mathcal{L})$:

$$\mathbf{P}_{\mathcal{W}_\mathcal{K}}(\sigma) \triangleq \sum_n \{\, p_n \;:\; \sigma \; is \top \; in \; \mathcal{V}_n \,\} \tag{1}$$

A random world $\mathcal{W}_\mathcal{K}$ is *consistent* with the agent's beliefs $\mathcal{B}$ if: $(\forall \beta \in \mathcal{B})(\mathbf{B}(\beta) = \mathbf{P}_{\mathcal{W}_\mathcal{K}}(\beta))$. That is, for each belief its derived sentence probability as calculated using Eqn. 1 is equal to its given sentence probability.

The *entropy* of a discrete random variable X with probability mass function $\{p_i\}$ is [12]: $\mathbf{H}(X) = -\sum_n p_n \log p_n$ where: $p_n \geq 0$ and $\sum_n p_n = 1$. Let $\mathcal{W}_{\{\mathcal{K},\mathcal{B}\}}$ be the "maximum entropy probability distribution over $\mathcal{V}_\mathcal{K}$ that is consistent with $\mathcal{B}$". Given an agent with $\mathcal{K}$ and $\mathcal{B}$, *maximum entropy inference* states that its *derived sentence probability* for any sentence, $\sigma \in \mathcal{L}$, is:

$$(\forall \sigma \in \mathcal{L})\mathbf{P}(\sigma) \triangleq \mathbf{P}_{\mathcal{W}_{\{\mathcal{K},\mathcal{B}\}}}(\sigma) \tag{2}$$

Using Eqn. 2, the derived sentence probability for any belief, β_i, is equal to its given sentence probability. So the term *sentence probability* is used without ambiguity.

If X is a discrete random variable taking a finite number of possible values $\{x_i\}$ with probabilities $\{p_i\}$ then the *entropy* is the average uncertainty removed by discovering the true value of X, and is given by $\mathbf{H}(X) = -\sum_n p_n \log p_n$. The direct optimization of $\mathbf{H}(X)$ subject to a number, θ, of linear constraints of the form $\sum_n p_n g_k(x_n) = \overline{g}_k$ for given constants $\overline{g}_k$, where $k = 1, \ldots, \theta$, is a difficult problem. Fortunately this problem has the same unique solution as the *maximum likelihood problem* for the Gibbs distribution. The solution to both problems is given by:

$$p_n = \frac{\exp\Big(-\sum_{k=1}^{\theta} \lambda_k g_k(x_n)\Big)}{\sum_m \exp\Big(-\sum_{k=1}^{\theta} \lambda_k g_k(x_m)\Big)} \tag{3}$$

$n = 1, 2, \cdots$ where the constants $\{\lambda_i\}$ may be calculated using Eqn. 3 together with the three sets of constraints: $p_n \geq 0$, $\sum_n p_n = 1$ and $\sum_n p_n g_k(x_n) = \overline{g}_k$. The distribution in Eqn. 3 is known as *Gibbs distribution*.

Given a prior probability distribution $\underline{q} = (q_i)_{i=1}^n$ and a set of constraints, the *principle of minimum relative entropy* chooses the posterior probability distribution $\underline{p} = (p_i)_{i=1}^n$ that has the least relative entropy with respect to $\underline{q}$:

$$\arg\min_{\underline{p}} \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} \tag{4}$$

and that satisfies the constraints. The principle of minimum relative entropy is a generalization of the principle of max-

imum entropy. If the prior distribution $\underline{q}$ is uniform, the relative entropy of $\underline{p}$ with respect to $\underline{q}$ differs from $-\mathbf{H}(\underline{p})$ only by a constant. So the principle of maximum entropy is equivalent to the principle of minimum relative entropy with a uniform prior distribution.

5. Acceptability of a Proposal.

Why would X_0 accept a $\mathrm{Delegate}(\cdot)$ or a $\mathrm{Join}(\cdot)$ proposal? Each deal, $\delta = ((X_0, \rho), (X_\omega, G_i))$, contains provision for an incentive ρ. However it is more realistic [16] to assume that the agents in an emergent process management system are benevolent [8] — that is, they will accept a responsibility for a process if they believe that they can achieve the process goal. So X_0 needs machinery to estimate the probability that if it takes responsibility for goal G_i then it will achieve it. Sec. 6 considers the converse problem: that is, how X_0 estimates the probability distribution over all possible responses that X_ω will respond in various ways.

The proposition $(\mathrm{Acc_d}((X_0, \rho), (X_\omega, G_i)) \mid \mathcal{I}_t)$ was introduced in Sec. 4.2. This section describes how the agent estimates $\mathbf{P}(\mathrm{Acc_d}(\delta) \mid \mathcal{I}_t)$ — i.e. the probability that X_0 attaches to the truth of this proposition for various δ. This is described for delegations only — $\mathrm{Join}(\cdot)$ is dealt with similarly.

X_0 forms its future expectations on the basis of past observations, including the expectations that it has about itself. Sec. 6 following describes how X_0 forms its expectations about a collaborator. The same approach is used estimate $\mathbf{P}(\mathrm{Acc_d}((X_0, \rho), (X_\omega, G_i)) \mid \mathcal{I}_t)$ — the integrity of past observations is continually discounted, new observations are fed in using minimum relative entropy inference — Eqn. 4. This yields a probability distribution over all possible outcomes that could occur if X_0 were to commit to a $\mathrm{Delegate}(\cdot)$ proposal. X_0 then uses this distribution to decide whether or not to commit on the basis of the simple criterion: $\mathbf{P}(\mathrm{Acc_d}((X_0, \rho), (X_\omega, G_i)) \mid \mathcal{I}_t) > \alpha$ for some personal 'comfort factor' α. The details of how this probability distribution is derived is the same as for $\mathrm{Agg_d}(\cdot)$ — this is described following.

6. Interaction

X_0 interacts with its collaborators $\{X_i\}_{i=1}^n$. It is assumed that goals are initially triggered externally to the system. For example, X_0's 'owner' may have an idea that she believes has value, and triggers an emergent process to explore the idea's worth. The interaction protocol is simple, if X_0 sends a $\mathrm{Delegate}(\cdot)$ or a $\mathrm{Join}(\cdot)$ message to X_ω then an interaction has commenced and continues until one agent sends an $\mathrm{Accept}(\cdot)$ or a $\mathrm{Quit}(\cdot)$ message. This assumes that agents respond in reasonable time which is fair in an essentially cooperative system.

To support the agreement-exchange process, X_0 has do two different things. First, it must respond to proposals received from X_ω — that is described in Sec. 5. Second, it must construct proposals, and possibly information, to send to X_ω — that is described now. Maximum entropy inference is used to 'fill in' missing values with the "maximally noncommittal" probability distribution. To illustrate this suppose that X_0 proposes to delegate a process to X_ω. This process involves X_ω delivering — using an $\mathrm{Inform}(\cdot)$ message — u chapters for a report in so-many days v. This section describes machinery for estimating the probabilities $\mathbf{P}(\mathrm{Agg_d}((X_0, u), (X_\omega, G_v))$ where the predicate $\mathrm{Agg_d}((X_0, u), (X_\omega, G_v))$ means "X_ω will accept X_0's delegation proposal $((X_0, u), (X_\omega, G_v))$".

X_0 assumes the following two preference relations for X_ω, and $\mathcal{K}$ contains:
$\kappa_{11}: \quad \forall x, y, z((x < y) \rightarrow$
$(\mathrm{Agg_d}((X_0, y), (X_\omega, G_z)) \rightarrow \mathrm{Agg_d}((X_0, x), (X_\omega, G_z))))$
$\kappa_{12}: \quad \forall x, y, z((x < y) \rightarrow$
$(\mathrm{Agg_d}((X_0, z), (X_\omega, G_x)) \rightarrow \mathrm{Agg_d}((X_0, z), (X_\omega, G_y))))$
As noted in Sec. 4.3, these sentences conveniently reduce the number of possible worlds. The two preference relations κ_{11} and κ_{12} induce a partial ordering on the sentence probabilities in the $\mathbf{P}(\mathrm{Agg_d}((X_0, u), (X_\omega, G_v)))$ array. There are fifty-one possible worlds that are consistent with $\mathcal{K}$.

Suppose that X_0 has the following historical data on similar dealings with X_ω. Three months ago X_ω asked for ten days to deliver four chapters. Two months ago X_0 proposed one day to deliver three chapters and X_ω refused. One month ago X_ω asked for eight days to deliver two chapters. $\mathcal{B}$ contains:
$\beta_{11} : \mathrm{Agg_d}((X_0, 4), (X_\omega, G_{10}))$;
$\beta_{12} : \mathrm{Agg_d}((X_0, 3), (X_\omega, G_1))$ and
$\beta_{13} : \mathrm{Agg_d}((X_0, 2), (X_\omega, G_8))$,
and assuming a 10% decay in integrity for each month: $\mathbf{P}(\beta_{11}) = 0.7$, $\mathbf{P}(\beta_{12}) = 0.2$ and $\mathbf{P}(\beta_{13}) = 0.9$

Eqn. 3 is used to calculate the distribution $\mathbf{W}_{\{\mathcal{K},\mathcal{B}\}}$ which shows that there are just five different probabilities in it. The probability matrix for the proposition $\mathrm{Agg_d}((X_0, u), (X_\omega, G_v))$ is:

$v \diagdown u$	1	2	3	4	5
11	0.9967	0.9607	0.8428	0.7066	0.3533
10	0.9803	0.9476	0.8330	**0.7000**	0.3500
9	0.9533	0.9238	0.8125	0.6828	0.3414
8	0.9262	**0.9000**	0.7920	0.6655	0.3328
7	0.8249	0.8019	0.7074	0.5945	0.2972
6	0.7235	0.7039	0.6228	0.5234	0.2617
5	0.6222	0.6058	0.5383	0.4523	0.2262
4	0.5208	0.5077	0.4537	0.3813	0.1906
3	0.4195	0.4096	0.3691	0.3102	0.1551
2	0.3181	0.3116	0.2846	0.2391	0.1196
1	0.2168	0.2135	**0.2000**	0.1681	0.0840

In this array, the derived sentence probabilities for the three

sentences in $\mathcal{B}$ are shown in bold type; they are exactly their given values.

X_0's *interaction strategy* is a function $\mathbf{S} : \mathcal{K} \times \mathcal{B} \rightarrow \mathcal{A}$ where $\mathcal{A}$ is the set of actions that send $\mathrm{Delegate}(\cdot)$, $\mathrm{Join}(\cdot)$, $\mathrm{Accept}(\cdot)$, $\mathrm{Reject}(\cdot)$, $\mathrm{Inform}(\cdot)$ and $\mathrm{Quit}(\cdot)$ messages to X_ω. If X_0 sends any message to X_ω then she is giving X_ω information about herself.

6.1. An 'even-handed' agent

An agent may be motivated to act for various reasons — three are mentioned. First, if there are costs involved in the interaction due *either* to changes in the value of the interaction object with time *or* to the intrinsic cost of conducting the interaction itself. Second, if there is a risk of breakdown caused by a collaborator dropping out of a negotiation. Third, if the agent is concerned with establishing a sense of trust [13] with the collaborator — this could be the case in the establishment of a business relationship. Of these three reasons the last two are addressed here. The risk of breakdown may be reduced, and a sense of trust may be established, if the agent appears to its collaborator to be "approaching the interaction in an even-handed manner". One dimension of "appearing to be even-handed" is to be equitable with the value of information given to the collaborator. Various interaction strategies, both with and without breakdown, are described in [2], but they do not address this issue. An interaction strategy is described here that is founded on a principle of "equitable information gain". That is, X_0 attempts to respond to X_ω's messages so that X_ω's expected information gain similar to that which X_0 has received.

X_0 models X_ω by observing her actions, and inferring beliefs about her future actions in probability distributions such as $\mathbf{P}(\mathrm{Agg_d})$. X_0 measures the value of information that it receives from X_ω by the change in the entropy of this distribution as a result of representing that information in $\mathbf{P}(\mathrm{Agg_d})$. More generally, X_0 measures the value of information received in a message, μ, by the change in the entropy in its entire representation, $\mathcal{J}_t = \mathcal{K}_t \cup \mathcal{B}_t$, as a result of the receipt of that message; this is denoted by: $\Delta_\mu|\mathcal{J}_t^\Pi|$, where $|\mathcal{J}_t^\Pi|$ denotes the value (as negative entropy) of X_0's information in $\mathcal{J}$ at time t. Although both X_0 and X_ω will build their models of each other using the same data — the messages exchanged — the observed information gain will depend on the way in which each agent has represented this information. It is "not unreasonable to suggest" that these two representations should be similar. To support its attempts to achieve "equitable information gain" X_0 assumes that X_ω's reasoning apparatus mirrors its own, and so is able to estimate the change in X_ω's entropy as a result of sending a message μ to X_ω: $\Delta_\mu|\mathcal{J}_t^\Omega|$. Suppose that X_0 receives a message $\mu = \mathrm{Delegate}(\cdot)$ from X_ω and observes an information gain of $\Delta_\mu|\mathcal{J}_t^\Pi|$. Suppose that X_0 wishes to reject this agreement by sending a counter-proposal, $\mathrm{Delegate}(\cdot)$, that will give X_ω expected "equitable information gain". $\delta = \{\arg\max_\delta \mathbf{P}(\mathrm{Acc_d}(\delta) \mid \mathcal{I}_t) \geq \alpha \mid (\Delta_{\mathrm{Delegate}(\delta)}|\mathcal{J}_t^\Omega| \approx \Delta_\mu|\mathcal{J}_t^\Pi|)\}$. That is X_0 chooses the most acceptable agreement to herself that gives her collaborator expected "equitable information gain" provided that there is such an agreement. If there is not then X_0 chooses the best available compromise $\delta = \{\arg\max_\delta(\Delta_{\mathrm{Delegate}(\delta)}|\mathcal{J}_t^\Omega|) \mid \mathbf{P}(\mathrm{Acc_d}(\delta) \mid \mathcal{I}_t) \geq \alpha\}$ provided there is such an agreement — this strategy is rather generous, it rates information gain ahead of personal acceptability. If there is not then X_0 quits.

7. Delegation

The mechanism that X_0 uses for managing process delegation is described in full. $\mathrm{Join}(\cdot)$ messages are managed similarly. This next section discusses the sorts of payoff measures and estimates that are available, and that are combined to give a value for the expected payoff vector $\underline{\nu_i}$ for each agent. Let $\mathbf{P}(A \gg)$ denote A is the 'best choice' in terms of some combination of the parameter estimates described following. These measurements are then used by agent X_0 to determine $\mathbf{P}(X_i \gg)$, and then in turn to determine the delegation strategy $(p_i)_{i=1}^n$.

7.1. The Performance Parameters

Agent X_0 continually measures the performance of itself and of other agents in the system using four measures. Three are: *time*, *cost* and *likelihood of success* which are attached to all of its delegations-in and delegations-out. The last one is a *value* parameter that is attached to other agents. Time is the total time taken to termination. Cost is the actual cost of the of resources allocated. For example, the time that the agent — possibly with a human 'assistant' – actually spent working on that process. The likelihood of success is the probability that an agent will deliver its response within its constraints. The value parameter is the value added to a process by an agent. Unfortunately, value is often very difficult to measure — it is treated here by a subjective estimate delivered by users of the system.

The three parameters *time*, *cost* and likelihood of *success* are observed and recorded every time an agent, including X_0, delivers, or fails to deliver, its commitments. This generates a large amount of data whose significance can reasonably be expected to degrade over time. So a cumulative estimate only is retained. The integrity of information 'evaporates' as time goes by. If we have the set of observable outcomes as $O = \{o_1, o_2, \ldots, o_m\}$ then complete ignorance of the expected outcome means that our expectation over these outcomes is $\frac{1}{m}$ — i.e. the unconstrained

maximum entropy distribution. This natural decay of information integrity is offset by new observations.

Given one of the parameters, u, with m possible outcomes[1], suppose that $P^t(u' \mid \delta)$ is the estimate at time t of the probability that the actual outcome u' will be observed given that the agent being observed has committed to δ. Suppose that X_0 observes the actual outcome r, on the basis of this outcome X_0 believes that the probability of r being observed at the next time is g_r. Then let $P^t_{g_r}(u' \mid \delta)$ be the posterior minimum relative entropy distribution calculated using Eqn. 4 with prior distribution $P^t(u' \mid \delta)$ and satisfying the constraint that $P^t_{g_r}(r \mid \delta) = g_r$. Then update $P^t(u' \mid \delta)$ with:

$$P^{t+1}(u' \mid \delta) = \frac{1-\rho}{n} + \rho \cdot P^t_{g_r}(u' \mid \delta) \qquad (5)$$

This equation determines the development of $P^t(u' \mid \delta)$ for some large $\rho \in [0, 1]$.

X_0 uses the method in Eqn. 5 to update its estimates for all probability distributions representing each of the agents that it deals with. For example, if $P^t(\cdot)$ is X_0's estimate of the time that X_ω will take to deliver on a particular type of agreement. Suppose that at time t, X_ω delivers her response after having taken time u. Then X_0 attaches a belief (i.e. a sentence probability) to the proposition that this is how X_ω will behave at time $t+1$. This becomes the constraint in the minimum relative entropy calculation and then Eqn. 5 gives $P^{t+1}(\cdot)$.

The *process delegation problem* belongs to the class of resource allocation games which are inspired by the 'El Farol Bar' problem — see [6] for recent work.

7.2. Choosing the 'best' collaborator

The probability distributions described above may be used to determine the probability that one agent is a better choice than another by calculating the probability that one random variable is greater than another in the usual way. This method may be extended to estimate the probability that one agent is a better choice than a number of other agents. For example, if there are three agents to choose from, A, B, and C, then:

$\mathbf{P}(A \gg) = \mathbf{P}((A \gg B) \wedge (A \gg C))$
$= \mathbf{P}(A \gg B) \times \mathbf{P}((A \gg C) \mid (A \gg B))$

The difficulty with this expression is that there is no direct way of estimating the second, conditional probability. This expression shows that:

$\mathbf{P}(A \gg B) \times \mathbf{P}(A \gg C) \leq \mathbf{P}(A \gg) \leq \mathbf{P}(A \gg B)$

By considering the same expression with B and C interchanged:

$\mathbf{P}(A \gg B) \times \mathbf{P}(A \gg C) \leq \mathbf{P}(A \gg)$
$\mathbf{P}(A \gg) \leq \min[\mathbf{P}(A \gg B), \mathbf{P}(A \gg C)]$

So for some $\tau_A \in [0, 1]$:

$\mathbf{P}(A \gg) = \mathbf{P}(A \gg B) \times \mathbf{P}(A \gg C) +$
$\tau_A \times [\min[\mathbf{P}(A \gg B), \mathbf{P}(A \gg C)] -$
$\mathbf{P}(A \gg B) \times \mathbf{P}(A \gg C)]$

Similar expressions may be constructed for the probabilities that B and C are the best agents respectively. This is as far as probability theory can go without making some assumptions. To proceed assume that:$\tau_A = \tau_B = \tau_C = \tau$; this assumption is unlikely to be valid, but it should not be "too far" from reality. Either A or B or C will be the best plan, so the sum of the three expressions for the probabilities of A, B and C being the "best" plan will be unity. Hence:

$\tau = \frac{1-d}{q-d}$ where:

$d = [(\mathbf{P}(A \gg B) \times \mathbf{P}(A \gg C)) +$
$(\mathbf{P}(B \gg C) \times \mathbf{P}(B \gg A)) + (\mathbf{P}(C \gg A) \times \mathbf{P}(C \gg B))]$

$q = [\min[\mathbf{P}(A \gg B), \mathbf{P}(A \gg C)] +$
$\min[\mathbf{P}(B \gg C), \mathbf{P}(B \gg A)] +$
$\min[\mathbf{P}(C \gg A), \mathbf{P}(C \gg B)]]$

This expression for τ is messy but is easy to calculate. The probability that each of the three agents A, B and C is the "best" choice is $\mathbf{P}(A \gg)$, $\mathbf{P}(B \gg)$ and $\mathbf{P}(C \gg)$. An alternative to the above is simply to use Eqn. 1 to estimate the probability of the propositions that each of the agents is the 'best' collaborator. This alternative approach involves a maximum entropy calculation whereas the above approach does not.

Agent X_0 will choose the g_0 function to reflect its own preferences and to reflect the nature of the process for which responsibility is being delegated.

7.3. Delegation Strategy

A delegation strategy is a probability distribution $\{p_i\}_{i=1}^n$ that determines who from $\{X_i\}_{i=1}^n$ to offer responsibility to for doing what. A delegation strategy has the properties:

$$\begin{array}{l} \textit{if } \mathbf{P}(X_i \gg) > \mathbf{P}(X_j \gg) \textit{ then } p_i > p_j \\ \textit{if } \mathbf{P}(X_i \gg) = \mathbf{P}(X_j \gg) \textit{ then } p_i = p_j \\ p_i > 0 (\forall i) \text{ and } \sum_i p_i = 1 \end{array}$$

The delegation strategy achieves this stochastically by determining instead n probabilities $(p_1, \ldots, p_n)$ where p_i is the probability that the i'th agent will be selected, and $\sum_i p_i = 1$. The choice of the agent to delegate to is then made with these probabilities. The expression of the delegation strategy in terms of probabilities enables the strategy to balance conflicting goals, such as achieving process quality and process efficiency.

$\mathbf{P}(X_i \gg)$ is the probability that X_i is the 'best' choice. A strategy that continually chose the 'best' on the basis of historic data is flawed because an agent who "goes through

1 The *success* parameter has only two possible outcomes 'succeed' and 'fail'.

a bad patch" may never be chosen — this means that if an agent wants "the quiet life" all is would have to do is make a series of mistakes. So the delegation strategy chooses agents with probability $p_i = \mathbf{P}(X_i \gg)$. That is, the probability that X_0 will attempt to delegate a process to X_ω is equal to the probability that X_0 estimates X_ω to be the 'best' choice for the job.

8. Conclusion

Emergent processes are collaborative business processes whose execution is determined by the prior knowledge of the agents involved and by the knowledge that emerges during a process instance. In an emergent process, the process goal may mutate, and so does not provide clear direction for process management. As emergent processes may involve informal interaction, there is a limit to the extent to which the processes *per se* can be "managed". However, the *collaboration* can be managed. The solution proposed builds on ideas from information theory and entropy-based inference. These inference methods are logic-based and so operate with multi-issue interaction with ease — this is particularly significant for the interactions involved in these high-level processes. The establishment of a sense of trust contributes to the establishment of business relationships and to preventing breakdown during interaction. This is addressed by the agents attempting to exhibit 'fair play' by applying the principle of equitable information revelation.

To manage collaboration the agent is equipped with $\mathrm{Delegate}(\cdot)$, $\mathrm{Join}(\cdot)$, $\mathrm{Accept}(\cdot)$, $\mathrm{Reject}(\cdot)$, $\mathrm{Inform}(\cdot)$ and $\mathrm{Quit}(\cdot)$ interaction predicates. This discussion has focussed on the $\mathrm{Delegate}(\cdot)$ predicate. $\mathrm{Join}(\cdot)$ is dealt with similarly. $\mathrm{Inform}(\cdot)$ is used to satisfy a delegation goal, and the remaining predicates are necessary to support the interaction.

The agents in the system are 'essentially benevolent' — they do no necessarily require motivation to contribute to a collaborative group or to take responsibility for a subprocess. Despite this, the agents also have a responsibility to their own user. So our agent does not attempt to second-guess the motives of the other agents in the system. Instead it takes advantage of the large amount of readily available information concerning past performance to estimate, using maximum entropy methods, expectations about future performance. The information in the system is based on past observations and so its integrity is in a permanent state of decay [1]. The agent selects its collaborators from the system by using a stochastic strategy. This strategy identifies a collaborator with a probability that is equal to the agent's estimate that she is the 'best' choice. This strategy provides a reasonable balance between getting things done in the best way and spreading the work around — thus ensuring that the agent always has performance expectations for a number of potential collaborators.

References

[1] D. Bernhardt and J. Miao. Informed trading when information becomes stale. *The Journal of Finance*, LIX(1), February 2004.

[2] J. Debenham. Bargaining with information. In N. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, editors, *Proceedings Third International Conference on Autonomous Agents and Multi Agent Systems AAMAS-2004*, pages 664 – 671. ACM, July 2004.

[3] P. Dourish. Using metalevel techniques in a flexible toolkit for CSCW applications. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 5(2):109 – 155, June 1998.

[4] M. Esteva, J. Padget, and C. Sierra. Formalizing a languages for institutions and norms. In J. Meyer and M. Tambe, editors, *Intelligent Agents VIII*, pages 348 – 366. Springer-Verlag, Berlin, Germany, 2002.

[5] L. Fischer. *The Workflow Handbook 2003*. Future Strategies Inc., 2003.

[6] A. Galstyan, S. Kolar, and K. Lerman. Resource allocation games with changing resource capacities. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems AAMAS-03*, pages 145 – 152, 2003.

[7] J. Halpern. *Reasoning about Uncertainty*. MIT Press, 2003.

[8] M. Huhns and M. Singh. Managing heterogeneous transaction workflows with cooperating agents. In N. Jennings and M. Wooldridge, editors, *Agent Technology: Foundations, Applications and Markets*, pages 219 – 239. Springer-Verlag: Berlin, Germany, 1998.

[9] A. Jain, M. Aparicio, and M. Singh. Agents for process coherence in virtual enterprises. *Communications of the ACM*, 42(3):62 – 69, 1999.

[10] E. Jaynes. *Probability Theory — The Logic of Science*. Cambridge University Press, 2003.

[11] N. Jennings, P. Faratin, T. Norman, P. O'Brien, and B. Odgers. Autonomous agents for business process management. *Int. Journal of Applied Artificial Intelligence*, 142(2):145 – 189, 2000.

[12] D. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[13] S. Ramchurn, N. Jennings, C. Sierra, and L. Godo. A computational trust model for multi-agent interactions based on confidence and reputation. In *Proceedings 5th Int. Workshop on Deception, Fraud and Trust in Agent Societies*, 2003.

[14] M. Singh. Business Process Management: A Killer Ap for Agents? In N. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, editors, *Proceedings Third International Conference on Autonomous Agents and Multi Agent Systems AAMAS-2004*, pages 26 – 27. ACM, July 2004.

[15] H. Smith and P. Fingar. *Business Process Management (BPM): The Third Wave*. Meghan-Kiffer Press, 2003.

[16] W. van der Aalst and K. van Hee. *Workflow Management: Models, Methods, and Systems*. The MIT Press, 2002.

[17] M. Wooldridge. *Multiagent Systems*. Wiley, 2002.