# Privacy and Utility in Business Processes

Adam Barth
Stanford University
abarth@cs.stanford.edu

Anupam Datta
Carnegie Mellon University
danupam@cmu.edu

John C. Mitchell
Stanford University
mitchell@cs.stanford.edu

Sharada Sundaram
Tata Consultancy Services
sharada.sundaram@tcs.com

## Abstract

*We propose an abstract model of business processes for the purpose of (i) evaluating privacy policy in light of the goals of the process and (ii) developing automated support for privacy policy compliance and audit. In our model, agents that send and receive tagged personal information are assigned organizational roles and responsibilities. We present approaches and algorithms for determining whether a business process design simultaneously achieves privacy and the goals of the organization (utility). The model also allows us to develop a notion of minimal exposure of personal information, for a given process. We investigate the problem of auditing with inexact information and develop methods to identify a set of potentially culpable individuals when privacy is breached. The audit methods draw on traditional causality concepts to reduce the effort needed to search audit logs for irresponsible actions.*

## 1 Introduction

Privacy is an increasingly important business concern in health care, financial services, and other organizations. Hospitals, clinics, banks, credit card clearing houses, customer support centers, and academic institutions all maintain databases with sensitive information. These databases are used regularly by employees to carry out business-critical tasks. Organizations that collect and use personal information face the growing challenge of conducting their business effectively while managing privacy risks and compliance requirements. The risks are very real, with the theft of 26 million veteran records in May 2006 demonstrating how easily sensitive information can fall into unauthorized hands [11]. In the United States, privacy legislation, such as HIPAA [28] for the health care sector and GLBA [18, 16] for financial institutions, has spurred many business, including 68% of the Direct Marketing Association member companies as of 2001 [13], to appoint Chief Privacy Officers whose primary job is privacy issues and policies.

One of the biggest problems that privacy-sensitive organizations face is designing their internal activities and information practices to simultaneously serve their customers effectively and manage risks from disclosure of sensitive information. This fundamental problem arises in hospitals and clinics, where personal health information must be used to provide effective health care, but must also be protected from indiscriminate sharing to respect the privacy of patients—a requirement made more precise by HIPAA. Financial institutions use sensitive financial information to decide whether to grant loans, for example, and suffer direct loss and brand erosion if sensitive information is lost. Retail enterprises use credit card details in resolving charge-back disputes (where the privacy concerns are exacerbated by the common practice of outsourcing this task). College admissions officers review confidential letters of recommendation and transcripts. In all of these situations, the organization must carefully design the way it processes and uses information to balance the competing goals of privacy and the usefulness, or utility, of the business process.

Business process designs involve instructing individuals how and when to access and use information, coupled with access and use policies embedded in information processing systems. Because considering utility or privacy alone does not provide enough information to make meaningful management decisions, our goal is to develop a framework and model for designing, evaluating, and auditing business processes to achieve utility goals while minimizing privacy risks. We propose an abstract model of business processes, utility, and privacy, present some specific results, and illustrate our concepts using MyHealth@Vanderbilt [36], a web-based patient portal built and used at the Vanderbilt Medical Center. Examining the MyHealth portal led to many insights captured in our general theory.

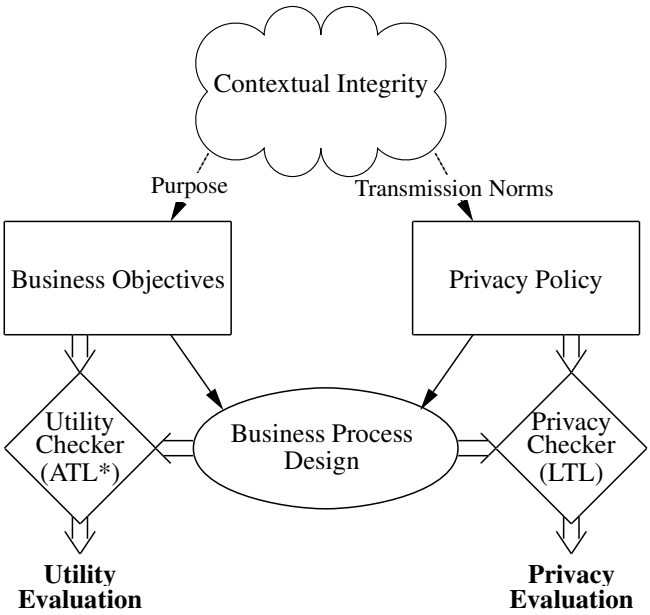


**Figure 1. The design of a business process. A well-designed business process achieves business objectives while complying with privacy policy.**

Our approach builds on *contextual integrity*, a conceptual framework for understanding privacy expectations and their implications developed in the literature on law, public policy, and political philosophy [26]. The primary tenant of contextual integrity is that people interact in society not simply as individuals in an undifferentiated social world, but as individuals in certain capacities or roles, in distinctive social contexts (e.g., health care or banking). For example, the individuals in MyHealth act as patients, doctors, nurses, or secretaries, according to a specific workflow for scheduling appointments, viewing lab results, and asking and answering heath questions.

Each context is characterized by its business objectives, or *utility* goals, and its norms of transmission. For example, one utility goal for MyHealth is to respond to health questions from patients. The norms of transmission identify conditions under which personal information can be communicated from one party to another. These norms are represented by the *privacy* goals of a workflow. A privacy goal for MyHealth is to restrict health information to doctors and nurses, the health care providers.

Using a model of *actions* that transmit personal information from a sender in one role to a receiver in a possibly different role, agents may accumulate and send different types of personal information they receive. These messages represent emails, web forms, database entries, workflow data structures, and arguments to actions. We assume that messages have associated *tags* (e.g., "health information") to indicate their contents, but consider business processes in which human agents may tag messages incorrectly. Since agents may act independently, with different motives, we express privacy and utility goals using a form of alternating-time temporal logic, which we call the *Logic of Privacy and Utility (LPU)*, interpreted over the concurrent game structure [2] of agent actions. In this logical setting, privacy is a trace property expressible in LTL [23], while utility requires that agents have strategies to achieve certain useful outcomes, and is therefore expressed naturally using the stronger ATL* [2] path quantifiers. We also formulate workflows in temporal logic, by associating a *responsibility* to each agent role. For example, in the patient portal workflow, doctors are responsible for answering health questions and secretaries are responsible for scheduling appointments. We consider both a general class of workflows presented abstractly by logical formulas and a more concrete subclass of practical workflows presented as a labeled graph or automata. Within this setting, we formulate and address design-time and run-time questions about whether a given workflow achieve its privacy and utility goals, without assuming that human agents always follow their assigned responsibilities.

1. *Does a given workflow achieve privacy and utility if all agents act responsibly?*
   We present algorithms for answering this question, using the components illustrated in Figure 1. Specifically, privacy properties may be evaluated using standard LTL model-checking over the traces generated by responsible executions of the concurrent game structure. Evaluating utility is more involved because of the ATL* path quantifiers and, in general, is undecidable [2] because agents learn only of messages they send or receive. Because of this limitation, we present a sound decision procedure for a restricted, but useful, class of formula defined.

2. *Can irresponsible agents be detected and held accountable for violations?*
   If the execution of a workflow satisfying our design criteria actually violates privacy, then some agent must have caused the violation by acting irresponsibly. These violations can be caught at run time, and the accountable agent determined using *auditing* algorithms we present (see Fig. 2). These algorithms are not fully automatic (or else they could be used for enforcement), but require an oracle (such as a human auditor) to determine the accuracy of message tags. We seek to minimize the number of oracle calls (reducing the human auditor's work) by using classical causality ideas in distributed computing and a new notion of "suspicious events."

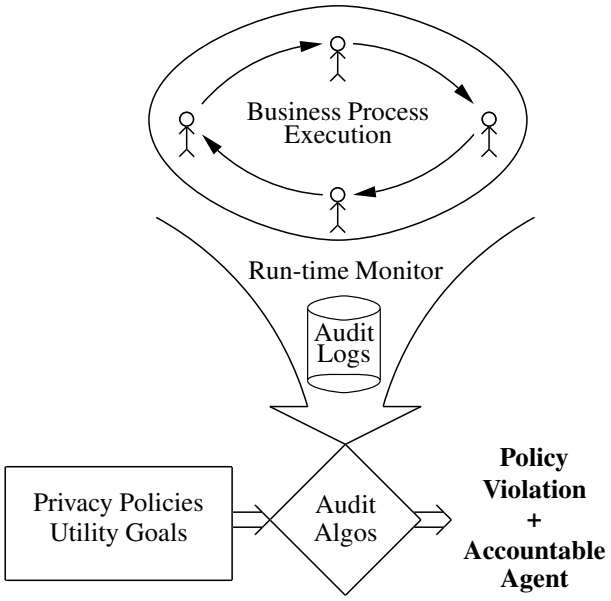


**Figure 2. Auditing logs from business process execution**

Privacy advocates often recommend reconciling the competing interests of privacy and utility with the principle of *minimum necessary disclosure*: disclose the minimum information necessary to achieve the utility goal. This principle is included expressly in several influential privacy policies, including the HIPAA Privacy Rule and the Markle Connecting for Health Common Framework [24]. We leverage our unified model of privacy and utility to provide a formal definition of this principle.

We apply these concepts to the MyHealth patient portal and recommend several design changes to the MyHealth developers at Vanderbilt. Message tags are themselves one such suggestion, enabling finer grained message routing. Our auditing algorithms were developed in response to the MyHealth developers' concern about incorrectly tagged messages. In this paper, we suggest further privacy improvements in the MyHealth workflow based on tagging and illustrate our auditing methods using a hypothetical execution of MyHealth with an irresponsible agent.

After a review of related work, the remainder of the paper is organized as follows. Section 2 describes the MyHealth@Vanderbilt patient portal. Section 3 defines the process model and temporal logic for stating properties of a model. Workflows and responsibilities are explained in Section 4, with results relating privacy and utility in Section 5. Auditing is explored in Section 6, with applications to MyHealth in Section 7 and conclusions in Section 8.

### 1.1 Related Work

The Logic of Privacy and Utility is based on the privacy language CI [9], a formalization of contextual integrity's transmission norms which has received some recent media attention [17]. This work extends CI in two key directions, (i) capturing notions of utility by extending the labeled transition system to a concurrent game structure, and (ii) capturing uncertainty about message contents by permitting senders to attach arbitrary tags to messages. Using these two extensions, we are able to investigate trade-offs between privacy and utility in workflow design, and auditing issues in workflow execution not visible in CI.

Several privacy-only languages have been proposed. P3P [30, 12], built into Internet Explorer, is the most widely-deployed privacy language, but is the least expressive and has some semantic anomalies [10]. P3P also assumes the semantic contents of message are known, and recent work [21] has examined enforcing P3P policies in database systems where the type of information is known explicitly. EPAL [19, 8, 32] is a privacy language designed by IBM to enforce privacy policies within the enterprise. EPAL, like the general access control language XACML [4, 3], defines its semantics in terms of an authorization algorithm, making a utility extension difficult. Finally, Privacy APIs [25], based on HRU access control, is privacy language that can express privacy legislation. Other work [5, 6] formalizes privacy policies used by organizations but does not consider how these policies affect the design of organizational workflow.

Several formalisms have been considered for specifying workflows, most notably Petri Nets [1], UML Activity Diagrams [14], and BPEL [27]. The Petri Net model is useful for understanding reachability and parallelism properties of workflows, but is difficult to apply to privacy because the stones in the net, which represent performing tasks or exchanging messages, are untyped (or, more precisely, typed implicitly by their location). The formalism for UML Activity Diagrams is graphical, complicating integration with the linguistic formalisms of privacy policies. Alternately, BPEL, for which Oracle provides an execution engine [29], views a workflow as conglomeration of web services. Unfortunately, BPEL resembles an imperative programming language and is Turing-complete, foreclosing the possibility of deciding whether a BPEL workflow complies with a privacy policy or achieves a utility goal.

Workflow and authorization have been considered together previously, both for access control [7] and for privacy [34], but those works treat the workflow as given and do not consider the utility goal as a constraint on workflow design. Moreover, neither consider auditing deviations from prescribed workflow execution. Conflicts between privacy and utility goals have been recognized in other set-

tings, such as in $k$-Anonymity [31, 33]: when a database cell is suppressed, privacy is enhanced and utility is diminished. When Dwork and Nissim [15] perturb a dataset to achieve privacy, utility concerns motivate them to minimize the amount of added noise.

## 2 Example: MyHealth@Vanderbilt

The MyHealth patient portal at Vanderbilt University Hospital allows patients to interact with their doctors and other healthcare professionals through a web-based messaging system (see `www.myhealthatvanderbilt.com`). This innovative system at a leading research hospital, like related commercial ventures elsewhere, aims to provide better medical care at reduced costs, in a way that is more convenient to patients. In the MyHealth system, patients ask health questions and receive answers by exchanging messages with their doctors, labs send test results to patients, and patients send appointment requests to secretaries. While HIPAA does not allow certain email communication to patients, patient authentication allows web-based systems to be HIPAA-compliant. Figure 3 depicts a portion of the MyHealth message passing system that deals with appointment requests and health questions.

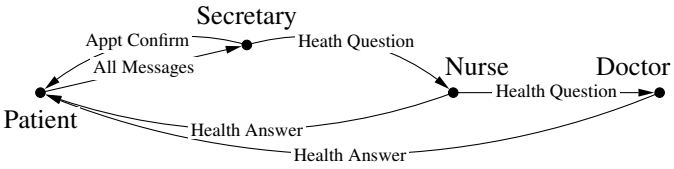


**Figure 3. Current MyHealth Workflow**

This portion of MyHealth serves two utility goals: patients can schedule appointments and can receive answers to health questions. All messages are directed to the secretary, who is responsible for scheduling appointments and forwarding health questions to the nurse. This design reduces the doctor's workload, but the secretary learns the patient's sensitive health information contained in health questions. A more privacy conscious MyHealth design would deliver health questions directly to the nurse, bypassing the secretary. However, patients write messages in free text, making it difficult for MyHealth to route health questions to nurses and appointment requests to secretaries. We suggest permitting patients to tag messages with their contents using a simple drop-down control on the portal, enabling MyHealth to route message according to their tag. Figure 4 depicts our proposed tag-based workflow for MyHealth.
Our initial workflow proposal did not permit communication between the secretary and the nurse, but that workflow

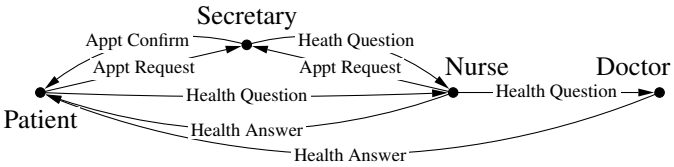


**Figure 4. Proposed MyHealth Workflow**

failed to achieve a utility goal. Vanderbilt employees would have been unable to answer health questions if a patient mistakenly tagged a health question as an appointment request as the secretary would be unable to correct the tag and forward the question to the nurse. We present a more formal analysis of MyHealth and discuss auditing in Sect. 7 after developing the tools in the intervening sections.

## 3 A Logic of Privacy and Utility

In this section, we develop a logic for reasoning about privacy and utility. The logic is based on the CI language from [9], but extended in two ways. First, the model has operational semantics describing how agents interact to produce traces. The executions of a system are represented using a concurrent game structure [2], and the logic has a corresponding "strategy quantifier" that allows selective quantification over execution paths. This quantifier is useful for expressing utility goals. Second, by distinguishing between the actual contents of a message and the tags attached to it, we need not assume that mechanical agents can determine the actual contents of messages, an assumption that fails for systems like MyHealth that process free text messages.

### 3.1 Model of Agents and Communication

We begin with a model of communicating agents, similar to the CI model [9], in which agents send each other messages containing personal information about each other. Upon receipt of a message, an agent incorporates the contents of the message into his or her knowledge state, enlarging the set of possible messages the agent can send in the future.

**Agents.** Let $\mathcal{P}$ be a set of *agents*, $\mathcal{M}$ be a set of *messages*, and $\mathcal{T}$ be a set of *attributes* equipped with a partial order $\preceq$. For attributes $t_1$ and $t_2$, if $t_1 \preceq t_2$, then $t_1$ can be computed from $t_2$. For example, *postal-code* $\preceq$ *mailing-address*. Associated with each message $m \in \mathcal{M}$ is a set $\mathrm{contents}(m) \subseteq \mathcal{P} \times \mathcal{T}$, indicating what attributes about which agents are actually contained in the message. We extend the CI model by associating a set $\mathrm{tags}(m) \subseteq \mathcal{P} \times \mathcal{T}$

with each message $m$, indicating the tags carried by the message. The tags of a message are the purported contents of the message and need not bear any relation to the actual contents of the message. Human agents can ascertain the actual contents of the message, whereas mechanical agents have access only to the tags. $\mathsf{Send}(p, q, m)$ is an *action* where $p, q \in \mathcal{P}$ are agents and $m \in \mathcal{M}$ is a message. The action $\mathsf{Send}(p, q, m)$ occurs when agent $p$ sends message $m$ to agent $q$.

**Knowledge.** In order to send $m$, an agent must know the contents of $m$. After receiving $m$, an agent learns the contents of $m$. The *knowledge state* of an agent $p$ is a subset $\kappa_p \subseteq \mathcal{P} \times \mathcal{T}$, indicating the information $p$ has received. The *global knowledge state* $\kappa$ is a function mapping each agent to his or her knowledge state. Associated with each knowledge state is a set of actions *available* to $p$.

$$\mathrm{available}_p(\kappa) = \{\tau\} \cup \{\mathsf{Send}(p, q, m) \mid \mathrm{contents}(m) \subseteq \kappa_p\},$$

where $\tau$ is the null action that does not transmit a message. Notice that agents are free to select arbitrary tags when sending messages. Recipient learn the contents of messages they receive: for every $\mathsf{Send}(p, q, m) \in \mathrm{available}_p(\kappa)$,

$$\kappa \xrightarrow{\mathsf{Send}(p,q,m)} \kappa[q \mapsto \mathrm{cl}_{\preceq}(\kappa_q \cup \mathrm{contents}(m))],$$

where $\mathrm{cl}_{\preceq}(\kappa) = \{(p, t) \mid \exists t'.(p, t') \in \kappa \wedge (t \preceq t')\}$ is the set of attributes computable from $\kappa$. The relation $\xrightarrow{\tau}$ is the identity relation on knowledge states. For every finite set of actions $A = \{a_1, \ldots, a_n\}$ available in $\kappa$, let $\kappa \xrightarrow{A} \kappa'$ if $\kappa \xrightarrow{a_1} \cdots \xrightarrow{a_n} \kappa'$. This is well-defined because $\mathrm{available}_p$ is monotonic on $\rightarrow$ and the resulting knowledge state is independent of the enumeration of $A$. A *trace* is a sequence of sets of actions labeling transitions beginning with an *initial knowledge state*.

**Moves.** We further extend the CI model with operational semantics by embedding the labeled transition system of knowledge states into a concurrent game structure [2], which we refer to as $\mathcal{G}$. If $\mathcal{G}$ is currently in state $\kappa$, each agent selects a *move*, a set of actions $A_p \subseteq \mathrm{available}_p(\kappa)$, and $\mathcal{G}$ advances to the unique knowledge state $\kappa'$ such that

$$\kappa \xrightarrow{\bigcup_p A_p} \kappa'.$$

An action $\mathsf{Send}(p_1, p_2, m)$ is *visible* to agent $p$ if $p$ is the sender, $p_1$, or the recipient, $p_2$. An agent $p$'s *view* of a trace $\pi$ is the subsequence $\pi \restriction p$ containing all and only those actions visible to $p$. Agents are unaware of actions outside their view. Each agent $p$ decides which moves to make according to a *local strategy*, a function from finite $p$-views to moves for $p$. This locality requirement makes $\mathcal{G}$ a game of imperfect information.

## 3.2 Syntax of the Logic

The syntax of the logic is a particular signature of Alternating-time Temporal Logic [2]. The signature is similar to CI [9], but extended with a strategy quantifier $\langle\langle \vec{p} \rangle\rangle \varphi$ to make use of the game structure and a predicate tagged to access the tags of messages. We employ both the contains and tagged predicates to be able to compare the purported and actual contents of messages. We do not need the incontext predicate because the workflows we study are present implicitly in a single context.

$$\begin{aligned}\varphi ::= {} & \mathsf{send}(p, q, m) \mid \mathsf{inrole}(p, r) \mid t_1 \preceq t_2 \mid \\ & \mathsf{contains}(m, p, t) \mid \mathsf{tagged}(m, p, t) \mid \\ & \varphi \wedge \varphi \mid \neg\varphi \mid \varphi \mathbf{U} \varphi \mid \varphi \mathbf{S} \varphi \mid \mathbf{X}\varphi \mid \langle\langle \vec{p} \rangle\rangle \varphi \mid \exists x.\varphi\end{aligned}$$

Informally, the predicate $\mathsf{send}(p, q, m)$ indicates that agent $p$ has just sent agent $q$ a message $m$; $\mathsf{inrole}(p, r)$ indicates that agent $p$ is in role $r$; $\mathsf{contains}(m, p, t)$ indicates that message $m$ contains attribute $t$ about agent $p$; $\mathsf{tagged}(m, p, t)$ indicates that the message $m$ is tagged with tag $t$ about agent $p$; $t_1 \preceq t_2$ indicates that attribute $t_1$ can be computed from attribute $t_2$. We use the standard abbreviations $\wedge$, $\rightarrow$, and $\forall$ as usual in first order logic and $\mathbf{F}\varphi \equiv \mathit{true}\mathbf{U}\varphi$ for "eventually" and $\mathbf{G}\varphi \equiv \neg\mathbf{F}\neg\varphi$ for "henceforth" as is usual in temporal logic (e.g., [23]). A formula $\varphi$ is an *LTL formula* if it is free of strategy quantifiers.

Formulas are interpreted over the concurrent game structure $\mathcal{G}$ as usual for ATL [2] with imperfect information. A formula $\langle\langle \vec{p} \rangle\rangle \varphi$ holds if set of agents $\vec{p}$ have a local strategy to bring about $\varphi$, that is there exists a function mapping $\vec{p}$'s view of the history thus far to moves for each agent in $\vec{p}$ that ensures $\varphi$ holds regardless of the actions of actions of other agents.

**Actions.** Formally, the syntax of the logic contains only send actions. Other actions can be faithfully represented as sending messages, as in object-oriented programming languages such as Smalltalk and Java. The below formula expresses that a customer's telephone number should not be used for telemarketing:

$$\begin{aligned}\mathbf{G}\forall p, q, d.\, & \mathsf{inrole}(p, \mathit{customer}) \\ & \wedge \mathsf{contains}(d, p, \mathit{telephone\text{-}number}) \\ & \rightarrow \neg\, \mathsf{send}(q, \mathit{telemarketing}, d)\end{aligned}$$

Syntactic sugar could be added to more naturally express general actions without affecting the technical results of this paper provided all the actions are recorded in the audit log.

**Parameterized Roles.** Some policies require finer distinctions between roles. For example, in order to Alice to perform an action, the policy might require that not only

is Alice a doctor, but in fact that she is Bob's doctor. The logic can be extended to express these parameterized roles by replacing the two-ary inrole predicate with a three-ary inrole predicate, as in $\mathsf{inrole}(\textit{Alice}, \textit{Bob}, \textit{doctor})$. Semantically, roles become pairs containing a role identifier and an agent. Again, the main results of the paper are unaffected.

## 4 Workflows and Responsibility

A workflow is a division of responsibility among human agents and a mechanical workflow engine. By assigning responsibilities to human agents, workflows can achieve privacy and utility goals beyond the capabilities of mechanical systems. We develop general design principles and desiderata using abstract workflows, which are collections of formulas specifying agent and engine responsibilities, and specific auditing algorithms for practical workflows like MyHealth based on communication graphs.

**Abstract Workflows.** Abstractly, a *workflow* is an LTL sentence $\varphi$ together with an LTL formula $\varphi_r(x)$ for each role $r$. The mechanical workflow engine is responsible for achieving $\varphi$ and a human agent $p$ in role $r$ is responsible for achieving $\varphi_r(p)$. As the workflow engine can only prevent communication initiated by human agents, it can enforce only safety properties (properties that fail at a finite time). Also, as a mechanical agent, the workflow engine must authorize communication based on message tags (and not the inaccessible message contents).

**Def.** A workflow is *feasible* if $\varphi$ is a safety formula[1] without the contains predicate and $\mathcal{G} \models \forall p.\varphi \wedge \mathsf{inrole}(p, r) \rightarrow \langle\langle p \rangle\rangle \varphi_r(p)$, for every role $r$.

In a feasible workflow, agents have local strategies for living up to their responsibilities and, thus, know which actions or inaction are responsible based on their observations of previous actions (assuming the workflow engine is functioning property). Moreover, if an agent is responsible for sending a message, feasibility ensures the agent will be able to send the message.

**Graph-based Workflows.** The MyHealth workflow depicted in Fig. 4 is a practical kind of workflow based on a *workflow graph*, a set of nodes $\mathcal{R}$, the roles of the workflow, and a function $T$ such that $T(r_1, r_2) \subseteq \mathcal{T}$ for all $r_1, r_2 \in \mathcal{R}$, where $\mathcal{T}$ is the set of attributes. The workflow engine permits a message $m$ to be sent from an agent in role $r_1$ to an agent in role $r_2$ if, and only if, $\mathrm{tags}(m) \subseteq T(r_1, r_2)$.

[1]There is a standard syntactic definition of safety formulas [23].

The responsibility of the workflow engine is the conjunction over $r_1, r_2 \in \mathcal{R}$ of

$$\mathbf{G}\forall p_1, p_2, q, m.\, \mathsf{inrole}(p_1, r_1) \wedge \mathsf{inrole}(p_2, r_2) \\ \wedge \mathsf{send}(p_1, p_2, m) \wedge \mathsf{tagged}(m, q, t) \rightarrow t \in T(r_1, r_2).$$

The workflow engine is stateless and distributed because the decision about whether to block a communication is based only on the current action, not on past or non-local actions. The responsibilities of human agents are divided into two parts: *tagging responsibilities* and *progress responsibilities*. Agents are responsible for tagging messages contents they are permitted to send. The tagging responsibility for role $r$ is the conjunction over $t \in T(r, *)$ of

$$\mathbf{G}\forall p_2, q, m.\, \mathsf{send}(x, p_2, m) \wedge \mathsf{contains}(m, q, t) \\ \rightarrow \mathsf{tagged}(m, q, t).$$

In MyHealth, nurses are responsible for tagging messages with *health-question*, *appt-request*, and *health-answer*. Tagging constraints are feasible because agents are free to select the tags on messages they send. Progress responsibilities require agents to eventually send messages. They are essential in achieving the liveness requirements of privacy goals (such as notification requirements) and in achieving utility goals. A progress responsibility is a formula of the form $\mathbf{G}\forall \vec{x}.\psi \rightarrow \mathbf{F}\theta$, where $\psi$ and $\theta$ are past formulas. For example, in MyHealth, doctors have the following progress responsibility.

$$\mathbf{G}\forall p, q, m.\, \mathsf{send}(p, x, m) \\ \wedge \mathsf{contains}(m, q, \textit{health-question}) \rightarrow \\ \mathbf{F}\exists m'.\, \mathsf{contains}(m, q, \textit{health-answer}) \wedge \mathsf{send}(x, q, m')$$

We do not make explicit how $m'$ depends on $m$ because $m'$ is created by a human agent, not by a piece of code. A more detailed model might bind the health answer to the question by way of a transaction identifier or a more explicit notion of causality.

## 5 Privacy and Utility in Workflow Design

Organizations design workflows to accomplish certain tasks while complying with legislation and privacy policies. This section contains algorithms for determining if a workflow design achieves these utility and privacy goals provided all agents act responsibly. If several workflows achieve privacy and utility, many privacy advocates [24] recommend deploying a workflow that minimizes the information disclosed to agents. We formulate a design criteria called *minimality* that makes this intuition precise.

### 5.1 Privacy

Many of the privacy provisions found in US legislation can be expressed in CI [9]. For example, one of MyHealth's privacy goals, derived from HIPAA, is that no patient should learn answers to another patient's health questions.

$$\mathbf{G}\forall p_1, p_2, q, m.\, \mathsf{send}(p_1, p_2, m) \wedge \mathsf{inrole}(p_2, \mathit{patient}) \\ \wedge \mathsf{contains}(m, q, \mathit{health\text{-}answer}) \rightarrow q = p_2$$

Responsible executions of a workflow $(\varphi, \varphi_{\mathcal{R}})$ is characterized in LTL as follows.

$$\mathit{agents\text{-}responsible} \equiv \varphi \wedge \bigwedge_{r \in \mathcal{R}} \forall p.\, \mathsf{inrole}(p, r) \rightarrow \varphi_r(p)$$

If $\mathcal{G} \models_{\text{LTL}} \mathit{agents\text{-}responsible} \rightarrow \mathit{privacy\text{-}policy}$, then responsible agents will achieve a privacy goal *privacy-policy*. However, this requirement imposes a design constraint only on agent responsibilities, not the responsibility of the workflow engine. When the tags are correct, the workflow engine is capable of enforcing safety properties but is incapable of enforcing liveness properties. The executions of the workflow with correct tags is expressed by the LTL formula

$$\mathit{tags\text{-}correct} \equiv \varphi \wedge \forall p_1, p_2, q, m, t.\, \mathsf{send}(p_1, p_2, m) \\ \rightarrow (\mathsf{tagged}(m, q, t) \leftrightarrow \mathsf{contains}(m, q, t)).$$

A workflow achieves a privacy goal if responsible agents fulfill the privacy goal and the workflow engine fulfills the safety component of the privacy policy when operating with correct tags.

**Def.** A workflow *achieves* a privacy goal *privacy-policy* if

$\mathcal{G} \models_{\text{LTL}} \mathit{tags\text{-}correct}\, \mathbf{U}\, \mathit{agents\text{-}responsible} \rightarrow \mathit{privacy\text{-}policy}$.

Algorithmically, we consider the propositional case in which there are a finite number of agents, messages, attributes, and roles.

**Theorem 1.** *It can be decided whether a workflow achieves a privacy goal using space polynomial in the description of the workflow and the number of agents.*

In practice, if an organizations has a large number of agents, we suggest the standard practice of evaluating whether a workflow achieves privacy on a smaller model for which the model-checking problem is tractable.

### 5.2 Utility

A workflow is useful if some execution accomplishes a task, though that task need not be accomplished in every execution. A workflow achieves a utility goal if some of the agents have a strategy for accomplishing the task. Formally, a utility goal for agents in a vector of roles $\vec{r}$ is a sentence of the form

$$\forall \vec{p}.\, \mathsf{inrole}(\vec{p}, \vec{r}) \rightarrow \langle\langle \vec{p} \rangle\rangle \psi,$$

where $\psi$, the task, is an LTL formula. For example, one utility requirement in MyHealth is that patients can receive answers to their health questions.

$$\forall p.\, \mathsf{inrole}(p, \mathit{patient}) \rightarrow \langle\langle p \rangle\rangle \mathbf{F} \exists p_1, m. \\ \mathsf{send}(p_1, p, m) \wedge \mathsf{contains}(m, p, \mathit{health\text{-}answer}).$$

A workflow achieves a utility goal *utility-goal* if the agents have a strategy for responsibly accomplishing their task if the other agents act responsibly, formalized as the ATL* entailment $\mathcal{G} \models \mathit{agents\text{-}responsible} \rightarrow \mathit{utility\text{-}goal}$.

In general, deciding whether a workflow satisfies this condition is undecidable [2] because $\mathcal{G}$ is a game of imperfect information: an agent's strategy is based only on the messages that the agent has sent or received, not on messages exchanged between other agents. For example, a patient's strategy for obtaining an answer to his or her health question cannot depend on whether messages have been exchanged between a doctor and a nurse. To avoid undecidability, we use a sound approximation based on local communication games. This approximation has proven adequate for the examples we have examined thus far.

**Local Communication Game.** Instead of checking the formula in the full game $\mathcal{G}$, we check the formula in the *local communication game* for agent $p$, $\mathcal{G}_p$, a game of perfect information where we can apply the standard model-checking algorithm for ATL*. Checking the formula in the local model is sound (but not complete) for a certain syntactic class of formulas that includes utility goals. The local communication game $\mathcal{G}_p$ is defined from the full communication game $\mathcal{G}$ by way of $\sim_p$, the smallest equivalence relation such that $\kappa_1 \sim_p \kappa_2$ if $\kappa_1 \xrightarrow{a} \kappa_2$, where $\kappa_1$ and $\kappa_2$ are knowledge states of $\mathcal{G}$ and $a$ is invisible to $p$.

**Proposition 2.** *$\hat{\kappa} \sim_p \kappa$ implies $\hat{\kappa}_p = \kappa_p$, for all knowledge states $\hat{\kappa}$ and $\kappa$.*

The states of $\mathcal{G}_p$ are the equivalence classes $[\kappa]_p$ of knowledge states $\kappa$ of $\mathcal{G}$ under $\sim_p$. A action is available in $[\kappa]_p$ if $p$ considers the action possible. For all sets of actions $A$, $[\kappa]_p \xrightarrow{A} [\kappa']_p$ if there exist $\hat{\kappa} \sim_p \kappa$ and $\hat{\kappa}' \sim_p \kappa'$ with $\hat{\kappa} \xrightarrow{A} \hat{\kappa}'$.

**Lemma 3.** *For all sets of actions $A$ and all knowledge states $\kappa$, $\kappa_1$, and $\kappa_2$,*

$$[\kappa]_p \xrightarrow{A} [\kappa_1]_p \textit{ and } [\kappa]_p \xrightarrow{A} [\kappa_2]_p \textit{ implies } \kappa_1 \sim_p \kappa_2.$$

*Proof.* It suffices to prove the statement for $A$ containing a single action. If $p$ is not the recipient of a send message, then $[\kappa_1]_p = [\kappa_2]_p = [\kappa]_p$. If $p$ is a recipient, then $\kappa_1(p)$ and $\kappa_2(p)$ and the other agents can invisibly exchange messages to equalize their knowledge as well, yielding $\kappa_1 \sim_p \kappa_2$. □

The main idea of the proof is contained in Lemma 4, which connects the truth value of visible formulas in $\mathcal{G}_p$ with their truth value in $\mathcal{G}$. The atomic formulas *visible* to agent $p$ are $\mathsf{send}(p,*,*)$, $\mathsf{send}(*,p,*)$, $\mathsf{inrole}(*,*)$, $\mathsf{contains}(*,*,*)$, and $\mathsf{tagged}(*,*,*)$, where $*$ is an arbitrary term. A formula is *visible* to $p$ if, and only if, all its atomic formulas are visible to $p$. The truth values of formulas visible to $p$ are determined by the view of $p$.

**Lemma 4** (Soundness)**.** *For all LTL formulas $\varphi$ that are visible to p,*

$$\mathcal{G}_p \models \langle\langle p \rangle\rangle \varphi \textit{ implies } \mathcal{G} \models \langle\langle p \rangle\rangle \varphi.$$

*Proof Sketch.* If $p$ has a strategy to force $\varphi$ in $\mathcal{G}_p$, then $p$ must also have a strategy to force $\varphi$ in $\mathcal{G}$ because the set of moves available to $p$'s opponents in $\mathcal{G}$ is a subset of the set of moves available to $p$'s opponents in $\mathcal{G}_p$. Fix a strategy $\Gamma_p$ for $p$ that forces $\varphi$ in $\mathcal{G}_p$. For every finite $p$-view $\pi \restriction p$ of $\mathcal{G}$, let $\Gamma(\pi \restriction p) = \Gamma_p([\pi]_p)$. $\Gamma$ is a local strategy for $p$ in $\mathcal{G}$ that forces $\varphi$ because $\varphi$ is visible to $p$. □

We can now state a sound algorithm for determine whether $\mathcal{G} \models \psi \rightarrow \langle\langle p \rangle\rangle \varphi$ for propositional LTL formulas $\psi$ and $\varphi$.

**Algorithm.** Let $G$ be the labeled transition system of $\mathcal{G}$ and let $A^\psi$ be the Büchi automata for $\psi$ (see, for example, [23]). Construct $G^\psi$ as the conjunction of the automata $G$ and $A^\psi$ and use it as the frame of the concurrent game structure $\mathcal{G}^\psi$. Let $\mathcal{G}^\psi_p$ be the local communication game. Report *true* if $\mathcal{G}^\psi_p \models \langle\langle p \rangle\rangle \varphi$ (determined using standard ATL* model checking, i.e. [2]).

**Theorem 5.** *Algorithm 5.2 is sound for deciding whether*

$$\mathcal{G} \models \textit{agents-responsible} \rightarrow \textit{utility-goal}.$$

*Proof Idea.* Intuitively, algorithm 5.2 uses deduction to translate the problem into $\mathcal{G}, \textit{agents-responsible} \models \textit{utility-goal}$ and then applies Lemma 4 to use the standard ATL* model checking algorithm. Each step in the algorithm preserves soundness, so the entire algorithm is sound. □

### 5.3 Minimal Workflow

The Markle Foundation [24], among many others, advocates the principle of *minimum necessary disclosure* for systems processing personal information. Under this principle, each agent should receive only the information needed for the workflow to achieve its utility goals. We begin to make this notion precise by inducing a partial order relation on workflows.

**Def.** One workflow $W_1 = (\varphi_1, \varphi_\mathcal{R})$ is *at least as restrictive as* another workflow $W_2 = (\varphi_2, \varphi_\mathcal{R})$, written $W_1 \leq W_2$, if $\mathcal{G} \models \varphi_1 \rightarrow \varphi_2$.

If $W_1 \leq W_2$, the workflow engine permits agents to learn no more information in $W_1$ than in $W_2$. For graph-based workflows, $W_1(\mathcal{R}, T_1)$ is at least as restrictive as $W_2(\mathcal{R}, T_2)$ if $T_1(r_1, r_2) \subseteq T_2(r_1, r_2)$ for all $r_1, r_2 \in \mathcal{R}$. In this initial formulation, workflows are only comparable under $\leq$ if they have the same agent responsibilities.

**Proposition 6** (Monotonicity)**.** *For all workflows $W_1$ and $W_2$, if $W_1 \leq W_2$ and $W_2$ achieves a privacy goal, then $W_1$ also achieves the privacy goal.*

The ordering $\leq$ is conservative in the sense that if two workflows are related by $\leq$, then the smaller one discloses less information, but, if two workflows are incomparable under $\leq$, one might still disclose less information. There does not appear to be a direct connection between this ordering and utility goals as whether an agent has a strategy to achieve a goal might be helped or hindered by strengthening the engine responsibility.

**Def.** A workflow $W$ is *minimal* for a utility goal if $W$ achieves the utility goal and all feasible workflows $W' < W$ fail to achieve the utility goal (where $W' < W$ if $W' \leq W$ and $W \not\leq W'$).

Minimal workflows provide the strongest privacy for a given utility goal, as advocated by the principle of minimum necessary disclosure. For abstract worflows, minimal workflows (as defined) fail to exist because the engine responsibility $\varphi$ of a candidate minimal workflow can always be strengthened by conjoining extraneous conditions. This definition is useful as it provides a precise metric for evaluating workflow designs, but other definitions are likely to have more desirable properties.

**Proposition 7.** *Given a set of roles $\mathcal{R}$, a responsibility for each role $\varphi_\mathcal{R}$, and a set of utility goals, there exists a graph-based workflow $(\varphi, \varphi_\mathcal{R})$ that is minimal among graph-based workflows.*

A minimal graph-based workflow can be computed using brute force by iteratively increasing the tags permitted on each edge of the workflow graph and testing whether the workflow achieves utility.

## 6 Auditing Workflow Execution

In evaluating workflow designs, we considered only responsible executions of the workflow. In an actual deployment, agents can act irresponsibly, either out of malice or

by mistake, leading to policy violations. To hold agents accountable for these actions, organizations should record communication in an *audit log*. In this section, we present auditing algorithms for finding agents accountable for policy violations and for periodically scanning the log for signs of irresponsible actions. These algorithms are not fully automatic, but require an oracle that reports the actual contents of messages. We seek to minimize the number of oracle calls as we expect the oracle to be implemented by a human auditor. Additionally, we recommend the audit log maintain the Lamport causality [20] relation between events to facilitate efficient auditing.

### 6.1 Policy Violations and Accountability

A safety property is violated on a finite trace, but the agent who performed the last action in that trace might not be blameworthy. For example, HIPAA does not permit the publication of protected health information in a newspaper, but HIPAA does not hold the reporter accountable for publishing the information. Instead, the person in the hospital who leaked the information caused the violation by acting irresponsibly and should be held accountable. Below, we make this intuition precise by defining *policy violation*, *causality* and *accountability*.

**Policy Violations.** To define when an action violates a policy, we employ the notion of strong compliance [9]: an action is strongly compliant with a policy if there exists a continued execution that satisfies the policy. Formally, given a finite past history $\sigma$, an action $a$ *strongly complies* with a policy $\theta$, written $a \in \mathrm{compliant}_\theta(\sigma)$, if there exists a trace $\sigma'$ such that $\sigma \cdot a \cdot \sigma' \models \theta$. We require of policies that agents can determine whether their actions strongly comply with the policy. This ensures that policy violations are visible to the agents violating the policy and prevents policy compliance from depending on unrelated actions.

**Def.** A privacy policy $\theta$ has *local compliance* if, for all agents $p$ and all traces $\pi_1$ and $\pi_2$,

$$\pi_1 \upharpoonright p = \pi_2 \upharpoonright p \text{ implies } \mathrm{compliant}_\theta(\pi_1) \cap A_p = \mathrm{compliant}_\theta(\pi_2) \cap A_p,$$

where $A_p$ is the set of actions for which $p$ is the sender.

**Causality.** Workflows do not define any explicit causal relation between messages. For this reason, we resort to a standard trace-based notion of causality [20].

**Def.** The *possibly-caused* relation for a trace $\pi$, written $\rightsquigarrow_\pi$, is the minimal transitive relation such that $i \rightsquigarrow_\pi j$ if event $i$ occurs before event $j$ in the view $\pi \upharpoonright p$ of some agent $p$.

In a trace $\pi$, the set of *causes* of an event $j$ is the set of events $\mathrm{causes}_\pi(j) = \{i \mid i \rightsquigarrow_\pi j\}$. The set of causes of an event is an over-approximation in the sense that if an event $i$ actually caused event $j$ (under some non-trace-based notion of actual causation, such as [22]), then $i \in \mathrm{causes}_\pi(j)$, but it is possible that $i \in \mathrm{causes}_\pi(j)$ without $i$ being an actual cause for $j$.

**Accountability.** An agent is *accountable* for policy violation $i$ in a trace $\pi$ if the agent undertook an action in $\mathrm{causes}_\pi(i)$ and did not fulfill his or her responsibilities in $\pi$. This definition is also an over approximation because every agent whose irresponsibility actually caused a policy violation is classified as accountable, but not every accountable agent actually caused a policy violation.

**Lemma 8** (Accountability)**.** *For all policies with local compliance, all graph-based workflows achieving the privacy policy, and all traces $\pi$, if $\pi$ contains an action that violates the policy, then there exists an accountable agent.*

*Proof.* Given a trace $\pi$ with an action $i$ undertaken by agent $p$ that violates the privacy policy, we construct a trace $\hat{\pi}$ that contains only the causes of $i$. The events $\hat{\pi}$ form a trace because the actions available to each agent at a given time depend only on the set messages previously received by that agent, and all those events are included in $\hat{\pi}$. Moreover, $\pi \upharpoonright p = \hat{\pi} \upharpoonright p$ because all the events in $p$'s view possibly caused the event $i$. Because the privacy policy has local compliance, $i$ is also a violating event in $\hat{\pi}$, and there must exist an irresponsible agent $q$ who undertook an action that possibly caused the violation. Finally, $q$'s actions are also irresponsible in $\pi$ and $q$ is an accountable agent. □

### 6.2 Finding Accountable Agents

Our auditing algorithm for finding accountable agents in an audit log uses an oracle $\mathcal{O}$ such that $\mathcal{O}(m) = \mathrm{contents}(m)$ to determine whether an agent acted responsibly by comparing the actual contents of messages with their tags. In practice, this oracle can be implemented by a human auditor, possibly with the assistance of a text classification algorithm.[2] The algorithm is formulated as a search on the causality graph of the audit log. The *causality graph* of a trace $\pi$ is the graph with a node for each event in $\pi$ and with an edge from event $i$ to event $j$ iff (1) $i \rightsquigarrow_\pi j$ and (2) there is no event $k$ with $i \rightsquigarrow_\pi k$ and $k \rightsquigarrow_\pi j$. The causality graph can be constructed mechanically as it does not depend on the contents of the messages. Moreover, it can be constructed incrementally as actions are logged.

[2]The University of Medicine & Dentistry of New Jersey determines whether to encrypt outgoing messages by scanning the messages for keywords [35], essentially guessing the correct tags mechanically.

**Algorithm.** Given a policy violation $i$ and an audit log $\pi$, let $G$ be the causality graph of $\pi$ with the edges reversed. Beginning at $i$, perform a breath-first search of $G$. Upon encountering an action $\mathsf{Send}(p, q, m)$, compare $\mathcal{O}(m)$ with $\mathrm{tags}(m)$. If $p$ failed to add a tag for which he or she was responsible, output $p$ as an accountable agent and terminate.

If the human auditor decides that the agent found by the algorithm did not actually cause the policy violation, he or she can continue the search. The algorithm will eventually enumerate all the accountable agents, one of whom must have actually caused the policy violation.

**Theorem 9** (Correctness)**.** *For every violation of a policy with local compliance in an audit log of a graph-based workflow achieving the policy, the algorithm outputs an accountable agent.*

*Proof.* Lemma 8 guarantees the existence of an irresponsible agent who possibly caused the policy violation. In graph-based workflows, irresponsible agents can be recognized by comparing the actual contents of messages with the tags of the messages. The algorithm searches the events that possibly caused the violation for such irresponsible tagging of messages. □

The algorithm reduces the number of oracle calls by restricting the search to actions that possibly caused the policy violation. In a deployment with thousands of agents exchanging messages, the portion of the log examined is likely to be several orders of magnitude smaller than the entire log, especially if an accountable agent is found at a shallow depth in the causality graph. The algorithm can use fewer oracle calls (while maintaining correctness) if the human auditor guides the search towards events that appear to be more relevant to the policy violation. The search can also be directed towards suspicious actions, a mechanical heuristic developed in the following section.

### 6.3 Monitoring for Irresponsible Actions

In addition to finding accountable agents after a policy violation occurs, auditing can also prevent some policy violations by detecting irresponsible actions before they lead to violations. The workflow engine can prevent irresponsible actions that can be detected mechanically, but it cannot prevent all irresponsible actions. In this section, we consider the problem of detecting irresponsible actions with the help of the oracle, but while minimizing work done by the human auditor. The simplest approach to detecting irresponsible actions in graph-based workflows is to sample communication at random and check, using the oracle, whether the sending agent tagged the message responsibly. This simple auditing algorithm requires many oracle calls to find a few irresponsible actions if the vast majority of communications are tagged responsibly. We suggest a more sophisticated approach based on a heuristic for *suspicious* actions.

**Suspicious Events.** Events appear suspicious if they indicate that some message tags were incorrect. The auditing engine can track the knowledge state of each agent using as in Sect. 3, using the tags as proxy for message contents:

$$\kappa \xrightarrow{\mathsf{Send}(p,q,m)} \kappa[q \mapsto \mathrm{cl}_{\preceq}(\kappa_q \cup \mathrm{tags}(m))]$$

The set of available action can also be modified to use tags:

$$\mathrm{available}_p^{\mathrm{tags}}(\kappa) = \{\tau\} \cup \{\mathsf{Send}(p, q, m) \mid \mathrm{tags}(m) \subseteq \kappa_p\}$$

These definitions are feasible for mechanical agents because they rely on message tags instead of message contents. If messages were always tagged correctly, the knowledge states computed from the tags would coincide with the real knowledge states. and every action undertaken by an agent would appear available from the tags. If some of the tags are incorrect, however, the apparent knowledge states are unlikely to coincide with the actual knowledge of agents. When agents undertake apparently unavailable actions, those actions are suspicious.

**Def.** An action $a$ by agent $p$ is *suspicious* if $a \notin \mathrm{available}_p^{\mathrm{tags}}(\kappa)$, where $\kappa$ is the current knowledge state computed using message tags.

"Suspicious" is a heuristic. Irresponsible actions can lead to policy violations without triggering suspicion, and suspicious actions can occur unrelated to any irresponsible action. Suspicious actions do indicate related incorrect tags.

**Proposition 10.** *For all audit logs $\pi$, if agent $p$ undertakes a suspicious action, then there exists a message in $\pi \restriction p$ with incorrect tags.*

*Proof.* The actions available to an agent are determined by his or her view of a trace. If an agent undertakes a suspicious action, then his actual knowledge differs from that computed from tags and thus a tag must be incorrect. □

**Algorithm.** Given a suspicious event $i$ by agent $p$ and an audit log $\pi$, consult the oracle $\mathcal{O}$ about the message sent during $i$ and the messages received by $p$ before $i$ in reverse chronological order until an incorrectly tagged message is found. If the message was tagged irresponsibly, output the sending agent.

## 7 Formalizing MyHealth@Vanderbilt

In this section, we illustrate the formal concepts developed in the preceding sections using the MyHealth@Vanderbilt patient portal. Based on this formal

analysis, we made several concrete design recommendations to the MyHealth developers. Specifically, we recommend introducing tags into the system and modifying the workflow to route health questions directly to nurses.

**Workflow.** MyHealth is a graph-based workflow with roles *patient*, *secretary*, *nurse*, and *doctor* and attributes *appt-request*, *appt-confirm*, *health-question*, and *health-answer*. There are two subsumption relations between attributes, enabling health answers to be computed from health questions and appointment confirmations to be computed from appointment requests.

$$\textit{health-answer} \preceq \textit{health-question}$$
$$\textit{appt-confirm} \preceq \textit{appt-request}$$

In the initial knowledge state, patients know their own *health-question* and *appt-request*, and no other attributes are known. The edges of the workflow graph are labeled as depicted in Fig. 4. For example, the edges emanating from the nurse to the other roles are labeled as follows:

$$T_{\textit{nurse,patient}} = \{\textit{health-answer}\}$$
$$T_{\textit{nurse,secretary}} = \{\textit{appt-request}\}$$
$$T_{\textit{nurse,doctor}} = \{\textit{health-question}\}$$

MyHealth has several progress responsibilities. Nurses are responsible for either answering or forwarding health questions, for sending misdirected appointment requests to the secretary, and for sending health answers only to the appropriate patient, as depicted in Fig. 5.

**Privacy.** The salient privacy goal of MyHealth is to comply with HIPAA. Many of the requirements of the HIPAA Privacy Rule can be expressed in the logic [9]. For simplicity, we consider two specific requirements: only health care providers receive protected health information and patients receive only their own health answers.

$$\mathbf{G}\forall p_1, p_2, q, m.\, \mathsf{send}(p_1, p_2, m) \wedge \mathsf{contains}(q, \textit{health-question}) \rightarrow (\mathsf{inrole}(p_2, \textit{nurse}) \vee \mathsf{inrole}(p_2, \textit{doctor}))$$

$$\mathbf{G}\forall p_1, p_2, q, m.\, \mathsf{send}(p_1, p_2, m) \wedge \mathsf{inrole}(p_2, \textit{patient}) \wedge \mathsf{contains}(m, q, \textit{health-answer}) \rightarrow q = p_2$$

The proposed MyHealth workflow achieves both these privacy goals, but the current workflow does not achieve the first. While these properties are easy to check for this simple workflow, we could have used the algorithm in Sect. 5.1.

**Utility.** Two utility goals of MyHealth are that patients can schedule appointments and receive answers to their health questions. We state these goals in terms of the existence of strategies.

$$\forall p.\, \mathsf{inrole}(p, \textit{patient}) \rightarrow \langle\langle p \rangle\rangle \mathbf{F} \exists q, m. \mathsf{send}(q, p, m) \wedge \mathsf{contains}(m, p, \textit{health-answer})$$

MyHealth does achieve these utility goals. Another utility goal is worth analyzing because some of the patients who use MyHealth might not be technically savvy. The nurse and doctor have a strategy for reacting to health questions sent by patients with health answers, even if the patient incorrectly tags his or her health question and it is directed to the secretary. MyHealth without patient responsibilities achieves this privacy goal, but a variant of the workflow that does not include edges between the secretary and the nurse does not achieve this utility goal.

**Auditing.** We demonstrate the auditing algorithms on a trace $\pi$ that violates the privacy policy because Alice, a secretary, received a health answer. Alice received the message because it was sent by Bob, another secretary, and tagged as an appointment request. Tracing backwards through the causality structure, the auditing algorithm queries the human auditor for the contents of several messages sent to Bob until it find an irresponsible message sent by Charlie, a nurse. Charlie sent the health answer to Bob irresponsibly tagged as an appointment request and can be held accountable for this action. Moreover, this irresponsible action is likely to be found by monitoring because it is suspicious: Bob had not previously received a message tagged as an appointment request.

## 8 Conclusions and Future Work

Privacy and regulatory compliance are important business and social concerns. Existing laws and societal expectations are complex and difficult for modern enterprises to understand and manage. We believe there is a need for a general, clear, and comprehensive framework for reducing high-level requirements to specific operating guidelines that can be applied at individual steps in a business process or organizational workflow.

We have developed a general framework for specifying and verifying privacy and utility goals of business processes that include both human agents and electronic systems. The agents interact in a concurrent game structure with imperfect information. The formulas of our logic are interpreted over this game structure. Expressing utility goals requires a strategy quantifier not needed in earlier work dealing only with privacy. Specifically, we use ATL* to express utility properties and LTL to express privacy properties. This

$$\mathbf{G}\forall p_1, q, m.\, \mathsf{send}(p_1, x, m) \wedge \mathsf{contains}(m, q, \textit{health-question}) \rightarrow \mathbf{F}(\exists d.\, \mathsf{inrole}(d, \textit{doctor}) \wedge \mathsf{send}(x, d, m)) \vee (\exists m'.\, \mathsf{contains}(m', q, \textit{health-answer}) \wedge \mathsf{send}(x, q, m'))$$

$$\mathbf{G}\forall p, q, m.\, \mathsf{send}(p, x, m) \wedge \mathsf{contains}(m, q, \textit{appt-request}) \rightarrow \mathbf{F}\exists s.\, \mathsf{inrole}(s, \textit{secretary}) \wedge \mathsf{send}(x, s, m)$$

$$\mathbf{G}\forall p, q, m.\, \mathsf{send}(x, p, m) \wedge \mathsf{contains}(m, q, \textit{health-answer}) \rightarrow p = q$$

**Figure 5. Responsibilities for Nurses in MyHealth@Vanderbilt**

framework also distinguishes between information visible to mechanical agents and to human agents. Specifically, we assume messages have explicit tags that can be read by mechanical agents, but these tags need not match the actual semantics contents of messages. This distinction is useful in evaluating which policy enforcement tasks can be carried out by electronic systems and which require human agents. Currently, we assume that privacy and utility goals depend only on the type of information (rather than the actual data values) and that information is about a single individual (rather than about groups of individuals). We plan to relax these restrictions in subsequent work and develop precise connections with research in data privacy and aggregation. While our current model focuses on communication actions, we also plan to develop extensions for tracking the purpose for which data is used within a business process.

We model business processes as workflows: divisions of responsibility among agents in various organizational roles. Workflows also include a mechanical engine that aids in enforcing policy. We present two sets of general technical results. The first set of results concerns workflow design. We present algorithms for checking whether a workflow design achieves the privacy and utility goals, assuming all agents are responsible. We also formulate a minimality condition on workflow designs that makes precise the informal principle of minimum necessary disclosure advocated by the privacy community. Currently, we consider only qualitative notions of privacy and utility. In subsequent work, we hope to explore more quantitative notions and appropriate extensions of the minimality condition. In particular, we believe that the ordering relation on workflows defined in this paper can be extended to align better with utility goals. In addition, the definition of workflows can be extended to capture the causal relation between two messages—when one message is sent in response to another. For example, in a messaging system, such as the one in MyHealth, messages can be bound together (using a thread or transaction identifier) so that it is possible to identify that a health answer from a doctor corresponds to a specific health question sent by a patient. Such causality relations can also be useful for improving the performance of the auditing algorithms.

The second set of results concerns auditing workflows carried out by a combination of responsible and irresponsible agents. Specifically, we present an algorithm for finding agents accountable for policy violations. This algorithm is aided by a human auditor accessed through an oracle, but aims to minimize the number of oracle calls. We also present a heuristic for identifying irresponsible agents by analyzing the audit log for suspicious events. We believe that these algorithms are practical and reduce significantly the workload on human auditors. However, we do not view these results as the final word on the auditing problem. We plan to investigate this problem further in subsequent work, specifically exploring fully automated auditing techniques with reasonable false positive and false negative rates and developing (semi-)automated techniques for identifying policy violations by analyzing audit logs.

We applied these methods to analyze the design of MyHealth, a patient portal deployed at Vanderbilt. Our analysis led to concrete suggestions for improving the privacy and utility of the workflow. We are currently in discussions with the MyHealth designers about incorporating these changes in the portal. We believe that the methods in this paper are applicable to systems in a wide range of sectors including health care and financial services, where business processes routinely handle personal information and privacy and utility concerns are significant. In subsequent work, we hope to carry out case studies of other health care systems using automated tools. Another application area we hope to investigate is the outsourcing of business processes that deal with sensitive information such as social security and credit card numbers. In this setting, minimal workflows are particularly useful for tasks such as credit card charge-back that require access to real credit card numbers.

While it is unreasonable to expect the manager of a hospital, call center, or credit card processing organization to become fluent in temporal logic, we believe that the most productive way to address the basic problem is to develop precise, unambiguous foundations and use them to develop more accessible principles and guidelines. The latter topic is another interesting direction for future work.

## References


[1] N. R. Adam, V. Atluri, and W.-K. Huang. Modeling and analysis of workflows using petri nets. *J. Intell. Inf. Syst.*, 10(2):131–158, 1998.

[2] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

[3] A. Anderson. Key differences between XACML and EPAL. Ottawa new challenges for access control, 2005.

[4] A. Anderson, A. Nadalin, B. Parducci, D. Engovatov, E. Coyne, F. Siebenlist, H. Lockhart, M. McIntosh, M. Kudo, P. Humenn, R. Jacobson, S. Proctor, S. Godik, S. Anderson, and T. Moses. Extensible access control markup language (XACML) version 2.0, 2004.

[5] A. I. Antón, J. B. Earp, and A. Reese. Analyzing website privacy requirements using a privacy goal taxonomy. In *Requirements Engineering 2002*, pages 23–31, 2002.

[6] A. I. Antón, Q. He, and D. L. Baumer. Inside JetBlue's privacy policy violations. *IEEE Security and Privacy*, 2(6):12–18, 2004.

[7] V. Atluri and W.-K. Huang. An authorization model for workflows. In *European Symposium on Research in Computer Security (ESORICS)*, volume 1146 of *LNCS*, pages 44–64. Springer–Verlag, 1996.

[8] M. Backes, B. Pfitzmann, and M. Schunter. A toolkit for managing enterprise privacy policies. In *European Symposium on Research in Computer Security (ESORICS)*, volume 2808 of *LNCS*, pages 101–119. Springer–Verlag, 2003.

[9] A. Barth, A. Datta, J. C. Mitchell, and H. Nissenbaum. Privacy and contextual integrity: Framework and applications. In *IEEE Symposium on Security and Privacy*, pages 184–198. IEEE Computer Society, 2006.

[10] A. Barth and J. C. Mitchell. Enterprise privacy promises and enforcement. In *Workshop on Issues in the Theory of Security*, pages 58–66. ACM Press, 2005.

[11] CNN. FBI seeks stolen personal data on 26 million vets. `www.cnn.com/2006/US/05/22/vets.data/`.

[12] L. F. Cranor. *Web Privacy with P3P*. O'Reilly and Associates, Inc., 2002.

[13] CSO Online. Safety in numbers. `http://www.csoonline.com/metrics/viewmetric.cfm?id=265`.

[14] M. Dumas and A. H. M. ter Hofstede. UML activity diagrams as a workflow specification language. In *UML 2001*, pages 76–90. Springer–Verlag, 2001.

[15] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO 2004: 24th Annual International Cryptology Conference*, volume 3152 of *LNCS*, pages 528–544. Springer–Verlag, 2004.

[16] EPIC. The Gramm–Leach–Bliley Act. `http://www.epic.org/privacy/glba/`.

[17] C. Evans-Pughe. The logic of privacy. *The Economist*, 382(8510):65–66, 2007.

[18] Federal Trade Commission. In brief: the financial privacy requirements of the Gramm–Leach–Bliley Act. `http://www.ftc.gov/bcp/conline/pubs/buspubs/glbshort.htm`, 2002.

[19] G. Karjoth and M. Schunter. A privacy policy model for enterprises. In *IEEE Computer Security Foundations Workshop*, page 271. IEEE Computer Society, 2002.

[20] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, 1978.

[21] K. LeFevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, Y. Xu, and D. DeWitt. Limiting disclosure in hippocratic databases. In *30th Int'l Conf. on Very Large Data Bases, , August 2004.*, Toronto, Canada, 2004.

[22] D. Lewis. Causation as influence. *The Journal of Philosophy*, 97(4):181–197, 2000.

[23] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer–Verlag, 1995.

[24] Markle Foundation. The Connecting for Health Common Framework. `http://www.connectingforhealth.org/commonframework/`, 2006.

[25] M. J. May, C. A. Gunter, and I. Lee. Privacy apis: Access control techniques to analyze and verify legal privacy policies. In *IEEE Workshop on Computer Security Foundations*, pages 85–97. IEEE Computer Society, 2006.

[26] H. Nissenbaum. Privacy as contextual integrity. *Washington Law Review*, 79(1):119–158, 2004.

[27] OASIS. OASIS Web Services Process Execution Language (WSBPEL). `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel`.

[28] Office for Civil Rights. Summary of the HIPAA privacy rule. US Department of Health & Human Services, 2003.

[29] Oracle. Oracle BPEL Process Manager. `http://www.oracle.com/technology/products/ias/bpel/`.

[30] J. Reagle and L. F. Cranor. The platform for privacy preferences. *Communications of the ACM*, 42(2):48–55, 1999.

[31] P. Samarati. Protecting respondent's privacy in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[32] M. Schunter, P. Ashley, S. Hada, G. Karjoth, C. Powers, and M. Schunter. Enterprise privacy authorization language (EPAL 1.1). `http://www.zurich.ibm.com/security/enterprise-privacy/epal/Specification/`, 2003.

[33] L. Sweeney. k-Anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[34] W. Teepe, R. P. van de Riet, and M. S. Olivier. Workflow analyzed for security and privacy in using databases. In *Working Conference on Database Security*, pages 271–282. Kluwer, B.V., 2001.

[35] University of Medicine & Dentistry of New Jersey. HIPAA Security Standards FAQ's. `http://www2.umdnj.edu/hipaaweb/security/security_emailFAQ02.htm`.

[36] Vanderbilt Medical Center. MyHealthAtVanderbilt. `https://www.myhealthatvanderbilt.com/`.