The Thing Itself Speaks

Accountability as a Foundation for Requirements in Sociotechnical Systems

Amit K. Chopra
Lancaster University
Lancaster LA1 4WA, UK
a.chopra1@lancaster.ac.uk

Munindar P. Singh
North Carolina State University
Raleigh, NC 27695-8206, USA
m.singh@ieee.org

***Abstract*—We consider sociotechnical systems (STSs) that facilitate social interaction among autonomous principals. Accountability is a foundational concept in such systems. Although established requirements engineering methods support traceability (it is possible to tell who did an action), they do not support accountability in the broad sense of calling to account of one party by another.**

We present a novel formulation of requirements in STSs that gives prominence to accountability. Specifically, an accountability requirement involves two principals, one accountable party to the other, with regard to some conditional expectation. We propose a metamodel for sociotechnical systems in which relational constructs such as commitments, authorization, and prohibition, are treated as accountability requirements. We apply our metamodel to a healthcare case study from the literature and show how it helps address the problems of ineffective interaction noted in that study.

I. Introduction

This paper concerns sociotechnical systems that involve social interaction among multiple autonomous principals (individuals or organizations). Collaborative systems for resource sharing, business organizations, and emerging visions such as of smart cities fall within this scope. The principals interacting within the sociotechnical system would nominally be its stakeholders and their interactions would be facilitated via information technology (IT).

Autonomy and accountability are fundamental concepts in understanding sociotechnical systems. Autonomy means each principal is free to act as it pleases; accountability means that a principal may be called upon to account for its actions. Mamdani and Pitt [1] use autonomy and accountability as bases for distinguishing principals from their computational agents. In general, balancing autonomy and accountability is crucial for ensuring that an STS would not devolve into the extremes of chaos or tyranny. The above echoes the well-known intuitions from studies in political theory [2] and healthcare [3], among other fields. Accountability doesn't conflict with autonomy in that a principal can violate any expectation for which it is accountable: it would merely be held to account. Nor does accountability entail sanctioning: in general, an accountable party may be assigned responsibility and be penalized or rewarded, as appropriate.

The thesis of this paper is that to model an STS is to precisely capture the accountability relationships between its principals. Anything less would lead to unsound solutions; anything more would lead to over-coupled solutions. Notice that we do not presume a black-box model for components. In many cases, internal implementation details may be subject to an accountability requirement and may need to be suitably exposed.

Accountability is classically understood, e.g., in political theory [2] and healthcare [3], in terms of the standing of one party—the *account-taker*—to expect certain behavior from another—the *account-giver*. That is, accountability is inherently a normative relationship. In seeking to formalize accountability, traditional computing approaches lose this core intuitive basis of accountability. Some computing approaches labeled "accountability", e.g., [4], [5], address traceability of actions: traceability is an important mechanism for holding someone accountable, but is neither necessary nor sufficient for accountability.

As a case in point, a patient may hold a hospital accountable for privacy loss, not the nurse who leaks the information. A customer may hold a network provider accountable for loss of connectivity even if the loss was caused by a third-party attacker or by equipment failure. Further, if a person, say Alice, circumvents traceability, e.g., by getting another person, say Bob, to act on her behalf, Alice still remains accountable for the requirements she violated. Alice's circumvention may on the one hand protect her but may on the other hand subject her to greater sanctions for having acted in bad faith.

A. What Makes a System Sociotechnical?

Broadly, we consider *persona iuris*, that is, legal persons, to be autonomous principals. Thus a person or an organization could be a principal. We use the label STS for any system that facilitates interaction between two or more autonomous principals. The term thus applies to large-scale systems of the sort that Sommerville et al. [6] describe. They conceptualize large-scale complex IT systems as coalitions of independent organizations. Each member of the coalition has its own information system but collaborates with others on the basis of agreed-upon protocols. In our terms, each coalition is an STS as is a coalition of coalitions. The member organizations are principals, as are the coalitions. Sommerville et al. give the example of a coalition of organizations, who interact via their software systems in the execution of transactions in equity markets. The mutual accountabilities of the organizations in the coalition would be key to the conception of such a coalition. For example, a member of an equity market would be

accountable to the governing stock exchange for all messages and orders that come from a trading system that operates under the member's trading codes. Further member firms are accountable to the exchange for testing their internal systems before connecting to the exchange's trading system but the testing must not be against the trading system as it may impact the market.

The meeting scheduler of RE lore is an STS as well. The meeting initiator and participant are principals; the employer they work for too is a principal, as is the (vendor and) operator of their calendaring service. The employer holds the initiator accountable for requesting meetings during business hours. The participant holds the initiator accountable for requesting useful meetings. The initiator holds the participant accountable for providing schedule information. The participant holds the initiator accountable for not publishing the schedule information without permission.

B. Benefits of Accountability Requirements

Conceptually, an accountability relationship represents a requirement on the account-giver for the thing it is accountable for. If the requirement is not fulfilled, the account-taker would have a legitimate reason to complain. Conceptualizing requirements for STSs in terms of accountability requirements brings several benefits.

Managing complexity. Accountability requirements serve as high-level representations of protocols. The benefit here is modularity: it is enough that principals know their accountability requirements. Each principal can implement its software system in view of its own accountability requirements independently of others. In practical settings, especially competitive ones, specifying anything more than the accountability requirements turns out to be impossible (that is the reason we have contracts and standards).

Software systems in equity markets are exceedingly complex. Many members run complex automated trading algorithms with the aim of maximizing their profits. Sommerville et al. conceptualize the entirety of the system as "interacting algos". Accountability requirements would help manage this complexity by giving a basis for firms for implementing their internal systems in relation to others but still leaving them free to use sophisticated and private trading algorithms to maximize their profit. Meeting scheduling is arguably less complex than trading in equity markets. However, commercially available meeting scheduling software is complex, supporting a variety of configuration options for all user roles (initiator, employer, participant, and so on). Software based on accountability requirements will be simpler: software implementations of particular user roles can be packaged separately and even when the software implements all user roles, accountability requirements would serve to clarify user interfaces.

Organization-IT alignment. The literature is replete with studies that highlight the difficulty of aligning IT systems with the organizational context in which they are deployed, e.g., [7], [8]. Accountability requirements capture the formal aspects of organizational context. For any principal, its accountability requirements would act as requirements on its behavior, and consequently, on its software, thus supporting alignment.

If a stock exchange member implements its information systems to conform with the stock exchange's requirements, then, to that extent, we can claim alignment of the member's systems with the organizational context. Analogously, for principals involved in meeting scheduling.

Informing sanctioning processes. In general, accountability is crucial in guiding sanctioning processes in sociotechnical systems, including for determining reward (positive sanction) and blame (negative sanction). Note that to hold a principal accountable is not the same as blaming the person, although we would normally not blame a principal for something that it is not accountable for.

Sommerville et al. give the example of anomalous behavior of US stock markets during the *2010 Flash Crash*. Subsequent investigation and analysis did not reveal any accountability requirement violations (no firm was found to have broken any rules) but instead pointed toward a complex pattern of trading as the potential cause. But had some irregularities been revealed, sanctioning processes would likely have kicked in. In the meeting schedule example, invitees who accept but do not attend and initiators who call meetings for frivolous reasons would likely face sanctions.

Transparency. In a variety of settings, including e-government and crowdsourcing, transparency arises as a key challenge, especially concerning how information is obtained and processed [9], [10]. Accountability requirements would ideally inform the transparency policies of organizations. Indeed, the thing one is accountable to another for implies transparency with regard to that thing. If that thing is unobservable to the account-taker, then accountability is meaningless.

If a member is accountable to the stock exchange for testing its information systems, then the details of the tests must be produced before the stock exchange when it so demands (for example, during an inquiry). In other words, there must be transparency with regard to the testing, otherwise to be accountable for testing would be meaningless.

C. Contributions

1) We introduce the notion of accountability requirements and propose that accountability requirements are essential to effectively capturing the relational (among principals) dimension of STSs. We compare accountability requirements with the influential abstraction of intentional dependencies, as conceptualized in i* [11] and Tropos [12]. Our proposal echoes the idea in healthcare ethics literature that *organizational purposes are implemented by systems of accountability* [13].
2) We introduce a metamodel for STSs based on accountability requirements. We apply the model to an extensive healthcare case study from the literature and show how it can help mitigate the problems mentioned by the authors of the case study.

D. Organization

The rest of the paper is organized as follows. Section II discusses the shortcoming of current methods in RE from the point of view of accountability. Section III describes a metamodel for accountability requirements. Section IV introduces a case study of interdepartmental coordination in a large hospital. We show how accountability requirements can help tackle the problems noted by the case study authors. Section V discusses related work in requirements engineering and charts out some directions for future work.

II. Introducing Accountability to RE

At the risk of over-simplification, we observe that the RE literature approaches the engineering of STSs from two main perspectives. First, the information systems dominant analyses consider the social and organizational aspects of deploying software solutions in real-life organizations. These approaches concern themselves with themes that affect the success or failure of an IT deployment, including organizational culture and the motivations and incentives of the participants. A shortcoming of these approaches is that, though they introduce relevant concepts, they provide no clear computational logical path to reasoning about them, in essence, relegating the concepts to be no more than informal guidance for designers.

Second, the modeling dominant approaches provide a formal notation in which to express elements of requirements. Prominent among these are the goal-oriented approaches such as KAOS [14] and Tropos [12]. These approaches concern themselves with rendering selected social and organizational notions in formal terms to guide the design process. However, the particular concepts chosen in these approaches have either a purely functional or at best a cognitive underpinning, in essence, disregarding social relationships such as accountability.

A. Dependencies in RE

i* [11] and Tropos models of sociotechnical systems [12] are especially perspicuous because of their use of various kinds of dependencies (goal, softgoal, resource, and so on) among the actors in the system. An accountability requirement is a kind of dependency among principals. Specifically, the account-taker depends on the account-giver for the satisfaction of the requirement. However, accountability requirements and Tropos dependencies are different kinds of dependencies. We list below the important distinctions between the two by referring to a Tropos model of a meeting scheduling system (Figure 1). Below we discuss goal dependencies but analogous considerations apply to the other kinds of Tropos dependencies.

For clarity, we first distinguish between the two ways Tropos dependencies are characterized in the literature.

One, formally, goal dependencies are characterized in terms of the Tropos actors' beliefs, goals, abilities, and so on—in other words, their internals [11]. This formal conception is often stated in plain English as the following: when x depends on y for a goal p it means that x is a goal of p and y is capable and willing to achieve p (in practical terms, making p a goal of y as well). From such a conception, one cannot tell if either x or y is accountable to anyone for anything. For example, as shown in Figure 1, MI depends upon MP for its goal that the latter attend the meeting. But just because MP and MI have the requisite internal properties (beliefs, goals, and so on), does it mean that the MP is accountable for attending? And if it is, to whom? The goal dependency does not contain that information. To see that the notion of goal dependency is orthogonal to that of an accountability requirement, imagine a meeting scheduling domain with the same *attends meeting* goal dependency but an accountability requirement where MI is accountable to IP that MP attends the meeting.

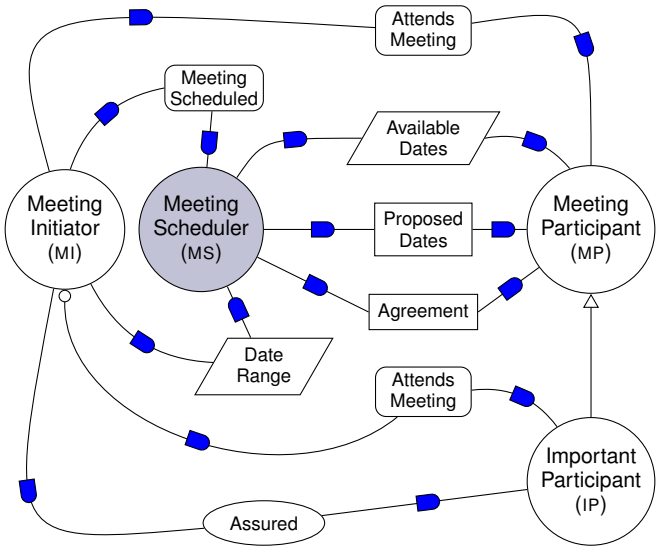

Fig. 1: The meeting scheduling *system to be* [15]. The machine is the **Meeting Scheduler** actor.

Because goal dependencies ascribe internal properties to actors, even the strategic dependency model of Tropos describes the internals of an actor (even if in a highly abstract way). Accountability requirements say nothing about the goals of the principals. If a principal is accountable for attending a meeting, that does not imply his or her goal of attending the meeting. Indeed, the principal may have the goal of accepting but not attend the meeting, thereby violating the requirement. Nonetheless, he or she remains accountable for the requirement.

Two, informally, a goal dependency is often characterized as the following: "an actor depends on another to make a condition in the world come true" [11]. (Notice that both the formal and informal characterizations are from the same paper. Therefore, for the authors, the formal characterization is really an elaboration of this informal.) It is possible to adopt the informal characterization of a goal dependency but without adopting its formal characterization. If we did that, then a goal dependency would be a primitive concept that simply states that one actor depends on another for something. Then

we could treat it as the statement of a relational requirement (between two actors) and one that does not ascribe any internal properties to any actor. The rationale behind doing so would be to capture a very general notion of dependency.

However, even this informal, more general notion of goal dependency does not contain information about accountability. In particular, goal dependency does not imply accountability. MI may depend on MP for attending, but unless, e.g., MP committed to MI for showing up for the meeting, MP will not be accountable for not showing up. In practice, it is the accountability requirement (in this example, a commitment) that makes it reasonable to depend on others. To give another example, MI could depend on an IT operator (OP) to have software-related issues fixed, but that dependency would be unreasonable if OP hasn't committed to MI for doing so.

A final point of distinction is that even software can be an actor in a goal dependency (regardless of how it is characterized). For example, in Figure 1, Meeting Scheduler (MS) is software that the principals use to carry out meeting scheduling, and MI depends on the MS for the being meeting scheduled. However, it makes little sense to say that the MS—a piece of software—is accountable to the MI for the meeting being scheduled. In contrast with goal dependencies, only autonomous principals may appear as an a-giver or a-taker in accountability requirements.

In summary, the notion of goal dependency has proved to be a useful abstraction for conceptualizing requirements as relations among actors. There are two main shortcomings that need to be addressed. One, the formal characterization of goal dependency ascribes internal state to the actors. And both the formal and informal characterizations say nothing about accountability, which is what really makes dependence reasonable. The notion of accountability requirements addresses these shortcomings.

B. Accountability Requirements for Meeting Scheduling

Meeting scheduling involves social interaction among the participating principals. And even if not represented, there would also be accountability requirements among the interacting principals. For example, one can imagine an organizational setting with the following accountability requirements.

- $\mathcal{A}_1$. The meeting initiator is accountable to the organization for the purpose of the meeting and the list of invitees.
- $\mathcal{A}_2$. The meeting initiator is accountable to the organization for not scheduling meetings before 10AM and after 3PM.
- $\mathcal{A}_3$. If the participant (invitee) has responded in the affirmative, then unless notified otherwise by the initiator, the invitee is accountable to the initiator for attending.
- $\mathcal{A}_4$. The meeting initiator is accountable to the organization for ensuring that after the meeting, all electrical appliances, including lights, are powered off.
- $\mathcal{A}_5$. Each participant is accountable to the organization for clearing the room of any material that he or she brought in.

And so on. Actions such as "responding," "notifying," "inviting," and so on would manifest themselves (e.g., as buttons) in (software) user interface. However, the accountability requirements would fall outside the software in the broader sociotechnical system. In fortunate cases, these would be laid out in plain text in documents such as contracts. In many cases, however, they would be unspecified in any form except perhaps verbally. Ideally, accountability requirements should have a formal representation so that it is possible to reason about them at design time, just as it is with goal models [16]. Additionally, they should be explicitly represented in the sociotechnical system at runtime so that principals can inform their decisions accordingly. Doing so would be analogous to using goal-based representations to support software adaptation [17], [18], [19].

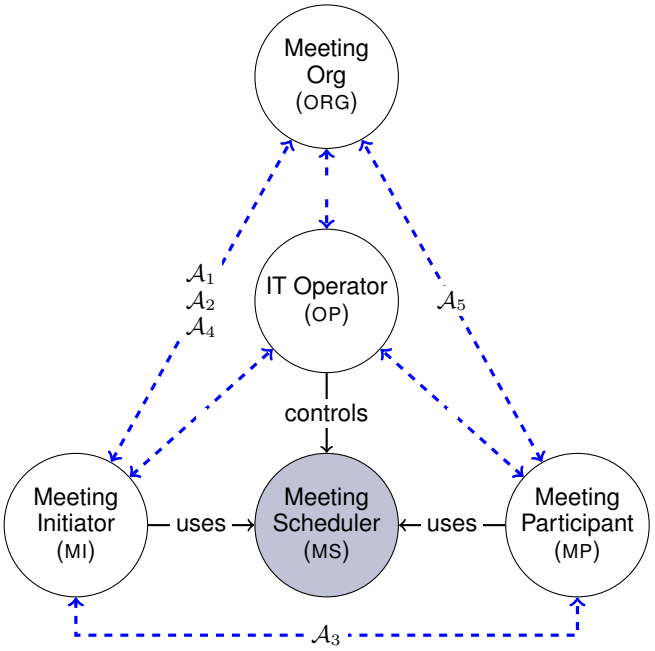

Fig. 2: Accountability view of the meeting scheduler. The dashed lines indicate the directed accountability relationships, some of which are listed in the text.

C. Regulations and Compliance

A well-recognized RE challenge for STSs is ensuring the legal compliance of information systems. Some approaches, e.g., [20], [21] derive requirements from documents such as contracts and government-imposed regulations. Noting the lack of support in i* [15] for modeling normative concepts, Siena et al. [22] augment i* with Hohfeldian legal concepts [23]. They consider the applicable normative propositions as distinct from requirements, which they model as goals. Their system models are therefore unclear about accountability: notably, they freely use i* dependencies, which relate actors but say nothing about accountability, with normative propositions. Ghanavati et al. [24] and Rifaut and Dubois [25] derive goal models from regulations but do not formally model the accountability requirements.

Breaux et al. [26] use undirected obligations to model

healthcare regulations. For them, a requirement is accountable if a mechanism exists to verify that the requirement has been satisfied. Breaux et al. give the example of the requirement that user passwords be at least eight characters in length. For them, this requirement would be accountable if users were supported by a software program that determines password length. Notice that whereas they talk about the accountability of requirements, we talk about accountability requirements, and through them, the accountability of principals. In our approach, the IT Operator (basing it on the meeting scheduler example from Figure 2) could be accountable for ensuring the password meets some criteria. Or, the user could be made accountable for choosing a strong password regardless of software support, though this approach may not be realistic. The main point being that there must always be a principal to be held to account.

III. A Framework for Accountability Requirements

To accommodate the autonomy of principals, an accountability requirement is directed—from an *account-giver* or *a-giver* principal to an *account-taker* or *a-taker* principal. The a-giver is accountable to the a-taker. Appropriating some legal language, we might state that that a-giver is modeled as one who is *sui iuris*, i.e., an autonomous party, whereas an a-taker is modeled as one who has *locus standi* or standing (to complain).

Our formulation of accountability is purely normative: accountability requirements describe how principals ought to act in each other's eyes, providing a basis for their mutual expectations. The foregoing view captures the key intuition of scholars outside of IT [13], [2]. As a normative conception, accountability is independent from both support mechanisms (such as the traceability of actions to principals) and sanctioning processes. Importantly, trustworthy behaviors sometimes involve violating an accountability requirement when that's what's called for. For example, consider that Bob is accountable to Alice for not sharing her data. Alice may trust Bob more than otherwise if he violates this requirement for saving her life, but not if it was just to gossip. And, Bob may earn, not a penalty, but a reward for his heroics.

Figure 3 presents our metamodel for accountability requirements. An *Org* stands in for an STS, and serves as the context of an accountability requirement. Principals communicate and collaborate within the scope of an Org of which they are members. The crucial function of an Org is to systematize the accountability requirements among its members. An Org may additionally provide an authority to which its members may complain regarding accountability violations by others, and which may apply appropriate sanctions on some members; in computational settings, where Orgs lack coercive capabilities, such sanctions typically include canceling a principal's membership, as in a club [27], or further escalating the complaint against it.

Normally, accountability requirements would be specified with reference to roles in the Org. A principal would enroll in an Org by adopting a role and accepting its applicable accountability requirements. (At the computational level, an agent representing a principal may do a lot of the work but it is the principal who is subject to the requirements [1].) Conceptually, an Org is itself a principal and can participate in another Org by adopting a role. Further, an Org *qua* principal may interact with and enter into accountability relationships with its own members. For example, a hospital is an Org that has contracts with its physicians and nurses.

Although we focus on accountability here, we find it useful to introduce the construct of an *expectation*, which describes what one principal may expect from another. Some expectations acquire additional normative force, e.g., by being institutionalized, thereby becoming accountability requirements. In such cases, the *expectee* and *expecter* would map to the a-giver and a-taker, respectively. For example, a meeting group may follow a convention that the last person out of a room turns off the lights. Such a convention would be a mere expectation until its status is raised, e.g., through some explicit declaration. The challenges of emergence are out of our present scope but the possibility of acquiring accountability requirements through experience is an important enabler of innovations being introduced in STSs.

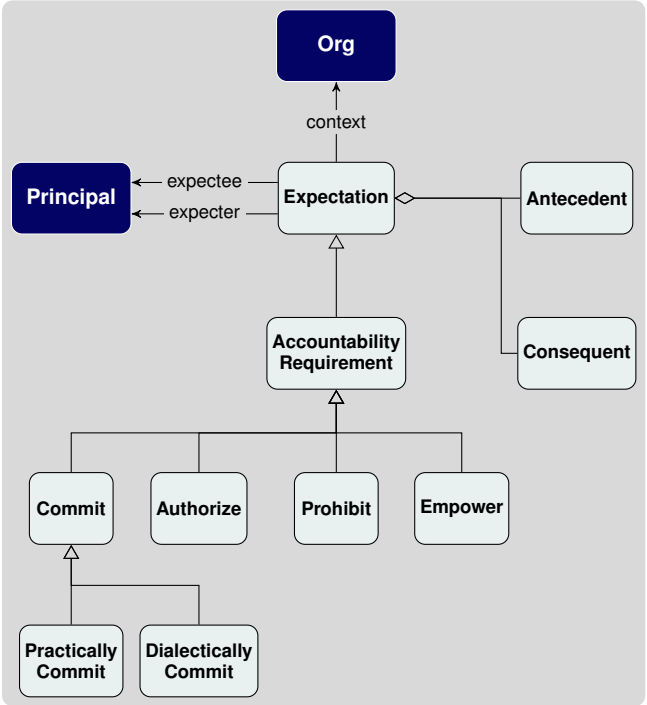

Fig. 3: Accountability requirements metamodel. Accountability refines the relationships of expecter and expectee to a-taker and a-giver, respectively.

An accountability requirement has an antecedent and a consequent; in other words, it is conditional. For example, an invitee is accountable for attending (the consequent) *if* he or she accepts the meeting invitation (the antecedent).

Figure 3 shows the primary kinds of accountability require-

ments. A *commitment* means that its a-giver commits to its a-taker to ensure the consequent if the antecedent holds. (Singh [28] refers to the a-giver and a-taker of a commitment as debtor and creditor, respectively. We adopt the more neutral terminology here to support a unified model for all the requirement types.) In a purchasing contract, commitments are generally prominent in scenarios such as product delivery and payment. Commitments are of two subtypes [29].

- A *dialectical* commitment is a claim staked by its subject, i.e., that the consequent is true if the antecedent is. A party's representations and warranties (e.g., the seller owns what she is selling) are its dialectical commitments. Likewise, an agreement as to the facts is a dialectical commitment by each of the agreeing parties. In an IT setting, an endorsement or certificate is a dialectical commitment. Accountability: The a-giver is the principal who commits; the a-giver is accountable for the truth of the claim: that is, if the antecedent is true so must the consequent be.
- A *practical* commitment is a promise to ensure that the consequent will be brought about if the antecedent becomes true. For example, a seller's offer to a prospective buyer to provide specified goods for a specified payment is a practical commitment. Accountability: The a-giver is the principal who commits; the a-giver is accountable for the success of the claim: that is, if the antecedent is true but the consequent does not become true.

An *authorization* means that its a-taker is authorized by its a-giver for bringing about the consequent if the antecedent holds. Notice that this is consistent with treating authorization as a privilege for the authorized party. The intuition is that an authorization concerns a "physical" action, i.e., a domain-level action as being conceptualized. For example, in a manufacturing contract, the manufacturing facility owner may authorize a client to visit a facility with restricted access. In healthcare, a patient may authorize a radiologist to forward her diagnosis to a primary care physician. Accountability: The a-giver is the granter of an authorization, who is accountable for ensuring it can be exercised. That is, if the grantee of an authorization is unable to exercise it, the grantee has standing to complain against the granter.

A *prohibition* means that its a-giver is forbidden by its a-taker from bringing about the consequent if the antecedent holds. For example, in an employment contract, the employee may be forbidden from revealing the employer's confidential information to outsiders. A prohibition informally appears to be a negation of an authorization, an intuition that traditional deontic logics formalize. In contrast, we make the computationally crucial distinction whereby a technical means such as a resource monitor can enforce an authorization but a prohibition can be enforced only through social means, i.e., through sanctions. Accountability: The a-giver is the principal who is prohibited: that is, if the antecedent and consequent are both true, the prohibition is violated.

A *power* means that its a-taker is empowered by its a-giver to bring about the consequent if the antecedent holds. A power refers to the ability to perform actions that change their normative relationships [23], [30]. That is, a power concerns a "social" or normative action, i.e., an action being conceptualized at a level of the relationships between the principals, and thus above the level of an action in the domain. For example, in a manufacturing contract, the purchaser may cancel an order with prior notice, that is, it can terminate a commitment at will, thereby changing the normative relationship between itself and the manufacturer. In healthcare, a radiologist Alice may empower her radiology fellow Bob to issue a diagnosis for a patient. The existence of a power does not suggest that exercising it is always allowed [30]. For example, Alice may empower Bob only for issuing a diagnosis for a patient who has come in for a routine exam, but not for a patient for whom a tumor is being suspected. Accountability: Like for an authorization, the a-giver is the granter of the power. The granter is expected to ensure that if the antecedent is true, so is the consequent, or else it is deemed to have failed.

Table I shows the mapping of meeting scheduler accountability requirements in Section I on to these types.

IV. Case Study

Abraham and Reddy [8] studied the inter-departmental coordination of patient transfers in a large academic hospital (501 beds; 50,000 Emergency Department visits per year). We begin with their description of the *idealized* patient transfer process, then discuss the specific coordination problems they note, and finally demonstrate how representing and reasoning about accountability requirements can help address these problems.

A. Interdepartmental Patient Transfer Workflow

The departments involved in the study were the Emergency Department (ED), the Neurosciences Department (NSD), and the Inpatient Access Department (IPA). Figure 4, reproduced from [8] depicts the patient transfer workflow. The admitting physician first enters a patient transfer order in the Electronic Medical Record (EMR) system. An IPA staff member uses the bed tracking system (BTS) to assign a bed to the patient. The staff then notifies the charge nurse (CN) of the department that has to receive the patient. If the bed is confirmed by the receiving department, then the staff notifies the CN of the sending department. The sending department transfers the necessary patient information to the receiving department. The sending department finally arranges the physical transfer of the patient.

Abraham and Reddy observe that the foregoing scenario suffers from the following shortcomings.

$\mathcal{P}_1$. The clinical staff in the departments have significant control over patient care activities. The IPA, however, has complete authority in making bed assignment decisions. This leads to a sense of disability among the clinical staff and affected their interactions with IPA staff adversely. In particular, this led to inappropriate patient transfers.

$\mathcal{P}_2$. Priorities vary across departments. ED's priority is quick patient turnaround to accommodate the constant influx of

TABLE I: Meeting scheduling accountability requirements.

Label	Type	A-Giver	A-Taker	Accountable For
$\mathcal{A}_1$	Dialectical C	Initiator	Org	Purpose of meeting being useful and list of invitees
$\mathcal{A}_2$	Prohibition	Initiator	Org	Not scheduling before 10AM and after 3PM
$\mathcal{A}_3$	Practical C	Invitees	Initiator	Attending, if accepted
$\mathcal{A}_4$	Practical C	Initiator	Org	Turning off electrical appliances at the end of the meeting
$\mathcal{A}_5$	Practical C	Invitee	Org	Clearing the room at the end of the meeting

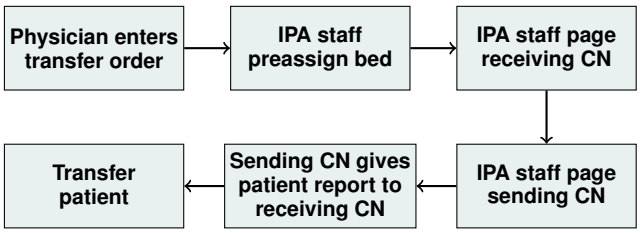

Fig. 4: Operational view of patient transfer [8].

new patients. Inpatient departments, on the other hand, tend to prioritize patient care and long term treatment over rapid patient flow. This creates bottlenecks in the patient flow.

$\mathcal{P}_3$. Further, the inpatient departments were concerned that transfer patients from the ED were not well-kept and their labwork would often be incomplete. The ED CN though did not think that well-kept patients was the ED's responsibility.

$\mathcal{P}_4$. The sending department nurse provides a report to receiving department nurse. However, delays often occur because the receiving department nurse would not be ready to take report. This often led to patients staying on hours longer than necessary in the ED. This conflicts with the ED's requirement of quick turnaround for patients.

$\mathcal{P}_5$. Often, because of the delays, the sending department nurse would forget to mention important patient details when the receiving department nurse eventually called back.

$\mathcal{P}_6$. All clinical departments must provide bed availability information to the IPA. But they do not always to so for various reasons. For example, nurses wanted to avoid an hour's worth of cumbersome work when close to a shift change.

$\mathcal{P}_7$. Hidden transfers, that is, transfers without the knowledge of the IPA, occur sometimes. The sending department's motivation is that if the available bed is hidden, then the IPA can't key it in as *dirty*.

$\mathcal{P}_8$. Even though only the IPA can make bed assignment decisions, all clinical departments have access to the information provided by the BTS. The universal availability of this information sometimes led to conflicts between the clinical departments and the IPA. Abraham and Reddy document a case where the CN of the sending department was unhappy about a bed assignment made by the IPA. The conflicts led the CNs to take actions such as withholding information regarding beds, refusing to accept transfers, delaying discharges in their departments in order to avoid new admissions, and so on.

$\mathcal{P}_9$. A related challenge raised by the availability of information from the BTS was that a department could initiate a transfer based on the information but without involving the IPA and the receiving department. As a result, patients would arrive unexpectedly at the receiving department. This would disrupt the normal working of the receiving department and the IPA would have to take mitigating actions to resolve the situation.

$\mathcal{P}_{10}$. Finally, the EMR does not notify the CN that the physician has entered a transfer order. Nurses must "poll" the status of their patients in the EMR system to learn of these orders. This sometimes resulted in the nurses learning of the transfer order only when the IPA informed them that a bed was available to move the patient. This results in delays in patient transfer.

B. Solution

Many problems in traditional organizations, such as the hospital in the case study, arise due to the ambiguity of the mutual accountabilities of the principals. In particular, roles are conceived not in terms of accountability requirements, but in terms of the "job". Hierarchies in organizations further complicate the picture: hierarchical structures neglect interactions that cut across hierarchies, but which is where most of the work is done in practice. Our solution makes explicit the accountabilities of the various principals: accountability requirements are not top-down relations but peer-to-peer, as Figure 5 illustrates.

$\mathcal{A}_6$. $\mathcal{P}_1$ is mitigated by making the IPA dialectically committed (therefore accountable) to the clinical departments for bed assignment decisions.

$\mathcal{A}_7$. $\mathcal{P}_2$ is really a resource problem. The thing to note is that the hospital is committed to the patient for providing adequate care. All departments are in turn committed to the hospital for providing care to the patient. This acts as a balance in the drive for turning patients around quicker. Notice that Abraham and Reddy treat the patient as an "object" of interaction among other principals. When we think of accountability requirements, however, the patient appears as a principal on par with other principals involved in the transfer.

$\mathcal{A}_8$. $\mathcal{P}_3$ is a case of ambiguous responsibility. It is potentially addressed by making the ED practically committed to the receiving departments for providing a complete patient report. However, concerns of work overload (a likely

reason for incomplete reports) are legitimate. A balance could be achieved by making the hospital practically committed to all departments for monitoring workloads.

$\mathcal{A}_9$. $\mathcal{P}_4$ is likely a problem of resources. The hospital could address this if it had a record of the problem. This can be done if the sending department CN is committed to sending the report (electronically and asynchronously) to the receiving department CN as soon as he or she is ready with the information.

$\mathcal{A}_{10}$. $\mathcal{A}_9$ would also address $\mathcal{P}_5$.

$\mathcal{A}_{11}$. $\mathcal{P}_6$ could be addressed by making the nurses committed to the hospital for updating bed availability information as soon as the bed is free. Although this does mean nurses will actually do so, but it gives the hospital formal grounds for holding them accountable.

$\mathcal{A}_{12}$. In addition to $\mathcal{A}_{11}$, making the sending and receiving departments practically committed to notifying the IPA about the transfer would help address $\mathcal{P}_7$.

$\mathcal{A}_{13}$. $\mathcal{P}_8$ is potentially addressed by $\mathcal{A}_6$ above.

$\mathcal{A}_{14}$. $\mathcal{P}_9$ could be mitigated by $\mathcal{A}_{12}$.

$\mathcal{A}_{15}$. $\mathcal{P}_{10}$ can be addressed by the having the IT Operator practically commit to notifying nurses of transfer orders. This means that the IT Operator must implement the patient scheduling software accordingly.

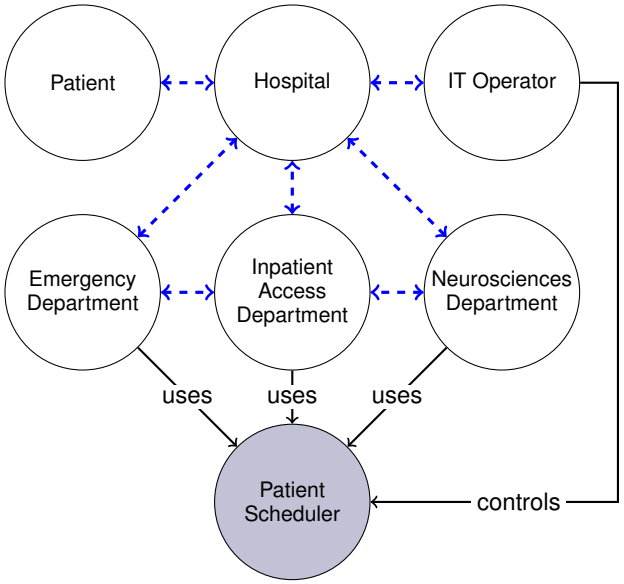

Fig. 5: Accountability view of Abraham and Reddy's scenario. The dotted lines indicate the various accountabilities ($\mathcal{A}_6$–$\mathcal{A}_{15}$).

Modeling accountability requirements in patient transfer does not mean that problems $\mathcal{P}_1$–$\mathcal{P}_{10}$ will be resolved. After all, one cannot force principals to fulfill their requirements. However, conceiving a system in terms of accountabilities and having an accountable principal for every requirement grounds the requirements in the organizational context, leads to greater awareness of mutual expectations, and provides a basis for systematic analysis of organizational problems.

C. Limitations of Workflow-Oriented Representations

It is instructive to note the misalignment between organizational problems (as indicated in $\mathcal{P}_1$–$\mathcal{P}_{10}$) and representations such as workflows for modeling organizational processes (as in Figure 4). One of the key factors is the limitation of workflows in faithfully capturing the sociotechnical aspects of the STS. Workflow management and other traditional techniques emphasize the software modules and their functionality and deemphasize the accountability relationships, thereby making it difficult for intelligent and cooperative participants to interact effectively.

As a case in point, exceptions in particular are not modeled readily in workflows. Figure 4, for example, does not say what would happen if the receiving department were to not confirm the bed or if the physician were to cancel the transfer order.

Neither is it clear who is accountable to whom for what at various stages of the patient transfer process. If the physician places a transfer order, but the patient is not transferred, who would be accountable? And to whom? Would the physician be accountable to the patient because he or she placed the order? Would the IPA staff be accountable to the physician for ensuring and confirming the physical transfer of the patient? Workflows, including the one in Figure 4, lack a basis for answering such questions. Representations based on accountability requirements (of which Figures 2 and 5 are examples) represent an alternative to workflows.

V. Discussion

We considered the setting of STS involving social interaction among autonomous principals; we argued that in such STSs, conceptualizing requirements in terms of accountability relationships among the principals offers considerable benefits. We discussed the limitations of existing approaches in RE. Specifically, even though they consider social and organizational factors, the accountabilities of principals find no formal representation—neither in the user software nor in the broader STS. Approaches such as i* and Tropos consider relational requirements (dependencies), however, they have two shortcomings. They say both too much (about actor internals) and too little (about accountability). We presented a metamodel for accountability requirements and several kinds of accountabilities such as commitments, prohibitions, and authorizations. We also modeled an interdepartmental process for patient transfers in hospitals and demonstrated the benefits of introducing clear accountability relationships. Below, we discuss connections with broader literature and conclude with directions of work.

A. Multiagent Systems

Research in multiagent systems provides some of the elements our approach. Artikis et al. [31] specify institutions, analogous to our Orgs, with roles associated with normative relationships such as powers, permissions, prohibitions, and obligations. Their enforcement policies for norm violations apply at the level of an institution. They encode specifications of institutions in causal logic, whose implementation may be

used to run queries regarding applicable norms given certain actions have been executed. Our accountability requirements are similar to their norms. Vasconcelos et al. [32] model an organization as applying norms, thereby infringing on principals' autonomy. Their norms are undirected and thus unable to express accountability. Vasconcelos et al. do not provide an account of violations or sanctioning. However, they address the important problem of resolving conflicts among norms, which can arise when a principal plays two or more roles.

B. Security

Deterrence is sometimes used as equivalent to accountability. Feigenbaum et al. [33], [34] treat accountability as the negative utility accrued by the accountable party for failing to act as expected. Consider this example to understand the shortcomings of this view. A nurse Bob is prohibited from giving a Schedule III Controlled Substance to a patient Charlie without a prescription from a physician Alice. Let's suppose Bob risks losing his bonus if he violates the prohibition. First, negative payoffs may serve as a deterrent but in providing an assurance mechanism, they remove accountability. In essence, instead of accountability requirement *A*, Bob is accountable for the requirement "*A*, but if you violate *A*, then penalty." He need no longer give an account for violating *A* provided he pays the penalty. Second, seeing that Charlie is flatlining, Bob may know that the probability of punishment is zero, but that doesn't mean Bob is not accountable for administering controlled drugs. Third, sanctioning (including rewarding) an accountable party is a process that is subsequent to accountability, not incorporated in its definition [13], [2]. Indeed, Bob could potentially be rewarded if his quick action saves Charlie's life.

Access control mechanisms, such as role-based access control (RBAC) [35], assume social relationships between principals expressed via roles. However, these relationships are compiled out and reduced merely to an attribute check in policy mechanism, which regiments access. In contrast, in our approach, relationships expressed via accountability requirements are present in the computational machinery. Thus a principal may act flexibly in light of the progressing social state, overriding the requirements when it needs to.

The newer approaches demonstrate powerful features such as dynamic attributes (e.g., of ongoing activities) and proactive decision making. Park et al. [36] separate the main activities of users from administrative activities (performed by users or on their behalf). Their approach helps express and enforce policies that capture the preferences of users in terms of how they interact with others and how they wish to modulate the interactions of others, for example, when a parent controls the policies by which a child interacts with others. Chen et al. [37] incorporate risk assessment in decision making, supporting policies being violated when necessary provided a responsible party takes on an obligation to clean up after the fact.

Our approach in the present paper supports the above cases but supports greater precision in representation and flexibility in reasoning by formulating accountability requirements explicitly and prominently, including how accountability requirements apply to the creation, manipulation, and violation of other accountability requirements. It ensures accountability: every action is performed by an explicit decision by a principal. For example, suppose there were a resource capacity limit. Traditionally, a resource monitor may deny requests once the capacity is reached. In our approach, the principal decides how to deal with a request; therefore, the principal could (and should in general) be made accountable for that decision.

C. Norms and Sanctions

The idea of norms and sanctions bear discussion in relation to accountability requirements. Barth et al. [38] use the term "norms" to describe constraints on the transmission of information. A norm may either allow or disallow sharing. These norms are not directed. Further, the "context" they refer to is simply a set of roles, and lacks the normative (and hence accountability) representation of our approach. Pieters and Coles-Kemp [39] give various examples of how cultural norms may conflict with an organization's security policies. They recommend modeling and analyzing security policies in light of relevant norms. Pieters and Coles-Kemp highlight the challenge of ascribing responsibility in case of conflicting norms.

A *sanction* specifies the penalties or rewards its a-giver faces from its a-taker because of the state of another norm. The sociological literature, which we follow, considers both positive and negative sanctions conveying approvals and disapprovals, respectively [40], [41]. In healthcare, a physician who violates a prohibition against prescribing addictive pain killers to children may be sanctioned by having her board certification revoked. Our approach support sanctions being specified on par with other accountability relationships. Sanctions to be applied by a community [42] can be captured because an Org can be the party that applies it. In general, the sanctioning process can go beyond the specified sanctions, e.g., if a principal autonomous decides to shun another. Such behaviors are possible because each principal is autonomous. Of course, if Alice sanctions Bob illegitimately (violating a prohibition, say) may herself face sanctions.

Important directions of future work include (1) expressive accountability requirements and their formalization, (2) methodologies for deriving and modeling STS specifications in terms of accountability requirements, and (3) methodologies for deriving software specifications from accountability requirements.

Acknowledgments. We thank the anonymous reviewers of RE 2014 for their comments and suggestions, which have helped improve this paper.

References

[1] E. A. Mamdani and J. Pitt, "Responsible agent behavior: A distributed computing perspective," *IEEE Internet Computing*, vol. 4, no. 5, pp. 27–31, Sep. 2000.

[2] R. W. Grant and R. O. Keohane, "Accountability and abuses of power in world politics," *American Political Science Review*, vol. 99, no. 1, pp. 25–43, Feb. 2005.

[3] L. L. Emanuel, "A professional response to demands for accountability: Practical recommendations regarding ethical aspects of patient care," *Annals of Internal Medicine*, vol. 124, no. 2, pp. 240–249, Jan. 1996.

[4] K. Argyraki, P. Maniatis, O. Irzak, S. Ashish, and S. Shenker, "Loss and delay accountability for the Internet," in *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, 2007, pp. 194–205.

[5] A. Haeberlen, "A case for the accountable cloud," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 2, pp. 52–57, Apr. 2010.

[6] I. Sommerville, D. Cliff, R. Calinescu, J. Keen, T. Kelly, M. Kwiatkowska, J. Mcdermid, and R. Paige, "Large-scale complex IT systems," *Communications of the ACM*, vol. 55, no. 7, pp. 71–77, Jul. 2012.

[7] M. Hartswood, R. Procter, M. Rouncefield, and R. Slack, "Making a case in medical work: Implications for the electronic medical record," *Computer Supported Cooperative Work (CSCW)*, vol. 12, no. 3, pp. 241–266, 2003.

[8] J. Abraham and M. C. Reddy, "Challenges to inter-departmental coordination of patient transfers: A workflow perspective," *International Journal of Medical Informatics*, vol. 79, no. 2, pp. 112–122, 2010.

[9] J. C. S. do Prado Leite and C. Cappelli, "Software transparency," *Business & Information Systems Engineering*, vol. 2, no. 3, pp. 127–139, 2010.

[10] S. V. Scott and W. J. Orlikowski, "Reconfiguring relations of accountability: Materialization of social media in the travel sector," *Accounting, Organizations and Society*, vol. 37, no. 1, pp. 26–40, 2012.

[11] E. S. K. Yu and J. Mylopoulos, "Understanding "why" in software process modelling, analysis, and design," in *Proceedings of the 16th International Conference on Software Engineering*. IEEE Computer Society Press, 1994, pp. 159–168.

[12] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.

[13] E. J. Emanuel and L. L. Emanuel, "What is accountability in health care?" *Annals of Internal Medicine*, vol. 124, no. 2, pp. 229–239, Jan. 1996.

[14] A. Dardenne, A. V. Lamsweerde, and S. Fickas, "Goal-directed requirements acquisition," *Science of Computer Programming*, vol. 20, no. 1–2, pp. 3–50, 1993.

[15] E. S. K. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering*, 1997, pp. 226–235.

[16] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with goal models," in *Conceptual Modeling—ER 2002*, ser. LNCS, 2003, pp. 167–181.

[17] L. Baresi, L. Pasquale, and P. Spoletini, "Fuzzy goals for requirements-driven adaptation," in *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference*. IEEE Computer Society, 2010, pp. 125–134.

[18] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein, "Requirements-aware systems: A research agenda for RE for self-adaptive systems," in *Proceedings of the 18th IEEE International Requirements Engineering Conference*, 2010, pp. 95–103.

[19] F. Dalpiaz, P. Giorgini, and J. Mylopoulos, "Adaptive socio-technical systems: A requirements-driven approach," *Requirements Engineering*, in press.

[20] T. D. Breaux, D. G. Gordon, N. Papanikolaou, and S. Pearson, "Mapping legal requirements to IT controls," in *Proceedings of the Sixth International Workshop on Requirements Engineering and Law (RELAW)*, 2013, pp. 11–20.

[21] X. Gao and M. P. Singh, "Extracting normative relationships from business contracts," in *Proceedings of the 13th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, Paris, May 2014, pp. 1–8, to appear.

[22] A. Siena, G. Armellin, G. Mameli, J. Mylopoulos, A. Perini, and A. Susi, "Establishing regulatory compliance for information system requirements: An experience report from the health care domain," in *Proceedings of the 29th International Conference on Conceptual Modeling*, ser. LNCS, vol. 6412. Springer, 2010, pp. 90–103.

[23] W. N. Hohfeld, *Fundamental Legal Conceptions as Applied in Judicial Reasoning and other Legal Essays*. New Haven, CT: Yale University Press, 1919, a 1919 printing of articles from 1913.

[24] S. Ghanavati, D. Amyot, and L. Peyton, "Towards a framework for tracking legal compliance in healthcare," in *Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE)*, ser. LNCS, vol. 4495. Springer, 2007, pp. 218–232.

[25] A. Rifaut and E. Dubois, "Using goal-oriented requirements engineering for improving the quality of ISO/IEC 15504 based compliance assessment frameworks," in *Proceedings of the 16th IEEE International Requirements Engineering Conference*. IEEE, 2008, pp. 33–42.

[26] T. D. Breaux, A. I. Antón, and E. H. Spafford, "A distributed requirements management framework for legal compliance and accountability," *Computers & Security*, vol. 28, no. 1–2, pp. 8–17, 2009.

[27] J. M. Buchanan, "An economic theory of clubs," *Economica*, vol. 32, no. 125, pp. 1–14, Feb. 1965.

[28] M. P. Singh, "An ontology for commitments in multiagent systems: Toward a unification of normative concepts," *Artificial Intelligence and Law*, vol. 7, no. 1, pp. 97–113, Mar. 1999.

[29] T. J. Norman, D. V. Carbogim, E. C. W. Krabbe, and D. N. Walton, "Argument and multi-agent systems," in *Argumentation Machines*, C. Reed and T. J. Norman, Eds. Kluwer, 2004, ch. 2.

[30] A. J. I. Jones and M. J. Sergot, "A formal characterisation of institutionalised power," *Logic Journal of the IGPL*, vol. 4, no. 3, pp. 427–443, Jun. 1996.

[31] A. Artikis, M. J. Sergot, and J. V. Pitt, "Specifying norm-governed computational societies," *ACM Transactions on Computational Logic*, vol. 10, no. 1, pp. 1:1–1:42, Jan. 2009.

[32] W. Vasconcelos, M. J. Kollingbaum, and T. J. Norman, "Resolving conflict and inconsistency in norm-regulated virtual organizations," in *Proceedings of the 6th International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. Honolulu: IFAAMAS, May 2007, pp. 632–639.

[33] J. Feigenbaum, A. D. Jaggard, and R. N. Wright, "Towards a formal model of accountability," in *Proceedings of the 14th New Security Paradigms Workshop (NSPW)*, Marin County, California, Sep. 2011, pp. 45–56.

[34] J. Feigenbaum, J. Hendler, A. D. Jaggard, D. J. Weitzner, and R. N. Wright, "Accountability and deterrence in online life (extended abstract)," in *Proceedings of the 3rd International Web Science Conference*. Koblenz: ACM Press, Jun. 2011, pp. 7:1–7:7.

[35] R. S. Sandhu, "Lattice-based access control models," *IEEE Computer*, vol. 26, no. 11, pp. 9–19, Nov. 1993.

[36] J. Park, R. S. Sandhu, and Y. Cheng, "ACON: Activity-centric access control for social computing," in *Proceedings of the 6th International Conference on Availability, Reliability and Security (ARES)*. Vienna: IEEE, 2011, pp. 242–247.

[37] L. Chen, J. Crampton, M. Kollingbaum, and T. Norman, "Obligations in risk-aware access control," in *Proceedings of the 10th Annual International Conference on Privacy, Security and Trust (PST)*. IEEE Computer Society, 2012, pp. 145–152.

[38] A. Barth, A. Datta, J. C. Mitchell, and H. Nissenbaum, "Privacy and contextual integrity: Framework and applications," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, Berkeley, California, May 2006, pp. 184–198.

[39] W. Pieters and L. Coles-Kemp, "Reducing normative conflicts in information security," in *Proceedings of the 2011 New Security Paradigms Workshop*. New York, NY, USA: ACM, 2011, pp. 11–24.

[40] A. R. Radcliffe-Brown, "Social sanction," in *Encyclopedia of the Social Sciences*, E. R. A. Seligman, Ed. Macmillan Publishers, 1934, vol. XIII, p. 531.

[41] V. Goldschmidt, "Primary sanction behavior," *Acta Sociologica*, vol. 10, no. 1/2, pp. 173–190, 1966.

[42] C. Bicchieri, *The Grammar of Society: The Nature and Dynamics of Social Norms*. Cambridge, UK: Cambridge University Press, 2006.