Accountability as a Foundation for Requirements in Sociotechnical Systems

Amit K. Chopra, *Lancaster University, Lancaster, LA1 4YW, U.K.*
Munindar P. Singh, *North Carolina State University, Raleigh, NC, 27695, USA*

We understand sociotechnical systems (STSs) as uniting social and technical tiers to provide abstractions for capturing how autonomous principals interact with each other. Accountability is a foundational concept in STSs and an essential component of achieving ethical outcomes. In simple terms, accountability involves identifying who can call whom to account and who must provide an accounting of what and when. Although accountability is essential in any application involving autonomous parties, established methods do not support it. We formulate an accountability requirement as one where one principal is accountable to another regarding some conditional expectation. Our metamodel for STSs captures accountability requirements as relational constructs inspired from legal concepts, such as commitments, authorization, and prohibition. We apply our metamodel to a healthcare process and show how it helps address the problems of ineffective interaction identified in the original case study.

We consider *sociotechnical systems (STSs)* as a basis for realizing applications that involve social interaction between two or more autonomous principals. STSs arise in virtually any major application, including hospital organizations, smart cities, and supply chains.

Broadly, we consider *personae iuris*, that is, legal persons, to be autonomous principals. Thus, a principal is a person or an organization. The principals interacting within an STS would nominally be its stakeholders and their interactions would be facilitated via information technology.

Autonomy and accountability are fundamental concepts in understanding STSs. Autonomy means each principal is free to act as it pleases. Accountability classically means the standing of one principal—the *account-taker*—to call upon another another—the *account-giver*—to explain its actions.[1,2] Appropriating some legal language, we might state that an account-giver is modeled as one who is *sui iuris*, i.e., an autonomous party, whereas an account-taker is modeled as one who has *locus standi* or standing (to complain).

The standing of the account-taker arises from the account-taker's expectation, provided it is deemed legitimate, that the account-giver behave in a certain way. For example, a patient may expect a hospital to keep the patient's health records private. In case of a leak, the patient has the standing to demand an explanation and possibly remediation from the hospital. Note that perfect remediation may be impossible in some cases but accountability can serve as a basis for promoting ethical outcomes.

Accountability makes autonomy acceptable: a stakeholder can violate any expectation for which it is accountable, but if so, it would be held to account. Balancing autonomy and accountability is crucial for ensuring that an STS would not devolve into the extremes of chaos or tyranny.

Our contributions are the following. One, we show how precisely to capture accountability requirements between concerned parties in the real world. Anything less strict would lead to unsound solutions; anything stricter would lead to overcoupled solutions. Two, we introduce a metamodel for STSs based on accountability requirements. We apply the metamodel to an

1089-7801 © 2021 IEEE
Digital Object Identifier 10.1109/MIC.2021.3106835
Date of publication 24 August 2021; date of current version 6 December 2021.

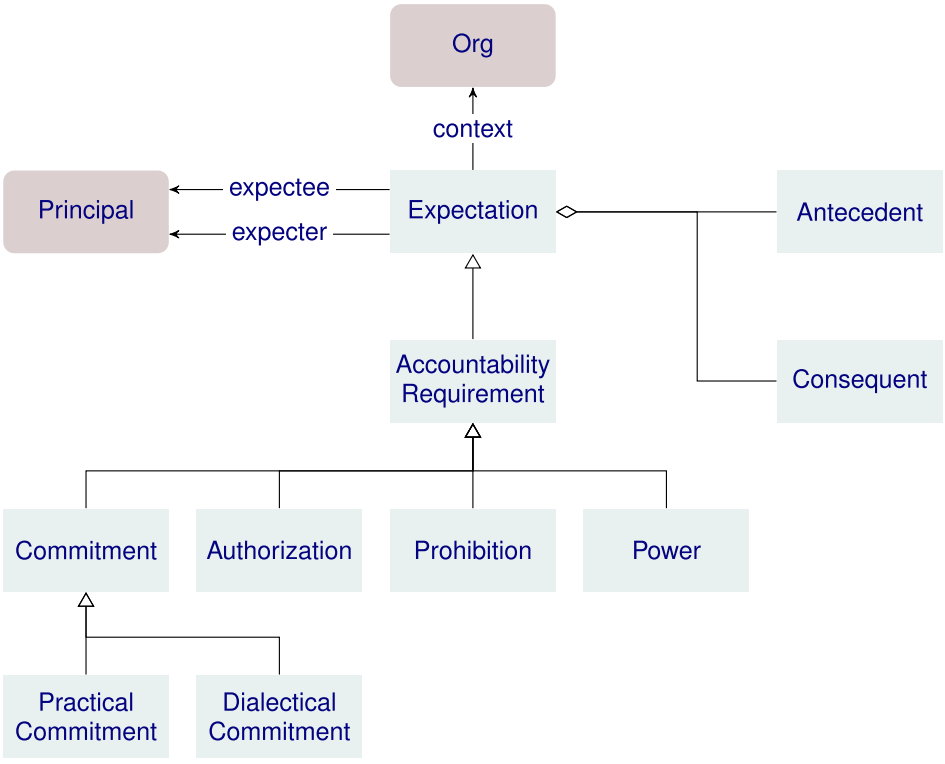

FIGURE 1. Accountability requirements metamodel. Accountability refines the relationships of expecter and expectee to account-taker and account-giver, respectively.

existing extensive healthcare case study and show how it can help mitigate the problems identified by the case study.

A FRAMEWORK FOR ACCOUNTABILITY REQUIREMENTS

Figure 1 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 and make legitimate the accountability requirements its members have of each other. An Org may additionally serve as 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 or escalating the complaint to an Org with wider scope.

Normally, accountability requirements are specified with reference to roles in the Org. A principal would enroll in an Org by adopting a role and accepting the role's applicable accountability requirements. 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 form accountability relationships with its own members. For example, a hospital is an Org that has contracts with its physicians and nurses.

We introduce the construct of an *expectation* as describing what one principal may expect from another. For example, a meeting group may follow a convention that the last person out of a room switches off the lights. Expectations that acquire normative force by being recognized as legitimate within an Org become accountability requirements. For example, declaring the above expectation would raise the convention to an accountability requirement and make whoever is last out of the room accountable for switching off the lights.

We represent accountability via legal norms[3] because they support reasoning and manipulation to support desired accountability processes.[4] Specifically, we adopt a form of directed, conditional, and contextual norms.[5] An accountability requirement is

conditional and expressed via an antecedent and a consequent. For example, a buyer would become accountable for paying for an item (the consequent is paying) *if* the buyer accepts the offered price (the antecedent is accepting the price and receiving the item).

Figure 1 shows the primary kinds of accountability requirements. A *commitment* means that its account-giver commits to its account-taker to ensure the consequent if the antecedent holds. In a typical purchasing contract, commitments for product delivery and payment are prominent. Commitments are of two subtypes.[6]

- A *dialectical* commitment is a claim staked by its subject, i.e., that the consequent is true if the antecedent is. Dialectical commitments model a party's representations and warranties, e.g., that the seller owns the product being sold. Dialectical commitments capture when a party attests to some fact, e.g., in some hospitals, if a nurse calls Code Blue, then it is because their patient has gone into cardiac arrest.
  Accountability: The account-giver is the principal who commits; the account-giver is accountable for the truth of the claim: if the antecedent is true, so is the consequent.
- 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 account-giver is the principal who commits; the account-giver is accountable for ensuring that if the antecedent is true, so is the consequent.

An *authorization* means that its account-taker is authorized by its account-giver for bringing about the consequent if the antecedent holds. Notice that the above formulation is consistent with treating authorization as a privilege for the authorized party.[3] An intuition is that an authorization concerns a "physical" action, i.e., conceptualized as a domain-level action.[5] For example, in a manufacturing contract, the manufacturing facility owner may authorize a client to visit a facility that restricts access. In healthcare, a patient may authorize a radiologist to forward her diagnosis to a primary care physician.

Accountability: The account-giver is the granter of an authorization, and 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 account-giver is forbidden by its account-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, besides observing as indicated above that an authorization is a prohibition on the authorizer,[3] 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 account-giver is the principal who is prohibited. If the antecedent and consequent are both true, the prohibition is violated.

A *power* means that its account-taker is empowered by its account-giver to bring about the consequent if the antecedent holds. A power is an ability to perform actions that change normative relationships.[7] That is, a power concerns a "social" or normative action, i.e., an action conceptualized at a level of the relationships between the principals, and thus above the level of an action in the domain.[5] 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.

Conversely, the creditor may release the debtor from its commitment, which too terminates the commitment and changes the normative relationship between them. 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.[7] 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: As for an authorization, the account-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 the power is deemed to have failed.

CASE STUDY

Abraham and Reddy[8] studied the interdepartmental coordination of patient transfers in a large academic hospital (501 beds; 50,000 Emergency Department visits per year). We introduce their description of the *idealized* patient transfer process; discuss the

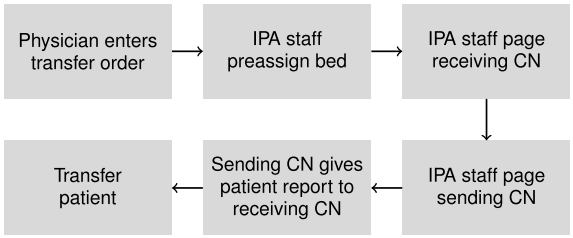

FIGURE 2. Workflow view of patient transfer.[8]

specific coordination problems they observe; and demonstrate how representing and reasoning about accountability requirements helps address these problems.

Interdepartmental Patient Transfer Workflow

The study involved three departments: Emergency, Neurosciences (an inpatient department), and Inpatient Access. The patient transfers were from Emergency (the sending department) to Neurosciences (the receiving department). Figure 2 depicts the patient transfer workflow.[8] The admitting physician first enters a patient transfer order in the Electronic Medical Record (EMR) software. An Inpatient Access staff member uses the Bed Tracking System to assign a bed to the patient. The staff then notifies the Charge Nurse (abbreviated CN in Figure 2) of the receiving department. If the bed is confirmed by the receiving department, the staff notifies the Emergency Charge Nurse. Emergency transfers the necessary patient information to the receiving department. Upon doing so, Emergency proceeds to arrange the physical transfer of the patient.

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

$\mathcal{P}_1$. Although the clinical staff in the various departments control patient care activities, Inpatient Access has complete authority over bed assignment decisions. This arrangement leads to a sense of disability among the clinical staff and can adversely affect their interactions with Inpatient Access staff. Thus, inappropriate patient transfers ensue.

$\mathcal{P}_2$. Priorities vary across departments. Emergency's priority is quick patient turnaround to accommodate the constant influx of new patients whereas the inpatient departments prioritize patient care and long-term treatment over rapid patient flow. This misalignment creates bottlenecks in the patient flow.

$\mathcal{P}_3$. The inpatient departments were concerned that transfer patients from Emergency were not well kept and their labwork would often be incomplete. The Emergency Charge Nurse though did not think that Emergency was responsible for ensuring the patients were well kept.

$\mathcal{P}_4$. Emergency's Charge Nurse provides a report to the Neurosciences Charge Nurse on a patient being transferred to Neurosciences. Delays occur when the Neurosciences Charge Nurse is not ready to receive the report, leading to patients staying in Emergency for hours longer than necessary, conflicting with Emergency's requirement of quick turnaround for patients.

$\mathcal{P}_5$. Often, because of the delays, by the time Neurosciences Charge Nurse accepted a transfer, Emergency's Charge Nurse may have forgotten important patient details and may fail to mention them.

$\mathcal{P}_6$. All clinical departments ought to provide bed availability information to Inpatient Access, but they often fail to do so for various reasons. For example, nurses wanted to avoid an hour's worth of cumbersome work when close to a shift change.

$\mathcal{P}_7$. Transfers can occur without Inpatient Access having been informed. Emergency's motivation is that if an available bed is "hidden," Inpatient Access cannot code it as *dirty*, which would require immediate cleaning.

$\mathcal{P}_8$. Although only Inpatient Access can make bed assignment decisions, all clinical departments have access to the information provided by the Bed Tracking System. The universal availability of this information sometimes can lead to conflicts between the clinical departments and Inpatient Access.

Abraham and Reddy document conflicts between the departments about bed assignments. Such conflicts led the Charge Nurses to withhold information regarding beds, refuse transfers, and delay discharges to avoid new admissions.

$\mathcal{P}_9$. Since the Bed Tracking System published information about beds, Emergency could initiate a transfer without involving Inpatient Access and Neurosciences. Thus, patients would arrive unexpectedly at Neurosciences, disrupting work and requiring Inpatient Access to resolve the situation.

$\mathcal{P}_{10}$. The EMR software does not notify the Charge Nurse that the physician has entered a transfer order. Nurses must "poll" the status of their patients in the EMR software to learn of these orders. This approach sometimes results in

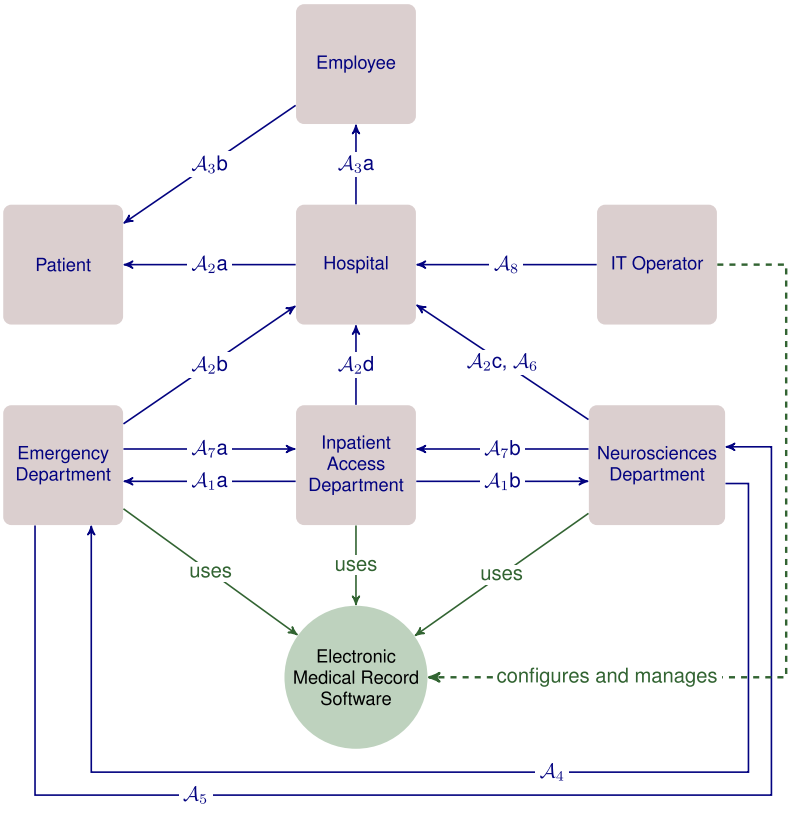

FIGURE 3. Accountability view of Abraham and Reddy's scenario. The dotted lines indicate the various accountability requirements ($\mathcal{A}_1$–$\mathcal{A}_8$) explained in the text with letters appended when a requirement involves more than one pair of principals. The rectangular entities are the principals; the circle refers to the technical infrastructure created and operated by the IT Operator and used by the three departments.

nurses learning of a transfer order only when Inpatient Access informs them that a bed was available to move the patient, and causes delays in patient transfer.

The foregoing problems appear as process errors, but are also indicative of ethical failures in the STS. For example, the staff face perverse incentives that lead them to falsify information about available beds or send patients unannounced, potentially endangering them or at least lowering the quality of their care.

Solution Based on Accountability

Many problems in traditional organizations, such as the hospital in the above case study, arise due to unclarity in accountability requirements. Organizational hierarchies further complicate the picture: hierarchical structures neglect accountability-inducing work-related interactions that cut across hierarchies. Our solution makes explicit the accountability requirements on the various principals. The accountability requirements ($\mathcal{A}_1$–$\mathcal{A}_8$) are peer-to-peer relations, as Figure 3 illustrates.

$\mathcal{A}_1$. $\mathcal{P}_1$ can be alleviated by making Inpatient Access practically committed, and thus accountable, to both the receiving and sending departments to ensure that bed assignments remain in line with the reasonable constraints of the departments regarding availability and time needed to receive and settle a patient in.

$\mathcal{A}_2$. $\mathcal{P}_2$ is a conflict in objectives. The hospital is committed to the patient for providing adequate care. Each department is committed to the hospital for providing care to the patient. These requirements counterbalance the drive for turning patients around quicker. Notice that Abraham and Reddy treat the patient as an "object" of interaction among the principals. When we think of

accountability requirements, however, the patient appears as a principal on par with other principals involved in the transfer.

- $\mathcal{A}_3$. $\mathcal{P}_3$ is a case of ambiguous responsibility. It is potentially addressed by making Emergency practically committed to the patient to and ensure their labwork was complete. However, work overload is a legitimate reason for patients not being well kept. Therefore, in addition, we make the hospital empower each employee to decline assignments above their capacity. That is, the hospital (and thus its departments are accountable) to their employees and employees can decline assignments when necessary. We can imagine the hospital would have negotiated these terms with the employees' union or the governmental labor board.
- $\mathcal{A}_4$. $\mathcal{P}_4$ is addressed by making Neurosciences authorize Emergency to upload the report (electronically and asynchronously) to Neurosciences. Hence, the Emergency Charge Nurse would not be stuck waiting for the Neurosciences Charge Nurse. This norm concerns and presupposes a suitable technical tier.
- $\mathcal{A}_5$. $\mathcal{P}_5$ is mitigated by making Emergency committed to Neurosciences to upload the report for any patient being transferred in a timely manner. Although a commitment is no guarantee against forgetfulness, making the accountability explicit should lead Emergency to institute processes to satisfy this commitment.
- $\mathcal{A}_6$. $\mathcal{P}_6$ is addressed by making Neurosciences committed to the hospital for updating bed availability information as soon as a bed comes free. The commitment does not guarantee compliance, but gives the hospital grounds for holding Neurosciences accountable.
- $\mathcal{A}_7$. Making the sending and receiving departments practically committed to notifying Inpatient Access about the transfer helps address $\mathcal{P}_7$ and $\mathcal{P}_9$.
- $\mathcal{A}_8$. $\mathcal{P}_{10}$ is addressed by having the IT operator practically commit to the hospital for notifying departments of transfer orders. This means that the IT operator must implement or configure the EMR software accordingly.
- $\mathcal{A}_9$. $\mathcal{P}_8$ is addressed partly by making all parties in the hospital dialectically committed to the patient for acting in the best interests of the patient. And, partly it is addressed by the other accountability requirements stated above that address the transfers and concerns such as workload.

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

DISCUSSION

Autonomy and accountability are two sides of the same coin. We showed how STSs can be modeled in terms of accountability requirements expressed via normative abstractions and how such modeling can help address organizational problems. Accountability is a useful approach for realizing effective STSs, abstracting away from implementations of individual agents. Even in STSs, accountability is beneficial when relevant technical (e.g., monitoring and reasoning) components exist and the principals are willing to hold each other to account.

Identifying Accountability Requirements

An important question is how the accountability requirements relevant to a given sociotechnical setting may be determined. Figure 3 is reminiscent of influential requirements modeling approaches such as Tropos.[9] Their notion of *goal dependency* captures a relationship between the actors in a system being specified. For example, in Tropos, one can model that Patient *depends* upon Physician for the goal of being examined. A Tropos dependency, however, does not capture accountability. In contrast, an accountability requirement is what justifies a dependence.

Accountability requirements reconcile the needs of autonomous stakeholders. Protos[10] provides an abstract method for doing so for commitments. The stakeholder requirements may be extracted from user stories[11] or online discussions.[12]

Our norm language is based on legal constructs,[3] but not all norms have the force of law. For example, the patient transfer requirements given above do not. However, requirements protecting patient privacy, ensuring adequate care, or against false billing would have the force of law. Our approach provides a way to build software that reflects legal requirements when appropriate.

Operationalizing Accountability

To implement accountability, in our recommended approach, we would begin by expressing accountability requirements in a norm language.[13] Depending on the setting, we may operationalize the norms via a reasoner that agents helping different principals can use or generate a decentralized multiagent system,[14] where the schemas and meanings of messages exchanged by the agents are based on the norms.

Parvizimosaed *et al.*[15] show how to formalize contracts via a language that resembles ours. In carrying out its part of a contract, each party would realize the relevant accountability requirements. One key difference is that Parvizimosaed *et al.*[15] apply external triggers for norms, which therefore carry no normative force, whereas our norms are self-contained. We accompany accountability with role capabilities, which determine what computations may be produced by the agents playing the various roles when an STS is instantiated.[14]

Business processes can operationalize accountability if they are modeled based on commitments and allied constructs that reflect accountability.[16] However, existing approaches "compile out" the accountability requirements into procedural representations. Thus, they cannot support runtime reasoning about accountability, e.g., by an agent who is deciding how to exercise its autonomy, and potentially violate a requirement that it is accountable for.[17]

Autonomy, Violability, Ethics, and Trust

To support autonomy means that accountability requirements may be violated.[17] A violation may be desirable for the agent when the accountability requirement conflicts with some other agent requirement. For example, a physician may miss a routine appointment with a patient because they are called into an urgent consultation at the hospital.

Accountability requirements enable trust by providing a basis for determining correct behavior and, in case of violation, any accounting to be provided. A violation by an agent followed by inadequate account for the violation would normally be a basis for diminished trust in that agent. For example, a patient may lower their trust in a physician who misses appointments but if the patient learns it was for an emergency involving another patient, they may trust the physician more for having acted to help a patient. A violation may thus help produce ethical outcomes.

Current work on AI ethics primarily considers *micro ethics* or the decision making of individual agents whereas important challenges arise in *macro ethics*, i.e., at the level of an STS.[2] We posit that modeling accountability requirements would simplify realizing STSs that yield improved ethical outcomes. As Figure 3 shows, we can identify concerned humans (the patient) as principals and make clear that there is an accountable principal behind whatever a technical entity (the software) seems to do. Further, expressing interactions via accountability gives the principals maximal flexibility, including to violate any requirements they find objectionable. Additionally, a violation (accompanied by an accounting for it) can provide a basis for innovating—revising the STS, e.g., by determining what information to gather and what norms to revise.[2,4]

Cranefield *et al.*[18] state that an accountable agent must be able to deal with expectations; explain its reasoning, including via argumentative dialogs; adapt its reasoning mechanisms; and accommodate human values. We agree that these are useful capabilities to realize an STS using agents who respond to accountability requirements. An important direction would be an approach by which stakeholders may negotiate the norms in an STS, e.g., to remove some of the current norms that often need to be violated for good reason.

Hewitt's *ORGs*[19] are similar to our Orgs in being motivated from principles of real-world organizations and emphasize accountability. Hewitt discusses the ramifications of inconsistencies between the parties concerned but lacks a formal model for accountability and norms that we develop.

Accountability Versus Traceability and Deterrence

Our approach contrasts with traditional computing approaches that formalize accountability in ways that neglects its core intuitive basis. For example, Haeberlen[20] considers only traceability of actions, which though an important mechanism for holding someone accountable, is neither necessary nor sufficient for accountability.

Feigenbaum *et al.*[21] conflate accountability with punishment for violating a norm. Such conflation has two drawbacks. One, a principal may be in fact be rewarded for violating a norm (e.g., if, in an emergency, a nurse saves a patient life by administering a prohibited drug). Two, punishing or rewarding an accountable party are processes that apply accountability, but are not incorporated in its definition.[1]

Although we separate violations from sanctioning, we allow additional requirements about sanctioning. For example, given a requirement that a Charge Nurse mark a clean bed as available, a second requirement

could commit the hospital to demoting a Charge Nurse who violates the first requirement. The second requirement is simply another accountability requirement that could be identified during requirements elicitation and analysis.

CONCLUSION

Conceptualizing STSs in terms of accountability requirements promises important benefits in software engineering. One, it can help manage complexity by modularizing requirements: it is enough that principals know their respective accountability requirements and build their software systems accordingly. Two, it can support organization-IT alignment[8] by formally capturing organizational context. Three, it can force transparency, especially concerning how information is obtained and processed,[22] by identifying the social dependencies that might underlie goal dependencies.[10] Indeed, whatever predicate features in an accountability requirement must be *transparent* in that both parties ought to be able to observe that predicate. Developing software engineering methodologies that enable realizing these benefits is a major research opportunity.

ACKNOWLEDGMENT

This work was supported by EPSRC under Grant EP/N027965/1 and the NSF under Grant IIS-1908374.

REFERENCES

1. R. W. Grant and R. O. Keohane, "Accountability and abuses of power in world politics," *Amer. Political Sci. Rev.*, vol. 99, no. 1, pp. 25–43, Feb. 2005.
2. A. K. Chopra and M. P. Singh, "Sociotechnical systems and ethics in the large," in *Proc. AAAI/ACM Conf. Artif. Intell., Ethics, Soc.*, Feb. 2018, pp. 48–53.
3. G. H. Von Wright, "Deontic logic: A personal view," *Ratio Juris*, vol. 12, no. 1, pp. 26–38, Mar. 1999.
4. M. Baldoni, C. Baroglio, R. Micalizio, and S. Tedeschi, "Robustness based on accountability in multiagent organizations," in *Proc. 20th Int. Conf. Auton. Agents MultiAgent Syst.*, May 2021, pp. 142–150.
5. M. P. Singh, "Norms as a basis for governing sociotechnical systems," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 1, pp. 21:1–21:23, Dec. 2013.
6. T. J. Norman, D. V. Carbogim, E. C. W. Krabbe, and D. N. Walton, "Argument and multi-agent systems," *Argumentation Machines: New Frontiers in Argument and Computation, Ser. Argumentation Library*, vol. 9, C. Reed and T. J. Norman, Eds. Norwell, MA, USA: Kluwer, 2004, ch. 2, pp. 15–54.
7. A. J. I. Jones and M. J. Sergot, "A formal characterisation of institutionalised power," *Log. J. IGPL*, vol. 4, no. 3, pp. 427–443, Jun. 1996.
8. J. Abraham and M. C. Reddy, "Challenges to interdepartmental coordination of patient transfers: A workflow perspective," *Int. J. Med. Inform.*, vol. 79, no. 2, pp. 112–122, Feb. 2010.
9. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *J. Auton. Agents Multi-Agent Syst.*, vol. 8, no. 3, pp. 203–236, May 2004.
10. A. K. Chopra, F. Dalpiaz, F. B. Aydemir, P. Giorgini, J. Mylopoulos, and M. P. Singh, "Protos: Foundations for engineering innovative sociotechnical systems," in *Proc. 22nd IEEE Int. Requirements Eng. Conf.* Karlskrona, Sweden: IEEE Computer Society, Aug. 2014, pp. 53–62.
11. T. Günes and F. B. Aydemir, "Automated goal model extraction from user stories using NLP," in *Proc. 28th IEEE Int. Requirements Eng. Conf.*, 2020, pp. 382–387.
12. G. M. Kanchev, P. K. Murukannaiah, A. K. Chopra, and P. Sawyer, "Canary: Extracting requirements-related information from online discussions," in *Proc. 25th IEEE Int. Requirements Eng. Conf.*, 2017, pp. 31–40.
13. A. K. Chopra and M. P. Singh, "Custard: Computing norm states over information stores," in *Proc. 15th Int. Conf. Auton. Agents Multiagent Syst.*, May 2016, pp. 1096–1105.
14. M. P. Singh and A. K. Chopra, "Clouseau: Generating communication protocols from commitments," in *Proc. 34th Conf. Artif. Intell.*, Feb. 2020, pp. 7244–7252.
15. A. Parvizimosaed, S. Sharifi, D. Amyot, L. Logrippo, and J. Mylopoulos, "Subcontracting, assignment, and substitution for legal contracts in Symboleo," in *Proc. 39th Int. Conf. Conceptual Model.*, vol. 12400, Nov. 2020, pp. 271–285.
16. K. Taveter and G. Wagner, "Agent-oriented enterprise modeling based on business rules," in *Proc. 20th Int. Conf. Conceptual Model.*, vol. 2224, 2001, pp. 527–540.
17. M. P. Singh and A. K. Chopra, "Computational governance and violable contracts for blockchain applications," *Computer*, vol. 53, no. 1, pp. 53–62, Jan. 2020.
18. S. Cranefield, N. Oren, and W. W. Vasconcelos, "Accountability for practical reasoning agents," in *Proc. 6th Int. Conf. Agreement Technol.*, vol. 11327, Dec. 2018, pp. 33–48.
19. C. Hewitt, "ORGs for scalable, robust, privacy-friendly client cloud computing," *Internet Comput.*, vol. 12, no. 5, pp. 96–99, 2008.
20. A. Haeberlen, "A case for the accountable cloud," *ACM SIGOPS Oper. Syst. Rev.*, vol. 44, no. 2, pp. 52–57, Apr. 2010.

21. J. Feigenbaum, J. Hendler, A. D. Jaggard, D. J. Weitzner, and R. N. Wright, "Accountability and deterrence in online life (extended abstract)," in *Proc. 3rd Int. Web Sci. Conf.*, Jun. 2011, pp. vol. 7, pp 1–7:7.
22. J. C. S. do Prado Leite and C. Cappelli, "Software transparency," *Bus. Inf. Syst. Eng.*, vol. 2, no. 3, pp. 127–139, 2010.

AMIT K. CHOPRA is a Senior Lecturer with the School of Computing and Communications, Lancaster University, U.K. His interests include software engineering and decentralized multiagent systems. Contact him at amit.chopra@lancaster.ac.uk.

MUNINDAR P. SINGH is a Professor in Computer Science and a Co-Director of the Science of Security Lablet, NC State University. His research interests include AI ethics and formal models of sociotechnical systems. He is a former Editor-in-Chief of *IEEE Internet Computing* and ACM Transactions on Internet Technology. He is a Fellow of the IEEE, AAAI, and AAAS, and a Member of Academia Europaea. He is the corresponding author of this article. Contact him at singh@ncsu.edu.