# Sociotechnical Trust: An Architectural Approach

Amit K. Chopra, Elda Paja, and Paolo Giorgini

Department of Information Engineering and Computer Science,
University of Trento, Italy
{chopra,paja,paolo.giorgini}@disi.unitn.it

**Abstract.** Current approaches in sociotechnical systems consider trust to be either *cognitive*—referring to actors' mental models of each other—or *technical*—referring to an actor's trust of a technical artifact. In this paper, we take a more expansive view of trust: in addition to the cognitive, we also consider trust in the *architectural* sense. Broadly, architectural trust applies at the level of roles. Our principal claim is that sociotechnical systems are essentially specified in terms of architectural trust. Whereas previous work has considered dependencies between actors as a fundamental social relation, we claim that no dependency can exist without the corresponding architectural trust.

## 1 Introduction

Sociotechnical Systems (STS) are complex: they consist of humans, organizations, and their information systems. While STS have been around for a while, sociotechnical approaches to building such systems are still in their infancy. Traditional approaches to building information systems focus solely on operational aspects; sociotechnical approaches would require taking into account even the social relationships among the actors involved. One of the key challenges in software engineering is coming up with the right abstractions for modeling sociotechnical systems. Various high-level models for STS have been proposed—for example, intentional [3], goal-oriented [19], based on responsibility [18], and so on.

In this paper, we propose *sociotechnical trust* as the fundamental social relationship among actors of an STS. When we say *sociotechnical trust*, we refer to the trustworthiness of an STS. Our key intuition is that STS are organized along trust relationships. Any STS exists in the first place because actors are not omnipotent; they necessarily depend on each other to get things done. It is trust that makes dependence on others reasonable. When an actor trusts another for something, it expects the latter to do that thing. Lacking such a trust relationship, an actor can hardly depend on another for anything. We want to make explicit these trust relationships in order to provide prospective participants with information regarding the dependencies they would have upon the STS in case they decide to interact with it. For example, let's consider a healthcare STS. Alice would not make a payment to ModernLabs if she did not trust that ModernLabs would deliver test results if she paid. We use the notation $\mathsf{T}(\mathsf{Alice}, \mathsf{ModernLabs}, \mathsf{paid}, \mathsf{delivered})$ to mean that Alice trusts ModernLabs to deliver the results if she has paid for the tests.

This paper is about the engineering of *sociotechnical trust*: how should we design an STS so that it *ensures* trust relationships such as the above between Alice and

M. Jeusfeld, L. Delcambre, and T.W. Ling (Eds.): ER 2011, LNCS 6998, pp. 104–117, 2011.

ModernLabs in the healthcare STS? Ensuring means that we want this relationship to hold not only between Alice and ModernLabs, but between every patient-laboratory pair relationship participating in the healthcare STS. In other words, the trust relationship has to be somehow encoded into the architecture of the STS irrespective of individual actors. Hence, we refer to such trust relationships as *architectural*.

Further, we want to be able to compare two STS in a particular domain, and be able to objectively say which STS fares better from the trust perspective. For example, all other things being equal, one could objectively say that the healthcare system which encodes the above trust relationship between patients and laboratories is better for (all) patients than the system without. Objectively means that anyone, even someone who has no intention or need of participating in the STS being compared, would come to the same conclusion.

We make such comparisons all the time in our day to day lives. For example, we intuitively know that from a customer's point of view, an online marketplace that mandates that merchants refund customers for returned products within a month of purchase is better than one that does not allow such returns; the customers trust the merchants more in the former. It is the same reason why we deem credit card holders to be better off in the credit card system resulting from the passage of the Credit Card Act of 2009 [1]—in the new system, credit card holders now trust that banks will not arbitrarily raise interest rates, and so on. This paper explores the computational basis behind such intuitions.

*Contributions.* Singh proposed the notion of architectural trust [17]; our contributions are in applying this notion to the engineering of STS, for which we coin the term *sociotechnical trust*.

- We characterize the notion of what it means for an STS to ensure more trust than another. We do this via the notion of one trust relationship *supporting* another trust relationship. We give a computational grounding for sociotechnical trust in terms of communication (commitments). Our conceptual model enables one to claim if one system is more trustworthy than another from a particular role's perspective.
- We show the notion of sociotechnical trust to be different from cognitive or technical trust, two kinds of trust relationships commonly modeled in STS software engineering.
- We evaluate our approach through a case study on European food safety law (understood as an STS for ensuring food safety)

*Organization.* The rest of the paper is organized as follows. Section 2 introduces a conceptual model of sociotechnical trust, discussing the computational grounding of trust connectors in terms of commitments. It also discusses what it means to objectively compare STS from a trust perspective, presenting as well a notation for representing the system. Section 3 discusses how sociotechnical trust is different from the prevalent notions of trust and places them all in a single framework. Section 4 evaluates our approach against a case study from the European food safety law. Section 5 concludes the paper with a survey of the literature and future directions.

## 2 A Conceptual Model of Sociotechnical Trust

Our conceptual model emphasizes the social entities abstracted away as roles and the trust relationships among them. In contrast to other approaches in the literature, our conceptual model is notable in that it does not rely upon intentional—varying from actor to actor—concepts such as goals, capabilities, risk, and so on.

Following [6], we conceptualize an STS system as a specification of trust relationships with reference to roles, not particular actors. Individual actors, perhaps completely unknown at the time of the design of STS, would adopt roles in an STS depending on their business requirements and constraints.

Our idea of a *sociotechnical system* is that of a set of roles and the trust relationships that hold between them. Formally, let $\phi$ be a set of symbols. Let $\mathcal{P}$ be the set of all propositions over $\phi$ (including $\top$, the constant for truth) using the connective of propositional logic. Let $p, q, \ldots$ range over $\mathcal{P}$. Let $\mathcal{R}$ be a set of roles; let $x, y, \ldots$ be variables over roles. A system $\mathcal{S}$ is a set such that $S \subseteq \mathcal{R} \times \mathcal{R} \times \mathcal{P} \times \mathcal{P}$. We say $\mathsf{T}(x, y, p, q)$ if and only if $(x, y, p, q) \in \mathcal{S}$.

For instance, taking again into consideration the healthcare domain, one system, let's call it HealthcareSystem1, could be the one composed of the roles Hospital and Patient. In this system, the Patient trusts the Hospital will provide accurate test results (1).

Another system, HealthcareSystem2, includes the roles Hospital, Patient, and Lab. In this setting, the Patient trusts the Hospital will provide accurate test results (2). On the other hand, the Hospital depends on the Lab to analyze test outcomes and produce accurate results (3). Labs are more specialized entities, hence the reason for Hospitals to outsource the tests. Also, the Hospital commits to the Patient that personal data will be confidential and not disclosed to other third parties (4).

$$\mathsf{T}(\mathsf{Patient}, \mathsf{Hospital}, \mathsf{takeTest}, \mathsf{receiveAccurateResults}) \quad (1)$$

$$\mathsf{T}(\mathsf{Patient}, \mathsf{Hospital}, \mathsf{takeTest}, \mathsf{receiveAccurateResults}) \quad (2)$$

$$\mathsf{T}(\mathsf{Hospital}, \mathsf{Lab}, \top, \mathsf{provideAccurateResults}) \quad (3)$$

$$\mathsf{T}(\mathsf{Patient}, \mathsf{Hospital}, \mathsf{providePersonalData}, \mathsf{ensureConfidentiality}) \quad (4)$$

### 2.1 Computational Grounding

Architecturally, sociotechnical trust is what connects two given participants of the STS. We propose a commitment-based approach to engineering sociotechnical trust (in the rest of the paper, when we say *trust* we mean *sociotechnical trust*, unless otherwise specified). Recent advances have proposed commitment-based architectural styles for SOA [15]. The notion of sociotechnical trust exploits the connection between commitments and architecture. Commitments are a simple yet powerful abstraction to model interactions between two agents in terms of a contractual relation [16]. A commitment is a quaternary relation C(debtor, creditor, antecedent, consequent) that stands for a promise made by the debtor to the creditor that if the antecedent is brought about, the consequent will be brought about. Engineering STS for trust means reasoning from a role-based perspective and establish enough commitments into the system so that an agent adopting a role would trust others adopting other roles. Alice trusts ModernLabs for delivery upon

payment, that is, T(Alice, ModernLabs, paid, delivered) if ModernLabs has committed to Alice for doing so, in other words C(ModernLabs, Alice, paid, delivered). In the case of trust relationships, we refer to $x$ and $y$ as the *truster* and *trustee*, respectively, whereas in the case of commitments, we refer to $x$ and $y$ as the *creditor* and *debtor*, respectively. Commitments have been extensively studied in multiagent systems [16], and have been recently applied to understanding the notion of dependencies in systems involving multiple actors [7]. Commitments are rooted in communication: they are created and they evolve when agents exchange messages. Commitments can be *created*, *discharged*, *canceled*, and *released*. $\mathsf{T}(x, y, r, u)$ is created when $\mathsf{C}(x, y, r, u)$ is created. Similarly, the trust is discharged and violated when the corresponding commitment is discharged and violated; analogously for canceled and released.

Within a given domain there is a variety of STS operating on the basis of trust relationships among their actors, as explained by the aforementioned example. They offer different levels of trust with respect to a particular role's perspective. An actor can either *Join* or *Leave* an STS. Joining means binding to a role and therefore instantiating (partially perhaps) the trust relationships in which the role appears. Ideally, an actor should not leave the system without fulfilling the expectations that others have of him by way of instantiated trust relationships. Referring to the two healthcare systems, HealthcareSystem1 and HealthcareSystem2, we could, intuitively and objectively speaking, say that the second system is better or more trustworthy than the first one from the point of view of the Patients, as it includes additional trust relationships that contribute positively to the fulfillment of the basic requirement of receiving accurate test results, thereby enhancing Patients' trust in this system. However, this might not be the case when it come to Hospitals. From their point of view the first healthcare system is better, as it produces only one commitment from their part, that is, provide accurate results to the Patients, whereas the second healthcare system imposes on them more restrictions. They have to ensure accuracy of results and depend on Labs for this activity. Furthermore, to ensure confidentiality of Patients' personal data, they might have to undertake measures to satisfy this requisite.

Essentially, through the conceptual model we want to make clearer the role perspective in order to provide each prospective participant of any of the systems within a given domain with a clear view while choosing the system he wants to play a role at.

However, establishing enough commitments into the system may be not enough; we also need mechanisms that can support them. Having mechanisms for *monitoring* or *enforcing* commitments supports the establishment of trust relations.

### 2.2 Trust Supporting Mechanisms

Some of the trust relationships present in the system influence positively on other trust relationships. We say that these relationships *support* the satisfaction of other relationships, with respect to a given role's perspective, enhancing the trust this role has about the system. We represent this type of relation as in (5). This means that the trust relation T(x,y,p,q) *supports* (positively) the trust relation T(x,z,r,s) from $x$'s perspective.

$$\mathsf{T}(x, y, p, q) \succ \mathsf{T}(x, z, r, s) \tag{5}$$

If we denote $\mathsf{T}$(x,y,p,q) with $T_1$ and $\mathsf{T}$(x,z,r,s) with $T_2$, to express that $T_1$ *supports* $T_2$ from $x$'s perspective, we use $T_1 \succ_x T_2$ (more generally $T_1 \succ_{\mathcal{R}} T_2$, for any $x \in \mathcal{R}$).

But, what is the meaning of *supports* and why it actually increases trust from a role's perspective? Our intuition is that $T_1 \succ_{\mathcal{R}} T_2$ only if $T_1$ handles exceptions that arise from $T_2$. Following this intuition, we specify the *supports* relation through a series of mechanisms (Table 1), which are used to enhance trust of a role about the system. The purpose of these mechanisms is to handle exceptions that might arise from the existing trust relations. By doing so, the resulting system is more robust, and thus more trustworthy for the given role. Table 1 gives a list of basic trust supporting mechanisms induced from the set of patterns presented in [15]. It is basic because they refer to the basic operations concerning commitments such as create, delegate, cancel, etc.

**Table 1.** Some trust enhancing mechanisms

| **Name** | **Trust Encoding** |
|---|---|
| $delegation(x, y, z, p, q)$ | $\mathsf{T}(x, y, \mathsf{threatened}(x, y, p, q), \mathsf{T}(x, z, p, q))$ |
| $compensate(x, y, p, q, r, s)$ | $\mathsf{T}(x, y, \mathsf{violated}(x, y, p, q), \mathsf{T}(x, y, r, s))$ |
| $undo(x, y, p, q, r)$ | $\mathsf{T}(x, y, \mathsf{undo}(q), \mathsf{T}(x, y, r, \mathsf{undo}(p)))$ |
| $renegotiate(x, y, p, q, r, s)$ | $\mathsf{T}(x, y, \mathsf{unreasonable}(p, q), \mathsf{T}(x, y, \mathsf{renegotiated}(r, s), \mathsf{T}(y, x, r, s)))$ |

The propositions violated, threatened, undo, unreasonable, renegotiated, and so on are domain-specific. Referring to our running example, in case ModernLabs cannot deliver the test results to Alice, the Hospital can take care of that, offering yet another way to satisfy Alice's needs. Delegating the responsibility of delivering the results to the Hospital, makes the relationship between Alice and ModernLabs more robust (as long as Alice gets the results, she does not care how they got to her).

Redundancy is another mechanism, but we consider it as a special case of delegation. The system that offers redundancy is better not only for Alice, but for any patient that would decide whether to pay and take a test at ModernLabs or any other laboratory that offers the same services. Generally speaking, for any patient $p$ interacting with some laboratory $l$, if $T_a$ and $T_b$ stand for $\mathsf{T}(\mathsf{p}, \mathsf{l}, \mathsf{paid}, \mathsf{delivered})$ and $\mathsf{T}(\mathsf{p}, \mathsf{l}, \mathsf{threatened}(p, l, paid, \mathsf{delivered}), \mathsf{T}(\mathsf{p}, \mathsf{h}, \mathsf{paid}, \mathsf{delivered}))$ respectively, (h stands for hospital), then $T_b \succ_p T_a$.

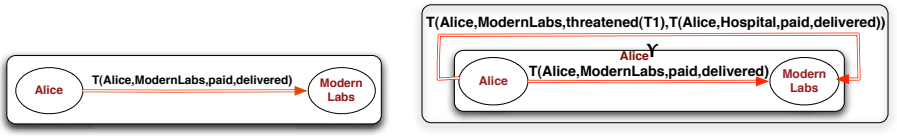
Compensate and undo can be used to capture return-refund scenarios. For example, let $T_g$ and $T_v$ be $\mathsf{T}(\mathsf{p}, \mathsf{l}, \mathsf{paid}, \mathsf{delivered})$ and $\mathsf{T}(\mathsf{p}, \mathsf{l}, \mathsf{violated}(\mathsf{p}, \mathsf{l}, \mathsf{paid}, \mathsf{delivered}), \mathsf{refund} \wedge \mathsf{discountCoupon}))$ respectively. $T_v \succ T_g$, since the trust relationship for delivering the test results upon payment is violated by the laboratory, then a trust relationship that ensures refunds along with a discount coupon for future tests makes the relationship more robust from the patient's point of view. Basically, the laboratory is compensating for the violation by offering a discount coupon and undoing the payment done by the patient. As a result, the patient considers the latter system to be a better choice for him (more trustworthy).

Renegotiate on the other hand offers more alternatives for the role to choose. For example, in case of $T_x$: $\mathsf{T}(\mathsf{p}, \mathsf{l}, \mathsf{paid}, \mathsf{delivered})$ the laboratory might deliver the results of the test in a moment of time that is considered late for the patient. The patient may

renegotiate for the time of the delivery by getting a commitment from the laboratory that will deliver the results on time, $T_y$: T(p, l, paid, deliverOnTime). Again $T_y \succ_p T_x$.

The *supports* relationship shows how trust within a system is improved, from a roles perspective, and it helps possible participants while choosing a system from a range of systems in the domain. By looking at these relationships prospective participants can determine when an entire system (a set of trust relationships) is better than the rest of the systems, within the same domain, from their perspective. Taking this into account we could for instance establish that, if $T_1 \succ_R T_2$, a system $\{T_1, T_2\}$ is better from R's perspective than a system having only $\{T_2\}$.

We provide a graphical representation of the notions composing the system (Fig. 1). Roles are graphically represented as ovals, while trust relationships are shown using double stroke arrows pointing to the trustee, labeled with the actual trust relationship among two given roles (1a). Fig. 1b depicts the usage of the delegation as a trust *supporting* mechanism. The *supports* relation is labeled with the subscript "Alice" to show how Alice's trust is enhanced.




(a) Trust relation (b) Supports Relation

**Fig. 1.** Graphical representation of the system

Apart from the mechanisms shown in Table 1 there could be others that influence the trustworthiness of a system. These mechanisms are domain specific and emerge from the study and analysis of the given domain. We will use a case study to discover new mechanisms to support trust (section 4). Our approach for sociotechnical trust is based on commitments, thus to enhance trust in the system enough commitments are built to make it more trustworthy. The *supports* relation reflects this intuition as well. For instance, let us consider two healthcare systems from our running example, say system $S_1 = \{T_i\}$ and system $S_2 = \{T_i, T_j, T_k\}$.

$S_1$ represents a system in which only one trust relationship holds (6). Hospitals commit on ensuring confidentiality of Alice's personal data (to comply with HIPAA[1]), hence the trust relationship between Alice and the Hospital that if Alice provides her personal data to adhere to Hospital's services, the Hospital will ensure confidentiality of those data. Information included in the medical records of the patients should also be protected and not shared with other parties, unless the patient gives a written permit for that.

$S_2$ ((7),(8), and (9)) represents a system in which the Hospital commits to ensure confidentiality of patient's personal data, it also commits to share patient's personal data only upon their written permission, and the doctor attending the patient commits to protect and ensure privacy of patient information and medical records. The trust

[1] https://www.cms.gov/hipaageninfo/

relationships present in $S_2$ emerge from these commitments. $S_2$ involves the trust relationships $T_j$ and $T_k$, each of which supports the trust relation $T_i$ (that is, $T_j \succ_{Alice} T_i$ and $T_k \succ_{Alice} T_i$). Having two more trust relationships (hence more *commitments*) that support the basic trust relationship, will influence the overall trust of the system.

We could say that $S_2$ is more trustworthy from the patient's (Alice) point of view than $S_1$ because it provides more commitments to ensure confidentiality.

$$T_i = \mathsf{T}(\mathsf{Alice}, \mathsf{Hospital}, \mathsf{personalData}, \mathsf{ensureConfidentiality}) \quad (6)$$

$$T_i = \mathsf{T}(\mathsf{Alice}, \mathsf{Hospital}, \mathsf{personalData}, \mathsf{ensureConfidentiality}) \quad (7)$$

$$T_j = \mathsf{T}(\mathsf{Alice}, \mathsf{Doctor}, T, \mathsf{privacyOfMedicalRecords}) \quad (8)$$

$$T_k = \mathsf{T}(\mathsf{Alice}, \mathsf{Hospital}, \mathsf{permit}, \mathsf{sharePersonalData}) \quad (9)$$

**Definition 1.** *Let $S_1$ and $S_2$ be two systems. We say that $S_2$ is more trustworthy than $S_1$ from $x$'s perspective ($S_2 \gg_x S_1$) if and only if:*

1. $(x, y, p, q) \in S_1$ *implies* $(x, y, p, q) \in S_2$, *and*
2. $\exists (x, z, r, s) \in S_2$ *such that* $\mathsf{T}(x, z, r, s) \succ \mathsf{T}(x, y, p, q)$

The definition captures the intuition behind the fact that a participant ($x$) will trust more the system that will provide more commitments to him (i.e more trust relationships) that *support* the initial interaction he wants to start with that system. This initial interaction is a relation that is provided by both systems to $x$ (1); $x$ considers $S_2$ more trustworthy because it provides other relations that *support* the basic interaction (2).

In the aforementioned example, there are two trust relationships in $S_2$, namely $T_j$ and $T_k$, that support the trust relation in $S_1$, $T_i$. Therefore, the system $S_2$ is said to be more trustworthy from Alice's point of view.

## 3 Kinds of Trust in Sociotechnical Systems

We highlight the difference between sociotechnical trust and two other notions of trust that have been applied to modeling and reasoning about sociotechnical systems. We show with the help of examples that the three are orthogonal notions, and that each plays a role in the smooth functioning of a live STS.

### 3.1 Technical Trust

By actors, we refer only to *social* entities. In practice, this means only humans and organizations (or their software surrogates). For example, consider a healthcare system. Hospitals, patients, doctors, laboratories, insurance companies, and so on are all actors in the system. A hospital may provide the service of appointment scheduling via a Web application; clearly, the application itself is not an actor in the same sense that a hospital is. Similarly, a laboratory may use several devices in providing testing services to patients, for example, a CT scanner; clearly, the CT scanner too is not an actor. Thus, in this paper, we do not talk about *technical trust*—whether the hospital trusts the scheduling Web application (to work well) or whether the laboratory trusts the CT scanner. We broadly identify technical trust with assurance [13], also sometimes referred to as dependability [8].

The contrast with technical trust is an important one. Technical trust necessarily treats the system under consideration as a monolithic entity that can be deployed and evaluated for desirable properties. However, sociotechnical systems are not monolithic. Each actor is necessarily *autonomous* and will implement the functionality it desires independently from other actors. For example, a hospital will implement its information systems independently from a laboratory. Therefore, it is impractical to dwell upon whether a hospital considers a laboratory's information systems and devices dependable; however, it is critically important that the hospital trusts the laboratory for providing accurate test results. In general, technical trust applies at a lower level than social trust. Figure 2 illustrates in an architectural sense the difference between social and technical trust. Incidentally, the traditional model of sociotechnical systems from RE is similar to Figure 2(B).

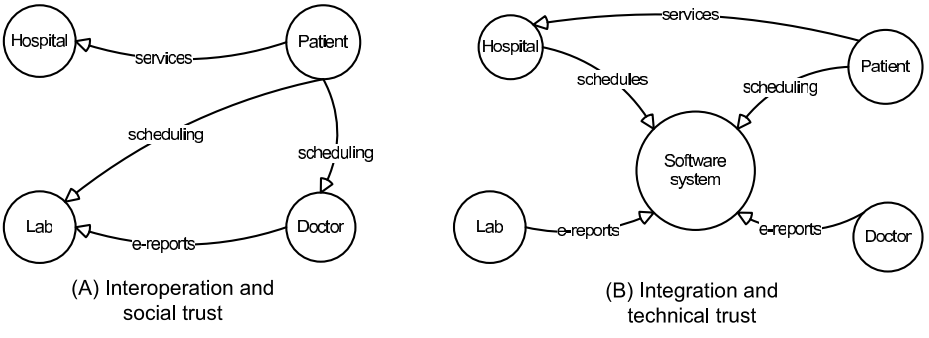



**Fig. 2.** Social versus technical trust

Let $T_T$ denote the technical trust relation. For example, $T_T$(ModernLabs, CTScanner, operatedProperly, deliveredAccurateResults) means that ModernLabs trusts the CTScanner to deliver accurate results if operated properly.

Technical trust is not relevant for us. It is not the scope of our work. Instead, we concentrate on the roles interacting in the system, not the information systems or devices being used. We are interested in the actual actors operating the technical systems, so the technical trust is encapsulated within the social trust. Alice trusts ModernLabs to deliver accurate results, but she does not care how ModernLabs achieves that. It is ModernLabs' responsibility assuring the correctness of the results. As long as the results are delivered being accurate, Alice trusts ModernLabs. It is this kind of relationship that we want to exploit.

### 3.2 Cognitive Trust

Most of the predominant computational approaches to trust have a cognitive bias. In such approaches, each agent has a mental, and therefore necessarily private, model of other agents, based on which it chooses whom to interact with. Some approaches to trust are based on reputation. Although reputation itself is a social concept in that an agent's reputation is public, it also is mostly applied in a cognitive way—as an input to the agents' mental models.

Let $\mathsf{T_C}$ denote the cognitive trust relation. For example, $\mathsf{T_C}(\mathsf{Alice}, \mathsf{ModernLabs}, \mathsf{paid}, \mathsf{delivery})$ means that Alice cognitively trusts ModernLabs. Sociotechnical trust wants to influence in some way the enhancement of cognitive trust, however we cannot enforce the latter. If a system inspires more trust than other systems in the same domain, it might be the case that prospective participants become more willing to participate in that system.

### 3.3 Orthogonality

Our conceptual model helps determine whether a system is trustworthy, architecturally speaking. However we do not address issues dealing with cognitive trust, such as *"within a system, who to interact with?"*, nor issues concerning technical trust. Each of them has a different consequence on the actors' decisions (or influences differently actors' decisions). Let's consider a situation where hospital SantaChiara is known to be better than hospital SanCamillo, and in which Alice trusts more (cognitively) Dr.Giusti rather than Dr.Falconi. Based on sociotechnical trust considerations, we could say objectively that Alice would rather be visited by Dr.Giusti (Dr.G) in hospital SantaChiara (S.Ch) than by Dr.Falconi (Dr.F) in hospital SanCamillo (S.Cam) (10). However, we could not say anything about Alice's trust (where would she get a visit) in case doctor Dr.Giusti were to work at hospital SanCamillo and Dr.Falconi were to work at hospital SantaChiara (11).

$$\mathsf{T}(\mathsf{Alice}, \mathsf{Dr.G}, \mathsf{S.Ch}, \mathsf{goodVisit}) \gg_{Alice} \mathsf{T}(\mathsf{Alice}, \mathsf{Dr.F}, \mathsf{S.Cam}, \mathsf{goodVisit}) \quad (10)$$

$$\mathsf{T}(\mathsf{Alice}, \mathsf{Dr.G}, \mathsf{S.Cam}, \mathsf{goodVisit}) \not\gg_{Alice} \mathsf{T}(\mathsf{Alice}, \mathsf{Dr.F}, \mathsf{S.Ch}, \mathsf{goodVisit}) \quad (11)$$

Similarly, if Alice trusts cognitively ModernLabs to deliver the results once she has paid for the tests, this does not mean that she technically trusts ModernLabs, and vice versa.

## 4 The Food Law Case Study

The European Parliament and the Council adopted Regulation (EC)178/2002[2] on January 2002, to harmonize all Member States food legislation in a general Food Law regulation, whose primary objective is consumer protection throughout Europe. Food Law lays down all requirements on food safety. It imposes requirements on any substance that is intended or expected to be incorporated into a food/feed during its manufacture, preparation or treatment [5]. These requirements should be applied by all food operators in order to comply with the regulation.

**Identifying Roles and Trust Relationships.** Food law exposes the following participants (roles): Member States (MS), Food Safety Authority(FSA), Food Business Operator (FBO), European Consumer (EC). Stakeholders in the food/feed chain are a lot, among which food/feed manufacturers, importers, brokers, farmers, distributors, etc., but we classify all as FBO. To be in compliance with (EC)178/2002 requirements and specifications, the trust relations in Table 2 should hold.

[2] `http://ec.europa.eu/food/food/foodlaw/index_en.htm`

**Table 2.** Trust relations based on roles perspective

| Trust relation | Description |
|---|---|
| European Consumers' Perspective | |
| $\mathsf{T}(\mathsf{C}, \mathsf{FBO}, \mathsf{onMarket}(\mathsf{x}), \mathsf{safe}(\mathsf{x}))$ | Consumers trust food operators that every product they placed in the market is safe |
| $\mathsf{T}(\mathsf{C}, \mathsf{FBO}, \mathsf{onMarket}(\mathsf{x}), \mathsf{labelAdequately}(\mathsf{x}))$ | Consumers trust food operators that every product they placed in the market is labeled adequately |
| $\mathsf{T}(\mathsf{C}, \mathsf{FSA}, \neg\mathsf{safe}(\mathsf{x}), \mathsf{prohibitPlacingOnMarket}(\mathsf{x}))$ | Consumers trust the authorities that if food is found to be unsafe, it will be prohibited to be placed on the market |
| $\mathsf{T}(\mathsf{C}, \mathsf{FBO}, \mathsf{violated}(C, FBO, onMarket(x), safe(x)), \mathsf{T}(\mathsf{C}, \mathsf{FBO}, \mathsf{informOn}(x), \mathsf{withdraw}(x)))$ | Consumers trust food operators that if food is found to be unsafe when it had already been placed on the market, they will be informed and food will be withdrawn from the market |
| $\mathsf{T}(\mathsf{C}, \mathsf{FBO}, \mathsf{hazardsIdentifiedOn}(x), \mathsf{assessRisksRelatedTo}(x))$ | Food operators should perform risk assessment and analysis for any identified possible hazard related to a given food product |
| $\mathsf{T}(\mathsf{C}, \mathsf{FBO}, \mathsf{product}(x, FBO) \wedge \mathsf{ingredients}(x, y_0) \wedge \ldots \wedge \mathsf{ingredients}(x, y_n), \mathsf{record}(y_0, supplier) \wedge \ldots \wedge \mathsf{record}(y_n, supplier) \wedge \mathsf{record}(x, customer))$ | Food operators should keep record of all the suppliers of ingredients of the products they sell and of all the immediate customers |
| Food Business Operator Responsibility | |
| $\forall suppliers\, \mathsf{T}(\mathsf{FSA}, \mathsf{FBO}, \mathsf{monitor}(\mathsf{suppliers}), \mathsf{risksEarlyDetected})$ | Problems with food will be detected early if food operators monitor their suppliers |
| $\mathsf{T}(\mathsf{FSA}, \mathsf{FBO}, \mathsf{onRisk}, \mathsf{notify})$ | Food safety authority expects to receive notification on risk |
| $\mathsf{T}(\mathsf{MS}, \mathsf{FSA}, \mathsf{notify}, \mathsf{inform})$ | Food safety authority should inform member states when notified for some risk |

**Enhancing Consumers' Trust.** We provide a semi-formal graphical representation in a multilayer perspective (Figure 3) starting from the basic situation (container denoted with R). The core requirement is that consumers are provided with safe food, that is, every food that is placed on the market is safe and adequately labeled. Food operators are responsible for ensuring this, therefore we show the trust relation that should hold between consumers and food operators. The Food Safety Law specifies mechanisms that will help improve this situation, thereby enhancing the chances of achieving the primary goal of having safe food on the market. These mechanisms are actually the *supports* relations specific to this domain. The structuring of the trust relationships performed in Figure 3 serves the purpose of applying the *supports* relations to improve the sociotechnical trust at each layer, introducing them at the appropriate layer. Thus, in the second

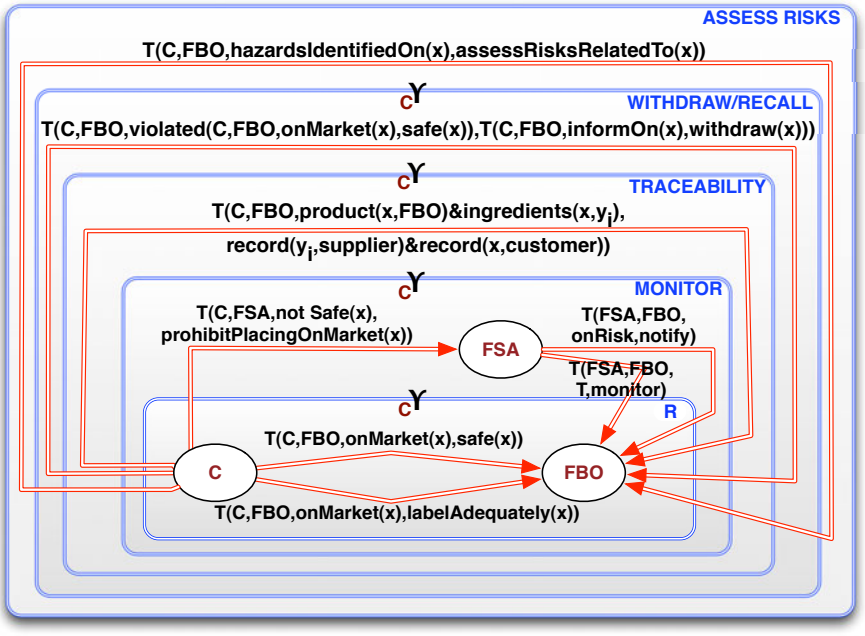



**Fig. 3.** Food Law System Enhancing Consumers' Trust

layer (*Monitoring*) monitoring mechanisms are enabled, such as the ones for monitoring suppliers and food. Monitoring will increase the level of trust consumers will have on the products placed on the market. However, monitoring alone is not enough. That is why *Traceability* mechanisms are applied on top of monitoring (third layer), so that any time a given risk is identified, it is possible to find the node in the food chain in which the breach occurred. Whenever a hazard is encountered, traceability is used to identify the source of such risk, afterward food operators can perform risk assessment to verify the status of the given product. If it is found to violate food safety requirements, customers should be notified and the product needs to be withdrawn or recalled (forth layer: *Withdraw/Recall*). Risk assessment procedures will influence the decision to withdraw or recall a product from the market (fifth layer: *Assess Risks*).

We use the *supports* relationship notation labeled with C, at the border between the subsequent layers, to represent the fact that trust relationships in the above layer *support* trust relationships in the layer below, enhancing at each subsequent layer Consumers' trust about the system.

**Conclusion.** We presented a graphical representation of an STS from the Food Law case study showing trust relations that hold between the different roles (mainly consumers and food operators) along the food chain. We used the case study to identify domain specific *supports* relationships that aim to increase a consumer's trust in this STS. Based on the identified *supports* relationships, we built a structured graphical representation of the trust relations that hold in the system to help possible participants

decide which system they want to play a role at. This representation makes the role perspective clearer by showing the relations they may be involved. Furthermore, this structuring allows to see how trust is increased after each layer by representing the trust supporting mechanisms. All mechanisms represented in terms of *supports* trust relationships serve the purpose of ensuring consumers' safety and inspiring their trust in the system.

**Is the Approach Scalable for Larger Case Studies?** Our approach depends on the clarity of the information regarding the considered domain, as the process of discovering *supports* relationships is domain specific.

## 5 Discussion

In this paper, we captured the intuition behind sociotechnical trust, as referring to the trustworthiness of an STS. We consider an STS to be organized along trust relationships. We want to make make explicit these trust relationships in order to provide prospective participants with information regarding the dependencies they would have upon the STS in case they decide to play a role in. Based on this information, one is able to decide whether, on his perspective, an STS is more trustworthy than another. This requires that we capture *sociotechnical* trust in an architectural sense as a relationship among roles in the system. When specific actors adopt the roles, those relationships are instantiated. We showed the notion of sociotechnical trust to be different from cognitive and technical trust, two prominent concepts in sociotechnical systems research. We analyzed the European food safety regulation on the basis of our approach. The case study reveals that a STS is not a list of trust relationships, but there is structure in the sense that some trust relationships enhance some others. Understanding the structure of a system will prove valuable for those who want to decide their participation in a system. A key feature of our conceptual model is that we take the role perspective in saying whether one system is more trustworthy than another. Thus whereas a healthcare STS may be more trustworthy from the patient point of view; it may be less trustworthy from a laboratory's point of view. Such a perspective follows naturally from the fact that trust is a directed relationship.

The *trustworthiness* of a software is traditionally seen as a measure of assurance that the software is free from vulnerabilities [13]. Notably, such a notion of trustworthiness treats software as a monolithic entity; it is technical trust. Examples of such software include operating systems and browsers. Many trustworthy computing initiatives, including Microsoft's [12], to mitigate security concerns fall under this category. Castelfranchi [4] lists different kinds of trust that often come into play: trust in the environment and infrastructure (technical trust), trust in one's own agent (technical trust because it amounts to trust in the software that an actor uses), and trust in authorities and partners (both cognitive). These are all important kinds of trust; however none of these is sociotechnical. Asnar *et al.* [2] model trust relationships among actors in order to analyze risk; however in their approach, trust is an agent's subjective belief about another, in other words, cognitive. Similar to our approach, Giorgini *et al.* [9] model social trust at the role level. They present examples where any agent adopting some role must trust another adopting some role for something, however, cognitively it may not. Giorgini

*et al.* deem this as a conflict. In our approach, such a situation is not a conflict—social and cognitive trust are orthogonal concepts. Our idea of assuming domain-specific trust enhancing (the *supports*) relationships is not unfounded. Jones *et al.* [11] present a list of trust requirements for e-business. Haley *et al.* [10] introduce the notion of trust assumptions to help discharge concerns about system security. Siena *et al.* [14] care about compliance of actors with regulations, we care about understanding the structure of the regulations themselves, and how the regulation may have potentially come about in the first place.

**Future Directions.** In order to build architecturally better systems in terms of trust, we want to explore a formal semantics for the *supports* relationship. However, given the diversity of domains and trust relationships, this is an especially challenging task. Our initial approach for defining supports was via formulating trust enhancing mechanisms. We will further exploit commitments to identify more such mechanisms.

**Acknowledgments.** This work has been partially supported by the EU-FP7-IST-IP-ANIKETOS and EU-FP7-IST-NOE-NESSOS project. Amit K. Chopra was supported by a Marie Curie Trentino cofund award.

## References

1. Credit card accountability responsibility and disclosure act of (2009), `http://www.govtrack.us/congress/bill.xpd?bill=h111-627`
2. Asnar, Y., Giorgini, P., Massacci, F., Zannone, N.: From trust to dependability through risk analysis. In: Proceedings of the Second International Conference on Availability, Reliability and Security, pp. 19–26 (2007)
3. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems 8(3), 203–236 (2004)
4. Castelfranchi, C., Tan, Y.-H.: The role of trust and deception in virtual societies. International Journal of Electronic Commerce 6(3), 55–70 (2002)
5. Standing Committee on The Food Chain and Animal Health. Food law implementation guidelines. World Wide Web electronic publication, January 2010, Lastchecked (December 2010)
6. Chopra, A.K., Dalpiaz, F., Giorgini, P., Mylopoulos, J.: Modeling and reasoning about service-oriented applications via goals and commitments. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 113–128. Springer, Heidelberg (2010)
7. Chopra, A.K., Dalpiaz, F., Giorgini, P., Mylopoulos, J.: Modeling and Reasoning about Service-Oriented Applications via Goals and Commitments. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 113–128. Springer, Heidelberg (2010)
8. Dewsbury, G., Sommerville, I., Clarke, K., Rouncefield, M.: A Dependability Model for Domestic Systems. In: Anderson, S., Felici, M., Littlewood, B. (eds.) SAFECOMP 2003. LNCS, vol. 2788, pp. 103–115. Springer, Heidelberg (2003)
9. Giorgini, P., Massacci, F., Mylopoulos, J., Zannone, N.: Requirements engineering for trust management: Model, methodology, and reasoning. International Journal of Information Security 5, 257–274 (2006)
10. Haley, C., Laney, R., Moffett, J., Nuseibeh, B.: Using trust assumptions with security requirements. Requirements Engineering 11, 138–151 (2006)

11. Jones, S., Wilikens, M., Morris, P., Masera, M.: Trust requirements in e-business. Communications of the ACM 43(12), 81–87 (2000)
12. Lipner, S.: The trustworthy computing security development lifecycle. In: Proceedings of the 20th Annual Computer Security Applications Conference, pp. 2–13 (December 2004)
13. Mead, N.R., Jarzombek, J.: Advancing software assurance with public-private collaboration. IEEE Computer 43(9), 21–30 (2010)
14. Siena, A., Armellin, G., Mameli, G., Mylopoulos, J., Perini, A., Susi, A.: Establishing regulatory compliance for information system requirements: An experience report from the health care domain. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) ER 2010. LNCS, vol. 6412, pp. 90–103. Springer, Heidelberg (2010)
15. Singh, M.P., Chopra, A.K., Desai, N.: Commitment-based service-oriented architecture. Computer 42(11), 72–79 (2009)
16. Singh, M.P.: An ontology for commitments in multiagent systems: Toward a unification of normative concepts. Artificial Intelligence and Law 7, 97–113 (1999)
17. Singh, M.P.: Trust as a basis for social computing (2010), `http://www.csc.ncsu.edu/faculty/mpsingh/papers/positions/Trust-formal-architecture-talk.pdf`
18. Strens, R., Dobson, J.: How responsibility modelling leads to security requirements. In: Proceedings of the New Security Paradigms Workshop, pp. 143–149 (1993)
19. van Lamsweerde, A.: From system goals to software architecture. In: Bernardo, M., Inverardi, P. (eds.) SFM 2003. LNCS, vol. 2804, pp. 25–43. Springer, Heidelberg (2003)