A Disclosure Framework for Service Accountability in SOA

Joe Zou[1,2], Christopher J. Pavlovski[1], Yan Wang[2]
[1]IBM Australia, [2]Macquarie University
Sydney, NSW, Australia
joezou@au1.ibm.com, chris_pav@au1.ibm.com, yanwang@ics.mq.edu.au

Abstract

Accountability has become a key concern for management in recent years since the collapse of several large corporations in energy, communications and financial consulting industries. Traditionally, accountability has not been a significant concern in the IT literature. Recently, some protocols, models and frameworks that address certain aspects of accountability have emerged. However, an architecture framework that addresses the overall accountability requirements is nonetheless missing in a Service-Oriented computing environment. In this paper, we review the accountability literature in an IT context and argue that disclosure, as a crucial requirement of accountability, has been largely ignored in the current literature. Based on this proposition, we propose a new architectural framework to address the important disclosure requirement that extends and integrates the existing architecture models. The proposed framework can be used to build SOA solutions that support the key accountability requirements.

Keywords: Accountability, Disclosure, SOA, Web Services, Trust, Reputation

1. Introduction

Traditionally, accountability is a term that is mainly used in management and law literature, but has received considerably less attention in an IT context. In the commercial sector, it is a common sense approach that vendors or service providers are expected to be held accountable to consumers for the products or services that they supply. This implies that the vendors or service providers need to make honest disclosure regarding their products or services. Once the consumer decides to purchase a product or service, the vendor, or service provider, is obligated to deliver these in accordance with what has previously been disclosed. Moreover, they are also liable for any defects or malfunctions inherent in their products or services.

Disclosure is a basic requirement of accountability. From buying simple products or services to investing in complex financial instruments, consumers rely upon the vendors' disclosures to make informed decisions. In recent years, the push for accountability has become an important item on the business agenda. Public opinion has also called for a higher standard of accountability in corporate governance and business practice since the collapse of several large institutions. For instance, in the Enron case, Chan-Fishel points out that inadequate or misleading corporate disclosure is a significant contributor to widespread misery and misfortune, and that new reporting measures can promote corporate accountability [1]. Law makers and governments have responded with legislation and regulations that set mandatory requirements on disclosure in the corporate world. This is evident in Sarbanes–Oxley Act, which holds management accountable for the timely disclosure of information that has a material impact on the financial performance of a business [2].

Service Oriented Architecture may be considered a relatively new IT paradigm. Enabling accountability in IT systems is becoming increasingly important in today's computing environments; especially at a time when the industry is moving towards adopting the business centric SOA. In this paper, we build upon existing theories in accountability and apply these to the SOA environments. We formally define disclosure and illustrate how this is crucial to ensuring accountability in an SOA environment. We then propose an architectural framework that further illustrates how such disclosure may be developed within an SOA solution.

2. Background and Related Work

In [3] an accountability meta-model is given. A formal definition of service accountability is also defined as follows [3]:

Accountability in services refers to the obligation that several persons, groups, or organizations assume for the execution and fulfillment of a service. This obligation includes:

- *answering, providing an explanation or justification, for the execution of that authority and/or fulfillment of that responsibility;*
- *full disclosure on the results of that execution and/or fulfillment;*
- *undeniable liability for those results (non-repudiation); and*
- *obtaining trusted agreement of accountability from all entities involved in the service who in turn are bound to the obligations set out above.*

In [4], Yumerefendi and Chase promote accountability as a central design goal in network services by introducing a round processing framework. In this framework, the service processes a sequence of requests in rounds. All accepted requests bear a timestamp matching the round. At the end of the round the service publishes a signed, timestamped, non-repudiable digest of its internal state to external observers. This framework provides some preliminary disclosure capability, mainly on the disclosure of service execution state and outcome.

In [5], Tseng *et al* position accountability as the ownership of the responsibility to meet requirements in an end-to-end business process. The authors propose Accountability Centered Approach (ACA) for business process engineering. The ACA approach suggests iterative decomposition of accountability to appropriate levels and mapping of sub-accountabilities into activities. Zhang *et al* propose a 3-D approach (Detect, Diagnose, and Defuse) in their accountability model to discover and eliminate the root cause of problems when violations of service level agreement occur in business processes [6]. The approach adopts Bayesian Network reasoning for root cause analysis and service reputation model to address problematic web services.

In [7], a review is provided on some accountability protocols and the architecture frameworks in the current literature. Aiming at defining a distributed accountability model for a peer-to-peer network, they compare and contrast several accountability models' strengths and weaknesses. At the end, they combine Zhang's 3D model and the peerMint model [8] to support accountability in a peer-to-peer trading environment, addressing the requirements of non-repudiation of sharing resources as well as root cause traceability. While in [9], Lin, Panahi and Zhang have created a prototype of Intelligent Accountability Middleware Architecture (LLAMA), which provides a dynamic and efficient service infrastructure to support service monitoring, root cause analysis and reconfiguration of service process after problem diagnosis. All these works do not treat disclosure in their reviews or models.

In summary, the existing accountability protocols and architecture frameworks mainly focus on non-repudiation, with some emphasis on quality of service (QoS), monitoring and root cause traceability. However, disclosure, or in particular, disclosure prior to service consumption, is largely ignored in the current literature.

3. Disclosure in Accountability

In moving towards a detailed discussion on disclosure, we first propose a formal definition for this term, since it is a fundamental property to ensuring accountability.

The term disclosure is often referenced in definitions such as the freedom of information act, informed consent, and trade secrets, where a legal obligation exists in providing certain information [10]. In the context of computer security, full disclosure has traditionally been used to refer to the release of all known security software defects [11]. While this is an important area, the terms of reference are limited to treating security and software defects. Presently, no formal definition has been provided in the context of accountability and ICT.

A legal definition for disclosure refers to the legal obligation of parties to inform each other to the existence of any relevant documents [12]. While in a business transaction, disclosure is the need to tell the "whole truth" about any matter which the other party should know in deciding to buy or contract [29].

The accountability definition provided in [3] states that 'full disclosure on the results of the execution and/or fulfillment' of a service be given. Recalling that accountability refers to the obligation that a service provider has in providing a service to a person or group of people, the disclosure aspect of this relates to any information that relates to this service obligation. There are several perspectives to consider when information is released under a disclosure agreement. This is from the service provider, consumer, or (regulatory) third party perspective. A service provider may generally be motivated to provide a base set of information that it interprets as sufficient when disclosing information. This elementary disclosure will be often guided by a regulatory constraint or customer agreement. The customer however, may seek further information to satisfy their needs in disclosure resulting in a dispute of disclosure status. A third perspective is that of a (trusted) notary or regulatory body to settle any possible dispute that arises. In this context, further information may be requested which may be considered sensitive, such as trade secrets, of the service providing organization.

During an act of disclosure, all parties involved have been satisfied to the level of disclosure provided. However, this may be achieved where certain details are still withheld, being not considered relevant by one or more parties. While the information may have been considered irrelevant, this touches upon a related topic of disclosure termed *transparency*.

It may be considered that in order to behave in a transparent manner, the involved parties provide information in a voluntary manner, whereas normal disclosure would occur under some legal obligation that is bounded by an agreement or regulatory constraint. Hence a full transparent disclosure includes the making available of all additional information, or making known the existence of such information, which may or may not be considered relevant to all parties for access as required. We now formally define disclosure in accountability (see Table 1).

Disclosure is the obligated release of all information assets, including the existence of all information assets, relating to an act or actors that are accountable in providing a service. This may include, but not limited to, identities involves, roles, responsibility, service status, and results of service execution.

Table 1 Disclosure

4. Approaches to Disclosure

In this section we now explain the ways in which disclosure may be addressed in IT systems and propose several elementary techniques that may be adopted in an SOA architecture.

4.1 Levels of Disclosure

We suggest that there are three fundamental levels of disclosure that may be adopted. The *first level of disclosure* is a manual-based, free text format disclosure, which is currently adopted by the industry. This form of disclosure is normally hosted on the service provider's website, and can include a variety of content, such as "about us", "contact us", service terms and conditions and privacy policy. The Service Level Agreement (SLA) may be considered a form of disclosure on both the functional and the QoS properties. While this approach may be sufficient for some business applications, there are several problems on this level of disclosure. For instance, this approach does not cater for machine interpretation. That is, it does not support dynamic service consumption in a Service-Oriented Architecture. A further issue is that the non-standard format of the disclosure creates difficulty for Trust and Reputation engines to source input from the service provider. In addition, the content for such disclosure is aimed at people with a legal background and not for general consumers.

The *second level of disclosure* is the emerging syntax-based XML format disclosure. There are several Web Services specifications that can be used for this purpose. WS-Policy provides a framework for general syntax for expressing capabilities and policies. WS-Agreement can be used to disclose obligations. WSLA may also be used to express the Service Level Agreement. While the syntactic level of disclosure holds a great promise in terms of supporting dynamic service consumption, it is nonetheless immature and yet to be widely accepted.

The *third level of disclosure* is the semantics-based disclosure. This level is more sophisticated, relying upon a commonly agreed ontology established amongst service providers and service consumers. Currently OWL is the most widely used language to create such an ontology. OWL-S allows a semantic description of a web service [13]. However, an ontology can become quite complex, especially when this delves into domain specific concepts. This approach is a more immature area when compared to the syntax-based disclosure.

4.2 Tool for SOA

There are perhaps several approaches to implementing disclosure in an SOA architecture. The first approach is a registry based disclosure. This approach treats disclosure information as part of the service meta-data, and publishes the meta-data to a public registry to facilitate service discovery and matching. This approach is similar to publishing and discovery in SOA. Two key registry standards are currently available, UDDI and ebXML. Currently, UDDI is mainly used for the publishing of service description. Applied in this way, the data model makes it difficult to support the disclosure of roles, responsibility, execution state and outcome. On the other hand, ebXML allows business to first disclose its business profile information in an ebXML registry and then form an agreement based on the disclosed profiles of trading partners. The profile information includes roles, capabilities, constraints and implementation details, while the agreement may take the form of a Collaboration Protocol Agreement. Hence, the ebXML form provides stronger disclosure capability than UDDI. However, ebXML is normally applied to business-to-business (B2B) scenarios and may be considered too heavyweight for most business-to-consumer (B2C) scenarios.

The second disclosure approach is based on a Publish and Subscribe architectural pattern. A service provider can publish the disclosure information, which can be subscribed by service consumers. Publish-Subscribe is a mature technology and it has been implemented in numerous commercial-off-the-shell products. In a web services environment, WS-Notification is a family of related white papers and specifications that define a standard approach to develop publish-subscribe systems [33].

The third disclosure approach is to treat disclosure as a policy requirement to be enforced by a policy enforcer. In [14], the authors provide a survey on five policy frameworks: IETF, Ponder, KAoS, Rei and WS-Policy. They conclude that KAoS and Rei are more suitable for SOA systems due to their support for dynamic policy update. However, KAoS and Rei are ontology-based policy frameworks, which may not be easily applied in the normal syntactic web services environment. On the other hand, the authors also assert that WS-Policy is a low level policy language that is not suitable for managing an overall SOA system.

Currently neither UDDI nor ebXML has been widely adopted in real SOA implementations due to their complexity and limitations. Hence, using the registry-based approach for disclosure will be a considerable task to achieve this. On the other hand, the policy-based technology is still emerging and yet to be mature. Thus it may not be a good choice for building disclosure mechanisms at present.

5. A Disclosure Architecture Framework

Building upon the tools in the previous section we now describe an architectural framework for implementing disclosure.

5.1 Framework Overview

Based on the analysis above, we propose an architectural framework for disclosure based on the publish-subscribe pattern. The proposed framework enables disclosure of accountability information in a Service-Oriented Architecture. As illustrated in Figure 1, the traditional service provider, service consumer and service registry roles, are extended with two addition roles to provide the disclosure and reputation rating capabilities. These are the *disclosure broker* and a *trust & reputation engine* respectively. We now describe these two roles further.

Disclosure Broker

Disclosure broker implements WS-Notification family of specifications and acts as a third party broker that is responsible for dissemination of accountability information to the involved parties. This entity utilizes the Atom 1.0 standard as the payload format for disclosure information. This approach has been verified in [15], where Wu and Zhang present an ATOMServe architecture that utilises Atom feed as the mechanism for publishing and discovery of service meta-data. The Atom 1.0 standard supports a flexible format of content, which in theory can cater for all three levels of disclosure previously discussed; that is, manual, syntax and semantic based disclosure.

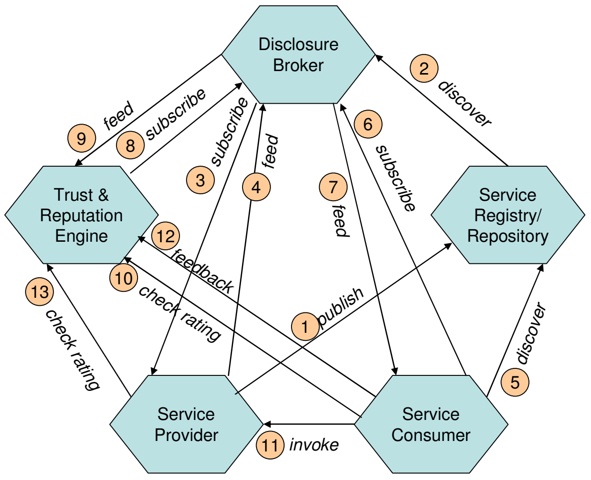

Figure 1 Disclosure Framework

Trust & Reputation Engine

The Trust & Reputation engine stores trust history information, collects feedback on service usage from the service consumers, calculates reputation rating for each service and responds to the check rating requests from the service consumers or the service providers.

With these two new entities in mind, we now explain in more detail the interactions illustrated in Figure 1.

1) The Service Provider publishes its service descriptions to a Service Registry.
2) The Disclosure Broker discovers the newly published service in the service registry.
3) The Disclosure Broker subscribes to service disclosure information from the service provider.
4) The Service Provider publishes the service disclosure information as an Atom feed. The Disclosure Broker receives the feed, parses it, and stores it into its database.
5) The Service Consumer discovers the service description from the Service Registry.
6) The Service Consumer subscribes to information about the service from the Disclosure Broker.

7) The Disclosure Broker publishes the information as an Atom feed to the Service Consumer. The Service Consumer parses the feed, decides whether or not the service matches its requirements.
8) The Trust & Reputation engine subscribes to service disclosure information from the Disclosure Broker.
9) The Disclosure Broker publishes the service disclosure information as an Atom feed to the Trust & Reputation Engine.
10) The Service Consumer queries the Trust & Reputation engine for a reputation rating of the service.
11) The Service Consumer finally decides to use the service, invoking the service from the Service Provider.
12) After using the service, the Trust & Reputation engine may collect feedback from the Service Consumer. The engine recalculates the reputation rating based on the feedback.
13) The Service Provider may check rating of its service from the Trust & Reputation engine.

5.2 Component Architecture Model

The entities depicted in Figure 1 are now described. The component architectural model is shown at Figure 2, which illustrates the sub-components of each key disclosure entity.

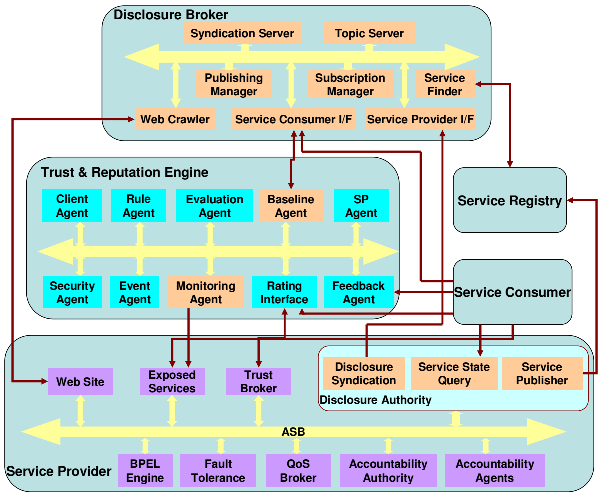

Figure 2 Accountability Component Model

In this model, we design a Disclosure Broker that acts as an independent third party to facilitate disclosure of service providers' accountability information. We also extend the LLAMA ESB architecture in [9] with a Disclosure Authority component. This component acts as the service providers' disclosure control point that is responsible for disclosing accountability information, publishing service description to a service registry and providing service execution state inquiry services. For the Trust & Reputation engine, we extend Wang and Lin's Trust Management Framework [16], adding a Baseline Agent and a Monitoring Agent. We now describe further the role of each of these new components.

5.2.1. Disclosure Broker

Web Crawler. The Web Crawler is used to search the service provider's web site for accountability information. Normally it is only used if the service provider does not have a Disclosure Authority. The Crawler searches the web site and attempts to obtain accountability information from pages such as *About Us, Terms & Conditions*, and *privacy policy*. It then categorizes the information based on a predefined topic scheme and finally stores this within a database.

Service Provider I/F. A Service Provider Interface is used to interact with the service provider's Disclosure Authority. Its responsibility is to subscribe the service provider's accountability information based on WS-Notification, receive the information feed, parse the feed and finally store this within the database.

Service Consumer I/F. The Service Consumer Interface allows consumers to subscribe to the service accountability information feed using WS-Topics' topic expression dialect.

Service Finder. This component is responsible for locating new services in the service registry. Based on the service information returned from the service registry, the Disclosure Broker can contact the service provider and subscribe the accountability information.

Subscription Manager. The Subscription Manager maintains the service consumers' subscription lists.

Publishing Manager. The Publishing Manager instructs the syndication server to build feeds based on the subscription topics. It will notify the service consumer when a feed is created or refreshed for use.

Syndication Server. The Syndication Server receives instructions from the Publishing Manager, interacts with the Topic server to construct Atom feeds. Each Web service is associated with a feed, while each entry of the feed corresponds to a piece of disclosure information. For example, the service provider's role information can be an entry in the feed, while the SLA agreement may be a further entry within the feed.

Topic Server. Finally, the Topic server organises topics into a hierarchical scheme and maintains the associated metadata. Topics are used to organise and categorise items of interest for subscription. They are the links between the subscription and the notification.

5.2.2. Trust and Reputation Engine

The Trust and Reputation engine is used to store historical trust information and evaluate current service reputation. We extend Wang *et al* Trust Management Framework [16] by adding two components to support real-time measurement of QoS against disclosed SLAs.

Baseline Agent. The Baseline Agent subscribes the disclosure feed through the Disclosure Broker's Consumer interface. This component uses the disclosed SLA as a baseline for QoS measurement.

Monitoring Agent. The Monitoring Agent acts as a service consumer and uses the service of the Service Provider for monitoring purposes. The QoS of the service is measured against the predefined service level agreement. The degree of conformance forms part of the evaluation criteria for reputation rating calculation.

5.2.3. Service Provider

In a service provider's internal environment, we extend the LLAMA accountability service bus [9] to support accountability. We augment LLAMA with a Disclosure Authority to strengthen accountability by supporting information disclosure. The Disclosure Authority further consists of three components.

Service Publisher. This component is responsible for publishing service descriptions to a public registry such as UDDI.

Service State Inquiry. This component enables service consumers to check the service status during or after the consumption of the service. For long running services, information such as the current step of the service is supplied. This also provides non-repudiable evidence of the service outcome, such as a signed receipt of transaction.

Disclosure Syndication. Disclosure Syndication creates an Atom feed per service. The entries in the feed correspond to the different aspects of disclosure information.

On a high-level, the architecture framework addresses the disclosure requirements in addition to trust, monitoring, root cause traceability and non-repudiation.

6. Conclusions and Further Work

This paper raises the issue of disclosure in the context of service accountability. We examine the content of disclosure and the disclosure levels, outlining tools that may be used for building disclosure frameworks. Finally, a new framework is proposed that integrates the existing accountability models. The proposed framework provides the disclosure capability in an accountable SOA environment.

Future work is needed on the definition of an accountability assertion language to support disclosure. Semantics-based disclosure is also another area that requires further investigation.

7. References

[1] M. Chan-Fishel, After Enron: How Accounting and SEC Reform Can Promote Corporate Accountability While Restoring Public Confidence, Journal of Environmental Law Reporter (ELR), Environmental Law Institute, Washington DC, Vol. 32(8), Aug. 2002, pp.10965-10979.

[2] SEC-Rules, Sarbanes-Oxley, Financial and Accounting Disclosure Information, Section 409, 2002. Available at: http://www.sarbanes-oxley.com.

[3] J. Zou and C.J. Pavlovski, Towards Accountable Enterprise Mashup Services, IEEE International Conference on e-Business Engineering (ICEBE '07), Hong Kong, 2007, pp.205-212.

[4] A.R. Yumerefendi and J.S. Chase, Trust but verify: accountability for network services, Proceedings of the 11th workshop on ACM SIGOPS European workshop, ACM Press, 2004.

[5] M.M. Tseng, J.S. Chuan, and Q.H. Ma, Accountability Centered Approach to business process reengineering, Proceedings of the 31st Hawaii International Conference on System Sciences, Vol. 4, January 1998, pp.345 – 354.

[6] Y. Zhang, K.J. Lin, and T. Yu, Accountability in Service-Oriented Architecture: Computing with Reasoning and Reputation, Proceedings of IEEE International Conference on e-Business Engineering, 2006, pp.123-131.

[7] P. Malone and B. Jennings, Distributed Accountability, Identity, and Trust, Deliverable D4.2: Distributed Accountability Model for an Autopoietic Peer-to-Peer Network, Open Philosophies for Associative Autopoietic Digital Ecosystems, (OPPALS), 2007.

[8] D. Hausheer and B. Stiller, PeerMint: Decentralized and Secure Accounting for Peer-to-Peer Applications, NETWORKING, Springer Berlin, May 2005, pp.40-52.

[9] K. J. Lin, M. Panahi, and Y. Zhang, 2007, The Design of an Intelligent Accountability Architecture, Proc. of IEEE International Conference on e-Business Engineering, 2007, pp.157-164.

[10] West's Encyclopedia of American Law, 2nd Edition, 13 Vols., J. Lehman, and S. Phelps, (Eds), Thomson Gale, 2004.

[11] B. Schneier, Crypto-Gram Newsletter: Full Disclosure, Counterpane Internet Security, November 2001. Available at http://www.schneier.com/crypto-gram-0111.html.

[12] Disclosure, Wikipedia. Available at http://en.wikipedia.org/wiki/Disclosure.

[29] The Law Dictionary, Full Disclosure, Law.Com, Available http://dictionary.law.com/definition2.asp.

[13] D. Martin, *et al*, OWL-S: Semantic Markup for Web Services, W3C Member Submission, 2004.

[33] S. Graham *et al*, Publish-Subscribe Notification for Web services, Oasis, 2004, Available at: http://www.ibm.com/developerworks/library/ws-pubsub/WS-PubSub.pdf.

[14] T. Phan, J. Han, J-G. Schneider, T. Ebringer, and T. Rogers, A survey of policy-based management approaches for Service Oriented Systems, 19th Australian Conference on Software Engineering, Perth Australia, 2008, pp.392-401.

[15] C. Wu and E. Chang, AtomServ Architecture:Towards Internet-scaled Service Publish, Subscription, and Discovery, IEEE International Conference on e-Business Engineering, 2006, pp.571-578.

[16] Y. Wang, K.J. Lin, D. S. Wong and V. Varadharanjan, The Design of A Rule-based and Event-driven Trust Management Framework, IEEE International Conference on e-Business Engineering, 2007, pp.97-104.