OPAALS PROJECT

Contract n° IST-034824

WP3: Autopoietic P2P Networks

Del3.8 – Final Accountability Model

Project funded by the European Community under the "Information Society Technology" Programme

Contract Number: IST-034824

Project Acronym: OPAALS

Deliverable N°: 3.8

Due date: M29

Delivery Date: M33

Short Description: This document reports the status of the distributed accountability model at M29. The model has been refined since D4.2 and the original protocol has been developed into two protocols for both public and private accountability. The document also demonstrates how the work integrates with the Trust and Transaction models from WP4. Finally an implementation outline is provided.

Author: Paul Malone, WIT

Partners contributed: WIT

Made available to: Public

Versioning		
Version	**Date**	**Name, organization**
0.1	01/10/08	Revised Model, WIT
0.2	04/12/08	Introduction of private accountability protocol, WIT
0.3	31/01/09	Development of public and private accountability protocols
0.4	12/02/09	Integration Points and Data Model
0.5	20/03/09	Internal Review
1	27/03/09	Final Version

Quality check

Internal Reviewers: Pedro Bueso (UniZar), Paul Krause (Surrey)

Dependencies:

Achievements*	The model has been refined since D4.2 and the original protocol has been developed into two protocols for both public and private accountability. The model has been integrated with the Trust and Transaction models from WP3. Finally an implementation outline is provided. No Implementation work has been performed to date. This is due to a lack of resources becoming available to the partner internally over the timeframe of the reporting period. The resource has become available recently and the implementation work has begun.
Work Packages	The work contributes to the provision of integrating accountability in the developing platform via WP5. This is achieved by the specification of the model as well as protocols to execute accountability in a fully distributed digital ecosystem. Data collectors for this model need to be developed within WP5. The work also provides a discussion point for WP 12 where a socio-economic framework for Identity, Trust and Accountability is being addressed as well as a task on governance.
Partners	IPTI (platform development), TechIdeas (integration), Surrey (p2p platform),
Domains	Accountability, Distributed Computing, Cryptography, Security
Targets	Other Researchers, System Implementers, SMEs, Public Administrators, Social Scientists
Publications*	The initial mode was published in DEST 2008. The revised model and the protocols are yet to be published. Malone, P. and Jennings, B., 2008, Distributed Accountability Model for Digital Ecosystems, *2nd IEEE International Conference on Digital Ecosystems and Technologies*, Phitsanulok, Thailand, February 2008.
PhD Students*	N/A
Outstanding features*	The work provides an incremental change in the state of the art by providing a model and protocol to enable a service composition capable accountability framework to operate in a distributed and private manner.
Disciplinary domains of authors*	Paul Malone, WIT, Computer Science

Table of Contents

1 Introduction

The availability of strong Accountability mechanisms in a system deploying commercial or sensitive data or services helps to deliver trust in that system. Bullock [1] provides a model linking governance, accountability and legitimacy (trust in the system). This is a useful reference point in showing the relationship between these subjects and how changes in governance influences trust in the system via accountability. If we are to provide a distributed platform for commercial or sensitive applications, then accountability is required to ensure trustworthiness in the platform and thus encourage take-up and usage.

A useful and regularly cited definition of accountability is given by Schedler [2] as *"A is accountable to B when A is obliged to inform B about A's (past or future) actions and decisions, to justify them, and to suffer punishment in the case of eventual misconduct"*. Within our context *A* and *B* could represent individual players in the ecosystem, or *A* could represent a participant and *B* represent the ecosystem or community, or indeed both could represent ecosystems and their accountability to each other.

The goal of this document is to update the Accountability Model initially proposed in deliverable D4.2 [3]. The main element lacking in the original was the ability to make data private while also leveraging on the distributed and decentralised nature of the ecosystem in the verification of parties' reports on how transactions progress.

1.1 Privacy versus Accountability

Accountability and privacy bring a tension of interests together. Public accountability requires a transparency that is not always available when private accounting is in place. There is much discussion across many disciplines on this dilemma. See [4] [5] [6] for some discussion related to information technology.

While the introduction of a privacy mechanism to ensure no unauthorised access to the accounted data brings clear benefits in terms of protection of business interests it also brings about the case where entities reports of experience actions are not verifiable by the community as a whole. This in turn leads to a case where Experience Reports generated to evolve trust in transacting entities cannot be verified as accurate by the mediators or the account holders.

With this in mind, it is the part of the purpose of protocol development in this task to cater for a model capable of performing in a publicly accountable manner in addition to a private accountability one.

1.2 Accountability Scenarios

There are 3 separate Accountability scenarios which need to be supported by our model.

1. **No Accountability:** This is the simplest case where a service or data consumption is open and there is no need for accountability (e.g. accessing web pages on a public website). This is the absence of accountability and is a trivial case but the need for this prescribes the case that accountability is optional and is in operation on a per-service case.
2. **Public Accountability:** This is the case where service or data consumption is to be accounted for and the usage data describing the course of the transaction is open and available to anyone. This is useful in the usage of accountability data in verifying trust updates when incorporated with the OPAALS Trust Model.
3. **Private Accountability:** This is the case where the service or data consumption is to be accounted for but the usage data describing the course of the transaction is only available to the participating parties. This data is less useful in the verification of trust updates as the semantics of what the data contains cannot be substantiated by third parties. However the provision of this type of mechanism is necessary for the sensitivity of the usage data.

The rest of this document is arranged as follows:

Section 2 recalls the initial model as described in [3], defines the roles of the actors and provides protocols for the model. Section 3 shows how the Accountability Model relates and integrates with the Trust and Transaction models developed in WP3. Section 4 describes the initial implementation design incorporating data model design and metering usage data using the JXTA Metering and Monitoring Project. Finally there is a concluding summary and next steps section.

2 Distributed Accountability Model

In phase 1 of OPAALS an initial Accountability Model was developed in workpackage 4 and was previously described in deliverable D4.2 [3].

The model is influenced by the *PeerMint* model published by Hausheer and and Stiller [7] combined with an *Accounting Authority* to cater for composed services similar to that of Zhang et al [8]. This model is shown below in Figure 1. Although the diagram shows a simple transaction between two services, the introduction of the *Accounting Authority* provides the required functionality necessary for composed services. The role of the *Accounting Authority* which is to ensure accountability for composed services can be provided by the super-set of all mediation peers involved in the service composition.

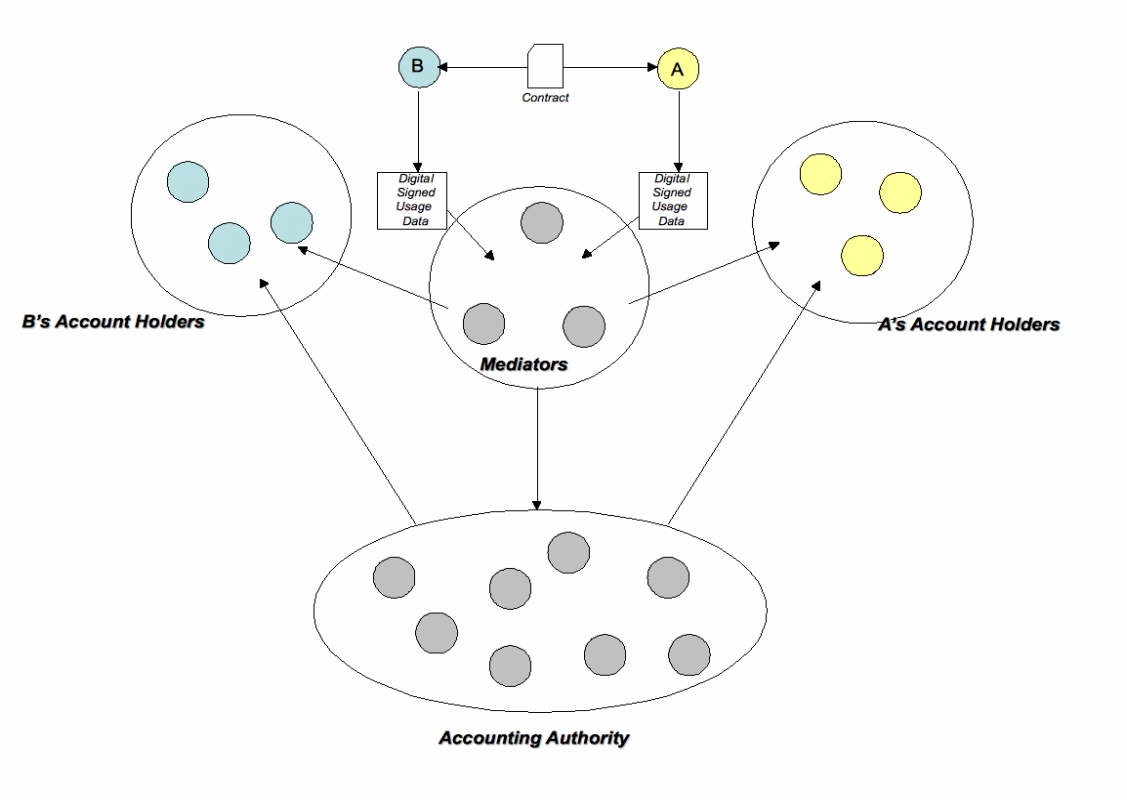

Figure 1: Distributed Accountability Model

The model involves two peers (*A* and *B*) interacting in a distributed services/content access environment.

Prior to interaction between the peers, each peer is assigned a set of trusted peers as account holders. This assignment is based on historical availability of peers and this set of peers is updated over time as the system evolves. Upon joining the network, each peer is assigned a unique peer ID calculated from that peer's public key using a secure hashing method. The Mediators are selected

using a hash on the combination of the interacting peers IDs in combination with a time stamp.

2.1 Reducing the ability for group infiltration.

In order to reduce the possibility of cheating, it is important that sets of Account Holders or Mediators cannot be infiltrated by either participating party or by a third party. In the case that *A*'s Account Holders identities can be predetermined it might be possible for an entity to infiltrate this set of nodes and inject fraudulent data into the nodes. A non-deterministic periodic issue of a randomly created nonce by each node to perform a recalculation of its Account Holders would make this very difficult to achieve and can be considered a suitable mechanism against infiltration. Mediator selection is done on a per-session basis. When a session begins Mediators are assigned to the task of checking accounted data for consistency. At the end of the session, the set of Mediators is released. The selection of Mediators is performed by performing a hash function on the combination of both peers' identities combined with a processID, serviceID and a timestamp to ensure that Mediator selection cannot easily be pre-determined by potentially rogue peers.

2.2 Securing the Accounted Data

Securing the data relies on a PKI facility being in place. Each node has a public/private key pair which can be used to create shared keys (e.g. Diffie-Hellman [9]). When *B* wishes to communicate with *A*, it sends its public key together with a time-stamped process identifier. If *A* wishes to continue the communication, it returns its public key together with the time-stamped process identifier. Now each party can use their private keys in combination with the other's public keys to create a shared secret for the session. This shared secret is kept private to each party and is not transmitted. This shared secret can be used for two purposes, securing a channel of communication and encrypting accounted data.

2.3 Actors and Protocols

The model comprises Actors and Protocols for transmission of messages. There are four Actors in the model. Their roles are explained here together with protocols for transmission of accountable data in a public and private message exchange.

2.3.1 Actors

Peers

The Peers are responsible for the creation of the evidence and the delivery to the Mediators. Peers

can also act as Mediators, Account Holders and members of Accounting Authorities.

Mediators

The Role of the Mediators is to collect the evidence from each Peer and compare both records for consistency. Alarms are raised when disagreement is discovered. Mediators also release periodic digests to Account Holders and Accounting Authorities. Selected Peers take on the Mediator role for the duration of the session or transaction.

Account Holders

The Account Holders are responsible for the delivery, persistence and retrieval of accounted data. Account Holders are more long-lived than Mediators and Peers are periodically reselected through the issue of a nonce by the Peer for which they persist data.

Accounting Authorities

The Accounting Authority deliver service composition relative data to Account Holders for persistence as well as raising alarms in the case of service composition inconsistencies. Peers operate as Accounting Authorities when they are Mediators and Peers of a service composition scenario.

2.3.2 Protocols

Two protocols are presented, one for accountability of public data and one for private data exchange. The protocols are similar except that in the case of the private accountability the messages are encrypted with a shared secret.

Public Accountability Protocol

1. *A* requests service access from *B*
2. *B* grants access and provides *A* with its public key and a signed contract.
3. *A* acknowledges receipt and acceptance of the service contract, signs it and sends *A* its public key. The set of Mediators is now calculated using a pre-agreed hash function on a combination of *A* and *B*'s Ids, the time stamp and the process ID. The contract and other relevant information is lodged with the mediators.
4. *A* consumes the service and each message sent and received is signed with *A'*s private key, then lodged with the Mediators. *B* also signs each message and sends to the Mediators. The Mediators examine each pair of messages and compare for agreement. In the case of disagreement, both *A* and *B* are notified of the disagreement by the Mediators.
5. When service consumption ceases both *A* and *B* send a final statement to the Accounting Authority. Also, the Mediators send a digest of the accounted data to both *A*'s and *B*'s Account Holders where the data is persisted.

The algorithm as it is written above considers full public accountability. The messages are not encrypted and are open to everyone. It is possible to insert enciphering and deciphering of the messages to ensure more confidentiality. If this is done the Mediators would require the public keys to ensure that the data can be verified semantically. The protocol is described more formally in Table 1 below.

Step	*Flow*	*Message*	*Description*
1	$A \Rightarrow B$	B, T, S	*A* sends a time-stamp and service name to *B*
2	$B \Rightarrow A$	$B_{k_{pub}}, A, Con_S, T, S, P$	*B* sends its public key, signed contract, time-stamp, service name, and processID to *A*.
3	$A \Rightarrow B$	$Con_{ACK}, B, A_{k_{pub}}, S$	*A* acknowledges receipt of the service contract and passes its public key *B.*
4a	$A \Leftrightarrow B$	M_c, A, B, P	*A* and *B* exchange signed messages messages.
4b	$A, B \Rightarrow M$	M_c, A, B, P	*A* and *B* transmit each message to the *Mediators*. The *Mediators* examine for consistency
5a	$A, B \Rightarrow AccA$	R_{final}, A, B, P	*A* and *B* transmit a final report to the *Accounting Authority*. The *Accounting Authority* compares the two statements for consistency
5b	$M \Rightarrow A_{ACC}, B_{ACC}$	M_{digest}, A, B, P	The *Mediators* transmit a digest of the collected data to *A* and *B*'s *Account Holders.*

Table 1: Public Accountability Protocol

Private Accountability Protocol

1. *A* requests service access from *B*
2. *B* grants access and provides *A* with its public key and a signed contract.
3. *A* acknowledges receipt and acceptance of the service contract, signs it and sends A its public key. The set of Mediators is now calculated using a pre-agreed hash function on a combination of *A* and *B*'s Ids, the time stamp and the process ID. Both parties generate a secret using the other's public key and their own private key. The contract and other relevant information is lodged with the Mediators.
4. *A* consumes the service and each message sent and received is encrypted with the secret and signed using *A*′s private key, then lodged with the Mediators. *B* also encrypts and signs each

message and sends to the Mediators. The Mediators examine each pair of messages and compare for agreement without being able to examine the contents of the encrypted message. In the case of disagreement, both *A* and *B* are notified of the disagreement from the Mediators.

5. When service consumption ceases both *A* and *B* send a final statement to the Accounting Authority. Also, the Mediators send a digest of the accounted data to both *A*'s and *B*'s Account Holders where the data is persisted.

Step	*Flow*	*Message*	*Description*
1	$A \Rightarrow B$	B, T, S	*A* sends a time-stamp and service name to *B*
2	$B \Rightarrow A$	$B_{k_{pub}}, A, Con_s, T, S, P$	*B* sends its public key, signed contract, time-stamp, service name, and processID to *A*.
3	$A \Rightarrow B$	$Con_{ACK}, B, A_{k_{pub}}, S$	*A* acknowledges receipt of the service contract and passes its public key *B*. Both parties generate a shared secret.
4a	$A \Leftrightarrow B$	M_c, A, B, P	*A* and *B* exchange messages encrypted with the shared secret for the duration of the transaction
4b	$A, B \Rightarrow M$	M_c, A, B, P	*A* and *B* transmit each message to the *Mediators* encrypted with the shared secret. The *Mediators* cannot interpret the meaning of the messages, just validate its integrity
5a	$A, B \Rightarrow AccA$	R_{final}, A, B, P	*A* and *B* transmit a final report to the *Accounting Authority*. The *Accounting Authority* compares the two statements with other statements for consistency in a service composition
5b	$M \Rightarrow A_{ACC}, B_{ACC}$	M_{digest}, A, B, P	The *Mediators* transmit a digest of the collected data to *A* and *B*'s *Account Holders* where it is persisted

Table 2: Private Accountability Protocol

3 Integration with Trust and Transactions

3.1 The Use of Accountability Data in Evolving Trust

Accountability data is a source of experience for OPAAS Trust Manager overlay network. See deliverable D3.9 for more details on the trust overlay approach. There are several ways in which the data can be used for gathering Experience Reports for trust evaluation.

- Accountability data can be used directly as Experience Reports. Algorithms can be be developed to operate directly on the accounted data and published to the Trust Manager.
- Retrieved data from an Account Holder can be analysed by the Entity and used as an input to pre-published trust algorithms.
- Access to accountable data can be inserted in algorithms and the Trust Manager can run the algorithm against the resultant query.
- Alarms raised by Mediators and Accounting Authorities can also be used for the generation of trust. These alarms are reported in the accounted data.

Other sources of experience reports for trust can also be checked against the accountability data. For example, if a peer's trust is degraded due to false reports, the accountability framework provides a means of the peer disputing the false report with accountable evidence. In this regard, private accountability presents a situation where false reports cannot be defended against without revealing private keys to view the evidence. Algorithms for trust are discussed in the related deliverable D3.9 Final Distributed Identity and Trust Model.

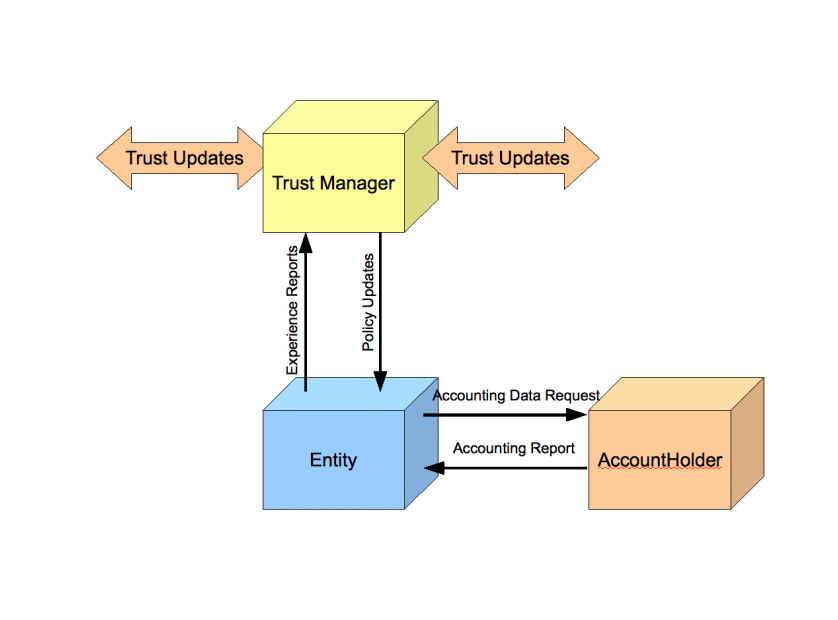

Figure 2: Accounting and Trust

3.2 Integration with Transaction Model

The Accountability model developed is primarily concerned with atomic interactions, where more complex composed services can be modelled as a set of those atomic transactions. The Transaction model developed in WP3 considers the fact that there can be dependencies between services.

In Figure 3 below service s_1 is provided by service provider SP_1, service s_2 by SP_2, etc. By the logic of our Accountability Model this would mean that there is a contract and an accountability evidence trail between the Initiator I and each of the service providers SP_1, SP_2, SP_3, SP_4 and between I and the data providers DP_1 and DP_2.

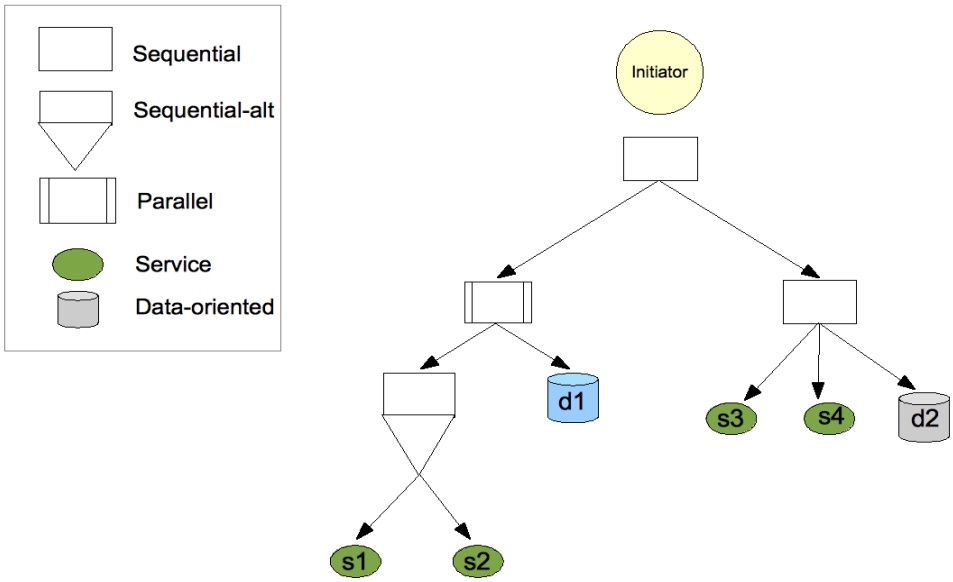

Figure 3: Sample Transaction Tree

In the most simple case this would be sufficient, but very often there may be dependencies between these services in the form of business relationships between the service providers. For example let's suppose that SP_1 has a relationship with SP_3 in the form of a financial incentive. This might be of the form of "SP_3 will provide a 10% discount in the event of s_3 being bundled with s_1". By translating this to our Accountability model, this dependency would be referenced in the contract exchanged between *I* and SP_1. Figure 4 below shows how the initiator agrees contracts with each service provider and how these these contracts can be inter-related via dependencies.

As the initiator is a choreographer and consumer of each of these services (via local co-ordinators), each of these contracts is linked to one set of accountability data. Each of these interactions can be accounted for in whichever way the service consumer has prescribed (i.e. No Accountability, Public Accountability, Private Accountability). In this regard the role of the Accounting Authority of our model can be performed by the actors of the transaction.

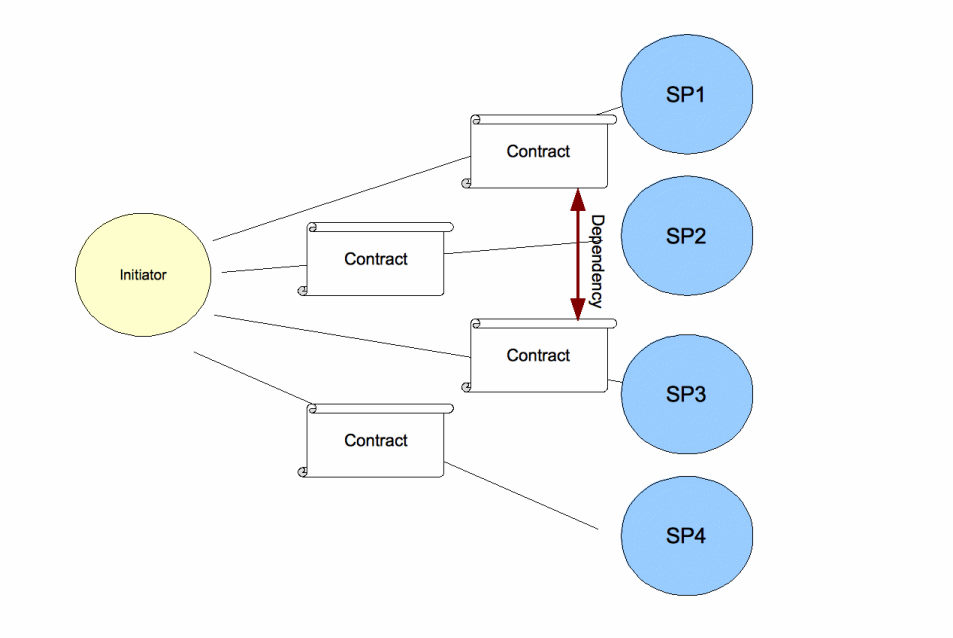

Figure 4: Transaction Initiator and Contracts

The Transaction Management facility can also supply accountability data via the Transaction log to the Account Holders. This data can be used as supporting evidence for accountability data or to evolve trustworthy transactions via the Trust Model.

4 Implementation

For the implementation of the model we provide a data model, a class model of the Accountability Model and a discussion of how the data will be collected and transformed into data that conforms with the data model.

4.1 Data Model

An XML Schema to record the usage data allowing for signing and enciphering of data is shown below in Figure 5. Each instance of a UsageData document contains a UsageHeader and a number of UsageRecords. The UsageHeader contains information about the provider and consumer of the service as well as the serviceID, the processID and the time stamp when the transaction began. Each UsageRecord contains an attribute indicating which protocol is being used, the ID of the initiator (who generated the record), the encrypted messageData, the initiator's signature and a time stamp of when the data was metered.

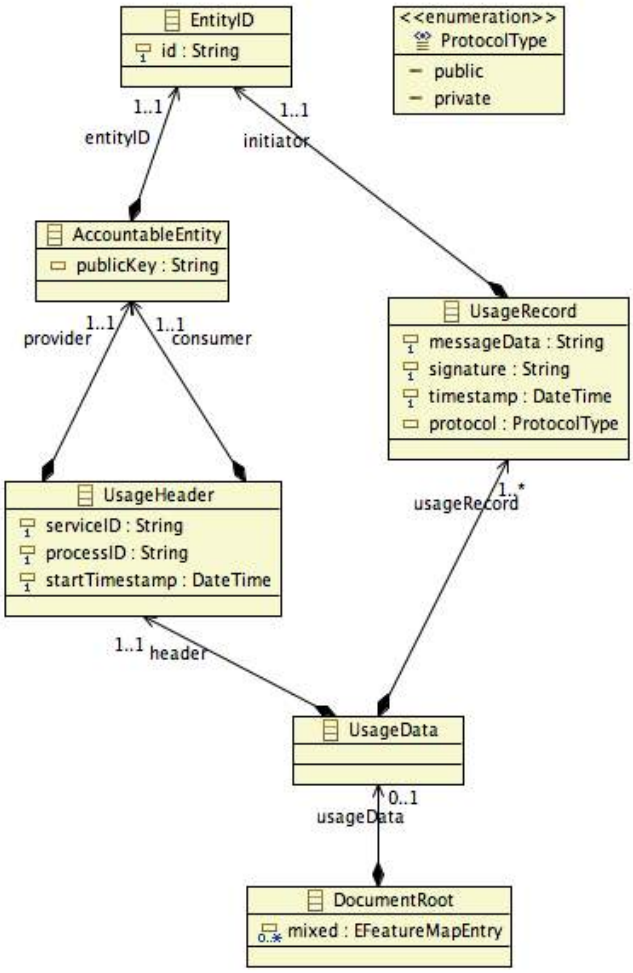

Figure 5: Usage Data Schema

4.2 Data Collection

The collection of metering data is platform specific and is reliant on the types of interceptors or monitors available for the platform in use. In our case we intend to deploy the digital ecosystem on a JXTA platform. For the collection of usage data in JXTA we plan to use the JXTA MMP to preform service monitoring.

4.2.1 JXTA Metering and Monitoring Project

The primary goal of the JXTA Metering and Monitoring project[1] is to provide a dynamic and extendible framework for gathering and reporting metrics about JXTA services running within groups of JXTA PeerGroups. The types of metrics maintained for a service are defined on a peer service implementation with an XML representation of each type of metric. In addition to providing an API for obtaining metrics from PeerGroups running locally, the optional JXTA Peer Information Protocol (PIP)[2] is a specified means for obtaining these metrics from remote peers.

The MonitorManager choreographs all cumulative and asynchronous reporting of metered data. It provides a Service Provider Interface (SPI) view to the underlying Services via ServiceMonitors. The MonitorManager interacts with all registered ServiceMonitors delegating appropriate information and requests to the service-specific ServiceMonitors which will internally optimise and collect their own data.

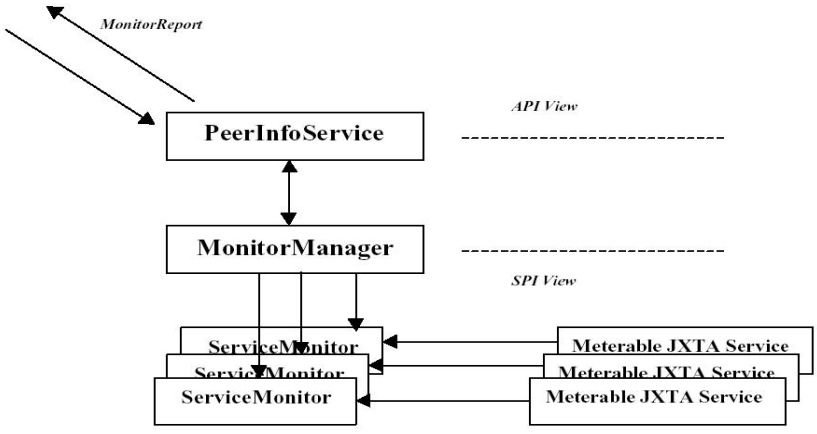

Figure 6: JXTA MMP Architecture

1 *JXTA Metering and Monitoring Project*, http://meter.jxta.org/

2 *JXTA Peer Information Protocol*, http://spec.jxta.org/nonav/v1.0/docbook/JXTAProtocols.html#proto-pip

4.2.2 Usage Data Collection and Transformation

The JXTA MMP project will be leveraged on to access metered data about services running in the JXTA platform. These Monitor Reports are in the form of XML and can be transformed to our usage data model via XSLT transformations. This transformed UsageData is published to appropriate Mediators selected for the service instance. This is the same process regardless of whether the entity is providing or consuming the service. This is shown below in Figure 7.

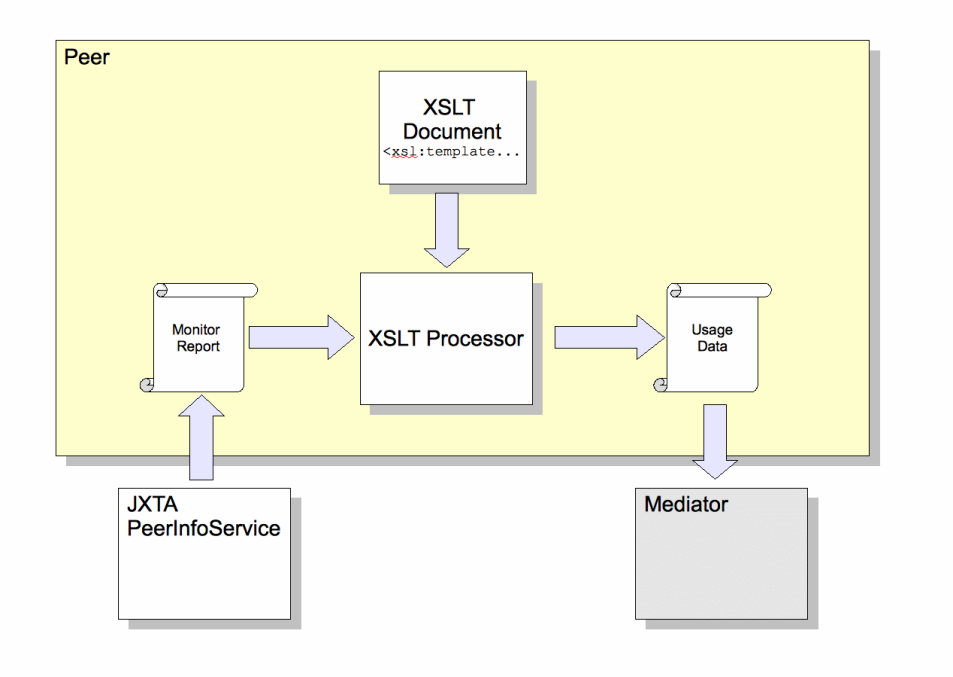

Figure 7: Transforming the MonitorReport to UsageData

4.3 UML Model

A UML class diagram of the accountability actor classes (org.opaals.accountability) and how they relate to the data model and the MonitorReport is shown below in Figure 8. The class AccountableEntity is any entity that should be accountable. The diagram also shows how an AlarmListener class and an Alarm class are related to the model. Alarms are raised when inconsistencies occur in the usage data.

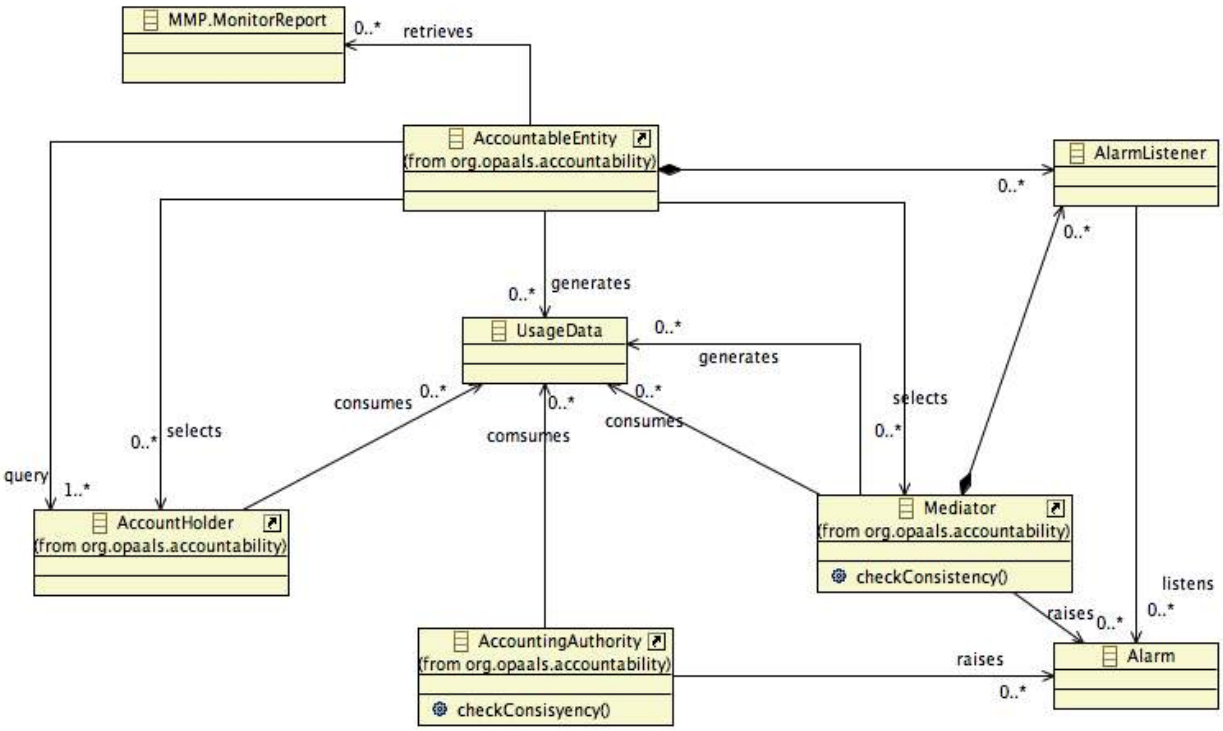

Figure 8: Accountability Model UML Class diagram

5 Conclusion and Next Steps

This document has described the Distributed Accountability Model. The model has been refined since the last iteration and protocols have been provided for public and private accountability. In addition integration points with Trust Model and Transaction Model have been identified. A usage data model and a model diagram of the Actors and their relationships is described in the implementation section.

The next steps for this work are as follows:

- Simulation of this model which will address scalability, peer selection, overhead of encryption in private accountability.
- Implementation of this model and integration with identity and trust and the peer-to-peer platform being developed.
- Close integration with Trust Model implementation through the development of trust algorithms which will take accountability data as an experience report and perform trust updates on trust values per context. See D3.9 for more details on how this can be achieved.
- Integration with Transaction Model implementation including the selection of Accounting Authority members from the transaction model instantiation.

References

[1] Bullock, G., 2006, *Governance, Accountability, and Legitimacy*, Working Paper Series, Consumer Information Laboratory, University of California, Berkeley, available at nature.berkeley.edu/infolab/files/u3/InfoLab_WP06-01_Governance.pdf

[2] Schedler, A., *Conceptualizing Accountability* , The Self-Restraining State: Power and Accountability in New Democracies, pp13-28, 1999, Lynne Reiner Pulishers.

[3] OPAALS Consortium, *Distributed Accountability model for an Autopoietic P2P network*, 2007, http://files.opaals.org/OPAALS/Year_1_Deliverables/WP04/D4.2.pdf

[4] Viégas, F. B., 2005, *Bloggers' expectations of privacy and accountability: An initial survey. Journal of Computer-Mediated Communication*, Journal of Computer-Mediated Communication, at http://jcmc.indiana.edu/vol10/issue3/viegas.html , .

[5] Brin D., *The Transparent Society: Will Technology Force Us to Choose Between Privacy and Freedom?*, 1999, Perseus Books.

[6] Winn, Peter, A., 2004, *Online Court Records: Balancing Judicial Accountability and Privacy in an Age of Electronic Information*, Washington Law Review, at http://projects.ischool.washington.edu/lawsymposium/docs/winnp.pdf , University of Washington School of Law.

[7] Hausheer, D., Stiller B., *NETWORKING 2005* , NETWORKING 2005, pp40,52, 2005, Springer Berlin / Heidelberg.

[8] Zhang, Y., Lin, K.,, 2007, *Hierarchical Management of Service Accountability in Service Oriented Architectures*, in Proceedings of IEEE International Conference on Service-Oriented Computing and Applications, IEEE Press

[9] Diffie, W., Hellman, M., 1976, *New directions in cryptography*, IEEE Transactions on Information Theory.

Appendix A – Usage Data XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.opaals.org/schema/usagedata"
    xmlns:tns="http://www.opaals.org/schema/usagedata"
    elementFormDefault="qualified">
    <xsd:element name="usagaeData">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="header" type="tns:usageHeaderType" minOccurs="1" maxOccurs="1"/>
                <xsd:element name="usageRecord" type="tns:usageRecordType" minOccurs="1"
maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:complexType name="usageHeaderType">
        <xsd:sequence>
            <xsd:element name="consumer" type="tns:accountableEntityType" minOccurs="1"
maxOccurs="1"></xsd:element>
            <xsd:element name="provider" type="tns:accountableEntityType" minOccurs="1"
maxOccurs="1"></xsd:element>
            <xsd:element name="serviceID" type="xsd:string" minOccurs="1"
maxOccurs="1"></xsd:element>
            <xsd:element name="processID" type="xsd:string" minOccurs="1"
maxOccurs="1"></xsd:element>
            <xsd:element name="startTimestamp" type="xsd:dateTime" minOccurs="1"
maxOccurs="1"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="entityIDType">
        <xsd:sequence>
            <xsd:element name="id" type="xsd:string" minOccurs="1" maxOccurs="1"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="accountableEntityType">
        <xsd:sequence>
            <xsd:element name="entityID" type="tns:entityIDType"></xsd:element>
            <xsd:element name="publicKey" type="xsd:string"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="usageRecordType">
        <xsd:sequence>
            <xsd:element name="initiator" type="tns:entityIDType"/>
            <xsd:element name="messageData" type="xsd:string"/>
            <xsd:element name="signature" type="xsd:string"/>
            <xsd:element name="timestamp" type="xsd:dateTime"/>
        </xsd:sequence>
        <xsd:attribute name="protocol">
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:enumeration value="public"/>
                    <xsd:enumeration value="private"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:complexType>
</xsd:schema>
```